Improving our Mechanistic Understanding of Cell Cycle Dynamics

Paul F Lang

Wolfson College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

Hilary 2022

To my parents
Sabine and Reinhold

»I stand at the seashore, alone, and start to think. There are the rushing waves ... mountains of molecules, each stupidly minding its own business .... trillions apart ... yet forming white surf in unison.

Ages on ages ... before any eyes could see ... year after year ... thunderously pounding the shore as now. For whom, for what? ... on a dead planet, with no life to entertain.

Never at rest ... tortured by energy ... wasted prodigiously by the sun ... poured into space. A mite makes the sea roar.

Deep in the sea, all molecules repeat the patterns of one another till complex new ones are formed. They make others like themselves ... and a new dance starts.

Growing in size and complexity ... living things, masses of atoms, DNA, protein ... dancing a pattern ever more intricate.

Out of the cradle onto the dry land ... here it is standing ... atoms with consciousness ... matter with curiosity.

Stands at the sea ... wonders at wondering ... I ... a universe of atoms ... an atom in the universe.«

Richard Feynman – The Value of Science [1]

Acknowledgements

Personal

First, I wish to express my gratitude to my supervisors **Prof Bela Novak** and **Prof Jonathan Karr**. Bela's close guidance helped me to get my research off the ground. His trust in my abilities gave me the scientific freedom to shape the project into its current form. Jonathan gave me the opportunity to intern at his lab, where he introduced me to software engineering routines that facilitate my day-to-day work. Our conversations were and continue to be highly influential on my thoughts on science and modeling.

Second, I would like to thank my collaborators for enabling my highly interdisciplinary doctoral project. I greatly appreciate **Prof Julio Banga's** interest in my parameter optimisation problem. It is very unlikely I would have obtained a satisfactory solution without his expertise and support. Big thanks also go to **Dr David Rodriguez Penas** for adapting the optimisation algorithm to meet the demands of my problem and relentlessly running optimisations until convergence. Likewise, my gratitude goes to **Prof Chris Bakal** and **Dr Lucas Dent** for their interest in acquiring experimental data needed for parameter optimisation. I enjoyed and greatly appreciate the close interaction with Lucas in planning the experiments.

Further, I would especially like to thank **Dr Stefan Heldt** and **Dr John Sekar**. It was Stefan, who implanted the idea of reconstructing pseudo-time courses of cell cycle regulators from single cell data in my head; and it was John who introduced me to rule-based model descriptions. Without their inputs this thesis would look very different.

This work also benefitted from the experience of **Joel Luethi** with image analysis and the support of **Prof Lucas Pelkmans**, who generously provided access to image analysis software and hardware. My house mates **Andras**, **Hanifi** and **Oana** pointed me towards CRISPR interference for perturbation of cell cycle regulators, and my former colleagues, **Tobias**, **Leo** and **Hyojin**, as well as **Mo** helped with designing the CRISPRi plasmids. **Prof James Faeder**, **Prof Bill Hlavacek**, **Dr Daniel Weindl** and **Dr Fabian Fröhlich** provided quick advice and support with software issues. **Tobias John** encouraged me to write this thesis based on the Latex template created by himself, **Keith Gillow**, **Sam Evans** and **John McManigle**. Sincere appreciation therefore goes to everyone listed above for their valuable contributions.

I would also like to thank all my friends who accompanied this journey on and off the bike, shared my pain during chain gang and a pizza in the pub afterwards. You made this chapter of my life to the fantastic experience it has been.

Last but not least, I would like to thank my parents **Sabine** and **Reinhold** for their unconditional lifelong support of my academic and non-academic endeavours and my siblings **Annegret** and **Simon** for the close connection between us.

Institutional

I would like to acknowledge funding from the **University of Oxford** and the **EPSRC & BBSRC Centre for Doctoral Training in Synthetic Biology** (grant EP/L016494/1) that gave me the opportunity to work 4.5 years on a project that fascinates me.

Declaration of Authorship

This report, and the work of which it is a record, were carried out by myself unless otherwise acknowledged.

Paul Lang
Oxford, April 2022

Abstract

The mammalian cell cycle is regulated by a well-studied but complex biochemical reaction system. Computational models provide a particularly systematic and systemic description of the mechanisms governing mammalian cell cycle control. They facilitate a detailed understanding of cell cycle control mechanisms and are in part also able to aggregate this knowledge into full cell cycle models that explain periodic cell cycle oscillations. This dissertation aims at improving on these models along four dimensions: model structure, validation data, validation methodology and model reusability.

Presented is a core model structure of the full cell cycle that qualitatively explains the behaviour of unperturbed and perturbed cells. Using rule-based model descriptions, the core model was conveniently extended by a DNA damage checkpoint and a separation in a nuclear and cytoplasmic compartment. The dissertation conceptualises a methodology for generating highly informative validation data, including cell cycle perturbation with CRISPR interference. More specifically, time courses of 292 features as measured by indirect immunofluorescence imaging experiments are reconstructed from single cell snapshot data using the reCAT algorithm. This data and the cell cycle model are then cast into the PEtab format for specifying parameter estimation problems in biochemical reaction networks. By combining a powerful hybrid global-local optimiser with hierarchical optimisation of parameters, a cell cycle model that explains the validation data is presented. The PEtab specification allows any modeler to reuse the model, the data and/or the optimisation results.

Further experimental conditions, for instance in form of CRISPR interference, are expected to significantly improve parameter identifiability and provide a way for testing the predictive power of the model. Given the central role of the cell cycle in health and disease, such a predictive model may aid in the discovery of new therapeutic targets.

Contents

List of Figures

List of Tables

List of Algorithms

List of Boxes

List of Abbreviations

4i . . . . . Iterative indirect immunofluorescence imaging

AIC . . . . . Akaike information criterion

APC/C . . . . Anaphase promoting complex or cyclosome

BIC . . . . . Bayesian information criterion

BNGL . . . . . BioNetGen Language

Cdk . . . . . Cyclin dependent kinase

CeSS . . . . . Cooperative enhanced scatter search

CRISPR . . . Clustered regularly interspaced short palindromic repeats

CRISPRi . . . Clustered regularly interspaced short palindromic repeats interference

DHC . . . . . Dynamic hill climbing

eSS . . . . . Enhanced scatter search

FBA . . . . . Flux balance analysis

FISPO . . . . . Full Input-State-Parameter Observability

GEARS . . . . Global parameter Estimation with Automated Regularisation via Sampling

gRNA . . . . . guide ribonucleic acid

hESC . . . . . human embryonic stem cells

HPC . . . . . High performance computing

MLE . . . . . Maximum likelihood estimator

MSE . . . . . Mean squared error

ODE . . . . . Ordinary differential equation

RP . . . . . Restriction point

saCeSS . . . . Self-adaptive cooperative enhanced scatter search

SCF . . . . . Skp, Cullin, F-box containing complex

SSA . . . . . Stochastic simulation algorithm

SBML . . . . . Systems Biology Markup Language

XML . . . . . Extensible Markup Language

1 Introduction

»The conviction of pure science must be unshakable.«

Aristotle – Posterior Analytics [2]

Contents

1.1 Motivation

Human and non-human welfare is directly tied to biology through our physiology and environment, and our interventions with these systems. Hence, if used responsibly, a better understanding of biological systems will translate to more human and non-human welfare through medical, technological, agricultural and environmental applications. Such a systems level understanding of biology requires the aggregation and interconnection of knowledge scattered across large numbers of publications, which is then published in review papers and textbooks. However, such textual and graphical formats are read by humans whose brains struggle to simultaneously read large amounts of heavily interlinked information into memory and perform rigorous consistency checks on them. Representing the same knowledge in a machine readable format allows to run more rigorous, albeit perhaps less flexible, consistency checks on the information. In particular, we often want to understand how systemic behaviour emerges from the biochemical reactions within a cell. Such knowledge can be aggregated and interconnected in form of dynamic mathematical models. Therefore, a mathematical model can be seen as a simulatable scientific review of certain biological information, that allows more rigorous consistency checks in form of **model verification** and **model validation**. It can also be seen as a hypothesis that can be tested with statistical methods. Verified and validated mechanistic mathematical models represent a systematic and systemic understanding of biochemical dynamics that is key to enable rational interventions with biological systems.

Box 1.1: Model verification

Model verification is a process that ensures that the implementation of the model is in agreement with the conceptual ideas, assumptions and specifications underlying the model. In other words, model verification ensures that the implementation of the model does not contain errors.

Box 1.2: Model validation

Model validation tests if the model accurately represents the real system. Model validation approaches range from subjective assessments to objective statistical tests.

1.2 Context

Developing models that accurately represent the dynamics of medium- to large-scale biochemical reaction networks is a challenging task. Larger models require more data for model validation, and the validation is only objective if systematic approaches are used. To achieve satisfactory results, models may need to go through several iterations of refinements, which are potentially performed by different research teams. Therefore, models and data should also be available in a reusable form. Over the last decades, a substantial body of research has focused on developing mathematical models that represent the dynamics of the cell cycle as a central part of behaviour in proliferating cells. These efforts have been reviewed in depth by Abroudi *et al.* [3, 4]. Here, I want to describe four recent models of the full mammalian cell cycle and one particularly sophisticated model of the full budding yeast cell cycle. Table 1.1 compares these models with respect to four dimensions: model structure, validation data, validation methodology and model reusability.

Singhania *et al.* [5] developed a cell cycle model consisting of ordinary differential equations for cyclins, and a binary sequence of activities for other cell cycle regulators as determined in a Boolean model by Li *et al.* [6]. They manually adjusted the

parameters to fit flow cytometry data of cyclin A, cyclin B and DNA. The Singhania model is very simplistic, which allows for finding analytical solutions of its ordinary differential equations (ODEs). In stark contrast, Gauthier and Pohl [7] developed the most fine-grained mammalian cell cycle model to date. While most cell cycle models describe only one protein per gene plus phosphorylated forms and protein complexes, their model accounts for about a dozen of RNA and protein species per gene. Despite the complexity of the model, Gauthier and Pohl managed to reproduce the oscillatory behaviour of the cell cycle. However, they did not use experimental time course data to validate their model. Unfortunately, the model is specified in form of ODEs rather than reactions. While a biochemical description in form of reactions would be agnostic to the simulation framework, the mathematical description in form of ODEs restricts the simulation method to ODE solvers. In addition, extending ODE models is more cumbersome and error prone, as the reaction-to-ODE conversion step has to be done manually in this setting. Weis *et al.* [8] on the other hand used flow cytometry data to manually fit and extended their cell cycle model, which combines and extends the cell cycle models of Conradie *et al.* [9] and Csikasz-Nagy *et al.* [10]. However, the Weis model requires resetting some G1 variables to starting conditions to produce stable cycles. Abroudi *et al.* [11] updated the Iwamoto model [12] and used the model to study DNA damage response. This model has also been used to develop methods for model abstraction [13, 14]. Yet, the model only produces dampened oscillations and was never validated with experimental time course data. Unfortunately, none of these four models were originally made available in a machine readable form. Thankfully, however, Ashley Xavier converted the Weis and Abroudi models into the Systems Biology Markup Language (SBML) and submitted the files to the *BioModels* repository [15].

The sophisticated yeast cell cycle model combines many of the desirable features that are partially present in the mammalian models discussed above. Originally developed by Tyson, Novak and coworkers, the model has been refined for almost two decades [10, 16–19]. Most recently, Mitra *et al.* developed the Biological Property Specification Language to exploit qualitative data in parameter optimisation procedures [20] and automated the parameterisation of this model [21]. They also deposited model and data on *GitHub*, to facilitate reproducibility and reusability. Nevertheless, even this well developed model has several limitations. For instance, the model behaviour does not purely emerge from biochemical reactions. More specifically, the model makes use of conditional expressions (so-called *events*) to describe biological behaviour that results from a set of potentially unknown biochemical reactions. The model parameters were estimated by fitting to mRNA expression levels, despite the model species describe protein dynamics. Using mRNA concentration as a proxy for protein concentration is especially problematic in the context of cell cycle regulation, where protein concentration is not only controlled by synthesis from mRNA, but also by targeted degradation. Moreover, the authors did not generate ensembles of candidate models and select the most appropriate one with statistically motivated model selection criteria. Finally, the molecular species represented in the model lack unambiguous machine-interpretable annotations. Taken together, while present models provide several important insights into the fundamental principles of cell cycle control, much work is still needed to accurately represent the dynamics of protein abundance, especially in a mammalian system.

Table 1.1 | Features of selected cell cycle models.

	Model structure					
Model	Organisms	Checkpoints	Compartments	Species	Reactions	Parameters
Singhania et al. (2011)[5]	Mammals	None	CE	4 + 6[1]	16	19
Gauthier et Pohl (2011)[7]	Mammals	None	CP, NU	428	-	332
Weis (2014)[8]	Mammals	None	CE	25	73	136
Abroudi et al. (2017)[11]	Mammals	DNA damage	CE	66	138	150
Mitra et al. (2019)[21]	Yeast	-	CE	44	39	153

continued ...

... continued

	Validation Data			Validation methodology		
Model	Observables	Time points	Conditions	Objective function	Optimiser	Selection criterion
Singhania et al. (2011)	2	>10000	1	-	manual	None
Gauthier et Pohl (2011)	0[2]	1	-	-	manual	None
Weis (2014)	3	~ 40	3	-	manual	None
Abroudi et al. (2017)	0	0	0	-	-	None
Mitra et al. (2019)	10[3]	17-24[4]	1[5]	Least squares[6]	PyBioNetFit[7]	None

continued ...

... continued

	Model reusability				
Model	Representation	Format	Annotation	Availability	Remarks
Singhania et al. (2011)	ODE/predetermined logical sequence	-	-	-	-
Gauthier et Pohl (2011)	ODEs	Word/ Matlab	None	Supplementary material	[a]
Weis (2014)	Reactions	SBML[8]	None	BioModels	[b]
Abroudi et al. (2017)	Reactions	SBML[8,9]	None	BioModels	[c]
Mitra et al. (2019)	Reactions, Events	SBML	None	GitHub	-

CE: cell, CP: cytoplasm, NU: nucleus.
[1] 4 ODE states, 6 Boolean states.
[2] The model was calibrated to match summary statistics about the cell and the cell cycle.
[3] mRNA instead of protein observables.
[4] Exact number differed between cell cycle synchronisation method.
[5] One condition with time course data plus 119 knock out conditions with qualitative data.
[6] Penalties for qualitative constraint violations were added to the least squares objective.
[7] Using a scatter search algorithm with a simplex local optimizer.
[8] Conversion to SBML performed by Ashley Xavier from the European Bioinformatics Institute.
[9] Simulating the SBML model with zero DNA damage does not reproduce publication results.
[a] Very detailed resolution (e.g. 12 mRNA and protein species per gene).
[b] Oscillations are not stable. Simulation of SBML model crashes after 10.8397 time units.
[c] Based on Iwamoto (2011) [12]. Oscillations are not stable.

1.3 Scope and objectives

Motivated by the belief that mechanistic mathematical models are a particularly pristine representation of scientific knowledge, this dissertation aims at improving our mechanistic understanding of mammalian cell cycle dynamics by addressing the shortcomings of current models. The models in this work shall describe the dynamics of cell cycle regulator abundance. Making the simplifying assumption that all molecular species are well mixed within a compartment, spatial dimensions can be ignored. The system boundaries of the cell cycle regulation network will be chosen subjectively with interoperability with adjacent pathways in mind. Within this scope, the main shortcomings of current models directly translate into the objectives of this thesis, which fall into four categories:

- **Model structure:** The model structure shall allow the description of the dynamics of cell cycle regulator abundance and compartmental localisation. It shall allow emergence of stable oscillations under unperturbed conditions and support the simulation of several cell cycle perturbations. These perturbations include limited growth factor availability and DNA damage.

- **Validation data:** The model shall be validated on time course data for protein and phosphoprotein abundance and localisation.

- **Validation methodology:** To increase objectivity, reproducibility and predictive power, model validation shall be automated and adhere to best practices, unless constrained by current technological limitations.

- **Model reusability:** inspired by the spirit of sustainable and collaborative software development, the model shall be represented in a form that facilitates extensions and improvements by other researchers.

The approach to the first three objectives will be described in the respective Chapters 3 to 5. To meet the last point, the model, data, and optimisation problem will be specified in software independent community standards. The model species will be described with a clear nomenclature and the models will be constructed using version control.

In addition to working on these main objectives, significant efforts went into side projects aiming at developing software tools for annotating, optimising and working with models of biochemical reaction networks. These side projects were conducted during an internship in the group of Prof Jonathan Karr at the Icahn School of Medicine at Mount Sinai, New York (USA), and during the COVID19 pandemic to bridge delays of experimental data acquisition. More specifically, contributions were made to BpForms [22], a toolkit to unambiguously describe the primary structure of modified biopolymers, SBML2Julia [23], a Python interface between PEtab [24] and Julia for Mathematical Programming [25], and SBMLToolkit.jl, an importer of SBML models into the scientific machine learning ecosystem of the Julia programming language (available under https://github.com/SciML/SBMLToolkit.jl). Abstracts of these side projects can be found in Appendix A.

1.4 Organisation of this thesis

1.4.1 Structure

The presentation is divided into six chapters. Every chapter starts with a table of contents. Chapter 2 provides an overview of model representation concepts, relevant experimental methods to study cell cycle dynamics and approaches to solving mathematical optimisation problems. Readers who are familiar with these methods can skip Chapter 2. Chapters 3 to 6 describe the results of my doctoral

research. Chapter 3 covers the construction of computational models that describe the structure of the mammalian cell cycle network in various degrees of depth and breadth. The generation and analysis of experimental data will be presented in Chapter 4. The work towards Chapter 4 was performed in collaboration with Professor Chris Bakal and Dr Lucas Dent from the Institute of Cancer Research, London (UK), and Dr Tobias Strittmatter from ETH Zurich (Switzerland). How the kinetic parameters of the models from Chapter 3 are fit to experimental data from Chapter 4 will be discussed in Chapter 5. The research described in Chapter 5 was conducted in collaboration with Prof Julio Banga and Dr David Rodriguez Penas from the Spanish National Research Council in Vigo (Spain), and Dr Daniel Weindl from the Helmholtz Centre Munich (Germany). All contributions will be clearly indicated in the respective chapters. Finally, Chapter 6 will summarise the findings, discuss their significance and limitations, and provide an outlook for future work and potential applications.

Much of the presented research consisted of spreadsheets and machine readable model and optimisation problem specifications. This data is available online under the following links[1]:

Chapter 3

GitHub[2]: https://github.com/paulflang/cell_cycle_model

GitFront: https://gitfront.io/r/user-5513095/FLByZr3eHbhR/cell-cycle-model/

[1]The GitHub links redirect to the most recent version of the repository. The GitFront links redirect to the state of the repository at the time of submission, i.e. 22 April 2022.

[2]Please email your GitHub username to the author to gain access to private repositories or use the GitFront link.

Chapter 4

GitHub[3]: https://github.com/paulflang/cell_cycle_time_course

GH-Pages: https://paulflang.github.io/cell_cycle_time_course/

Chapter 5

GitHub: https://github.com/paulflang/cell_cycle_petab

GitFront: https://gitfront.io/r/user-5513095/60e3c45e3e1375bd98733c6f8a430439310a9c42/cell-cycle-petab/

Hyperlinks to relevant documents will also be provided throughout the text.

1.4.2 A word on notation

This thesis contains mathematical equations in multiple places. The notation follows the convention that vectors are represented by bold lowercase letters $\boldsymbol{v}$, and matrices by bold uppercase letters $\boldsymbol{M}$. Variables are printed in italics and operators in upright font ($\sin x$). $\log_b$ denotes the logarithm to the base b, and $\log$ the natural logarithm. In computer code it is good practice to give variables self-explanatory multi-letter names (*myvariable*). Similarly in biochemistry it is common practice to use multi-letter variables to describe complexes of several molecules. This thesis will therefore also use multi-letter variables, which can be distinguished from products of single-letter variables by the context or explicit clarification in the text.

1.4.3 Special text elements

To facilitate readability, this thesis makes use of two special text elements: *Boxes* and *Algorithms*. Boxes describe **specialised terms** in more detail. These specialised terms are printed in bold font on first occurrence in the main text.

[3]To view the interactive HTML figures hosted in this repository, please download the file and open it in a browser or use the GH-Pages link.

Box 1.3: Specialised term

Readers who are familiar with the terminology can ignore the descriptions in infoboxes.

Algorithms are written in pseudocode to describe computational procedures more concisely. They are referenced in the text. For instance, Algorithm 1.1 describes the multiplication of two whole numbers using only summation. Lists of infoboxes and algorithms, as well as figures, tables and relevant abbreviations can be found at the beginning of this thesis.

Algorithm 1.1 Whole number multiplication

Input: a, b

Output: $res = a \cdot b$

```
res ← a      ▷ The variable on the left gets the value of the variable on the right.
while b > 1 do
    res ← res + res
    b ← b − 1
end while
```

2

Methodological Background

»I was induced to seek some other method which would comprise the advantages ... and be exempt from their defects. ... I believed that the ... following would prove perfectly sufficient ... .«

Rene Descartes – Discourse on the Method[26]

Contents

2.1 Model representation

To understand the physical world, humans build mechanistic models in their mind with or without the help of mathematics and computational methods. These mechanistic models provide insight into the inner workings of the studied system down to a certain level of abstraction. This lowest level of abstraction defines the granularity of the model and is often approximated with pure phenomenological descriptions. How these abstractions are made is arbitrary, for instance the abstractions through consecutive layers of a neural network model do not necessarily align with the abstractions of a handcrafted model describing the same phenomenon. Yet, both models may still have the same predictive power. Nevertheless, the abstractions provide nested frames of reference to reason about the model, and some frames of reference might be more useful than others. For instance, some forms of human behaviour are better described with psychology, others with neuroscience; some mathematical operations are more conveniently performed in polar coordinates, and others in Cartesian coordinates. Well-chosen frames of reference may therefore increase the utility of models. These benefits go beyond potential increases in predictive power for inter- and especially extrapolations by adding structure and thus bias to the model. They include better interpretability, thus enabling rational interventions; better extensibility and modifiability, thus enabling progress of knowledge/science; and having a defined language/set of concepts, thus enabling communication of the model. In the context of biological modeling, models need to be communicated between humans and machines. This section will outline typical frames of reference or representations that are commonly used for modelling biochemical reaction networks. The models developed in this thesis will be described

with mathematical representations in form of ordinary differential equations (ODEs) and biochemical representation in form of reactions or rules.

2.1.1 Mathematical representation

Traditionally, models of biochemical reaction networks are cast into mathematical equations. One of the most simple models are Boolean models described by binary variables and logical functions. More fine-grained models are often described by ODEs, which treat the variables (incl. time) as continuous. ODEs only contain one independent variable (e.g. time). An ODE is called nonlinear if the dependent variable appears in a nonlinear expression and linear otherwise. It is called homogeneous if it does not include constant terms and inhomogeneous otherwise. Homogeneous ODEs always include $x(t) = 0$ as trivial solution. The highest derivative defines the order of the ODE. Chemical reaction networks are described by first order ODEs that are frequently nonlinear and inhomogeneous. Consider for instance a system of dependent variables x_1 and x_2, where x_1 catalyses its own synthesis, x_2 converts x_1 into x_2 and is synthesized and decayed at a constant rate. Using mass action kinetic laws, these four reactions can be described by the ODE system

$$\frac{dx_1(t)}{dt} = k_{SyX1} \cdot x_1(t) - k_{Co} \cdot x_1(t) \cdot x_2(t), \qquad x_1(0) = 1 \tag{2.1}$$

$$\frac{dx_2(t)}{dt} = k_{SyX2} + k_{Co} \cdot x_1(t) \cdot x_2(t) - k_{De} \cdot x_2(t), \quad x_2(0) = 0, \tag{2.2}$$

where $k_{SyX1}, k_{SyX2}, k_{Co}, k_{De} > 0$ are kinetic rate constants. Setting $\frac{d\boldsymbol{x}(t)}{dt}$ to zero, we can calculate the steady states of the system. The system can be linearised around a steady state through differentiation by $\boldsymbol{x}$ to study its stability properties. This yields the Jacobian matrix $\frac{d\frac{d\boldsymbol{x}}{dt}}{d\boldsymbol{x}}$ of the system. The solutions of this linearised system are described by a weighted sum of exponentials, with the product of time and the

eigenvalues of the Jacobian as exponent. Therefore, we can use the eigenvalues of the Jacobian to determine if the steady state is stable (all real parts < 0) or unstable (any real part > 0; saddle point if some real parts < 0, some real parts > 0 and all imaginary parts $= 0$). If the eigenvalues contain imaginary parts, the systems spirals into (real parts < 0) or out of (real parts > 0, system may approach a limit cycle trajectory) the steady state.

If the rate of change of different variables happens at different time scales, the dimensionality of the system can be reduced by assuming a quasi-steady state for quickly adapting variables. This is done by setting their time derivatives to zero, so that their current value is an algebraic function of the other variables. The resulting system of equations is called differential algebraic equation system.

In matrix notation, the system can also be written as

$$\frac{d\boldsymbol{x}(t)}{dt} = \underbrace{\begin{pmatrix} 1 & -1 & 0 & 0 \\ 0 & 1 & 1 & -1 \end{pmatrix}}_{\boldsymbol{S}} \cdot \underbrace{\begin{pmatrix} k_{SyX1} \cdot x_1(t) \\ k_{Co} \cdot x_1(t) \cdot x_2(t) \\ k_{SyX2} \\ k_{De} \end{pmatrix}}_{\boldsymbol{v}}, \tag{2.3}$$

with stoichiometry matrix $\boldsymbol{S}$ and flux vector $\boldsymbol{v}$. In this notation, it can be readily observed that in steady state, i.e. when $\frac{d\boldsymbol{x}}{dt} = 0$, the fluxes $\boldsymbol{v}$ must be in the nullspace of $\boldsymbol{S}$. This is used for flux balance analysis (FBA) to analyse the flow through metabolic reactions.

FBA and ODE systems are continuous and deterministic, however, molecule numbers are discrete, and collisions and thus biochemical reaction networks are stochastic. To account for the stochasticity, they can be described by the chemical master equation and simulated with stochastic simulation algorithms (SSA).

The mathematical representations can be combined to hybrid simulations such as ODE/Boolean [5] and ODE/SSA [27]. Dynamic FBA [28] alternates through

steps of FBA and ODE simulations, where the FBA step sets the parameters for the ODE step, and the ODE step sets the flux boundaries for the FBA step. SSA-FBA [29] follows the same idea as dynamic FBA, except that it uses stochastic simulations instead of ODEs. Regulatory FBA [30] and integrated FBA [31] add a Boolean layer of gene regulation on top of (dynamic) FBA to inform metabolic enzyme abundance, which determines which reactions have non-zero flux. To a certain degree, such hybrid techniques can help to integrate biologically distinct processes into a single model.

2.1.2 Biochemical representation

2.1.2.1 Reaction-based representation

While mathematical representations can offer direct insight into several characteristics of a model, they have at least three disadvantages. First, the representation already constrains the simulation, i.e. an ODE model has to be simulated with an ODE integrator, and a stochastic model with a stochastic simulation algorithm. Conversion is not always possible, as the identification of kinetic laws from equations may be ambiguous, unless stoichiometry matrix and flux vector are explicitly known. Second, while a chemical reaction network always specifies an ODE system (and if all reactions are mass action kinetics also a chemical master equation), the inverse is not true. This leaves, quite literally, a lot of space for errors to be made when representing models with mathematical equations. Third, casting a chemical reaction network into equations is more conveniently and reliably done by modelling software than by hand. The chemical reaction network underlying

Equation (2.3) can be expressed as

$$x_1 \rightarrow x_1 + x_1, \quad k_{SyX1} \cdot x_1 \tag{2.4}$$

$$x_1 + x_2 \rightarrow x_2 + x_2, \quad k_{Co} \cdot x_1 \cdot x_2 \tag{2.5}$$

$$\emptyset \rightarrow x_2, \quad k_{SyX2} \tag{2.6}$$

$$x_2 \rightarrow \emptyset, \quad k_{De} \cdot x_2. \tag{2.7}$$

Using software like COPASI (GUI) [32], Tellurium (Python) [33], SBToolbox (Matlab) [34] and Catalyst.jl (Julia) [35, 36], allows automatic conversion of reaction networks to simulatable ODE or stochastic models. Similarly, COBRApy (Python) [37] and COBREXA.jl (Julia) [38] enable constrained based modelling and FBA of metabolic networks.

2.1.2.2 Rule-based representation

Reaction-based representations require explicit enumeration of all species and reactions. However, in signalling networks, proteins can be post-translationally modified at multiple sites and form complexes with multiple binding partners. This can lead to a combinatorial explosion of the number of species, and thus reactions. Rule-based models exploit the circumstance that many reactions are not affected by the exact configuration of several modification sites or binding partners, and collapse them all into a single reaction rule with a single rate law[1]. Rather than using arbitrary identifiers to refer to molecular species, rule-based models use a more structured representation. For instance the rule-based modelling language BioNetGen [40] considers *molecules* as indivisible entities. Molecules can

[1]The constraint of having a shared rate law is lifted in energy-based modeling, an extension to rule-based modeling, which also satisfies thermodynamic constraints arising from reaction loops by construction. The interested reader is pointed to Chapter 2 of Justin S. Hogg's dissertation [39].

have functional *components* such as sites for epigenetic/post-transcriptional/post-translational modifications or for association with other molecules. Components may have an internal *state*, such as a phosphorylation, or be *stateless*. The blueprint of a molecule with all its components is called *molecule type* and has to be defined before simulation. For instance, `A(B, C~D~E)`, defines a molecule type `A` with a stateless component `B` and a component `C`, that can either be in state `D`, or `E`; i.e. possible states of a component are preceded by a tilde symbol. Molecules can form complexes through *bonds*. For instance, `A(B!1, C~D~E).F(G!1)` specifies a complex between `A` and `F`. The exclamation mark indicates a component that is involved in a bond. Components on different molecules that are bonded together have to carry the same identifier symbol (here `1`). Like in reaction-based models, the term *species* is used to refer to a unique configuration of one molecule or complex. Omission indicates any configuration. For instance, `F()` represents any species that contains the molecule `F`, whereas `F(G)` represents the free molecule `F` only. There are two special identifiers that can succeed the exclamation mark: `"+"` represents a bond to any molecule and `"?"` a bond to any or no molecule. This syntax enables the BioNetGen language (BNGL) to specify reaction rules that expand to multiple concrete reactions per rule via pattern matching. A *pattern* is a partially specified complex or molecule and a reaction rule is a set of reactant patterns with corresponding transformations applied to those patterns. The following transformations are allowed and can be combined in reaction rules: [41]:

- Forming and breaking a bond, e.g. `A(B) + F(G) <-> A(B!x).F(G!x)`
- Changing the state of a component, e.g. `A(C~D) -> A(C~E)`
- Creating and destroying molecules and complexes, e.g. `0 -> F(G)`

The abstraction of multiple reactions in a single reaction rule prevents human errors of not considering all combinatorial possibilities of species a certain chemical transformation could act on. Automatic expansion to reaction networks allows ODE and stochastic simulations. However, BNGL also allows specification of infinite-sized reaction networks through endless polymerisation. Such systems can be simulated using network-free simulations. Like exact stochastic methods, they consider species as colliding particles and produce equivalent results, but they do produce and destroy species on the fly, rather than considering the whole network at any time. For reaction systems with more than several hundred species, network-free simulations can be faster than conventional stochastic simulation [42]. Software that supports rule-based modelling includes RuleBender [43] and PySB [44].

2.1.3 Community standards

Exchanging biological models between different research groups and software interfaces, and storing them beyond the lifetime of the software to create them, requires machine (and human) readable community standards and formats. The most common standards are CellML [45] and the Systems Biology Markup Language (SBML) [46]. Both are based on the Extensible Markup Language (XML). CellML takes a more mathematical view and is frequently used for physiological models, while SBML takes a more biochemical view. The biochemical description in core SBML naturally supports ODE and stochastic simulation. Boolean models [47], FBA [48] and rule-based models [49] are supported by packages that add additional features to the core layer.

2.2 Experimental methods to study the cell cycle

Mathematical as well as mental models are typically incomplete or inconsistent. These gaps and inconsistencies can be turned into research questions and hypotheses that can be tested experimentally. Ultimately, the experimental evaluation of the hypotheses improves the model by reducing the number of inconsistencies and gaps. On theoretical grounds, it has been shown that complex, consistent formal models can never be complete [50], which also has consequences on the physical world [51]. On practical grounds, the gaps in biochemical models can typically be filled by designing and learning from the right experiments. For biochemical models that contain kinetic rate constants, a typical research question is to find the right value for the kinetic parameters. To this end optimal experimental design algorithms can help to identify a set of experiments that will reduce the uncertainty in parameter values the most [52, 53]. While an experiment may or may not include a perturbation of the observed system, it always includes a measurement. Often the raw measurement needs to be converted into a suitable form to evaluate the hypothesis in question. While experimental observations can be of many modalities, in the following section we will focus on methods that help us to identify kinetic parameters by generating time courses of the abundance of cell cycle regulators in perturbed an unperturbed conditions. Chapter 4 will discuss why iterative indirect immunofluorescence imaging (4i) [54] was chosen to measure the abundance of cell cycle regulators. It will further describe how time courses were reconstructed from 4i snapshot measurements with the reCAT algorithm [55].

2.2.1 Quantifying abundance of cell cycle regulators

As cell cycle regulation mostly depends on proteins, this section will discuss three popular methods for protein quantification: live cell imaging of fluorescently tagged proteins, mass spectrometry and immunofluorescence. These methods differ in their

- detection limit,
- ability to multiplex,
- ability to detect posttranslational protein modifications,
- spatial resolution,
- and destructiveness.

Before deciding on the most appropriate method for addressing the problem at hand, all these features need to be considered carefully.

2.2.1.1 Live cell imaging of fluorescently tagged proteins

Live cell imaging for protein quantification typically makes use of fluorescent protein tags. Fluorescent proteins were first discovered in the jellyfish *Aequorea victoria* [56]. While the natural protein emitted green light, fluorescent proteins have been engineered to emit various wave lengths across and beyond the visible spectrum. They have also been improved to increase brightness and half-life [57]. Fluorescent proteins can be fused to the protein of interest by fusing their corresponding genes together. This instructs the cell to express the protein of interest with a fluorescent tag. The fluorescence signal can then be detected in widefield or confocal fluorescence microscopy. However, care must be taken that the protein fusion does not interfere with the functionality of the original protein.

Due to spectral overlap between fluorophores and the labour intensive procedure of developing cell lines with fluorescent fusion proteins, live cell imaging can only detect a small number of proteins simultaneously. Moreover, live cell imaging does also not allow to detect posttranslational modifications, with few notable exceptions where fluorescent protein-based phosphorylation sensors have been developed [58, 59]. However, as a microscopy technique the spatial resolution of live cell imaging is physically only limited by the diffraction limit of visible light, with super-resolution microscopy techniques being able to 'break' this limit under certain conditions. One big advantage of live cell imaging is that, as the name suggests, the measurement does not kill the cells. This allows researchers to compare one and the same cell across multiple time points.

2.2.1.2 Mass spectrometry

Mass spectrometry is an experimental technique where molecules in the sample are ionized and separated by their mass-to-charge ratio. This technique can also be applied to protein extracts of cell populations, and more recently also single cells [60]. While there are multiple different ways to perform protein mass spectrometry experiments, they all need to (a) identify proteins and (b) measure their abundance. Identification typically follows via chromatographically separating digested protein extracts, ionizing the eluted peptides and separating the ions by their mass-to-charge ratio. To increase the specificity of identification of proteins in a complex sample, an additional mass spectrometry step is needed. To this end, the most frequent peptide ions from the previous step are selected for fragmentation in a collision chamber one after another. There, they collide with chemically inert gas molecules. The resulting fragmentation products are again separated by their mass-to-charge ratio in another mass spectrometer step and hit a detector that measures the mass spectrum (a

histogram with mass-to-charge ratio on the X-axis and intensity on the Y-axis). While it is possible to infer the peptide sequence from the fragmentation products that hit the detector, it is usually sufficient for protein identification to match the list of fragmentation product masses with protein database entries. As many factors can influence the intensity of a mass spectrum peak, mass spectrometry is not an inherently quantitative technique. Nevertheless, it is possible to compare the area under intensity peaks for one and the same protein across samples.

Mass spectrometers can simultaneously quantify thousands of proteins and detect posttranslational modifications. However, they typically cannot resolve subcellular localisation of the proteins. Mass spectrometry also is a destructive technique, which does not allow to observe identical cells or cell populations over time. Nevertheless, pseudo-time courses of cell cycle regulators can be obtained by separating cells into different samples that coarsely represent different cell cycle stages before quantifying the abundance of cell cycle regulators with mass spectrometry. This separation can be done by classifying cells based on their size [61] or their expression of certain cell cycle markers [62, 63]. Alternatively, cells can also be arrested in different cell cycle stages [64].

2.2.1.3 Immunofluorescence and 4i

Immunofluorescence is an experimental method that exploits the specificity of antibodies for binding their target protein. By conjugating a fluorescent dye to the antibody, its target can be detected under a fluorescence microscope. To visualize intracellular targets, cells need to be treated with a fixation agent that terminates all cellular processes and preserves the sample. After permeabilisation of cellular membranes the antibodies can then enter the cell and bind to their targets. In indirect immunofluorescence these antibodies are called primary antibodies and

are not conjugated to a fluorophore. Instead, after binding of primary antibodies to their targets, so-called secondary fluorophore conjugated antibodies against the constant regions of the primary antibodies are used to detect binding of the primary antibodies. As multiple copies of a secondary antibody can bind to a single copy of a primary antibody the indirect immunofluorescence approach amplifies signal strength compared to direct immunofluorescence. Yet, similar to live cell imaging the number of simultaneously detectable fluorophores is limited by the spectral overlap of different fluorescent dyes. However, the sample fixation theoretically allows iterative application of antibodies against different targets, if antibodies can be efficiently eluted without deteriorating the sample. This efficient elution is compounded by photoinduced crosslinking of antibodies to their targets. To avoid this photoinduced crosslinking, Gut *et al.* [54] have developed imaging and elution buffer solutions that allow over 20 iterations of antibody staining, imaging and elution. By using two primary antibodies with different constant regions per iteration, this iterative indirect immunofluorescence imaging method (Figure 2.1) can detect over 40 protein targets with very little sample deterioration.

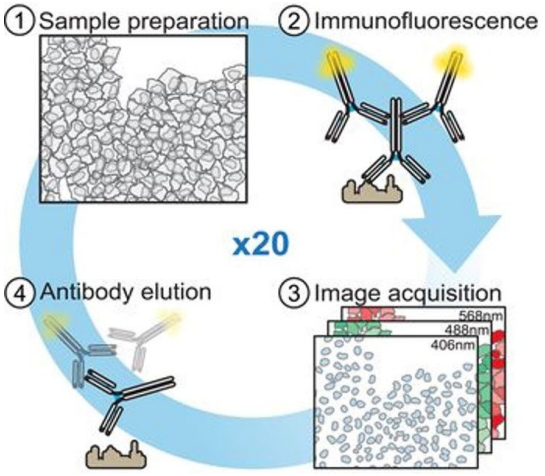


Figure 2.1 | Iterative indirect immunofluorescence imaging (4i). 4i iterates through four main steps. (1) Sample preparation: cells or tissues are fixated and permeabilised to allow diffusion of antibodies into cells and compartments. (2) Immunofluorescence: primary and secondary antibodies are applied to detect the target. By using primary antibodies with different constant domain, it is possible to detect multiple targets per iteration. (3) Image acquisition. (4) Antibody elution. The procedure can be repeated over 20 times without substantial sample deterioration or residual antibody signal. Figure from Gut *et al.* [54]. Reprinted with permission from AAAS.

Indirect immunofluorescence methods are able to detect very low abundant proteins. Posttranslational protein modifications can be detected by using antibodies that only bind the modified target. Immunofluorescence imaging is subject to the same constraints in resolution as live cell imaging. However, immunofluorescence requires fixation of cells, thus providing only snapshot rather than time-course measurements. Nevertheless, immunofluorescence imaging can quantify protein abundance in individual single cells, which allows for identifying cell trajectories and pseudo-time courses.

2.2.1.4 Generating time-course data from snapshot measurements

Single cell measurements like single cell RNA sequencing, flow cytometry, mass cytometry, single cell mass spectrometry and immunofluorescence imaging can simultaneously measure multiple state variables, for instance the abundances of macromolecules. Every cell can therefore be represented as a point in a multidimensional state space. Dimensionality reduction methods like principal component analyses [65, 66] and t-distributed stochastic neighbour embedding [67, 68] allow for representing such cell populations in a lower (often two-) dimensional space. While multiplexed single cell snapshot measurements yield the states of different cells at the same time, when looking at genetically identical cells it makes sense to think of it as looking at the same cell at different times. Assuming that short distances in state space locally correlate with short distances in time, one can reconstruct cellular trajectories through the state space (Figure 2.2). Globally, however short distances in state space may not correlate with short distances along the trajectory. Consider for instance a C-shaped trajectory: the two points at the ends of the trajectory are very close in Euclidian state space despite being the most distant points along the trajectory.

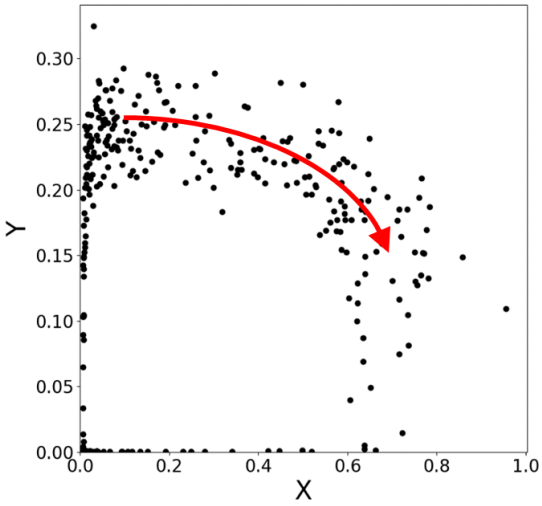


Figure 2.2 | Schematic idea for reconstructing trajectories from multiplexed single cell measurements. Dots represent different cells at the time of measurement of states, X and Y. The red arrow indicates a reconstructed trajectory. If X and Y are oscillating without drift, the trajectory will form a closed loop.

Many approaches can account for this circumstance. Kafri *et al.* [69] reconstruct trajectories by starting at the point with highest cell density. Next, they draw a small circle around this point and move to the point with the highest density on the circumference of the circle. This procedure is repeated until the trajectory is reconstructed. However, this trajectory reconstruction method does not scale well to/cannot efficiently exploit multidimensional data.

This is addressed by the Wanderlust algorithm [70], which builds a **k-nearest-neighbour graph** for the cell population. Representing the cell population as a graph eliminates the notion of spatial dimensions and considering only the k nearest neighbours eliminates long distance relationships between the cells.

Diffusion pseudo-time [71] on the other hand is based on the idea that the location of each cell in state space can be described as a time dependent wave function. The transition probability between two locations is normally distributed with the distance of the states, which puts much higher weight on local than global relationships. At time zero, the wave function of a cell is localised at the cell's position. As time goes to infinity the wave functions of two distinct cells converge to a stationary distribution. Intuitively, the diffusion pseudo-time between two cells corresponds to the distance between the time integrals of their wave functions.

The diffusion pseudo-time algorithm automatically finds root cells and returns diffusion pseudo-time distance from these root cells.

CellCycleTRACER [72] disregards long distance relationships by classifying cells in consecutive cell cycle phases using decision trees on their state variables that initialize **Gaussian mixture models** for further refinement of the classification. It then projects the cells on a linear trajectory of cell cycle pseudo-time that is constructed such that the ordering of cell cycle phases is maximally preserved.

DeepCycle [73] disregards long distance relationships by classifying cells into four cell cycle phases using a convolutional neural network. The penultimate four-dimensional layer of the neuronal network is used for trajectory reconstruction by projecting it onto a two dimensional space, where cells are localised on a circular structure from which the cell cycle trajectory can be recovered. DeepCycle is distinct from all other methods in this section, as it takes raw 2-channel (brightfield and nuclear stain) images as input instead of vectors representing the features of cells. This shows that cell cycle time is accurately encoded in structural features alone and does not require information about the abundance of cell cycle regulators. As part of this theses, we tested the Cycler [74] and reCAT [55] algorithms, which shall be described in more detail in the following sections.

2.2.1.5 Cycler

Cycler [74] was developed to account for cell cycle variation in multiplexed immunofluorescence imaging experiment. It is a version of the Wanderlust algorithm [70] for reconstructing trajectories of cellular differentiation, with several modifications to optimise for reconstructing cell cycle trajectories. The next paragraph will describe the Wanderlust algorithm, followed by paragraphs explaining the rationale of this algorithmic design and its modification in Cycler.

Box 2.1: k-nearest-neighbour graph

A graph is a mathematical structure where vertices are connected via edges. Graphs are often visualised as a set of circles that are connected by lines. A graph is called directed when all edges have a direction and undirected if all edges are undirected. In weighted graphs the edges can differ in their strength. A graph is called fully connected if there is an edge between all possible pairwise combinations of vertices. Mathematically, graphs can be represented with an adjacency matrix $\boldsymbol{A}$ where the $\boldsymbol{A_{ij}}$ element indicates the weight (typically one in unweighted graphs) of the edge from vertex i to j (or vice-versa, depending on the definition) when vertices are connected and zero otherwise. In a k-nearest neighbour graph points are represented as vertices that are connected to vertices that represent its k nearest neighbouring points according to some distance metric. k-nearest-neighbour graphs are directed and their adjacency matrices therefore not generally symmetric, but the directionality is often ignored.

Box 2.2: Gaussian mixture models

Gaussian mixture models try to describe the probability distribution of data as the weighted sum of multiple Gaussian distributions. The parameters of the Gaussian mixture models are the mean, covariance and prior probability (weight) of each Gaussian. These parameters can be estimated with an expectation-maximisation algorithm (Algorithm 2.1). However, expectation-maximisation algorithms only find local minima. In order to find global or better local minima, expectation-maximisation is therefore often initialised with centroids from **k-means clustering**, which is equivalent to a Gaussian mixture model where the covariances are constrained to be diagonal, infinitesimally small and equal for all clusters.

Wanderlust requires two inputs: cells represented as vectors of features like abundances of cell cycle regulators and a user defined start cell that indicates the start point of trajectory reconstruction. This start cell becomes one of n_l so called waypoint cells, with the other n_l-1 waypoint cells chosen at random. Next Wanderlust obtains a weighted k-nearest-neighbour graph of the whole cell

Algorithm 2.1 Expectation-maximisation for Gaussian mixture models

Input: The number of Gaussians k

Input: n datapoints $\boldsymbol{x}_j$

Input: $\boldsymbol{\mu}_{i_0}, \boldsymbol{\Sigma}_{i_0}, \boldsymbol{\pi}_{i_0} \quad \forall \quad i \in 1, \ldots, k$

repeat

▷ Expectation step

$\gamma_{ij} \leftarrow \frac{\pi_i f(\boldsymbol{x}_j|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)}{\sum_{l=1}^{k} \pi_l f(\boldsymbol{x}_j|\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)}$ ▷ f denotes a normal distribution

▷ Maximisation step

$\pi_i \leftarrow \frac{1}{n} \sum_{j=1}^{n} \gamma_{ij}$

$\boldsymbol{\mu}_i \leftarrow \frac{\sum_{j=1}^{n} \gamma_{ij} \boldsymbol{x}_j}{\sum_{j=1}^{n} \gamma_{ij}}$

$\boldsymbol{\Sigma}_i \leftarrow \frac{\sum_{j=1}^{n} \gamma_{ij} (\boldsymbol{x}_j - \boldsymbol{\mu}_j)(\boldsymbol{x}_j - \boldsymbol{\mu}_j)^T}{\sum_{j=1}^{n} \gamma_{ij}}$

until Convergence

Box 2.3: k-means clustering

k-means clustering is an unsupervised learning method to categorise datapoints in multidimensional space into k clusters, such that the sum of within cluster variances is minimised. While the problem is computationally difficult, simple heuristics quickly converge to a local minimum. After the user provides the number of clusters k, k-means clustering can select k points at random as initial centroids for the k clusters. Next, it assigns each datapoint to the cluster with the closest centroid. Then the centroid is updated for each cluster and the procedure repeats until convergence. Note that the clustering can depend on the initialisation of the heuristic and better solutions may be found by running the heuristic from different choices of initial cluster centroids.

population using Euclidian distances between cells. By randomly selecting l<k edges for each cell, Wanderlust creates an ensemble of n_g l-out-of-k-nearest-neighbour graphs. For every l-out-of-k-nearest-neighbour-graph the shortest distance from each waypoint cell to each cell X is constructed. The waypoint-based distance from the start cell is then the distance of the waypoint cell from the start cell plus the distance of cell X from the waypoint cell (where “plus” is replaced with “minus” if and only if the distance of cell X from the start cell is smaller than the distance

of the waypoint cell from the start cell). The overall trajectory position is the weighted average of waypoint-based distances, with higher weight on waypoints that are close to cell X. Note that waypoint cells are cells themselves and thus subject to this averaging so that their trajectory position is not equal to their plain distance from the start cell. Therefore, Wanderlust iterates through realignment steps where waypoint-based distances are calculated using waypoint trajectory positions rather than plain distances from the start cell. Once the ordering of trajectory positions of each waypoint cell has converged in each l-out-of-k-nearest-neighbour-graph in the ensemble, the cell cycle pseudo-time for each cell is calculated by taking the mean over all n_g graph specific trajectory positions. It may seem peculiar why Wanderlust introduces waypoint cells rather than directly using plain distances from the start cell, and why it generates l-out-of-k-nearest-neighbour graphs rather than using the k-nearest-neighbour-graphs directly. The rationale behind choosing waypoint cells is that plain distances become noisy with increasing distances from the start cell. Introducing waypoint cells provides anchors that improve the local ordering of cells that are far away from the start. The rationale behind creating an ensemble of l-out-of-k nearest neighbour graphs is that with increasing number of cells, it becomes increasingly likely that outlier cells short circuit the k-nearest-neighbour graph. While an outlier cell does not have any close neighbours, its nearest neighbour may be close to the start of the trajectory, and its second nearest close to the end of the trajectory, if the trajectory is curved in Euclidian state space. By choosing l much smaller than k, Wanderlust eliminates short circuit edges from most l-out-of-k-nearest-neighbour graphs.

There are three particular modifications Cycler makes to optimise the Wanderlust algorithm for reconstructing cell cycle trajectories from non-iterative immunofluorescence imaging data. First, Cycler does not select waypoints at random. As most cells

in an asynchronously dividing cell population are in G1 phase, most waypoints would be close to the start cell. To improve local ordering in all cell cycle phases, waypoints are chosen such that they are uniformly distributed across the whole cell cycle using initial plain distances from the start cell as proxy. Second, trajectory reconstruction is more sensitive to noise and outliers in the low-dimensional setting of non-iterative immunofluorescence imaging. To reduce the risk of choosing an outlier as waypoint, waypoints were refined with a median filter. To this end the median of the k nearest neighbours of a waypoint was calculated and the cell closest to this median was labelled as waypoint instead of the original waypoint cell. Last, for calculation of waypoint-based distances for each cell, Cycler uses Gaussian weights to calculate the contribution of each waypoint, whereas Wanderlust uses a linear weighting scheme.

2.2.1.6 reCAT

Wanderlust and Cycler scale well to large cell populations and high dimensional data. However, neither of them can reconstruct trajectories of cells along a closed loop. Therefore, a full cell cycle trajectory can only be reconstructed by merging partial cell cycle trajectory reconstructions, which are generated by excluding cells in a small window of cell cycle stage from the analysis. This caveat is addressed in reCAT [55], which was originally developed to reconstruct trajectories from single cell RNA sequencing data. Like Cycler, reCAT takes feature vectors of cells as input. After logarithmic transformation to reduce the dominance of excessively high expression levels, reCAT divides the cell population into several clusters using a Gaussian mixture model. reCAT then reconstructs the cell cycle trajectory by finding the shortest circular path to visit all mean values of the clusters. This problem is well known as travelling salesman problem and computationally expensive. To nevertheless find a good solution within a reasonable time, reCAT uses the so-called

consensus-TSP heuristic. While reCAT can find the start point and direction of the circular trajectory for the dataset used in the original publication, this does not generalise well to other data sources. Therefore, the user has to manually select the start point and direction of the trajectory. In direct comparison with Cycler, reCAT has the advantage of constructing full cell cycle trajectories, but does not scale well with the number of cells.

2.2.2 Perturbation experiments

Time courses provide rich information for estimating the parameters of dynamic systems. However, certain parameters may be unidentifiable. One cause for unidentifiabilities are system variables (e.g. protein isoforms) that do not only correlate in their downstream effects, but also their expression pattern over time. In other words, they perform the same action at the same time. Therefore, the ratio between the kinetic constants of the (equivalent) downstream reactions they participate in cannot be identified either. For example, consider the two interchangeable species X_1 and X_2, which degrade an observed protein P. Assuming mass action kinetics, the overall degradation velocity v_{De} for P is then described by $v_{De} = k_1 \cdot X_1 + k_2 \cdot X_2$, where k_1 and k_2 are unknown degradation rate constants and all other variables are observed. As there are two unknowns and, due to the correlation of expression over time, just one equation, there is no unique solution for k_1 and k_2. In such cases, one can decorrelate their expression across perturbation experiments. Typical approaches are the use of small molecule inhibitors or activators, small interfering RNAs, and more recently the Clustered regularly interspaced short palindromic repeats (CRISPR) gene editing system [75]. The key components of CRISPR gene editing are the Cas9 endonuclease and a guide RNA (gRNA) sequence. After forming a complex with the gRNA, the Cas9 endonuclease is guided

to its target via base-paring of the gRNA and cleaves both DNA strands. The double strand break is then repaired by the host cell via homology directed repair or non-homologous end joining. Homology directed repair in the presence of donor sequences, flanked by homologous regions, copies the donor sequence into the cut site. Non-homologous end joining introduces deletions or insertions into the cleavage site, which often leads to frameshift mutations that knock out the target gene. Alternatively, knockdowns may be performed with dead Cas9 (dCas9) enzymes, i.e. enzymes with mutated active site. Fusing a Krüppel associated box (KRAB) transcriptional repression domain to the dCas9 protein and selecting an appropriate gRNA sequence allows transcriptional repression of target proteins. This process is referred to as CRISPR interference (CRISPRi) [76]. Using transcriptional activation domains instead of KRAB domains enables gene activation (CRISPRa) [77].

2.3 Parameter optimisation

Dynamic models of biochemical reaction systems describe what reactions occur and how fast these reactions occur. The former can be described by a stoichiometry matrix, the latter by rate laws. These rate laws contain model variables (representing chemical species) and constant parameters. However, these parameters are frequently unknown. One way to find their value is to perform test tube experiments with the corresponding reaction in isolation. However, this scales poorly to a larger number of reactions and recreating the conditions of an intracellular environment in a test tube can be challenging. Alternatively, one can measure features like time courses of chemical species and fit the model parameters such that they optimise agreement with the experimental data. This section will focus on such parameter optimisation methods. We will first formally describe optimisation problems before

discussing the challenges for optimising the parameters of biochemical reaction network models. Finally, we will present the methods and concepts discussed in Chapter 5 to overcome these challenges.

2.3.1 Optimisation problem formulation

In this section, we will briefly formalise generic optimisation problems, before motivating a least squares objective function and concretising the optimisation problems resulting from fitting ODE models to experimental time course measurements. In general, optimisation problems can be written in the form

$$\begin{aligned} \min_{\boldsymbol{\theta}} \quad & J(\boldsymbol{\theta}) \\ \text{s.t.} \quad & u_i(\boldsymbol{\theta}) \leq 0, \quad i = 1, \ldots, m \\ & v_j(\boldsymbol{\theta}) = 0, \quad j = 1, \ldots, p \end{aligned} \tag{2.8}$$

where

- $J \colon \mathbb{R}^n \to \mathbb{R}$ is called objective function,
- $\boldsymbol{\theta}$ is a vector of decision variables,
- $u_i(\boldsymbol{\theta}) \leq 0$ are inequality constraints, and
- $v_j(\boldsymbol{\theta}) = 0$ are equality constraints.

Inequality constraints are frequently used to restrict the parameter search to a finite window in parameter space. Equality constraints can be used to encode scientific laws and mechanistic interactions, such as those described by ODE models. Based on the domain of J, we can distinguish discrete, continuous and mixed optimisation problems. If J or the constraints contain exclusively linear expressions, we speak of linear optimisation problems, otherwise we speak of non-linear optimisation

Box 2.4: Convexity

A function is called convex if a line segment between any two points on the graph of the function never lies below the function graph.
A subset of Euclidian space is called convex, if a line segment between any two points in the subset is fully contained within the subset.

problems. If both, J and the feasible region described by the constraints are **convex**, we speak of convex optimisation problems.

The choice of the objective function depends on the problem at hand. In classical utilitarian ethics the objective is to maximise welfare, for many economical problems the objective is to reduce costs and for modelling physical phenomena the objective is typically to find the model parameters $\boldsymbol{\theta}$ with the highest probability for explaining the data $\bar{\boldsymbol{y}}$. That is, the goal is to maximise

$$P(\boldsymbol{\theta}|\bar{\boldsymbol{y}}) = \frac{L(\bar{\boldsymbol{y}}|\boldsymbol{\theta}) \cdot P(\boldsymbol{\theta})}{P(\bar{\boldsymbol{y}})}, \tag{2.9}$$

where

- $P(\boldsymbol{\theta}|\bar{\boldsymbol{y}})$ is called posterior probability,
- $L(\bar{\boldsymbol{y}}|\boldsymbol{\theta})$ the likelihood function (to be distinguished from probabilities in that likelihoods do not necessarily sum/integrate to 1),
- $P(\boldsymbol{\theta})$ the prior or marginal probability of the parameters, and
- $P(\bar{\boldsymbol{y}})$ the marginal probability of the data.

As the data is a given input that does not change, we can ignore $P(\bar{\boldsymbol{y}})$. Assuming that we have no prior knowledge about the parameters, we can also ignore the prior and Equation (2.9) simplifies to

$$P(\boldsymbol{\theta}|\bar{\boldsymbol{y}}) = L(\bar{\boldsymbol{y}}|\boldsymbol{\theta}), \tag{2.10}$$

For Gaussian noise the likelihood of a single data point is

$$L(\bar{y}_i|\boldsymbol{\theta}) = \frac{1}{\sqrt{\sigma_i^2 2\pi}} e^{-(\bar{y}_i - y_i(\boldsymbol{\theta}))^2/2\sigma_i^2}, \tag{2.11}$$

where $y(\boldsymbol{\theta})$ describes the simulation results, and datapoints are indexed with i.

Therefore, the likelihood of n independently distributed data points is

$$L(\bar{\boldsymbol{y}}|\boldsymbol{\theta}) = \prod_{i=1}^{n} L(\bar{y}_i|\boldsymbol{\theta}). \tag{2.12}$$

Maximising the likelihood yields the maximum likelihood estimator (MLE). However, the evaluation of the likelihood function can be numerically unstable due to its large number of multiplicative terms [78]. Taking the logarithm to circumvent this problem, we obtain the log-likelihood

$$\log L(\bar{\boldsymbol{y}}|\boldsymbol{\theta}) = -\frac{1}{2}\sum_{i=1}^{n}\left(\frac{\bar{y}_i - y_i(\boldsymbol{\theta})}{\sigma_i}\right)^2 - \log 2\pi\sigma_i^2. \tag{2.13}$$

As the logarithm is a strictly monotonously increasing function, the log-likelihood and the likelihood have the same optima. Ignoring the constant terms $\log 2\pi\sigma_i^2$ and multiplying with minus two yields the weighted least squares objective function

$$J(\boldsymbol{\theta}) = \sum_{i=1}^{n}\left(\frac{\bar{y}_i - y_i(\boldsymbol{\theta})}{\sigma_i}\right)^2, \tag{2.14}$$

where the denominator can be ignored to obtain the least squares objective function if σ_i is identical for all data points. In Section B.1.2 we will see how the Gauss-Newton method can exploit this least squares structure to calculate approximate Hessians. In Section 2.3.4 we will generalise the above equations to a situation where we can make a prior guess about the parameters $\boldsymbol{\theta}$.

In biological experiments the readout does often not correspond directly to the concentration or count of molecular species $\boldsymbol{x}$. To calculate the theoretical readout $\boldsymbol{y}$ we therefore need to formulate an observation function

$$\boldsymbol{y}(\boldsymbol{\theta}) = h(\boldsymbol{x}(\boldsymbol{\theta}), \boldsymbol{\theta}), \tag{2.15}$$

which may for instance contain parameters for offset and scaling.

In ODE problems the amount of species is calculated from the initial value problem

$$\begin{aligned} \dot{\boldsymbol{x}}(t, \boldsymbol{\theta}) &= f(\boldsymbol{x}(t, \boldsymbol{\theta}), \boldsymbol{\theta}), \\ \boldsymbol{x}(t_0, \boldsymbol{\theta}) &= \boldsymbol{x_0}(\boldsymbol{\theta}). \end{aligned} \tag{2.16}$$

We can distinguish parameters $\boldsymbol{\theta}$ that appear in differential equations, initial conditions and the observation function by referring to them as kinetic parameters, initial parameters and observation parameters, respectively.

2.3.2 Challenges of optimising biological models

Parameter fitting problems of biochemical reaction networks frequently result in objective functions with the following undesirable properties.

- Unknown parameter ranges: The time scales for biological reactions such as gene expression, protein degradation, complex dissociation and enzyme catalysis can differ by multiple orders of magnitude. Even the time scales of reactions of the same type can differ by several orders of magnitude. For instance, the search engine Datanator [79] reports a median catalytic rate of $5.7 \times 10^5\,\mathrm{s}^{-1}$ for human catalase and a median catalytic rate of $3.6 \times 10^1\,\mathrm{s}^{-1}$ for human alcohol dehydrogenase.

- High dimensionality: Biochemical reaction networks can easily include dozens of reactions and therefore dozens of potentially distinct kinetic parameters [80]. This results in a very large parameter search space, as its volume increases exponentially with the number of parameters.

- Unknown parameters in the observation function and noise model: While the physical relationship between a chemical species and the measured observable

may suggest the form of an observation function, its parameters were often not determined experimentally. Similarly, physical considerations and practical experience may suggest the form of a noise model (e.g. multiplicative normally distributed noise). However, its parameters (e.g. standard deviation) may not have been determined experimentally. Therefore, the parameters of the observation function and noise model enter the objective function and must be estimated, too. This is particularly problematic, as adding even a small number of observation function parameters to an objective function can result in substantial drops of optimiser performance [81] for poorly understood reasons. In Section 2.3.3.2, we will discuss hierarchical optimization [82] as an approach to overcome this issue.

- Multiple local optima: In general an optimisation problem for ordinary differential equations can contain multiple local minima. In particular, objective functions arising from systems with oscillatory dynamics contain many local minima [83]. Such optimisation function landscapes require the use of global methods, which can often avoid getting stuck in local minima. This comes at the cost of optimisation speed, as local optimisation methods typically converge faster. Even more problematic than slow convergence is the circumstance that there is no guarantee that the found optimum is indeed a global optimum. Confirming a solution as the global optimum would require some form of global information about the objective function or exhaustive search of the parameter space. However, global information is rarely available for ODE problems, and the parameters $\boldsymbol{\theta}$ are generally continuous, which prevents exhaustive search [84].

- Local optima of identical objective function value: Symmetries in the model specification can result in multiple local optima with identical objective function value. This implies that no single optimal solution can be found, i.e. the parameters are globally unidentifiable. However, if the number of local optima is finite, the parameters are locally identifiable.

- Discontinuity of the objective function: Biochemical reaction networks may contain bistable switches. A bifurcation plot shows the discontinuity of steady state concentrations as a function of a parameter. A dynamical system that can approach alternate steady states depending on its parameters will therefore display discontinuities in its corresponding objective function. As many local optimisation methods rely on differentiability, they will not perform well in regions with discontinuities.

- The global optimum lies in an infinitely extended flat valley: a global optimum in an infinitely extended flat valley of an objective function necessarily and sufficiently implies that the model parameters cannot be identified. We distinguish two kinds of such valleys: perfectly flat valleys that extend infinitely, and relatively flat valleys that extend infinitely without exceeding a finite objective function value [57]. The former are equivalent to structural parameter unidentifiabilities, meaning that the parameter optimisation problem has infinite optimal solutions (is ill-posed). Structural unidentifiabilities can only be resolved by changing the structure of the model or by obtaining new and different experimental data. The latter are equivalent to practical unidentifiabilities, which result from noisy or insufficient data. Practical unidentifiabilities can be resolved by obtaining more or better measurements of the same kind.

2.3.3 Optimisation algorithms

Optimisation algorithms attempt to find minima in an objective function. Based on whether they try to find global or local optima, they can be classified as global and local optimisation algorithms, respectively. However, this distinction is not always clear-cut [78]. For example, some global optimisation algorithms are also used for local optimisation when the gradients of the objective function are not known. The following will describe global-local hybrid optimisation as used in Chapter 5. For a more general discussion of global and local optimisation algorithms please refer to the Extended Methodological Background, Section B.1.

2.3.3.1 Hybrid global-local search

Hybrid global-local search tries to combine the advantages of global and local search. These hybrid methods use global search to explore the parameter space for good starting points for local searches. This increases the convergence rate compared to pure global search while still allowing escape from local minima. Empirically, **enhanced scatter search** (**eSS**) [85] has proven to be successful for optimising the parameters of dynamic biological models [86]. ESS (Figure 2.3) is a population-based evolutionary optimisation method consisting of three modules: an initialiser, a global search cycle and a local search cycle. All three modules act on a population of n solutions called RefSet that is iteratively improved in global and local search cycles. The RefSet is initialised with an initialisation method that generates a large set of solutions. The objective function is evaluated for all solutions and a small number of top-ranked solutions is chosen for the RefSet. Additionally, several solutions are chosen for the RefSet at random, to increase the diversity. Until a termination criterion is met, eSS then iterates through global exploration and local intensification cycles. In the global cycle, the *subset generation method* creates

$\binom{n}{2}$ SubSets of two solutions. The *solution combination method* draws two equally sized hyper-rectangles. Their centres correspond to the solutions. Their edges are aligned with the coordinate system. The midpoint of the line segment that connects the two solutions represents one corner of each hyper-rectangle. These hyper-rectangles are then slightly shifted towards the better solution and candidate solutions are randomly sampled within each hyper-rectangle. If a child outperforms its parent the *improvement method* creates a new hyper-rectangle: one corner is the child and one corner is the child plus the parent-to-child vector. This procedure continues with adaptive hyper-rectangle sizes until a child no longer outperforms its immediate parent, resulting in an improved set of candidate solutions. The *update method* then replaces every RefSet member that was outperformed by its direct or indirect offspring, leading to an improved RefSet. After a defined number of global cycles, the *local refinement method* is invoked to drive RefSet members into their corresponding local minimum.

The key hyperparameters for eSS are the number of RefSet members, the number of iterations in the global cycle before a local cycle is invoked, and the local balance, which determines the tendency for putting more emphasis on quality or diversity when selecting starting points for local searches. These hyperparameters allow to set a balance between intensification and diversification of the search. Optimisation efficiency can therefore be increased by selecting appropriate hyperparameters. Additional speedups can be achieved by parallelising eSS. In principle, eSS allows for parallelisation on two levels. Fine-grained parallelisation executes objective function evaluations for different solution vectors in an eSS process simultaneously. Coarse-grained parallelisation runs multiple eSS processes simultaneously and allows them to exchange their results in regular time intervals. **Cooperative enhanced Scatter Search** (**CeSS**) tries to combine possible speedups through appropriate

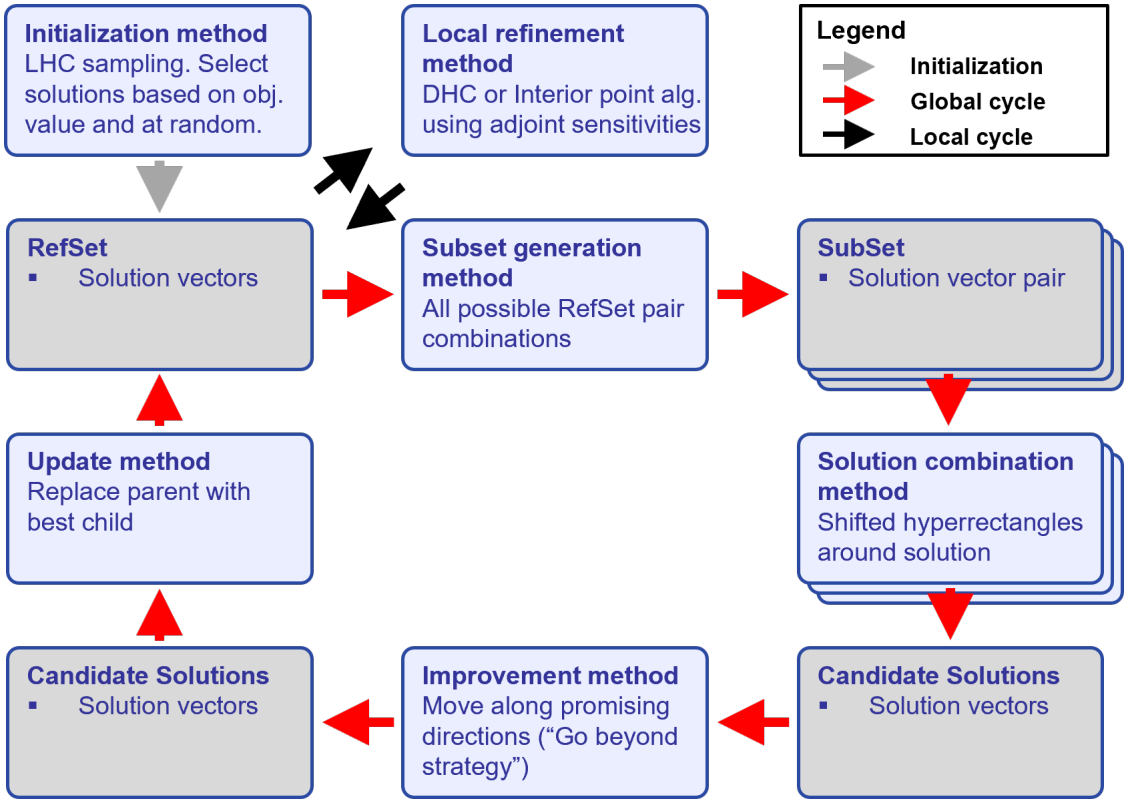


Figure 2.3 | Enhanced scatter search. Scatter search algorithms iteratively improve a population of solutions called *RefSet*. They consist of methods for initialisation, subset generation, solution combination, improvement/refinement and updating (blue boxes). The operations carried out by these methods depend on the exact implementation of scatter search. This figure describes enhanced scatter search devised by Egea *et al.* [85] and improved and implemented by Villaverde *et al.* [87]. LHC: latin hypercube. DHC: dynamic hill climbing.

.

hyperparameter selection and through parallelisation [87]. As finding ideal hyperparameters is difficult *a priori*, CeSS simply uses coarse-grained parallelisation to run threads with different hyperparameter settings simultaneously. Similarly to parallel tempering (see Extended Methodological Background Section B.1.1), the threads of different aggressiveness exchange their solutions in regular time intervals. However, the threads are not adapting their hyperparameters to more promising settings over time.

This successive adaptation of hyperparameters is one of several improvements

implemented in **self-adaptive cooperative enhanced scatter search** (**saCeSS**) [88]. Additionally, saCeSS makes improvements to the cooperation strategy and cooperation timing, combines fine-grained and coarse-grained parallelisation and can exploit high performance computing (HPC) hardware. Current HPC systems connect multiple compute nodes with a message passing interface for inter-node communication. Every single node consists of multiple cores that share a common memory. SaCeSS exploits this infrastructure via coarse-grained parallelisation across nodes and fine-grained parallelisation across cores within nodes. The cooperation between nodes is managed with a controller/responder star architecture, where all responder nodes are only connected via a single controller node. The responder nodes execute eSS threads with differently configured hyperparameters. The controller node transfers solutions between responders and keeps a scoreboard of the responders. This process is not synchronized between responders. Instead, each responder triggers communication with the controller as soon it has finished an eSS iteration. It first checks, whether the best know solution of the controller outperforms its own best solution. If this is the case, this solution replaces the so-called foreign solution (or its direct or indirect offspring) in the RefSet. After the first iteration, there is no foreign solution in the RefSet, so the worst solution is labelled as *foreign*. The reason why the foreign solution instead of the worst solution is replaced, is to avoid premature intensification. If instead the own best solution outperforms the best known solution by the controller, the own best solution is sent to the controller. However, the controller only accepts this solution if it is significantly better than the its presently known best solution. The threshold of what counts as significantly better is halved after the controller has consecutively rejected more solutions than there are responder nodes. The reason why only significantly better solutions are accepted and transferred between nodes is to avoid rapid infiltration of foreign

elite solutions in RefSets of individual eSS processes, as this would lead to loss of RefSet diversity and premature intensification, especially when the number of communicating nodes is high. If a solution is accepted, the score of the responder is updated to $n \cdot t$, where n is the total number of accepted solutions from this responder so far, and t the elapsed time. In this way, the scoreboard reflects how often and how recently a responder found a new best solution. Whenever a responder stops finding new best solutions while other responders do, it sends a reconfiguration request to the controller to become updated with the hyperparameters of the top-scoring responder. More precisely, if $(s_r > 10 \cdot s_s + 20)$ AND $(n_{eval} > 5000 \cdot d)$, where s_r and s_s are the number of received and send solutions since the last sent solution, respectively, n_{eval} is the number of function evaluations since the last sent solution, and d is the number of parameters of the optimisation problem.

2.3.3.2 Hierarchical optimisation

In biological experiments it is often not possible to observe the system state $\boldsymbol{x}$ directly. Rather one observes a readout that is described by an observation function and corrupted by noise $\boldsymbol{\epsilon}$:

$$\bar{\boldsymbol{y}}(\boldsymbol{\theta}) = h(\boldsymbol{x}(\boldsymbol{\theta}), \boldsymbol{\theta}) + \boldsymbol{\epsilon}. \tag{2.17}$$

In this section, we will assume that the measurement noise is normally distributed, i.e. $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{\sigma}^2)$. If the parameters of the observation function are not known, they need to be estimated in the optimisation problem. Unfortunately, this not only increases the dimensionality of the optimisation problem, but also significantly reduces optimizer performance [81] for reasons that are not yet fully understood. Hierarchical optimisation [82, 89, 90] avoids this issue by exploiting the structure of the observation function to analytically calculate (part of) its parameters. Frequently,

the observation function includes a linear scaling parameter $\boldsymbol{s}$ and an offset parameter $\boldsymbol{b}$, which we can pull out of the observation function h to obtain

$$\bar{\boldsymbol{y}}(\boldsymbol{\theta}_r, \boldsymbol{s}, \boldsymbol{b}) = \boldsymbol{s} \cdot \tilde{h}(\boldsymbol{x}(\boldsymbol{\theta}_r), \boldsymbol{\theta_r}) + \boldsymbol{b} + \boldsymbol{\epsilon}, \tag{2.18}$$

where $\boldsymbol{\theta_r}$ denotes the remaining parameters. After accounting for scaling and offset parameters, we will often find that $\tilde{h}(\boldsymbol{x}(\boldsymbol{\theta}_r), \boldsymbol{\theta_r}) = \boldsymbol{x}$. Schmiester et al. [82] show that the global optimum of the standard optimisation problem

$$\min_{\boldsymbol{\theta}} \quad J(\boldsymbol{\theta}) \tag{2.19}$$

is preserved in the hierarchical optimisation problem

$$\min_{\boldsymbol{\theta_r}} \quad J(\boldsymbol{\theta_r}, \boldsymbol{s}(\boldsymbol{\theta}_r), \boldsymbol{b}(\boldsymbol{\theta}_r), \boldsymbol{\sigma}(\boldsymbol{\theta}_r)) \tag{2.20}$$

in which

$$(\boldsymbol{s}(\boldsymbol{\theta}_r), \boldsymbol{b}(\boldsymbol{\theta}_r), \boldsymbol{\sigma}(\boldsymbol{\theta}_r)) = \underset{\boldsymbol{s},\boldsymbol{b},\boldsymbol{\sigma}}{\operatorname{argmin}} \quad J(\boldsymbol{\theta}_r, \boldsymbol{s}, \boldsymbol{b}, \boldsymbol{\sigma}). \tag{2.21}$$

They derive analytical solutions for Equation (2.21) using the necessary condition for a local minimum in $\boldsymbol{s}, \boldsymbol{b}, \boldsymbol{\sigma}$ given $\boldsymbol{\theta}_r$,

$$\nabla_{\boldsymbol{s},\boldsymbol{b},\boldsymbol{\sigma}} J(\boldsymbol{\theta}_r, \boldsymbol{s}, \boldsymbol{b}, \boldsymbol{\sigma}) = \boldsymbol{0}. \tag{2.22}$$

Assuming that datapoints that share scaling parameters also share offset parameters and vice versa, and that datapoints that share scaling/offset parameters also share noise parameters, they arrive at

$$s_\alpha(\boldsymbol{\theta}_r) = \left(\sum_{i \in I_\alpha} \frac{\tilde{h}_i^2}{\sigma_i^2} \right)^{-1} \left(\sum_{i \in I_\alpha} \frac{(\bar{y}_i - b_i)\tilde{h}_i}{\sigma_i^2} \right), \tag{2.23}$$

$$b_\alpha(\boldsymbol{\theta}_r) = \left(\sum_{i \in I_\alpha} \frac{1}{\sigma_i^2} \right)^{-1} \left(\sum_{i \in I_\alpha} \frac{\bar{y}_i - s_i\tilde{h}_i}{\sigma_i^2} \right), \tag{2.24}$$

where I_α is the set of all datapoints that share the scaling factor s_α (and per assumption therefore also offset factor b_α). To obtain non-interdependent equations, (2.23) can be inserted in (2.24), which will also eliminate the noise parameters. The noise parameters can then be calculated from known scaling and offset parameters as

$$\sigma_\beta(\boldsymbol{\theta}_r) = \left(\sum_{i \in I_\beta} 1\right)^{-1} \left(\sum_{i \in I_\beta} (\bar{y}_i - (s_i \tilde{h}_i + b_i))^2\right). \tag{2.25}$$

In this way, hierarchical optimisation reduces the parameter space and avoids performance losses due to observation function parameters, resulting in increased optimisation efficiency.

Algorithm 2.2 Hierarchical optimisation

Input: $J(\boldsymbol{\theta_r}, \boldsymbol{s}(\boldsymbol{\theta}_r), \boldsymbol{b}(\boldsymbol{\theta}_r), \boldsymbol{\sigma}(\boldsymbol{\theta}_r)), \quad \boldsymbol{\theta}_{r_{init}}$

- Simulate $\tilde{h}_{init}(\boldsymbol{x}(\boldsymbol{\theta}_r), \boldsymbol{\theta_r})$ ▷ expensive
- Compute $\boldsymbol{s}_{init}$, $\boldsymbol{b}_{init}$ and $\boldsymbol{\sigma}_{init}$
- **repeat**
  - Compute J
  - Compute $\nabla_{\boldsymbol{\theta}_r} J$ ▷ expensive
  - Update $\boldsymbol{\theta}_r$
  - Simulate $\tilde{h}(\boldsymbol{x}(\boldsymbol{\theta}_r), \boldsymbol{\theta_r})$ ▷ expensive
  - Compute $\boldsymbol{s}$, $\boldsymbol{b}$ and $\boldsymbol{\sigma}$
- **until** Stop condition

2.3.4 Posterior probability and regularisation methods

In Section 2.3.1 we derived a least squares objective function (Equation (2.14)) that describes the deviation of the model from the training data. However, a key goal of modelling is to predict future data, that is, in the end we want to minimise the prediction error, not the training error. When we decrease the training error at the cost of increasing the prediction error, we are overfitting the model. The more constraining knowledge and assumptions we put into the

model, the more robust it will be against outlier data. In other words, by putting bias into the model, we can reduce the variance in parameter estimates between optimisation runs on replicate datasets. This section will introduce methods for introducing assumptions to adjust this bias-variance tradeoff. For methods to select models with the optimal bias-variance tradeoff please refer to the Extended Methodological Background Section B.2.

2.3.4.1 Subset selection

Adding parameters to a model will allow it to better approximate the data, including outliers. One way of introducing assumptions is therefore to do the inverse: eliminating parameters from a model, or equivalently, setting them to zero. However, for a model with d parameters there are 2^d different possible parameter combinations. This exponential scaling with the number of parameters makes exhaustive best subset selection computationally intractable for larger models. Instead, the number of non-zero parameters (i.e. ℓ_0-norm) times a constant factor can be added to the objective function. However, this results in a mixed-integer nonlinear programming problem, that is neither differentiable nor continuous [78] and therefore precludes the choice of many efficient gradient-based optimisation algorithms. An alternative method that avoids exhaustive best subset selection is forward selection (Algorithm 2.3 [91]) or analogously, backward elimination. These approaches do not guarantee to find the best possible solution, but require only $\sum_{k=0}^{d-1}(d-k) = 1 + d(d+1)/2$ individual optimisations, i.e. have an asymptotic complexity of $\mathcal{O}(d^2)$. However, for the many cases where even performing just one optimisation is challenging, forward and backward search are intractable.

Algorithm 2.3 Forward stepwise selection

Input: $\mathcal{M}_0$ ▷ *null* model without parameters

for $k \in 0, \ldots, d-1$ **do**

Consider all $d-k$ models that augment $\mathcal{M}_k$ with one additional parameter

$\mathcal{M}_k \leftarrow$ best of the above $d-k$ models. ▷ *best* means lowest training error

end for

$\mathcal{M} \leftarrow$ best of $\mathcal{M}_0, \ldots, \mathcal{M}_d$ ▷ here *best* means lowest prediction error

2.3.4.2 ℓ_2 regularisation

Rather than eliminating parameters from the model, bias can be introduced in the Bayesian sense by adding a prior to the objective function. In Section 2.3.1 we assumed that we cannot guess a prior probability $p(\boldsymbol{\theta})$ in Equation (2.9) and excluded the prior, leading to a maximum likelihood objective function (Equation (2.14)). We can now leave a Gaussian prior in Equation (2.9) and follow the same operations as in the derivation of the maximum likelihood objective. As before, the goal is to maximise Equation (2.9):

$$P(\boldsymbol{\theta}|\bar{\boldsymbol{y}}) = \frac{L(\bar{\boldsymbol{y}}|\boldsymbol{\theta}) \cdot P(\boldsymbol{\theta})}{P(\bar{\boldsymbol{y}})}.$$

We can again ignore $P(\bar{\boldsymbol{y}})$, but this time we assume that we have prior knowledge about the parameters:

$$P(\boldsymbol{\theta}|\bar{\boldsymbol{y}}) = L(\bar{\boldsymbol{y}}|\boldsymbol{\theta})P(\boldsymbol{\theta}), \tag{2.26}$$

or

$$P(\boldsymbol{\theta}|\bar{\boldsymbol{y}}) = L(\bar{\boldsymbol{y}}|\boldsymbol{\theta}) \prod_{j=1}^{d} P(\boldsymbol{\theta}_j), \tag{2.27}$$

where d denotes the dimension of the parameter vector $\boldsymbol{\theta}$. Again using Gaussian noise for the likelihood of a single datapoint (Equation (2.11)), the posterior for n independently distributed data points is

$$P(\bar{\boldsymbol{y}}|\boldsymbol{\theta}) = \prod_{i=1}^{n} L(\bar{y}_i|\boldsymbol{\theta}) \prod_{j=1}^{d} P(\boldsymbol{\theta}_j). \tag{2.28}$$

and the log-posterior is

$$\log L(\bar{\boldsymbol{y}}|\boldsymbol{\theta}) = -\frac{1}{2}\sum_{i=1}^{n}\left(\frac{\bar{y}_i - y_i(\boldsymbol{\theta})}{\sigma_i}\right)^2 - \log 2\pi\sigma_i^2 - \frac{1}{2}\sum_{j=1}^{d}\left(\frac{\theta_j - \mu_j}{\sigma_{b_j}}\right)^2 - log2\pi\sigma_{b_j}^2, \tag{2.29}$$

where μ_j and σ_{b_j} are mean and standard deviation of the prior guess for the j^{th} parameter. Ignoring the constant terms and multiplying with minus two, we arrive at the objective function

$$J(\boldsymbol{\theta}) = \sum_{i=1}^{n}\left(\frac{\bar{y}_i - y_i(\boldsymbol{\theta})}{\sigma_i}\right)^2 + \sum_{j=1}^{d}\left(\frac{\theta_j - \mu_j}{\sigma_{b_j}}\right)^2. \tag{2.30}$$

The second sum therefore adds a penalty in form of an ℓ_2-norm to the least squares objective if parameters deviate from the prior guess. To optimise the predictive power of the model, the strength of the introduced bias needs to be adjusted via a tuning parameter λ:

$$J(\boldsymbol{\theta}) = \sum_{i=1}^{n}\left(\frac{\bar{y}_i - y_i(\boldsymbol{\theta})}{\sigma_i}\right)^2 + \lambda\sum_{j=1}^{d}\left(\frac{\theta_j - \mu_j}{\sigma_{b_j}}\right)^2. \tag{2.31}$$

Like best subset selection, this Gaussian bias reduces the variance. One typical choice to prevent overfitting is to set the prior mean to zero, i.e. assuming the reaction is not there. However if the data ever so slightly drives a parameter θ_j away from zero, a Gaussian prior can never drive parameters to exactly zero, which would be needed for model reduction. Intuitively, this can be explained by the shape of the Gaussian: close to the mean zero, the derivative too becomes almost zero. Therefore, there will be a point at which moving θ_j even closer to zero cannot compensate for increases in the likelihood term. An alternative way to see why a Gaussian prior can never drive parameter values exactly to zero is to look at the necessary condition for a minimum

$$\nabla J(\boldsymbol{\theta}) = \nabla\sum_{i=1}^{n}\left(\frac{\bar{y}_i - y_i(\boldsymbol{\theta})}{\sigma_i}\right)^2 + \lambda\nabla\sum_{j=1}^{d}\left(\frac{\theta_j - \mu_j}{\sigma_{b_j}}\right)^2 := 0, \tag{2.32}$$

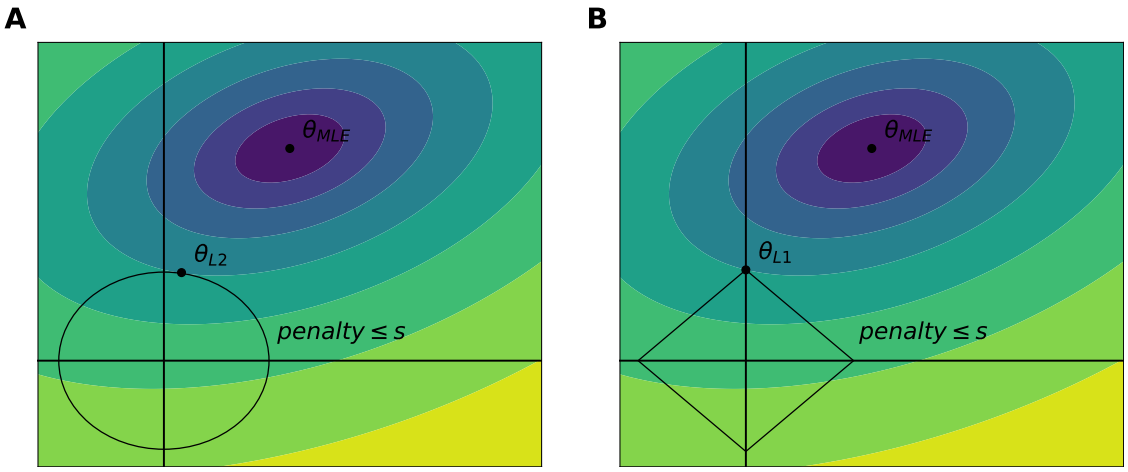


Figure 2.4 | ℓ_2 and ℓ_1 regularisation. ℓ_2 and ℓ_1 regularisation add penalty terms to the objective function. Geometrically these penalty terms can be interpreted as feasible regions of the optimisation problem. (**A**) ℓ_2 penalties correspond to ellipsoidal feasible regions $\sum_{j=1}^{d}\left(\frac{\theta_j-\mu_j}{\sigma_{b_j}}\right)^2 \leq s$ and cannot drive parameters to exactly zero. (**B**) ℓ_1 penalties correspond to to rhombic feasible regions $\sum_{j=1}^{d}\left(\frac{|\theta_j-\mu_j|}{\sigma_{b_j}}\right) \leq s$ and can eliminate parameters by driving them to zero.

and compare it to the constraint optimisation problem

$$\begin{aligned} \min_{\boldsymbol{\theta}} \quad & \sum_{i=1}^{n}\left(\frac{\bar{y}_i - y_i(\boldsymbol{\theta})}{\sigma_i}\right)^2 \\ \text{s.t.} \quad & \sum_{j=1}^{d}\left(\frac{\theta_j-\mu_j}{\sigma_{b_j}}\right)^2 \leq s. \end{aligned} \tag{2.33}$$

Using the method of Lagrange multipliers to optimise (2.33) we arrive at Equation (2.32) constraint to $\sum_{j=1}^{d}\left(\frac{\theta_j-\mu_j}{\sigma_{b_j}}\right)^2 \leq s$. In other words, for $s = \infty$, (2.33) and (2.32) are equivalent. More generally, for every $s \geq 0$ there is a λ such that (2.31) and (2.33) will give the same parameter estimates. Figure 2.4A illustrates why ℓ_2 regression will never force parameters to exactly zero.

2.3.4.3 ℓ_1 regularisation

While a Gaussian prior can reduce overfitting, we have seen that it cannot force parameters to zero, as it has zero slope at its maximum. An alternative choice

for a prior is a Laplace distribution,

$$P(\theta) = \frac{1}{2b} e^{-|\theta-\mu|/b}, \tag{2.34}$$

where

- μ is the mean and
- $b > 0$ is the scale parameter.

The Laplace distribution has an increasing slope as θ moves closer to zero. Plugging the Laplace distribution into Equation (2.28) and taking the natural logarithm we obtain

$$\log L(\bar{\boldsymbol{y}}|\boldsymbol{\theta}) = -\frac{1}{2}\sum_{i=1}^{n}\left(\frac{\bar{y_i} - y_i(\boldsymbol{\theta})}{\sigma_i}\right)^2 - \log 2\pi\sigma_i^2 - \sum_{j=1}^{d}\left(\frac{|\theta-\mu|}{b_j}\right) - \log 2b_j, \tag{2.35}$$

Ignoring the constant terms, multiplying with minus one and introducing the tuning parameter λ, we arrive at the objective function

$$J(\boldsymbol{\theta}) = \sum_{i=1}^{n}\left(\frac{\bar{y_i} - y_i(\boldsymbol{\theta})}{\sigma_i}\right)^2 + \lambda\sum_{j=1}^{d}\left(\frac{|\theta-\mu|}{b_j}\right). \tag{2.36}$$

The second sum therefore adds a penalty in form of an ℓ_1-norm to the least squares objective if parameters deviate from the prior guess. In analogy to ℓ_2 regularisation, minimising the objective function in Equation (2.36) is equivalent to

$$\begin{aligned} \min_{\boldsymbol{\theta}} \quad & \sum_{i=1}^{n}\left(\frac{\bar{y_i} - y_i(\boldsymbol{\theta})}{\sigma_i}\right)^2 \\ \text{s.t.} \quad & \sum_{j=1}^{d}\frac{|\theta-\mu|}{b_j} \leq s. \end{aligned} \tag{2.37}$$

Figure 2.4B illustrates why ℓ_1 regularisation can drive parameters to zero and therefore eliminate reactions from the model. Whether ℓ_1 or ℓ_2 regularisation have a lower error on test data depends on whether some parameters θ are indeed zero. To minimise the test error, it is critical to choose the optimal value of the

tuning parameter λ. This is done by running multiple optimisations at different λ and selecting the value that minimises the test error. Direct and approximate methods for obtaining the test error are described in the Extended Methodological Background Section B.2.

3

The Structure of the Mammalian Cell Cycle Model

»... the separation of science and non-science is not only artificial but also detrimental to the advancement of knowledge. If we want to understand nature, if we want to master our physical surroundings, then we must use all ideas, all methods, and not just a small selection of them.«

Paul Feyerabend – Against Method[92]

Contents

3.1 Introduction

3.1.1 Principles of cell cycle control

The most fundamental task of the cell cycle is the replication of the genome and the distribution of two identical copies to two daughter cells. In eukaryotes, genome replication happens during S phase and genome distribution happens during mitosis (M phase). The M-phase is again subdivided in prophase, metaphase, anaphase and telophase. M and S phase are separated by gap phases (G1 and G2). To prevent ploidy change, the processes of genome replication and segregation into two daughter cells must be strictly alternating, i.e. the cell cycle must follow the succession G1-S-G2-M (Figure 3.1). With few exceptions, this progression through the cell cycle must also be coordinated with the synthesis of all other cellular components in such a way

that the time between two successive divisions equals the mass doubling time [93]. A control system of checkpoints ensures that a cell adheres to these conditions [94]. The restriction point (RP) is a checkpoint in G1 that prevents cell cycle progression, until it is lifted by continuous or pulse like mitogen signalling [95]. First discovered by Temin [96], it is defined as a point after which progression through the cell cycle no longer requires mitogen signalling [97]. Recent evidence suggests that budding yeast [98] and mammalian cells [99, 100] also have a size control checkpoint in G1, perhaps coinciding with the RP. Additionally, the cell constantly monitors DNA damage throughout G1 in a mechanism referred to as DNA damage checkpoint. Cell cycle progression is only licensed when this checkpoint is lifted. Once the G1/S transition is passed, the genome becomes replicated. Returning to a G1 state is no longer possible and cell survival depends on completion of the remaining cell cycle. The completion of genome replication marks the end of S phase. In order to prevent re-replication of the genome, the chromosome replication machinery can only be assembled throughout late M and G1 phase. It is disassembled upon initiation of replication [101]. Mitotic entry and nuclear membrane breakdown at the G2/M transition are licensed by lifting another DNA-damage checkpoint, checking if the whole genome is duplicated and undamaged [102]. Correct segregation of two copies of the genome is ensured by the spindle assembly checkpoint. Lifting this checkpoint licences the metaphase/anaphase (M/A) transition. Nuclear membranes are formed around the two segregated copies of the genome at mitotic exit. In mammalian cells, this process is accompanied by cell division [103].

3.1.2 Molecular machinery of the cell cycle

Cell cycle progression is controlled by an intricate molecular machinery with multiple involved components. Cyclin dependent kinases (Cdks), and their counteracting

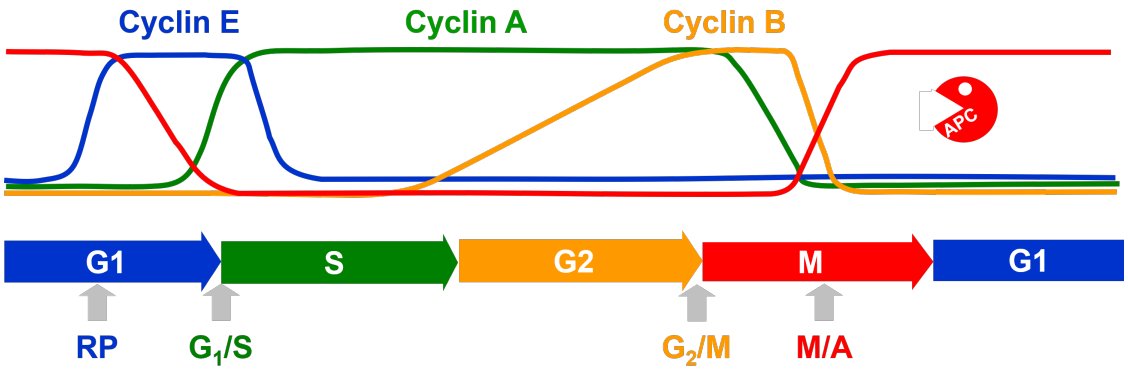


Figure 3.1 | Schematic time course of key cell cycle regulators. The cell cycle strictly follows the succession G1 - S - G2 - and M phase. Cyclins play an important role in regulating this process. Their synthesis is controlled by transcription factors and their degradation by the ubiquitin ligases SCF (for Skp, Cullin, F-box containing complex) and APC/C (for anaphase promoting complex or cyclosome). Rapid cyclin accumulation or depletion is intimately connected to the restriction point (RP), and the G1/S and G2/M and metaphase/anaphase (M/A) transitions (Figure adapted from Bela Novak's lecture slides and inspired by Morgan *et al.* [104]).

phosphatases are considered to be a central element of this machinery. While our knowledge of counteracting phosphatase regulation is rather limited, Cdk activity is largely determined by three mechanisms [3]: (1) As the name suggests, Cdks rely on the availability of binding partner proteins called cyclins. Cyclin D activates Cdk4 and Cdk6, cyclin E activates Cdk2, cyclin A activates Cdk1 and Cdk2, and cyclin B activates Cdk1. The availability of cyclins themselves is largely determined by transcriptional regulation and proteasomal degradation. The E3 ubiquitin ligases APC/C and SCF target cyclins (and many other cell cycle regulators) for degradation. Their substrate specificity is regulated by adapter subunits such as Cdh1 and Cdc20, and Skp2 and βTRCP, respectively. In addition, SCF dependent degradation is regulated by target phosphorylation [103, 105]. In wild type cells cyclin E accumulation is intimately connected with passing the RP. Cyclin A accumulation drives the G1/S transition and leads to cyclin E degradation.

Passing the spindle assembly checkpoint results in cyclin B degradation (cyclin A is degraded slightly earlier) [104, 106] (Figure 3.1). Despite cyclins can influence substrate specificity of their Cdks [107, 108] it appears that overall Cdk activity is the main driver of cell cycle progression [109, 110]. (2) Cdk inhibitors bind to and thereby inhibit cyclin:Cdk complexes. This mechanism is active at the DNA damage checkpoints in G1 and G2 [103]. (3) Cdk activity can be controlled by inhibitory tyrosine phosphorylation of the catalytic subunit (Cdk). Most prominently, removing an inhibitory phosphorylation of Cdk1 drives the G2/M transition [93].

3.1.3 Bistability in the cell cycle

Chapter 2 Section 2.1 discussed several model representation formats and simulation methods. Systems of ODEs are particularly well-suited to study how the cell cycle transitions and checkpoints emerge from biochemical reactions as they ignore confounding noise. Each ODE describes how the concentration of a species changes over time, as a function of the concentration of all species. Collectively, these changes describe a direction field in the space spanned by all species (phase space). In case of a system of only two ODEs, this can easily be visualized in a phase plane diagram (Figure 3.2A), where the arrows indicate how the concentrations of the species changes over time. Lines (or surfaces, in case of three or more species/dimensions) along which one of the variables does not change are called nullclines. They are obtained by setting the time derivative of the corresponding variable to zero. Points at which none of the variables change (i.e. where all nullclines intersect) are called steady states. Steady states can be qualitatively different and are coarsely classified into stable and unstable steady states. Stable steady states are approached by the system variables, unstable steady states repel the system variables. A system that has two stable steady states is called bistable. Which of the two stable steady states

is approached depends on the initial conditions of the system. Once a stable steady state is reached, the system will not jump to the other stable steady state unless it is actively perturbed. This behaviour is often compared to a toggle switch and called hysteresis [111]. The perturbation corresponds to a parameter change of the ODE system, which results in changes of the direction field and thus a nullcline shift and changes in the location of steady states. Thereby, a change of parameters can also lead to qualitative changes in the system, like changes in the number or properties of steady states. Such qualitative changes in system behaviour are called bifurcations and are visualized in bifurcation diagrams like the one shown in the top panel of Figure 3.2C. Increasing the parameter X over the upper bifurcation point leads to loss of the lower steady state and the system variable Y jumps up to the upper steady state. However, reducing X just under the upper bifurcation point does not lead to loss of the upper steady state and the system stays in the upper steady state. Therefore, bistability provides to a certain extent irreversibility.

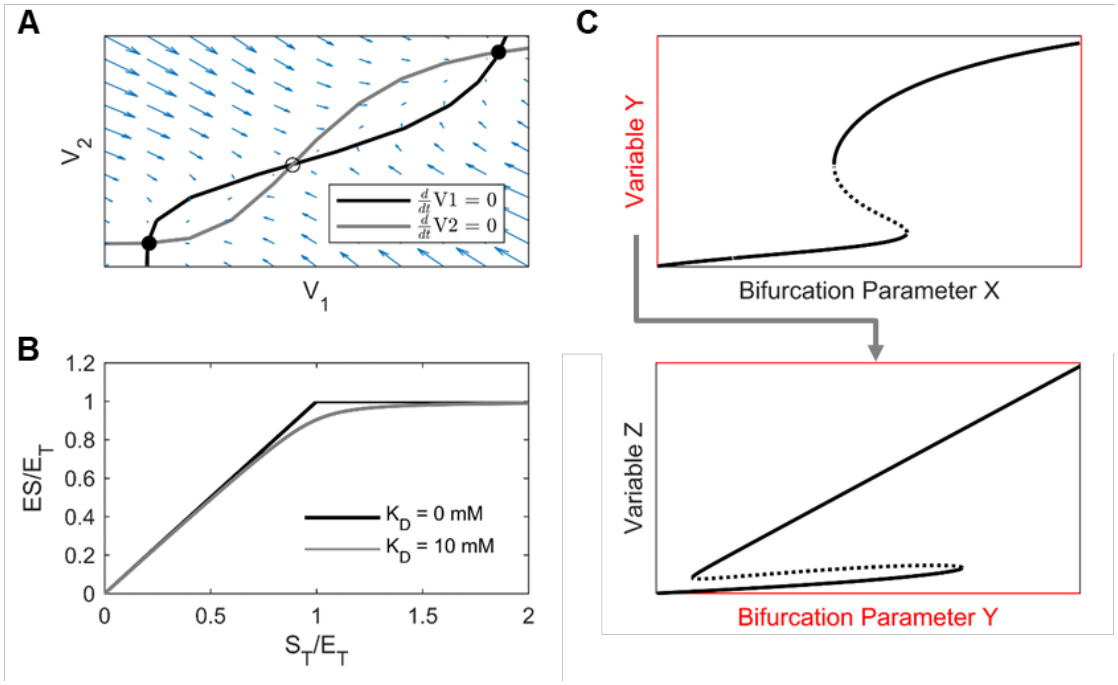


Figure 3.2 | Bistability and cell cycle progression. Ultrasensitivity and positive feedback loops in biochemical reaction networks can give rise to bistability. Cell cycle progression follows through a series of bistable cell cycle transitions that trigger each other. This makes cell cycle progression dynamically irreversible, ensuring that cell cycle events like DNA replication and chromosome segregation happen once and only once per cycle. **(A)** Visualisation of bistability in a phase plane diagram. V_1 and V_2 represent the concentrations of two variables. Their change over time as function of V_1 and V_2 is described by a system of two ODEs and indicated by the blue arrows. The V_1 nullcline (full black line) represents all points where V_1 does not change over time (analogous for the V_2 nullcline in grey). The V_1 and V_2 nullclines intersect at two stable (filled circle) and one unstable (unfilled circle) steady state. **(B)** Ultrasensitive responses. Enzyme:substrate complex concentration (ES) from Equation (3.4) as a function of total substrate concentration (S_T) normalized to total enzyme concentration (E_T) in protein-protein interaction networks. At dissociation constants (K_D) in the µM range, as usually observed for protein kinases or phosphatases [112, 113], the enzyme is fully saturated when S_T exceeds E_T. For reasons explained in the derivation of Equation (3.6) interactions between cell cycle regulators are therefore best modelled with mass action kinetics. This enzyme-substrate binding is formally identical to inhibitor-target protein binding. Therefore, when replacing enzyme with inhibitor and substrate with target protein, the figure also shows how target protein accumulation is almost completely buffered by complexation with its inhibitor until both concentrations are equal. Further target protein accumulation leads to a sharp, ultrasensitive, linear increase in free target protein concentration. This form of inhibitor ultrasensitivity is a common mechanism in cell cycle transition and checkpoint networks. ...

... **(C)** Schematic for merging two bistable switches. Solid lines represent stable and dotted lines unstable steady states. The response variable of the bistable switch in the upper panel serves as bifurcation parameter for the bistable switch in the lower panel.

The rapid changes of state variables observed at cell cycle transitions could in principle be explained by (a) a parameter-change induced rapid movement of stable steady state location, or (b) by the presence of two stable steady states before the transition and parameter-change induced loss of the current stable steady state at the transition (note that a parameter in a certain cell cycle transition model can be a system variable in a more complete model). However, only (b) can explain the robustness of irreversibility in the cell cycle transitions. This robustness is due to hysteresis, i.e. even if perturbations would partially revert the parameter change, leading to recreation of the previous steady state, the system will stay in the current steady state. The larger the bistable range with respect to a certain parameter, the more robust the steady with respect to that parameter. Indeed, experimentally determined hysteretic behaviour provides evidence for (b) in the RP [114] and the G2/M transition [111, 115]. Barr *et. al* provide further experimental data on the G1/S transition that is interpreted in favour of (b) [59, 116]. One may therefore think of the cell cycle as a ratchet wheel that can only move in one direction. But what are the requirements for a chemical reaction network to exhibit bistability? It has been shown that two conditions need to be fulfilled. First, there must be a positive feedback loop in the network. In biochemical networks this is often realized by mutual inhibition between to species. Second, one leg of the network must contain a sharp input output relationship termed ultrasensitive response [117]. In biochemical networks, this is often realized with Goldbeter-Koshland switches [118], Hill kinetics [119], or titration of an inhibitor. However,

the Michaelis-Menten kinetics required for ultrasensitivity in Goldbeter-Koshland switches, and Hill kinetics both assume that the total substrate concentration (S_T) is much larger than the concentration of the enzyme:substrate complex (ES). While this seems valid for instance in metabolic reactions, it seems less justified for the reactions of a protein-interaction network like the one governing the cell cycle. Here, the substrate is usually another protein that tightly binds to the enzyme and has a similar concentration as the enzyme. We therefore write the reaction velocity as

$$\frac{dS_T}{dt} = -k_{cat} \cdot ES, \tag{3.1}$$

where t is time and k_{cat} the turnover number of the enzyme. Making a steady state approximation for ES

$$ES \approx \frac{E_{free} \cdot S_{free}}{K_D}, \tag{3.2}$$

where

- E_{free} is the free enzyme concentration,
- S_{free} the free substrate concentration and
- K_D the dissociation constant of the enzyme:substrate complex,

substituting E_{free} with $E_T - ES$ and S_{free} with $S_T - ES$ to obtain the quadratic equation

$$ES \approx \frac{(E_T - ES) \cdot (S_T - ES)}{K_D}. \tag{3.3}$$

Solving for ES, we obtain an equation that is symmetric with respect to E_T and S_T:

$$ES \approx \frac{1}{2}(E_T + S_T + K_D - \sqrt{(E_T + S_T + K_D)^2 - 4 \cdot E_T \cdot S_T}). \tag{3.4}$$

Here, we used that ES can never exceed $(E_T + S_T)/2$ to determine that the "$\pm$" sign can only be "$-$". The graph for equation (3.4) is shown in Figure 3.2B.

Considering that for a typical enzyme involved in the cell cycle the K_D is in the µM range [112, 113], while typical cell cycle regulators appear in the nM range[1], saturation effects in enzymatic reactions are unlikely. Therefore, we can substitute E_T for E_{free} and S_T for S_{free} in equation (3.2) to obtain

$$ES \approx \frac{E_T \cdot S_T}{K_D}. \tag{3.5}$$

Plugging Equation (3.5) into Equation (3.1) we get

$$\frac{dS_T}{dt} \approx -k \cdot E_T \cdot S_T, \tag{3.6}$$

where $k = k_{cat}/K_D$. Therefore, despite enzymatic reactions are clearly no elementary reactions, the enzymatic reactions governing the cell cycle can be approximated by the form of mass action kinetics given in Equation (3.6). For simplicity, mass action kinetics will be used for all reactions[2]. We note that this simplicity comes at the cost of a less accurate phenomenological description of multisite-phosphorylation reactions. As described by Ferrell *et al.* [117], multiple mass action phosphorylation reactions can be approximated with a single reaction using Hill kinetics. However, due to the use of mass action rate laws, ultrasensitive responses in the model cannot result from Goldbeter-Koshland switches and Hill kinetics. Nevertheless, they can arise from titration of an inhibitor by its target protein. The concentration of the free target protein P_{free} is given by

$$P_{free} = P_T - PI, \tag{3.7}$$

where P_T denotes the total target protein concentration and PI the concentration of the target protein:inhibitor complex. The binding of an inhibitor to its target

[1] unpublished work using PaxDB data; Yin Hoon Chew and Jonathan Karr

[2] The only exception is the phosphorylation of nuclear pores in model version 4.0.0, where Hill kinetics is used.

protein is formally identical to the enzyme-substrate binding described by Equations (3.2)–(3.4) and Figure 3.2B. Consequently, we can obtain PI by replacing enzyme with inhibitor and substrate with target protein in Equations (3.3) and (3.4). For the usually tight binding between target protein and inhibitor, P_{free} therefore sharply accumulates in a linear way as soon as P_T exceeds the total inhibitor concentration.

3.1.4 Objectives and approach

The work in this chapter aims at identifying an extensive biochemical reaction network that is capable of explaining how the full cell cycle and its transitions can emerge from the underlying biochemical reactions. To turn the network into a simulatable model we will assign rate laws to all reactions. An accurate parameterisation of the model is NOT an objective of this chapter. Instead, kinetic parameters will be chosen freely to demonstrate the capability of the network to generate sustained oscillatory behaviour. Choosing an extensive network increases the chances of including the most relevant reactions in the cell cycle. It also increases the chances of overfitting, which is deliberately accepted as overfitting can be handled with model reduction and selection methods (see methods in Section 2.3.4. Despite being extensive, the model shall ignore random events like spindle fibre attachment at the M/A transition, discontinuous events such as chromosome segregation and thus DNA replication, and cell size and thus size control. Also not included will be pathways adjacent to the cell cycle, such as apoptosis, DNA damage signalling upstream of TP53, growth factor signalling upstream of cyclin D, energy availability signalling, and many others.

To deal with the complexity of this endeavour, the model design process will follow a sequence of less complex steps. First, models of the RP and the G1/S, G2/M and M/A transitions will be created independently. Accounting for experimental

evidence, the rapid changes of cell cycle regulator concentrations at RP, G1/S transition and G2/M transition shall be modelled as bistable switches. This requires ultrasensitivity in protein-protein interaction networks, which shall arise purely from fundamental mass action kinetics. To generate a full cell cycle model, the submodels shall then be combined in a stepwise manner, by using the response variable of the RP toggle as bifurcation parameter of the G1/S toggle (Figure 3.2C). Similarly, via a short sequence of linker reactions, the response variables of the RP and G1/S toggles will increase the bifurcation parameter value of the G2/M toggle. A negative feedback loop representing the M/A transition will reset the system. The RP toggle as first element in this chain will be flipped by Cyclin D as proxy of mitogen availability and only input parameter of the core cell cycle model. This core cell cycle model shall then be equipped with an interface to DNA damage response pathways and extended to a two-compartmental model to describe transport of cell cycle regulators between nucleus and cytoplasm.

3.2 Results and discussion

This section describes the submodels of the RP and the G1/S, G2M and M/A transitions, and their fusion to a full cell cycle model. All model variables are explained in Table 3.1 and measured in arbitrary units with characteristic scale ($AU_{ch.scale}$) to facilitate rescaling. We will also adhere to this terminology in the following text passages whenever the model variable description of Table 3.1 fits. We will further refer to the two different stable steady states of the bistable systems as *upper* and *lower* with respect to an indicated system variable. Similarly, bifurcation points will be referred to as *upper* or *lower* with respect to the bifurcation parameter value. All of the following equations are written in explicit form.

Symbols that appear on the left-hand side in one of the equations of a particular submodel are variables, the remaining symbols represent parameters. The parameter values and their description can be found in the directory /versions/v0.0.1/ of the cell_cycle_model GitHub repository. Since Cdks are in large excess over cyclins [120] the model assumes that cyclin:Cdk complex concentration is proportional to cyclin concentration. Further details on major model assumptions and on checking the ODE model for consistency with the underlying chemical reactions can be found in the methods (Section 3.4).

3.2.1 Restriction point submodel

The molecular basis of the restriction point is described by the Rb/E2f pathway shown in Figure 3.3A [114, 121]. E2f is a transcription factor that drives the transcription of cyclin E. Before the restriction point, E2f is kept inactive by binding to Rb [122]. However, this binding is inhibited by phosphorylation of Rb, which is mediated, for example, by CycD and CycE. The Rb/E2F pathway is described by Equations (3.8)–(3.15). It is assumed that transcription factors are present in much higher concentration than promoters, i.e. promoter binding has negligible effect on free transcription factor concentration. This assumption will be removed for full cell cycle model versions $\geq$ v2.1.0 (see also Table 3.2).

$$\frac{dRb}{dt} = k_{DpRb} \cdot pRb - k_{PhRb} \cdot (CycD + CycE) \cdot Rb + k_{DeE2f} \cdot E2f{:}Rb \tag{3.8}$$

$$\frac{dCycE}{dt} = k_{SyCe1} + k_{SyCe2} \cdot E2f{:}Px - k_{DeCe} \cdot CycE \tag{3.9}$$

$$\begin{aligned}\frac{dE2f}{dt} &= k_{SyE2f1} + k_{SyE2f2} \cdot E2f{:}Px - k_{DeE2f} \cdot E2f \\ &\quad + k_{DiE2fRb} \cdot E2f{:}Rb - k_{AsE2fRb} \cdot E2f \cdot fRb\end{aligned} \tag{3.10}$$

$$\frac{dtE2f}{dt} = k_{SyE2f1} + k_{SyE2f2} \cdot E2f{:}Px - k_{DeE2f} \cdot tE2f \tag{3.11}$$

$$\frac{dE2f{:}Px}{dt} = k_{AsEPx} \cdot E2f \cdot (t_{Dna} - E2f{:}Px) - k_{DiEPx} \cdot E2f{:}Px \tag{3.12}$$

$$pRb = t_{Rb} - Rb \tag{3.13}$$

$$fRb = Rb - (tE2f - E2f) \tag{3.14}$$

$$E2f{:}Rb = tE2f - E2f \tag{3.15}$$

Even without E2f autoactivation (i.e. replacing Equation (3.11) with $tE2f = 0.5$) the mutual inhibition of CycE and Rb, combined with the inhibitor ultrasensitivity conferred by Rb allows for bistability in the Rb/E2f pathway (Figure 3.3B,C). The bistable range and thus the robustness of this network with respect to variations in CycD concentration is further increased by transcriptional autoactivation of E2f. In agreement with the notion that once the restriction point is passed, cell cycle progression becomes independent of mitogen signalling [97], flipping the toggle to the high CycE state is a truly irreversible process in our submodel. Reverting the system would require negative CycD concentrations (Figure 3.3D). The time course at CycD close to the upper bifurcation point shows how Rb, CycE and E2f approach the high CycE steady state (Figure 3.3E). The system almost comes to rest when the variables are close to the lost low CycE steady state. The increasing concentrations in CycE and E2f will trigger CycA accumulation by switching the G1/S toggle. Their values in the high CycE steady state will therefore set upper boundaries for the upper bifurcation point in the G1/S submodel.

Table 3.1 | Variables of the core model.

ODE variable (Unit[1])	Description	BNGL equivalent
t (min)	time	time
CycE (AU_E)	cyclin E:Cdk2 complex	`CCNE()`
tE2f (AU_{E2f})	total transcription factor E2f	`E2F(DBD!?,RB1!?,Ser332)`
E2f (AU_{E2f})	free, unphosphorylated E2f	`E2F(DBD,RB1,Ser332~u)`
pE2f (AU_{E2f})	free, phosphorylated E2f	`E2F(DBD,RB1,Ser332~p)`
Rb (AU_{E2f})	unphosphorylated retinoblastoma protein (incl. (p)E2f:Rb)	`RB1(E2F!?,Ser807_Ser811~u)`
pRb (AU_{E2f})	phosphorylated retinoblastoma protein (always free)	`RB1(E2F,Ser807_Ser811~p)`
fRb (AU_{E2f})	free, unphosphorylated Rb	`RB1(E2F,Ser807_Ser811~u)`
pE2f:Rb (AU_{E2f})	phosphorylated E2f bound to retinoblastoma protein	`E2F(DBD,RB1!1,Ser332~p).` `RB1(E2F!1,Ser807_Ser811~u)`
E2f:PX (AU_{Px})	E2f bound to the promoter of gene X (CycA, CycE, E2f, FoxM1)	`E2F(DBD!1,RB1,Ser332~u).` `*_promoter(E2F!1)`
E2f:Rb (AU_{E2F})	unphosphorylated E2f bound to retinoblastoma protein	`E2F(DBD,RB1!1,Ser332~u).` `RB1(E2F!1,Ser807_Ser811~u)`
CycA (AU_A)	cyclin A:Cdk1/2 complex	`CCNA()`
tEmi1 (AU_{Cdh1})	total early mitotic inhibitor 1 (Emi1)	`FBXO5(APC!?,FZR1,Ser182)`
Emi1 (AU_{Cdh1})	free, unphosphorylated Emi1	`FBXO5(APC,FZR1,Ser182~u)`
pEmi1 (AU_{Cdh1})	phosphorylated Emi1 (always free)	`FBXO5(APC,FZR1,Ser182~p)`
Apc (AU_{Cdh1})	unphosphorylated anaphase promoting complex (APC/C) without substrate adaptor subunit	`APC(FZR1_CDC20,FBXO5,Ser355~u)`
pApc (AU_{Cdh1})	free, phosphorylated APC/C	`APC(FZR1_CDC20,FBXO5,Ser355~p)`
Cdh1 (AU_{Cdh1})	free, unphosphorylated APC/C substrate adaptor subunit Cdh1	`FZR1(APC,FBXO5,nTerm~u)`
pCdh1 (AU_{Cdh1})	phosphorylated Cdh1 (always fee)	`FZR1(APC,FBXO5,nTerm~p)`
Apc:Cdh1 (AU_{Cdh1})	APC/C bound to Cdh1	`APC(FZR1_CDC20!1,FBXO5,Ser355~u).` `FZR1(APC!1,FBXO5,nTerm~u)`
pApc:Cdh1 (AU_{Cdh1})	phosphorylated APC/C bound to Cdh1	`APC(FZR1_CDC20!1,FBXO5,Ser355~p).` `FZR1(APC!1,FBXO5,nTerm~u)`
Apc:Cdh1:Emi1 (AU_{Cdh1})	APC/C bound to Cdh1 and Emi1	`APC(FZR1_CDC20!1,FBXO5!2,Ser355~u).` `FZR1(APC!1,FBXO5!3,nTerm~u).` `FBXO5(APC!2,FZR1!3,Ser182~u)`
pApc:Cdh1:Emi1 (AU_{Cdh1})	phosphorylated APC/C bound to Cdh1 and Emi1	`APC(FZR1_CDC20!1,FBXO5!2,Ser355~p).` `FZR1(APC!1,FBXO5!3,nTerm~u).` `FBXO5(APC!2,FZR1!3,Ser182~u)`
tFoxM1 (AU_{Fox})	total transcription factor FoxM1	`FOXM1(DBD!?,Thr600)`
pFoxM1 (AU_{Fox})	phosphorylated (i.e. active) transcription factor FoxM1	`FOXM1(DBD,Thr600~p)`
tCycB (AU_B)	cyclin B:(p)Cdk1 complex	`CCNB(CDK1_Thr14_Tyr15)`
pFoxM1:PCycB (AU_{Pb})	phosphorylated FoxM1 bound to the promoter of cyclin B	`FOXM1(DBD!1,Thr600~p).` `CCNB_promoter(FOXM1!1)`
CycB:Cdk1 (AU_B)	cyclin B in complex with unphosphorylated (i.e. active) Cdk1	`CCNB(CDK1_Thr14_Tyr15~u)`
Wee1 (AU_{Wee})	unphosphorylated (i.e. active) Wee1 kinase	`WEE1(Ser123~u)`
pCdc25 (AU_{C25})	phosphorylated (i.e. active) Cdc25 phosphatase	`CDC25(pSites~p)`
pGw (AU_{Gw})	phosphorylated Greatwall kinase	`MASTL(Thr198~p)`
pEnsa (AU_{Ensa})	total (incl. pEnsa:B55) phosphorylated forms of Endosulfine alpha and Arpp19	`ENSA_ARPP19(PPP2R2B!?,` `Ser62_Ser67~p)`
B55 (AU_{Ensa})	protein phosphatase 2A core dimer in complex with the regulatory subunit B55	`PPP2R2B(ENSA_ARPP19)`
pEnsa:B55 (AU_{Ensa})	phosphorylated Ensa bound to B55 complex	`ENSA_ARPP19(PPP2R2B!1,` `Ser62_Ser67~p).` `PPP2R2B(ENSA_ARPP19!1)`
tCdc20 (AU_{Cdh1})	total APC/C substrate adaptor subunit Cdc20	`CDC20(APC!?)`
pFoxM1:PCdc20 (AU_{Pc})	phosphorylated FoxM1 bound to the promoter of Cdc20	`FOXM1(DBD!1,Thr600~p).` `CDC20_promoter(FOXM1!1)`
Cdc20 (AU_{Cdh1})	free Cdc20	`CDC20(APC)`
pApc:Cdc20 (AU_{Cdh1})	phosphorylated APC/C bound to Cdc20	`APC(FZR1_CDC20!1,` `FBXO5,Ser355~p).CDC20(APC!1)`

[1] As arbitrary units (AU) of dimension "concentration" with characteristic scale indicated in subscript.
* one of CCNE, CCNA, E2F, FBXO5, FOXM1.

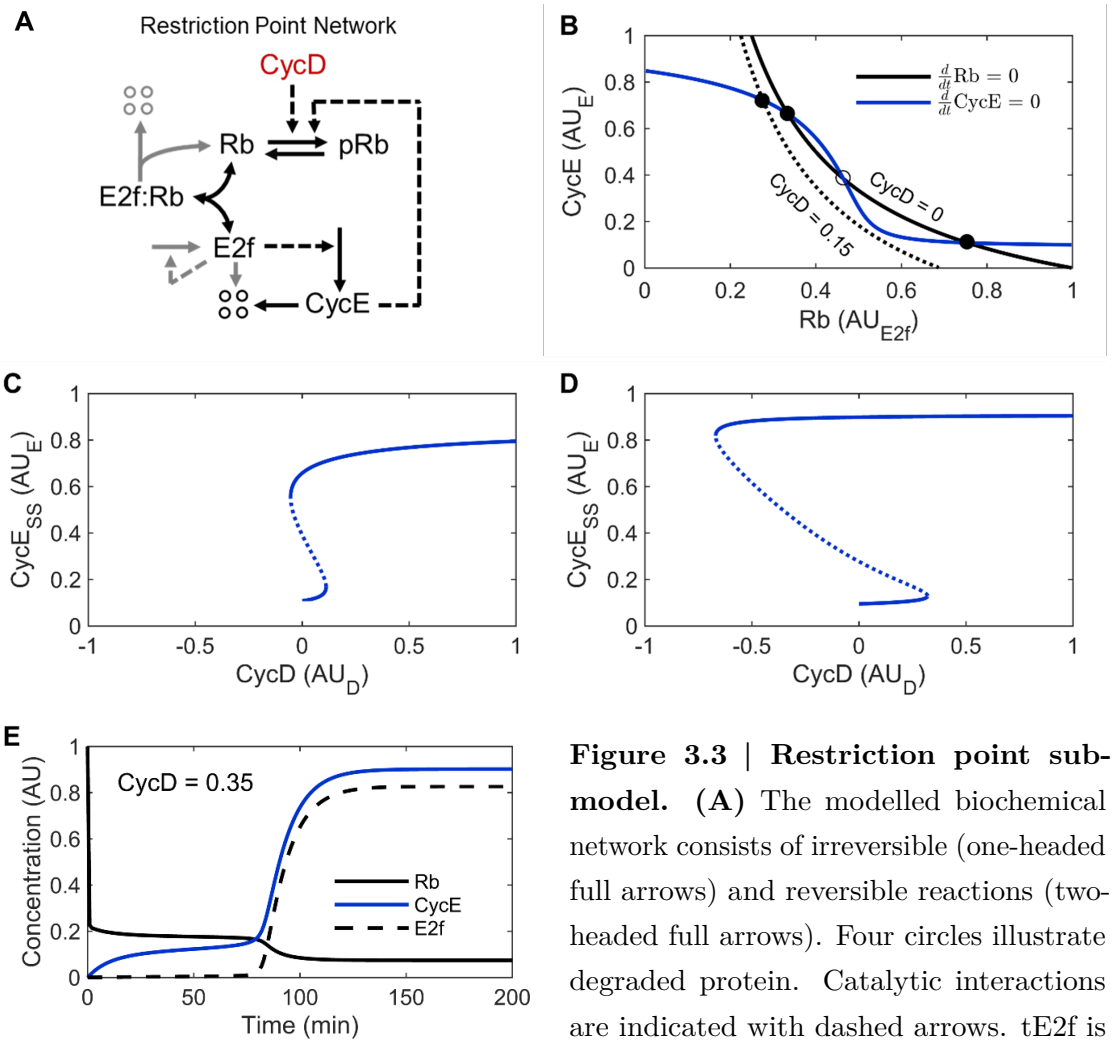


Figure 3.3 | Restriction point submodel. **(A)** The modelled biochemical network consists of irreversible (one-headed full arrows) and reversible reactions (two-headed full arrows). Four circles illustrate degraded protein. Catalytic interactions are indicated with dashed arrows. tE2f is considered constant except in the model underlying panels (D) and (E), with corresponding turnover reactions indicated by arrows in lighter colour. CycD (red) serves as bifurcation parameter. **(B)** Phase plane showing the nullclines for CycE and Rb intersecting at two stable (filled circles) and one unstable (unfilled circle) steady states when CycD is set to 0 AU_D. There is only one steady state at CycD = 0.15 AU_D The system was reduced to two dimensions by making a steady state approximation for E2f and calculating E2f:Rb and pRb via conservation laws. **(C, D)** Bifurcation diagrams of the non-reduced system with constant (C) and auto-activated (D) E2F, showing stable (solid line) and unstable (dotted line) steady states of CycE. **(E)** Time course of unphosphorylated Rb, CycE and E2f at CycD = 0.35 AU_D. For variable abbreviations please refer to Table 3.1. Parameter values are available the /versions/v0.0.1/ directory of the cell_cycle_model GitHub repository.

3.2.2 G1/S transition submodel

Like cyclin E, transcription of cyclin A is also activated by E2f. However, there is a delay between CycE and CycA protein accumulation, that shall be captured by the G1/S transition submodel. In contrast to cyclin E, cyclin A is susceptible to polyubiquitination by the Apc:Cdh1 complex. The high Apc:Cdh1 levels observed in G1 keep the CycA concentration relatively low, while the CycE concentration already rises. However, CycE and CycA can phosphorylate Cdh1, thereby preventing its association with Apc [103]. Additionally, E2f also activates the transcription of Emi1 [123], which is a very slowly polyubiquitinated pseudosubstrate of Apc:Cdh1. Its tight binding to Apc:Cdh1 further inhibits Apc:Cdh1 [124] (Figure 3.4A). This G1/S transition network is described by Equations (3.16)–(3.22). As total Apc appears to be in excess of total Cdh1 and tCdc20 [125], we assume in this submodel that all Cdh1 is bound to Apc. Similar G1/S transition submodels have been proposed by Barr *et al.* [59] and Novak *et* Tyson [126].

$$\begin{aligned}\frac{dtEmi}{dt} &= k_{SyEmi1} + k_{SyEmi2} \cdot E2f{:}Px - k_{DeEmi1} \cdot tEmi1 \\ &\quad - k_{DeEmi2} \cdot Apc{:}Cdh1{:}Emi1\end{aligned} \tag{3.16}$$

$$\begin{aligned}\frac{dCycA}{dt} &= k_{SyCa1} + k_{SyCa2} \cdot E2f{:}Px \\ &\quad - (k_{DeCa1} + k_{DeCa2} \cdot Apc{:}Cdh1) \cdot CycA\end{aligned} \tag{3.17}$$

$$\frac{dCycE}{dt} = k_{SyCe1} + k_{SyCe2} \cdot E2f{:}Px - k_{DeCe} \cdot CycE \tag{3.18}$$

$$\begin{aligned}\frac{dApc{:}Cdh1{:}Emi1}{dt} &= k_{AsACE} \cdot Apc{:}Cdh1 \cdot (tEmi - Apc{:}Cdh1{:}Emi1) \\ &\quad - (k_{DiACE} + k_{DeEmi1} + k_{DeEmi2} + k_{PhCdhE} \cdot CycE \\ &\quad + k_{PhCdhA} \cdot CycA) \cdot Apc{:}Cdh1{:}Emi1\end{aligned} \tag{3.19}$$

$$\begin{aligned}\frac{dpCdh1}{dt} &= (k_{PhCdhA} \cdot CycA + k_{PhCdhE} \cdot CycE) \cdot (t_{Cdh1} - pCdh1) \\ &\quad - k_{DpCdh} \cdot pCdh1\end{aligned} \tag{3.20}$$

$$\frac{dE2f{:}Px}{dt} = k_{AsEPx} \cdot E2f \cdot (t_{Dna} - E2f{:}Px) - k_{DiEPx} \cdot E2f{:}Px \tag{3.21}$$

$$Apc{:}Cdh1 = t_{Cdh1} - pCdh1 - Apc{:}Cdh1{:}Emi1 \tag{3.22}$$

The mutual inhibition between CycA and Apc:Cdh1, combined with the inhibitor ultrasensitivity conferred by Emi1 allows for bistability in this G1/S transition network. Using E2f as bifurcation parameter, this is illustrated in Figure 3.4B,C. However, bistability in this G1/S transition depends on the assumption that Cdh1 remains accessible to CycA if bound to Apc and Emi1. Setting E2f to 0.7 AU_{E2f} (i.e. just below its upper steady state in the RP submodel) and using CycE as bifurcation parameter instead, Figure 3.4D shows that CycA can robustly maintain its high steady state independently of CycE. This is a critical feature of the G1/S toggle, since the rising concentration of CycA after flipping the toggle will lead to CycE phosphorylation when fusing the submodels, marking it for SCF mediated polyubiquitination. The time course at E2f close to the upper bifurcation point

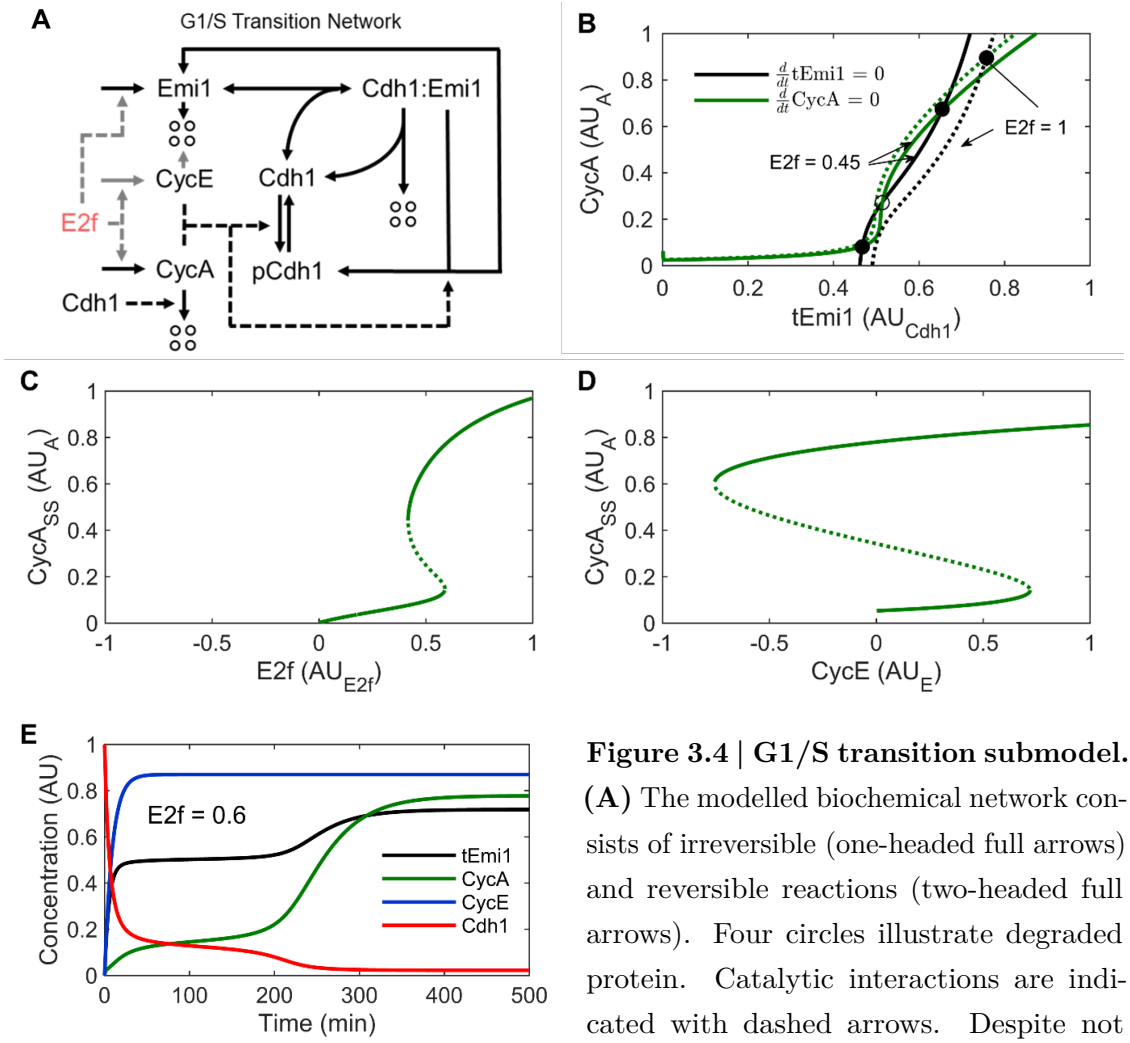


Figure 3.4 | G1/S transition submodel. **(A)** The modelled biochemical network consists of irreversible (one-headed full arrows) and reversible reactions (two-headed full arrows). Four circles illustrate degraded protein. Catalytic interactions are indicated with dashed arrows. Despite not shown here for reasons of clarity, total Apc is considered in excess [125] and all Cdh1 and Cdh1:Emi is considered to be bound to Apc. Lighter colour indicates reactions that are omitted in panel D. E2f (red) serves as bifurcation parameter in panels B and C. **(B)** Phase plane showing the nullclines for tEmi1 and CycA intersecting at two stable (filled circles) and one unstable (unfilled circle) steady states when E2f is set to 0.45 AU_{E2f}. There is only one steady state at E2f = 1 AU_{E2f}. The system was reduced to the two plotted dimensions by making steady state approximations for the remaining variables. **(C)** Bifurcation diagram of the non-reduced system with E2f as bifurcation parameter showing stable (solid line) and unstable (dotted line) steady states of CycA. **(D)** Bifurcation diagram of the non-reduced system at E2f = 0.7 AU_{E2f} with CycE as bifurcation parameter. **(E)** Time course of tEmi, CycA, CycE and Cdh1 at E2f = 0.6 AU_{E2f}. Cdh1 corresponds to (p)Apc:Cdh1; Cdh1:Emi corresponds to (p)Apc:Cdh1:Emi1. For the remaining variable abbreviations please refer to Table 3.1. Parameter values are available the /versions/v0.0.1/ directory of the cell_cycle_model GitHub repository.

from Figure 3.4C shows how CycA, CycE and Apc:Cdh1 approach the high CycA steady state (Figure 3.4E). The system almost comes to rest when the variables are close to the lost low CycA steady state. Rising CycA concentrations will facilitate cyclin B synthesis when merging the G1/S toggle with the G2/M toggle.

3.2.3 G2/M transition submodel

Cyclin B associates with Cdk1, which is kept inactive via Wee1 kinase mediated phosphorylation. Wee1 activity is counteracted by the phosphatase Cdc25. As Cdc25 is activated and Wee1 inactivated by CycB:Cdk1, this kinase phosphatase system contains two positive feedback loops. Modelling the phosphorylation and/or dephosphorylation of Cyclin B:Cdk1 with Michaelis-Menten functions as ultrasensitive Goldbeter-Koshland switch, the system can become bistable with the steady state depending on tCycB. However, recent experimental evidence suggests that the bistability of the G2/M transition network (Figure 3.5A) depends on the PP2A:B55/Ensa/Greatwall pathway [127, 128]. Consistent with this observation Vinod and Novak [129] published an ODE model for the G2/M transition, using a set of experimentally determined kinetic parameters [112]. This section will reproduce the results from Vinod and Novak. The model describes how CycB:Cdk1 phosphorylates/activates Greatwall, which in turn phosphorylates/activates Ensa. In what is often referred to as “unfair competition”, pEnsa tightly binds to the B55 subunit of protein phosphatase 2A, but is only slowly dephosphorylated. Thus, phosphorylated Ensa acts as stoichiometric inhibitor of B55, titrating it away from its other substrates pGw, phosphorylated Wee1 and pCdc25. This G2/M transition reaction network is described by Equations (3.23)–(3.29) (in the full model (Section 3.2.5) Equations (3.28) and (3.29) describe reaction rates that where rewritten as system of ODEs using mass action kinetics). The following

parameters were taken from [112] and rescaled to the following values and units: $t_{Ensa} = 1$ $\mathrm{AU_{Ensa}}$, $t_{B55} = 0.25$ $\mathrm{AU_{Ensa}}$, $k_{DipEB55} = 0.0068$ $\mathrm{min^{-1}}$, $k_{AspEB55} = 57$ $\mathrm{min^{-1}AU_{Ensa}^{-1}}$ and $k_{DpEnsa} = 0.05$ $\mathrm{min^{-1}}$.

$$\frac{dpEnsa}{dt} = k_{PhEnsa} \cdot pGw \cdot (t_{Ensa} - pEnsa) - k_{DpEnsa} \cdot pEnsa : B55 \quad (3.23)$$

$$\frac{dCycB{:}Cdk1}{dt} = r_{Cdc25} \cdot (t_{CycB} - CycB{:}Cdk1) - r_{Wee} \cdot CycB{:}Cdk1 \quad (3.24)$$

$$\begin{aligned} \frac{dpGw}{dt} &= k_{PhGw} \cdot CycB{:}Cdk1 \cdot (t_{Gw} - pGw) \\ &\quad - (k_{DpGw1} + k_{DpGw2} \cdot B55) \cdot pGw \end{aligned} \quad (3.25)$$

$$\begin{aligned} \frac{dpEnsa{:}B55}{dt} &= k_{AspEB55} \cdot B55 \cdot (pEnsa - pEnsa{:}B55) \\ &\quad - (k_{DipEB55} + k_{DpEnsa}) \cdot pEnsa{:}B55 \end{aligned} \quad (3.26)$$

$$B55 = t_{B55} - pEnsa{:}B55 \quad (3.27)$$

$$r_{Wee} = k_{Wee1} + \frac{(k_{Wee2} - k_{Wee1}) \cdot k_{DpWee} \cdot B55}{k_{DpWee} \cdot B55 + k_{PhWee} \cdot CycB{:}Cdk1} \quad (3.28)$$

$$r_{Cdc25} = k_{Cdc25\_1} + \frac{(k_{Cdc25\_2} - k_{Cdc25\_1}) \cdot k_{PhCdc25} \cdot CycB{:}Cdk1}{k_{PhCdc25} \cdot CycB{:}Cdk1 + k_{DpCdc25} \cdot B55} \quad (3.29)$$

While the inhibitor ultrasensitivity conferred by phosphorylated Ensa does not allow for bistability if only combined with the mutual inhibition between B55 and phosphorylated Ensa, robust bistable behaviour can be observed if combined with the mutual inhibition between B55 and pGw, and/or B55 and CycB:Cdk1 (Figure 3.5B,C) [129]. One critical property of the network is that the bifurcation parameter tCycB is an upper boundary of the system variable CycB:Cdk1 but not of B55. When reducing tCycB below the upper bifurcation point B55 remains bound to phosphorylated Ensa, while CycB:Cdk1 almost linearly decreases with tCycB. As B55 inactivity can stabilise its inactive state almost independently of CycB:Cdk1, it is possible to reduce tCycB to very low levels without activating B55 (Figure 3.5C). This feature will be exploited to reset the high tCycB at the G2/M transition to basal levels via the negative feedback loop represented by the M/A transition

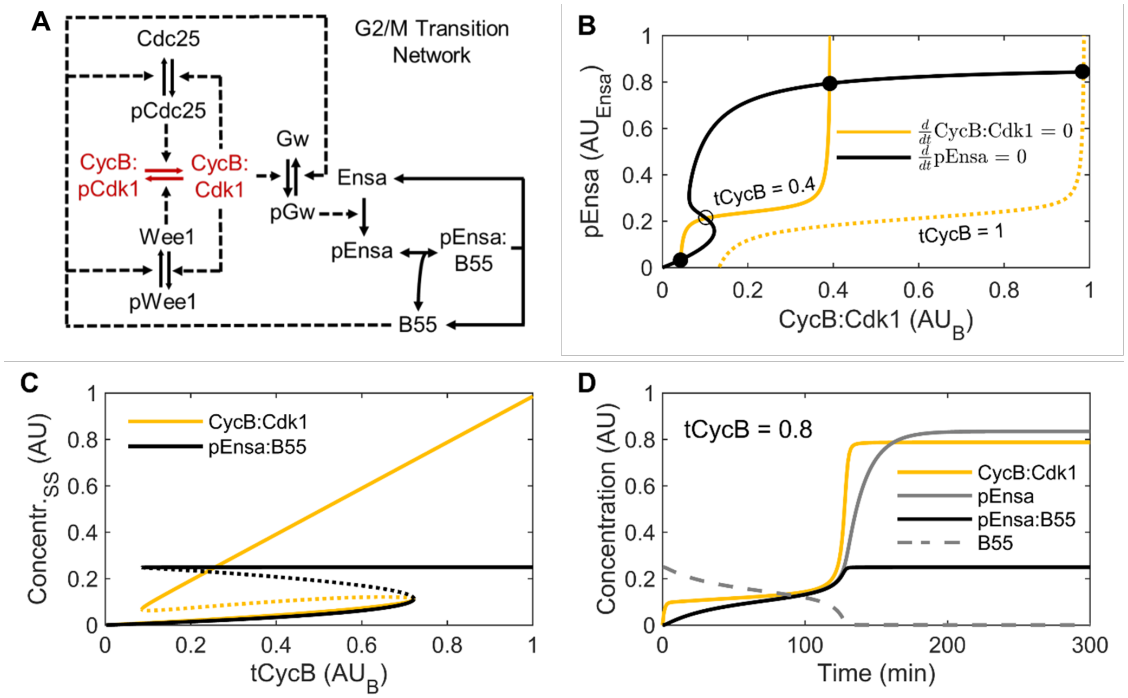


Figure 3.5 | G2/M transition submodel as developed by Vinod and Novak [129]. **(A)** The modelled biochemical network consists of irreversible (one-headed full arrows) and reversible reactions (two-headed full arrows). Four circles illustrate degraded protein. Catalytic interactions are indicated with dashed arrows. tCycB (red) consisting of CycB:Cdk1 and its phosphorylated form CycB:pCdk1 serves as bifurcation parameter in panels B and C. **(B)** Phase plane showing the nullclines for CycB:Cdk1 and pEnsa intersecting at two stable (filled circle) and one unstable (unfilled circle) steady states when tCycB is set to 0.4 AU_B. There is only one steady state at tCycB = 1 AU_B. The system was reduced to the two plotted dimensions by making steady state approximations for the remaining variables. Plot recreated from [129]. **(C)** Bifurcation diagram of the non-reduced system showing stable (solid line) and unstable (dotted line) steady state concentrations of CycB:Cdk1 and pEnsa:B55. **(D)** Time course of CycB:Cdk1, pEnsa, pEnsa:B55, and B55 at tCycB = 0.8 AU_B. pWee1: phosphorylated Wee1; Cdc25: unphosphorylated Cdc25, Gw: unphosphorylated Greatwall kinase; Ensa: unphosphorylated Endosulfine alpha. For the remaining variable abbreviations please refer to Table 3.1. Parameter values are available from the /versions/v0.0.1/ directory of the cell_cycle_model GitHub repository and relate to experimentally determined values by Williams *et al.* [112] as described in the main text.

submodel. The time course at tCycB close to the upper bifurcation point from Figure 3.5C shows how CycB:Cdk1, pEnsa, pEnsa:B55 and B55 approach the high

CycB:Cdk1 steady state (Figure 3.5D). The system almost comes to rest when the variables are close to the lost low CycB:Cdk1 steady state.

3.2.4 M/A transition submodel

The major hallmarks of the M/A transition are chromosome segregation and cyclin B degradation. Both processes are mediated by pApc:Cdc20, which is kept inactive via association to mitotic checkpoint proteins until all chromosomes are correctly attached to the microtubule spindle apparatus. The phosphorylation of Apc requires a shift in the ratio between Apc kinase and phosphatase activity. While it is well established that CycB:Cdk1 serves as Apc kinase [103], this mechanism cannot be exploited to degrade cyclin B back to baseline levels, if combined with the G2/M transition submodel and the corresponding experimentally determined parameters. This is because CycB:Cdk1 at the upper bifurcation point is higher than CycB:Cdk1 at the lower bifurcation point (Figure 3.5C). Due to a negative feedback between tCycB and pApc:Cdc20, CycB:Cdk1 would either become too low to promote further cyclin B degradation when moving down the upper branch of the steady state, or CycB:Cdk1 would become too high to allow further tCycB accumulation when moving up the lower branch of the steady state (Figure 3.5C). This problem could be solved by introducing a spindle assembly checkpoint that keeps pApc:Cdc20 inactive during the process of cyclin B:Cdk1 activation/phosphorylation, thus decoupling cyclin B:Cdk1 activation/phosphorylation from cyclin B degradation. However, introducing a spindle assembly checkpoint is beyond the scope of the present model, leaving the option of phosphatase mediated regulation of the Apc phosphorylation state. While little is known about phosphatases acting on Apc, it has been put forth that B55 is one of the phosphatases for Cdk1 substrates [130]. In contrast to CycB:Cdk1, pEnsa:B55 at the upper bifurcation point is much lower

than the pEnsa:B55 at the lower bifurcation point (Figure 3.5C), allowing to couple the bistable G2/M switch with B55 driven negative feedback (Figure 3.6A). This negative feedback is described by Equations (3.30)–(3.33).

$$\begin{aligned}\frac{dpApc}{dt} &= k_{PhApc} \cdot (t_{Apc} - pApc) \cdot (CycB : Cdk1) \\ &\quad - k_{DpApc} \cdot B55 \cdot pApc\end{aligned} \tag{3.30}$$

$$\begin{aligned}\frac{dpApc{:}Cdc20}{dt} &= k_{AsAC20} \cdot (t_{Cdc20} - pApc{:}Cdc20) \cdot (pApc - pApc{:}Cdc20) \\ &\quad - (k_{DiAC20} + k_{DpApc} \cdot B55) \cdot pApc{:}Cdc20\end{aligned} \tag{3.31}$$

$$\frac{dtCycB}{dt} = k_{SyCb} - (k_{DeCb1} + k_{DeCb2} \cdot pApc{:}Cdc20) \cdot t_{CycB} \tag{3.32}$$

$$\begin{aligned}\frac{dCycB{:}Cdk1}{dt} &= k_{SyCb} - (k_{DeCb1} + k_{DeCb2} \cdot pApc{:}Cdc) \cdot CycB{:}Cdk1 \\ &\quad + r_{Cdc25} \cdot (t_{CycB} - CycB{:}Cdk1) - r_{Wee} \cdot CycB{:}Cdk1\end{aligned} \tag{3.33}$$

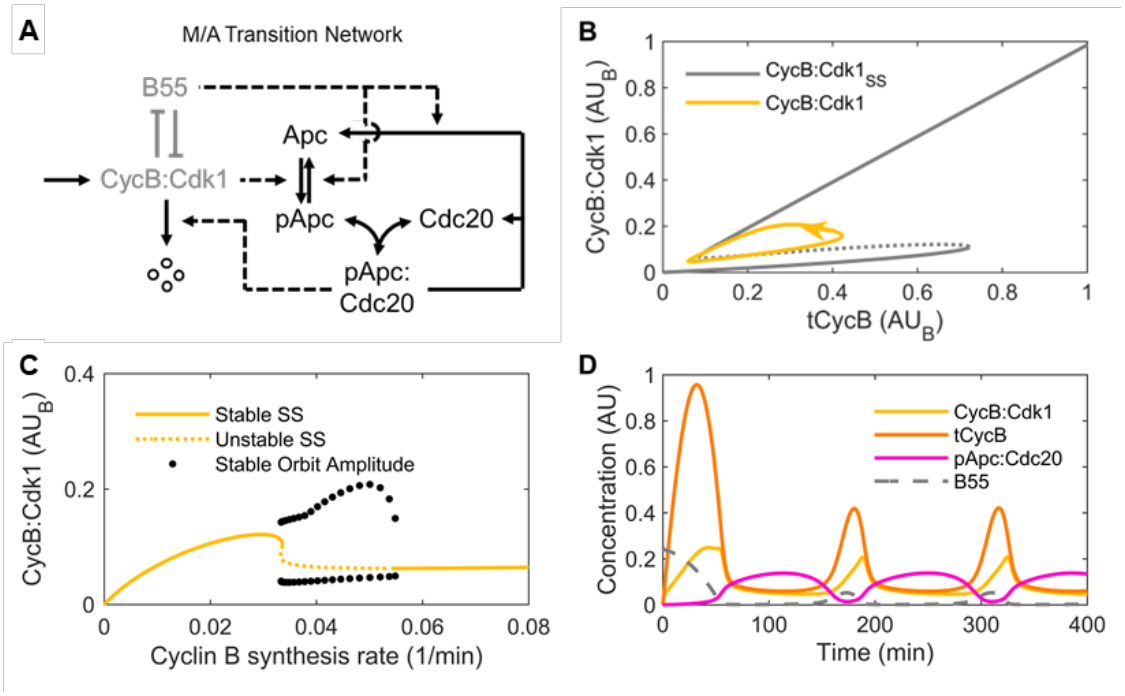


Figure 3.6 | G2/M transition submodel with negative feedback of the M/A transition. **(A)** The modelled biochemical network of the M/A transition consists of irreversible (one-headed full arrows) and reversible reactions (two-headed full arrows). Four circles illustrate degraded protein. Catalytic interactions are indicated with dashed arrows. Lighter colour represents a simplified schematic of the G2/M transition network shown in Figure 3.5A. **(B)** CycB:Cdk1 steady states (grey) of the G2/M transition submodel shown in the bifurcation diagram of Figure 3.5C are overlaid with a stable limit cycle trajectory (yellow) of the system in the CycB:Cdk1 – tCycB subspace at a cyclin B synthesis rate of kSyCb = $0.05\,\mathrm{min}^{-1}$. The periodic over- and undershooting of the upper and lower bifurcation point, respectively, results from combining the toggle switch of the G2/M submodel with the negative feedback of the M/A submodel (see main text for a detailed explanation). **(C)** Bifurcation diagram of the G2/M–M/A submodel showing stable and unstable steady states as well as lower and upper boundaries for the amplitude of stable orbits. **(D)** Time course of CycB:Cdk1, tCycB, pApc:Cdc20 and B55 at kSyCb = $0.05\,\mathrm{min}^{-1}$. For variable abbreviations please refer to Table 3.1. Parameter values are available the /versions/v0.0.1/ directory of the cell_cycle_model GitHub repository.

Combining the negative feedback from Equations (3.30)–(3.33) with the G2/M switch from Equations (3.23)–(3.29) results in limit cycle oscillations (Figure 3.6B–D) in a similar way to what has already been shown by Ferrell [131]. However, these oscillations disappear when cyclin B synthesis decreases below $0.033\,35\,\mathrm{min}^{-1}$

(Figure 3.6C). This feature will be exploited in the full cell cycle model, where cyclin B synthesis will be reduced at the M/A transition to keep its concentration low in G1 phase.

3.2.5 The core cell cycle model

While the submodels depicted above describe the dynamics of different sets of species largely in isolation, in a real biological cell many of these species interact with each other. To account for this circumstance when fusing the submodels to a full cell cycle model, we had to introduce (a) new reactions (e.g.: CycA mediated Rb phosphorylation; Apc:Cdh1 mediated CycB degradation) and (b) new species (e.g.: pApc:Cdh). To connect the G1/S submodel with the G2/M submodel, we also introduced the transcription factor FOXM1. FOXM1 is synthesized by E2f and activated by Cdks [132]. It binds to the promoter of cyclin B to promote its synthesis [133]. Furthermore, certain molecules that had accumulated during earlier cell cycle stages (tE2F, tEmi1, CycA, tFoxM1 tCdc20) still had to be degraded. For the SCF substrates tE2f [134] and tEmi1 [135] (but for simplicity not for CycE and CycA) this was accomplished by introducing a new, phosphorylated species that was targeted for ubiquitination. These species are phosphorylated by CycA and CycB, thus introducing additional negative feedback loops on E2F (e.g.: E2F activates CycA, which inhibits E2F; and E2F activates CycB via CycA mediated FOXM1 phosphorylation, and CycB inhibits E2F) and on Emi1 (e.g.: Emi1 inhibits Apc:Cdh1 inhibits CycA/B inhibits Emi1). For the (p)Apc:Cdh1 substrates tFoxM1 [136] and tCdc20 [137] a (p)Apc:Cdh1 mediated degradation reaction was introduced. This again results in new feedback loops. For instance, FoxM1 activates CycB. CycB and (p)Apc:Cdh1 mutually inhibit each other. (p)Apc:Cdh1 then inhibits FoxM1. The (p)Apc:Cdh1 mediated degradation of tFoxM1 ensures that tCycB is kept low

in G1 phase. Similarly, for CycA/B a pApc:Cdc20 mediated degradation reaction was introduced, resulting in yet another negative feedback loop (CycA inhibits (p)Apc:Cdh1, inhibits pApc:Cdc20, inhibits CycA). These changes also required modifications in the parameters to preserve characteristic features of each submodel:

- RP: The restriction point is a point after which cell cycle progression is mitogen independent.
- G1/S: There is a delay between CycE and CycA accumulation.
- G2/M: There is a sharp increase in CycB:Cdk1.
- M/A: Cyclin B is rapidly and almost completely degraded.

To reduce the complexity of this task, the submodels were fused in a stepwise manner (Figure 3.7A–C). The resulting model of the whole cell cycle (cell_cycle_v1.0.0) comprised 37 species (Table 3.1). The corresponding equations are available in file /versions/v1.0.0/cell_cycle_v1.0.0.ode of the cell_cycle_model GitHub repository. To a large extent, the time course of the modelled species (Figure 3.7C,D) qualitatively resembles the data in the literature [106, 114, 138] (see also Figure 3.1). However, the delay between CycA and CycB accumulation is reduced in the model. The reduced delay between CycB:Cdk1 accumulation and tCycB decline results from ignoring the random event of chromosome attachment to the microtubule spindle in our deterministic model. Similarly, the transition from pApc:Cdc20 to (p)Apc:Cdh1 is less sharp than in cells with intact spindle assembly checkpoint [139]. The doubling time of approximately 500 min corresponds to very fast dividing human cells [140]. To test whether the full cell cycle model still has a restriction point after which cell cycle progression becomes independent of mitogen signalling, we paused the simulation at 610 min and 620 min, respectively to reduce

CycD as proxy for mitogen availability from 1 to 0 AU_D (Figure 3.7E,F). After continuing the simulation until 1500 min, we observed that cell cycle progression was halted if CycD was depleted at 610 min. However, consistent with the presence of a restriction point between 610 and 620 min after simulation start, the current round of the cell cycle was finished if CycD was depleted at 620 min. While this explains the response to mitogen depletion very well, CycD is not an ideal proxy of mitogen depletion. Its transcription is activated via the transcription factor Myc, which is activated by mitogen signalling. Myc not only activates CycD, but also inhibits anti-proliferative genes and activates several other pro-proliferative genes [141–143]. Therefore, contrary to what the simulation with reduced CycD levels suggest, CycD knockout cells can proliferate [144]. Finally, we wanted to test if our model can qualitatively capture observations from knockout experiments. Specifically, we wanted to test if (a) CycE knockout cells can proliferate with elevated doubling time, as it has been demonstrated *in vivo* in mice [145, 146] and (b) CycA depletion leads to CycE accumulation without disrupting cell proliferation [147]. Using the same parameter values as for the simulation of Figure 3.7C,D, and without losing CycD dependence of continued proliferation, Figure 3.7G,H shows that our model could reproduce both features.

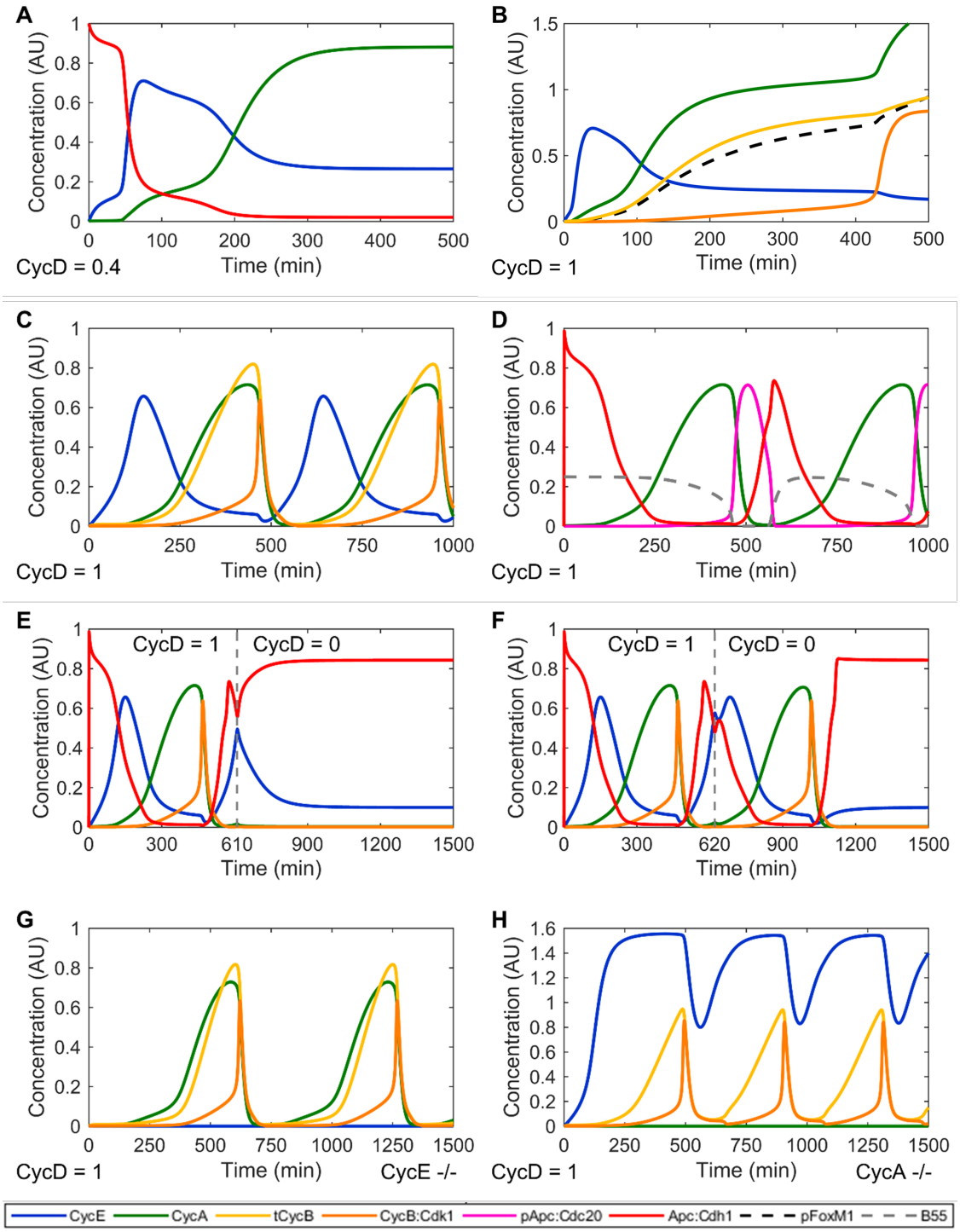


Figure 3.7 | Time courses of the merged cell cycle models. **(A)** Combined RP-G1/S model at CycD = 0.4 AU_D. **(B)** Combined RP–G1S–G2M model at CycD = 1 AU_D. **(C-D)** Full cell cycle model (RP–G1/S–G2/M–MA) at CycD = 1 AU_D. **(E-F)** Simulation of mitogen deprivation. The full cell cycle model was simulated as in (C-D) until 610 min (E) and 620 min (F), respectively. CycD as proxy for mitogen availability was then turned to 0 AU_D and the simulation was continued until 1500 min. **(G)** Full cell cycle model with CycE knockout. **(H)** Full cell cycle model with CycA knockout. For variable abbreviations please refer to Table 3.1. Parameter values are available the /versions/v0.0.1/ directory of the cell_cycle_model GitHub repository.

3.2.6 Conversion to a rule-based format and introducing a systematic naming convention

As indicated in Table 3.2, the original core model was continuously refined through versions 1.0.0–2.1.4 with only very minor effect on the simulation result (Figure 3.8A). Version 1.0.0 is available as ODE-based ODE file (for simulation with XPP/XPPAUT [148]) and as reaction-based CPS file (for simulation with COPASI [32]). With the major version bump to version 2.0.0, XPP/XPPAUT support was dropped. Versions 2.0.0–2.1.4 are available in reaction-based CPS or SBML descriptions, from which ODEs can be generated using any SBML compliant software (see the cell_cycle_model GitHub repository. However, the complexity of the biochemical reaction network described by the model impeded major model extensions. To facilitate extensions, the model was converted to a more abstract, rule-based description in the BioNetGen Language (BNGL) [149]. The introduction of support for rule-based descriptions is indicated with another major version bump to version 3.0.0. This rule-based description can be automatically expanded to reaction-networks and thus also to ODEs. Moreover, the BNGL syntax also lends itself well to add semantic meaning to the model variables. This enables to conceive the following naming convention:

- Proteins
  - Protein molecule types were named by there Human Genome Organization Gene Nomenclature Committee (HGNC) short name. No distinction is made between protein isoforms that are transcribed from the same gene. Example: `RB1()`.
  - Protein family types were described by a shortened protein HGNC name where unambiguously possible. Example: the set of CCNE1 and CCNE2

was collectively referred to as `CCNE()`. Otherwise, they were described as the concatenation of HGNC names separated by underscores. Example: `ENSA_ARPP19()`.

- Promoters
  - Promoters where described by the protein they activate, suffixed by `_promoter`. Example: `CDC20_promoter()`.
- Binding sites
  - If the binding site is bound by a protein or protein family, the binding site is named by the protein (family) name. Example: `PPP2R2B(ENSA_ARPP19)`.
  - If the binding site is bound by a promoter, the binding site is named `DBD` (short for DNA binding domain). Example: `FOXM1(DBD)`.
- Phosphorylation sites
  - The unphosphorylated state is indicated with a `u`, the phosphorylated state with a `p`.
  - Single site phosphorylation were named by the three letter amino acid code, suffixed with the amino acid position. Example: `MASTL(Thr198~u~p)`.
  - Known multisite phosphorylations were named by the concatenation of the single site phosphorylation names, separated with an underscore. Example `RB1(Ser807_Ser811~u~p)`. Note that no comma separates `Ser807` and `Ser811` in this example. Therefore, `Ser807_Ser811` is treated as a single site.
  - Unknown phosphorylation sites could have any name.
    Example `FZR1(APC,FBXO5,nTerm~u~p)`.

- Exceptions
  - `E2F()` means only the activating E2F transcription factors E2F1, E2F2 and E2F3.
  - The anaphase-promoting complex was named `APC()`.
  - The nuclear pore complex was named `NUP()`.

The exceptions indicate that some generalisability was traded off against concise syntax. Of note, as much as any biochemical model is incomplete, the use of the above syntax shall not imply that no other phosphorylation sites, binding sites or molecule types than those explicitly described by the model are involved in the real phenomenon of cell cycle control. Rather, it shall just help to describe more precisely which biological phenomena are considered in the model and which are not.

After conversion to a rule-based format (version 3.0.0) equivalency with version 2.1.4 was confirmed by comparing time course plots of the simulation for multiple variables. Figure 3.8B shows the comparison for `CCNE()`, `CCNA()` and `CCNB(CDKN1A,CDK1_Thr14_Tyr15~u)`. The rule-based format facilitated minor model refinements leading to version 3.0.1, the starting point for implementing an interface to DNA damage response pathways.

3.2.7 Implementing a DNA damage checkpoint

DNA damage response is in itself a complicated biological process. The main point of interaction with the cell cycle is through inhibition of cyclins via binding of unphosphorylated CDKN1A. CDKN1A, also known as p21 is a cyclin kinase inhibitor that is phosphorylated by Cdks. Its transcription is activated via TP53, a transcription factor that is activated by DNA damage. Polyubiquitination by the SCF complex marks CDKN1A for degradation. The SCF complex contains a so-called F-box

Table 3.2 | Changelog of model versions.

Version	Changes
1.0.0	-
2.0.0	- Rescaled cell cycle length to 20 h (H1 human embryonic stem cells). - Hard coded/eliminated parameters f1-f4 that were once introduced to describe loss of enzymatic access if target is in complex. - Renamed species to avoid ambiguity in case-insensitive software. - Ran version 1.0.0 for 9950 time units and used this state as initial conditions in version 2.0.0. - Deleted observables.
2.1.0	- Removed the simplifying assumption from the ODE model that (un)binding of transcription factor to promoter does not affect the concentration of free transcription factor.
2.1.1	- Created separate promoters for Ce, Ca, E2f, Emi and Fox from one E2f activated promoter (formerly called Px).
2.1.2	- Introduced observables for parameter optimisation purposes (were never used in the end and specified via PEtab instead).
2.1.3	- Introduced interfaces for small molecule mediated inhibition of Ce and transcriptional/translational inhibition for Cb (were never used in the end).
2.1.4	- Introduced interface for small molecule mediated inhibition of Wee1 and removed it for Ce.
3.0.0	- Converted model into BioNetGen language. - Introduced a systematic naming convention.
3.0.1	- Make E2F binding to DNA and its transcriptional activity independent of E2F phosphorylation state. - Introduce baseline APC dephosphorylation. - Allow dephosphorylation of DNA-bound FOXM1.
3.1.0	- Added DNA damage checkpoint (SKP2 and TP53 and CDKN1A).
3.2.0	- Added CDKN1B.
4.0.0	- Added compartmentalisation.

protein, which confers substrate specificity. CDKN1A is polyubiquitinated by SCF with SKP2 as F-box protein. Noteworthy, the mutual inhibition of CDKN1A and Cdks, combined with the ultrasensitivity conferring stoichiometric binding, results in bistability of the DNA damage checkpoint [150]. After adding CDKN1A, TP53, SKP2 and the rules coarsely described in words above, TP53 provides an interface for activating a DNA damage response checkpoint (version 3.1.0).

It has been shown that ultraviolet radiation induced DNA damage results in a TP53 pulse, while ionizing radiation results in TP53 oscillations [151]. If cells do

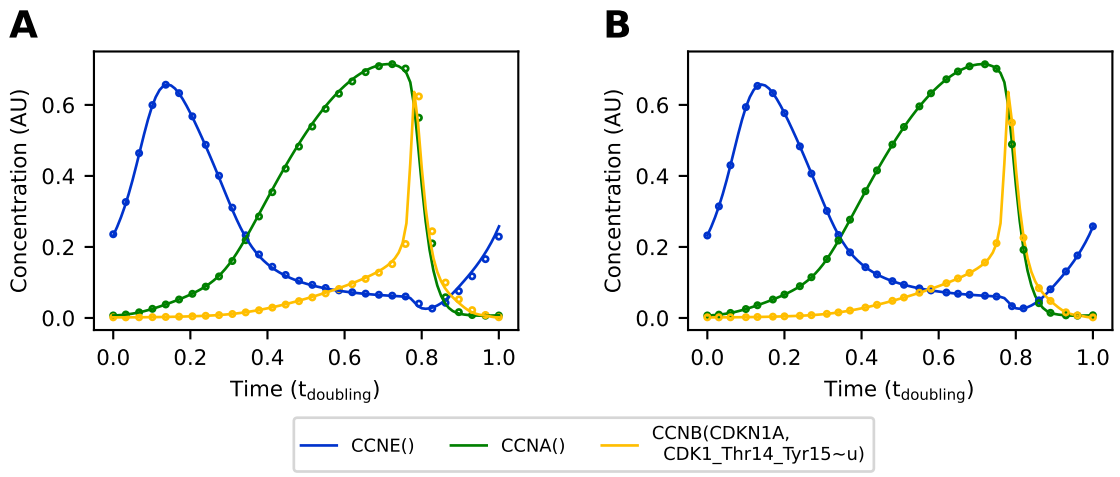


Figure 3.8 | Comparison between model versions. All simulations were performed with Copasi LSODA and default settings. Lines represent simulations of model version 2.1.4. **(A)** Circles represent second cycle of model version 1.0.0. First cycle is not shown, as the initial conditions of this model do not lie on the limit cycle trajectory. **(B)** Circles represent model version 3.0.0, simulated after export to SBML.

not succeed in repairing the DNA damage, they switch from TP53 oscillations to permanently elevated TP53 levels, which ultimately trigger apoptosis. Tsabar *et. al* [152] have developed a delay differential equation model that explains this dynamic in the DNA damage response pathway. However, the DNA damage response pathway, as well as the caspase cascades triggering apoptosis, shall be considered outside the system boundaries of the cell cycle model. For simplicity, figure 3.9 considers TP53 as either in high or low abundance. High abundance in G1 or G2 phase triggers a DNA damage checkpoint, while low abundance leads to continued cell cycle progression. During DNA damage checkpoint activation in G1, cyclin E levels rise higher than they would in proliferating cells. This is in agreement with experimental observations in RPE1 cells [153] and can be explained by suppression of cyclin A, which would otherwise target cyclin E for degradation via phosphorylation.

Like CDKN1A, the CDKN1B cycling kinase inhibitor, also known as p27, confers ultrasensitivity. Together, the RP, the G1/S transition, the DNA damage checkpoint,

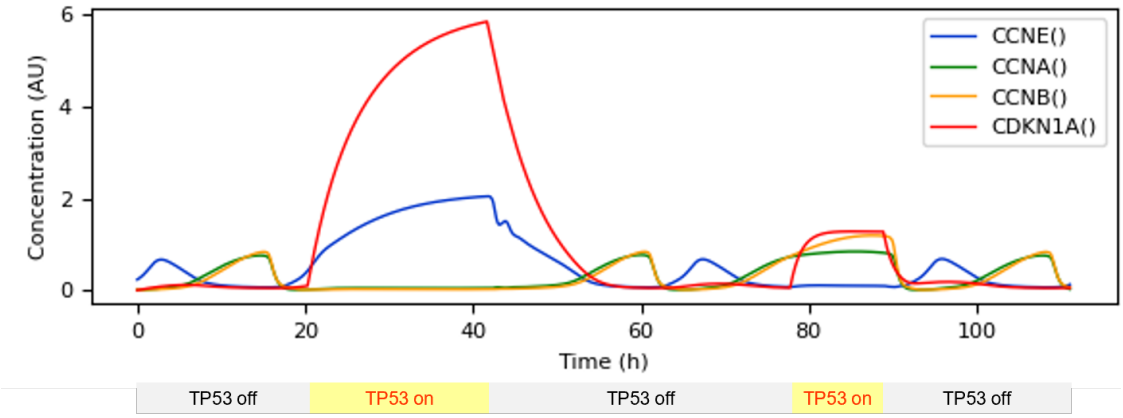


Figure 3.9 | Time courses with alternating TP53 levels. DNA damage was simulated by activating TP53. The first activation corresponds to G1 phase, the second to G2 phase. Inactivation of TP53 allows continued proliferation. For better visualisation the G1 checkpoint was lifted before the steady state was reached. Model version 3.1.0.

and the mutual inhibition between CDKN1B and cyclin:Cdk complexes represent four partially redundant mechanisms to ensure bistability in G1. Redundancy confers robustness in varying or perturbed conditions. As robustness under perturbation seems to be an important feature of the cell cycle, CDKN1B was added to the model, along with FOXO transcription factors which activate its expression [154]. Unlike CDKN1A, CDKN1B does not bind cyclin B:Cdk complexes [155]. There is also clear evidence that Thr187 phosphorylation not only prevents cyclin:Cdk binding, but also targets CDKN1B for SKP2 mediated degradation. Figure 3.10 shows how Cdk activity leads to CDKN1B degradation, while synthesis remains constant in the model.

3.2.8 Introducing the notion of compartmentalisation

The model presented so far assumes uniform spatial distribution of the cell cycle regulators. However, there is substantial evidence of nucleocytoplasmic shuttling of several cell cycle regulators [156–159], including evidence that such translocation

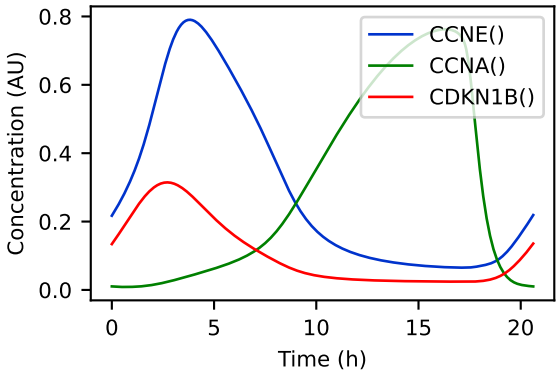


Figure 3.10 | Time course of CDKN1B. Model version 3.2.0.

plays important roles in cell cycle regulation, for instance as a source of bistability [160]. Such nucleocytoplasmic shuttling through nuclear pores is carried out by the Ran-GTP cycle, which depends on nuclear export and import signals on the transported proteins, which can be (de)activated by posttranslational modifications [161].

To enable more accurate representations of cell cycle control, the capability for representing nucleocytoplasmic shuttling of cell cycle regulators had to be implemented into the model. This capability enables parameter optimisation algorithms to tune import/export rate constants. If identifiable, these rate constants may point towards translocation mechanisms of hard-to-observe species, such as complexes and certain phosphoproteins. To implement nucleocytoplasmic shuttling, the following points were considered:

- The nucleocytoplasmic transport reactions described in the literature represent only a subset of the nucleocytoplasmic transport reactions that are happening in the cell.
- The objectives of this chapter (Section 3.1.4), namely:
  - identifying an extensive biochemical reaction network that is capable of explaining how the full cell cycle is regulated, without finding an accurate parameterisation.

– Demonstrating that the model can generate sustained oscillations.

In agreement with these considerations, translocation reactions for all molecular species, except for those that are exclusively located in either compartment (i.e. promoters), were added to the model. The kinetic constants for the transport reactions were set to very large and equal values (to be later constrained during parameter estimation). This ensured that the oscillatory behaviour of single-compartment versions of the model was retained. More specifically, the following adaptations were made to version 3.2.0:

- Four compartments were defined using BNGL: a $1 \times 10^{-7}\,\mathrm{dm}$ thick plasma membrane (`Plm`, $2 \times 10^{-14}\,\mathrm{dm}^3$), cytoplasm (`Cyt`, $1 \times 10^{-12}\,\mathrm{dm}^3$), a $1 \times 10^{-7}\,\mathrm{dm}$ thick nuclear membrane (`Num`, $9 \times 10^{-15}\,\mathrm{dm}^3$) and a nucleus (`Nuc`, $1 \times 10^{-12}\,\mathrm{dm}^3$).

- Nuclear pores `NUP(pSites~u~p)` were inserted into the nuclear membrane.

- Instead of describing the complete Ran-GTP pathway, simple import and export reactions through unphosphorylated nuclear pores were added for all species that do not contain promoters. The kinetic constants for import and export were set to same very large value to allow quasi-free diffusion of molecules.

- The initial concentration of promoters was set to zero in the cytoplasm and a positive value in the nucleus.

- Reaction rules were made independent of localisation. I.e. reactions are unaware of the compartment they occur in, use the same kinetic constants independent of the compartment, and fire at any location where all reactants

are present. Only protein synthesis reactions contain a notion of localisation, namely that proteins are synthesized into the cytoplasm.

- Nuclear envelope breakdown was imitated by introducing a rule for `CCNB(CDKN1A,CDK1_Thr14_Tyr15~u)` mediated nuclear pore phosphorylation, allowing for quasi-free diffusion of all pure protein species.

The above changes lead to model version `4.0.0`. As some previously published mammalian cell cycle models were not able to produce sustained oscillations, the model was simulated for nearly 2758 h, corresponding to 134 cell cycles to confirm sustained oscillations (Figure 3.11A). Nuclear envelope breakdown was switched off for this simulation, as it requires low simulation error tolerances, leading to several hours simulation time per cell cycle. One such cycle with nuclear envelope breakdown is shown in Figure 3.11B.

3.3 Conclusion

This chapter presented several versions of deterministic ODE and rule-based models that provide mechanistic insights into the reaction network that regulates the human cell cycle. The largest rule-based model (version `4.0.0`) consists of 123 species, 630 reactions and 327 parameters after expansion to a reaction network. It implements the notion of compartmentalisation and produces sustained oscillations. The models explain the experimental observations of bistability in the RP, G1/S and G2/M transition as an emergent property of mass action kinetics. They qualitatively agree with literature data [5, 106, 138] on time courses of several modelled species. When simulating mitogen depletion, CycE and CycA knockouts, we reproduced the notion of a restriction point and experimental findings of continued cell proliferation

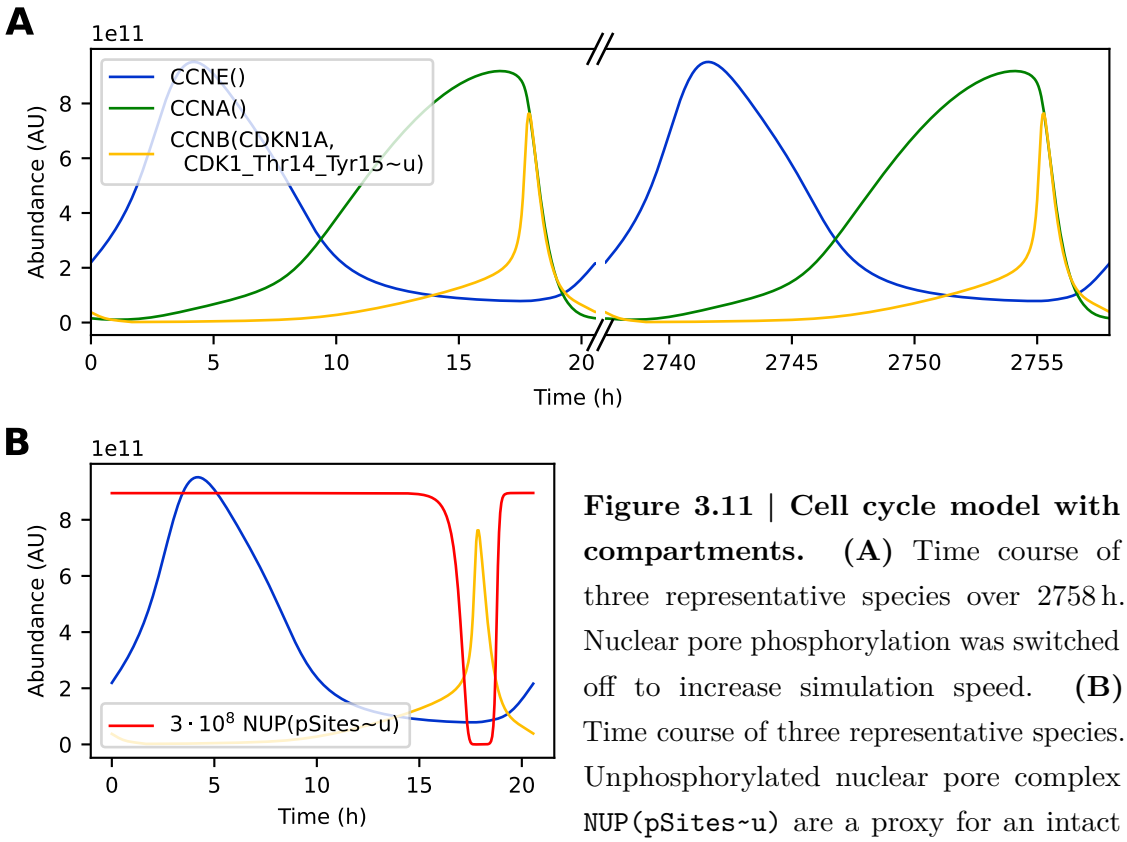


Figure 3.11 | Cell cycle model with compartments. **(A)** Time course of three representative species over 2758 h. Nuclear pore phosphorylation was switched off to increase simulation speed. **(B)** Time course of three representative species. Unphosphorylated nuclear pore complex `NUP(pSites~u)` are a proxy for an intact nuclear membrane. Both panels show simulations of model version 4.0.0.

in the corresponding mutants [145–147]. Similarly, simulations of DNA damage reproduced experimentally observed observations of elevated CycE levels [153].

However, the models contain a large number of parameters that were not experimentally determined. These parameters were chosen manually within biologically reasonable boundaries to roughly agree with a limited set of time course data. Therefore, a more systematic approach to parameter optimisation is still needed and will be described in Chapter 5. Moreover, the development of the model is based on several simplifying assumptions (see also the Section 3.4.3 in the Methods). Also, whether pApc is a substrate of B55, and how the association of a specific substrate with other proteins affects its accessibility by kinases or phosphatases needs to be subject of future experimental investigations. As observed in the G1/S transition, this property can qualitatively affect the behaviour of the whole system.

Further work is required to refine the present model. Especially, incorporating biochemical networks for cell size control and spindle assembly checkpoint will diminish inconsistencies with findings in the literature. Additionally, precise time course data of a large number of cell cycle regulators under unperturbed and perturbed conditions will be needed for estimating the model parameters.

3.4 Methods

3.4.1 Mathematical modeling

3.4.1.1 Core model

The core model and the transition models are formulated as systems of ODEs. Nullcline plots, bifurcation diagrams and time courses where calculated with the freely available software XPP/XPPAUT [148]. For steady state bifurcation analysis the bifurcation parameter was set to zero and XPP/XPPAUT was started from the corresponding steady state. This steady state bifurcation analysis also detects Hopf bifurcations. Periodic bifurcation analysis (Figure 3.6C) was started from the Hopf bifurcation at $k_{SyCb} \approx 0.055$ min^{-1}. The biochemical reactions making up the RP, G1/S and G2M submodels were chosen to comply with experimental observations of bistability [59, 111, 114–116] in these networks. These submodels where merged in a stepwise manner to yield an RP-G1S, RP-G1S-G2M, G2M-MA and finally a full cell cycle model (RP-G1S-G2M-MA). Additional reactions and species where added whenever necessary and to make the model more comprehensive. Reaction parameters where chosen based on Williams *et al.* [112], where applicable. For the remaining parameters an attempt was made to fulfil the following criteria: (a) Parameters shall be biologically reasonable compared to the parameters obtained by Williams *et al.*. (b) Parameters in the RP, G1/S and G2M submodels shall allow

for bistability. (c) After merging the submodels to the full model, parameters were adjusted to obtain sustained oscillations as observed in unperturbed cells, CycE [145] and CycA [147] knockouts to obtain model version 1.0.0.

3.4.1.2 Including DNA damage response and compartmentalisation

Prior to interfacing with DNA damage response regulator and compartmentalisation, the model underwent rescaling, minor changes and conversion to BioNetGen Language version 2.5.2 [149] using RuleBender version 2.3.1 [43] (Table 3.2, versions 1.0.0–3.0.0). The precise changes after model version 3.0.0, including the implementation of DNA damage response and compartmentalisation were documented and tracked with Git version control and made available on the cell_cycle_model GitHub repository.

3.4.2 Model verification

All relevant changes to the reaction network and all conversions between model description formats were subject to model verification. Version 1.0.0 was checked by rewriting the ODEs in a chemical reaction-based format using the freely available software COPASI [32]. The chemical reactions were automatically converted to a system of ODEs that were used for manual proofreading of the XPP/XPPAUT code. Time course simulations of XPP/XPPAUT and COPASI lead to identical results within a small margin of tolerance for numerical inaccuracy. Similarly, conversion from chemical reactions (version 2.1.4) to reaction rules (version 3.0.0) lead to identical

simulation results within a small margin of tolerance[3]. Starting with version 3.0.0, for all modifications to the reaction rules and molecule types, the expected new number of reactions generated by each rule and the expected number of total species were documented in files `/versions/v*/sim_log_expected.log`)[4] of the (cell_cycle_model GitHub repository. Equivalency with the auto-generated number of reactions per rule and total species (files `/cell_cycle_model/versions/v*/sim_log.log`) was confirmed by manually comparing relevant differences with the Visual Studio Code (Microsoft) file comparison tool.

3.4.3 Major model assumptions

Major assumptions include: (1) All reactions (even non-elementary reactions) are best modelled with mass action kinetics (exception: Hill function as phenomenological description of nuclear pore phosphorylation by `CCNB(CDK1_Thr14_Tyr15~u)`). (2) Cyclin:Cdk complex concentration is proportional to cyclin concentration. (3) The time for protein expression is negligibly small. (4) The model environment provides all protein subunits, metabolites, components of the transcription-translation and protein degradation machinery that are required by the reactions described in the model. It also provides constant total levels of `CCND()`, `RB1()`, `FZR1()`, `ENSA_ARPP19()`, `PPP2R2B()`, `MASTL()`, `WEE1()`, `CDC25()`, `APC()`, `TP53()` and DNA (see Table 3.1 for alternative names of these proteins). (5)

[3]Direct conversion from ODE to rule-based models is hard to achieve. This is because converting from ODEs to reactions adds information to the model. Similarly, converting from reactions (arbitrary naming of variables from which no judgement about their chemical relationship can be made without assuming certain naming conventions) to rules (syntax for complexes/sites/modifications contains information of how species relate chemically to each other) adds information to the model. Thus, conversion in this direction requires human input of additional assumptions and biochemical knowledge. Using a reaction system as intermediate helps to split the conversion task in more manageable subproblems.

[4]The asterisk is a placeholder for the version name. For v3.0.1, /versions/v3.0.1/n_species_reactions_expected.txt was used instead of a .log file.

(De)phosphorylation of a protein is not changed by binding to another protein (exceptions: `PPP2R2B(ENSA_ARPP19)` actively dephosphorylates `ENSA_ARPP19(PPP2R2B!?, Ser62_Ser67~p)`.`FZR1(nTerm~u)` phosphorylation is in the full model assumed to be reduced due to putative steric hindrances within the `APC(FZR!1,FBXO5!2) .FZR1(APC!1,FBXO5!3,nTerm~u).FBXO5(APC!2,FZR1!3)` complex). (6) `PPP2R2B (ENSA_ARPP19)` is a putative APC phosphatase. (7) Phosphorylated cyclin E and A are immediately degraded. (8) Effects of local enrichment of the modelled components can be ignored.

3.4.4 Data visualisation and illustrations

Data were visualised using MATLAB (R2018a) and Python (v3.9.5) with the matplotlib (v3.4.3) package. For plotting Figures 3.10 and 3.11 the first cell cycle was eliminated from the simulation results as its initial conditions do not match the conditions at the start of subsequent cycles. Illustrations were made with Microsoft PowerPoint.

4

Time Courses of Cell Cycle Regulators

»In so far as a scientific statement speaks about reality, it must be falsifiable: and in so far as it is not falsifiable, it does not speak about reality.«

Karl Popper – The Logic of Scientific Discovery[162]

Contents

4.1 Introduction

The identification of genes responsible for cell division [163, 164] and the discovery of cyclic time courses of proteins called cyclins [165] were historical milestones to unravel the fundamental molecular machinery that controls the cell cycle. The findings were awarded with the Nobel Prize in Physiology or Medicine to Leland H Hartwell, R Tim Hunt and Paul M Nurse in 2001. However, while we now know dozens of molecular components of the cell cycle machinery, obtaining accurate time course measurements of many of these components simultaneously remains technically very challenging. It can be assumed that availability of multiplexed time course measurements of the molecular cell cycle machinery will enable a deeper insight of how known cell cycle regulators systemically interact to control cell cycle progression. Further insight can be obtained by combining these time course measurements with genetic or drug-induced perturbations of the cell cycle. The combination of drug-induced perturbations with time course measurements of cell cycle regulators therefore enables a more systemic characterisation of the effects of experimental and clinically approved drugs. Genetic perturbations on the other hand are a means of very targeted interference with the cell cycle machinery, primarily relevant to basic research. However, such rich time course and perturbation data is difficult to interpret without formalising conceptual cell cycle models into mathematical equations. Conversely, mathematical cell cycle models contain dozens of unknown kinetic parameters. Their inference requires large amounts of experimental information. Time course measurements are a particular informative resource for identifying the kinetic parameters of mechanistic models of the cell cycle. Yet, some molecular species in these models are not observable in multiplexed experiments, which substantially increases the risk of parameter

non-identifiabilities. By measuring time courses in perturbed and unperturbed conditions, improvements in parameter identifiability can be expected.

Over the last years, multiplexed time course measurements of cell cycle regulators where largely obtained through mass spectrometry or immunofluorescence (Table 4.1). As mass spectrometry is a destructive method, cells cannot be followed over time. Therefore, Ly *et al.* combined mass spectrometry experiments with fractionation of cell populations at different cell cycle stages. In their 2014 paper [61] they separated an asynchronously dividing cell population based on their size using elutriation. In their 2015 paper [64] they arrested cells in different cell cycle phases. Most recently, they used fluorescent assisted cell sorting to separate cells based on the abundance of four cell cycle markers [62, 63]. In contrast, Mahdessian *et al.* [166] measured the abundance of cell cycle regulators with immunofluorescence. Like mass spectrometry, immunofluorescence is a destructive method. Therefore, the extent of cell cycle progression was estimated based on the expression of fluorescently tagged DNA replication factor Cdt1, and DNA replication inhibitor Geminin [167, 168]. I aggregated these data and rescaled it to molar protein concentration estimates[1] to consolidate time courses of cell cycle regulators (file /lit_review/time_course_data.xlsx on the cell_cycle_time_course GitHub repository). While these studies provided relevant insight into the dynamics of cell cycle regulators, many low abundance proteins like transcription factors were not present in the proteomic mass spectrometry dataset. In addition, time courses of protein phosphorylation frequently showed considerable variation between replicates in mass spectrometry-based approaches, and were not available from immunofluorescence data.

[1] unpublished work using PaxDB data; Yin Hoon Chew and Jonathan Karr

Table 4.1 | Time course studies.

Study	Cell type	Protein abundance	Cell cycle pseudo-time
Ly *et al.* (2014)	NB4 promyelocytic leukemia cell	mass spec	cell size/elutriation
Ly *et al.* (2017)	NB4 promyelocytic leukemia cell	mass spec	intracellular markers/FACS
Mahdessian *et al.* (2019)	U-2 OS osteosarcoma cells	immunofluorescence	FUCCI marker trajectory
Kelly *et al.* (2022)	NB4 promyelocytic leukemia cell	mass spec	intracellular markers/FACS

Genetic perturbation experiments of cell cycle regulators were so far not combined with multiplexed time course experiments, but with phenotypical readouts (Table 4.2). Neumann *et al.* [169] targeted all human genes with multiple different siRNAs at a time and observed phenotypical changes related to cell cycle control. McKinley *et* Cheeseman [170] specifically targeted 209 cell cycle regulators and visually observed phenotypic defects of cell cycle regulation, using two different CRISPR/Cas9 knockouts per gene. Yilmaz *et al* [171] developed a high throughput technique if CRISPR/Cas9 knockouts become enriched or depleted in a heterogeneous cell culture of wild-type and knockout cells. Unfortunately, closer inspection revealed that phenotypic effects of siRNA knockout are only rarely reproducible between replicates or different siRNAs targeting the same gene (see page 123-124 of /lit_review/time_courses_and_perturbations.pdf in the cell_cycle_time_course GitHub repository for reproducible siRNA gene knockdowns and corresponding phenotypes). Similarly, phenotypic patterns did not always correlate well between the two gRNA constructs per gene in the McKinley *et* Cheeseman screen, as cells with successful knockout in all chromosomes were not isolated. The Yilmaz dataset provides near complete genome coverage, however, the readout of evolutionary advantage is hard to utilize for parameter optimisation in cell cycle models.

In this chapter, we set out to combine perturbation of cell cycle regulators with multiplexed spatially resolved pseudo-time course measurements of cell cycle regulators, to acquire an information-rich dataset for estimating the kinetic parameters

Table 4.2 | Perturbation studies.

Study	Cell type	Perturbation	Readout
Ly *et al.* (2015)	NB4 promyelocytic leukemia cell	serum starvation, hydroxyurea, RO-3306	protein abundance (mass spec)
McKinley *et* Cheeseman (2017)	HeLa	Knockout attempt	phenotype after 4 days (fluorescence microscopy)
Yilmaz *et al.* (2018)	H1 hESC	Knockout attempt	fitness after 23 days (sequencing)
Neumann *et al.* (2010)	HeLa	Knockdown attempt	phenotype over 2 days (fluorescence microscopy)

from the complete cell cycle model described in Chapter 3[2]. We decided to use iterative indirect immunofluorescence imaging (4i) [54] to quantify the abundance of proteins. This technique allows sufficient multiplexing to simultaneously cover all protein components of our cell cycle model, detects low-abundance and phosphoproteins, and enables not only single-cell measurements, but also subcellular resolution of the readouts. Unlike time-lapse microscopy, 4i does not require laborious genetic fusion of fluorescence tags to cell cycle regulators, which may also interfere with some aspects of their native function. As 4i is a destructive technique, time courses cannot be measured directly, but need to be reconstructed. These reconstructed pseudo-time courses have the advantage of naturally representing the typical time course of cell cycle progression. Due to molecular noise, measuring elapsed real time directly would require stretching/compressing of elapsed time by unknown factors at unknown positions for individual single-cell time to obtain a typical time course. Perturbation of cell cycle regulators shall be performed with experimental and clinically approved drugs, and via the CRISPR interference (CRISPRi)[3] system [76]. Together, these experiments shall result in an information-rich dataset for

[2]This work is performed in collaboration with Prof Chris Bakal and Dr Lucas Dent from the Institute of Cancer Research, London (UK). Due to delays pertaining to the COVID-19 pandemic and broken equipment, the experiments are still ongoing. The results Section 4.2.2 will therefore discuss preliminary data and data from Stallaert *et al.* [172]

[3]The sequence of the plasmid backbone was drafted in collaboration with Dr Tobias Strittmatter from ETH Zurich, Switzerland.

optimising or testing computational models of the cell cycle and generating new insight and hypotheses about cell cycle control.

4.2 Results and discussion

The accuracy of the cell trajectory reconstruction depends on the algorithm, its assumptions on the input data, and the input data. The input data typically consists of measurements of multiple variables (cell cycle regulators) in hundreds to thousands of single cells. Each variable that varies over the cell cycle contributes information for cell cycle reconstruction. Conversely, variables that do not vary over the cell cycle only contribute noise to the reconstruction, which may negatively affect reconstruction accuracy. Some algorithms assume that the cells follow a closed circular trajectory, while others assume an open or even branched trajectory. By removing selected cells from branched or circular trajectories prior reconstruction, it is possible to better meet the assumptions of the used algorithm. In order to identify a suitable reconstruction algorithm for cell cycle trajectories, the models from Chapter 3 can be used to create a simulated dataset for which the ground truth trajectory is known. Section 4.2.1 will discuss the identification of a suitable cell cycle trajectory reconstruction algorithm using a simulated dataset, and Section 4.2.2 the cell cycle reconstruction from 4i data.

4.2.1 Time course reconstruction on simulated data

Over the last decade, several cell trajectory reconstruction algorithms have been developed [55, 69, 71–74]. Of these algorithms, Cycler [74] and reCAT [55] are both freely available and designed particularly for cell cycle reconstruction using multiplexed single-cell measurements. In this section their performance on a simulated dataset of the core cell cycle model described in Chapter 3 will be evaluated.

4.2.1.1 Cycler

As discussed in Section 2.2.1.5, Cycler [74] uses a k-nearest-neighbour graph representation of a cell population to calculate the distance from a user-provided start cell. This method requires linearisation of the closed-loop cell cycle trajectory, for instance by removing mitotic cells from the cell population and only reconstructing an interphase trajectory.

To test if Cycler can accurately reconstruct interphase trajectories, 15 000 cells were sampled from a simulation of model version `1.0.0`. As cell division creates two cells from one, asynchronously dividing cell populations contain twice as many cells just after cell division compared to just before cell division, with exponentially decaying cell density across cell cycle time between [69]. This distribution was reflected in sampling from the simulation. Figure 4.1A shows the time course of the sampled cell population. All cells form a perfect closed-loop trajectory (Figure 4.1)B, allowing manual reconstruction of the cell cycle trajectory from only two-dimensional data. To algorithmically reconstruct an interphase trajectory, mitotic cells were removed from the sample. For this simulated dataset, mitotic cells were simply defined as cells older than 0.751 doubling times. To simulate imperfect identification of the first cell in a real experimental dataset, the 50^{th} cell along the cell cycle trajectory was chosen to initiate Cycler with a start cell. As Cycler only ranks the cells by age, cell ranks need to be converted to pseudo-time. From Kafri *et al.* [69] we know that the cell density p at cell age t of a perfectly asynchronous cell population moving across a single circular trajectory can be described as

$$p(t) = k \left(\frac{1}{2}\right)^{\frac{t}{T}}, \tag{4.1}$$

where k is a scaling constant and T the doubling time of the cells. We also know that the whole population size N is the area under this curve

$$N = k \int_0^T \left(\frac{1}{2}\right)^{\frac{t}{T}} \, dt, \tag{4.2}$$

and therefore

$$k = \frac{2N \log 2}{T}. \tag{4.3}$$

The number r of cells younger than t is therefore

$$r(t) = \frac{2N \log 2}{T} \left(\frac{1}{2}\right)^{\frac{t}{T}}. \tag{4.4}$$

Rearranging we obtain the cell cycle time from r as

$$t = -\frac{T}{\log 2} \log\left(1 - \frac{r}{2N}\right). \tag{4.5}$$

In this derivation, we assumed that the number r of cells younger than a certain age is continuous, when in fact it is an integer ranking the cells according to their age. In practice, however, we only know that cell division happened somewhere between the last and the first cell of our sorted list of cells. That is, the rank r could start with any number in $[0, 1]$ an increment in steps of 1. Yet, with a large number of cells, precise timing of mitosis between the last and the first cell becomes negligible. For convenience we simply chose zero-based indexing/ranking of cells for calculating pseudo-times from ranks.

Using (4.5) and noise-free data, Cycler was able to reconstruct a high-quality interphase pseudo-time course (Figure 4.1C) with a Pearson correlation coefficient between reconstructed and true cell cycle time of 0.989 (Figure 4.1). However, experimental single-cell data contains a lot of molecular and measurement noise. This noise is well-approximated by a lognormal distribution [173]. Therefore, the the

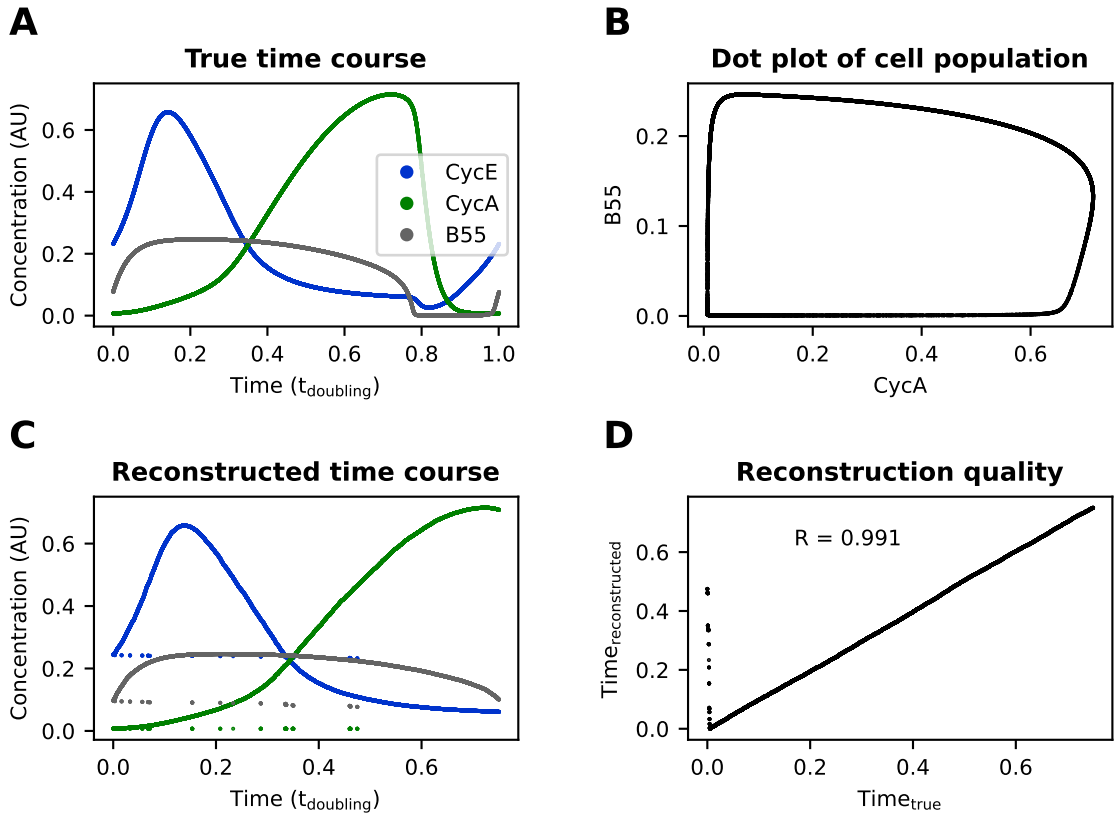


Figure 4.1 | Cell cycle trajectory reconstruction from noise-free simulated data with Cycler. (A) 15 000 cells were sampled across one cell cycle model simulation (version 1.0.0), such that cell density decreases exponentially over the cell cycle with one cell cycle length half-life. Of all 37 model variables, cyclin E, phosphorylated Cdk1:cyclin B and B55 are shown. **(B)**. Scatterplot of the sampled cells from A. **(C)** Reconstructed cell cycle trajectory. To simulate inaccurate identification of a starting cell from real world data, a cell at time 50 was chosen as starting point for cell cycle trajectory reconstruction using Cycler. Cells after 0.751 doubling times where discarded prior reconstruction to open the closed circular cell cycle trajectory. **(D)** Correlation between true and reconstructed cell cycle position for each cell.

above procedure was repeated, except that each variable of each cell was multiplied with a $Lognormal(1, 0.4^2)$ distributed random variable to imitate measurement and intrinsic molecular noise. The results of the interphase reconstruction are shown in Figure 4.2. The dot plot in Figure 4.2 shows a blurred cloud of cells, indicating that two dimensions barely provide enough information to reconstruct a cell cycle trajectory. However, exploiting the information of all 37 dimensions, Cycler

still reconstructed an accurate interphase trajectory with a Pearson correlation coefficient of 0.949 (Figure 4.2D).

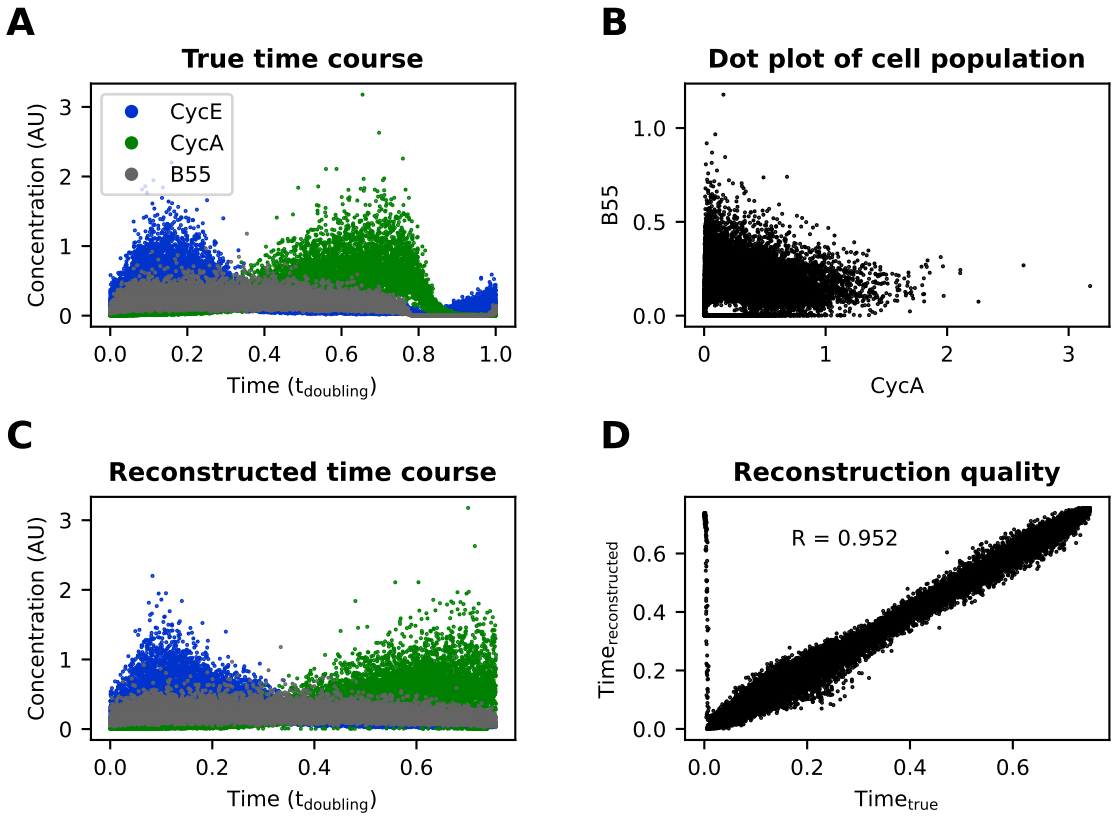


Figure 4.2 | Cell cycle trajectory reconstruction from noisy simulated data with Cycler. (A) 15 000 cells were sampled across one cell cycle model simulation (version 1.0.0), such that cell density decreases exponentially over the cell cycle with one cell cycle length half-life. Concentrations were multiplied with a $Lognormal(1, 0.4^2)$ distributed random variable. Of all 37 model variables, cyclin E, phosphorylated Cdk1:cyclin B and B55 are shown. **(B)**. Scatterplot of the sampled cells from A. **(C)** Reconstructed cell cycle trajectory. To simulate inaccurate identification of a starting cell from real world data, a cell at time 50 was chosen as starting point for cell cycle trajectory reconstruction using Cycler. Cells after 0.751 doubling times where discarded prior reconstruction to open the closed circular cell cycle trajectory. **(D)** Correlation between true and reconstructed cell cycle position for each cell.

4.2.1.2 reCAT

One major disadvantage of Cycler for reconstructing cell cycle trajectories, is its requirement for opening the closed-loop cell cycle trajectory, for instance by removing mitotic cells. As the reCAT algorithm [55] described in Section 2.2.1.6 tries to find the shortest circular path between cell clusters, it naturally reconstructs a circular trajectory. The algorithm assumes that it is provided with data from an asynchronous cell population that moves along a single, unbranched circular trajectory. Designed for single-cell transcriptomic data, which is typically of low cell number, reCAT does not scale well with cell population size. Nevertheless, reconstructions of cell cycle trajectories of a few hundred cells are conveniently possible with a personal computer. As only a few dozens of time points (i.e. cells) during the cell cycle provide a good resolution for all cell cycle events described by the models from Chapter 3, it is still possible to apply a narrow noise-smoothening filter to the output data, while still obtaining acceptable time resolution.

To test cell cycle trajectory reconstruction with reCAT, 300 cells were sampled from model version 2.1.4, using the same exponentially decaying cell density as for Cycler. First, all 49 model variables were provided for trajectory reconstruction without adding noise. Best results were achieved by transforming the data according to

$$\tilde{\boldsymbol{v}} = \log_2 \left(\frac{10^4 \cdot \boldsymbol{v}}{\frac{1}{n} \sum_{i=1}^{n} v_i} + 1 \right), \tag{4.6}$$

where $\boldsymbol{v}$ denotes a variable and n is the number of cells. As shown in Figure 4.3, reCAT was able to perfectly reconstruct the cell cycle trajectory from this dataset ($R = 1.0$). As a next step, the input data was corrupted with $lognormal(1, 0.8^2)$ noise (Figure 4.4A,B). Despite the substantial noise level, reCAT recovered the original cell cycle trajectory with near perfect correlation (Figure 4.4C,D). Yet,

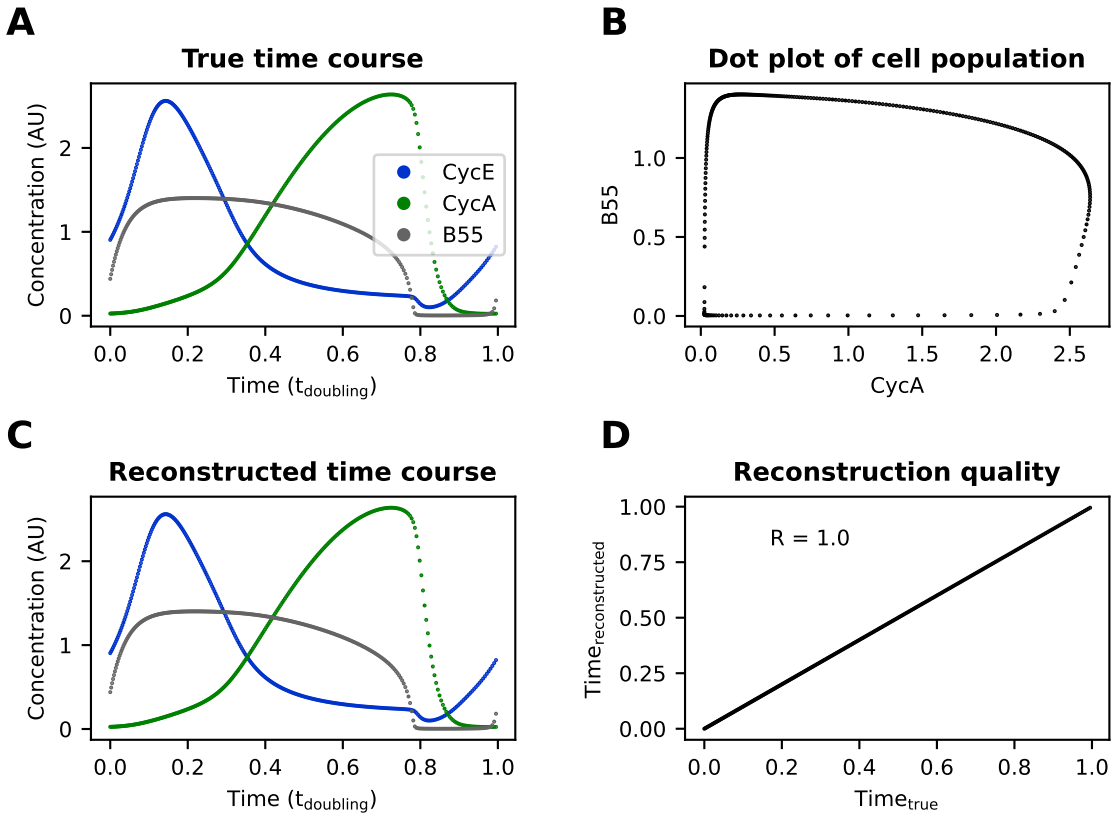


Figure 4.3 | Cell cycle trajectory reconstruction from noise-free simulated data with reCAT. (A) 300 cells were sampled across one cell cycle model simulation (version 2.1.4), such that cell density decreases exponentially over the cell cycle with one cell cycle length half-life. Of all 49 model variables, cyclin E, cyclin A and B55 are shown. **(B)**. Scatterplot of the sampled cells from A. **(C)** Reconstructed cell cycle trajectory. **(D)** Correlation between true and reconstructed cell cycle position for each cell.

Figure 4.4B indicates that two variables provide insufficient information for reconstructing a circular cell cycle trajectory. At the same time, measuring 49 variables that carry cell cycle information with 4i can be challenging. Therefore, reCAT was tested again with the same noise level, but only provided with 9 variables that carry substantial information about the position of a cell in the cell cycle: Ce, Ca, Cb[4], pCb[5], E2f, Fox[6], B55, Apc:Cdh1 and pApc:Cdc20. Despite the high noise

[4]unphosphorylated cyclin B
[5]phosphorylated cyclin B
[6]unphosphorylated FOXM1

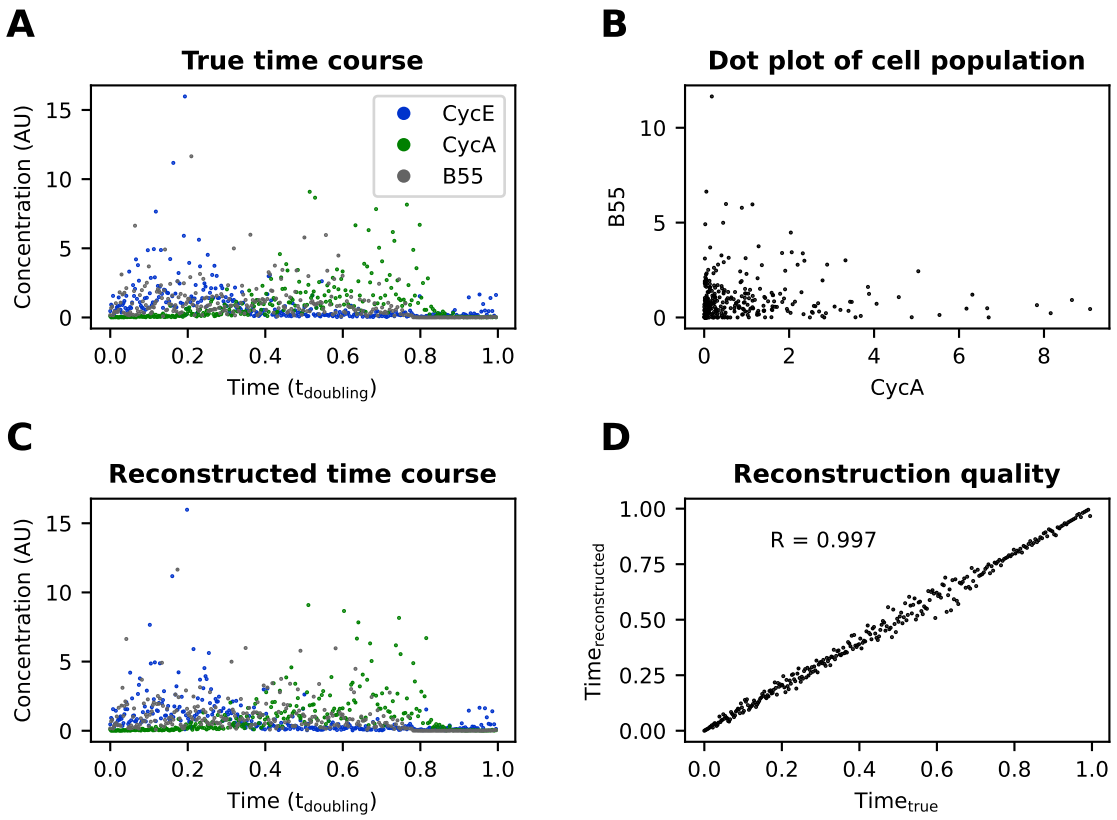


Figure 4.4 | Cell cycle trajectory reconstruction from noisy simulated data with reCAT. (A) 300 cells were sampled across one cell cycle model simulation (version 2.1.4), such that cell density decreases exponentially over the cell cycle with one cell cycle length half-life. Concentrations were multiplied with a $Lognormal(1, 0.8^2)$ distributed random variable. Of all 49 model variables, cyclin E, cyclin A and B55 are shown. **(B).** Scatterplot of the sampled cells from A. **(C)** Reconstructed cell cycle trajectory. **(D)** Correlation between true and reconstructed cell cycle position for each cell.

level and low number of variables, a Pearson correlation coefficient of $R = 0.987$ between reconstructed and true time was achieved (Figure 4.5). The reconstruction error was not equally distributed across the cell cycle, with best reconstruction accuracy in the last 20% of the cell cycle.

4.2.2 Time course reconstruction on real 4i data

For reconstructing time courses from real 4i data, reCAT was chosen over Cycler, due to its superior reconstruction accuracy despite higher noise levels (Figures

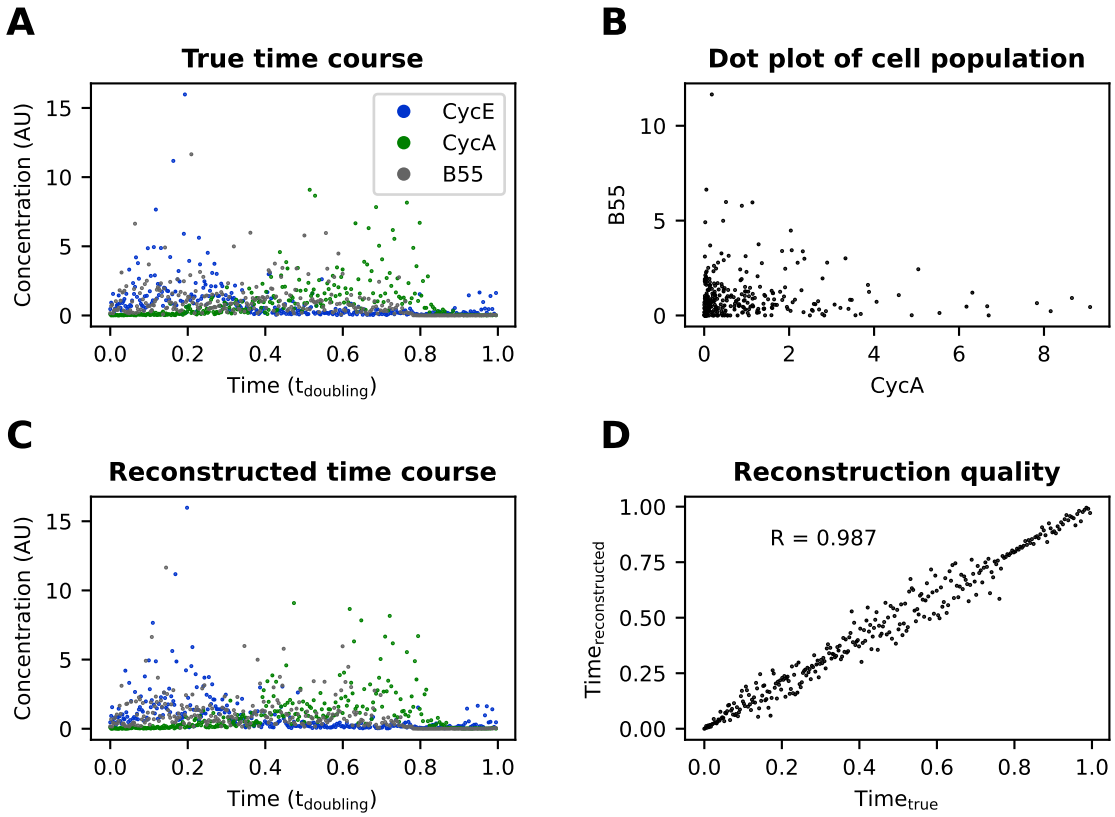


Figure 4.5 | Cell cycle trajectory reconstruction from noisy and reduced simulated data with reCAT. (A) 300 cells were sampled across one cell cycle model simulation (version 2.1.4), such that cell density decreases exponentially over the cell cycle with one cell cycle length half-life. Concentrations were multiplied with a $Lognormal(1, 0.8^2)$ distributed random variable. Of all 49 model variables, cyclin E, cyclin A and B55 are shown. **(B)**. Scatterplot of the sampled cells from A. **(C)** Reconstructed cell cycle trajectory. To simulate limited multiplexing of real world data, reCAT was only provided with 9 of the 49 model variables. **(D)** Correlation between true and reconstructed cell cycle position for each cell.

4.2D and 4.5D) and its ability to directly reconstruct closed-circular trajectories[7]. 4i measurements of the human retinal pigment epithelial 1 (RPE1) cell line were obtained for a DNA stain (DAPI) and six antibodies, targeting CCNA2, CCND1, RB1(Ser807_Ser811~p), CDKN1B and TUBA1A (Tubulin alpha 1a). The cells were

[7]Synthetic data for testing Cycler and reCAT was generated from different model versions (v1.0.0 and v2.1.4, respectively). However, the simulation results are practically identical (Figure 3.8A), so that this conclusion remains unaffected.

segmented into nuclear and cytoplasmic compartment based on DAPI and TUBA1A stains, respectively. Figure 4.6 shows a representative sample of segmented cells with an overlay of TUBA1A, CCND1, RB1(Ser807_Ser811~p) and CCNA2 stains. Mean antibody signals and structural features were extracted for all cells. The resulting data was pre-processed to remove incorrectly segmented cells, cells that detached during iterative elutions, and other artefacts (see Section 4.4.1 for details). In lack of stains or method to remove G0 cells, these remained in the dataset under the assumption that the fraction of G0 cells is low enough to not disturb the reconstruction of an unbranched, circular proliferative cell cycle trajectory.

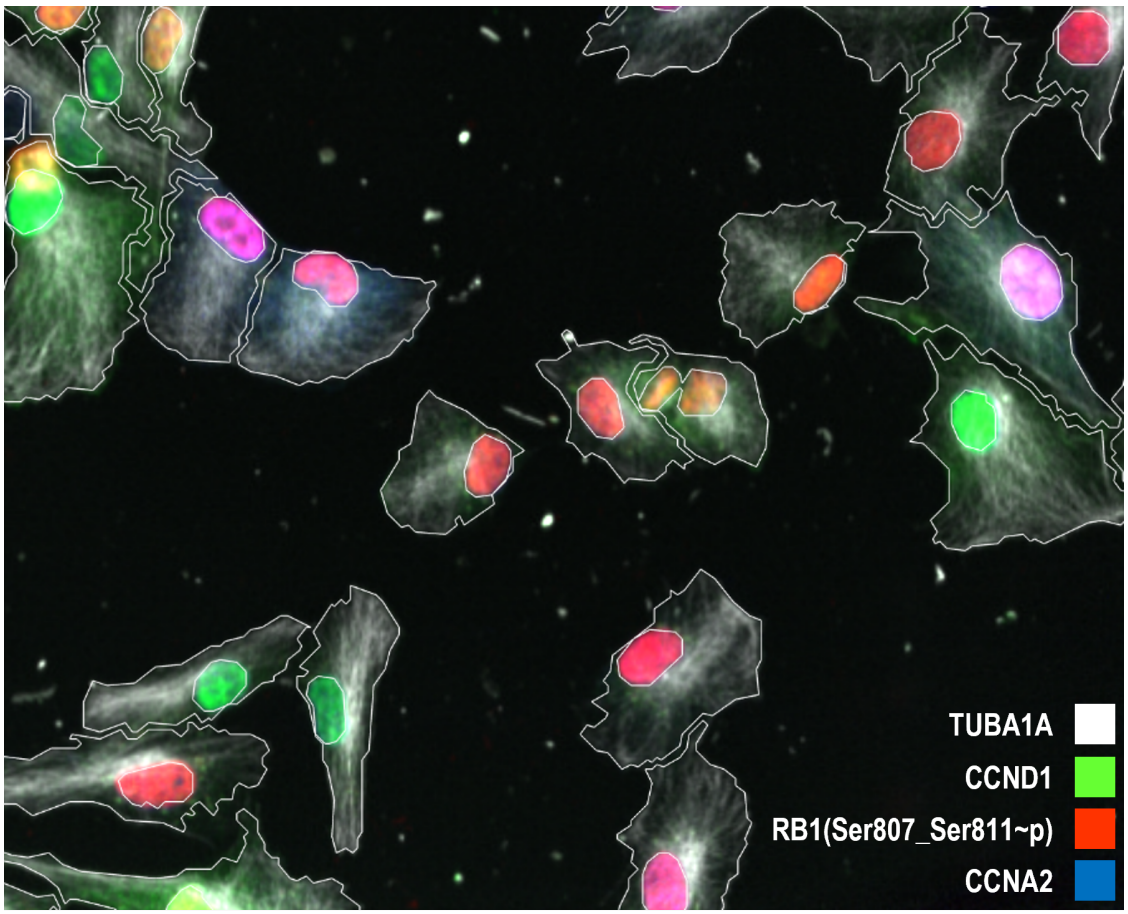


Figure 4.6 | 4i image of segmented RPE1 cells. RPE1 cells were stained with DAPI and six antibodies, four of which are shown in this image. Nuclei and cytoplasm were segmented based on DAPI and TUBA1A stains, respectively, using the TissueMAPS software.[8]

Of the extracted features, nine were chosen for trajectory reconstruction: mean nuclear and cytoplasmic CCNA2, CCND1 and RB1(Ser807_Ser811~p), as well as DNA content (i.e. integrated nuclear DNA intensity), cell area and nuclear area. These features were chosen as they were expected to carry significant information about the position of a cell in the cell cycle. Figure 4.7 shows a representative dot plot and reconstructed trajectory from 300 randomly selected untreated cells. Pseudo-time was calculated based on the observation of approximately 20 h doubling time. As expected, DNA content roughly doubles as the cell cycle progresses, and CCNA2 (see the /4i_dent_lang/2019082X_RPE1/untreated_sorted directory of the cell_cycle_time_course GitHub repository for full data including nuclear CCNA2) significantly increases before a rapid degradation. This CCNA2 degradation is indicative of the M/A transition and coincides with a reduction integrated nuclear DNA intensity. This reduction may be explained by chromatin condensation or cell division. The distinction between one and two cells can sometimes be unclear from snapshot microscopy images without knowing the recent history of the cells in question. This translates to some uncertainty in the precise location of mitosis in the time course data and thus a small error source for the rank to pseudo-time transformation via Equation (4.5). The same trajectory reconstruction was performed on cells treated with the WEE1 inhibitor MK1775 (/4i_dent_lang/2019082X_RPE1/MK1775_treated_sorted directory of the cell_cycle_time_course GitHub repository) and a single cell transcriptomic dataset of asynchronously dividing human embryonic stem cells (hESC) from Leng *et al.* [174] (/lit_review/raw/Leng2015_scRNAseq_reCAT_Kalman.xlsx directory of the cell_cycle_time_course GitHub repository). Unfortunately, both datasets are of limited additional use for optimising the parameters of the cell cycle models from Chapter 3. However, Stallaert *et al.* [172] recently used 4i to obtain measurements

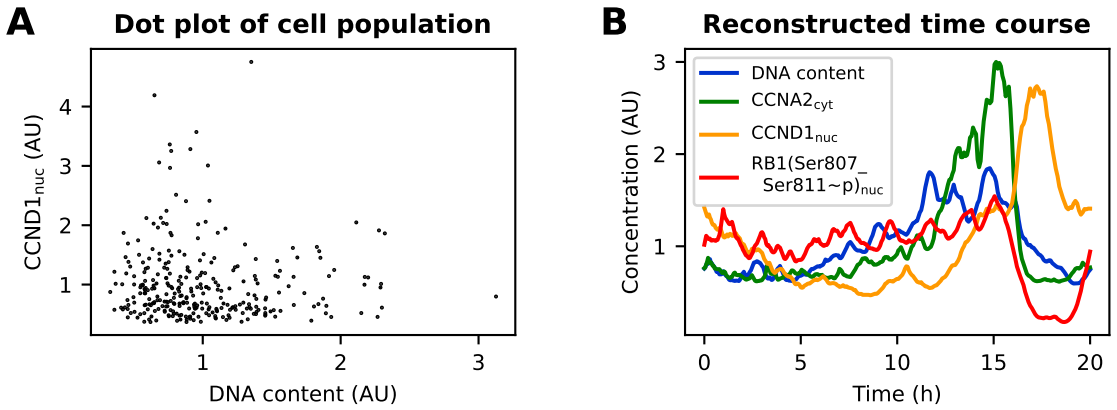


Figure 4.7 | Testing time course reconstruction of cell cycle regulators in RPE1 cells. For better visibility the variables are normalised to a mean of one and axes are clipped. **(A)** Dot plot of 300 untreated RPE1 cells. **(B)** Reconstructed time course of the cell population from (A). reCAT was provided with 9 variables, of which four are shown. Time was calculated using Equation (4.5) and data was smoothened using a Kalman filter.

of 48 cell cycle regulators in thousands of RPE1 cells. Using the dimensionality reduction method PHATE (Potential of Heat-diffusion for Affinity-based Trajectory Embedding) [175] they found that a significant proportion of RPE1 cells exit the proliferative cell cycle trajectory into a non-proliferative G0 arm (Figure 4.8). The re-entry trajectory does not follow the same path as the exit trajectory. Instead, they argue that the arrest state is not reversed, but overcome through elevated expression of proliferative effectors. This is in good agreement with the G1 DNA damage checkpoint in model version 3.1.0. More specifically, Figure 3.9 shows that CDKN1A is not fully reduced to baseline by the time the cell cycle continues, as indicated by rising CCNA/B levels. Since CDKN1A approaches baseline asymptotically, during continuation of cell cycle progression, there is no well-defined point of re-entry on the proliferative trajectory. A projection of the trajectory along the CDKN1A axis may suggest earlier re-entry than the full-dimensional picture. The figure also shows high levels of the proliferative effector CCNE before CCNA/B levels rise which may titrate

some CDKN1A away from the other two cyclins. However, any interpretation of the simulation results must be treated with caution, as the simulation was performed with arbitrarily chosen parameters an arbitrary units of concentration.

Stallaert *et al.* made their 4i dataset publicly available. In addition to some of the cell cycle regulators described in the cell cycle models from Chapter 3 it also contained annotations of a small sample of cells with their true elapsed time post mitosis and their cell cycle phase. Out of 292 measured features in their dataset, they identified the 40 that best predict these annotations using random forests.

Collapsing these 40 features into two dimensions via PHATE as shown in the original publication, allowed me to remove G0 cells from the Stallaert dataset (Figure 4.8). From Stallaert *et al.* it remained unclear, however, if all G0 cells enter the proliferative cell cycle at the same point of the trajectory where they left. Should they do parts of the proliferative trajectory twice, the calculated pseudo-time (Equation (4.5)) would become artificially stretched in these regions. Conversely, should they skip parts of the proliferative trajectory, the calculated pseudo-time would become artificially compressed. It is clear that repeating mitosis and skipping parts of S-phase via a G0 trajectory cannot be the norm, as this would lead to regular chromosome aberrations. Nevertheless, former G0 cells may move along a trajectory that is mostly parallel to the proliferative trajectory, but shifted in a few states for some time, before completely merging with the proliferative trajectory. Such parallel trajectories would bias the reconstructed trajectory in the direction of the shift. Yet, overall the trajectory reconstruction would only be strongly affected if alternative trajectories bypass large stretches of the proliferative trajectory in forward or reverse direction and a large amount of cells at any given time would move on these trajectories. As either can be considered highly speculative at present, it seems reasonable to assume that removal of G0 cells from an asynchronous cell

population leads to an asynchronous cell population that mostly moves along the proliferative cell cycle trajectory. That is, the cell population can be used for reconstructing a proliferative trajectory from snapshot data and for calculating pseudo-time from ranks via equation 4.5.

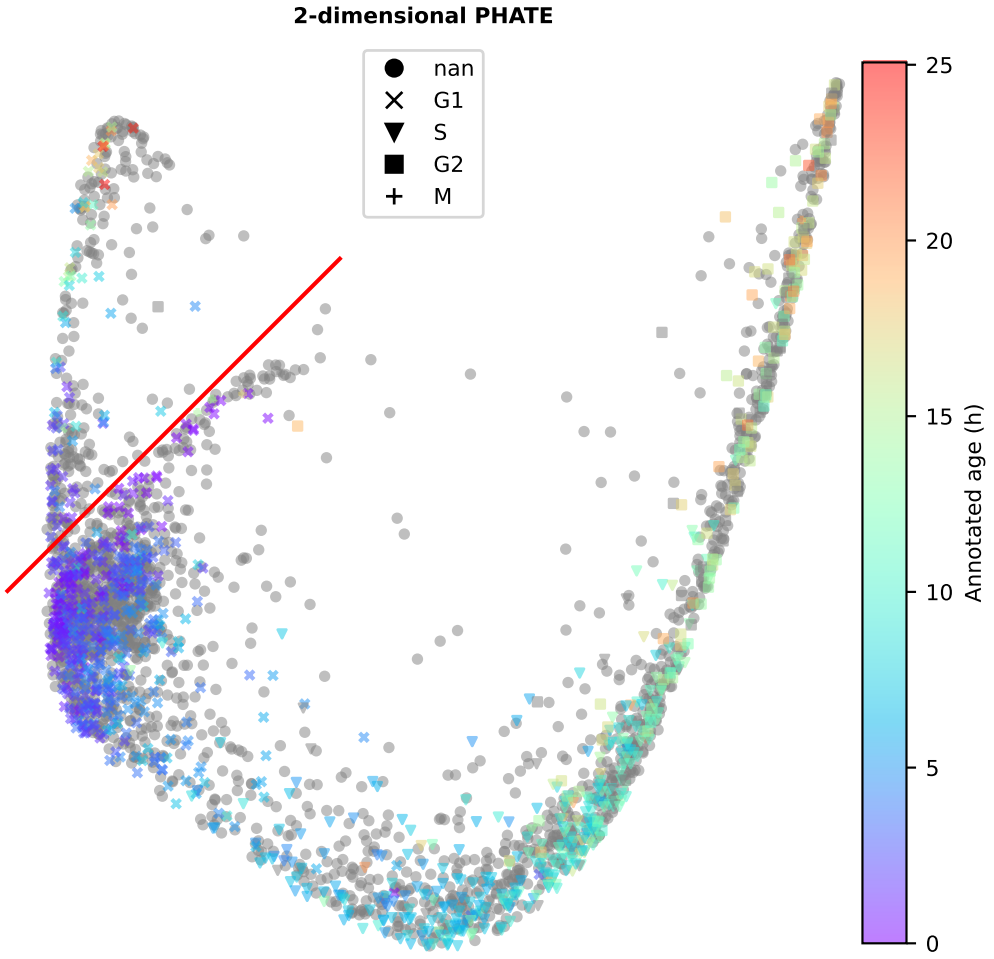


Figure 4.8 | Gating proliferating cells using PHATE. PHATE was performed on publicly available 4i measurements of an asynchronously dividing RPE1 population [172]. Cells above the line indicated by the red line segment were classified as G0. For better visualisation only one third of all cells are shown.

After removal of G0 cells, trajectory reconstruction was performed on 300 randomly selected cells. reCAT was only provided with the 36 of the 40 most predictive cell cycle features identified by Stallaert *et al.* that did not contain missing values.

The other features were ignored during reconstruction, as they may contribute more noise than cell cycle information. Nevertheless, ranking the cells results in time courses of all 292 features, 12 of which are shown in Figure 4.9 (the full dataset is available in table 4i_stallaert/2021_Stallaert_cycling_reCAT/smoothened_data.xlsx of the cell_cycle_time_course GitHub repository, and Kalman-smoothened data can be inspected with this interactive figure). Overall, the reconstructed time correlated well with manually annotated time post mitosis in 103 of the 300 cells ($R = 0.86$). The time courses of DNA content, cytoplasmic CCNA, nuclear RB1(Ser807_Ser811~p) and nuclear CCND1 qualitatively follow a similar pattern as in our own experiments (Figure 4.7). However, our dataset contained mean pixel intensities to approximate concentrations, whereas Stallaert *et al.* used the median. Both experiments found a peak in nuclear CCND1 and a dip in nuclear RB1(Ser807_Ser811~p) after CCNA degradation, but this pattern is less pronounced in the Stallaert data. Interestingly, CCNE concentration appears more stable across the cell cycle compared to life cell imaging experiments in HeLa cells [59] and immunofluorescence measurements in U2OS cells [166].

4.2.3 CRISPRi perturbation

Time course reconstruction relies on the assumption of a homogeneous cell population. For clustered regularly interspaced short palindromic repeats interference (CRISPRi) systems [76] , this can be achieved by integrating the KRAB-dCas9 protein and the gRNA into the genome of a cell, that is then expanded to a monoclonal cell population. However, it can be expected that in particular cell cycle genes are essential for proliferation, rendering the production of permanent knockout/down mutants impossible. Furthermore, creating stable cell lines is a labour intensive process that does not scale well to dozens of perturbations. Transient transfections

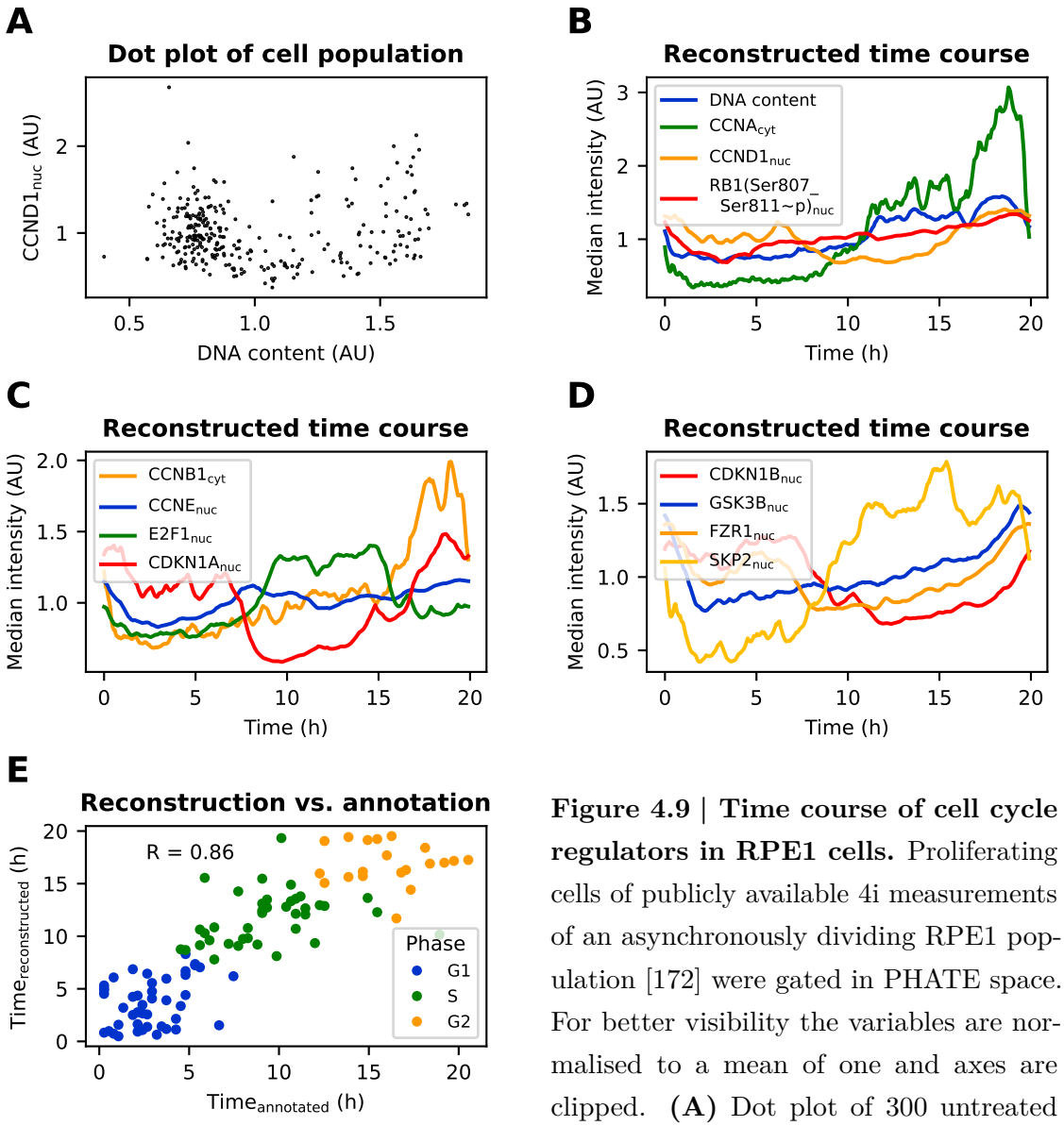


Figure 4.9 | Time course of cell cycle regulators in RPE1 cells. Proliferating cells of publicly available 4i measurements of an asynchronously dividing RPE1 population [172] were gated in PHATE space. For better visibility the variables are normalised to a mean of one and axes are clipped. **(A)** Dot plot of 300 untreated RPE1 cells. **(B–D)** Reconstructed time course of the cell population from (A). reCAT was provided with 36 variables. Time was calculated using Equation (4.5) and data was smoothened using a Kalman filter. **(E)** Correlation between reconstructed an annotated time were known. Cells are color coded by their annotated phase.

of KRAB-dCas9 and gRNAs scale better, but the stochasticity of transfection efficiency of single cells renders any reconstructed trajectories meaningless.

In an attempt to address these problems, the gRNA and KRAB-dCas9 protein was placed on the same plasmid (Figure 4.10). To control for transfection efficiency, KRAB-dCas9 was tagged with the bright red fluorescent protein mScarlet-I [176]. Thereby, single cell variance in transfection efficiency does no longer disturb trajectory reconstruction. Rather, it might be exploited as intrinsic multiplexing of repression strength, by grouping cells based on their mScarlet-I expression, if 4i secondary antibodies are chosen such that their spectra do not overlap with mScarlet-I. The expression of the gRNA is driven by an hU6 promoter, and optionally also a U6 promoter, if a second gene is targetted.

To also allow for stable integration as needed, the payload was flanked by terminal repeats for genomic integration via the piggyBac transposase [177]. However, it can be expected that target repression will not be as effective as with multiple gene copies on transiently transfected plasmids. Therefore, the designed plasmid carried the exceptionally strong KRAB domain of the human ZIM3 gene [178]. As constitutive strong repression of cell cycle regulators may cause problems with cell viability, KRAB-dCas9 expression was put under control of the tetracycline inducible Tet3G system, which operates at low tetracycline concentrations [179].

Four single gRNA constructs were designed for repression of TP53, RB1, CDKN1A and CDKN1B, and one tandem gRNA construct for repression of CCND1 and CCND2. Test for the efficacy of the plasmids are still ongoing.

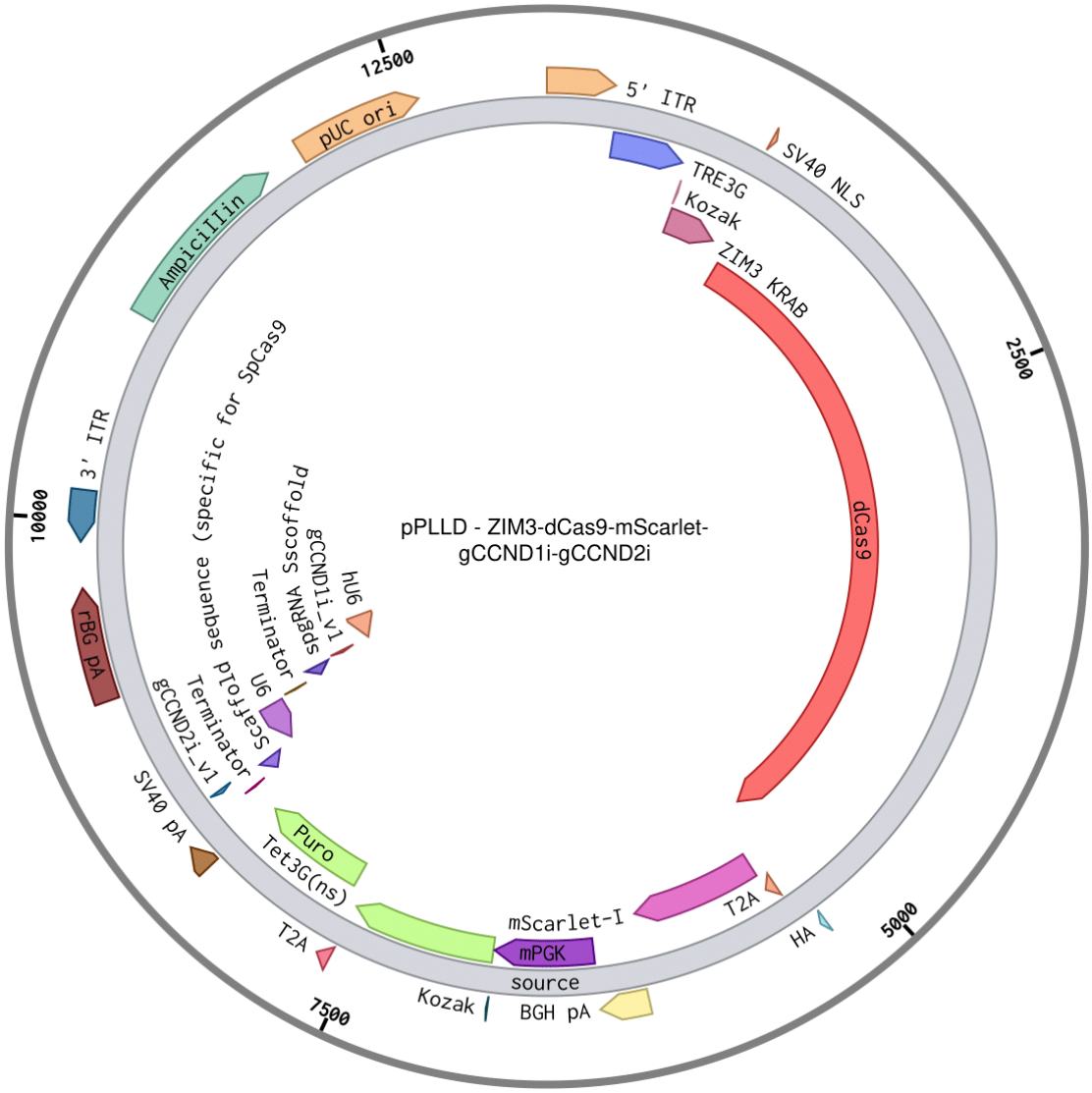


Figure 4.10 | CRISPRi plasmid map. ZIM3-dCas9 expression is driven by the third generation tetracycline response element. mScarlet-I is fused at the N-terminal end of dCas9 via a T2A ribosomal skipping sequence. The third generation reverse tetracycline transactivator is constitutively expressed from a PGK promoter. This figure illustrates a plasmid for cyclin D inhibition, with guide RNAs targetting CCND1 and CCND2 promoter regions, which are expressed from hU6 and U6 promoters, respectively. The whole construct is flanked by inverted repeats for integration via the piggyBac transposase.

4.3 Conclusion

Taken together, the reCAT algorithm demonstrated highly accurate reconstruction of cell cycle trajectories from simulated data, even when noise was high and variable

number low. It was therefore chosen to reconstruct cell cycle trajectories from three datasets: single cell transcriptomic measurements in H1 hESC [174], 4i data from untreated and WEE1 inhibited RPE1 cells[9], and highly multiplexed 4i measurements in untreated RPE1 cells [172]. The time courses for several cell cycle regulators including the cyclins CCNA and CCNB1 roughly matched the expectations from previous literature. CCNE, however, appeared more stable across the cell cycle than previously observed in other cell lines [59, 166].

From the perspective of using the reconstructed time courses for parameter estimation, the Stallaert data enabled the reconstruction of time courses for 292 features at a resolution of 300 time points per cycle. There is some overlap between these features and informative observables for optimising the parameters of the models from Chapter 3. However, as with many high throughput experiments, care must be taken when they differ from more targeted experiments in the literature. Potential error sources include insufficient or unspecific antibody binding, lack of background signal subtraction, illumination artefacts, or violation of the assumptions for trajectory reconstruction. If correction is not possible, such datapoints should be removed, or weighted lower during optimisation procedures.

Going forward, we aim to extend the 4i experiments from Section 4.2.2 to dozens of cell cycle variables specifically chosen to constrain the parameters of the models from Chapter 3. Combining this approach with small-molecule inhibitors and highly specific CRISPRi downregulation of selected cell cycle regulators, we expect to generate a dataset containing extensive information about cell cycle control. Such a dataset could readily be used to estimate parameters of cell cycle models, and benchmark their predictive power with respect to knockdowns.

[9] unpublished work; Lucas Dent, Paul F Lang and Chris Bakal

4.4 Methods

4.4.1 Data cleaning

Outlier cells[10] with respect to the following metrics were excluded from further analyses of our own 4i data: nuclei count, nuclear area, cellular area, difference in DAPI stain between first and last round (indicating detached cells), standard deviation of DAPI stains of all rounds (indicating cells with inconsistent staining pattern), cytoplasmic DAPI stain in all rounds (indicating cells with detached cells on top), and ratio between cellular and nuclear area (indicating segmentation errors).

Outlier cells[10] with respect to the following metrics were excluded from further analyses of the publicly available data from Stallaert *et al.* [153]: nuclear area, DNA content, CDKN1A(Thr145~p), cellular area, cytoplasmic DAPI stain in all rounds, and ratio between cellular and nuclear area (indicating segmentation errors).

4.4.2 Cycler

15 000 cells were sampled across one cell cycle model simulation (version 1.0.0), such that cell density decreases exponentially over the cell cycle with one cell cycle length half-life. Cells > 0.751 doubling times were classified as mitotic and removed to linearise the circular cell cycle trajectory. All 37 model variables were provided for trajectory reconstruction. The 50^{th} cell of the cell cycle was used as initial start cell. Cycler was run with default settings ($k = 15$ initial neighbours, $l = 8$ chosen neighbours, $n_l = 100$ landmark points and an ensemble of $n_g = 10$ 8-out-of-15-nearest-neighbour-graphs). The order of the cells was converted to pseudo-time via Equation (4.5).

[10]Exact thresholds available in the curation directories of the respective datasets on the cell_cycle_time_course GitHub repository. Values are displayed by hovering the mouse over the dashed red line indicating the threshold. Figures can be opened as described in Section 1.4.1 or the repository README.

4.4.3 reCAT

reCAT was provided with 300 cells sampled across one cell cycle model simulation (version 2.1.4). Variables were transformed according to Equation (4.6) as described in the main text. reCAT was run with default settings resulting in an ordered list of cells in a cycle. The direction and start of the cycle were chosen automatically from patterns of cell cycle variables and corrected manually if necessary. For the Stallaert dataset manual correction was informed by manual annotation of cell age and phase. Raw (non-transformed) variables were divided by their mean. A Kalman filter was applied to the time series, using the R programming language with functions `StructTS` and `tsSmooth`. The time series was padded with the last 10% of the cell cycle at the beginning and the first 10% of the cell cycle in the end. `StructTS` optimised the maximum likelihood of the "level" model, where the evolution of variable $x(t)$ is described by

$$x(t+1) = x(t) + \epsilon_x, \epsilon_x \sim \mathcal{N}(0, \sigma_x^2) \tag{4.7}$$

and the measurement $y(t)$ by

$$y(t) = x(t) + \epsilon_y, \epsilon_y \sim \mathcal{N}(0, \sigma_y^2). \tag{4.8}$$

Paddings were removed and the order of the cells was converted to pseudo-time via Equation (4.5).

4.4.4 CRISPRi plasmids

Plasmid backbones were drafted with the Benchling design tool (San Francisco, USA) and gRNA inserts were created with the GPP sgRNA Designer (Broad Institute, USA). Complete plasmids were ordered from VectorBuilder (Chicago, USA) and shipped as minipreps after minor adjustments to the drafted sequence.

4.4.5 Data visualisation and illustrations

Data was plotted using Python (v3.9.5) with the matplotlib (v3.4.3) package, except for the interactive online figures, which were created with plotly library (v4.9.1) in R (v3.6.1).

Collaborator contributions

Lucas Dent performed the 4i experiments and feature extraction from raw microscopy images leading to Figures 4.6 and 4.7. He made major contributions in planning these experiments and interpreting the results. Tobias Strittmatter (ETH Zurich, Switzerland) drafted the sequence of the CRISPRi plasmid backbone. The remainder of this chapter, including the design of the gRNAs, is the work of the author.

5

Estimating Parameters of the Cell Cycle Model

»Scientific knowledge differs from other kinds of knowledge, especially from everyday knowledge, by its higher degree of systematicity.«

Paul Hoyningen-Huene – Systematicity: The Nature of Science[180]

Contents

5.1 Introduction

Computational models of biochemical reaction networks are created with the goal of explaining experimentally observed data and even predicting the outcomes of new experiments. The set of reactions to be used can often be identified from the present literature. Kinetic laws can often be inferred from the reaction formula and contextual knowledge about the reaction. Typical choices include mass action kinetics (for elementary reactions), Michaelis-Menten kinetics (in case of certain enzymatic reactions) and hill kinetics (for instance in case of cooperative reactions or multisite phosphorylations). However, the values of the kinetic parameters have frequently not been determined, or not been in the context implied by the model. Therefore, these parameters need to be inferred from the experimental observations the model tries to explain.

However, such parameter estimation problems are hard to solve. To recapitulate the points from Section 2.3.2, the major challenges include unknown parameter bounds, high dimensional parameter space, unknown parameters in the observation function and noise model, presence of multiple local optima and structural and practical parameter unidentifiabilities. According to the no free lunch theorem, there is no single optimisation algorithm that can perform well on all tasks [181]. Selecting an appropriate algorithm for the problem at hand is therefore of utmost importance.

Managing these challenges is still a matter of ongoing research. Adjoint sensitivity analysis (see Extended Methodological Background, Section B Section B.1.2.4) provides a way of calculating gradients that has almost constant scaling with the number of parameters, i.e. is almost independent of the number of parameters [78]. Various regularisation techniques have been developed to prevent the risk of overfitting that is associated with high dimensional parameter spaces (section

2.3.4). Hierarchical optimisation enables analytical calculation of the parameters in observation functions and noise models [82]. Various global optimisation algorithms (see Extended Methodological Background, Section B.1.1) aim at finding a global optimum, but can typically not guarantee that the found optimum is globally optimal. Full Input-State-Parameter Observability (FISPO) can determine structural parameter unidentifiabilities prior to parameter estimation [182] and profile likelihoods can find structural and practical unidentifiabilities after optimisation [183]. However, both approaches do not scale to large models. Global parameter Estimation with Automated Regularisation via Sampling (GEARS) implements a strategy for adjusting parameter bounds and combines it with global optimisation, identifiability analysis and regularisation [83]. Yet, it too suffers from scalability issues. To manage the consequences of the no free lunch theorem, much thought has gone into benchmarking optimisation algorithms against typical problems arising from estimating the parameters of biochemical reaction networks [86, 184, 185]. Most significantly, however, the development of the PEtab format for specifying biological optimisation problems enables testing of various algorithms for the specific problem at hand [24]. Previously, every optimizer had its own way of specifying optimisation problems, leaving modelers with the pain-staking task of reformulating the optimisation problem for every optimiser they wanted to test.

As indicated in the title, this thesis aims at improving our mechanistic understanding of cell cycle dynamics by providing a mathematical model that explains how the observed dynamics of cell cycle regulators can emerge from the biochemical reaction network governing the cell cycle. Chapter 3 describes the creation of a simulatable biochemical reaction network with kinetic rate laws, that describes cell cycle control mechanisms. Chapter 4 discussed the reconstruction of time course data from cell cycle regulators. This chapter shall bring the results from the

previous two chapters together, by fitting the unknown parameters of the models from Chapter 3 to explain the time courses from Chapter 4[1].

5.2 Results and discussion

To identify an algorithm that is capable of finding the globally optimal parameters of the cell cycle models from Chapter 3 (in the following referred to as *solving the cell cycle optimisation problem* for brevity), the optimisation was first performed on simulated data, for which the ground truth parameters were known. Thereafter, the optimizer was provided with real experimental data.

5.2.1 Maximum likelihood fit on simulated data

5.2.1.1 Multi-start optimisation

Like the name suggests, multi-start optimisation starts several local optimisers from different random starting points in the parameter search space. Here, the MATLAB implementation from Villaverde *et al.* [86] with adjoint sensitivity gradients was used to solve the cell cycle optimisation problem (version 2.0.0), where 101 evenly spaced and noise-free time points of all 41 model species were provided as observables. All 128 parameters (incl. initial conditions) had to be found within a $\log_{10}$ transformed search space corresponding to a $0.1 \cdot \boldsymbol{\theta}_{true}$ to $10 \cdot \boldsymbol{\theta}_{true}$ search window prior to transformation. The optimisation from 100 start points chosen via random sampling took 2.3 days on a single core of a personal computer (Intel®Core™ i7-8550U CPU @ 1.80GHz). The ground truth parameters were not recovered and even the best fit remained unsatisfactory by visual inspection. Closer inspection of the solutions revealed that the objective function value of all

[1]This work is performed in collaboration with Prof Julio Banga and Dr David Rodriguez Penas from the Spanish National Research Council in Vigo (Spain), and Dr Daniel Weindl from the Helmholtz Centre Munich (Germany)

100 multi-starts differed from each other, suggesting that there are at least 100 different local optima in within the search space volume.

5.2.1.2 Cooperative enhanced scatter search

To avoid premature attraction to local optima, the same problem was provided to the Cooperative enhanced Scatter Search (CeSS) algorithm [87] described in Section 2.3.3.1, again using adjoint sensitivities to calculate gradients for the local search. Optimisation took 10.7 hours using four cores of a personal computer (Intel®Core™ i7-8550U CPU @ 1.80GHz). This time, the ground truth parameters were recovered with only minor deviations (all values were within $[0.958 \cdot \boldsymbol{\theta}_{true}, 1,219 \cdot \boldsymbol{\theta}_{true}]$), except for the initial conditions of Apc:Cdh and pCdh. In the cell cycle model total Cdh is constant. CeSS has recovered the correct amount for $Apc : Cdh(t = 0) + pCdh(t = 0) = 0.87$ up to the second decimal digit, but estimated the initial condition of Apc:Cdh higher and pCdh lower than their respective ground truth values. As all species were provided as observables, this confusion of initial values increases the objective function value. Therefore, the otherwise near perfect recovery of ground truth parameters indicates structural identifiability of the posed problem. Had there been structural unidentifiabilities, that is perfectly flat valleys in the objective function, it would be a highly unlikely coincidence that CeSS returns the ground truth parameters instead of any other point in the valley. Figure 5.1 shows the practically perfect overlap between the time course of the noise-free simulated datapoints and the simulation with estimated parameters.

5.2.1.3 Self-adaptive cooperative enhanced scatter search

Having found CeSS as an optimizer that is capable of solving the cell cycle optimisation problem on a personal computer, the next step was to speed up

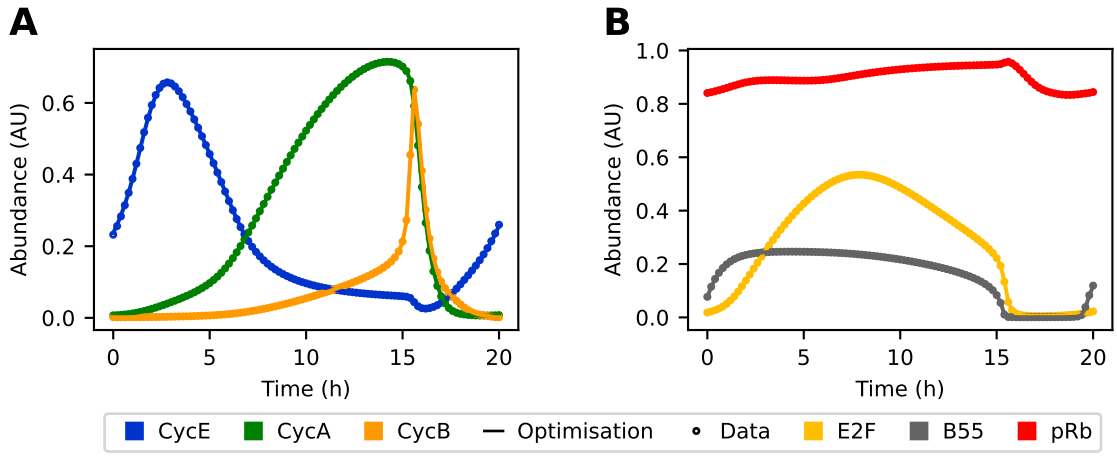


Figure 5.1 | Testing Cooperative enhanced Scatter Search (CeSS). 101 evenly spaced simulated, noise-free datapoints of 41 variables were generated with model version 2.0.0 (circles). CeSS recovered the parameters from within a $[0.1 \cdot \boldsymbol{\theta}_{true}, 10 \cdot \boldsymbol{\theta}_{true}]$ search window with good accuracy. The lines show simulated time courses for six of the 41 variables, using the parameter set found by CeSS.

optimisation. As discussed in Section 2.3.3.1, CeSS has been further improved by allowing successive adaptation of hyperparameters, improving the cooperation strategy and timing, and combining fine- and coarse-grained parallelisation on HPC hardware. This improved version is called self-adaptive cooperative enhanced scatter search (saCeSS) [88]. To test saCeSS on model version 3.0.0, the optimisation problem was updated to better mimic the experimental data from Chapter 4. In particular, the Stallaert *et al.* dataset includes five observables that are informative for this version of the cell cycle problem: CCNE, CCNA, CCNB1, E2F1 and RB1(Ser807_Ser811~p). The expected density of the 300 datapoints is exponentially decreasing over the cell cycle, with a half-life of one doubling time. Importantly, the observables represent antibody fluorescent intensities and were rescaled to a mean of one. This means that observables y map to model species via

$$y_k = o_k + s_k \sum_{i \in K} x_i, \tag{5.1}$$

where the index k denotes different antibodies, o_k represents an offset constant, s_k a scaling constant, K the set of all species to which the k^{th} antibody binds, and x their concentration. The resulting optimisation problem including the species-to-observable mapping, observable noise, simulated data, and estimated parameters with bounds was specified in the PEtab format and can be found in the /versions/v3.0.0/PEtab_PL_v3_0_0sim/ directory of the cell_cycle_petab GitHub repository. The simulated observables were corrupted with $\mathcal{N}(1, 0.1^2)$ multiplicative noise and o_k was set to zero for all k.

Since several species are no longer directly observed and scaling constants are introduced as new parameters, the problem is no longer expected to be structurally identifiable. In addition to causing unidentifiabilities, the scaling parameters increase the dimensionality of the optimisation problem. Therefore, hierarchical optimisation was implemented into the saCeSS optimiser. Two different local solvers were tested for the optimisation, the parPE solver using adjoint sensitivity gradients, and gradient free dynamic hill climbing (DHC). PEtab problems specify the objective as posterior probability. Here, uniform priors and a standard deviation of 0.1 was used, rendering the problem equivalent to least-shares optimisation constraint to parameter bounds. Both solvers went under the objective function value of the ground truth parameters, indicating slight overfitting. As expected, neither solver recovered the ground truth parameters, which can be attributed to structural unidentifiability and overfitting. The lowest objective function value was found with the parPE solver. A simulation with the corresponding parameter values and the convergence curve is shown in Figure 5.2.

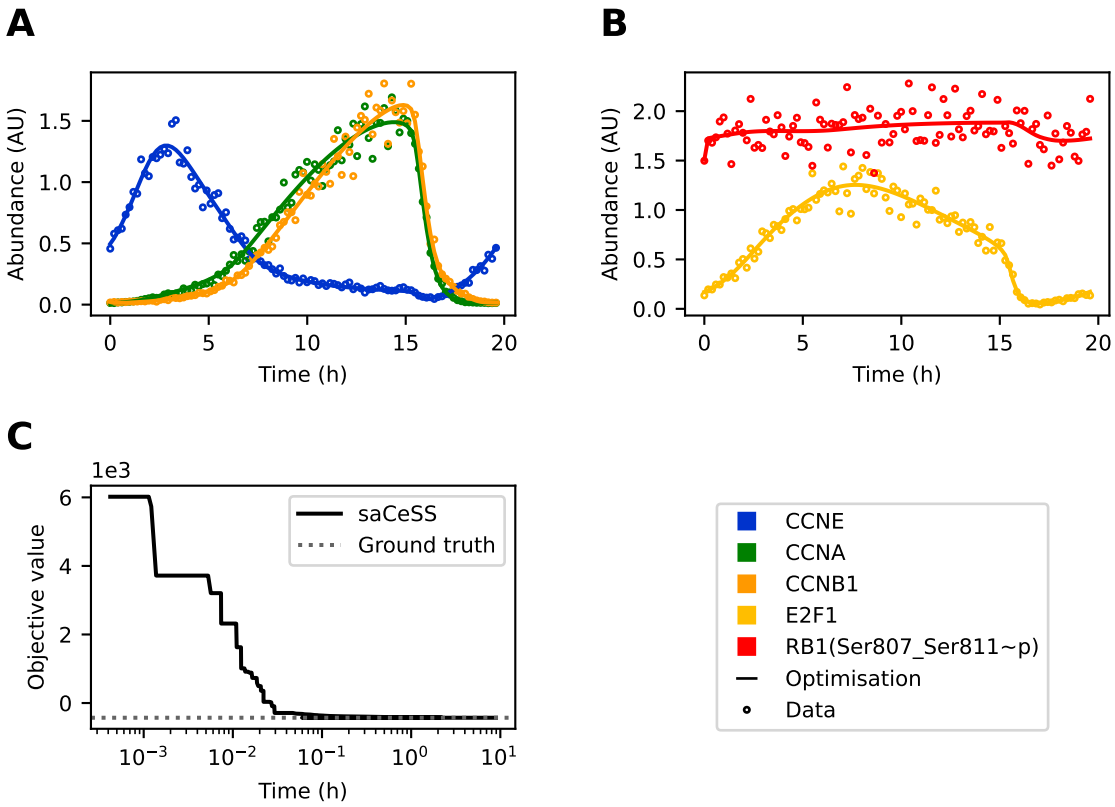


Figure 5.2 | Testing self-adaptive Cooperative enhanced Scatter Search (saCeSS). 300 datapoints of five observables with exponentially increasing spacing were generated from simulation results of model version 3.0.0. The datapoints were corrupted with $\mathcal{N}(1, 0.1^2)$ multiplicative noise (circles). saCeSS optimised the parameters from within a $[0.1 \cdot \boldsymbol{\theta}_{true}, 10 \cdot \boldsymbol{\theta}_{true}]$. **(A, B)** The lines show simulated time courses for the observables, using the parameter set found by saCeSS with the parPE local solver (legend in lower right panel). **(C)** Convergence curve of the negative log-likelihood during optimisation using additive measurement noise ($\sigma = 0.1$). Initial parameter guess was ignored).

5.2.2 Fitting real data

After the promising saCeSS results for simulated 4i data, the real experimental measurements were plugged into the PEtab problem, along with making a minor update to the model from version 3.0.0 to version 3.0.1. However, while there is a notion of compartmentalisation in the data, there is no such notion in the model. As cell cycle regulators mostly locate to the nucleus, only the nuclear

measurements were used for parameter optimisation. Using the real experimental data, four particular challenges were encountered during the optimisation.

First, simulations with the optimised parameters did not result in oscillations. By duplicating the experimental data to obtain two consecutive cycles, oscillatory behaviour could be enforced.

Second, simulations with the optimised parameters unexpectedly showed small dampened oscillations of observables for a short period of time just after the metaphase anaphase transition. Attempts of manually adjusting the parameters led to the following hypothesis. Residual CCNA and CCNB1 fluorescence signal after anaphase in the Stallaert dataset (section 4.2.2) enforce too high levels of these two species in the model. As a consequence, there is substantial E2F1 phosphorylation, and thus degradation especially when the activity of the counteracting phosphatase PPP2R2B is suppressed by action of CCNB1. To overcome these high degradation levels, E2F1 synthesis rates must be high. In other words, E2F abundance is highly responsive to changes in kinase/phosphatase ratio. Once kinase/phosphatase ratio drops, E2F1 shoots up, activates slow CCNA synthesis, which results in E2F1 phosphorylation and thus degradation. Attempts to manually reduce E2F1 degradation and synthesis rates failed to produce sustained oscillations. However, unlike model version 3.0.1, real cells contain cyclin kinase inhibitors, i.e. the active concentration of cyclins is lower than the concentration observed through antibody binding. More generally, any observed protein may be partially bound by known or unknown stoichiometric inhibitors or be held otherwise in an inactive state. To account for these biological possibilities, as well as potentially incomplete background subtraction in the Stallaert dataset, the offset parameters o_k in the observation model defined in Equation (5.1) were allowed to deviate from zero.

Third, using real world data, the global optimum may lie out of the specified parameter bounds. Therefore, active windows (i.e. windows where the best possible solution found by the optimiser) were shifted in the corresponding direction by a factor of 10. These shifts were performed iteratively between the multiple job submissions to that were required due to limitations of maximal job execution times on the cluster.

Fourth, saCeSS tended to perform better with DHC than parPE as local solver. Therefore, DHC was used for problems containing real experimental data.

To further incorporate knowledge that cyclins are present in higher concentrations than E2F1 and RB1, Laplacian priors were introduced for the scaling parameters of the cyclins. All these strategies were combined into a new problem specification. Using the parameters from the oscillatory but non-fitted model as initial guess, saCeSS was able to find a parameter set that leads to simulations that are in good agreement with experimentally observed time courses. Figure 5.3 shows the convergence curve and directly compares the measurements and the simulation. Kalman smoothened measurements are shown in Figure 4.9. The corresponding PEtab problem can be found in the /versions/v3.0.1/ directory of the cell_cycle_petab GitHub repository. Looking at CCNE and RB(Ser807_Ser811~p), it can be observed that the exact initial conditions are not reproduced when the second round of the cell cycle starts. This is because saCeSS may fit the initial conditions to the noise present in the first datapoint. However, this is not further problematic, as the simulation with optimised parameters shows nearly two identical consecutive oscillations, suggesting convergence to a limit cycle.

The optimisation procedure was repeated for model version 3.2.0. This version contains relevant variables to use three additional observables in the Stallaert [153] dataset: SKP2, CDKN1A and CDKN1B. The corresponding PEtab problem

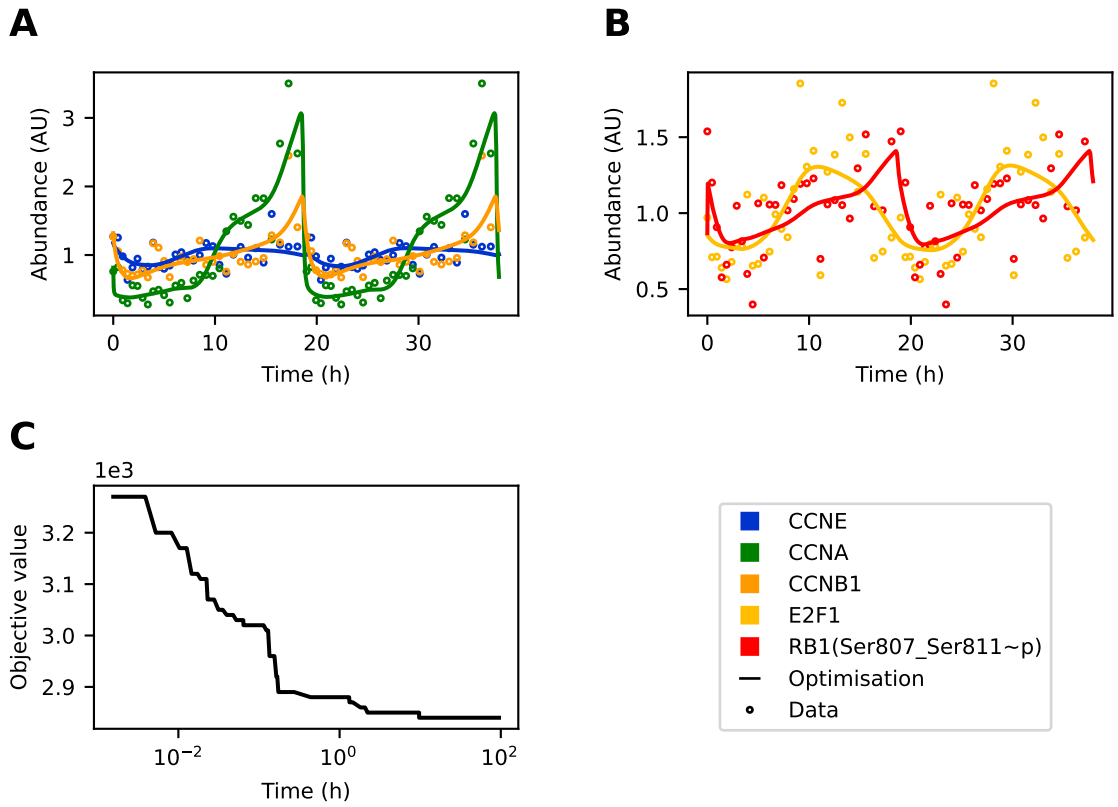


Figure 5.3 | Model version 3.0.1 fitted to pseudo-time courses of RPE1 cells. **(A, B)** Experimental measurements and simulation results using estimated parameters. For better visibility, only every 10^{th} measurement is shown. **(C)** Convergence curve of the negative log-posterior during optimisation using additive measurement noise ($\sigma = 1$). Initial parameter guess was provided (see PEtab parameter table).

can be found in the /versions/v3.2.0/ directory of the cell_cycle_petab GitHub repository. Figure 5.4 compares the simulation results using the estimated parameters to experimental measurements, and Figure 4.9 shows Kalman smoothened measurements. Again, the simulated time courses of the observables are in good agreement with the experimental measurements. In comparison with version 3.0.1, the time courses of 3.2.0 show similar patterns.

Although this model version contained the cyclin kinase inhibitors CDKN1A and CDKN1B, and the offset parameters for the nuclear CCNA and CCNB1 observables had a zero-centered Laplacian prior, their fitted value reached the

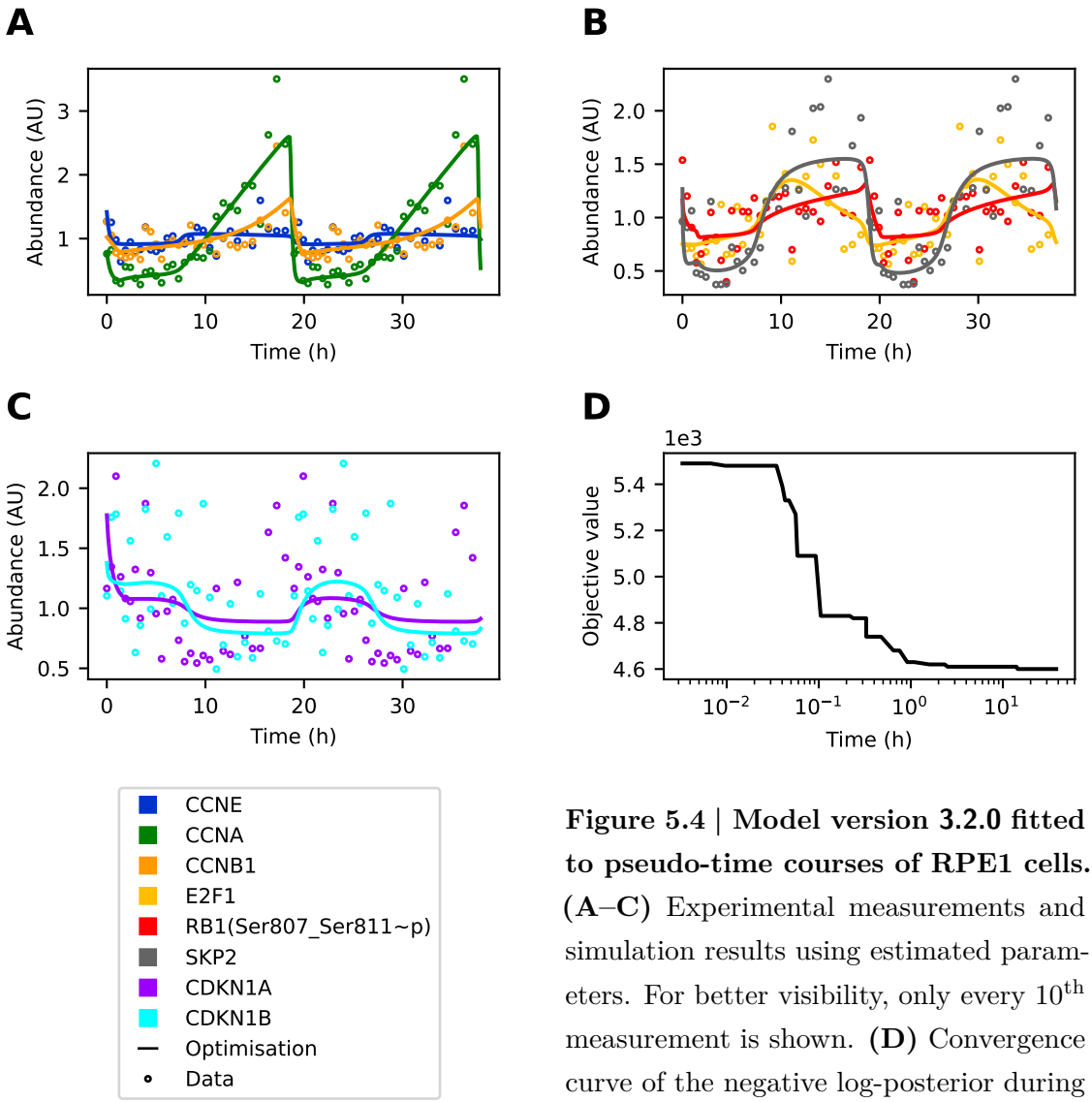


Figure 5.4 | Model version 3.2.0 fitted to pseudo-time courses of RPE1 cells. **(A–C)** Experimental measurements and simulation results using estimated parameters. For better visibility, only every 10^{th} measurement is shown. **(D)** Convergence curve of the negative log-posterior during optimisation using additive measurement noise ($\sigma = 1$). Initial parameter guess was provided (see PEtab parameter table).

upper boundary (set at 1.2 times the minimal non-smoothened observed value). This suggests that near complete Ckd-independent inactivation of CCNA and CDKN1B is required to explain the observed data. In the model simulation with estimated parameters, total CCNA and CCNB1 are degraded to 4.7% and 21% of their maximum values, respectively.

It would also be interesting to compare the optimised parameters to existing data from experiments or models. However, comparisons with experimental data such as those described by Williams *et al.*[112] are hardly possible. This is because our models use arbitrary concentration units, so that only concentration-independent parameter values can be directly compared. Furthermore, available experimental measurements are typically taken in a setting that is very different from the one described by the optimised model here, i.e. living RPE1 cells.

Comparing our optimised parameters to existing models is not straightforward either for several reasons. For instance, models with estimated parameters like the one presented here still contain unidentifiable parameters. It would only make sense to compare identifiable parameters, but it is often not known which parameters are identifiable. Even if that information were available, it would be a challenging task to map parameters from one model to another model. In many cases this is even impossible, as most models do not use clear semantics for the modelled species and thus reactions. In fact, in most models a model reaction does not describe a single biochemical reaction. Instead it describes a set of elementary biochemical reactions. The reactants and products of these reactions may differ across models in the protein isoforms, post-translational modification states and combinations thereof. Without being able to map reactions to each other, no comparisons can be made. In addition to that, most models were fit to different cell types, which may have different chromatin state and thus different parameters for many types of reactions.

5.3 Conclusion

In summary, (sa)CeSS has been identified as a suitable method for solving the cell cycle optimisation problem. All parameters are identifiable when noise-free time

courses of all states are known. This however, is only the case for simulated data. Finding a solution for the cell cycle optimisation problem using 4i data proved to be more challenging. Initially the found solutions did not produce oscillatory behaviour. This issue could be resolved by providing time courses of two consecutive cell cycles. Good fits to the experimental data were obtained after introducing offset parameters in the observation model, adapting parameter bounds as needed and using DHC as local optimiser. Plugging the found solutions into model version 3.2.0 led to the most complete mammalian cell cycle model known to the author that accurately explains time courses of at least one experimentally determined observable.

Yet, the parameters of cell cycle problem version 3.2.0 were structurally unidentifiable and predictions for small molecule or genetic cell cycle perturbations could not be tested in lack of appropriate experimental time course data. Given current data, improving model predictions, as well as solving the unidentifiability issue is possible by including prior knowledge about parameter values when specifying optimisation problems in the future. Such priors may be obtained by categorising reactions into 10 categories: synthesis, degradation, phosphorylation and dephosphorylation reactions (each for baseline and activated/catalysed reactions), as well as complex association and dissociation reactions. Priors may then be obtained by calculating the mean and standard deviation for each category from maximum likelihood estimates to parameterise normal or lognormal priors. This strategy likely results in better predictive power than using the ℓ_1 or ℓ_2 priors typically used in machine learning models. This is because the presented cell cycle model describes (reaction) mechanisms that are well documented in the literature. That means that we can believe that the reactions are there, i.e. priors should have nonzero mean. As a consequence, parameters/reactions will not be eliminated by the regularisation and partially redundant reactions (arising for instance from baseline and activated gene

synthesis or shared phosphorylation substrates of Cdk:cyclin complexes) will be maintained in the model. Retaining redundancy favours large over small models, but may be necessary to accurately predict outcomes of perturbation experiments. This comes not only at the expense of model interpretability, but also renders the use of model selection through Akaike information criterion (AIC) and Bayesian information criterion (BIC) (see Extended Methodological Background Section B.2 for a discussion of model selection methods) impossible, as both criteria only account for model size, not regularisation strength.

Therefore, further improvements will depend on increasing model complexity (e.g. finding a solution to the cell cycle optimisation problem version 4.0.0), acquiring time courses of perturbation experiments, and using cross validation to choose the optimal model and regularisation strength and determine the predictive power of the best model. In addition, the noise model can be refined to multiplicative noise that decreases exponentially over cell cycle time (to compensate for exponentially reduced cell cycle density over a cycle) and is hierarchically optimised for each observable.

5.4 Methods

5.4.1 Multistart and CeSS

Optimisation problems were specified and optimised as described by the software supplement of Villaverde *et al* [86]. For CeSS, the following hyperparameters were chosen: opts.ndiverse='auto'; opts.maxeval=5e6; opts.log_var=[]; opts.local.solver='fmincon'; opts.local.finish='fmincon'; opts.local.bestx=0; opts.local.tol=1; opts.local.n1=100; opts.dim_refset='auto'; opts.prob_bound=0.5; opts.n_threads=8; opts.n_iter=10; opts.maxtime_cess=1*3600. The hyperparameters for threads 1–8 were set to dim1=3; bal1=0; n2_1=4; dim2=5; bal2=0.25; n2_2=7; dim3=5; bal3=0.25; n2_3=10;

dim4=7.5; bal4=0.25; n2_4=15; dim5=10; bal5=0.5; n2_5=20; dim6=12; bal6=0.25; n2_6=50; dim7=15; bal7=0.25; n2_7=100; dim8=15; bal8=0.5; n2_8=10000000000.

5.4.2 saCeSS

All problems were specified in the PEtab format [24] and can be found in the cell_cycle_petab GitHub repository. As saCeSS automatically chooses the best hyperparameters, it was run with default settings. The number of runs (coarse-grained parallelisation) and processors (fine-grained parallelisation) is documented in /versions/PEtab_PLang_problem_v23.xlsx of the cell_cycle_petab GitHub repository.

5.4.3 Data visualisation and illustrations

Data were visualised using Python (v3.10.3) with the matplotlib (v3.5.1) package.

Collaborator contributions

David Rodriguez Penas from the Spanish National Research Council in Vigo (Spain) and Daniel Weindl from the Helmholtz Centre Munich (Germany) added relevant features to the saCeSS solver (parPE local solver, hierarchical optimisation and PEtab support). David Rodriguez Penas submitted the optimisation problems to the Finisterrae II supercomputer, provided by the Galicia Supercomputing Centre (CESGA). Julio Banga from the Spanish National Research Council in Vigo (Spain) provided valuable advice and experience leading to the results presented in sections 5.2.1.3–5.2.2. The author performed the preliminary work leading to the results presented in Section 5.2.1.2. He also defined and debugged all optimisation problems.

6

Summary and Suggestions for Future Work

»What I cannot create, I do not understand.«

Richard Feynman [186]

Contents

6.1 Summary and contributions

The general aim of this thesis was to improve our mechanistic understanding of cell cycle dynamics in form of a computational model that improves on existing models along four main dimensions: model structure, validation data, validation methodology and model reusability. Table 6.1 extends Table 1.1 to directly compare the contributions of this dissertation with previous works. In particular model version 3.2.0 compares favourably to previous efforts when all four dimensions are combined.

Model structure Chapter 3 describes improvements on the structure compared to previous cell cycle models. Starting from small models of isolated cell cycle transitions the present model explains observations of bistability as an emergent property of mass action kinetics. Thanks to the abstraction of rule-based modelling languages, it was possible to introduce a DNA checkpoint and the notion of compartmentalisation. It has been demonstrated that the model can produce continued oscillations, which can be stopped through DNA damage and limitation of growth factor availability. The corresponding cell cycle arrests occur in the expected cell cycle phases. In further agreement with experimental observations, CCNE is elevated in CCNA knockout [147] and G1 arrest [153] conditions, and continued proliferations are sustained under CCNE [145] and CCNA [147] knockout conditions. Beyond this unique combination of qualitative properties, with two compartments, 123 species, 630 reactions and 327 parameters, version 4.0.0 is probably the most comprehensive[1] cell cycle model to date.

[1] Gauthier et Pohl [7] have developed a more fine-grained model, including about a dozen of RNA and protein species per gene.

Table 6.1 | Comparison of this work with existing cell cycle models.

	Model structure					
Model	Organisms	Checkpoints	Compartments	Species	Reactions	Parameters
Singhania et al. (2011)[5]	Mammals	None	CE	4 + 6[1]	16	19
Gauthier et Pohl (2011)[7]	Mammals	None	CP, NU	428	-	332
Weis (2014)[8]	Mammals	None	CE	25	73	136
Abroudi et al. (2017)[11]	Mammals	DNA damage	CE	66	138	150
Mitra et al. (2019)[21]	Yeast	-	CE	44	39	153
Model version 3.2.0	Mammals	DNA damage	NU	73	332	181
Model version 4.0.0	Mammals	DNA damage	CP, NU	123	630	327

continued ...

... continued

	Validation Data			Validation methodology		
Model	Observables	Time points	Conditions	Objective function	Optimiser	Selection criterion
Singhania et al. (2011)	2	>10000	1	-	manual	None
Gauthier et Pohl (2011)	0[2]	1	-	-	manual	None
Weis (2014)	3	~ 40	3	-	manual	None
Abroudi et al. (2017)	0	0	0	-	-	None
Mitra et al. (2019)	10[3]	17-24[4]	1[5]	Least squares[6]	PyBioNetFit[7]	None
Model version 3.2.0	8	300	1	Least squares	saCeSS[10]	None
Model version 4.0.0	0[13]	-	-	-	manual	None

continued ...

... continued

	Model reusability				
Model	Representation	Format	Annotation	Availability	Remarks
Singhania et al. (2011)	ODE/predetermined logical sequence	-	-	-	-
Gauthier et Pohl (2011)	ODEs	Word/ Matlab	None	Supplementary material	a
Weis (2014)	Reactions	SBML[8]	None	BioModels	b
Abroudi et al. (2017)	Reactions	SBML[8,9]	None	BioModels	c
Mitra et al. (2019)	Reactions, Events	SBML	None	GitHub	-
Model version 3.2.0	Rules or Reactions	PEtab or BNGL or SBML	HGNC[11]	GitHub[12]	-
Model version 4.0.0	Rules or Reactions	PEtab or BNGL or SBML	HGNC[11]	GitHub[12]	-

CE: cell, CP: cytoplasm, NU: nucleus.
[1] 4 ODE states, 6 Boolean states.
[2] The model was calibrated to match summary statistics about the cell and the cell cycle.
[3] mRNA instead of protein observables.
[4] Exact number differed between cell cycle synchronisation method.
[5] One condition with time course data plus 119 knock out conditions with qualitative data.
[6] Penalties for qualitative constraint violations were added to the least squares objective.
[7] Using a scatter search algorithm with a simplex local optimizer.
[8] Conversion to SBML performed by Ashley Xavier from the European Bioinformatics Institute.
[9] Simulating the SBML model with zero DNA damage does not reproduce publication results.
[10] Using dynamic hill climbing as local optimizer.
[11] See Section 3.2.6.
[12] Private repository will be made public after publication.
[13] Calibrated to approximate the sequential cell cycle regulator activity known from textbooks.
[a] Very detailed resolution (e.g. 12 mRNA and protein species per gene).
[b] Oscillations are not stable. Simulation of SBML model crashes after 10.8397 time units.
[c] Based on Iwamoto (2011) [12]. Oscillations are not stable.

Validation data Chapter 4 aimed at addressing the scarcity of adequate validation data for cell cycle models. It presents an approach that can determine the abundance and localisation of dozens of cell cycle proteins at a time resolution of less than 5 minutes. If combined with CRISPRi technology, dozens of experimental conditions could be evaluated in one step. Due to delays pertaining to the pandemic, this data is still not available at the time of writing. Yet, using data from Stallaert *et al.* [153] time courses of 292 features in one experimental condition have been obtained, eight of which constituted informative observables for parameter optimisation. Only the validation data for the cell cycle model by Mitra *et al.* [21] contains more observables. However, their model describes the yeast instead of the mammalian cell cycle, and the observables describe mRNA abundances, which are not an ideal proxy for the protein abundances in the model.

Validation method Chapter 5 improves model validation compared to existing mammalian cell cycle models on two levels. First, it introduces a quantifiable objective function. Second, it replaces manual parameter estimation with a state-of-the-art global/local hybrid optimisation algorithm. Taken together, this increases objectivity and reproducibility of model validation.

Model reusability The model is represented in a rule-based format. This concise notation abstracts away unnecessary detail, thus facilitating human readability and extendibility. Rule-based model descriptions also avoid certain types of modelling errors by construction (e.g. ODEs that do not describe reaction networks, reaction networks lacking reactions or species that were intended but not explicitly described by the modeler).

The rule-based model description also allowed a form of intuitive verification of model extensions through comparison of expected versus auto-generated species in the model and expected versus generated number of reactions per rule. From version 3.0.0 onward, all model extensions, including their verification, were tracked with Git version control. This facilitates sustainable and collaborative model development by reducing the chances of introducing errors and increasing the chances of finding them should they still occur.

Combining the BNGL syntax with HGNC short names, a naming convention was introduced to clarify the semantic meaning of model species without adding extensive annotation.

Finally, the model is available in BNGL format and the community standard SBML for software-independent reuse by other researchers. The optimisation problem is available in the modular PEtab format. Any modeler can adapt or exchange the model file to see if the updated or new model better explains the available data. They can also add new experimental conditions or observables to the problem specification, or modify the parameter priors to improve parameter identifiability and reduce the chances of overfitting. As PEtab is supported by multiple optimisers, the presented cell cycle optimisation problem can be easily used by developers of parameter optimisation algorithms to benchmark their method against existing ones.

6.2 Suggested future work and applications

The improvements over existing models stated in the last section do not exhaust all possibilities of currently available data and software. This section will therefore discuss opportunities to further refine the cell cycle model given these constraints,

followed by suggestions for future research and development to break through these constraints. The chapter and this thesis will be concluded with remarks on potential future applications of computational models of the cell cycle and, more generally, biochemical reaction systems.

6.2.1 Model design

While being extensive, even version 4.0.0 does not describe several well established cell cycle reactions and regulators. For instance, the spindle assembly checkpoint and its components are not included. Similarly, there is no notion of cell size control in the present model. In addition, mass action kinetics may not be the most appropriate rate law for all reactions in the model. By including additional mechanisms and trying different rate laws the goodness of fit to existing data can be improved further.

6.2.2 Parameter optimisation

Further refinements can also be made to the optimisation strategy. Minor improvements can be achieved from using a more appropriate description of noise in the optimisation problem specification. More significant improvements can be expected by choosing suitable priors for the model parameters. ℓ_1 priors can be used for model reduction, which will increase model interpretability and reduce redundancies. Reaction type specific priors may be most appropriate to better predict the outcome of perturbation experiments. Introducing priors will reduce overfitting, and using various regularisation strengths will generate ensembles of cell cycle models that differ in their parameter values and, in case of ℓ_1 regularisation, also model complexity. All model versions resulting from inclusion of new mechanisms or variation of regularisation strength can then be evaluated against model selection criteria such as cross-validation.

6.2.3 Data acquisition

Substantial improvements on parameter identifiability and thus predictive power of the model can be expected from incorporation of new experimental data into the optimisation procedure. We are therefore in the process of generating a dataset that increases the number of observables by selecting appropriate antibodies, and increasing the number of experimental conditions by perturbing the cell cycle with small molecule inhibitors and CRISPRi plasmids. Importantly, having time course data for multiple experimental conditions will enable more rigorous testing of the predictive power of the cell cycle model. The data acquisition and parameter optimisation pipeline may also be applied to other healthy and cancer cell lines. The resulting cell line specific models may provide deeper insight into cell cycle regulation in health and disease.

6.2.4 Software development

A key bottleneck towards ever improving models not only of the cell cycle, but biochemistry in general, is the lack of suitable software infrastructure for reusable, sustainable and collaborative model development. Achieving an integrated and formalised understanding of biology requires to extend and combine existing tools and resources to an interoperable open-source Infrastructure for Collaborative Biological Model Development (ICBMD, Figure 6.1). Such an ICBMD may combine three communicating modules: (1) a database of reaction rules (RDB), (2) a database of experimental time course measurements (MDB), and (3) a model construction and optimization toolkit (MCOT) to create mechanistic models of biochemical reaction networks that predict experimentally observed behavior.

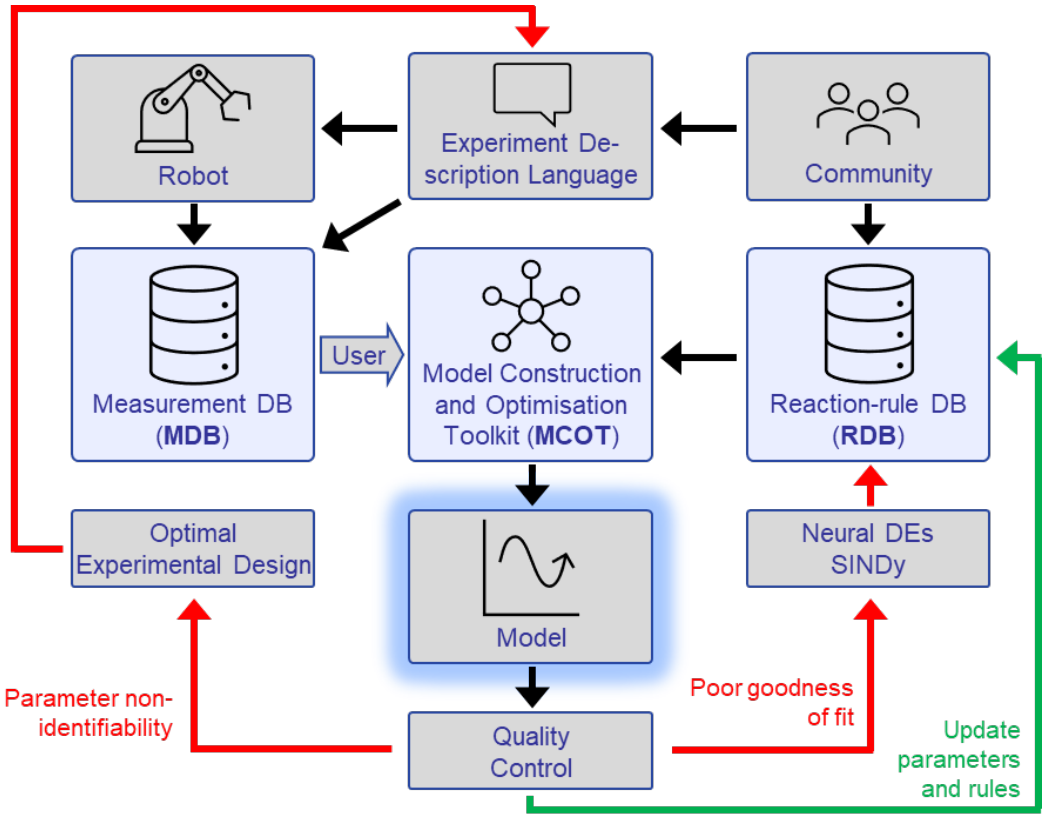


Figure 6.1 | Schematic of an Infrastructure for Collaborative Model Development (ICBMD). The ICBMD core elements (blue boxes) are a reaction-rule database (RDB), a measurement database (MDB) and a model construction and optimization toolkit (MCOT). The RDB can store reaction-rules, kinetic laws and kinetic parameters from previous optimization runs. The MDB can store experimental conditions and corresponding time-course measurements of observables that can be calculated from the chemical species in the RDB. Given experimental conditions and a set of observables, the MCOT can generously extract a large set of reaction-rules that may be required to explain the dynamics of the observables, leading to a simulatable model blueprint. The kinetic parameters of this model are then optimised and potentially driven to zero using regularization techniques to obtain a reduced model. As ICBMD development continues, further features are added. A quality control module checks goodness of fit and parameter identifiability. Poor goodness of fit indicates that the present set of reaction-rules in the RDB is insufficient to explain the dynamics of the particular experimental condition and observables. Therefore, new reaction-rules need to be proposed to the RDB, which can either be done by community members or automatically, using methods like neural differential equations and sparse identification of nonlinear dynamics (SINDy). If quality control passes in the next round, the proposed rules get added permanently to the RDB. Poor parameter identifiability can be resolved by adding more measurements. The needed experiments can be devised by community members or optimal experimental design algorithms and carried out manually or by lab robots. As the ICBMD is increasingly used by community members and machines to construct biological models, data starts to accumulate in the MDB and RDB, allowing ever larger and more accurate models.

The RDB component could be inspired by rule-based model descriptions. By (semi-)automatically pulling raw information from databases like Reactome [187] and converting them into a novel rule-based modelling language like wc_rules[2], the RDB can be fed with an initial set of reaction rules. These rules are associated with kinetic parameters that are progressively learned from the experimental MDB data. The MDB could be inspired by the PEtab format [24]. Importantly, the RDB and MDB have to use the same nomenclature to describe molecules and molecule-complexes. The MCOT would have to perform three main tasks. First, given a set of MDB entries, it will automatically extract potentially relevant reaction rules from the RDB to create an extensive mechanistic rule-based model that shall explain this experimental data. Second, it will fit the kinetic parameters of the model to this data. This will also eliminate superfluous reaction rules using appropriate model reduction methods and model selection criteria. Third, it will add updated kinetic parameter entries to the RDB reaction rules.

Developing an ICBMD would be a complex long-term project that needs to be broken down into several manageable subproblems, for instance:

1. Prototype an RDB for a hypothetical cell containing only five genes to describe the restriction point network (CCND1, CCNE1, CCNE2, E2F1 and RB1), plus a coarse grained description of the cellular environment in which this network operates (ATP synthesis, transcription/translation and degradation machinery). Group CCNE1 and CCNE2 to a CCNE family, nest a transcription and a translation rule within an E2F1 synthesis rule and glycolysis, Krebs cycle and oxidative phosphorylation rules within an ATP synthesis rule.

[2]unpublished work by John Sekar and Jonathan Karr; available on GitHub

2. Proof the concept of an MCOT by considering a full cell cycle model (for instance the one described in my dissertation) and simulated time courses of individual cell cycle transitions in PEtab format. Use L1 regularization to recover the individual cell cycle transition models from the full cell cycle model and cell cycle transition time courses.

3. Create an MDB format by adapting PEtab to be compatible with the RDB from (1) and requiring an experimental protocol that includes certain types of metadata, such as the genome of the used cell line.

4. Create an ICBMD prototype by reformulating and solving the problem from (2) in the RDB and MDB specifications.

5. Develop a web-based user interface for the ICBMD.

6. Advertise the ICBMD to the community. Users will be able to contribute to all three components of the ICBMD: systems biologists can suggest new rules to the RDB, which will be accepted if they improve the predictive power for relevant data in the MDB. Experimental researchers can add measurements to the MDB. Software developers can add new optimization algorithms to the MCOT.

7. Support developers trying to automate contributions to the ICBMD. For instance, if some of the kinetic parameters in the RDB suffer from poor identifiability, optimal experimental design algorithms can determine the best set of experiments to improve identifiability and send the corresponding instructions to a lab robot for execution. Conversely, if simulations are in poor agreement with the data in the MDB, algorithms based on neural differential

equations and sparse identification of nonlinear dynamics [188] can propose new reaction rules to the RDB.

If successful, creating an ICBMD that integrates an RDB, MDB and MCOT could connect biological models like the internet connects computers, enabling the construction of models with unprecedented detail and scope, thus revolutionising our understanding of dynamic biological processes.

6.2.5 Potential impact and applications

Companies like Cytocast, Physiomics, Pumas-AI and Turbine provide services for model-based discovery of druggable targets to accelerate drug development. As one of the most central cellular mechanisms, dysregulation of the cell cycle is associated with a large number of diseases, including cancer and neurodegenerative disorders [189]. Predictive models of cell cycle regulation may therefore be of substantial commercial interest. In absence of data from perturbation experiments, a conclusion about the predictive power of the cell cycle model under perturbed conditions cannot be made. Yet, as soon as such data becomes available, the predictions of the model can be tested. If the predictions are in good agreement with the experimental data of healthy and disease model cell lines, the model may become a valuable contribution to *in silico* drug target discovery. Otherwise, closer inspection of the contradictions may point to gaps or inconsistencies in our present knowledge that guide future research. Incorporating the findings of this research into the cell cycle model and repeatedly testing it against available data will lead to continuous improvements of our mechanistic understanding of cell cycle dynamics.

Appendices

A

Software Projects

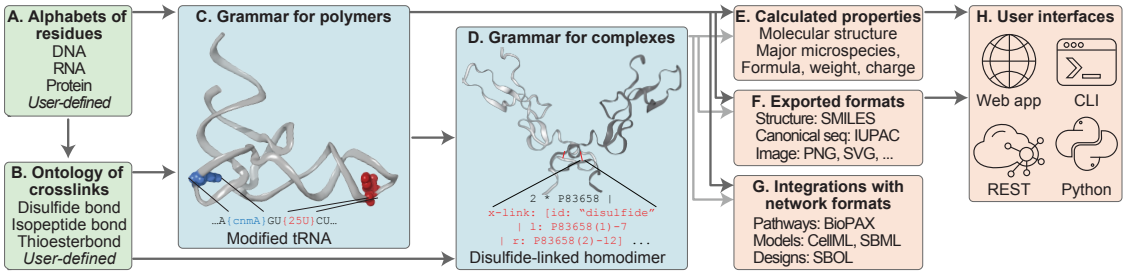
»And even when the apparatus exists, novelty ordinarily emerges only for the man who, knowing with precision what he should expect, is able to recognize that something has gone wrong.«

Thomas Kuhn – The Structure of Scientific Revolutions [190]

Contents

A.1 BpForms and BcForms: a toolkit for concretely describing non-canonical polymers and complexes to facilitate global biochemical networks

Abstract (adapted from [22]) Non-canonical residues, caps, crosslinks, and nicks are important to many functions of DNAs, RNAs, proteins, and complexes. However, we do not fully understand how networks of such non-canonical macromolecules generate behavior. One barrier is our limited formats for describing macromolecules. To overcome this barrier, we develop BpForms and BcForms, a toolkit for representing the primary structure of macromolecules as combinations of residues, caps, crosslinks, and nicks. The toolkit can help omics researchers perform quality control and exchange information about macromolecules, help systems biologists assemble global models of cells that encompass processes such as post-translational modification, and help bioengineers design cells. BpForms and BcForms are freely available under the MIT license. The source codes are available on the BpForms and BcForms GitHub repositories.



Author contributions Paul F Lang, Yassmine Chebaro, Xiaoye Zheng, Darren A Natale, and Jonathan R Karr built the alphabets of residues and the ontology of crosslinks. Xiaoye Zheng, Bilal Shaikh, and Jonathan R Karr developed the software. Xiaoye Zheng, Darren A Natale, and Jonathan R Karr developed the case studies. Paul F Lang, Yassmine Chebaro, and Jonathan R Karr wrote the manuscript.

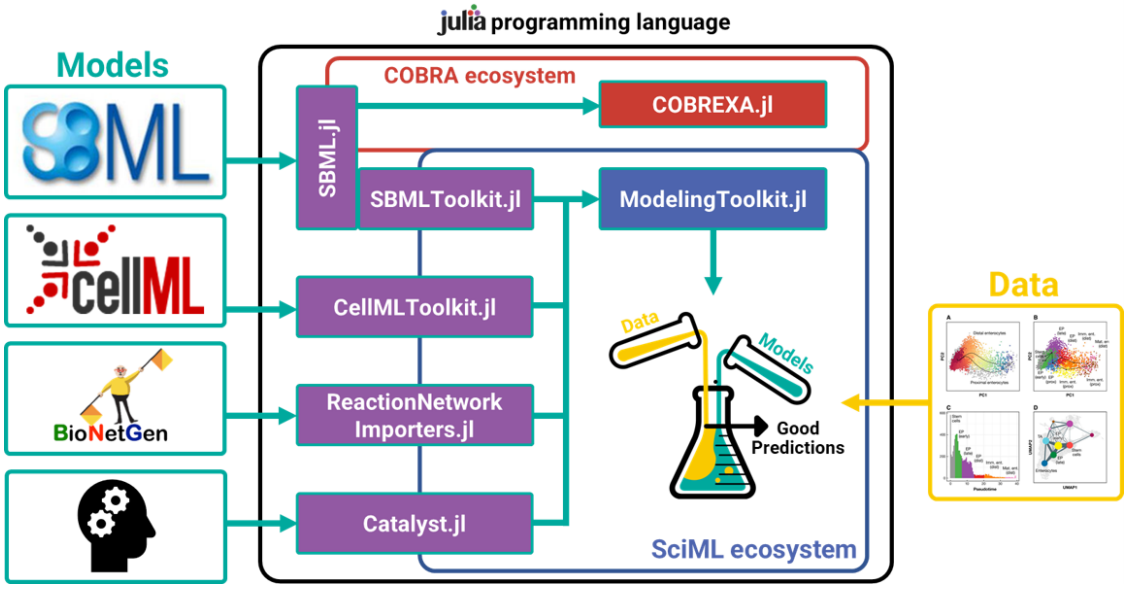
A.2 SBML2Julia: interfacing SBML with efficient nonlinear Julia modelling and solution tools for parameter optimization

Abstract (adapted from [23]) Estimating model parameters from experimental observations is one of the key challenges in systems biology and can be computationally very expensive. While the Julia programming language was recently developed as a high-level and high-performance language for scientific computing, systems biologists have only started to realise its potential. For instance, we have recently used Julia to cut down the optimization time of a microbial community model by a factor of 140. To facilitate access of the systems biology community to the efficient nonlinear solvers used for this optimisation, we developed SBML2Julia. SBML2Julia translates optimisation problems specified in SBML and TSV files (PEtab format) into Julia for Mathematical Programming (JuMP), executes the optimization and returns the results in tabular format. SBML2Julia is freely available under the MIT license. It comes with a command line interface and Python API. Internally, SBML2Julia calls the Julia LTS release v1.0.5 for optimisation. All necessary dependencies can be pulled from Docker Hub. The source code is available on the SBML2Julia GitHub repository.

Author contributions Paul F Lang and Sungho Shin developed the software. Paul F Lang and Victor M Zavala wrote the manuscript.

A.3 SBMLToolkit.jl

Abstract Julia is a general purpose programming language that was designed for simplifying and accelerating numerical analysis and computational science. Building on the computer algebra system Symbolics.jl, many high-performing solvers for constrained systems and differential equations have been developed in Julia. In particular the Scientific Machine Learning (SciML) ecosystem of Julia packages includes ModelingToolkit.jl, a modelling framework for high-performance symbolic-numeric computation in scientific computing and scientific machine learning. It allows for users to give a high-level description of a model for symbolic preprocessing to analyze and enhance the model. ModelingToolkit can automatically generate fast functions for model components, along with automatically sparsifying and parallelising the computations. This enabled highly performing solvers of differential equations, parameter optimisation algorithms and methodologies for automated model discovery. To give the systems biology community easy access to SciML, we developed SBMLToolkit.jl. SBMLToolkit.jl imports dynamic SBML models into the SciML ecosystem to accelerate model simulation and fitting of kinetic parameters. After model import, the SciML ecosystem also enables filling of mechanistic knowledge gaps in epidemiological models using neural networks and sparse identification of nonlinear dynamics. By providing the computational systems biologists with easy access to the open-source Julia ecosystem we hope to catalyse the development of further Julia tools in this domain, and the growth of the Julia bioscience community. SBMLToolkit.jl is freely available under the MIT license. The source code is available on the SBMLToolkit.jl GitHub repository.



Author contributions Paul F Lang and Anand Jain developed SBMLToolkit.jl. Christopher Rackauckas provided valuable advice and supported integration of SBMLToolkit.jl into the SciML ecosystem.

B

Extended Methodological Background

»If you wish to make an apple pie from scratch, you must first invent the universe.«

Carl Sagan – Cosmos: A Personal Voyage [191]

Contents

B.1 Optimisation algorithms

B.1.1 Global search

Global search can be performed deterministically or stochastically. While deterministic methods can guarantee to find global optima for some optimisation problems they scale poorly (often exponentially) to large problems. Stochastic methods on the other hand cannot guarantee global optimality. In practice, however, they often find good or even optimal solutions and are computationally more efficient [192]. This makes stochastic global optimisation methods more attractive for biological problems. We will therefore not further discuss deterministic global methods. Instead, we will describe five frequently used examples of stochastic global optimisation methods that are inspired by physical processes, swarm intelligence, or evolutionary mechanisms.

B.1.1.1 Simulated annealing

Simulated annealing [193] is inspired by the process of slowly cooling down materials, which gives the atoms time to find their energetically optimal position. One can therefore interpret the energy level E as objective function of the system parameters $\boldsymbol{\theta}$. The probability of such a system being in state $\boldsymbol{\theta}$ can be described as a function of the energy level in this state $E(\boldsymbol{\theta})$ by the the Boltzmann distribution

$$P(\boldsymbol{\theta}) \propto e^{-E(\boldsymbol{\theta})/k_B T} \tag{B.1}$$

where

- k_B is the Boltzmann constant, and
- T the temperature.

Note, that when $E(\boldsymbol{\theta})$ is the frequently chosen negative logarithm of the posterior, and $k_B T = 1$, $P(\boldsymbol{\theta})$ is proportional to the posterior probability. Changes in system

state are sampled with the **Metropolis-Hastings algorithm**, starting with high temperature levels and slowly progressing to zero temperature. At the beginning of the optimisation procedure, simulated annealing is therefore more permissive for jumps that increase the objective function value, which allows the algorithm to escape local minima. Towards the end, almost only jumps to lower energy levels will be accepted, allowing the algorithm to converge to a solution. As the chances of escaping local optima depend on the temperature, simulated annealing is sometimes considered to be a gradient-free local optimisation algorithm.

Box B.1: Metropolis-Hastings algorithm

The Metropolis-Hastings algorithm is used to sample from a probability density function. It starts by initialising a point at a random position in state space. Next, the point jumps to a new position in the feature space, with jumping probabilities described by a symmetric jumping distribution (the sequence of points is called Markov chain). The new position is then accepted with a probability of

$$P_{accept} = min\left(1, \frac{P(\boldsymbol{\theta_{new}})}{P(\boldsymbol{\theta_{original}})}\right) \tag{B.2}$$

B.1.1.2 Parallel tempering

Parallel tempering is similar to simulated annealing, except that the temperature is not decreasing over time. Instead, different **markov chains** are run in parallel. To avoid that low temperature chains get stuck in local minima, they can swap parameter values with higher temperature chains. To this end chains are ordered by decreasing temperature. In defined intervals, consecutive chains swap parameter values with a probability of

$$P_{accept} = min(1, e^{(E_{low}-E_{high})(\frac{1}{k_B T_{low}}-\frac{1}{k_B T_{high}})}), \tag{B.3}$$

starting with the highest temperature pair. Parallel tempering can be used for models with up to dozens of parameters [194].

Box B.2: Markov chain

A Markov chain is a sequence of states where the probability of each state exclusively depends on the previous state.

B.1.1.3 Particle swarm optimisation

Particle swarm optimisation is a swarm intelligence algorithm where a population of N candidate solutions $\boldsymbol{\theta_1}, \ldots, \boldsymbol{\theta_N} \in \mathbb{R}^d$, so-called *particles*, is **initialised** with a random position and velocity within a search window. Until a stopping criterion (e.g. certain number of iterations completed or little further improvement of the objective function between iterations) is met, the position of each particle θ_i in iteration j is updated according to

$$\boldsymbol{\theta_{i,j+1}} = \boldsymbol{\theta_{i,j}} + \omega \cdot \boldsymbol{v_{i,j}} + \varphi_p \cdot \boldsymbol{r_{i,p}} \cdot (\boldsymbol{p_i} - \boldsymbol{\theta_{i,j}}) + \varphi_g \cdot \boldsymbol{r_{i,g}} \cdot (\boldsymbol{g} - \boldsymbol{\theta_{i,j}}), \tag{B.4}$$

where

- $\boldsymbol{v_{i,j}}$ describes the velocity,
- $\boldsymbol{p_i}$ the historically best position of the i$^{\text{th}}$ particle,
- $\boldsymbol{g}$ the historically best position of the entire swarm,
- $\omega \in [0, 1]$ the inertia weight,
- $\varphi_p \geq 0$ the cognitive coefficient,
- $\varphi_g \geq 0$ the social coefficient and
- $\boldsymbol{r_{i,p}}$ and $\boldsymbol{r_{i,g}}$ uniformly distributed random variables in $\{0, 1\}^d$.

Similar to simulated annealing, the chances of escaping local optima depend on the hyperparameters of the algorithm. In particle swarm optimisation local search can be intensified by reducing the inertia weight ω [195]. Therefore, particle swarm optimisation can also be used for gradient-free local optimisation.

Box B.3: Initialisation methods

- **Random sampling:** all samples are chosen at random.
- **Latin hypercube sampling:** To sample N points in Latin hypercube sampling, each dimension of the search window is divided into N bins. Next, points are chosen at random, with the restriction that every bin of any dimension contains exactly one point. One a two-dimensional $N \times N$ chess board this would correspond to placing N rooks on the board such that they do not threaten each other. Latin hypercube sampling ensures that all dimensions are sampled more evenly.
- **Orthogonal sampling:** Orthogonal sampling extends Latin hypercube sampling by dividing the search space into equally probable subspaces (all larger than one hypercube on the Latin hypercube grid) and constraining the sampling so that all subspaces contain the same number of points.

B.1.1.4 Differential evolution

Differential evolution belongs to the family of evolutionary algorithms, which use the principles of recombination, evaluation and selection. First, a population of n agents $\boldsymbol{\theta_1}, \ldots, \boldsymbol{\theta_n} \in \mathbb{R}^d$ is chosen as initial population. Next, for each agent $\boldsymbol{\theta_i}$ three further and different agents $\boldsymbol{\theta_a}$, $\boldsymbol{\theta_b}$ and $\boldsymbol{\theta_c}$ are chosen. These three agents are recombined to generate a candidate agent. The candidate agent replaces the original agent if the former has a lower objective function value. In this way, differential evolution iteratively improves the population (Algorithm B.1). By adding the weighted difference between two particles two population members to

a third, differential evolution samples in directions of the spread of the current population. Combined with selection, this procedure self-organises the population to explore known areas of interest [196].

Algorithm B.1 Differential evolution

Input: $\{\boldsymbol{\theta_1}, \ldots, \boldsymbol{\theta_n}\}$ ▷ Initial population
Input: $F \in [0, 2]$ ▷ Differential weight
Input: $c \in [0, 1]$ ▷ Crossover rate
Output: $\boldsymbol{\theta_{best}}$

```
repeat
    for θ_i ∈ {θ_1, ..., θ_n} do
        Pick three different agents θ_a, θ_b, θ_c from {θ_1, ..., θ_n}
        Pick a random dimension R ∈ {1, ..., d}
        for j ∈ {1, ··· , d} do        ▷ Generate a proposed solution s ∈ ℝ^d
            Pick r_j from 𝒰(0, 1)
            if r_j < c or j = R then
                s_j ← θ_{a_j} + F · (θ_{b_j} − θ_{c_j})
            else
                s_j ← θ_{a_j}
            end if
        end for
        if J(s ≤ J(θ_i)) then
            θ_i ← s
        end if
    end for
    θ_best ← argmin_i θ_i
until Stop condition
```

B.1.2 Local search

Local optimisation methods typically converge faster than global optimisation methods, but may get stuck in local optima. To find local optima, they need to find a search direction and a step size. Local search algorithms can therefore be categorised by the order in which these are determined, which we will describe in

paragraph *Search strategies*. They can also be classified by the number of objective function derivatives they exploit, as will be discussed in the paragraphs *Gradient-free methods* and *Gradient-based methods*. Finally, local optimisation methods can be provided with derivatives that were calculated in several ways, which will be discussed in paragraph *Derivative calculation*.

B.1.2.1 Search strategies

Line-search methods Line-search methods first determine the direction $\boldsymbol{s}$ of the next step, and then compute an optimal step size γ that minimises the objective function (Algorithm B.2). This reduces the potentially high-dimensional optimisation problem to a one-dimensional optimisation problem [78]. Figure B.1 illustrates a line-search method that uses a quadratic approximation of the objective function.

Algorithm B.2 Line search algorithm

$k \leftarrow 0$
repeat
  Compute descent direction $\boldsymbol{s}_k$
  Compute $\gamma_k \geq 0$ to loosely minimise $f(\gamma) = J(\boldsymbol{\theta}_k + \gamma_k \boldsymbol{s})$
  $\boldsymbol{\theta}_{k+1} \leftarrow \boldsymbol{\theta}_k + \gamma_k \boldsymbol{s}_k$
  $k \leftarrow k + 1$
until Stop condition

Trust-region methods Trust region methods construct an approximate model m of the objective function around the current iteration. They continuously evaluate how well the approximate model fits the objective function. The region where the approximation is good enough is called trust region. The radius Δ_k of the trust region is continuously adapted based on this evaluation. A typical choice is to monitor how

well the algorithm can predict the reduction of the objective function in each step:

$$\rho_k = \frac{J(\boldsymbol{\theta}_k) - J(\boldsymbol{\theta}_k + \boldsymbol{s}_k)}{m(\mathbf{0}) - m(\boldsymbol{s}_k)}. \tag{B.5}$$

Once the radius of the trust region is determined, the optimal step is calculated as

$$\begin{aligned} \min_{\boldsymbol{s}} \quad & m_k(\boldsymbol{s}) \\ \text{s.t.} \quad & ||\boldsymbol{s}|| \leq \Delta_k. \end{aligned} \tag{B.6}$$

Algorithm B.3 [197] shows how trust-region methods iterate through adaptation of the trust-region radius and computation of the search step. Figure B.1 compares trust-region and line-search methods using a quadratic approximation model m for the objective function J.

Algorithm B.3 Trust-region algorithm

Input: $\Delta_{max} > 0$, $\Delta_0 \in (0, \Delta_{max})$, $\eta \in [0, \frac{1}{4})$
  $k \leftarrow 0$
  repeat
    Compute $\boldsymbol{s}_k$ according to (B.6)
    Compute ρ_k from eq:rho
    if $\rho_k < \frac{1}{4}$ **then**
      $\Delta_{k+1} \leftarrow \frac{1}{4}\Delta_k$
    else if $\rho_k > \frac{3}{4}$ and $||p_k|| = \Delta_k$ **then**
      $\Delta_{k+1} \leftarrow \min(2\Delta_k, \Delta_{max})$
    else
      $\Delta_{k+1} \leftarrow \Delta_k$
    end if
    if $\rho_k > \eta$ **then**
      $\boldsymbol{\theta}_{k+1} \leftarrow \boldsymbol{\theta}_k + \boldsymbol{s}_k$
    else
      $\boldsymbol{\theta}_{k+1} \leftarrow \boldsymbol{\theta}_k$
    end if
    $k \leftarrow k + 1$
  until Stop condition

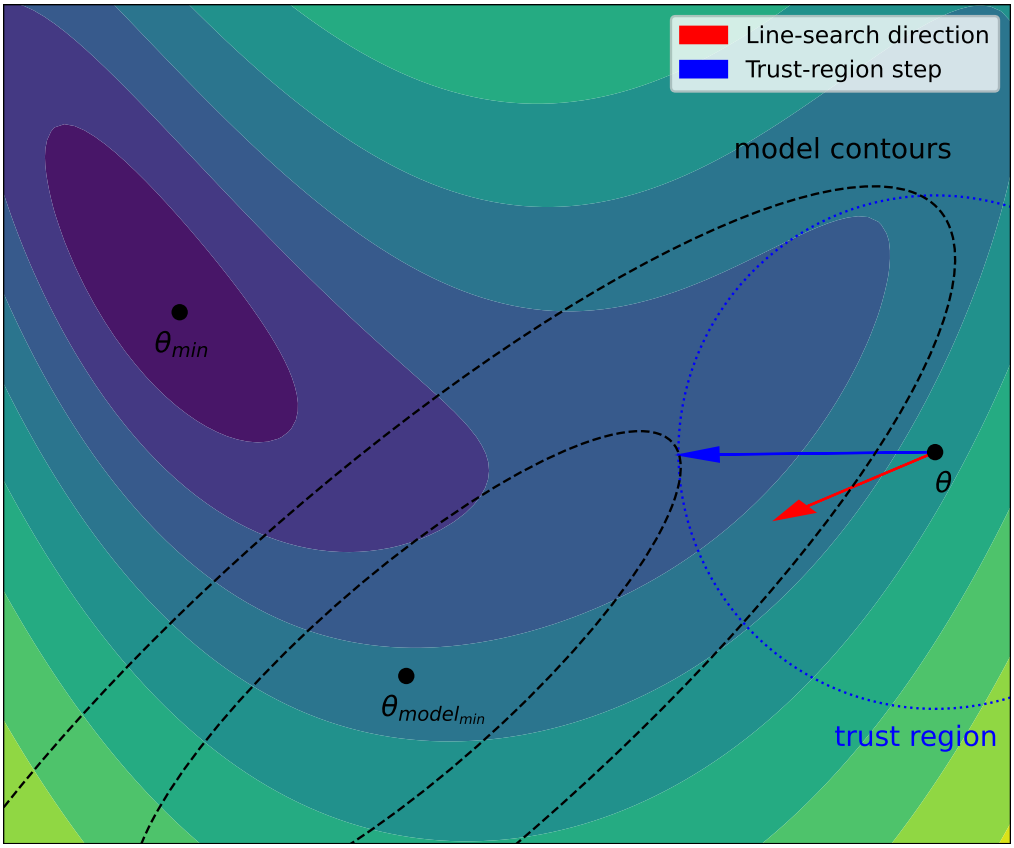


Figure B.1 | Comparison between line-search and trust-region search. The objective function (colored contours) is approximated with a quadratic model function (black-dashed contours) around the current point $\boldsymbol{\theta}$. Line-search tries to optimise the step size into direction of the minimum of the approximation $\boldsymbol{\theta}_{\boldsymbol{model}_{\boldsymbol{min}}}$. Trust-region search tries to find the minimum point of the approximation within a so called trust-region, where the approximation holds sufficiently well.

B.1.2.2 Gradient-free methods

Gradient-free methods are used when gradients are impossible to calculate. Some gradient-free methods like the inner loop of dynamic hill climbing (DHC) may also be competitive when gradients are numerically difficult to evaluate [86]. As the name suggests, the inner loop of DHC [198] adaptively modifies step size and search direction until a satisfactory solution is found (Algorithm B.4). The outer loop is a global optimisation technique chosen by the user.

Algorithm B.4 Dynamic hill climbing (inner loop)

Input: Initial point $\boldsymbol{\theta}$
Input: Orthogonal basis of the search space $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_d\}$
Input: *thresh*

$\boldsymbol{v}_{n+1}, \ldots, \boldsymbol{v}_{2d} \leftarrow \{-\boldsymbol{v}_1, \ldots, -\boldsymbol{v}_d\}$
$\boldsymbol{v}_{2d+1} \leftarrow \mathbf{0}$
$\boldsymbol{v}_{2d+2} \leftarrow \mathbf{0}$
$\boldsymbol{s}_1 \leftarrow \mathbf{0}$
$\boldsymbol{s}_2 \leftarrow \mathbf{0}$
$f \leftarrow ObjectiveFunction(\boldsymbol{\theta})$
repeat
  $l \leftarrow$ index of longest vector in $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{2d+2}\}$
  $\boldsymbol{\theta}_{next} \leftarrow \boldsymbol{\theta} + \boldsymbol{v}_l$
  $f_{next} \leftarrow ObjectiveFunction(\boldsymbol{\theta}_{next})$
  if $f_{next} < f$ **then**
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}_{next}$
    $f \leftarrow f_{next}$
    $\boldsymbol{v}_l \leftarrow 2 \cdot \boldsymbol{v}_l$
    if $\frac{\boldsymbol{v}_l}{\|\boldsymbol{v}_l\|} \neq \frac{\boldsymbol{s}_1}{\|\boldsymbol{s}_1\|}$ **then**
      $\boldsymbol{s}_2 \leftarrow \boldsymbol{s}_1$
      $\boldsymbol{s}_1 \leftarrow \boldsymbol{v}_l$
      $\boldsymbol{v}_{2d+1} \leftarrow \boldsymbol{s}_1 + \boldsymbol{s}_2$
      $\boldsymbol{v}_{2d+2} \leftarrow -\boldsymbol{v}_{2d+1}$
    end if
  else
    $\boldsymbol{v}_l \leftarrow \boldsymbol{v}_l/2$
  end if
until $\|\boldsymbol{v}_i\| \leq thresh \quad \forall\, i \in \{1 \,..\, 2d+2\}$

B.1.2.3 Gradient-based methods

Gradient-based methods are applicable and recommended when the objective function is continuously differentiable [78]. The calculated gradients can be used in line-search and trust-region methods. In this section we will assume that the objective function $J(\boldsymbol{\theta})$ is at least twice continuously differentiable in the region of attraction.

Gradient descent In gradient descent methods the search direction is given by the direction of steepest descent:

$$\boldsymbol{s_{grad}} = -\nabla J(\boldsymbol{\theta}). \tag{B.7}$$

However, moving in direction of $\boldsymbol{s_{grad}}$ may only decrease the objective function J for small step sizes and thus require a large number of iterations. This is for instance the case in the vicinity of saddle points of curved ridges.

Newton methods The issue of low step sizes can be addressed with the Newton method for finding function roots. Here, we are not interested in the roots of the objective function J, but instead in locations of zero gradient. Writing J as Taylor series to second order, we get

$$J(\boldsymbol{\theta} + \boldsymbol{s}_{newt}) \approx J(\boldsymbol{\theta}) + \boldsymbol{s}_{newt}^T \nabla J(\boldsymbol{\theta}) + \frac{1}{2!} \boldsymbol{s}_{newt}^T \nabla^2 J(\boldsymbol{\theta}) \boldsymbol{s}_{newt}, \tag{B.8}$$

where

- $\nabla^2 J(\boldsymbol{\theta})$ is the Hessian of J.

Setting the first derivative of $J(\boldsymbol{\theta} + \boldsymbol{s}_{newt})$ with respect to $\boldsymbol{\theta}$ to zero and ignoring derivatives of third or higher order we obtain

$$\mathbf{0}^T \approx \nabla J(\boldsymbol{\theta})^T + \boldsymbol{s}_{newt}^T \nabla^2 J(\boldsymbol{\theta}), \tag{B.9}$$

and rearranging for $\boldsymbol{s}_{newt}$ we get

$$\boldsymbol{s}_{newt} \approx -\nabla^2 J(\boldsymbol{\theta})^{-1} \nabla J(\boldsymbol{\theta}). \tag{B.10}$$

By using second-order derivative information, the Newton method circumvents the problem of small step sizes that can arise in gradient descent methods. In line-search the default value for γ is 1. However, the Newton method requires the computation of the Hessian, which is computationally expensive for most ODE models. Furthermore, as Newton methods search for zero gradients as necessary but not sufficient condition for local minima, the Newton step $\boldsymbol{s}_{newt}$ may not always be in a descent direction. Only if

$$\boldsymbol{s}_{newt}^T \nabla J(\boldsymbol{\theta}) < 0 \tag{B.11}$$

does the Newton step go into a descent direction. Substituting (B.10) in (B.11) we obtain

$$-\nabla J(\boldsymbol{\theta})^T \nabla^2 J(\boldsymbol{\theta})^{-1} \nabla J(\boldsymbol{\theta}) < 0. \tag{B.12}$$

This condition is satisfied if the inverse of the Hessian $\nabla^2 J(\boldsymbol{\theta})^{-1}$ is positive definite for all $\boldsymbol{\theta}$, i.e. J is convex.

Damping By adding a large enough multiple of the identity matrix $\boldsymbol{I}$, the Hessian can be converted to a positive definite matrix $(\nabla^2 J(\boldsymbol{\theta})^{-1} + \lambda \boldsymbol{I})$. This procedure is called damping, and geometrically corresponds to adding $\lambda \geq 0$ times the gradient descent step to the Newton step:

$$\begin{aligned} \boldsymbol{s}_{newt} + \lambda \boldsymbol{s}_{grad} &\approx -\nabla^2 J(\boldsymbol{\theta})^{-1} \nabla J(\boldsymbol{\theta}) - \lambda \nabla J(\boldsymbol{\theta}), \\ \boldsymbol{s} &\approx -\left(\nabla^2 J(\boldsymbol{\theta})^{-1} + \lambda \boldsymbol{I}\right) \nabla J(\boldsymbol{\theta}). \end{aligned} \tag{B.13}$$

In trust-region methods λ may also be chosen such that $||\gamma \boldsymbol{s}|| = \Delta$.

Gauss newton method The Gauss-Newton method exploits the least-squares that is typical for many objective functions to approximate the Hessian. It uses only first order derivative information and always results in a positive semi-definite approximation of the Hessian. Therefore, the Gauss-Newton method is not only computationally efficient, but also avoids steps in ascend directions. Writing the objective function J as the sum of n squared residuals

$$J(\theta) = \sum_{i=1}^{n} r_i(\theta)^2, \tag{B.14}$$

the gradient can be calculated using chain rule

$$\nabla J_j = 2 \sum_{i=1}^{n} r_i \frac{\partial r_i}{\partial \theta_j}, \tag{B.15}$$

and the Hessian using product rule

$$\nabla^2 J_{jk} = 2 \sum_{i=1}^{n} \left(\frac{\partial r_i}{\partial \theta_j} \frac{\partial r_i}{\partial \theta_k} + r_i \frac{\partial^2 r_i}{\partial \theta_j \partial \theta_k} \right) \tag{B.16}$$

The Gauss-Newton approximation of the Hessian now assumes small residuals r_i (at least close to the minimum) and only mild nonlinearity $\frac{\partial^2 r_i}{\partial \theta_j \partial \theta_k}$, so that

$$r_i \frac{\partial^2 r_i}{\partial \theta_j \partial \theta_k} \ll \frac{\partial r_i}{\partial \theta_j} \frac{\partial r_i}{\partial \theta_k}, \tag{B.17}$$

and thus

$$\nabla^2 J_{jk} \approx 2 \sum_{i=1}^{n} \frac{\partial r_i}{\partial \theta_j} \frac{\partial r_i}{\partial \theta_k} \tag{B.18}$$

or in matrix form

$$\nabla^2 J \approx 2 \nabla_\theta \boldsymbol{R}(\boldsymbol{\theta})^T \nabla_\theta \boldsymbol{R}(\boldsymbol{\theta}) \tag{B.19}$$

where

- $\nabla_\theta \boldsymbol{R}(\boldsymbol{\theta})$ is the Jacobian of $\boldsymbol{r}$ with respect to $\boldsymbol{\theta}$.

The Gauss-Newton approximated Hessian is the product of a transposed matrix with itself, which is always positive semidefinite:

$$\boldsymbol{x}^T\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{x} = (\boldsymbol{A}\boldsymbol{x})^T\boldsymbol{A}\boldsymbol{x} = ||\boldsymbol{A}\boldsymbol{x}||^2 \geq 0, \text{ for } \boldsymbol{x} \neq \boldsymbol{0}. \tag{B.20}$$

This ensures that Gauss-Newton steps will not go in ascend directions as long as the Gauss-Newton approximation holds.

Substituting the Jacobian $\nabla_\theta \boldsymbol{R}(\boldsymbol{\theta})$ in (B.15), we can write the gradient as

$$\nabla J(\boldsymbol{\theta}) = 2\nabla_\theta \boldsymbol{R}(\boldsymbol{\theta})^T \boldsymbol{r}(\boldsymbol{\theta}). \tag{B.21}$$

The whole Gauss-Newton step is then obtained by substituting (B.19) and (B.21) in (B.10) to obtain

$$\boldsymbol{s}_{newt} \approx -\left(\nabla_\theta \boldsymbol{R}(\boldsymbol{\theta})^T \nabla_\theta \boldsymbol{R}(\boldsymbol{\theta})\right)^{-1} \nabla_\theta \boldsymbol{R}(\boldsymbol{\theta})^T \boldsymbol{r}(\boldsymbol{\theta}). \tag{B.22}$$

B.1.2.4 Derivative calculation

For ODE problems, the objective function (2.14) depends on the observation function (2.15) and the solution of the ODE problem (2.16). As analytic solutions to ODE problems describing biochemical reaction networks rarely exist, the evaluation of the objective function and its gradients depends on the numerical simulation of the ODE problem (2.16). This section will discuss three ways of calculating gradients in this setting: finite differences, forward-sensitivity analysis, and adjoint sensitivity analysis.

Finite differences A very intuitive way of calculating gradients is to simulate the ODE problem with small deviations for every single parameter and evaluate the objective function. The elements of the gradient can then be approximated as:

$$\frac{J}{d\theta_k} \approx \frac{J(\boldsymbol{\theta} + a\boldsymbol{e}_k) - J(\boldsymbol{\theta} - b\boldsymbol{e}_k)}{a + b} \tag{B.23}$$

with $a, bgeq0$ and the k$^{\text{th}}$ unit vector $\boldsymbol{e}_k$. We can distinguish forward ($a > 0, b = 0$), backward ($a = 0, b > 0$) and central differences ($a > 0, b > 0$). However, the evaluation of $J(\boldsymbol{\theta} + a\boldsymbol{e}_k) - J(\boldsymbol{\theta} - b\boldsymbol{e}_k$ requires additional simulation of the ODE model. Therefore, the computational complexity of evaluating finite differences is proportional to the number of parameters [78].

Forward sensitivities A more involved, yet more informative approach is to derive an analytical expression for the parameter derivative of the system states, that can be used for calculating the objective function gradient. We can define a sensitivity matrix as

$$\boldsymbol{S}_k = \frac{d\boldsymbol{x}(t_k, \boldsymbol{\theta})}{d\boldsymbol{\theta}}, \tag{B.24}$$

where $\boldsymbol{S}_{k_{ij}}$ describes the derivative of the i$^{\text{th}}$ state at time point t_k with respect to the j$^{\text{th}}$ parameter. That is, the sensitivities change over time, which can be described as:

$$\dot{\boldsymbol{S}}(t) = \frac{d\dot{\boldsymbol{x}}(t, \boldsymbol{\theta})}{d\boldsymbol{\theta}}. \tag{B.25}$$

Substituting $\dot{\boldsymbol{x}}(t, \boldsymbol{\theta}) = f(\boldsymbol{x}(t, \boldsymbol{\theta}), \boldsymbol{\theta})$ (Equation (2.16)) into (B.25) we obtain

$$\dot{\boldsymbol{S}}(t) = \frac{df(\boldsymbol{x}(t, \boldsymbol{\theta}), \boldsymbol{\theta})}{d\boldsymbol{\theta}}, \tag{B.26}$$

or using chain rule

$$\dot{\boldsymbol{S}}(t) = \frac{\partial f(\boldsymbol{x}(t, \boldsymbol{\theta}), \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} + \frac{\partial f(\boldsymbol{x}(t, \boldsymbol{\theta}), \boldsymbol{\theta})}{\partial \boldsymbol{x}(t, \boldsymbol{\theta})} \frac{d\boldsymbol{x}(t, \boldsymbol{\theta})}{d\boldsymbol{\theta}}, \tag{B.27}$$

or

$$\dot{\boldsymbol{S}}(t) = \frac{\partial f(\boldsymbol{x}(t, \boldsymbol{\theta}), \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} + \frac{\partial f(\boldsymbol{x}(t, \boldsymbol{\theta}), \boldsymbol{\theta})}{\partial \boldsymbol{x}(t, \boldsymbol{\theta})} \boldsymbol{S}(t). \tag{B.28}$$

As the initial state does not depend on the kinetic parameters

$$\boldsymbol{S}(0) = \frac{d\boldsymbol{x}(t_k, \boldsymbol{\theta})}{d\boldsymbol{\theta}} = \boldsymbol{0}. \tag{B.29}$$

The resulting initial value problem for the sensitivities can be integrated alongside the initial value problem for the states. The state derivative $\frac{\partial f(\boldsymbol{x}(t,\boldsymbol{\theta}))}{\partial \boldsymbol{\theta}}$ and the Jacobian $\frac{\partial f(\boldsymbol{x}(t,\boldsymbol{\theta}))}{\partial \boldsymbol{x}(t,\boldsymbol{\theta})}$ can be computed ahead of time.

For an objective function of the form

$$J(h(\boldsymbol{x}(t,\boldsymbol{\theta}),\boldsymbol{\theta}),\boldsymbol{\theta}) = \sum_{i=1}^{n_x}\sum_{k=1}^{n_t} J(h(\boldsymbol{x_i}(t_k,\boldsymbol{\theta}),\boldsymbol{\theta}),\boldsymbol{\theta}) \tag{B.30}$$

the gradient is then calculated as

$$\nabla J = \sum_{i=1}^{n_x}\sum_{k=1}^{n_t} \frac{\partial J}{\partial h}\frac{\partial h}{\partial x_i(t_k)}\frac{\partial x_i(t_k)}{\partial \boldsymbol{\theta}} + \frac{\partial J}{\partial h}\frac{\partial h}{\partial \boldsymbol{\theta}} + \frac{\partial J}{\partial \boldsymbol{\theta}}, \tag{B.31}$$

where n_x and n_t denote the number of species and time points, respectively. Note that the number of equations in the ODE system in Equation (B.28) increases linearly with the number of parameters and the computational complexity of simulating ODE systems depends linearly on the number of equations [78]. Like finite differences, forward-sensitivity analysis scales therefore scales linearly with the number of states. However, forward-sensitivity analysis not only tells us the gradient of the objective function, but also the contribution of every single data point to that gradient.

Adjoint sensitivity analysis Adjoint-sensitivity analysis avoids the costly calculation of forward-sensitivities by introducing a state that is chosen such that the terms containing forward-sensitivities cancel out. It is the continuous analog to backpropagation through multiple layers in neural networks. As observation parameters are not optimised in adjoint sensitivity analysis, we will here assume that the observation function is $h(\boldsymbol{x},\boldsymbol{\theta}) = \boldsymbol{x}$ for better readability of the derivation. Let us therefore consider the simple objective function

$$\begin{aligned} \min_{\boldsymbol{\theta}} \quad & \int_0^T J(\boldsymbol{x}(t))\,dt \\ \text{s.t.} \quad & \dot{\boldsymbol{x}} = f(\boldsymbol{x},\boldsymbol{\theta}), \end{aligned} \tag{B.32}$$

where T denotes the simulation end time. We can now introduce the adjoint state $\boldsymbol{\lambda}(t)$ to obtain

$$J = \int_0^T J(\boldsymbol{x})\, dt + \int_0^T \boldsymbol{\lambda}(t) \cdot \left(\underbrace{\dot{\boldsymbol{x}} - f(\boldsymbol{x}, \boldsymbol{\theta})}_{=0} \right) dt \tag{B.33}$$

Taking derivatives with respect to $\boldsymbol{\theta}$

$$\frac{dJ}{d\boldsymbol{\theta}} = \int_0^T \frac{\partial J(\boldsymbol{x})}{\partial \boldsymbol{x}} \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\theta}}\, dt + \int_0^T \boldsymbol{\lambda}(t) \cdot \left(\underbrace{\frac{\partial \dot{\boldsymbol{x}}}{\partial \boldsymbol{\theta}} - \frac{\partial f(\boldsymbol{x}, \boldsymbol{\theta})}{\partial \boldsymbol{x}} \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\theta}} - \frac{\partial f(\boldsymbol{x}, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}}}_{=0} \right) dt,$$

expanding the integral to

$$\begin{aligned} \frac{dJ}{d\boldsymbol{\theta}} = &\int_0^T \frac{\partial J(\boldsymbol{x})}{\partial \boldsymbol{x}} \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\theta}}\, dt + \int_0^T \boldsymbol{\lambda}(t) \frac{\partial \dot{\boldsymbol{x}}}{\partial \boldsymbol{\theta}}\, dt \\ &- \int_0^T \boldsymbol{\lambda}(t) \frac{\partial f(\boldsymbol{x}, \boldsymbol{\theta})}{\partial \boldsymbol{x}} \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\theta}}\, dt - \int_0^T \boldsymbol{\lambda}(t) \frac{\partial f(\boldsymbol{x}, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\, dt, \end{aligned} \tag{B.34}$$

and integrating the second integral by parts, we get:

$$\begin{aligned} \frac{dJ}{d\boldsymbol{\theta}} = &\int_0^T \frac{\partial J(\boldsymbol{x})}{\partial \boldsymbol{x}} \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\theta}}\, dt + \left[\boldsymbol{\lambda}(t) \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\theta}} \right]_0^T - \int_0^T \dot{\boldsymbol{\lambda}}(t) \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\theta}}\, dt \\ &- \int_0^T \boldsymbol{\lambda}(t) \frac{\partial f(\boldsymbol{x}, \boldsymbol{\theta})}{\partial \boldsymbol{x}} \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\theta}}\, dt - \int_0^T \boldsymbol{\lambda}(t) \frac{\partial f(\boldsymbol{x}, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\, dt. \end{aligned} \tag{B.35}$$

The costly calculation of forward sensitivities $\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\theta}}$ can now be eliminated by choosing

$$\begin{aligned} \frac{\partial J}{\partial \boldsymbol{x}} := \dot{\boldsymbol{\lambda}}(t) + \boldsymbol{\lambda}(t) \frac{\partial f(\boldsymbol{x}, \boldsymbol{\theta})}{\partial \boldsymbol{x}}, \text{ and} \\ \boldsymbol{\lambda}(T) := \mathbf{0} \end{aligned} \tag{B.36}$$

to arrive at

$$\frac{dJ}{d\boldsymbol{\theta}} = \boldsymbol{\lambda}(0) \frac{\partial \boldsymbol{x}(0)}{\partial \boldsymbol{\theta}} - \int_0^T \boldsymbol{\lambda}(t) \frac{\partial f(\boldsymbol{x}, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\, dt, \tag{B.37}$$

where $\frac{\partial \boldsymbol{x}(0)}{\partial \boldsymbol{\theta}} = \mathbf{0}$ as the derivative of the initial state does not depend on the kinetic parameters, and therefore

$$\frac{dJ}{d\boldsymbol{\theta}} = -\int_0^T \boldsymbol{\lambda}(t) \frac{\partial f(\boldsymbol{x}, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\, dt. \tag{B.38}$$

The initial value problem in Equation (B.36) is called adjoint problem and contains the same number of ODEs as the original problem. The Jacobian $\frac{\partial f(\boldsymbol{x}, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$ can

be calculated ahead of time and the and $\frac{\partial J}{\partial \boldsymbol{x}}$ is easily calculated from the form of the objective function. Evaluating (B.38) scales linearly with the number of parameters, but can be done very efficiently. As the computational complexity of the time consuming steps of integrating the original model and the adjoint problem (the latter in reverse direction) are independent of the number of parameters, the computational cost of evaluating the model with gradients is only about twice the cost of evaluating just the model [78].

Importantly, adjoint sensitivity analysis can also be combined with hierarchical optimisation (section 2.3.3.2). The separation of Equation (2.19) in Equations (2.20) and (2.21) then allows to iterate through optimising the scaling, offset and noise parameters $\boldsymbol{s}$, $\boldsymbol{b}$ and $\boldsymbol{\sigma}$, respectively, and the remaining parameters $\boldsymbol{\theta}_r$ in sequence (Algorithm 2.2). This separation also allows the use of adjoint sensitivity analysis, as long as none of the remaining parameters $\boldsymbol{\theta}_r$ participate directly in the observation function $\tilde{h}$. While we have ignored an observation function in the derivation of adjoint sensitivity analysis (Equations (B.32)–(B.38)) for better readability, we see in Equation (B.36) that we need to know the gradient of the objective function with respect to the states $\frac{\partial J}{\partial \boldsymbol{x}}$ to calculate the adjoint state $\boldsymbol{\lambda}(t)$. In general, this gradient depends on the measurement noise and the observation function h. If the measurement noise and all parameters of the observation function are analytically obtained through Equations (2.23)–(2.25), the gradient $\frac{\partial J}{\partial \boldsymbol{x}}$ is known and adjoint sensitivity analysis can be performed.

B.2 Model selection methods

This section will discuss three metrics to compare various models against each other in order to select the best model with respect to the chosen metric.

B.2.1 Cross-validation error

The objective function provides a metric to assess the goodness of fit to training data. However, to avoid overfitting it is necessary to evaluate the model based on test data. A straight forward approach to do this is to randomly split the dataset into two parts, one for training, and one for testing. After the model parameters where estimated using the training data, the mean squared error (MSE) is evaluated on the test data

$$MSE = \sum_{i-1}^{n_{test}} \left(\frac{\bar{y_i} - y_i(\boldsymbol{\theta})}{\sigma_i} \right)^2, \tag{B.39}$$

where n_{test} is the number of datapoints in the test set. However, this splitting has two disadvantages. First, a lot of data cannot be used for training, leading to worse models and thus overestimation of the MSE. Second, the MSE can vary substantially between different data splits [91]. Therefore, a more popular procedure is k-fold cross validation. In k-fold cross validation, a dataset of n datapoints is split into $k \leq n$ approximately equally sized subsets. Each datapoint is member of exactly one subset. For each subset i, the model parameters are then estimated using all other subsets and the MSE is evaluated on subset i (Figure B.2). The cross-validation error is then calculated as the mean MSE across all iterations. If $k = 1$ the method is called leave-one-out cross-validation. As k increases, the model fits become approach the accuracy of fitting to all n datapoints. Therefore, the cross-validation error becomes less biased towards high values. However, increasing k increases the variance of the cross-validation error between replicate datasets. As a rule of thumb choosing k between 5 and 10 optimises the bias-variance trade-off.

B.2.2 Akaike Information Criterion

Cross-validation is a direct metric for assessing the test error. However, it is computationally expensive, as k-fold cross-validation requires k separate optimisation

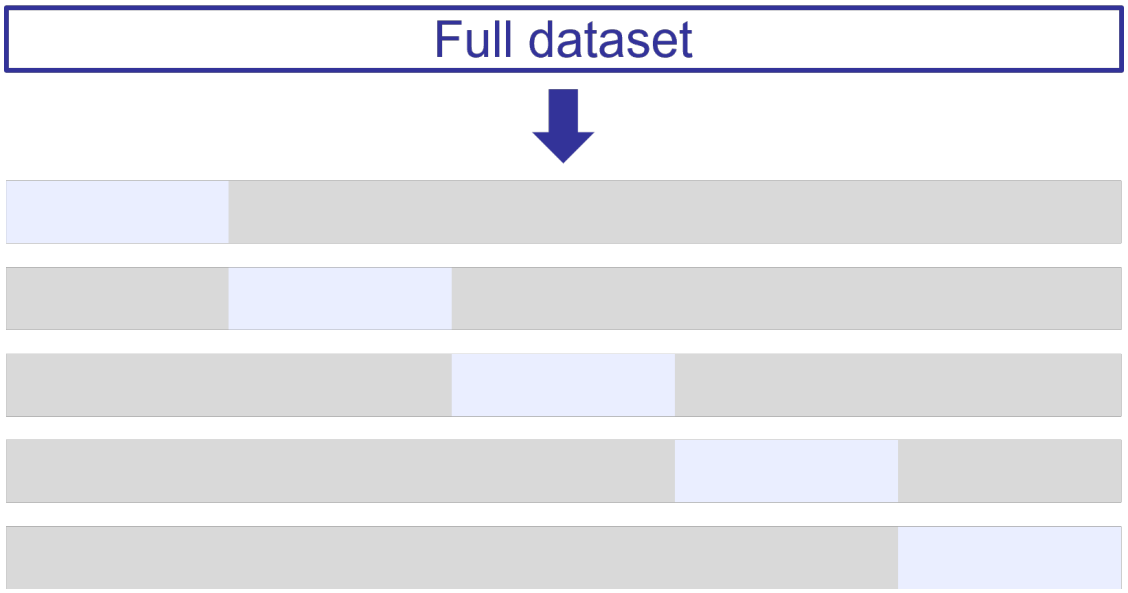


Figure B.2 | k-fold cross validation. This figure illustrates 5-fold cross-validation. The full dataset is split into 5 subsets. Each subset serves as test data (blue) in exactly one optimisation, while the other subsets serve as training data (grey).

runs. This can be prohibitive in situations where running just one optimisation is very time consuming. Akaike information criterion (AIC) [199] tries to rank models according to their predictive power. More specifically, it aims to rank models f according to their Kullback-Leibler divergence from the *true* or *generating* model g. The Kullback-Leibler divergence is motivated by **information entropy**. More specifically, the **mutual information** is the Kullback-Leibler divergence between a joint probability density function and the product of the corresponding marginal probability density functions. The Kullback-Leibler divergence is defined as

$$I\left(g,\ f\right) = E\left[\log \frac{g(\bar{\boldsymbol{y}})}{f(\bar{\boldsymbol{y}}|\boldsymbol{\theta})}\right], \tag{B.40}$$

where

- $f\colon \mathbb{R}^n \to \mathbb{R}$ and
- $g\colon \mathbb{R}^n \to \mathbb{R}$ are functionals assigning the measurement $\bar{\boldsymbol{y}} \in \mathbb{R}^n$ to a probability, and

- $E[\cdot]$ denotes the expected value under g.

That is

$$I\left(g,\ f\right) = \int g(\boldsymbol{y}) \cdot \log g\left(\overline{\boldsymbol{y}}\right)\ d\boldsymbol{y} - \int g(\overline{\boldsymbol{y}}) \cdot \log f\left(\overline{\boldsymbol{y}}|\boldsymbol{\theta}\right)\ d\boldsymbol{y}. \tag{B.41}$$

Multiplying by 2 we get

$$2I\left(g,\ f\right) = 2\int g(\boldsymbol{y}) \cdot \log g\left(\overline{\boldsymbol{y}}\right)\ d\boldsymbol{y} - 2\int g(\overline{\boldsymbol{y}}) \cdot \log f\left(\overline{\boldsymbol{y}}|\boldsymbol{\theta}\right)\ d\boldsymbol{y}. \tag{B.42}$$

As the first term does not contain f and we are only interested in ranking different models, we can ignore it and focus on minimising the second term, which we will refer to as the *Kullback discrepancy* d of our estimator $\hat{\boldsymbol{\theta}}$:

$$E\left[d(\hat{\boldsymbol{\theta}})\right] = -2\int g(\overline{\boldsymbol{y}}) \cdot \log f\left(\overline{\boldsymbol{y}}|\hat{\boldsymbol{\theta}}\right)\ d\boldsymbol{y}. \tag{B.43}$$

As we do not know the generating model g, it is necessary to assume that

$$g(\overline{\boldsymbol{y}}) = f(\overline{\boldsymbol{y}}|\boldsymbol{\theta}_{true}). \tag{B.44}$$

The expected Kullback discrepancy is therefore

$$E\left[d(\hat{\boldsymbol{\theta}})\right] = -2\int f(\overline{\boldsymbol{y}}|\boldsymbol{\theta}_{true}) \cdot \log f\left(\overline{\boldsymbol{y}}|\hat{\boldsymbol{\theta}}\right)\ d\boldsymbol{y}, \tag{B.45}$$

or

$$E\left[d(\hat{\boldsymbol{\theta}})\right] = E\left[-2\log f\left(\overline{\boldsymbol{y}}|\hat{\boldsymbol{\theta}}\right)\right], \tag{B.46}$$

or developing a Taylor series around $\boldsymbol{\theta}_{true}$

$$\begin{aligned} E\left[d(\hat{\boldsymbol{\theta}})\right] &= E\left[-2\log f\left(\overline{\boldsymbol{y}}|\boldsymbol{\theta}_{true}\right)\right] \\ &\quad + E\left[(\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_{true})^T \nabla^2 \left(-\log f(\overline{\boldsymbol{y}}|\boldsymbol{\theta}_{true})\left(\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_{true}\right)\right)\right] + \ldots, \end{aligned} \tag{B.47}$$

where we left out the first order term, because the gradient of the log-likelihood in θ_{true} is zero. As the Hessian of the negative log-likelihood is the inverse of the covariance matrix, the second term is χ^2_d distributed. The expected value of a

χ^2 distribution is its degrees of freedom, so the second term evaluates to d, the dimensionality of the model parameters. The first term is still unknown, because we do not know the ground truth parameters $\boldsymbol{\theta}_{true}$. However, we can develop a Taylor series around our estimator $\hat{\boldsymbol{\theta}}$

$$\begin{aligned} E\left[d(\boldsymbol{\theta}_{true})\right] &= E\left[-2\log f\left(\bar{\boldsymbol{y}}|\hat{\boldsymbol{\theta}}\right)\right] \\ &\quad + E\left[(\boldsymbol{\theta}_{true}-\hat{\boldsymbol{\theta}})^T\nabla^2\left(-\log f(\bar{\boldsymbol{y}}|\hat{\boldsymbol{\theta}}\right)(\boldsymbol{\theta}_{true}-\hat{\boldsymbol{\theta}})\right]+\ldots, \end{aligned} \tag{B.48}$$

where the expected value of the second order term again evaluates to d. Plugging Equation (B.48) into Equation (B.47) we arrive at

$$E\left[d(\hat{\boldsymbol{\theta}})\right] = E\left[-2\log f\left(\bar{\boldsymbol{y}}|\hat{\boldsymbol{\theta}}\right)\right] + 2d + \ldots. \tag{B.49}$$

In a setting where $d \ll n$ we can ignore higher order terms and define

$$AIC = -2\log f\left(\bar{\boldsymbol{y}}|\hat{\boldsymbol{\theta}}\right) + 2d. \tag{B.50}$$

AIC is an asymptotically unbiased and thus **efficient** estimator of the Kullback discrepancy d [200] and has been shown to be asymptotically equivalent to leave-one-out cross-validation in ordinary linear regression [201] and mixed effect models [202]. However, its use can be problematic in models with non-identifiable parameters [78].

B.2.3 Bayesian Information Criterion

While AIC tries to rank models according to their predictive power on replicate datasets, the Bayesian information criterion (BIC) aims at identifying the *"true"* or *generating* model. This requires the assumption that the generating model is a member of the set of evaluated models. The models k are ranked according to their posterior probability of being the generating model

$$P\left((k,\boldsymbol{\theta})|\bar{\boldsymbol{y}}\right) = \frac{L(\bar{\boldsymbol{y}}|\boldsymbol{\theta})\cdot P(\boldsymbol{\theta}|k)\cdot P(k)}{P(\bar{\boldsymbol{y}})}. \tag{B.54}$$

Box B.4: Information entropy

Information entropy describes how much information can be stored in a probability density function of a random variable. The broader the distribution, the higher the information entropy. More precisely, the information entropy H of a random variable X is the expected value of the negative log-probability:

$$H(X) = -\int P(X) \cdot \log_2 P(X)\, dx \tag{B.51}$$

Information entropy is measured in bits. The uniform distribution is the probability density function that contains maximum information entropy. Note that the formulation of information entropy is identical to the formulation of statistical entropy in physics, except of the Boltzmann constant.

Integrating over the parameters

$$P(k|\bar{\boldsymbol{y}}) = \int L(\bar{\boldsymbol{y}}|\boldsymbol{\theta}) \cdot P(\boldsymbol{\theta}|k)\, d\boldsymbol{\theta} \cdot \frac{P(k)}{P(\bar{\boldsymbol{y}})}. \tag{B.55}$$

Ignoring the model independent probability $P(\bar{\boldsymbol{y}})$, taking the logarithm and multiplying with minus two we define a score

$$S(k|\bar{\boldsymbol{y}}) = -2\log \int L(\bar{\boldsymbol{y}}|\boldsymbol{\theta}) \cdot P(\boldsymbol{\theta}|k)\, d\boldsymbol{\theta} - 2\log P(k). \tag{B.56}$$

To solve the integral we first develop a Taylor series to second order of the log-likelihood about the maximum likelihood estimator $\hat{\boldsymbol{\theta}}$

$$\log L(\bar{\boldsymbol{y}}|\boldsymbol{\theta}) \approx \log L(\bar{\boldsymbol{y}}|\hat{\boldsymbol{\theta}}) + \frac{1}{2}(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})^T \cdot \nabla^2(\log L(\bar{\boldsymbol{y}}|\hat{\boldsymbol{\theta}})) \cdot (\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}), \tag{B.57}$$

where the first order term was omitted, as the gradient of the log-likelihood in $\hat{\boldsymbol{\theta}}$ evaluates to zero. For brevity we define

$$\nabla^2(\log L(\bar{\boldsymbol{y}}|\hat{\boldsymbol{\theta}})) := nF(\bar{\boldsymbol{y}}, \hat{\boldsymbol{\theta}}). \tag{B.58}$$

We will use later that F is independent of the number of datapoints n. Therefore

$$L(\bar{\boldsymbol{y}}|\boldsymbol{\theta}) \approx L(\bar{\boldsymbol{y}}|\hat{\boldsymbol{\theta}}) \cdot \exp\left[\frac{1}{2}(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})^T \cdot nF(\bar{\boldsymbol{y}}, \hat{\boldsymbol{\theta}}) \cdot (\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})\right] \tag{B.59}$$

Plugging Equation (B.59) into Equation (B.56) we obtain

$$S(k|\bar{\boldsymbol{y}}) \approx -2\log \int L(\bar{\boldsymbol{y}}|\hat{\boldsymbol{\theta}}) \cdot \exp\left[\frac{1}{2}(\boldsymbol{\theta}-\hat{\boldsymbol{\theta}})^T \cdot nF(\bar{\boldsymbol{y}},\hat{\boldsymbol{\theta}}) \cdot (\boldsymbol{\theta}-\hat{\boldsymbol{\theta}})\right] \cdot P(\boldsymbol{\theta}|k)\, d\boldsymbol{\theta} \\ -2\log P(k). \tag{B.60}$$

The Taylor approximation only holds in a small neighbourhood to $\hat{\boldsymbol{theta}}$. In this neighbourhood the prior $P(\boldsymbol{\theta})$ remains quasi-constant compared to the likelihood, which may be very sensitive to parameter changes. We therefore ignore the prior by setting it to 1 and observe that the integral is over a Gaussian. Using that

$$\int e^{-\frac{1}{2}\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}}\, d\boldsymbol{x} = \sqrt{\frac{(2\pi)^d}{\det \boldsymbol{A}}}, \tag{B.61}$$

and that

$$\det n\boldsymbol{A} = n^d \det \boldsymbol{A}, \tag{B.62}$$

where $\boldsymbol{x} \in \mathbb{R}^d$ and $\boldsymbol{A} \in \mathbb{R}^{d\times d}$

$$\begin{aligned} S(k|\bar{\boldsymbol{y}}) &\approx -2\log\left(L(\bar{\boldsymbol{y}}|\hat{\boldsymbol{\theta}})\left(\frac{2\pi}{n}\right)^{d/2}\det F(\bar{\boldsymbol{y}},\hat{\boldsymbol{\theta}})^{-1/2}\right) - 2\log P(k) \\ &= -2\log L(\bar{\boldsymbol{y}}|\hat{\boldsymbol{\theta}}) - d\log 2\pi + d\log n + \log\det F(\bar{\boldsymbol{y}},\hat{\boldsymbol{\theta}}) - 2\log P(k). \end{aligned} \tag{B.63}$$

The BIC considers only the asymptotic case of large sample size n. At large n the first and third term become dominant and we arrive at

$$BIC = -2\log L(\bar{\boldsymbol{y}}|\hat{\boldsymbol{\theta}}) + d\log n. \tag{B.64}$$

The BIC can be used to determine the posterior probability of model k as

$$P(k|\bar{\boldsymbol{y}}) = \frac{\exp -\frac{1}{2}\Delta_k}{\sum_{l=1}^{L}\exp -\frac{1}{2}\Delta_l}, \tag{B.65}$$

where $\Delta_k = BIC_k - BIC_{best}$ and L is the total number of candidate models. BIC is an asymptotically consistent estimator of the posterior probability [205]. It has

been shown to be asymptotically equivalent to leave-many-out cross-validation [206]. However, like the AIC its use can be problematic in models with non-identifiable parameters [78]. In comparison to the AIC, the BIC has a harsher penalty, which also depends on the sample size. Therefore, the BIC prefers simpler models than the AIC, especially if the sample size is large.

Box B.5: Mutual information

Consider two random variables X and Y. Mutual information I describes, how much one random variable tells us about the other. In other words, it quantifies how different the joint probability distribution $P(X, Y)$ is from the product of the marginal probability distributions $P(X) \cdot P(Y)$. More specifically, it gives us the expected value of the logarithmised ratio between the joint probability density function and the product of marginal probability density functions. Note that this is more general than a correlation coefficients, which only describe linear dependence. There are many equivalent definitions of mutual information.

$$\begin{aligned} I(X,Y) &= \iint P(X,Y) \cdot \log_2 \frac{P(X,Y)}{P(X) \cdot P(Y)} \, dx \, dy \\ &= \iint P(X|Y) \cdot P(Y) \cdot \log_2 \frac{P(X|Y) \cdot \cancel{P(Y)}}{P(X) \cdot \cancel{P(Y)}} \, dx \, dy \\ &= \iint P(X|Y) \cdot P(Y) \cdot \log_2 P(X|Y) \, dx \, dy - \int P(X) \cdot log_2 P(X) \, dx \\ &= H(X) - H(X|Y) \end{aligned} \tag{B.52}$$

From Equation (B.52) we can see that mutual information is symmetric with respect to X and Y. It is also equivalent to the leftover information entropy in X once we have observed Y (Figure B.3).

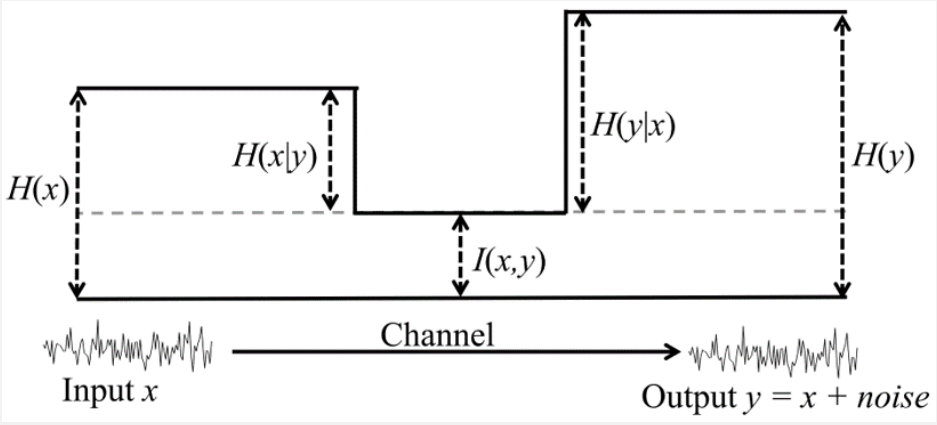


Figure B.3 | Measuring an output reduces the information entropy of the input to the mutual information. Figure distributed by [203] under a CC BY-SA 4.0 license.

As illustrated in Figure B.4, the relation of mutual information with

information entropy can be expressed as

$$\begin{aligned} I(X,Y) &= H(X) - H(X|Y) \\ &= H(Y) - H(Y|X) \\ &= H(X) + H(Y) - H(X,Y) \\ &= H(X,Y) - H(X|Y) - H(Y|X). \end{aligned} \tag{B.53}$$

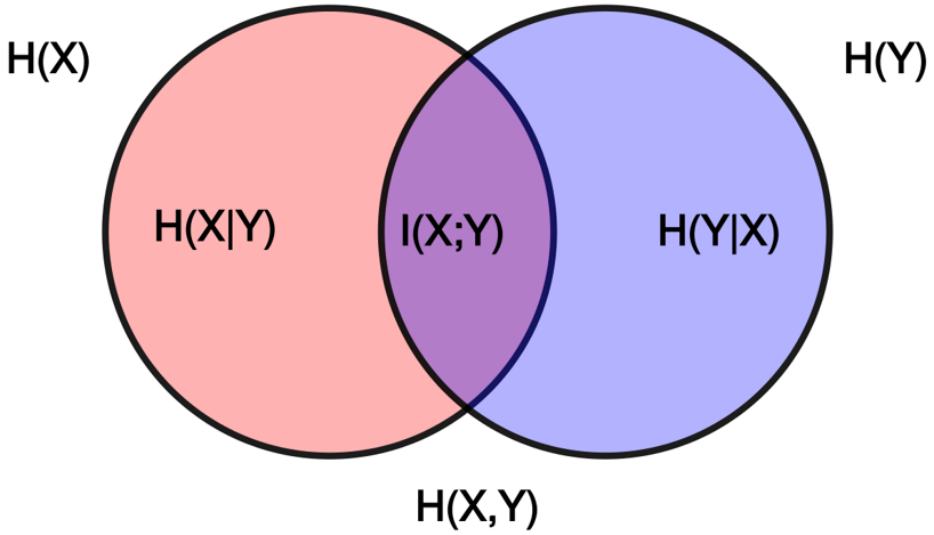


Figure B.4 | The relation of mutual information and information entropy. $H(X)$: left circle. $H(Y)$: right circle. $H(X,Y)$: union of left and right circle. $H(X|Y)$: red area. $H(Y|X)$: blue area. $I(X;Y)$: violet area. Figure released by [204] to the public domain.

References

[1] Richard P Feynman. "The Value of Science". *Engineering and Science* (1955).

[2] Aristotle. *Posterior analytics*. Loeb classical library 391. Harvard University Press.

[3] A Abroudi, S Samarasinghe, and D Kulasiri. "A Review of Computational Models of Mammalian Cell Cycle". 21st International Congress on Modelling and Simulation, Gold Coast, Australia, 2015.

[4] Ali Abroudi. "Holistic complex systems modelling approaches for cell signalling networks – Mammalian cell cycle control system". Thesis. Lincoln University, 2019.

[5] R. Singhania et al. "A hybrid model of mammalian cell cycle regulation". *PLoS Comput Biol* 7.2 (2011), e1001077.

[6] Fangting Li et al. "The yeast cell-cycle network is robustly designed". *Proceedings of the National Academy of Sciences* 101.14 (2004), pp. 4781–4786.

[7] John H Gauthier and Phillip I Pohl. "A general framework for modeling growth and division of mammalian cells". *BMC Systems Biology* 5.1 (2011), p. 3.

[8] Michael C. Weis et al. "A Data-Driven, Mathematical Model of Mammalian Cell Cycle Regulation". *PLOS ONE* 9.5 (2014), e97130.

[9] Riaan Conradie et al. "Restriction point control of the mammalian cell cycle via the cyclin E/Cdk2:p27 complex". *The FEBS Journal* 277.2 (2010), pp. 357–367.

[10] Attila Csikász-Nagy et al. "Analysis of a Generic Model of Eukaryotic Cell-Cycle Regulation". *Biophysical Journal* 90.12 (2006), pp. 4361–4379.

[11] Ali Abroudi, Sandhya Samarasinghe, and Don Kulasiri. "A comprehensive complex systems approach to the study and analysis of mammalian cell cycle control system in the presence of DNA damage stress". *Journal of Theoretical Biology* 429 (2017), pp. 204–228.

[12] Kazunari Iwamoto et al. “Mathematical modeling of cell cycle regulation in response to DNA damage: Exploring mechanisms of cell-fate determination”. *Biosystems* 103.3 (2011), pp. 384–391.

[13] Ali Abroudi, Sandhya Samarasinghe, and Don Kulasiri. “Towards abstraction of computational modelling of mammalian cell cycle: Model reduction pipeline incorporating multi-level hybrid petri nets”. *Journal of Theoretical Biology* 496 (2020), p. 110212.

[14] Mutaz Khazaaleh, Sandhya Samarasinghe, and Don Kulasiri. “A new hierarchical approach to multi-level model abstraction for simplifying ODE models of biological networks and a case study: The G1/S Checkpoint/DNA damage signalling pathways of mammalian cell cycle”. *Biosystems* 203 (2021), p. 104374.

[15] Chen Li et al. “BioModels Database: An enhanced, curated and annotated resource for published quantitative kinetic models”. *BMC Systems Biology* 4.1 (2010), p. 92.

[16] Katherine C. Chen et al. “Kinetic Analysis of a Molecular Model of the Budding Yeast Cell Cycle”. *Molecular Biology of the Cell* 11.1 (2000), pp. 369–391.

[17] Katherine C. Chen et al. “Integrative Analysis of Cell Cycle Control in Budding Yeast”. *Molecular Biology of the Cell* 15.8 (2004), pp. 3841–3862.

[18] Cihan Oguz et al. “Optimization and model reduction in the high dimensional parameter space of a budding yeast cell cycle model”. *BMC Systems Biology* 7.1 (2013), p. 53.

[19] Pavel Kraikivski et al. “From START to FINISH: computational analysis of cell cycle control in budding yeast”. *NPJ systems biology and applications* 1 (2015), p. 15016.

[20] Eshan D. Mitra et al. “Using both qualitative and quantitative data in parameter identification for systems biology models”. *Nature Communications* 9.1 (2018), p. 3901.

[21] Eshan D. Mitra et al. “PyBioNetFit and the Biological Property Specification Language”. *iScience* 19 (2019), pp. 1012–1036.

[22] Paul F. Lang et al. “BpForms and BcForms: a toolkit for concretely describing non-canonical polymers and complexes to facilitate global biochemical networks”. *Genome Biology* 21.1 (2020), p. 117.

[23] Paul F. Lang, Sungho Shin, and Victor M. Zavala. "SBML2Julia: interfacing SBML with efficient nonlinear Julia modelling and solution tools for parameter optimization". *arXiv:2011.02597 [q-bio]* (2020).

[24] Leonard Schmiester et al. "PEtab – interoperable specification of parameter estimation problems in systems biology". *arXiv:2004.01154 [q-bio]* (2020).

[25] Iain Dunning, Joey Huchette, and Miles Lubin. "JuMP: A Modeling Language for Mathematical Optimization". *SIAM Review* 59.2 (2017), pp. 295–320.

[26] René Descartes. *A discourse on method*. Everyman's library ; no. 570. J.M. Dent, 1637.

[27] Aurélien Alfonsi et al. "Adaptive simulation of hybrid stochastic and deterministic models for biochemical systems". *ESAIM: Proceedings* 14 (2005), pp. 1–13.

[28] Radhakrishnan Mahadevan, Jeremy S. Edwards, and Francis J. Doyle. "Dynamic Flux Balance Analysis of Diauxic Growth in Escherichia coli". *Biophysical Journal* 83.3 (2002), pp. 1331–1340.

[29] David S. Tourigny, Arthur P. Goldberg, and Jonathan R. Karr. "Simulating single-cell metabolism using a stochastic flux-balance analysis algorithm". *Biophysical Journal* 120.23 (2021), pp. 5231–5242.

[30] Markus W. Covert, Christophe H. Schilling, and Bernhard Palsson. "Regulation of Gene Expression in Flux Balance Models of Metabolism". *Journal of Theoretical Biology* 213.1 (2001), pp. 73–88.

[31] Markus W. Covert et al. "Integrating metabolic, transcriptional regulatory and signal transduction models in Escherichia coli". *Bioinformatics* 24.18 (2008), pp. 2044–2050.

[32] S. Hoops et al. "COPASI–a COmplex PAthway SImulator". *Bioinformatics* 22.24 (2006), pp. 3067–74.

[33] Kiri Choi et al. "Tellurium: An extensible python-based modeling environment for systems and synthetic biology". *Biosystems* 171 (2018), pp. 74–79.

[34] Henning Schmidt and Mats Jirstrand. "Systems Biology Toolbox for MATLAB: a computational platform for research in systems biology". *Bioinformatics* 22.4 (2006), pp. 514–515.

[35] Christopher Rackauckas and Qing Nie. "DifferentialEquations.jl – A Performant and Feature-Rich Ecosystem for Solving Differential Equations in Julia". *Journal of Open Research Software* 5.1 (2017), p. 15.

[36] Yingbo Ma et al. "ModelingToolkit: A Composable Graph Transformation System For Equation-Based Modeling". *arXiv:2103.05244 [cs]* (2021).

[37] Ali Ebrahim et al. "COBRApy: COnstraints-Based Reconstruction and Analysis for Python". *BMC Systems Biology* 7.1 (2013), p. 74.

[38] Miroslav Kratochvíl et al. "COBREXA.jl: constraint-based reconstruction and exascale analysis". *Bioinformatics* 38.4 (2022), pp. 1171–1172.

[39] Justin S. Hogg. *Advances in Rule-based Modeling: Compartments, Energy, and Hybrid Simulation, with Application to Sepsis and Cell Signaling*. University of Pittsburgh ETD. 2013.

[40] Michael L. Blinov et al. "BioNetGen: software for rule-based modeling of signal transduction based on the interactions of molecular domains". *Bioinformatics* 20.17 (2004), pp. 3289–3291.

[41] John A. P. Sekar and James R. Faeder. "Rule-Based Modeling of Signal Transduction: A Primer". *Computational Modeling of Signaling Networks*. Methods in Molecular Biology. Humana Press, 2012, pp. 139–218.

[42] Michael W. Sneddon, James R. Faeder, and Thierry Emonet. "Efficient modeling, simulation and coarse-graining of biological complexity with NFsim". *Nature Methods* 8.2 (2011), pp. 177–183.

[43] Adam M. Smith et al. "RuleBender: integrated modeling, simulation and visualization for rule-based intracellular biochemistry". *BMC Bioinformatics* 13.8 (2012), S3.

[44] Carlos F. Lopez et al. "Programming biological models in Python using PySB". *Molecular Systems Biology* 9.1 (2013).

[45] Autumn A. Cuellar et al. "An Overview of CellML 1.1, a Biological Model Description Language". *SIMULATION* 79.12 (2003), pp. 740–747.

[46] M. Hucka et al. "The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models". *Bioinformatics (Oxford, England)* 19.4 (2003), pp. 524–531.

[47] Claudine Chaouiya et al. "SBML qualitative models: a model representation format and infrastructure to foster interactions between qualitative modelling formalisms and tools". *BMC Systems Biology* 7.1 (2013), p. 135.

[48] Brett G. Olivier and Frank T. Bergmann. "SBML Level 3 Package: Flux Balance Constraints version 2". *Journal of Integrative Bioinformatics* 15.1 (2018).

[49] Fengkai Zhang et al. “Systems biology markup language (SBML) level 3 package: multistate, multicomponent and multicompartment species, version 1, release 2”. *Journal of Integrative Bioinformatics* 17.2-3 (2020).

[50] Kurt Gödel. “Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I”. *Monatshefte für Mathematik und Physik* 38.1 (1931), pp. 173–198.

[51] Toby S. Cubitt, David Perez-Garcia, and Michael M. Wolf. “Undecidability of the spectral gap”. *Nature* 528.7581 (2015), pp. 207–211.

[52] E. Balsa-Canto, A. A. Alonso, and J. R. Banga. “Computational procedures for optimal experimental design in biological systems”. *IET Systems Biology* 2.4 (2008), pp. 163–172.

[53] Alejandro F Villaverde et al. “A protocol for dynamic model calibration”. *Briefings in Bioinformatics* bbab387 (2021).

[54] Gabriele Gut, Markus D. Herrmann, and Lucas Pelkmans. “Multiplexed protein maps link subcellular organization to cellular states”. *Science* 361.6401 (2018), eaar7042.

[55] Zehua Liu et al. “Reconstructing cell cycle pseudo time-series via single-cell transcriptome data”. *Nature Communications* 8.1 (2017), p. 22.

[56] Osamu Shimomura, Frank H. Johnson, and Yo Saiga. “Extraction, Purification and Properties of Aequorin, a Bioluminescent Protein from the Luminous Hydromedusan, Aequorea”. *Journal of Cellular and Comparative Physiology* 59.3 (1962), pp. 223–239.

[57] Talley J. Lambert. “FPbase: a community-editable fluorescent protein database”. *Nature Methods* 16.4 (2019), pp. 277–278.

[58] Jinming Gu et al. “Cell Cycle-dependent Regulation of a Human DNA Helicase That Localizes in DNA Damage Foci”. *Molecular Biology of the Cell* 15.7 (2004), pp. 3320–3332.

[59] Alexis R. Barr et al. “A Dynamical Framework for the All-or-None G1/S Transition”. *Cell Systems* 2.1 (2016), pp. 27–37.

[60] Bogdan Budnik et al. “SCoPE-MS: mass spectrometry of single mammalian cells quantifies proteome heterogeneity during cell differentiation”. *Genome Biology* 19.1 (2018), p. 161.

[61] Tony Ly et al. “A proteomic chronology of gene expression through the cell cycle in human myeloid leukemia cells”. *eLife* 3 (2014), e01630.

[62] Tony Ly et al. “Proteomic analysis of cell cycle progression in asynchronous cultures, including mitotic subphases, using PRIMMUS”. *eLife* 6 (2017).

[63] Van Kelly et al. “Low Cell Number Proteomic Analysis Using In-Cell Protease Digests Reveals a Robust Signature for Cell Cycle State Classification”. *Molecular & Cellular Proteomics* 21.1 (2022), p. 100169.

[64] Tony Ly, Aki Endo, and Angus I Lamond. “Proteomic analysis of the response to cell cycle arrests in human myeloid leukemia cells”. *eLife* 4 (2015), e04534.

[65] Karl Pearson. “LIII. On lines and planes of closest fit to systems of points in space”. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 2.11 (1901), pp. 559–572.

[66] Harold Hotelling. “Relations Between Two Sets of Variates”. *Biometrika* 28.3/4 (1936), pp. 321–377.

[67] Geoffrey E Hinton and Sam Roweis. “Stochastic Neighbor Embedding”. *Advances in Neural Information Processing Systems*. Vol. 15. MIT Press, 2003.

[68] Laurens van der Maaten and Geoffrey Hinton. “Visualizing Data using t-SNE”. *Journal of Machine Learning Research* 9.86 (2008), pp. 2579–2605.

[69] Ran Kafri et al. “Dynamics extracted from fixed cells reveal feedback linking cell growth to cell cycle”. *Nature* 494.7438 (2013), pp. 480–483.

[70] Sean C. Bendall et al. “Single-Cell Trajectory Detection Uncovers Progression and Regulatory Coordination in Human B Cell Development”. *Cell* 157.3 (2014), pp. 714–725.

[71] Laleh Haghverdi et al. “Diffusion pseudotime robustly reconstructs lineage branching”. *Nature Methods* 13.10 (2016), pp. 845–848.

[72] Maria Anna Rapsomaniki et al. “CellCycleTRACER accounts for cell cycle and volume in mass cytometry data”. *Nature Communications* 9.1 (2018), p. 632.

[73] Luca Rappez et al. “DeepCycle reconstructs a cyclic cell cycle trajectory from unsegmented cell images using convolutional neural networks”. *Molecular Systems Biology* 16.10 (2020), e9474.

[74] Gabriele Gut et al. “Trajectories of cell-cycle progression from fixed cell populations”. *Nature Methods* 12.10 (2015), pp. 951–954.

[75] Martin Jinek et al. “A Programmable Dual-RNA–Guided DNA Endonuclease in Adaptive Bacterial Immunity”. *Science* 337.6096 (2012), pp. 816–821.

[76] Lei S. Qi et al. “Repurposing CRISPR as an RNA-Guided Platform for Sequence-Specific Control of Gene Expression”. *Cell* 152.5 (2013), pp. 1173–1183.

[77] Luke A. Gilbert et al. “Genome-Scale CRISPR-Mediated Control of Gene Repression and Activation”. *Cell* 159.3 (2014), pp. 647–661.

[78] Fabian Fröhlich, Carolin Loos, and Jan Hasenauer. “Scalable Inference of Ordinary Differential Equation Models of Biochemical Processes”. *Gene Regulatory Networks: Methods and Protocols*. Methods in Molecular Biology. Springer, 2019, pp. 385–422.

[79] Yosef D Roth et al. “Datanator: an integrated database of molecular data for quantitatively modeling cellular behavior”. *Nucleic Acids Research* 49.D1 (2021), pp. D516–D522.

[80] Philipp Städter et al. “Benchmarking of numerical integration methods for ODE models of biological systems”. *Scientific Reports* 11.1 (2021), p. 2696.

[81] Andrea Degasperi, Dirk Fey, and Boris N. Kholodenko. “Performance of objective functions and optimisation procedures for parameter estimation in system biology models”. *npj Systems Biology and Applications* 3.1 (2017), pp. 1–9.

[82] Leonard Schmiester et al. “Efficient parameterization of large-scale dynamic models based on relative measurements”. *Bioinformatics* 36.2 (2020), pp. 594–602.

[83] Jake Alan Pitt and Julio R. Banga. “Parameter estimation in models of biological oscillators: an automated regularised estimation approach”. *BMC Bioinformatics* 20.1 (2019), p. 82.

[84] Fabian Fröhlich et al. “Scalable Parameter Estimation for Genome-Scale Biochemical Reaction Networks”. *PLOS Computational Biology* 13.1 (2017), e1005331.

[85] Jose A. Egea et al. “Dynamic Optimization of Nonlinear Processes with an Enhanced Scatter Search Method”. *Industrial & Engineering Chemistry Research* 48.9 (2009), pp. 4388–4401.

[86] Alejandro F. Villaverde et al. "Benchmarking optimization methods for parameter estimation in large kinetic models". *Bioinformatics* 35.5 (2019), pp. 830–838.

[87] Alejandro F. Villaverde, Jose A. Egea, and Julio R. Banga. "A cooperative strategy for parameter estimation in large scale systems biology models". *BMC Systems Biology* 6.1 (2012), p. 75.

[88] David R. Penas et al. "Parameter estimation in large-scale systems biology models: a parallel and self-adaptive cooperative strategy". *BMC Bioinformatics* 18.1 (2017), p. 52.

[89] Patrick Weber et al. "Parameter Estimation and Identifiability of Biological Networks Using Relative Data". *IFAC Proceedings Volumes*. 18th IFAC World Congress 44.1 (2011), pp. 11648–11653.

[90] Carolin Loos, Sabrina Krause, and Jan Hasenauer. "Hierarchical optimization for the efficient parametrization of ODE models". *Bioinformatics* 34.24 (2018), pp. 4266–4273.

[91] Gareth James et al. *Introduction to Statistical Learning*. Springer Science+Business Media New York, 2013.

[92] Paul Feyerabend. *Against method*. 3rd ed. Verso, 1993.

[93] John J Tyson and Béla Novák. "Irreversible Transitions, Bistability and Checkpoint Controls in the Eukaryotic Cell Cycle: A Systems-level Understanding" (2013), p. 40.

[94] L. H. Hartwell and T. A. Weinert. "Checkpoints: controls that ensure the order of cell cycle events". *Science* 246.4930 (1989), pp. 629–34.

[95] Y. Zwang et al. "Two phases of mitogenic signaling unveil roles for p53 and EGR1 in elimination of inconsistent growth signals". *Mol Cell* 42.4 (2011), pp. 524–35.

[96] H. M. Temin. "Stimulation by serum of multiplication of stationary chicken cells". *J Cell Physiol* 78.2 (1971), pp. 161–70.

[97] A. B. Pardee. "A restriction point for control of normal animal cell proliferation". *Proc Natl Acad Sci USA* 71.4 (1974), pp. 1286–90.

[98] Frank S. Heldt et al. "Dilution and titration of cell-cycle regulators may control cell size in budding yeast". *PLOS Computational Biology* 14.10 (2018), e1006548.

[99] Clotilde Cadart et al. “Size control in mammalian cells involves modulation of both growth rate and cell cycle duration” (2017).

[100] G. Varsano, Y. Wang, and M. Wu. “Probing Mammalian Cell Size Homeostasis by Channel-Assisted Cell Reshaping”. *Cell Rep* 20.2 (2017), pp. 397–410.

[101] L. N. Truong and X. Wu. “Prevention of DNA re-replication in eukaryotic cells”. *J Mol Cell Biol*. Vol. 3. 1. 2011, pp. 13–22.

[102] M. J. O’Connell, N. C. Walworth, and A. M. Carr. “The G2-phase DNA-damage checkpoint”. *Trends Cell Biol* 10.7 (2000), pp. 296–303.

[103] Bruce Alberts et al. “Chapter 17: The Cell Cycle”. *Molecular Biology of the Cell*. Taylor & Francis Group, 2014.

[104] David O Morgan. *The Cell Cycle: Principles of Control*. New Science Press: London., 2006.

[105] Karen L. Craig and Mike Tyers. “The F-box: a new motif for ubiquitin dependent proteolysis in cell cycle regulation and signal transduction”. *Progress in Biophysics and Molecular Biology* 72.3 (1999), pp. 299–328.

[106] S. Gookin et al. “A map of protein dynamics during cell-cycle progression and cell-cycle exit”. *PLoS Biol* 15.9 (2017), e2003268.

[107] D. O. Morgan. “Cyclin-dependent kinases: engines, clocks, and microprocessors”. *Annu Rev Cell Dev Biol* 13 (1997), pp. 261–91.

[108] J. Mitra and G. H. Enders. “Cyclin A/Cdk2 complexes regulate activation of Cdk1 and Cdc25 phosphatases in human cells”. *Oncogene* 23.19 (2004), pp. 3361–7.

[109] B. Stern and P. Nurse. “A quantitative model for the cdc2 control of S phase and mitosis in fission yeast”. *Trends Genet* 12.9 (1996), pp. 345–50.

[110] D. Coudreuse and P. Nurse. “Driving the cell cycle with a minimal CDK control network”. *Nature* 468.7327 (2010), pp. 1074–9.

[111] W. Sha et al. “Hysteresis drives cell-cycle transitions in Xenopus laevis egg extracts”. *Proc Natl Acad Sci USA* 100.3 (2003), pp. 975–80.

[112] Byron C Williams et al. “Greatwall-phosphorylated Endosulfine is both an inhibitor and a substrate of PP2A-B55 heterotrimers”. *eLife* 3 (2014), e01695.

[113] M. Koivomagi et al. “Cascades of multisite phosphorylation control Sic1 destruction at the onset of S phase”. *Nature* 480.7375 (2011), pp. 128–31.

[114] G. Yao et al. "A bistable Rb-E2F switch underlies the restriction point". *Nat Cell Biol* 10.4 (2008), pp. 476–82.

[115] Joseph R. Pomerening, Eduardo D. Sontag, and James E. Ferrell Jr. "Building a cell cycle oscillator: hysteresis and bistability in the activation of Cdc2". *Nature Cell Biology* 5.4 (2003), pp. 346–351.

[116] Alexis R. Barr et al. "DNA damage during S-phase mediates the proliferation-quiescence decision in the subsequent G1 via p21 expression". *Nature Communications* 8 (2017), p. 14728.

[117] James E. Ferrell and Sang Hoon Ha. "Ultrasensitivity part III: cascades, bistable switches, and oscillators". *Trends in Biochemical Sciences* 39.12 (2014), pp. 612–618.

[118] A Goldbeter and D E Koshland. "An amplified sensitivity arising from covalent modification in biological systems" (1981).

[119] A V Hill. "The possible effects of the aggregation of the molecules of haemoglobin on its dissociation curves". *J Physiol (Lond)* 40 (1910), pp. 4–7.

[120] T. Arooz et al. "On the concentrations of cyclins and cyclin-dependent kinases in extracts of cultured human cells". *Biochemistry* 39.31 (2000), pp. 9494–501.

[121] Frank S. Heldt et al. "A comprehensive model for the proliferation-quiescence decision in response to endogenous DNA damage in human cells". *Proceedings of the National Academy of Sciences of the United States of America* 115.10 (2018), pp. 2532–2537.

[122] J. William Harbour and Douglas C. Dean. "The Rb/E2F pathway: expanding roles and emerging paradigms" (2000).

[123] J. Y. Hsu et al. "E2F-dependent accumulation of hEmi1 regulates S phase entry by inhibiting APC(Cdh1)". *Nat Cell Biol* 4.5 (2002), pp. 358–66.

[124] J. J. Miller et al. "Emi1 stably binds and inhibits the anaphase-promoting complex/cyclosome as a pseudosubstrate inhibitor". *Genes Dev* 20.17 (2006), pp. 2410–20.

[125] J. Y. Huang and J. W. Raff. "The dynamic localisation of the Drosophila APC/C: evidence for the existence of multiple complexes that perform distinct functions and are differentially localised". *J Cell Sci* 115.Pt 14 (2002), pp. 2847–56.

[126] Béla Novák and John J. Tyson. "Mechanisms of signalling-memory governing progression through the eukaryotic cell cycle". *Current Opinion in Cell Biology*. Cell Signalling 69 (2021), pp. 7–16.

[127] Scott Rata et al. "Two Interlinked Bistable Switches Govern Mitotic Control in Mammalian Cells". *Current Biology* 28.23 (2018), 3824–3832.e6.

[128] Nadia Hégarat et al. "Cyclin A triggers Mitosis either via the Greatwall kinase pathway or Cyclin B". *The EMBO Journal* 39.11 (2020), e104419.

[129] P. K. Vinod and Bela Novak. "Model scenarios for switch-like mitotic transitions". *FEBS Letters* 589.6 (2015), pp. 667–671.

[130] S. Mochida and T. Hunt. "Protein phosphatases and their regulation in the control of mitosis". *EMBO Rep* 13.3 (2012), pp. 197–203.

[131] J. E. Ferrell. "Feedback loops and reciprocal regulation: recurring motifs in the systems biology of the cell cycle". *Curr Opin Cell Biol* 25.6 (2013).

[132] J. Millour et al. "ATM and p53 regulate FOXM1 expression via E2F in breast cancer epirubicin treatment and resistance". *Mol Cancer Ther* 10.6 (2011), pp. 1046–58.

[133] H. Murakami et al. "Regulation of yeast forkhead transcription factors and FoxM1 by cyclin-dependent and polo-like kinases". *Cell Cycle* 9.16 (2010), pp. 3233–42.

[134] A. Marti et al. "Interaction between ubiquitin–protein ligase SCFand E2F-1 underlies the regulation of E2F-1 degradation". *Nature Cell Biology* 1.1 (1999), p. 14.

[135] Daniele Guardavaccaro et al. "Control of Meiotic and Mitotic Progression by the F Box Protein β-Trcp1 In Vivo". *Developmental Cell* 4.6 (2003), pp. 799–812.

[136] J. Laoukili et al. "FoxM1 is degraded at mitotic exit in a Cdh1-dependent manner". *Cell Cycle* 7.17 (2008), pp. 2720–6.

[137] C. M. Pfleger and M. W. Kirschner. "The KEN box: an APC recognition signal distinct from the D box targeted by Cdh1". *Genes Dev* 14.6 (2000), pp. 655–65.

[138] Y. Takahashi, J. B. Rayman, and B. D. Dynlacht. "Analysis of promoter binding by the E2F and pRB families in vivo: distinct E2F proteins mediate activation and repression". *Genes Dev* 14.7 (2000), pp. 804–16.

[139] T. Listovsky and J. E. Sale. "Sequestration of CDH1 by MAD2L2 prevents premature APC/C activation prior to anaphase onset". *J Cell Biol.* Vol. 203. 1. 2013, pp. 87–100.

[140] E. M. Abdelalim. "Molecular mechanisms controlling the cell cycle in embryonic stem cells". *Stem Cell Rev* 9.6 (2013), pp. 764–73.

[141] Yen-Chun Liu et al. "Global Regulation of Nucleotide Biosynthetic Genes by c-Myc". *PLoS ONE* 3.7 (2008), e2722.

[142] Ignacio Pérez-Roger et al. "Myc activation of cyclin E/Cdk2 kinase involves induction of cyclin E gene transcription and inhibition of p27Kip1 binding to newly formed complexes". *Oncogene* 14.20 (1997), pp. 2373–2381.

[143] Peng Dong et al. "Division of labour between Myc and G1 cyclins in cell cycle commitment and pace control". *Nature Communications* 5.1 (2014), p. 4750.

[144] Marcos Malumbres et al. "Mammalian Cells Cycle without the D-Type Cyclin-Dependent Kinases Cdk4 and Cdk6". *Cell* 118.4 (2004), pp. 493–504.

[145] S. Ortega et al. "Cyclin-dependent kinase 2 is essential for meiosis but not for mitotic cell division in mice". *Nat Genet* 35.1 (2003), pp. 25–31.

[146] Y. Geng et al. "Cyclin E ablation in the mouse". *Cell* 114.4 (2003), pp. 431–43.

[147] I. Kalaszczynska et al. "Cyclin A - Redundant in Fibroblasts, Essential in Hematopoietic and Embryonal Stem Cells". *Cell* 138.2 (2009), pp. 352–65.

[148] Bard Ermentrout. *Simulating, Analyzing, and Animating Dynamical Systems.* Software, Environments and Tools. Society for Industrial and Applied Mathematics, 2002.

[149] Leonard A. Harris et al. "BioNetGen 2.2: advances in rule-based modeling". *Bioinformatics* 32.21 (2016), pp. 3366–3368.

[150] K. Wesley Overton et al. "Basal p21 controls population heterogeneity in cycling and quiescent cell cycle states". *Proceedings of the National Academy of Sciences* 111.41 (2014), E4386–E4393.

[151] Eric Batchelor et al. "Stimulus-dependent dynamics of p53 in single cells". *Molecular Systems Biology* 7 (2011), p. 488.

[152] Michael Tsabar et al. "A Switch in p53 Dynamics Marks Cells That Escape from DSB-Induced Cell Cycle Arrest". *Cell Reports* 32.5 (2020), p. 107995.

[153] Wayne Stallaert et al. "The structure of the human cell cycle". *bioRxiv* (2021), p. 2021.02.11.430845.

[154] Fangfang Yang et al. "The Akt/FoxO/p27Kip1 axis contributes to the anti-proliferation of pentoxifylline in hypertrophic scars". *Journal of Cellular and Molecular Medicine* 23.9 (2019), pp. 6164–6172.

[155] Alessia Montagnoli et al. "Ubiquitination of p27 is regulated by Cdk-dependent phosphorylation and trimeric complex formation". *Genes & Development* 13.9 (1999), pp. 1181–1189.

[156] Anja Hagting et al. "Translocation of cyclin B1 to the nucleus at prophase requires a phosphorylation-dependent nuclear import signal". *Current Biology* 9.13 (1999), pp. 680–689.

[157] Andrew B. Gladden and J. Alan Diehl. "Location, location, location: The role of cyclin D1 nuclear localization in cancer". *Journal of Cellular Biochemistry* 96.5 (2005), pp. 906–913.

[158] Iordanka A. Ivanova, Alisa Vespa, and Lina Dagnino. "A Novel Mechanism of E2F1 Regulation Via Nucleocytoplasmic Shuttling: Determinants of Nuclear Import and Export". *Cell Cycle* 6.17 (2007), pp. 2186–2195.

[159] Helena Silva Cascales et al. "Cyclin A2 localises in the cytoplasm at the S/G2 transition to activate PLK1". *Life Science Alliance* 4.3 (2021).

[160] Silvia D. M. Santos et al. "Spatial Positive Feedback at the Onset of Mitosis". *Cell* 149.7 (2012), pp. 1500–1513.

[161] Tommaso Cavazza and Isabelle Vernos. "The RanGTP Pathway: From Nucleo-Cytoplasmic Transport to Spindle Assembly and Beyond". *Frontiers in Cell and Developmental Biology* 3 (2016).

[162] Karl Popper. *Logik der Forschung*. Vol. 5. Springer, 1935.

[163] Leland H. Hartwell. "Macromolecule Synthesis in Temperature-sensitive Mutants of Yeast". *Journal of Bacteriology* (1967).

[164] Paul Nurse, Pierre Thuriaux, and Kim Nasmyth. "Genetic control of the cell division cycle in the fission yeast Schizosaccharomyces pombe". *Molecular and General Genetics MGG* 146.2 (1976), pp. 167–178.

[165] T. Evans et al. "Cyclin: a protein specified by maternal mRNA in sea urchin eggs that is destroyed at each cleavage division". *Cell* 33.2 (1983), pp. 389–396.

[166] Diana Mahdessian et al. “Spatiotemporal dissection of the cell cycle regulated human proteome”. *bioRxiv* (2019), p. 543231.

[167] Asako Sakaue-Sawano et al. “Visualizing Spatiotemporal Dynamics of Multicellular Cell-Cycle Progression”. *Cell* 132.3 (2008), pp. 487–498.

[168] N. Zielke and B. A. Edgar. “FUCCI sensors: powerful new tools for analysis of cell proliferation”. *WIREs Developmental Biology* 4.5 (2015), pp. 469–487.

[169] Beate Neumann et al. “Phenotypic profiling of the human genome by time-lapse microscopy reveals cell division genes”. *Nature* 464.7289 (2010), pp. 721–727.

[170] Kara L. McKinley and Iain M. Cheeseman. “Large-Scale Analysis of CRISPR/-Cas9 Cell-Cycle Knockouts Reveals the Diversity of p53-Dependent Responses to Cell-Cycle Defects”. *Developmental Cell* 40.4 (2017), 405–420.e2.

[171] Atilgan Yilmaz et al. “Defining essential genes for human pluripotent stem cells by CRISPR–Cas9 screening in haploid cells”. *Nature Cell Biology* 20.5 (2018), p. 610.

[172] Wayne Stallaert et al. “The structure of the human cell cycle”. *Cell Systems* 0.0 (2021).

[173] Robert J. Prill et al. “Noise-Driven Causal Inference in Biomolecular Networks”. *PLOS ONE* 10.6 (2015), e0125777.

[174] Ning Leng et al. “Oscope identifies oscillatory genes in unsynchronized single-cell RNA-seq experiments”. *Nature Methods* 12.10 (2015), pp. 947–950.

[175] Kevin R. Moon et al. “Visualizing structure and transitions in high-dimensional biological data”. *Nature Biotechnology* 37.12 (2019), pp. 1482–1492.

[176] Daphne S. Bindels et al. “mScarlet: a bright monomeric red fluorescent protein for cellular imaging”. *Nature Methods* 14.1 (2017), pp. 53–56.

[177] Sheng Ding et al. “Efficient Transposition of the piggyBac (PB) Transposon in Mammalian Cells and Mice”. *Cell* 122.3 (2005), pp. 473–483.

[178] Nader Alerasool et al. “An efficient KRAB domain for CRISPRi applications in human cells”. *Nature Methods* 17.11 (2020), pp. 1093–1096.

[179] X. Zhou et al. “Optimization of the Tet-On system for regulated gene expression through viral evolution”. *Gene Therapy* 13.19 (2006), pp. 1382–1390.

[180] Paul Hoyningen-Huene. "Systematicity: The Nature of Science". *Philosophia* 36.2 (2008), pp. 167–180.

[181] D.H. Wolpert and W.G. Macready. "No free lunch theorems for optimization". *IEEE Transactions on Evolutionary Computation* 1.1 (1997), pp. 67–82.

[182] Alejandro F. Villaverde, Nikolaos Tsiantis, and Julio R. Banga. "Full observability and estimation of unknown inputs, states and parameters of nonlinear biological models". *Journal of The Royal Society Interface* 16.156 (2019), p. 20190043.

[183] A. Raue et al. "Structural and practical identifiability analysis of partially observed dynamical models by exploiting the profile likelihood". *Bioinformatics* 25.15 (2009), pp. 1923–1929.

[184] Clemens Kreutz. "Guidelines for benchmarking of optimization approaches for fitting mathematical models". *arXiv:1907.03427 [cs, stat]* (2019).

[185] Helge Hass et al. "Benchmark problems for dynamic modeling of intracellular processes". *Bioinformatics* 35.17 (2019), pp. 3073–3082.

[186] Richard P Feynman. *Richard Feynman's blackboard at time of his death*. 1988.

[187] Antonio Fabregat et al. "Reactome graph database: Efficient access to complex pathway data". *PLOS Computational Biology* 14.1 (2018), e1005968.

[188] Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. "Discovering governing equations from data by sparse identification of nonlinear dynamical systems". *Proceedings of the National Academy of Sciences* 113.15 (2016), pp. 3932–3937.

[189] B. Zhivotovsky and S. Orrenius. "Cell cycle and cell death in disease: past, present and future". *Journal of Internal Medicine* 268.5 (2010), pp. 395–409.

[190] Thomas S. Kuhn. *The structure of scientific revolutions*. International encyclopedia of unified science ; v. 2, no. 2. University of Chicago Press, 1962.

[191] Carl Sagan. *Cosmos: A Personal Voyage*. 1980.

[192] Carmen G. Moles, Pedro Mendes, and Julio R. Banga. "Parameter estimation in biochemical pathways: a comparison of global optimization methods". *Genome Research* 13.11 (2003), pp. 2467–2474.

[193] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. "Optimization by Simulated Annealing". *Science* 220.4598 (1983), pp. 671–680.

[194] Sanjana Gupta et al. "Evaluation of Parallel Tempering to Accelerate Bayesian Parameter Estimation in Systems Biology". *2018 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP)*. 2018, pp. 690–697.

[195] Yanxia Sun, Zenghui Wang, and Barend Jacobus van Wyk. "Local and Global Search Based PSO Algorithm". *Advances in Swarm Intelligence*. Lecture Notes in Computer Science. Springer, 2013, pp. 129–136.

[196] Jason Brownlee. *Clever Algorithms: Nature-Inspired Programming Recipes*. 1st. Lulu.com, 2011.

[197] Jorge Nocedal and Stephen J. Wright. "Trust-Region Methods". *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer, 2006, pp. 66–100.

[198] Michael De La Maza and Deniz Yuret. "Dynamic hill climbing". *AI expert* 9.26 (1994), p. 26.

[199] H. Akaike. "A new look at the statistical model identification". *IEEE Transactions on Automatic Control* 19.6 (1974), pp. 716–723.

[200] Joseph E. Cavanaugh and Andrew A. Neath. "The Akaike information criterion: Background, derivation, properties, application, interpretation, and refinements". *WIREs Computational Statistics* 11.3 (2019), e1460.

[201] M. Stone. "An Asymptotic Equivalence of Choice of Model by Cross-Validation and Akaike's Criterion". *Journal of the Royal Statistical Society. Series B (Methodological)* 39.1 (1977), pp. 44–47.

[202] Yixin Fang. "Asymptotic Equivalence between Cross-Validations and Akaike Information Criteria in Mixed-Effects Models". *Journal of Data Science* 9.1 (2021), pp. 15–21.

[203] Jvstone. *English: Mutual information. The relationships between information theoretic quantities.* 2017.

[204] KonradVoelkel. *English: This diagram shows the relation between the entropies of two random variables X and Y and their mutual information, joint entropy and relative (conditional) entropies.* 2010.

[205] Andrew A. Neath and Joseph E. Cavanaugh. "The Bayesian information criterion: background, derivation, and applications". *WIREs Computational Statistics* 4.2 (2012), pp. 199–203.

[206] Jun Shao. "An Asymptotic Theory for Linear Model Selection". *Statistica Sinica* 7.2 (1997), pp. 221–242.