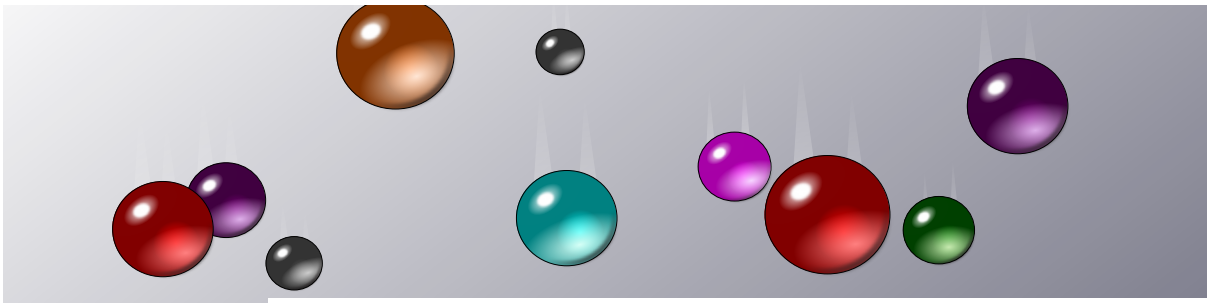
LUND UNIVERSITY

Computational modelling in systems biology: from rewriting cell fates to detecting tumours

Andersson, Emil

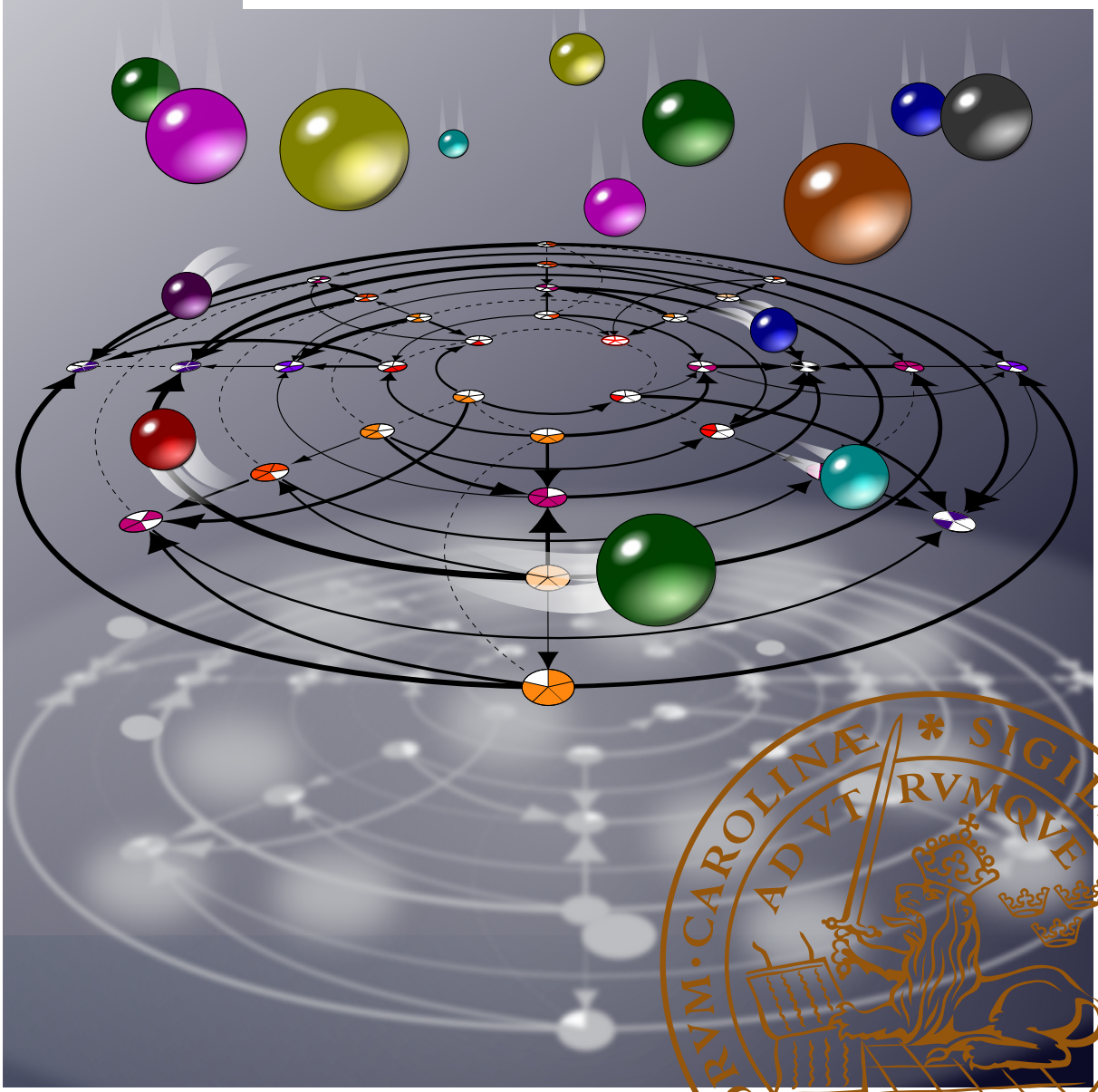
2023

Document Version:
Publisher's PDF, also known as Version of record



Computational modelling in systems biology: from rewriting cell fates to detecting tumours

EMIL ANDERSSON
COMPUTATIONAL BIOLOGY AND BIOLOGICAL PHYSICS | CEC | LUND UNIVERSITY



Printed by Media-Tryck, Lund 2023 NORDIC SWAN ECOLABEL 3041 0903

Reprogramming cells, investigating lineage fate commitment and detecting tumours: A computational study

Faculty of Science
Centre for Environmental and Climate Science
Computational Biology and Biological Physics

ISBN 978-91-8039-859-6

Computational modelling in systems biology
from rewriting cell fates to detecting tumours

Computational modelling in systems biology: from rewriting cell fates to detecting tumours

by Emil Andersson

LUND
UNIVERSITY

Thesis for the degree of Doctor of Philosophy
Thesis advisors: Victor Olariu Ahnell
Faculty opponent: Ala Trusina

To be presented, with the permission of the Faculty of Science of Lund University, for public criticism in Blå hallen at the Centre for Environmental and Climate Science, Computational Biology and Biological Physics on Friday the 8th of December 2023 at 10:00.

Organization
LUND UNIVERSITY

Centre for Environmental and Climate Science, Computational Biology and Biological Physics
Sölvegatan 37,
223 62 LUND, Sweden

Document name
DOCTORAL DISSERTATION

Date of disputation
2023-12-08

Sponsoring organization

Author(s)
Emil Andersson

Title and subtitle
Computational modelling in systems biology: from rewriting cell fates to detecting tumours

Abstract
Computational modelling is becoming an increasingly important tool in biological research. By performing computer simulations of models, it becomes possible to test theories about a biological system against experimental data. Simulations can also be used as a replacement for experiments otherwise unattainable. Well-constructed models have great predictive power which helps improve experimental protocols. All four papers included in this thesis concern the development of computational models of different nature and in different application areas.

In **Paper I**, we develop CELLoGeNe, a software tool which maps Boolean implementation of gene regulatory networks (GRNs) into energy landscapes. Within this framework, cell commitment and reprogramming are considered as movements in an energy landscape. As a part of CELLoGeNe, we develop a tool for visualising multi-dimensional energy landscapes in more than three dimensions. Furthermore, we provide a tool for stochastically analysing the shape of the energy landscape by simulating cell reprogramming in the form of weighted random walks in a landscape. Finally, we demonstrate CELLoGeNe on two GRNs governing different aspects of induced pluripotent stem cells, identifying experimentally validated attractors and revealing potential reprogramming roadblocks.

In **Paper II**, we develop a multi-scale model for early T-cell development. This multi-scale model contains a transcriptional level, an epigenetic level and a proliferation level. The model is tuned to experimental data and predicts state-switching kinetics validated with clonal data. In **Paper III**, we further develop this model by placing it into an agent-based framework. We use the full model to dissect the mechanism of when T-cell progenitors decide to commit the T-cell lineage and what role inheritance plays in the decision.

In **Paper IV**, we develop a machine learning tool that automatically detects skin tumour borders, which could provide useful aid to surgeons. We use data from hyperspectral images, training artificial neural networks only on spectra from small regions representing either healthy tissue or tumour. Then, the trained networks are used to generate prediction. Thereafter, a segmentation algorithm determines the skin tumour borders. Our approach therefore circumvents the need for a complete ground truth image. A separate model instance is trained for each individual patient which makes our approach interesting for emerging precision skin tumour diagnostics where adaptability toward the individual is key.

Key words
computational modelling, systems biology, gene regulatory network, cell reprogramming, energy landscape, iPSC, agent-based modelling, multi-scale modelling, T-cell development, machine learning, tumour diagnostics

Classification system and/or index terms (if any)

Supplementary bibliographical information

Language
English

ISSN and key title

ISBN
978-91-8039-859-6 (print)
978-91-8039-860-2 (pdf)

Recipient's notes

Number of pages
255

Price

Security classification

Signature _____ Date 2023-10-23

Computational modelling in systems biology: from rewriting cell fates to detecting tumours

by Emil Andersson

LUND
UNIVERSITY

A doctoral thesis at a university in Sweden takes either the form of a single, cohesive research study (monograph) or a summary of research papers (compilation thesis), which the doctoral student has written alone or together with one or several other author(s).

In the latter case the thesis consists of two parts. An introductory text puts the research work into context and summarizes the main points of the papers. Then, the research publications themselves are reproduced, together with a description of the individual contributions of the authors. The research papers may either have been already published or are manuscripts at various stages (in press, submitted, or in draft).

Cover illustration front: Cell reprogramming viewed as marbles rolling around in an energy landscape. This is how I envision a CELLoGeNe marble simulation would look like in physical form.

Cover illustration back: Me, on top of *Monte Composto* at Sankt Hans backar: my personal energy landscape. Lund, October 2023.

Funding information: This thesis and the work in it was supported by the United States Public Health Service (NIH) (USPHS grant R01HL119102). This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

Faculty of Science, Centre for Environmental and Climate Science, Computational Biology and Biological Physics

ISBN: 978-91-8039-859-6 (print)
ISBN: 978-91-8039-860-2 (pdf)

Printed in Sweden by Media-Tryck, Lund University, Lund 2023

Lugn är bästa stress

Contents

Acknowledgements

First and foremost, I would like to thank my supervisor Victor. Your enthusiasm and passion for modelling cell regulation are infectious and it is almost impossible to conclude a discussion with you, without a feeling that our model results will revolutionise the field. We complement each other well: when I get too deep into the nitty-gritty details of models, you remind me that the models say nothing if we don't get results, and sometimes I need to remind you that we don't get results without thinking about model details. In addition, thank you for always being positive and filling the corridor with laughter.

Next, I would like to thank the whole group of Computational Biology and Biological Physics (CBBP) for offering such a pleasant work environment. Thanks for all the fun lunch discussions and for always having the possibility to knock on someone's door and get help. Thank you to all the seniors: Anders B, Anders I, Bosse, Carl, Carsten, Mattias, Patrik, Tobias and Victor, for enabling this fruitful climate. A special thanks to my CBBP PhD student-colleagues, both current: Axel, Daqu, Erik, Lucas, Malin, Suze and Thomas; and previous: Adrian, Albertas, Björn, Daniel and Najmeh, for many great lunch meetings and CBBPPPPP-nights. I also want to thank the PhD students at theoretical particle physics for bringing me into your group and letting me join activities such as the daily zoom-push-ups and kubb-nights. I will also take this opportunity to again thank Caroline, Louise, Eva and Camilla for all the help you have given me with administrative tasks.

I would also like to express my gratitude to all my collaborators on the different projects. To Mattias whom I developed CELLoGeNe with; to Carsten, Ellen, Mary and Wen for our cross-Atlantic late-night zoom meetings discussing modelling of T-cells; to Aboma, Malin, Alice, Carl and Patrik for interesting discussions and brainstorming about tumour detection; and to Daniella, Christina and Alexander for all the interesting meetings looking at UMAPs, trying to figure out the PV trajectories. Also, a big thank you to Lucas, Anders B, Anders I, Victor, Carl, Timea and Daniel who proofread and commented on my thesis.

One thing that I have enjoyed immensely during my time at Theoretical Physics is the large community of runners. Thank you to team Teoretisk fysik for all our races together. Thanks to Lucas, Anders I, Johan, Leif G, Andrew and Lucas for all running discussions by the coffee machine. A special shout-out to Robin for our ultra-long discussions about ultra running; they were sometimes almost as long as a few of the races we did together.

Finally, I will express my gratitude to those persons who mean the most to me. To my parents Petra and Leif, thank you for your never-ending support, for the encouragement and homework help during my school years, and for always being interested in trying to understand what I'm working with. To my sister Sofie, thank you for being an important role model and inspiration when we grew up. Even if I sometimes annoyed you (as every little brother should), I always looked up to you and still do. To my nephew Caspian and niece Celine, thank you for all the well-needed playtime. Last, but the opposite to least, to Timea. You have helped me bring my mind from work to focus on what is really important. Your support during these years has meant the world to me.

List of publications

This thesis is based on the following publications:

I **CELLoGeNe – an energy landscape framework for logical networks controlling cell decisions**
E. Andersson, M. Sjö, K. Kaji, and V. Olariu
iScience **25** (8), 104743 (2022)
doi: `j.isci.2022.104743`

II **Multi-scale dynamical modeling of T cell development from an early thymic progenitor state to lineage commitment**
V. Olariu, M. A. Yui, P. Krupinski, W. Zhou, J. Deichmann, **E. Andersson**, E. V. Rothenberg, and C. Peterson
Cell Reports **34** (2), 108622 (2021)
doi: `j.celrep.2020.108622`

III **T-cell commitment inheritance – an agent-based multi-scale model**
E. Andersson, E. V. Rothenberg, C. Peterson, and V. Olariu
Submitted to *Communications Biology*
doi: `bioRxiv: 10.1101/2023.10.18.562905`

IV **Facilitating clinically relevant skin tumor diagnostics with spectroscopy-driven machine learning**
E. Andersson, J. Hult, C. Troein, M. Stridh, B. Sjögren, A. Pekar-Lukacs, J. Hernandez-Palacios, P. Edén, B. Persson, V. Olariu, M. Malmsjö, and A. Merdasa
Submitted to *Cell Reports Medicine*
doi: `medRxiv:10.1101/2023.10.14.23296584`

All papers are reproduced with permission of their respective publisher, with minor stylistic changes in the layout and wording.

Works which are not included in the thesis but were published or worked on during the time of my doctoral studies:

V **Lineage trajectory analysis of human parvalbumin interneurons during reprogramming**
C.-A. Stamouli, A. Degener, A. Bruzelius, **E. Andersson**, J. Giacomoni, E. Cepeda-Prado, S. Kidnapillai, M. Parmar, V. Olariu, and D. Ottosson
In preparation

VI **AI framework for 3D melanoma delineation from PAI data**
A. Fracchia, **E. Andersson**, J. Hult, P. Edén, A. Merdasa, V. Olariu, and M. Malmsjö
In preparation

VII **Hic2 fast-tracks iPS cell generation by suppressing KLF4-dependent epidermal detour during reprogramming**
M. Beniazza, M. Yoshihara, D. F Kaemena, J. Ashmore, S. Zhao, **E. Andersson**, V. Olariu, S. Katayama, K. Yusa, and K. Kaji
Under review at *Nature Communications*

Populärvetenskaplig sammanfattning på svenska

"Ultralöpning är som livet: det finns inga gränser för vad som är möjligt."

— Pace on Earth

Om modeller
Människan har i alla tider kommit på modeller för att beskriva naturens fenomen och uppbyggnad. Redan i antikens Grekland formulerade Demokritos den första modellen för atomens odelbarhet och Arkimedes beskrev sambandet mellan objekts densitet och hur mycket de tränger undan vatten. Under renässansen formulerade Kepler lagarna för planeters banor runt solen och Newton revolutionerade fysiken och dess förmåga att analytiskt beskriva omvärlden genom att formulera lagarna för klassisk mekanik. Med Newtons lagar blev det möjligt att beskriva och – kanske framförallt – förutsäga beteende av allt från en projektils trajektoria till himlakroppars rörelse under samma teoretiska ramverk. Vetenskapsmän kunde göra mycket noggranna beräkningar för hand, men det hade av naturliga skäl sina begränsningar. Datorernas intåg i vetenskapen med start under 1930-talet tillhandahöll ett nytt kraftfullt verktyg och skapade ett nytt forskningsfält: beräkningsvetenskap. Nu öppnades möjligheter att hitta lösningar till avancerade och komplexa system som är omöjliga att lösa med penna och papper, så som vädermodeller och diverse andra sammankopplade system.

Fysiken – vetenskapen om materia, energi och krafter – har länge ansetts vara den främsta vetenskapen när det gäller möjligheten att beskriva verkligheten med precisa teoretiska modeller. Det ansågs länge vara omöjligt att beskriva biologin – livets vetenskap – med fysikaliska modeller på grund av dess komplexitet. Men under 1940-talet banade Schrödinger väg för det nya fältet biofysik. Det var även då som han först introducerade idén om att det skulle kunna finnas en "genetisk kod". Genom att formulera minimalistiska modeller går det att beskriva kärnan av biologiska system. När kärnan har blivit undersökt och förstådd, går det att bygga på och utöka modellerna med detaljer och komplexitet, vilket gör modellernas potential att förutsäga systems beteende ännu större.

Det finns två huvudskolor av beräkningsmodeller: klassiska modeller där modellutvecklaren manuellt definierar modellens alla egenskaper, och så kallade maskininlärningsmodeller där modellen själv anpassar sina parametrar för att passa till datan den får inmatad. Maskininlärning är inspirerat av hur människor lär sig. Föreställ dig hur en förälder skulle lära sitt barn att identifiera bananer och apelsiner. Föräldern skulle först visa en banan och säga "banan" och sedan visa en apelsin och säga "apelsin", och sedan upprepa den här processen många gånger. Om föräldern senare visar upp en banan och får "banan" som svar, då skulle barnet få veta att det hade rätt (troligen genom ett gällt, positivt glädjeutrop). Om barnet

istället hade svarat "apelsin" så skulle det bli tillrättavisat av föräldern. Genom den här processen får barnet själv klura ut hur det ska känna igen frukterna, under uppsikt av föräldern. Det är genom samma princip som övervakad maskininlärning verkar. En maskininlärningsmodell blir visad många objekt tillhörande olika kategorier, med tillhörande etikett, och så anpassas modellparametrarna för att kunna särskilja kategorierna. Detta innebär också att modellutvecklaren faktiskt inte behöver veta exakt vad som definierar de olika grupperna som ska identifieras, modellen definierar själv hur kategorierna särskiljs. För att fullfölja liknelsen med barn och föräldrar skulle en förälder tillhörande den klassiska modelleringsskolan istället säga till sitt barn att "en banan är avlång, böjd och gul, medans en apelsin är klotrund och orange". Från den här analogin kan det kanske framstå som att maskininlärning är den generelt bättre modelltypen, vilket inte stämmer; båda modelltyperna har sina användningsområden där den ena är mer lämplig än den andra. En klassisk modell är användbar när modellutvecklaren har idéer om hur ett system fungerar och vad som är systemets nyckelegenskaper. Genom att implementera detta specifikt, kan modellen testas mot verkligheten för att få reda på om idéerna stämmer eller inte. Maskininlärning å andra sidan är användbar när egenskaperna i sig inte är det viktiga, utan möjligheten att skilja på olika objekt. Då kan modellen själv definiera de särskiljande egenskaperna från den givna datan.

I den här avhandlingen använder och utvecklar jag diverse beräkningsmodeller av båda skolor för att beskriva och utforska två olika områden inom biologi och medicinsk tillämpning. Det första är cellers identitet och förmåga att omprogrammeras. Den andra är att utveckla ett maskininlärningsverktyg för att hjälpa kirurger att identifiera hudtumörgränser.

Om avhandlingen

Människan är ett ytterst komplext biologiskt system som består av circa tio tusen miljarder celler av 200 olika celltyper. Det är celltyper som hudceller, blodceller, muskelceller och hjärnceller – för att nämna några. I ett tidigt stadium av människans utveckling som embryo består vi mestadels av en enda celltyp: stamceller. Dessa är så kallade pluripotenta celler, det vill säga att de har potential att utvecklas och bilda vilken annan typ av cell som helst. Som vuxen människa har man dock inte kvar den här typen av stamceller. Vuxna har en annan typ av stamceller, så kallade multipotenta, som endast kan utvecklas till speciallicerade celler inom en viss cellfamilj. Forskare har funnit att pluripotenta stamceller skulle kunna användas för att ersätta skadade celler om man till exempel har drabbats av en degenerativ sjukdom såsom Alzheimers sjukdom. Det är dock problematiskt att få tag på stamceller att använda till detta då dessa enbart återfinns i embryon, vilket leder till etiska dilemman. Dessutom fungerar användandet av stamceller som vilken annan transplantation som helst såtillvida att kroppen riskerar att stöta bort de nya cellerna. Istället har forskare upptäckt att det går att omprogrammera mogna celler till stamceller. Man kan exempelvis ta en patients hudceller, en celltyp som man har i stor mängd och som dessutom enkelt

kan återbildas, och göra om dessa till stamceller. Då kringår man både det etiska dilemmat av att använda stamceller från embryon och problematiken kring risken att kroppen stöter bort de transplanterade cellerna då de kommer från samma individ.

Så hur går det då till att omprogrammera celler? Alla kroppens celler innehåller samma DNA, alltså samma genetiska kod. Det som avgör vilken celltyp en cell faktisk är beror på vilka gener som är aktiva och vilka som är avstängda. Genom att aktivera och stänga av gener kan man på så vis omprogrammera en cell från exempelvis en hudcell till en stamcell. Det tillkommer dock en del komplexitet som försvårar problemet. Generna interagerar med varandra. Gen A kan exempelvis ha som effekt att den aktiverar gen B som i sin tur stänger av gen C vilken hade som uppgift att hålla gen A aktiv. Geninteraktionerna verkar på det här viset i komplexa nätverk vilket gör det svårt att förutsäga vilken effekt det kommer att ha att sätta på eller stänga av enskilda gener. Detta är anledningen till att omprogrammeringseffektiviteten i laboratorier än så länge är mycket låg. Det är här beräkningsmodeller kommer in i bilden. Genom att formulera ekvationer som beskriver de komplexa genreguleringsnätverken kan vi simulera utfallet av att stänga av och sätta på gener och på så vis få en bättre förståelse för vad det faktiskt är som händer. Därmed kan vi också komma fram till vad man borde göra istället för att få önskat utfall och därmed öka effektiviteten.

I den första av mina artiklar utvecklar jag en ny modell för att få en komplett överblicksbild av vad som kan hända givet ett genreguleringsnätverk och applicerar det på just omprogrammering av hudceller till stamceller. I den andra och tredje artikeln använder jag istället beräkningsmodeller för att undersöka hur utvecklingen av en specifik typ av immunförsvarscell – T-celler – går till och försöker finna mekanismen för när i utvecklingen av ett förstadium till T-cellen som den bestämmer sig för att bli just en T-cell.

Den mänskliga kroppen är som sagt ett oerhört komplext system; därmed är det inte så konstigt att det går fel ibland. Ett fel som kan uppstå är att celler i ett område börjar dela på sig i mycket högre grad än normalt. Detta kan leda till cancerbildning. I den fjärde artikeln jobbar jag specifikt med hudcancer. Ytliga tumörer går ofta att avlägsna kirurgiskt; dock är det viktigt att inte lämna kvar några cancerceller. Därför skär kirurgerna ofta ut tumörerna med stora säkerhetsmarginaler. Detta leder dock till grov ärrbildning vilket kan ge både kosmetiska men framförallt funktionella problem. Därför har jag utvecklat en metod baserad på maskininlärningsverktyg som automatiskt hittar konturerna på tumören. Detta verktyg kan verka som ett hjälpmedel för kirurger att avlägsna tumören med mycket större sannolikhet och med mindre marginaler.

Popular introduction in English

"Sometimes, if you stand on the bottom rail of a bridge and lean over to watch the river slipping slowly away beneath you, you will suddenly know everything there is to be known."

— Winnie the Pooh

Humankind has always tried to describe and explain the world they see around them with models. Already in ancient Greece, Democritus philosophised about the indivisible atom as the smallest constituent of matter and Archimedes described the connection between an object's density and how much it displaces water when submerged. During the Renaissance, Kepler formulated his laws on planetary motion and Newton revolutionised physics by formulating the laws of classical mechanics. With Newton's laws of physics, it became possible to analytically describe, and perhaps more importantly, make predictions for everything from the trajectory of a projectile to the orbits of celestial bodies within the same theoretical framework. Scientists could do precise and advanced calculations by hand, but it naturally had its limitations. This started to change during the 1930s when the computer made its first entrance into the scientific world and opened the door to a new scientific field: scientific computing. With the computer as a tool, more complex models and algorithms could be developed, such as weather models, coupled systems and Monte Carlo simulations.

Physics – the study of matter, energy and forces – is often considered to be the prime field of science when it comes to constructing models describing the world. For a long time, it was considered impossible to describe biology – the science of life – with physical models due to its complexity. But during the 1940s, Schrödinger sparked the new field of biophysics when he tried to explain how life apparently defies the 2nd law of thermodynamics and he also introduced the idea of a genetic code. With minimalistic models, it now became possible to describe the core of a biological system and gain a deep understanding of the system. When a minimalistic model has been developed and understood, it can be extended to include more details and complexity, making its predictive power greater.

There are two main branches of computational models: classical models, in which the developer manually defines all the model's features and details, and so-called machine learning models where the model itself adapts its parameters to fit data presented to it. Machine learning is inspired by how humans learn. To contrast classical models with machine learning models, consider how a parent would teach its child to recognise and distinguish bananas from oranges. The parent would first show a banana and say "banana", then show an orange and say "orange", and repeat this many times. If the child answers "banana" correctly when later being presented with a banana, it would get to know that it was correct (probably with a positive squeaking sound), but if the child instead answers "orange" it

would be corrected by the parent. By this process, the child gets to figure out how to recognise the two types of fruit by itself, under the supervision of the parent. Machine learning models use the same principle. A machine learning model is shown many instances of each class together with the correct label, and the model adjusts its parameters to be able to distinguish the classes. To complete the analogy of parenting, a parent using the classical programming approach would instead tell the child: "a banana is yellow, elongated and curved while an orange is spherical and orange". From this analogy, it could seem like the classical approach is worse, but both approaches have their own different cases when they are more appropriate. A classical model is useful when the scientist has ideas about how a system works and what the key properties are. By programming this specifically, one can test against reality if these ideas are correct or not. Machine learning, on the other hand, is useful when the ability to distinguish objects is important, not the objects' properties. Then the model can define the distinguishing properties itself from the given data.

In this thesis, I am developing and using models from both schools of modelling to describe and explore two different areas of biology. I am using the classical approach to analyse cell identities and their ability to be reprogrammed, while I use the machine learning approach to develop a tool to aid surgeons in identifying skin tumour borders.

List of abbreviations

DNA	–	**D**eoxyribo**n**ucleic **A**cid
RNA	–	**R**ibo**n**ucleic **A**cid
mRNA	–	**m**essenger **RNA**
GRN	–	**G**ene **R**egulatory **N**etwork
TF	–	**T**ranscription **F**actor
ODE	–	**O**rdinary **D**ifferential **E**quation
MSE	–	**M**ean **S**quared **E**rror
ML	–	**M**achine **L**earning
ANN	–	**A**rtificial **N**eural **N**etwork
CNN	–	**C**onvolutional **N**eural **N**etwork
MLP	–	**M**ulti**l**ayer **P**erceptron
HSI	–	**H**yper**s**pectral **I**maging
CELLoGeNe	–	**C**omputation of **E**nergy **L**andscapes of **Lo**gical **Ge**ne **Ne**tworks
iPSC	–	**i**nduced **P**luripotent **S**tem **C**ell
MEF	–	**M**ouse **E**mbryonic **F**ibroblast
ESC	–	**E**mbryonic **S**tem **C**ell
LCA	–	**L**ast **C**ommon **A**ncestor
ETP	–	**E**arly **T**hymic **P**rogenitor
DN	–	**D**ouble **N**egative
WT	–	**W**ild-**T**ype
KD	–	**K**nock**d**own

Introduction

"The day science begins to study non-physical phenomena, it will make more progress in one decade than in all the previous centuries of its existence."

— Nikola Tesla

This thesis is a collection of the research papers I produced during my 4.5 years as a PhD-student. During this period, I have ventured from a background in theoretical physics into the interesting world of the interdisciplinary sciences of computational biology, biological physics and systems biology. From my background, I brought computational tools and the physicist's way of thinking about models, which I have applied in, for me, new scientific areas. After these years, I have realised how complex biological systems are, as well as what impressive work experimentalists are doing in their wet labs, but I have also seen that their work can be better understood, and the outcome of their experiments predicted, with the help of the work of theoretical modellers.

The first three papers of my thesis concern model development within systems biology while the fourth one is about developing a machine learning model for tumour diagnostics. In Paper I, we develop a model that converts a gene regulatory network into an energy landscape. With this model, it is possible to simulate cell reprogramming as a random walk in the energy landscape, which provides a full overview of the possible cell states and reprogramming roadblocks. Papers II and III both concern modelling the early development and commitment of T-cells. In paper II, we develop a multi-scale model which is fitted to and validated with experimental data. In paper III, we put this multi-scale model into an agent-based framework and investigate the role of inheritance in the commitment process. Finally, in PaperIV, I develop – in close collaboration with surgeons – a machine learning model which finds the borders of superficial skin tumours from hyperspectral images. This model aims to aid surgeons in safely excising tumours with minimal margins.

In the introductory part of this thesis, I will introduce sufficient background knowledge so that the four papers should be understandable for those with physics, computational science and biology backgrounds alike.

- In Section 1, I introduce the biological aspects related to the papers.
- In Section 2, I give an overview of how biological systems can be connected to computational models through systems biology.
- In Section 3, I go through the most important computational tools I have used.
- In Section 4, I provide some final concluding remarks.

Finally, I give an overview of the included papers and specify my contribution to each of them.

1 Biological background

"Life finds a way."

— Dr. Ian Malcolm, Jurassic Park

Biology – the science of life – is a complex science which differs a lot from physics. Physics studies matter and energy, which behave in predictable ways, while biology deals with life's unpredictability; "Life finds a way" as famously uttered in Jurassic Park. Yet, by breaking down the building blocks of life, *cells*, in their smallest constituents, it is possible to apply physical models to explain how life operates.

In this chapter, I will give a summary of the most essential parts of biology needed to understand the papers included in this thesis. I will describe the cells that build up humans, what differentiates different cell types, how DNA – the code of life – is read, how genes are expressed and regulated, and what happens when cell division goes wrong and creates tumours. The topics summarised below can be found in detail in standard text books about cell biology [1, 2].

1.1 Cells: The building blocks of life

"...cells evolved to function and did not evolve to be comprehensible..."

— Uri Alon

The human body is incredibly complex. We are all built up of small living building blocks, cells, which provide both structure and functionality. There are approximately 10^{13} individual cells in the body, but they can all be categorised into 200 cell types [1]. These vary in size between approximately 7 µm and 120 µm [3]. The different cell types constitute the different tissues in our bodies. A few examples of cell types are muscle cells, blood cells, neurons, and liver cells, as depicted in Figure 1. Despite having very different shapes and functions, all cells contain the same DNA (deoxyribonucleic acid). DNA, contains the complete genetic information for an individual. However, not all genetic material is active in all cells. Which cell type a certain cell belongs to is determined by which genes that are active and – as importantly – inactive.

The human body contains many different specialised cell types, but during the early stages of development as an embryo, we mostly contain one unspecialised cell type: *stem cells*. These cells are said to be pluripotent, meaning that they have the potential to differentiate into any specialised cell type. As adults, we actually have some stem cells in our bodies as

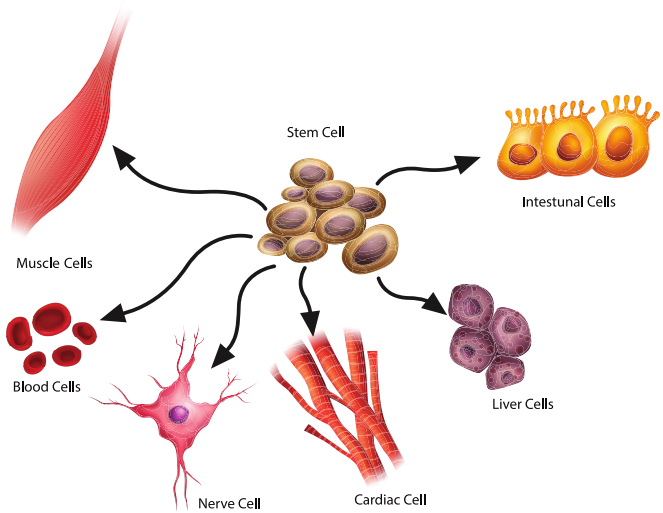


Figure 1: Examples of a few different cell types in the human body. Image credit iStock.com/blueringmedia.

well. However, these are not pluripotent, meaning that they can only differentiate within a specific cell lineage and are said to be multipotent. For instance, a haematopoietic stem cell can only differentiate into different kinds of blood cells and immune cells, such as red blood cells and T-cells.

Replacement of old or damaged cells, as well as development of new specialised cells are conducted by cells that undergo the cell cycle. During the cell cycle, cells divide into two 'identical' copies which in some cases later differentiate. The cell cycle consists of two main phases: the interphase where the cells grow and duplicate the genetic material, and the mitosis where the actual division takes place. Cells spend roughly 90 % of their lifetime in the interphase which is also when they carry out their cell functions. The fastest dividing cells duplicate daily, while most cells duplicate on the scale of weeks or months [3]. Some cells, e.g. those constituting the central nervous system, do not go through cell division at all. This makes them vulnerable to damage since the body cannot replace them. The interphase is divided into several phases which are controlled by specific cell cycle genes. If the cell does not fulfil the right conditions or does not function properly, it can not pass on to the next phase. The cell cycle is then either paused or the cell can undergo apoptosis, i.e. programmed cell death[1]. Sometimes, these control mechanisms are not enough to stop malfunctioning cells, resulting in that the malfunctioning cells undergo cell division in an uncontrolled manner. This can lead to an accumulation of malfunctioning cells, which, if malevolent, develop into cancerous tumours in the body.

[1]As opposed to uncontrolled cell death called necrosis.

1.2 DNA and genes: The code of life

"Any fool can write code that a computer can understand. Good programmers write code that humans can understand."

— Martin Fowler

DNA is often described as a "blueprint" for the human body or a computer program detailing how to construct and maintain the human body. Both of these analogies are right in their on way, but these pictures may be quite misleading. When comparing the DNA to a blueprint, it sounds like the DNA contains a sketch of exactly how the human body will look like and function. This is only true for a few properties; it is often possible to deduce the eye colour or blood type from someone's genes, but nowhere in the DNA is stated exactly how tall one will become or even that most humans have 10 fingers. The DNA does not simply contain a master plan for humans. The comparison to a computer program gives a slightly more accurate picture. The DNA code itself is built up information-wise in a similar way as computer code, which I will return to soon. The DNA contains instructions on how to build proteins which are then tasked with all possible functions in the body. This is similar to how a computer program can be programmed to perform specific tasks. However, the DNA has no "main method" which specifies how, and in which order, the commands should be executed. DNA is more like a list containing all possible functions but with no instructions for how and when to run them.

Before diving deeper into the structure of DNA and how it produces proteins, I need to briefly introduce what proteins are. Proteins are more than the building blocks of muscles, which is probably what most people know about them. Proteins are a class of complex

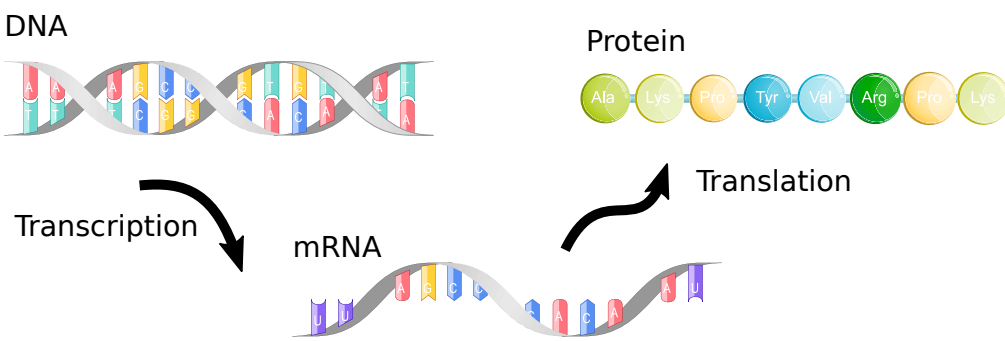


Figure 2: The double-helix DNA molecule uses the four nucleotide bases `A`, `C`, `G` and `T` to code the genetic material. A gene gets transcribed into the single-helix mRNA molecule by RNA polymerase. RNA uses the nucleotide `U` instead of `T`. The ribosome translates the mRNA into proteins, assembling amino acids, where each amino acid is coded by a set of three nucleotides called a codon. Image credit iStock.com/ttsz

molecules which has countless tasks. Besides providing structural components for the body and cells, they can act as catalysts of chemical reactions (these proteins are called *enzymes*), and facilitate cells' communication, both within a cell and between cells by carrying signals and acting as transporters. Antibodies, another kind of protein, have important roles in immune system. There is another set of proteins, *transcription factors* (TF), that regulate which genes are read from the DNA.

Now, let us return to the DNA and how proteins are produced. Similarly to how information on a computer is stored in bits as zeros and ones, DNA stores its information in just four components called nucleotide bases. These are adenine, cytosine, guanine and thymine, commonly referred to using only the first letter of each name (`A`, `C`, `G` and `T`). The structure of a DNA molecule is illustrated in Figure 2. The nucleotide bases are arranged into groups of three letters, forming a codon, similar to a byte in a computer. Since there are four different letters, `A`, `C`, `G` and `T`, the codons can code for $4^3 = 64$ different objects. These objects are *amino acids* which are the building blocks of proteins. However, there are only 22 different amino acids constituting proteins, meaning that there is some redundancy. For instance, the amino acid lysine is coded by both `AAA` and `AAG`. This is a smart feature since it allows for small coding errors; different combinations of nucleotides still can yield the same amino acid. The codons are structured into genes where a gene spells out a certain sequence of amino acids that constitutes a protein. The human genome is estimated to contain approximately 21 000 protein-coding genes [3].

1.2.1 Gene regulation

The process of how the information contained in the genes is turned into proteins can be divided into two parts: transcription and translation, as illustrated in Figure 2. During the transcription, the information contained in a gene is copied into a slightly different format: the messenger RNA (mRNA). RNA is similar to DNA but with a few notable differences: it contains ribonucleic acid instead of deoxyribonucleic acid (hence the R instead of the D in the name), it is much shorter than the DNA (since it only contains information for one gene), and it uses uracil (`U`) instead of thymine (`T`). A gene is transcribed into mRNA by a protein called RNA polymerase. The same gene can be transcribed multiple times creating many copies of the same mRNA. By transcribing a gene into mRNA, the genetic information becomes accessible for the next step of the protein creation. During translation, proteins are produced from the information contained in the mRNA. A *ribosome*[2] reads the codons and connects corresponding amino acids delivered by transcript RNA, constructing the encoded protein step-by-step . As opposed to DNA, mRNA degrades after a while, meaning that the protein cannot be translated any more.

[2]A ribosome is a big and complex structure consisting of ribosomal RNA (rRNA) and many proteins.

However, the genes are not just transcribed passively. There are mechanisms to regulate the transcription. Each gene has a promoter region nearby on the DNA strand. Activating or repressing proteins – transcription factors (TFs) – can bind to the promoter regions and, by doing so, either enhance the production or inhibit it. Smaller molecules in the cell environment can also act as regulators by binding to the promoter regions or regulating proteins. For some genes, several proteins can act as regulators. For others, several proteins are required to cooperate. This creates a complex system of gene regulation which I will return to later in Section 2.

1.2.2 Epigenetics

To make the matter of gene regulation even more complex, there are external regulation mechanisms as well: *epigenetic* mechanisms [4, 5]. The prefix *epi* means "above", so the epigenetics alludes to "above the genome". Epigenetic mechanisms do not alter the genetic material but rather affect how the genes are read. These mechanisms are connected to how the DNA is structured in the cell. The DNA strand is not floating around in the cell unstructured, it is wound around histone proteins. The histones are coiled together in larger and larger structures, building up a chromosome, as illustrated in Figure 3. For a gene to be transcribed, the DNA strand needs to unwind locally around the gene.

An epigenetic mechanism which controls the winding of DNA is called histone acetylation. An acetyl group binds to the histone which loosens up the coiled DNA enabling it to be transcribed. If the acetyl group is unbound, the DNA is tightened again, blocking transcription. Another epigenetic mechanism is DNA methylation which is when a methyl

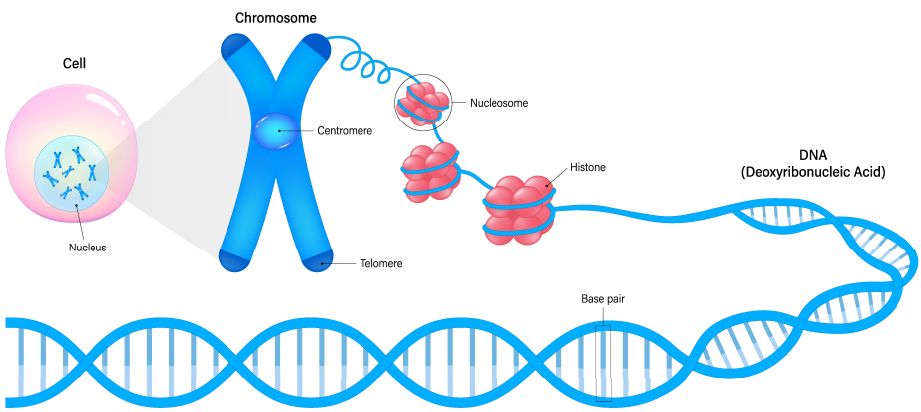


Figure 3: The DNA strand is tightly wound around histones which coils up into chromosomes, located in the cell's nucleus. Image credit iStock.com/Rujirat Boonyong.

group binds directly to the DNA. The methyl group acts as a blockade for the DNA polymerase, hindering the gene to be transcribed. There is a third epigenetic mechanism that works similarly to the DNA methylation, but post-transcriptionally on the mRNA. In this mechanism, small snippets of RNA, microRNA, bind to matching sites of the mRNA, hindering the ribosome from translating the mRNA into a protein. An important feature of the epigenetic mechanisms is that they are inheritable between cells. For instance, if a cell has methylated a gene, the gene will often stay methylated in the daughter cells after the cell divides.

2 Systems Biology: Bridging biological systems with computational modelling

"Ask five biomedical researchers to define systems biology, and you'll get 10 different answers…or maybe more."

— Christopher Wanjek

Systems biology is a new field of science, emerging around the year 2000, which – as hinted by the quote above – does not have one singular clear definition. A rather unconventional explanation of systems biology was given in a Nature editorial [6]:

> *"What is the difference between a live cat and a dead one? One scientific answer is 'systems biology'. A dead cat is a collection of its component parts. A live cat is the emergent behaviour of the system incorporating those parts."*

A rather more descriptive definition of systems biology was provided by Ron Germain [7]:

> *[Systems biology is] "a scientific approach that combines the principles of engineering, mathematics, physics, and computer science with extensive experimental data to develop a quantitative as well as a deep conceptual understanding of biological phenomena, permitting prediction and accurate simulation of complex (emergent) biological behaviours."*

To summarise these perspectives, one can say that systems biology is an interdisciplinary field of science that combines mathematics with computational analyses to describe complex biological systems. The field stems from the idea that system properties emerge from a complex system which are not derivable by investigating the components' properties individually, but where every component needs to be studied in the setting of the full complex system. Some definitions restrict systems biology to only include complex systems within cell biology, but it could also be considered as a more general field where applications such as studying population dynamics or disease spreading are also included. In this thesis, I am using the techniques of systems biology to study gene regulatory networks which govern cells' gene expressions and ultimately also cell states.

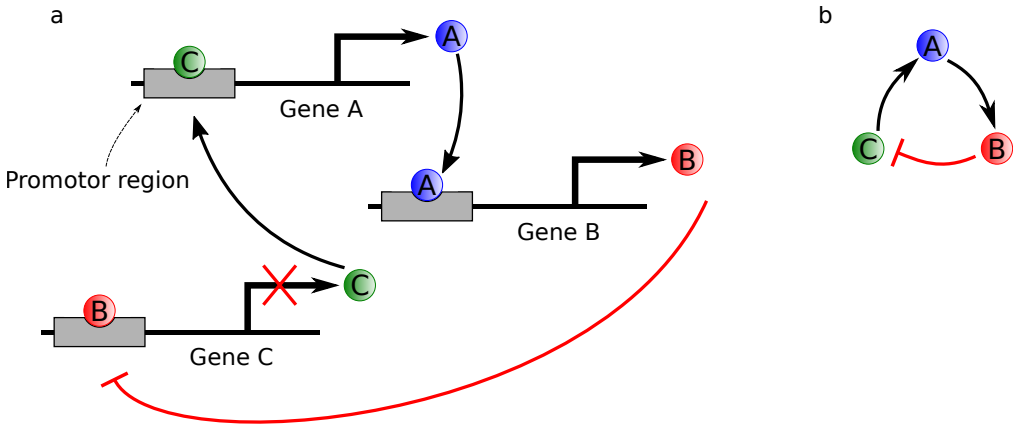


Figure 4: (**a**) An example of a gene regulatory network (GRN). Gene A may code for the activator for gene B which codes for the inhibitor of gene C, which in turn codes for the activator of gene A. (**b**) A simplified depiction of the same GRN where the step of a "gene A coding for the activating promotor of gene B" has been merged into "gene A activates gene B". Activation is depicted with black arrows and inhibition is depicted with red blunted arrows.

2.1 Gene regulatory networks

"Everything should be made as simple as possible, but not simpler."

— **Albert Einstein**

As described in Section 1.2.1, the transcription of genes is regulated by proteins called *transcription factors* (TFs). Transcription factors may have activating as well as inhibiting effects. What makes gene regulation extra complex is that many genes code for proteins that act as regulators for other genes, which in turn code for regulators of other genes, which in turn… and so forth. For instance, gene A may code for the activator for gene B, which codes for the inhibitor of gene C, which in turn codes for the activator of gene A, creating a feedback loop. In order to keep track of how the genes interact, one can construct a gene regulatory network (GRN) as depicted in Figure 4a. One simplification that is often made to keep the GRNs clearer, is to merge the step "gene X codes for the regulator of gene Y", and simply say "gene X activates/inhibits gene Y". Correspondingly, the steps are also merged in the GRN, as shown in Figure 4b. From now on, we will keep to this simplified view and use expressions such as "gene X is produced" instead of the more cumbersome "the protein encoded by gene X is produced". In depictions of GRNs, I will use black arrows to represent activations (X ➔ Y) and red blunted arrows to represent inhibition (X ⊣ Y).

GRNs are important tools within systems biology since they describe how cells are governed. Common research topics in systems biology are to describe cell development or reprogram-

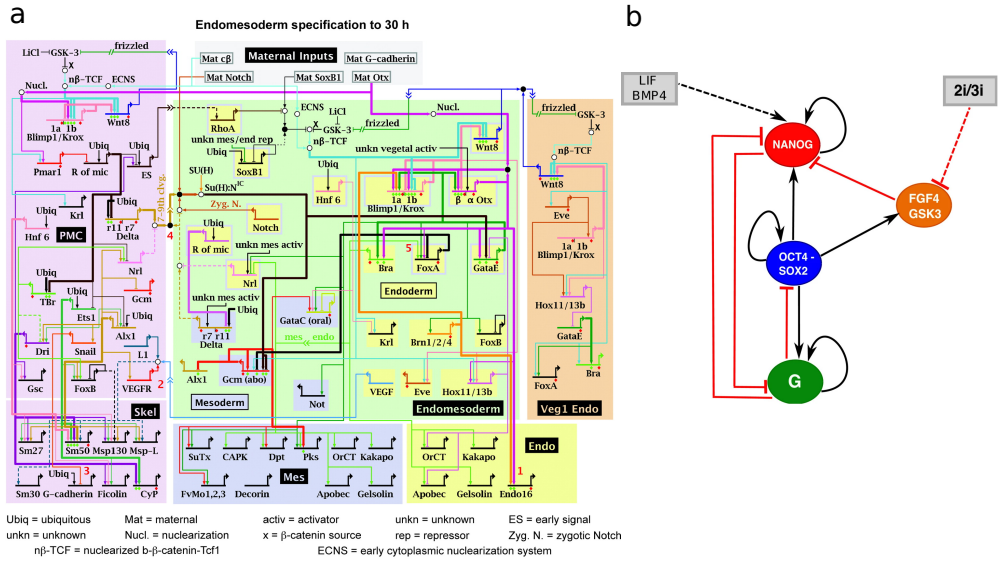


Figure 5: Examples of gene regulatory networks (GRNs). (**a**) The full developmental GRN of the sea urchin, figure taken from Ref. [8]. (**b**) A minimal core GRN governing maintenance of embryonic stem cells, figure taken from Ref. [9].

ming[3] of cell identities. Even with today's scientific technologies, it is close to impossible to map the complete gene regulatory network of an organism, with the exception of a few simple organisms such as the sea urchin [8] which is depicted in Figure 5a. Thankfully, a complete GRN of an entire organism is not necessary for most applications. It is often enough to isolate the specific genes involved in a certain process. When modelling cell development or reprogramming, it is often desirable to construct a minimal GRN, containing only the minimal amount of genes which still capture the necessary transcriptional dynamics. An example of a minimal GRN is shown in Figure 5b. This GRN governs the maintenance of embryonic stem cells and consists of only four core genes and two external signals. GRNs can be constructed by performing experiments which elucidate how genes interact. When a GRN has been discovered, it can be used to simulate cell development or reprogramming to gain understanding of a system and make predictions. In chapter Section 3, I will describe common methods to simulate GRNs. In Papers I, II and III, GRNs are key parts where we in Paper I develop a general model to analyse GRNs and in Papers II and III develop a GRN for a specific cell development process.

[3]Cell reprogramming is when one cell type is converted to another, this will be covered in Section 2.4.

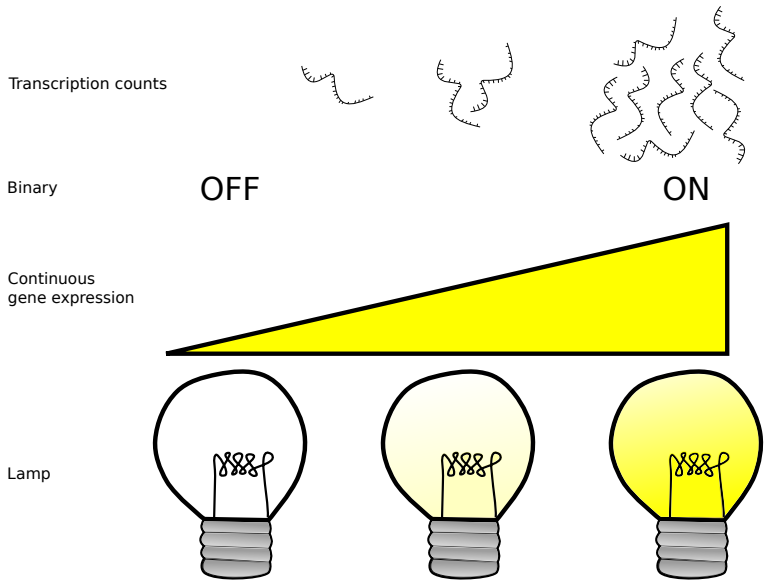


Figure 6: Different treatments of gene expression. Gene expression can be measured by counting each copy of mRNA transcripts. In the binary treatment, the genes can only be either `ON` or `OFF`. The transcription counts can also be transformed into a continuous gene expression value. The two treatments can be compared to a lamp which can be either switched `ON` or `OFF` (binarised), or shine with an intermediate brightness if connected to a dimmer (continuous).

2.2 Gene expression

"Happiness can be found, even in the darkest of times,
if one only remembers to turn on the light."

— **Albus Dumbledore**, Harry Potter and the Prisoner of Azkaban

A common way to treat gene expression is to binarise it. If the gene is transcribed, the gene is considered `ON`, and if the gene is not transcribed[4], the gene is considered `OFF`. This can be viewed as the behaviour of a light switch, where a gene can be turned `ON` or `OFF` just as a lamp can, as illustrated in Figure 6. However, even though binary gene expression is a useful simplification, it is not enough for studies of gene expression rate dynamics. Since gene expression is based on mRNA transcription counts, a gene can be expressed at many different levels, similar to how the brightness of a lamp can be controlled with a dimmer.

The binary gene expression can be expressed mathematically by claiming that a gene's expression g exists in the discrete space $g \in \mathbb{B} = \{0, 1\}$. If we consider a GRN with n

[4]Or only transcribed at a basal rate, meaning the small activity of transcription that can occur even without positive regulation.

genes, the cell can be described by a state vector $\boldsymbol{s} = (g_0, g_1, \ldots, g_{n-1})$ where $\boldsymbol{s} \in \mathbb{B}^n$. Hence, the cell can only be in any of the 2^n states. For the continuous gene expression, g can instead take a value in the interval $[0, 1]$, where the expression level is assumed to be normalised to the fully expressed transcription count.

Using the continuous description has the drawback that the number of possible cell states is not limited as for the binary case; instead this description presents a more accurate representation. The two treatments of gene expression are useful in different settings, and I have used both formalisms in this thesis: I use binarised gene expression in Paper I since it allows us to gain an overview picture of complete cellular reprogramming landscapes. In Papers II and III, we instead use the continuous description since we conducted dynamic simulations of a GRN.

2.3 Cell reprogramming

"Until man duplicates a blade of grass,
nature can laugh at his so-called scientific knowledge."

— Thomas Edison

Cell development potential is often envisaged as a mountain, as first suggested by Waddington in 1957 [10]. A pluripotent cell – which can become any other cell type – can in this analogy be thought of as a marble located at the top of the mountain peak, as illustrated in Figure 7. Differentiation of the cell, leading to it becoming more specialised, can be viewed as the marble rolling down the mountain on differentiation paths. here are a few different branching paths at the top that the marble could take, leading the pluripotent cell to commit to certain cell lines, such as haematopoietic cells or neuronal cells. Then the marble continues to roll down choosing between finer and finer branching points and ultimately becoming a specialised ("mature") cell, as depicted with the coloured paths in Figure 7.

In natural development, the marble can only roll down the mountain (cyan arrow in Figure 7). However, by forcing the expression of certain transcription factors, it is possible to reprogram cells, i.e. push the marble up the mountain and let it roll down another path (magenta arrow in Figure 7). This was first done by reprogramming mouse embryonic fibroblasts into myoblasts in 1987 [11]. Reprogramming one cell type directly into another like this is called direct reprogramming. In 2006 and 2007, Yamanaka and Takahashi showed in mice and human cells, respectively, that it is possible to reprogram fibroblasts into stem cells, so-called induced pluripotent stem (iPS) cells [12, 13]. They showed that it was possible to do this by just activating the four transcription factors Klf4, Sox2, Oct3/4 and

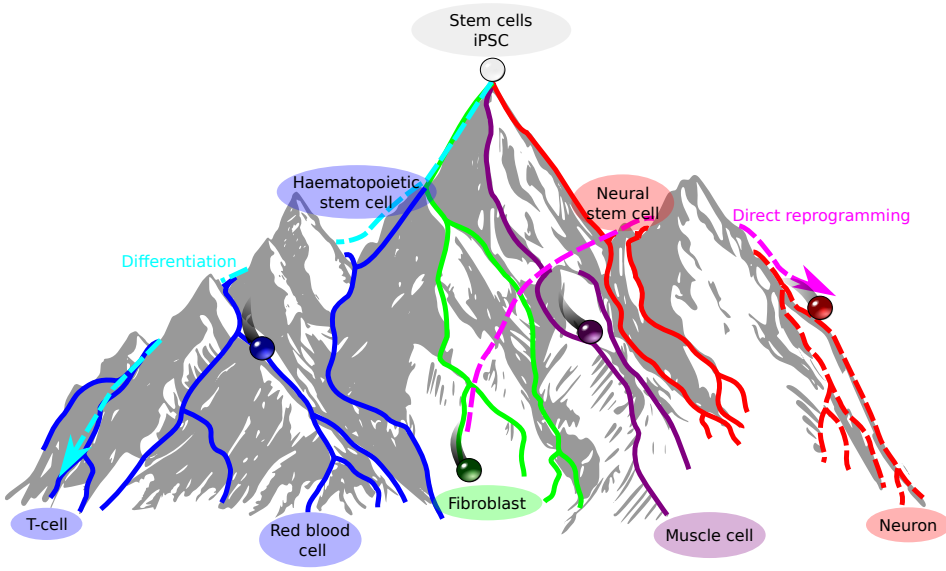


Figure 7: A Waddington landscape of cell developmental potential envisaged as a mountain [10]. During cell differentiation, a cell (here illustrated as a marble) rolls down the mountain on paths leading to more and more specialised cell fates. By pushing a marble up the mountain and letting it roll down another path, the cell is reprogrammed. Image credit (mountains) iStock.com/Grishina Tatiana.

c-Myc that are normally ON in stem cells. The iPS cells have the same properties as pluripotent stem cells and can be used to generate any cell type in the body [14]. This gives iPS cells great potential to be used in medical treatment as they could be used to regenerate damaged cells. They also have two great benefits compared to other available sources of stem cells – embryonic stem cells. Firstly, the ethical issue of using cells harvested from embryos is avoided. Secondly, it is possible to use the patient's own cells to obtain the iPS cells, meaning that the body has a lower risk of rejecting the cells.

Reprogramming cells is, however, not entirely problem-free. Due to the complexity of GRNs, the process is often very inefficient. The conversion rate for obtaining iPS cells is often below 1 % [14]. The unconverted cells may either not be affected, or may happen to end up in another unwanted cell state. Some cell states act as roadblocks hindering the marbles passing through and becoming the target cell type. Therefore, it is of great importance to construct and study GRNs for the conversion process. By understanding the governing GRN, the reprogramming can potentially be made more efficient.

In Paper I, we develop a model, CELLoGeNe[5], where a complete *energy landscape* is calcu-

[5]The name CELLoGeNe is an abbreviation that stands for Computation of Energy Landscapes of Logical Gene Networks.

lated for a given GRN. An energy landscape can in this context be viewed as a quantitative version of the mountain analogy of the developmental potential. By using binary gene expression, CELLoGeNe assigns an energy value to each of 2^n possible cell states. Low energy is assigned if the cell state agrees with the effect of the GRN and high energy is assigned in case of disagreement. As an example, consider the simple GRN A ⊣ B. The state where A is ON and B is OFF is in complete agreement with the effect of the GRN[6] and is assigned low energy (-1). The state where both gene A and B are ON disagrees with the GRN and is assigned high energy $(+1)$. An important feature of CELLoGeNe is that a gene that is OFF does not affect its target gene, i.e. both states where A is OFF are assigned neutral energy (0).[7] If we now consider a cell as a marble in the energy landscape, it will roll down the landscape towards lower energies by transitioning to neighbouring states, i.e. states that only switch the expression of one gene. For a GRN governing cell reprogramming, the energy landscapes provide us with a complete overview of potential roadblocks where cells could get stuck. Hence, we can simulate cell reprogramming as a random walk in the energy landscapes, as artistically envisaged on the front cover of this thesis.

[6] That is, gene A wants gene B to be OFF.

[7] This is not the standard treatment when considering binarised gene expressions. Then A ⊣ B is translated into the logical rule B = NOT A which means that A activates B by being OFF, thus introducing symmetry despite the network being directed.

3 Computational modelling

"All models are wrong, but some are useful."

— George Box

As mentioned previously, models have been used by humankind for a long time, but why are theoretical models actually needed? Is it not enough to just perform experiments and collect data? This is of course as important as constructing a theoretical model. Without the connection to reality a model would be useless, but without a model, one would not be able to use the information gained in the experiments. It is of course important to remember, as stated in the quote by George Box, that all models are wrong; they are just idealised pictures of reality. However, a model provides a setting in which to generalise and make predictions. With a well-constructed model, one can calculate the outcome of a future event. For example, by knowing the model for object trajectories, one can know where a ball is going to land if we know with what velocity we throw it.

In many fields of science, computational models become very important complements to or even substitutes for experiments. By performing computer simulations of a model, one can predict outcomes of experiments which can confirm (or disprove) the theory. In some cases, simulations can be performed when the corresponding experiments are unattainable, for instance, if it would be too expensive to perform or if the technology has not been invented yet. In these cases, the simulation outcome may either confirm that this experiment would be important enough to invest the money in, or the experiment would not even be needed anymore since the results of the simulations provided the needed insights to progress.

The above reasoning is certainly true in cell biology and reprogramming of cells. By constructing models of cell reprogramming, we can perform simulations to test different hypothesis. For instance, what would happen if gene X is removed, or if gene Y is added to the GRN in a certain way? Many of these questions could be answered in the wet lab as well, but by performing *in silicio*[8] simulations, one can test many different hypotheses resource-efficiently. For instance, in Paper I, we develop a model which provides an overview of all potential cell states given a GRN. Thus, we can use this model to predict where cells get stuck in the cell reprogramming process, and gain an understanding of how this could be avoided during experiments.

In all the projects in my thesis, computational models are central parts and the computer is my lab. For this reason, I find it important to introduce these essential tools in detail. In the following sections, I will describe the most important tools I have used.

[8] *In silico* means performing experiments on a computer (i.e. simulations) where "*silico*" refers to the silicon in computer chips. In biology, similar terms exists, such as *in vitro* ("in glass") referring to "test-tube" experiments and *in vivo* ("within the living") referring to experiments inside a living organism.

3.1 Dynamical modelling of gene regulation

"I know some things. I can, you know, do math and stuff."

— **Harry Potter**, Harry Potter and the Philosopher's Stone

A common way to model cell development or reprogramming is by constructing a dynamical model of the genes' expression levels [15]. When constructing a dynamical model, one is interested in the *rate of change*[9] of a gene X, which in the general form can be described by an ordinary differential equation (ODE)

$$\frac{\mathrm{d}X(t)}{\mathrm{d}t} = f(t), \tag{1}$$

where f describes how the level of X changes and $X(t)$ is the expression level of X at time t. Simply put, there are only two things that can happen: a gene can be transcribed into mRNA or the existing mRNA can be degraded. Thus, f can be split into two parts

$$\frac{\mathrm{d}X}{\mathrm{d}t} = p(t) - d(t), \tag{2}$$

where $p(t)$ is the production rate and $d(t)$ is the degradation rate. As described previously, gene regulation is governed by complex networks, GRNs, within which the genes affect each other. Thus, in general, for a GRN including N genes, we instead have a set of coupled ODEs where the production and degradation rates depend on the GRN genes:

$$\begin{aligned}
\frac{\mathrm{d}X_1(t)}{\mathrm{d}t} &= p_1(X_1(t), \ldots, X_N(t)) - d_1(X_1(t), \ldots, X_N(t)) \\
\frac{\mathrm{d}X_2(t)}{\mathrm{d}t} &= p_2(X_1(t), \ldots, X_N(t)) - d_2(X_1(t), \ldots, X_N(t)) \\
&\vdots \\
\frac{\mathrm{d}X_N(t)}{\mathrm{d}t} &= p_N(X_1(t), \ldots, X_N(t)) - d_N(X_1(t), \ldots, X_N(t)).
\end{aligned} \tag{3}$$

Collecting the gene expressions into a vector defined as $\boldsymbol{X}(t) = (X_1(t), \cdots X_N(t))$, this can be compactly expressed as

$$\frac{\mathrm{d}\boldsymbol{X}(t)}{\mathrm{d}t} = \boldsymbol{p}(\boldsymbol{X}(t)) - \boldsymbol{d}(\boldsymbol{X}(t)), \tag{4}$$

where the production and degradation rates are also expressed in vector format. Given a set of initial conditions $\boldsymbol{X}(t_0) = \boldsymbol{X}_0$, where t_0 is the initial time, the system of ODEs can be

[9] E.g. the transcription rate of the gene minus the degradation of the mRNA.

solved to find the dynamics of the gene expressions. A set of coupled ODEs can in general not be solved analytically, and must instead be solved numerically. This can either be done deterministically or stochastically, which will be described in Section 3.2.

So how can a GRN be translated into a set of rate equations? There are two main approaches: using the law of mass action or principles of statistical mechanics. We will now investigate these two approaches separately.

3.1.1 Law of mass action

The law of mass action is a way to calculate reaction rates for chemical reactions of the form

$$\nu_1 X_1 + \ldots + \nu_m X_m \underset{k_b}{\overset{k_f}{\rightleftharpoons}} \nu_{m+1} X_{m+1} + \ldots + \nu_n X_n, \tag{5}$$

where ν_i are stoichiometric[10] coefficients defining how many copies of reacting (molecular) species X_i are required, and k_f and k_b is the forward and backward rate respectively. The law of mass action [16] states that the reaction rates are

$$\frac{\mathrm{d}X_i}{\mathrm{d}t} = \begin{cases} -k_f \prod\limits_{1\leq j\leq m} [X_j]^{\nu_j} + k_b \prod\limits_{m+1\leq j\leq n} [X_j]^{\nu_j}, & \text{if } i \leq m \\ +k_f \prod\limits_{1\leq j\leq m} [X_j]^{\nu_j} - k_b \prod\limits_{m+1\leq j\leq n} [X_j]^{\nu_j}, & \text{if } m < i \leq n \end{cases} \tag{6}$$

and that at equilibrium

$$\frac{[X_{m+1}]^{\nu_{m+1}} \times \ldots \times [X_n]^{\nu_n}}{[X_1]^{\nu_1} \times \ldots \times [X_m]^{\nu_m}} = \frac{k_f}{k_b}, \tag{7}$$

where $[X]$ denotes the concentration of X.

The simplest model for gene regulation is obtained by using the Michaelis-Menten formalism for complex formation [17], where one assumes that the regulating transcription factor binds to the DNA to form a complex. In this setting, the DNA plays the role of the enzyme in the standard Michaelis-Menten formalism. For an activating transcription factor, we have

$$TF_A + DNA \underset{k_2}{\overset{k_1}{\rightleftharpoons}} TF_A DNA \overset{k_3}{\to} P + TF_A DNA, \tag{8}$$

where $TF_A DNA$ is the complex formed by the activating transcription factor (TF_A) and the DNA, and P is the protein. Here, it is assumed that the binding and releasing of the transcription factor to the DNA is fast compared to the production of the protein. By

[10] "Stoichiometric" refers to the relation between the products and reagents during chemical reactions. The total mass of the involved reagents needs to be conserved through the reaction.

further assuming that there is only one copy of the DNA strand involved, we can use that $DNA + TF_A DNA = 1$. By using Equation (6) and the stated assumptions, one obtains the production rate of the protein (P)

$$\frac{\mathrm{d}P}{\mathrm{d}t} = V_{\max}\frac{[TF_A]}{K + [TF_A]}, \tag{9}$$

where $K = k_2/k_1$ and $V_{\max} = k_3$. This can be interpreted such that the protein production is equal to the fraction of time the transcription factor is bound multiplied by the translation rate. If we instead have the situation that the transcription is active when no transcription factor is bound and inactivated when a repressive transcription factor is bound, the corresponding chemical reaction is

$$\begin{aligned} TF_R + DNA &\underset{k_2}{\overset{k_1}{\rightleftharpoons}} TF_R DNA \\ DNA &\overset{k_3}{\rightarrow} P + DNA, \end{aligned} \tag{10}$$

with protein production rate

$$\frac{\mathrm{d}P}{\mathrm{d}t} = V_{\max}\frac{K}{K + [TF_R]}. \tag{11}$$

Now, the protein production is instead the fraction of time the transcription factor is not bound times the translation rate. One extension that can be made to this model is to allow cooperativity, i.e. that many copies of the transcription factor can bind to the DNA at once. Then, one instead obtains the Hill equations

$$\frac{\mathrm{d}P}{\mathrm{d}t} = V_{\max}\frac{[TF_A]^h}{K^h + [TF_A]^h}, \tag{12}$$

and

$$\frac{\mathrm{d}P}{\mathrm{d}t} = V_{\max}\frac{K^h}{K^h + [TF_R]^h}, \tag{13}$$

where h is the Hill coefficient representing the number of transcription factors needed [18]. When a protein is regulated by two different transcription factors, their contributions must be combined. This can either be done with `AND`-logic, i.e. that both TFs are required, or with `OR`-logic, i.e. that it is enough that one of the TFs is bound. This can be achieved by either multiplying (for `AND`) or adding (for `OR`) the contributions for the two TFs, i.e.

$$\frac{\mathrm{d}P}{\mathrm{d}t} = V_{\max}\frac{[TF_{A,1}]^{h_1}}{K_1^{h_1} \times [TF_{A,1}]^{h_1}} + V_{\max}\frac{[TF_{A,2}]^{h_2}}{K_2^{h_2} + [TF_{A,2}]^{h_2}}, \tag{14}$$

for two activators combined with `AND`-logic and

$$\frac{\mathrm{d}P}{\mathrm{d}t} = V_{\max}\frac{K_1^{h_1}}{K_1^{h_1} + [TF_{R,1}]^{h_1}} + V_{\max}\frac{K_2^{h_2}}{K_2^{h_2} + [TF_{R,2}]^{h_2}}, \tag{15}$$

for two repressors combined with `OR`-logic.

3.1.2 Shea-Ackers formalism

The alternative approach of constructing rate equations is to derive them from principles of statistical physics, which Shea and Ackers were the first to do [19, 20]. In the Shea-Ackers formalism, the partition sum[11] is calculated for all possible states of the involved transcription factors and RNA polymerase. If we once again consider a GRN with N transcription factors, then there will be one partition sum for each ODE in the system. The partition sum for gene X becomes

$$Z(X) = \sum_{\sigma_1 \ldots \sigma_N \sigma_p} p_{RNA}^{\sigma_p} e^{\frac{\Delta G_{\sigma,X}}{k_B T}} \prod_{i=1}^{N} [TF_i]^{\sigma_i}, \tag{16}$$

where p_{RNA} denotes the amount of RNA polymerase, σ_i is a binary number, denoting whether the corresponding transcription factor (or RNA polymerase) is bound or not where

$$\sigma_i = \begin{cases} 1 & \text{if bound} \\ 0 & \text{if not bound,} \end{cases} \tag{17}$$

and $e^{\Delta G_\sigma / RT}$ is the Boltzmann factor where $\Delta G_{\sigma,X}$ is the free energy energy difference, with a suitable zero level, associated with the specific state, k_B is the Boltzmann constant and T is the temperature. The Boltzmann factor could theoretically be known by measurement, but this is mostly not the case in practice. Therefore, the Boltzmann factor can be reduced to an unknown parameter $\alpha_{\sigma,X}$, i.e.

$$Z(X) = \sum_{\sigma_1 \ldots \sigma_N \sigma_p} p_{RNA} \alpha_{\sigma,X} \prod_{i=1}^{N} [TF_i]^{\sigma_i}. \tag{18}$$

The partition sum can be divided into two parts: terms which represent that transcription is `ON` and terms which represent that transcription is `OFF`,

$$Z(X) = Z_{\texttt{ON}}(X) + Z_{\texttt{OFF}}(X). \tag{19}$$

Hence, the transcription rate of gene X is proportional to the probability of the transcriptionally active states

$$p(X) = \alpha \frac{Z_{\texttt{ON}}(X)}{Z_{\texttt{ON}}(X) + Z_{\texttt{OFF}}(X)}, \tag{20}$$

where α is a rate constant. For a simple case of two transcription factors A and B, each of the two transcription factors as well as the RNA polymerase can either be bound to the

[11] The partition sum is a concept from statistical physics where the energy of each possible microstate of a system is summed. The partition sum acts as the normalising factor when calculating the probability for a system being in a specific microstate.

DNA or not, resulting in $2^3 = 8$ possible states. The partition sum for the regulation of A becomes

$$\begin{aligned} Z(A) &= \sum_{\sigma_A \sigma_B \sigma_p} \alpha_{\sigma,A} [A]^{\sigma_A} [B]^{\sigma_B} p_{RNA}^{\sigma_p} \\ &= 1 + \alpha_{p,A} p_{RNA} + \alpha_{A,A}[A] + \alpha_{B,A}[B] + \alpha_{Ap,A}[A] p_{RNA} \\ &\quad + \alpha_{Bp,A}[B] p_{RNA} + \alpha_{AB,A}[A][B] + \alpha_{ABp,A}[A][B] p_{RNA}. \end{aligned} \tag{21}$$

The terms that include A represent self-regulation. Note that the partition sum is not affected by how A and B are connected in a GRN; however, which of the terms that count as activating or inhibiting depends on what roles the transcription factors have in the GRN. The RNA polymerase is often treated as a constant that needs to be formally included; thus, some terms can be combined

$$\alpha_{A,A}[A] + \alpha_{Ap,A}[A] p_{RNA} = (\alpha_{A,A} + \alpha_{Ap,A} p_{RNA})[A] \to \alpha_{A,A}[A], \tag{22}$$

where we redefine $\alpha_{A,A} + \alpha_{Ap,A} p_{RNA} \to \alpha_{A,A}$ in the last step. The same principle can be applied to the terms with $[B]$ and the combined $[A][B]$. The term $\alpha_{Ap,A}[A] p_{RNA}$ can be considered a basal transcription rate. Furthermore, this formalism includes both the `AND`- and `OR`-logic since the partition sum includes both single terms with $[A]$ and $[B]$ as well as their product $[A][B]$. If it is known which one of the logic principles should be applied, the parameters for the unused logic can be set to zero manually. Otherwise, all the terms are kept.

For the general case, many of the terms in the partition sum can be excluded (i.e. setting the corresponding parameter to zero). For instance, if a gene in a five-gene GRN is only regulated by two of the other genes, all the terms containing the non-regulating genes can be disregarded. The complete reaction rate of gene X can be summarised as

$$\frac{\mathrm{d}X}{\mathrm{d}t} = \alpha_X \frac{\beta_{\text{act},X} + \sum_{T \in \text{act}} \alpha_{T,X}[T]}{1 + \beta_{\text{act},X} + \beta_{\text{rep},X} + \sum_{T \in \text{rep}} \alpha_{T,X}[T] + \sum_{T \in \text{act}} \alpha_{T,X}[T]} - \delta_X [X], \tag{23}$$

where T includes single terms of transcription factors as well as all relevant cross terms. This makes the Shea-Ackers formalism very intuitive to use. One simply has to sum all the activators (including cross terms) and place them in the numerator of the production rate function. Both activating and repressing terms are placed in the denominator. The Shea-Ackers formalism could also be extended by allowing for cooperativity by introducing Hill-coefficients [2]. One should also note that the Hill equation is a special case of the Shea-Ackers formalism where only linear terms are included.

When constructing rate equations with the Shea-Ackers formalism, one introduces many unknown parameters. Some of them would be possible to measure, but some have for simplicity been merged and lost their direct biophysical meaning. Therefore, these parameters

need to be found by fitting the rate equations to data. I will describe this process in Section 3.3. In Paper II, we set up rate equations with the Shea-Ackers formalism describing the the transcriptional part of early T-cell development.

3.2 Solving ODEs numerically

"The consequences of our actions are always so complicated, so diverse, that predicting the future is a very difficult business indeed."

— **Albus Dumbledore**, Harry Potter and the Prisoner of Azkaban

When the rate equations have been constructed, we need to solve them as well. The type of problems we are interested in solving is called initial value problems, where an initial condition $\boldsymbol{x}_0$ is given (in our case often from experiments), and we want to evolve the system in time. For a few simple cases, it is possible to solve a set of coupled ODEs analytically, but that is in general not the case. Then, numerical methods must be applied. There are two classes of numerical methods for solving ODEs: deterministic and stochastic methods. A deterministic solution will always yield the same outcome if the same initial conditions are used. On the contrary, a stochastic method will yield different outcomes for the same initial conditions since it includes random elements.

3.2.1 Deterministic solution

The conceptually most simple way to deterministically solve a set of ODEs is with the Euler method [21]. It is the least accurate method, and therefore not very practically useful, but as it illustrates the concept very well, I will describe it shortly. For a general set of coupled ODEs

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \boldsymbol{f}(\boldsymbol{x}, t), \tag{24}$$

with initial conditions $\boldsymbol{x}(t_0) = \boldsymbol{x}_0$, the idea is to evaluate the derivative at time point t_n and then take a small step in time of length h in the direction of the derivative:

$$\begin{aligned} t_{n+1} &= t_n + h, \\ \boldsymbol{x}_{n+1} &= \boldsymbol{x}_n + h\boldsymbol{f}(\boldsymbol{x}_n, t_n) + \mathcal{O}(h^2). \end{aligned} \tag{25}$$

A sketch of this principle is shown in Figure 8a. The Euler method is said to be a 1^{st}-order method since its local truncation error is $\mathcal{O}(h^2)$ and global truncation error $\mathcal{O}(h)$.[12]

[12] A method is conventionally called an m^{th}-order method if its global truncation error is $\mathcal{O}\left(h^m\right)$.

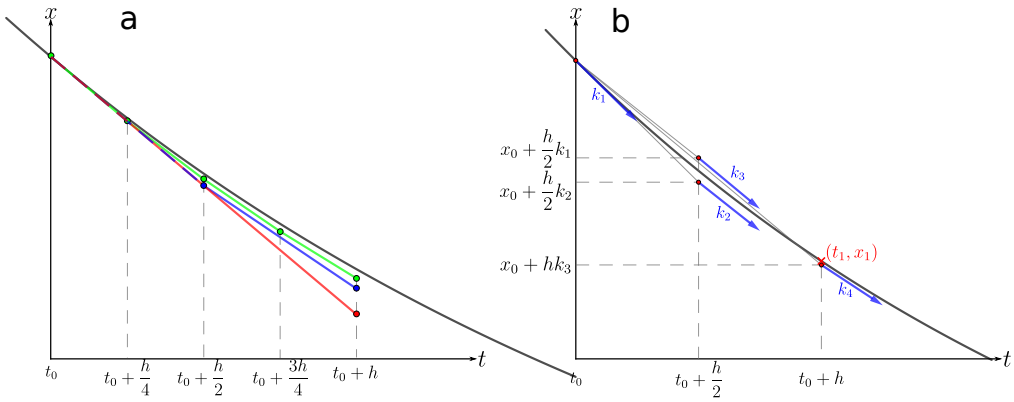


Figure 8: (**a**) Illustration of the Euler method using three different step-sizes where red is step size h, blue step size $h/2$ and green step size $h/4$. (**b**) Schematics of the four trial steps used in the 4^{th}-order Runge-Kutta method. The analytic solution is showed in black in both (**a**) and (**b**).

Here, we only evaluate and use the derivative value at the starting point of the interval. We could improve the accuracy by evaluating more points. The simplest extension is called the Runge-Kutta 2^{nd}-order method, where a first trial step of half the interval is used, upon which the derivative is re-evaluated. The new value is then used to take a full step in time. This principle of trial steps can be extended further where more intermediate steps are used. In the 4^{th}-order Runge-Kutta method, the derivative is evaluated four times: once at the initial point, twice at the interval midpoint, and once at the interval endpoint. These four evaluations are then considered together to execute the final step[13], as sketched in Figure 8b. This process is summarised as

$$\begin{aligned} t_{n+1} &= t_n + h, \\ \boldsymbol{x}_{n+1} &= \boldsymbol{x}_n + \left(\frac{1}{6}\boldsymbol{k}_1 + \frac{1}{3}\boldsymbol{k}_2 + \frac{1}{3}\boldsymbol{k}_3 + \frac{1}{6}\boldsymbol{k}_4\right) + \mathcal{O}(h^5), \end{aligned} \tag{26}$$

where

$$\begin{aligned} \boldsymbol{k}_1 &= h\boldsymbol{f}\left(t_n, \boldsymbol{x}_n\right), \\ \boldsymbol{k}_2 &= h\boldsymbol{f}\left(t_n + \frac{1}{2}h, \boldsymbol{x}_n + \frac{1}{2}\boldsymbol{k}_1\right), \\ \boldsymbol{k}_3 &= h\boldsymbol{f}\left(t_n + \frac{1}{2}h, \boldsymbol{x}_n + \frac{1}{2}\boldsymbol{k}_2\right), \\ \boldsymbol{k}_4 &= h\boldsymbol{f}\left(t_n + h, \boldsymbol{x}_n + \boldsymbol{k}_3\right). \end{aligned} \tag{27}$$

[13]How the trial steps should be taken can be derived by expanding $\boldsymbol{x}(t_n + h)$ as well as the Runge-Kutta update ansatz and balancing the parameters so that the correct terms remain.

When it comes to choosing which method to use, it becomes a question about optimising the accuracy in relation to a method's efficiency and which step size can be used. If the step size is halved for a 2^{nd}-order method, then the global error is improved by a factor of 4. Thus, using a lower-order method requires taking many more steps to perform equally well as a higher-order method. In practice, the 4^{th}-order Runge-Kutta method is sufficiently accurate and efficient for most problems. If needed, it can be extended with an adaptable step size or replaced with a Richardson extrapolation method [21].

3.2.2 Stochastic solution: The Gillespie Algorithm

Since gene transcription is a stochastic process[14], it would make sense to also solve the dynamical rate equations stochastically. This can for example be done with the Gillespie algorithm [22, 23]. In the Gillespie algorithm, the gene expression levels are treated discretely, such that every mRNA transcription of a gene is counted. In contrast to the deterministic solvers described above, the Gillespie algorithm does not have a fixed time step. Instead, the length of the time step is drawn randomly from a distribution. At every time step, one single event happens. For every event occurrence, one transcription count is either added or removed from one of the genes.

To derive the Gillespie algorithm, we consider a set of genes X_i which are produced with production rates $p_i(t)$ and removed with degradation rates $d_i(t)$ as stated in Equation (3). We can collect the production and degradation rates in a_μ so that $a_1 = p_1$, $a_2 = d_1$, $a_3 = p_2$, $a_4 = d_2$ and so forth. The fundamental premise of the algorithm is that event μ will occur in the interval $[t, \mathrm{d}t]$ for a given $\boldsymbol{X}(t)$. For a time step τ, we need to determine the probability $P(\mu, \tau)\mathrm{d}\tau$ that event μ taking place in the time interval $[t+\tau, t+\tau+\mathrm{d}\tau]$. We impose that no events have already taken place in the interval $[t, t+\tau]$. We denote the probability that no events take place before τ with $P_0(\tau)$. Thus we can express the probability that event μ takes place between times $t+\tau$ and $t+\tau+\mathrm{d}\tau$ as

$$P(\mu, \tau)\mathrm{d}\tau = P_0(\tau)a_\mu \mathrm{d}\tau. \tag{28}$$

Now, let us determine $P_0(\tau)$. In order to do so, we use the probability $P_0(\tau + \mathrm{d}\tau)$, i.e. the probability that no events take place in the interval $[t, t+\tau+\mathrm{d}\tau]$,

$$P_0(\tau + \mathrm{d}\tau) = P_0(\tau)\left(1 - \sum_\mu a_\mu \mathrm{d}\tau\right), \tag{29}$$

where we sum over all possible events μ. Hence,

$$\frac{P_0(\tau + \mathrm{d}\tau) - P_0(\tau)}{\mathrm{d}\tau} = -P_0(\tau)\sum_\mu a_\mu, \tag{30}$$

[14] Most biological processes act under natural noise stemming from thermodynamics.

that for $\mathrm{d}\tau \to 0$ becomes the derivative

$$\frac{\mathrm{d}P_0(\tau)}{\mathrm{d}\tau} = -P_0(\tau)\sum_\mu a_\mu, \tag{31}$$

with the solution

$$P_0(\tau) = e^{-a_0\tau}, \tag{32}$$

where we introduce $a_0 = \sum_\mu a_\mu$ and use that no events can have taken place at time zero, i.e. $P_0(\tau = 0) = 1$. Inserting Equation (32) into Equation (28) yields

$$P(\mu, \tau)\mathrm{d}\tau = a_\mu e^{-a_0\tau}\mathrm{d}\tau. \tag{33}$$

This is a composite probability distribution of two different probabilities:

$$P(\mu, \tau)\mathrm{d}\tau = P_1(\tau) \cdot P_2(\mu)\mathrm{d}\tau, \tag{34}$$

where $P_1(\tau)$ is the probability for an event taking place in the time interval $[t+\tau, t+\tau+\mathrm{d}\tau]$ and $P_2(\mu)$ is the probability that event μ happens. To find P_1, we sum over all possible events

$$P_1(\tau) = \sum_\mu P(\mu, \tau) = \sum_\mu a_\mu e^{-a_0\tau} = a_0 e^{-a_0\tau}. \tag{35}$$

To find P_2, we instead integrate with respect to time from 0 to ∞

$$P_2(\mu)\mathrm{d}\tau = \int_0^\infty P(\mu, \tau)\mathrm{d}\tau = \int_0^\infty a_\mu e^{-a_0\tau}\mathrm{d}\tau = \frac{a_\mu}{a_0}. \tag{36}$$

To sample a time-step from $P_1(\tau)$, we use the cumulative distribution of P_1 and the fact that P_1 is normalised, which is commonly known as inverse sampling. Thus,

$$r_1 = \int_0^\tau P_1(t)\mathrm{d}t = a_0 \int_0^\tau e^{-a_0 t}\mathrm{d}t = 1 - e^{-a_0\tau}, \tag{37}$$

where r_1 is a uniformly distributed number in the interval $[0, 1)$. Hence, the time-step τ can be found in terms of r_1 as

$$\tau = \frac{1}{a_0}\ln\frac{1}{r_1}, \tag{38}$$

where we note that $1 - r_1$ is also a uniformly distributed number in the interval $[0, 1)$, thus relabeling $r_1 = 1 - r_1$. The event that is taking place is determined by

$$\mu = \text{ smallest integer } \mu \text{ for which } \sum_{k=1}^{\mu} a_k > r_2 a_0, \tag{39}$$

where $r_2 \in [0, 1)$, which is called discrete inverse sampling and is illustrated schematically in Figure 9.

With the knowledge of how to sample an event type and time, the Gillespie algorithm can be summarised:

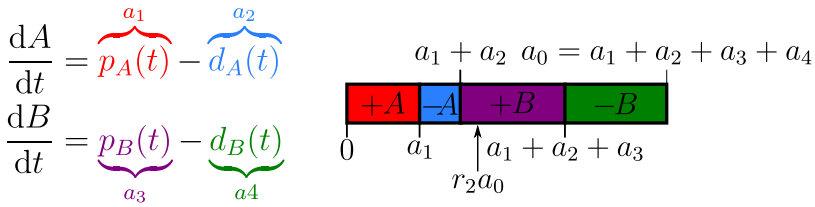


Figure 9: Sketch of how the event is chosen in the Gillespie algorithm. For a system with two genes A and B, their respective production and degradation rates are evaluated at time t, and a_0 can then be calculated. A uniformly distributed random number r_2 in the interval $[0, 1)$ is sampled to determine which event takes place. In this example $a_1 + a_2 < r_2 a_0 < a_1 + a_2 + a_3$, meaning that event $\mu = 3$ takes place, i.e. one transcript of gene B is added.

0. Initialise $t = t_0$ and $\boldsymbol{X} = \boldsymbol{X}_0$.
1. Evaluate all a_μ and a_0 given the system in state $\boldsymbol{X}$ at time t.
2. Generate τ and μ by using Equations (38) and (39).
3. Update the system $t = t + \tau$ and $\boldsymbol{X} = \boldsymbol{X} + \boldsymbol{x}_\mu$, where $\boldsymbol{x}_\mu$ adds or removes one unit of the correct species.
4. Record $\boldsymbol{X}$ and t. Repeat from step 1 unless the end condition is fulfilled. The end condition is usually to perform a fixed number of steps or until a maximum time is reached.

3.3 Parameter optimisation

"An expert is a person who has made all the mistakes that can be made in a very narrow field."

— Niels Bohr

In order to solve a set of rate equations numerically, all the parameters of the equations must be determined. These are parameters such as production and degradation rate parameters as those present in Equation (23). Without knowing these parameters, the system of ODEs has an infinite number of solutions and it is not possible to know which ones may correspond to something biologically relevant. Therefore, experimental data is needed to tune the parameters of the model so that the model output fits known biological behaviour. The model can be used to predict outcomes of new proposed experiments only after the parameters have been tuned.

In order to tune the parameters, an error function is needed [21]. Such a function compares the given data with the model outcome for the same time points for a given set of parameters.

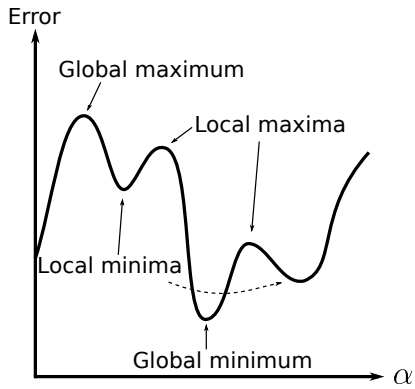


Figure 10: Sketch of an error function depending on one parameter α with global- and local minima and maxima marked.

The error function, however, needs to be dressed with some specific characteristics to be useful. The error function value should become smaller the closer the model output is to the data, with the smallest possible value obtained if the model output and data are exactly the same.[15]

Consider a set of m parameters which we can collect in a vector

$$\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_m) \tag{40}$$

for a set of ODEs $\boldsymbol{f}$ that we want to optimise. Then we can construct an error function $E(\boldsymbol{f}_n(\boldsymbol{\alpha}), \boldsymbol{d}_n)$ which considers the given experimental data $\boldsymbol{d}$ for n time points and the corresponding model values for the same time points. Two common choices for the error function is the mean squared error (MSE)

$$E(\boldsymbol{\alpha}) = \frac{1}{n} \sum_{k=1}^{m} \sum_{j=1}^{n} (f_{k,j}(\boldsymbol{\alpha}) - d_{k,j}) \tag{41}$$

and the log-likelihood (LL) function

$$L(\boldsymbol{d}|\boldsymbol{\alpha}) = \ln \left(\prod_{k=1}^{m} \prod_{j=1}^{n} \frac{1}{\sqrt{2\pi\sigma_{kj}^2}} \exp \left(-\frac{1}{2\sigma_{kj}^2} \left(d_{kj} - f_{kj}(\alpha) \right)^2 \right) \right). \tag{42}$$

These functions can be adapted to include known biological constraints by various fit to data.

[15] That closer agreement should yield smaller value is just a convention. It could just as well be that better-fit yields the greater value. The important quality is that the error function has an extremum value at the point of best fit.

Finding the minimum value of a function, *optimisation*, is not a trivial task. Typically, the global minimum of the function is desired, but there are often local minima which makes the search more difficult, as sketched in Figure 10. There are several algorithms that are very good at finding local minima based on calculating the gradient of the error function and using it in clever ways. The simplest version is to take steps in the direction of steepest gradient from a starting point, as first suggested by Cauchy [24]. Another, more refined version where the second derivatives in form of the Hessian matrix are approximated is the L-BFGS-B algorithm [25]. This algorithm is commonly used for optimising problems with many variables since it is implemented in a memory-saving fashion. By initiating many instances of a local optimisation algorithm at well-distributed starting points, one increases the chances of finding the global minimum, but it is not guaranteed to be found.

Another class of optimisation methods is non-gradient based methods which all employ various strategies to avoid getting stuck in local minima. Three of these methods are simulated annealing [26], genetic algorithms [27] and particle swarm optimisation [28]. Simulated annealing is based on thermodynamics and is stochastically probing the error function to find lower and lower values. A temperature parameter controls the probability to keep worse errors, i.e. to go uphill again, which could potentially help the algorithm to escape local minima. Genetic algorithms are based on evolutionary ideas on "survival of the fittest" where many points in the parameter space are sampled. These points are combined and mutated randomly within a scheme and the points with lowest error are kept. This is repeated until the population converges or a fixed number of generations has been reached. Particle swarm optimisation is based on swarm behaviour where a swarm of points in the parameter space are moving around. The individual points' velocities are affected both by each point's historically best tried position as well as the swarm's currently best global position. The swam is updated until all points converge or a maximum number of iterations is reached.

All of these methods, local as well as global, take steps in the parameter space in one way or another, meaning that the function $\boldsymbol{f}(\boldsymbol{\alpha})$ must be evaluated at every step. Evaluation of $\boldsymbol{f}(\boldsymbol{\alpha})$ includes solving the system of ODEs deterministically as described in Section 3.2.1. The chosen numerical solver must be efficient since the system has to be solved many times. When the optimisation algorithm has been run, one typically uses the found parameter set which yields the minimum value of the error function. However, it is important to check that the behaviour of the best solution seems biologically realistic, which is not always the case.

Two practical notes on good practice for parameter optimisation are to be made. One is to not initialise the starting points which are randomly distributed. Secondly, one should not use linear search spaces for the parameter values [29]. Perhaps counterintuitively, uniform, randomly distributed starting points do not yield the most evenly distributed points. Instead, an alternative method is to use latin hypercube sampling (LHS) [30] where the search

space is divided into small m-dimensional hypercubes and one set of parameters is drawn from each hypercube. Most of the parameters represent transcription or degradation rates and are therefore constrained biologically. As an example, assume that we know that the transcription controlled by a parameter α is taking place, but we do not know if it is one transcription every few hours or ten per hour. Then we can approximate a constraint for α to be in the interval $[10^{-4}, 10]$. However, if we search this interval uniformly, we will put much more weight on values between 1 and 10, than between 10^{-4} and 1. To instead give each order of magnitude the same weight when searching, one can instead search for a dummy parameter β in the uniform interval $[-4, 1]$ and use $\alpha = 10^{\beta}$, i.e. using a logarithmic search interval. In paper II, I used these two techniques together with the L-BFGS-B algorithm [25] to optimise the parameters of the rate equations for our dynamical model.

3.4 Machine learning

"The ability to speak does not make you intelligent."

— **Qui-Gon Jinn**, Star Wars: Episode I – The Phantom Menace

The words "Machine learning" may lead you to think that computers can learn the same way as humans do, but is that the case? Well, machine learning is a tool in which computer models learn in similar ways as humans do at least. Instead of directly programming handcrafted features of a model as is the conventional way, one lets the model learn from data – *training data*. A major use case of machine learning [16] models is in classification problems where objects belong to different categories and the model should learn how to distinguish between the categories. In this way, a machine learning model just needs to be presented with data points with corresponding labels. This could for instance be an image or a set of measured medical test values together with a corresponding label such as the identity of the object in the image or a medical diagnosis. The specific machine learning model I use in Paper IV is called an Artificial Neural Network (ANN) [31]. The name may sound like an attempt to simulate the human brain's function and ability to think, but that is not the case. ANNs are only using the same principle of propagating information. They are built up by nodes which receive input signals, combine the input in some way, and propagate the signal to other nodes, similar to how the neurons in the brain propagate information, and does thus comprise a network of artificial neurons.

A sketch of a basic artificial neuron, also called a *perceptron*, is shown in Figure 11a. The input signals x_k where $k = 1, \ldots, p$ are multiplied by weights ω_k and added before the

[16] Here, I make the restriction to supervised machine learning since the task we are solving in Paper IV is a classification problem .

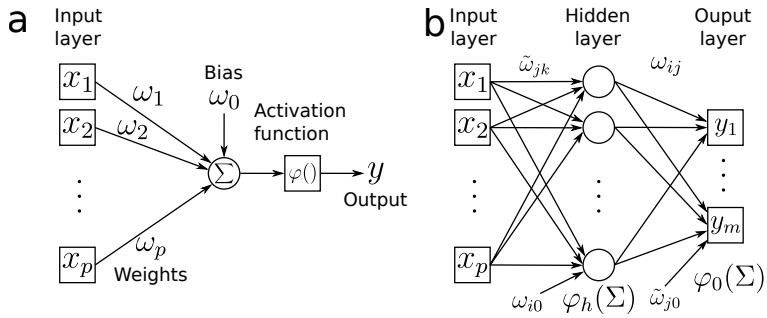


Figure 11: (**a**) Sketch of perceptron where the input signals are weighted, summed and put through an activation function to produce an output signal. (**b**). Sketch of a multi layer perceptron with one hidden layer and multiple output nodes.

sum is put through an activation function φ which produces the output value

$$y = \varphi\left(\omega_0 + \sum_{k=1}^{p} \omega_k x_k\right), \tag{43}$$

where ω_0 is a bias. The role of the weights is to indicate how important a feature is for the output. For instance, if one wants to determine the height of a person, then the input corresponding to the person's sex will probably have a higher weight and be more important than the input corresponding to the person's hair colour.

By connecting many perceptrons into several layers, one can create a *multilayer perceptron* (MLP) which has the potential to find more complex and abstract patterns in data, as illustrated in Figure 11b. The layers between the input and output layers are often referred to as hidden layers. It is also possible to introduce several output nodes, which for instance is useful when performing multiclass classification tasks with m classes. The output y_i for an MLP with one hidden layer with q hidden nodes can be formulated as

$$y_i(\boldsymbol{x}_n) = \varphi_0\left(\sum_{j=0}^{q} \omega_{ij}\varphi_h\left(\sum_{k=0}^{p} \tilde{\omega}_{jk} x_{nk}\right)\right), \tag{44}$$

where we distinguish between the output activation function φ_0 and the activation function of the hidden layer φ_h. This formalism can be extended for an arbitrary number of hidden layers recursively by replacing x_{nk} with a new sum over the nodes in the next hidden layer.

The activation functions φ have an important role in introducing non-linearity. If only linear activation functions are used, it can be shown that all the hidden layers can be reduced to one layer. Non-linearity also introduces flexibility to the model to create complex decision boundaries to separate the classes. The activation function often gives larger output value

for increased input values and has a threshold property at some critical level. Commonly used activation functions include the logistic function, hyperbolic tangent, rectified linear unit (ReLU) and softmax which all have their different cases of use. The output activation function is also problem-specific, and different functions are used for binary classification problems, multiclass classification problems and regression problems. For binary classification problems, the logistic function is often used which produces output between 0 and 1 with a sharp transition, a *threshold*, at 0.5. This output value can be interpreted as the probability for a data point to belong to one of the two classes.

When training an ANN, an error function is optimised, similar to that described in Section 3.3. The error function of choice depends on the type of problem at hand. The training process is typically done iteratively where the weights are updated in such a way so that the training error is reduced over time. Various methods can be employed to avoid getting stuck in the local minima of the error function. A common problem when training ANNs is overfitting. Then, the network's decision boundary gets too specifically adapted to the training data and loses its capability of generalising. A separate held-out validation data set is typically used to monitor the model's performance on data not used during training. For an overtrained model, the validation error becomes high while the training error remains low. There are ways to combat overtraining with various regularisation techniques [31].

The procedure of training an ANN model introduces many hyperparameters which become part of the full model. The model architecture itself introduces hyperparameters such as the number of hidden layers and the number of hidden nodes per layer. There are training-specific hyperparameters like the learning rate, training time and batch size. If one uses regularisation to combat overtraining one also introduces regularisation parameters. Unfortunately, it is not enough to find the best value for each hyperparameter individually; the parameters affect each other. Therefore, one needs to use a model selection strategy where multiple hyperparameter settings are tested. Two common approaches for this are to either perform a grid search where all combinations of a set of chosen hyperparameter values are tested, or to perform a random search where random combinations of hyperparameter values are tested. Typically one uses the validation performance to pick the final model to use.

A fully trained machine learning model can make predictions of data it has never seen before. For instance, a model trained on pictures of cats and dogs can be fed a completely new image you just took of your dog, and the network will recognise it as a dog. However, an ANN can never know anything about something it has never seen before. A machine learning model is a way to find generalised features from a large collection of data, but if it is presented with something completely new, it has no way of treating it properly. For instance, our just-mentioned cats and dogs network would not be able to recognise an image of a horse. It would not even know that it is not an image of a cat or dog it is processing, since it knows of no other kinds of animals. Hence, the best case scenario would be that it

assigns 50 % probability for the horse both being a cat and a dog, but the model will most likely be (incorrectly) quite certain that it is one of them.

Deep learning models is a subclass of artificial neural networks. Deep learning is not just stacking an MLP with many hidden layers. Such a network does not often outperform MLPs with just a few hidden layers. Instead, deep learning refers to multi-layer networks which have some smart variations of the architecture. One common deep learning model is the convolutional neural network (CNN) [31]. These networks have been very successful when it comes to detecting and classifying objects in images. A CNN utilises the mathematical convolution operator which in practice means that a small kernel slides over the image, processing a local neighbourhood at a time, and then connects all the information. This operation can pick up spatial features in an image and is to some extent translationally invariant, i.e. it does not matter where in an image the object is located. The same principle of using a CNN to process data with neighbouring features can also be used for one dimension input data.

In Paper IV, I use both an MLP and a one-dimensional CNN to analyse hyperspectral data. A hyperspectrum consists of a signal with absorbance values for neighbouring wavelengths. When using an MLP, all the wavelength channels are used as input and are treated individually. By using a one-dimensional CNN, potentially useful information contained in local spectral neighbourhoods becomes available.

3.5 Segmentation

"Do or do not, there is no try."

— **Yoda**, Star Wars: Episode V – The Empire Strikes Back

For humans, there is no difficulty finding an object in an image by simply looking at it. But it is not as easy for a computer. A computer needs some algorithm which defines what an object is and how to find it. For a simple case, it might just be a matter of finding a threshold in a grey-scale image that separates the intense object from the less intense background, as illustrated in Figure 12a. The example shows a greyscale microscopy image of a few cells. In order to separate the cells from the background and count the number of cells, a threshold is applied where all pixels with intensity above the threshold are coloured black and all pixels below the threshold are coloured white. For this simple case, the cells get clearly identified.

However, for more complex cases, a threshold might not be enough. If the microscopy image would contain more and overlapping cells, the threshold would not have worked as well. If the image is of lower quality, there would be noise or other artefacts that need to be

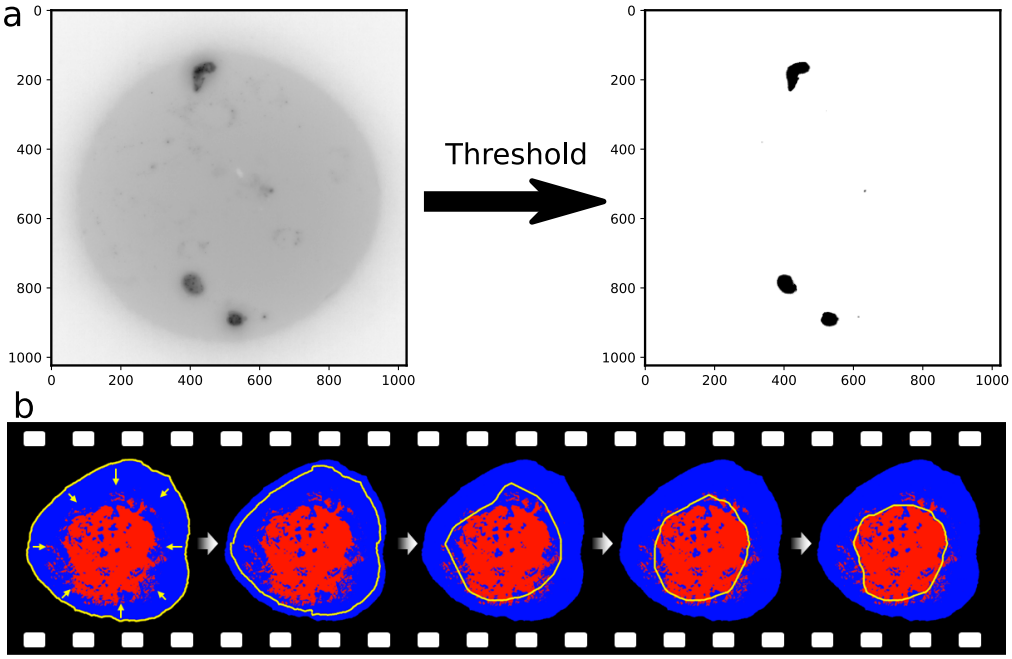


Figure 12: (**a**) Example of segmentation by thresholding. All pixles above the pixel intensity threshold are coloured black and all pixels below the threshold are coloured white. Image credit Rothenberg lab, Dr. Mary Yui. (**b**) An active contour algorithm is used to find the contour of a tumour. The active contour is initialised along the edge of the sample and is tightened so that it passes over noise but eventually outlines the tumour.

ignored. Then, other, more complex and adaptable algorithms are needed. Other instances where more refined segmentation could be needed are for instance if the image has multiple colour channels (e.g. an RGB image), or if high-intensity pixels should be included in the objects sometimes, but not other times, depending on the pixels' spatial context.

In Paper IV, I use a segmentation algorithm to find the borders of skin tumours in images I have preprocessed with machine learning. Here, applying a simple threshold is not enough since the images contain noise and the goal is to outline the tumour. Therefore, I applied an active contour algorithm [32]. The principle of active contour segmentation is based on a classical rope. Such a rope is tightened around the image object by minimising an energy function until the object border is found, as exemplified in Figure 12b. This makes the algorithm very versatile since it can be adapted depending on the specific use-case. The algorithm is described in detail in Paper IV.

4 A segway to the papers and conclusion

"So perhaps the best thing to do is to stop writing introductions and get on with the book."

— Winnie the Pooh

Cells are marvellous biological units of life. Not only do they comprise an incredible complex by themselves, but they also collectively build up even more complex structures such as the human body. A countless number of processes are taking place constantly within a cell, not the least the gene regulation process which determines and maintains a cell's identity. Cell type governing GRNs are the objects of study in Paper I. We use the binary formalism for gene expression to construct energy landscapes which we mainly use to study cell reprogramming by stochastic simulations. In this way, we can obtain information about the existing cell states and the risk of getting stuck in an undesired state during the reprogramming process. This is a general method which can be applied to any cell system governed by a GRN.

The gene transcription is not only regulated transcriptionally by transcription factors, but also by epigenetic mechanisms. In Paper II, we develop a multi-scale model which includes both a level of transcriptional regulation and a level of epigenetic regulation. This model is applied to studying early T-cell development, where proliferating cell colonies are simulated. The T-cell development has been studied extensively which makes it an important model system, meaning that findings about the development mechanisms in this system carry relevance for other systems as well. The computational methods used in Paper II encompass most of the methods described in Section 3. We model the gene regulation dynamically and construct the rate equations with the Shea-Ackers formalism. The parameters are optimised with data and during this process, the rate equations are solved deterministically. Once the model parameters have been set, the full multi-scale model simulations are carried out stochastically using the Gillespie algorithm. The same model base is used in Paper III, but then we further develop the model by placing it in an agent-based framework [33]. By doing so, we can construct lineage trees of the simulated T-cell colonies and we find that inheritance plays an important role in the mechanism when the progenitor T-cells commit to the T-cell fate lineage.

In Paper IV, we encounter a classification problem when we want to find the borders of superficial skin tumours from hyperspectral images. Thus, we employ a machine learning model where we use the hyperspectral signal in individual pixels as input. The model is trained using pixels from small regions of the healthy tissue and tumour respectively. We compare the use of an ANN and a one-dimensional CNN, where the CNN acts on the spectral signal. A fully trained network is used to predict the identity of each pixel in a network which creates a prediction map. A segmentation algorithm is employed on the prediction map to find the tumour contour.

Computational modelling brings an interesting aspect to biological research since it provides a means of analysing systems which are hard to explore experimentally. The predictive power of a well-constructed model can provide valuable information about how to improve experimental procedures. In this thesis, I have used several computational methods and devised a couple of new models to explore a small corner of biology. The scientific field of computational biology and biological physics is full of opportunities since there is an uncountable number of systems yet to be explored and an uncountable number of new models and tools to develop for achieving this.

References

"When in doubt, go to the library."

— **Ron Weasley**, Harry Potter and the Chamber of Secrets

[1] B. Alberts, A. Johnson, J. Lewis, *et al. Molecular Biology of the Cell – 5th ed.* Garland Science, Taylor & Francis Group (2008).

[2] R. Phillips, J. Kondev, J. Theriot, *et al. Physical Biology of the Cell*. Garland Science (2012).

[3] R. Milo & R. Phillips. *Cell Biology by the Numbers*. Garland Science (2015).

[4] E. Gibney & C. Nolan. "Epigenetics and gene expression". *Heredity*, **105** (1) (2010), 4–13. doi:10.1038/hdy.2010.54.

[5] J. C. Kiefer. "Epigenetics in development". *Developmental Dynamics: an Official Publication of the American Association of Anatomists*, **236** (4) (2007), 1144–1156. doi:10.1002/dvdy.21094.

[6] No authors listed. "In pursuit of systems". *Nature*, **435** (1). doi:10.1038/435001a.

[7] C. Wanjek. "Systems biology as defined by NIH: an intellectual resource for integrative biology". *The NIH Catalyst*, **19** (6) (2011), 1–12.

[8] S. B.-T. de Leon & E. H. Davidson. "Gene regulation: gene control network in development". *Annu. Rev. Biophys. Biomol. Struct.*, **36** (2007), 191–212. doi:10.1146/annurev.biophys.35.040405.102002.

[9] V. Chickarmane, V. Olariu, & C. Peterson. "Probing the role of stochasticity in a model of the embryonic stem cell–heterogeneous gene expression and reprogramming efficiency". *BMC Systems Biology*, **6** (2012), 1–12. doi:10.1186/1752-0509-6-98.

[10] C. Waddington. *The Strategy of the Genes*. George Allen & Unwin (1957).

[11] R. L. Davis, H. Weintraub, & A. B. Lassar. "Expression of a single transfected cDNA converts fibroblasts to myoblasts". *Cell*, **51** (6) (1987), 987–1000. doi:10.1016/0092-8674(87)90585-X.

[12] K. Takahashi & S. Yamanaka. "Induction of pluripotent stem cells from mouse embryonic and adult fibroblast cultures by defined factors". *Cell*, **126** (4) (2006), 663–676. doi:10.1016/j.cell.2006.07.024.

[13] K. Takahashi, K. Tanabe, M. Ohnuki, *et al.* "Induction of pluripotent stem cells from adult human fibroblasts by defined factors". *Cell*, **131** (5) (2007), 861–872. doi:10.1016/j.cell.2007.11.019.

[14] K. Takahashi. "Cellular reprogramming". *Cold Spring Harbor Perspectives in Biology*, **6** (2) (2014), a018 606. doi:10.1101/cshperspect.a018606.

[15] V. Olariu & C. Peterson. "Kinetic models of hematopoietic differentiation". *Wiley Interdisciplinary Reviews: Systems Biology and Medicine*, **11** (1) (2019), e1424. doi:10.1002/wsbm.1424.

[16] C. Guldberg & P. Waage. "Studier i affiniteten (translation: Studies on affinities)". *Videnskabs-Selskabet i Christiana*, **35**.

[17] L. Michaelis, M. L. Menten, *et al.* "Die kinetik der invertinwirkung". *Biochem. z*, **49** (333-369) (1913), 352.

[18] A. V. Hill. "The possible effects of the aggregation of the molecules of hemoglobin on its dissociation curves". *J. Physiol.*, **40** (1910), iv–vii.

[19] G. K. Ackers, A. D. Johnson, & M. A. Shea. "Quantitative model for gene regulation by lambda phage repressor". *Proceedings of the National Academy of Sciences*, **79** (4) (1982), 1129–1133. doi:10.1073/pnas.79.4.112.

[20] M. A. Shea & G. K. Ackers. "The OR control system of bacteriophage lambda: A physical-chemical model for gene regulation". *Journal of Molecular Biology*, **181** (2) (1985), 211–230. doi:10.1016/0022-2836(85)90086-5.

[21] W. H. Press. *Numerical recipes 3rd edition: The Art of Scientific Computing*. Cambridge university press (2007).

[22] D. T. Gillespie. "A general method for numerically simulating the stochastic time evolution of coupled chemical reactions". *Journal of Computational Physics*, **22** (4) (1976), 403–434. doi:10.1016/0021-9991(76)90041-3.

[23] D. T. Gillespie. "Exact stochastic simulation of coupled chemical reactions". *The Journal of Physical Chemistry*, **81** (25) (1977), 2340–2361.

[24] A. Cauchy *et al.* "Méthode générale pour la résolution des systemes d'équations simultanées". *Comp. Rend. Sci. Paris*, **25** (1847) (1847), 536–538.

[25] R. H. Byrd, P. Lu, J. Nocedal, *et al.* "A limited memory algorithm for bound constrained optimization". *SIAM Journal on Scientific Computing*, **16** (5) (1995), 1190–1208. doi:10.1137/0916069.

[26] S. Kirkpatrick, C. D. Gelatt Jr, & M. P. Vecchi. "Optimization by simulated annealing". *Science*, **220** (4598) (1983), 671–680. doi:10.1126/science.220.4598.67.

[27] J. H. Holland. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press (1992).

[28] J. Kennedy & R. Eberhart. "Particle swarm optimization". In "Proceedings of ICNN'95-international conference on neural networks", volume 4, pages 1942–1948. IEEE (1995). doi:10.1109/ICNN.1995.488968.

[29] A. Raue, M. Schilling, J. Bachmann, *et al.* "Lessons learned from quantitative dynamical modeling in systems biology". *PLOS ONE*, **8** (9) (2013), e74 335. doi:10.1371/journal.pone.0074335.

[30] M. D. McKay, R. J. Beckman, & W. J. Conover. "A comparison of three methods for selecting values of input variables in the analysis of output from a computer code". *Technometrics*, **42** (1) (2000), 55–61. doi:10.1080/00401706.2000.10485979.

[31] I. Goodfellow, Y. Bengio, & A. Courville. *Deep Learning*. MIT Press (2016). `http://www.deeplearningbook.org`.

[32] M. Kass, A. Witkin, & D. Terzopoulos. "Snakes: Active contour models". *International Journal of Computer Vision*, **1** (4) (1988), 321–331. doi:10.1007/BF00133570.

[33] J. H. Holland & J. H. Miller. "Artificial adaptive agents in economic theory". *The American Economic Review*, **81** (2) (1991), 365–370.

Overview of publications

Below follows a short description of each of the publications included in this thesis and my contribution to each one.

Paper I

CELLoGeNe – an energy landscape framework for logical networks controlling cell decisions

E. Andersson, M. Sjö, K. Kaji, and V. Olariu
iScience **25** (8), 104743 (2022)
doi: `j.isci.2022.104743`

In this paper, we develop CELLoGeNe (Computation of Energy Landscapes of Logical Gene Networks), a software tool which maps Boolean implementation of gene regulatory networks (GRNs) into energy landscapes. To consider cell commitment and reprogramming as movements in an energy landscape is a powerful computational method to elucidate mechanisms controlling cell fate decisions during development and reprogramming. By separating a gene's expression from its effect on its target genes, we introduce a three-state logic which removes inadvertent symmetries in the energy landscapes normally arising from standard Boolean operators. We also develope a method for visualising multi-dimensional energy landscapes in more than three dimensions. Furthermore, with CELLoGeNe, we provide a tool for stochastically analysing the shape of the energy landscape by simulating cell reprogramming in the form of weighted random walks in the landscape. Finally, we demonstrate CELLoGeNe on two GRNs governing different aspects of induced pluripotent stem cells, identifying experimentally validated attractors and revealing potential reprogramming roadblocks. CELLoGeNe is a general framework that can be applied to various biological systems, offering a broad picture of intracellular dynamics otherwise inaccessible with existing methods.

This project originated as the bachelor project of Mattias, where he and Victor conceived the original idea behind the model. Mattias implemented a first version of CELLoGeNe, while I refined and further developed it. I came up with and implemented the stochastic analysis of the landscapes, the *marble simulations*. I also conducted the biological applications and computational analysis. All three of us designed the study and I wrote the manuscript together with Victor and Mattias.

Paper II

Multi-scale dynamical modeling of T cell development from an early thymic progenitor state to lineage commitment

V. Olariu, M. A. Yui, P. Krupinski, W. Zhou, J. Deichmann, **E. Andersson**, E. V. Rothenberg, and C. Peterson
Cell Reports **34** (2), 108622 (2021)
doi: `j.celrep.2020.108622`

In this study, we investigate the development of multipotent haematopoietic precursor cells into committed T-cell progenitors, both experimentally and theoretically. This transition encompasses programmed shutoff of stem/progenitor genes, upregulation of T-cell specification genes, proliferation, and ultimately commitment. To explain these features in light of reported chromatin effects and experimental kinetic data, we develop a three-level dynamic model of commitment based on regulation of the commitment-linked gene Bcl11b. The levels are (1) a core GRN architecture from transcription factor perturbation data, (2) a stochastically controlled chromatin-state gate, and (3) a single-cell proliferation model validated by experimental clonal growth and commitment kinetic assays. After tuning the model to our experimental data, the model predicts state-switching kinetics validated by measured clonal proliferation and commitment times. The resulting multi-scale model provides a mechanistic framework for dissecting commitment dynamics.

My main contribution to this project was conducting parameter optimisation for level 1 of the model as well as performing confidence bounds calculations. I also read and provided feedback on the manuscript.

Paper III

T-cell commitment inheritance – an agent-based multi-scale model

E. Andersson, E. V. Rothenberg, C. Peterson, and V. Olariu
Submitted to *Communications Biology*
doi: `bioRxiv: 10.1101/2023.10.18.562905`

Paper III is a follow-up study to paper II. In this project, we extend our multi-scale model and put it in an agent-based framework. This model provides a powerful tool as it enables us to keep track of each individual cell and obtain full knowledge of a simulated colony's lineage tree. This allows us to investigate T-cell commitment inheritance and dissect the commitment decision mechanism. We do so by introducing the concept of the last common ancestors (LCA) of committed cells and analysing their relations, at both single-cell level and population level. We also conduct knockdown analysis as a part of elucidating the commitment mechanism.

I performed all the model implementations and came up with the usage of lineage tree and LCA analysis. The study was designed by all authors together. I performed the analysis in discussion with Victor and Carsten. Ellen provided key biological insights and perspectives. I wrote the manuscript together with Victor, and Carsten and Ellen provided feedback.

Paper IV

Facilitating clinically relevant skin tumor diagnostics with spectroscopy-driven machine learning

E. Andersson, J. Hult, C. Troein, M. Stridh, B. Sjögren, A. Pekar-Lukacs, J. Hernandez-Palacios, P. Edén, B. Persson, V. Olariu, M. Malmsjö, and A. Merdasa
Submitted to *Cell Reports Medicine*
doi: `medRxiv:10.1101/2023.10.14.23296584`

Skin cancer is the third most common kind of cancer, and the need for efficient removal is paramount. When a tumour is excised, it is important that no tumourous tissue is left. At the same time, the margins must not be overly large since this can lead to excessive scarring. In this study, we develop a machine learning tool which automatically detects the tumour borders. We use data from hyperspectral images, training only on spectra from small regions representing either healthy tissue or tumour. By first training artificial neural networks, we use these to generate prediction maps of which pixels represent healty tissue or tumour. Thereafter, a segmentation algorithm determines the skin tumour borders. Our approach therefore circumvents the need for a complete ground truth image. All the training data is contained within images from each individual patient, which links to an important strength of our approach as we develop individual network models for each patient. This makes our approach relevant also for emerging precision skin tumour diagnostics where adaptability toward the individual is key. This tool could provide useful aid to surgeons excising tumours.

I, together with Victor, Aboma and Malin, conceptualised and designed the study and we also wrote the manuscript. I developed the machine learning and segmentation model, implemented, and performed all the analysis under supervision and discussion with Victor, Carl, Patrik and Aboma.