PONTIFICIA UNIVERSIDAD CATOLICA DE CHILE

SCHOOL OF ENGINEERING

DEVELOPMENT OF RULE-BASED MODELS FOR REGULATORY MECHANISMS OF GENE EXPRESSION AND BACTERIAL METABOLISM

RODRIGO ALBERTO SANTIBÁÑEZ PALOMINOS

Thesis submitted to the Office of Graduate Studies in partial fulfillment of the requirements for the Degree of Doctor in Engineering Sciences

Advisor:

DANIEL GARRIDO CORTES

Santiago de Chile, May, 2020

PONTIFICIA UNIVERSIDAD CATOLICA DE CHILE

SCHOOL OF ENGINEERING

DEVELOPMENT OF RULE-BASED MODELS FOR REGULATORY MECHANISMS OF GENE EXPRESSION AND BACTERIAL METABOLISM

RODRIGO ALBERTO SANTIBÁÑEZ PALOMINOS

Members of the Committee:

DANIEL GARRIDO

TIMOTHY RUDGE

MARÍA RODRÍGUEZ

ÁLVARO OLIVERA

ERNESTO PEREZ-RUEDA

GUSTAVO LAGOS

Thesis submitted to the Office of Graduate Studies in partial fulfillment of the requirements for the Degree of Doctor in Engineering Sciences

Santiago de Chile, May, 2020

"Without imperfection, you or I would not exist."
— Sir Stephen Hawking,
***Into the Universe with Stephen Hawking*,**
The Discovery Channel

To my goddaughter.

ACKNOWLEDGEMENTS

I am deeply grateful to Prof. Daniel Garrido for his incredible guidance and advice through the developmental process involved in this thesis. I cannot express enough with words the help received from him during the years. Similarly, I am equally thankful to Prof. Alberto J.M. Martín and Dr. Tomás Perez-Acle for their support, and along with Prof. Daniel Garrido, I thank them for their continuous support at a professional and personal level.

Many persons contributed directly and indirectly to the development of this thesis. I would express my gratitude to my family, friends, past and current coworkers, anonym reviewers, and the members of the committee for their contribution to the many aspects the doctoral process had. Special thanks to Dr. Javiera López, Dr. Paulina Torres, and Paulina Araya for being there even in moments I did not realize I need them. In addition, I thank Paola Rozas and Andrés Alarcón for choosing me as the godfather of their daughter Amparo and of course, thanks to my mother Ángela, my father Abraham, and my sister Romina.

I also thank to Francisco Pinto, Verónica Ortuzar, and Natalia Gutiérrez, former members of the System Microbiology Lab and their current members Romina Diaz, Kevin González, Marco Vega, Loreto Román, Belén Hirmas, Aline Ovalle, María José Barra, Kineret Severinsky, Guillermo Orellana, and Javiera Munizaga; to Eduardo "Lalo" Martinez, Sebastián Contreras, Leandro Murgas, Verónica Latapiat, Pablo Berrios, Javier Villacreces, and Sandra Sepúlveda, current members of the Network Biology Lab; to Prof. Mauricio Saez, Prof. Annette Trombert, Dr. Carolina Sánchez, and Prof. Andrea Miyasaka from the Center of Genomics and Bioinformatics, Universidad Mayor; to Dr. Sebastián "Guti" Gutierrez, Dr. Felipe Villanelo, Alejandro Bernardin, Ignacio Fuenzalida, Jorge Carrasco, Pablo Monares, Dr. César Ravello, and other members of the Computational Biology Lab from Fundación Ciencia & Vida and their former members Dr. José Antonio "Toño" Garate, Yerko Escalona, and Paolo Arancibia; thanks to Valentina Frenkel, Magdalena Ribbeck, and Dr. Inti Pedroso.

I especially thank Prof. Guy Bart Stan from Imperial London College for their time and kind disposition to give me the opportunity and a desk to keep working while abroad and also the few talks we have about science and I hope “sparking the debut of a possible collaboration” will be honored sooner than later.

Special thanks to Mr. *Cardboard*, from the DLab, Fundación Ciencia y Vida.

Finally, this thesis was possible with the work and help from many developers, of them I thank especially to Pierre Boutillier and others at the Kappa-dev Team; to Alex Lubbock, Leonard Harris, Jeremy Muhlich, and others at the PySB development team; to Ingrid Keseler, Peter Karp and others at the EcoCyc, MetaCyc, and Pathway Tools software development team; to the SciPy, Numpy, pandas, Dask, and matplotlib development teams; to the Jupyter development team, and to the Python Software Foundation.

Financial support: The realization of this thesis was possible with the financial support from Comisión Nacional de Ciencia y Tecnlogía CONICYT through the Programa de Formación de Capital Humano Avanzado Doctorado 2014 (PCHA-2014-21140377), the partial funding from FONDECYT Proyecto de Iniciación (11140342 to Alberto J.M. Martín); the partial funding from FONDECYT Regular (11130518 to Daniel Garrido Cortés and 1181089 to Alberto J.M. Martin); the partial funding from Programa Iberoamericano de Ciencia y Tecnología para el Desarrollo CYTED (P918PTE0261 to Alberto J.M. Martin); the partial funding from CONICYT-PIA Proyecto Basal Fundación Ciencia & Vida (PFB16); the partial funding from Iniciativa Científica Milenio/Ministerio de Economía, Fomento y Turismo Proyecto Centro Interdisciplinario de Neurociencia de Valparaíso (ICM-Economía P09-022-F); the partial funding from the US Air Force Office of Scientific Research (AFOSR) Project FA9550-16-1-0111, the funding from Pontificia Universidad Católica de Chile Vice-rectory of Research (VRI), and funding from the School of Engineering of Pontificia Universidad Católica de Chile.

Powered@NLHPC: This thesis was partially supported by the supercomputing infrastructure of the NLHPC (ECM-02).

This thesis was partially supported by the computing infrastructure of the Centro de Genómica y Bioinformática, Universidad Mayor.

LIST OF PAPERS

The thesis is based on the following papers and book chapter, referred to in the text by the respective chapters:

Chapter I: Bustos A.M., Fuenzalida I., **Santibáñez R.**, Pérez-Acle T., Martin A.J.M. (2018). **Rule-Based Models and Applications in Biology.** In: von Stechow L., Santos Delgado A. (eds) Computational Cell Biology. Methods in Molecular Biology, vol 1819. Humana Press, New York, NY

Chapter II: Pérez-Acle T., Fuenzalida I., Martin A.J.M., **Santibáñez R.,** Avaria R., Bernardin A., Bustos A.M., Garrido D., Dushoff J., Liu J.H. (2018). **Stochastic simulation of multiscale complex systems with PISKaS: A rule-based approach.** *Biochemical and Biophysical Research Communications* 498:2, 342-351

Chapter III: **Santibáñéz R.,** Garrido D., Martin A.J.M. **Automatic rule-based modeling of bacterial gene regulation and metabolism.** Manuscript under review.

Chapter IV: **Santibáñéz R.,** Garrido D., Martin A.J.M. (2019). ***Pleione*: A tool for statistical and multi-objective calibration of Rule-based models.** *Scientific Reports* 9:1, 15104

Chapter V: **Santibáñéz R.,** Rodríguez-Fernández M., Garrido D., Martin A.J.M. ***Sterope:* an open-source package for global sensitivity analysis of Rule-Based Models.** Manuscript under preparation.

Annex Chapter I: Thomson P., **Santibáñez R.,** Aguirre C., Galgani J.E., Garrido C. (2019). **Short-Term Impact of Sucralose Consumption on the Metabolic Response and Gut Microbiome of Healthy Adults.** *British Journal of Nutrition* 122:8, 856-862

Annex Chapter II: Monares P., Liu J.H., **Santibáñez R.,** Bernardin A., Fuenzalida I., Pérez-Acle T., Zhang R. Jiqi. **Assessing the Role of Trust Profiles for the Economic Growth of Societies: A Stochastic Rule-based Simulation using the Prisoner´s Dilemma Game.** *IEEE Transactions on Computational Social Systems* 7:4, 849-857

PROCEEDINGS

Parts of this work have been presented at international congresses under the following references:

Rodrigo Santibáñez, Daniel Garrido, Tomás Pérez-Acle, Alberto J.M. Martin.
Stochastic Modeling of Gene Regulatory Networks in Escherichia coli
In: 4th International Synthetic & Systems Biology Summer School,
Cambridge, United Kingdom, July 2017 (Oral and poster presentations)
In: 13th ISCB Student Council Symposium,
Prague, Czech Republic, July 2017 (Poster presentation)
In: 25th Intelligent Systems for Molecular Biology & the 16th European Conference on Computational Biology joint conference, ISMB/ECCB 2017
Prague, Czech Republic, July 2017 (Poster presentation)
In: 3rd ISCB UK Regional Student Group´s Bioinformatics Student Symposium,
Aberystwyth, United Kingdom, August 2017 (Oral and poster presentations)

Rodrigo Santibáñez, Daniel Garrido, Alberto J.M. Martin.
Automatic Rule-Based Model Reconstruction and Model Calibration of a Gene Regulatory Network
In: 5th European Student Council Symposium,
Athens, Greece, September 2018 (Poster presentation)
In: 17th European Conference on Computational Biology, ECCB2018
Athens, Greece, September 2018 (Poster presentation)

Rodrigo Santibáñez, Daniel Garrido, Alberto J.M. Martin.
Automatic Reconstruction of Rule-Based Gene Regulatory Network Models and its Calibration
In: 3rd Latin America Student Council Symposium,
Viña del Mar, Chile, November 2018 (Poster presentation)

In: 5th ISCB-LA SOIBIO EMBnet 2018 Joint Bioinformatics Conference,
Viña del Mar, Chile, November 2018 (Oral and poster presentations)

Rodrigo Santibáñez, Daniel Garrido, Alberto J.M. Martin.
Stochastic modeling of gene regulation in Escherichia coli K-12
In: XXIV Congreso Latinoamericano de Microbiología, ALAM2018,
Santiago, Chile, November 2018 (Poster presentation)

James Liu, Tomas Perez-Acle, and Rodrigo Santibáñez
Simulating the Rise and Fall of Prosperity in Developed and Developing Societies
In: Cultural Evolution Society Conference CES2018,
Tempe, Arizona, USA, October 2018 (Oral presentation)

Rodrigo Santibáñez, Daniel Garrido, Alberto J.M. Martin.
An automatically reconstructed model of bacterial gene regulation enables simulation, prediction, and perturbation of gene responses
In: 8th Congress of European Microbiologists,
Glasgow, Scotland, United Kingdom, July 2019 (Oral presentation)

Rodrigo Santibáñez, Daniel Garrido, Alberto J.M. Martin.
Pleiades Toolkit: Automatic rule-based modeling of bacterial gene regulation enables simulation, prediction, and perturbation of gene responses
In: 15th ISCB Student Council Symposium,
Basel, Switzerland, July 2019 (Oral and poster presentations)
In: 27th Intelligent Systems for Molecular Biology & the 18th European Conference on Computational Biology joint conference, ISMB/ECCB 2019 (Oral and poster presentations)

FIGURE INDEX

TABLE INDEX

ABBREVIATION INDEX

ABC: ATP-Binding Cassette; also Approximate Bayesian Computation

ADI: Acceptable Daily Intake

a_i: Propensity (reactivity) of reaction or rule *i*

a_0: Total propensity, total reactivity

ANOVA: Analysis of Variance

API: Application Programming Interface

A_r: Set of transformations to apply on a site graph

AMP: Adenosine Monophosphate

ATP: Adenosine Triphosphate

AU: Arbitrary Unit

AUC: Area Under the Curve

ABM: Agent-Based Model

ASV: Amplicon Sequence Variant

BMI: Body Mass Index

BNG: BioNetGen stochastic simulator

BNF: BioNetFit (calibration tool)

BNGL: BioNetGen Language

BS: DNA Binding Site

C: City

CDS: Coding DNA Sequence

CI: Confidence Interval

CME: Chemical Master Equation

CRP: Catabolite Repression Protein

CS: Complex System

D: Dimension

DIN: Dynamic Influence Network

EcoCyc: Encyclopedia of *Escherichia coli* Genes and Metabolism

EGFR/ERK: Epidermal Growth Factor Receptor/Extracellular signal-Related Kinases

ENA: European Nucleotide Archive

FBA: Flux Balance Analysis

FDA: Food and Drug Administration

GA: Genetic Algorithm

GDP: Gross Domestic Product

GEM: Genome-Scale Metabolic Model

GEO: Gene Expression Omnibus

GPR: Gene-Protein-Reaction

GRN: Gene Regulatory Network

GSMM: Genome-Scale Metabolic Model

GT: Game Theory

GTI: Global Trust Inventory

HOMA-IR: Homeostatic Model Assessment Insulin Resistance

IR: Interaction rate

K: Equilibrium constant

k: Reaction rate

k_i: Rate for reaction i

KaSa: Kappa Static Analyzer

KaSim: Kappa Simulator

LHS: Left-Hand Side

MCA: Metabolic Control Analysis

MOGA: Multi-Objective Genetic Algorithm

MPI: Message Passing Interface

NAS: Non-caloric Artificial Sweetener

NFsim: Network-Free simulator

OCaml: Objective Categorical Abstract Machine Language

OCDE: Organisation for Economic Co-operation and Development

ODE: Ordinary Differential Equation

OGTT: Oral Glucose Tolerance Test

PCA: Principal Component Analysis

PD: Prisoner´s Dilemma

PDE: Partial Differential Equation

PDG: Prisoner´s Dilemma Game

PISKaS: Parallel Implementation of a Stochastic Kappa Simulator

PRO: Promoter

PySB: System Biology modeling in Python

RBM: Rule-Based Model

RBS: Ribosome Binding Site

RHS: Right-Hand Side

RNAP: RNA Polymerase

ROI: Region of Interest

r_0: The growth rate of a disease

SBML: Systems Biology Markup Language

SALib: Sensitivity Analysis Library

SD: Social Dilemma, Standard Deviation

SDE: Stochastic Differential Equation

SEIRD: Susceptible-Exposed-Infected-Removed-Dead

SF: Sigma Factor

SIR: Susceptible-Infected-Removed

SLURM: Simple Linux Utility for Resource Management

SOGA: Single-Objective Genetic Algorithm

S_r: Site graph for rule r

SSA: Stochastic Simulation Algorithm

t: Time

T: Expected time for next reaction

TER: (transcription) terminator

T_R: Expected time for reaction R

TOST: Two One-Sided t-Tests

UPGMA: Unweighted Pair Group Method with Arithmetic Mean

$Var(\bar{\theta})$: The variance of estimations from Jackknife datasets

WCM: Whole-Cell Model

Greek Letters:

κ: *kappa* (an RBM language)

τ: Time; also, the Kendall rank correlation coefficient

θ_i: An estimation using Jackknife datasets

$\bar{\theta}$: Mean of the estimations from Jackknifes

$\hat{\theta}$: An estimation using the totality of measurements

$\hat{\theta}_{Jack}$: Jackknife estimation after bias correction

Other symbols

∅: Empty set

∞: Infinite

LIST OF CONTENT

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
SCHOOL OF ENGINEERING

DEVELOPMENT OF RULE-BASED MODELS FOR REGULATORY MECHANISMS OF GENE EXPRESSION AND BACTERIAL METABOLISM

Thesis submitted to the Office of Research and Graduate Studies in partial fulfillment of the requirements for the Degree of Doctor in Engineering Sciences by

RODRIGO ALBERTO SANTIBÁÑEZ PALOMINOS

ABSTRACT

Systems modeling is a wide field in science and engineering aimed to understand how the components of a system evolve through time and space. For centuries, modeling has been used for the description of a variety of simple systems and with the advent of computation, we were able to model highly detailed systems, analyze them, and propose hypotheses for further experimental testing.

In this doctoral thesis, the utilization of rule-based models was proposed for the rapid development of draft models describing the regulation of bacterial gene expression. The developed methodology is able to model transcription, translation, and the degradation of macromolecules as well the bacterial metabolism. The aforementioned processes are essential for cell viability, albeit modeling is widely adopted for metabolism. To circumvent the time-consuming development of models, the modeling proposition is accompanied by the development of computational tools to automatically model each process, a technique previously available only for metabolic models.

Regulation of gene expression is essential for cell homeostasis and adaptation. This regulation relies on transcription factors and other proteins that bind specific DNA sequences and control genetic programs. However, the complexity of this regulatory network precludes efforts to model gene regulation at a genome-scale. In the first place, we propose a methodology build upon the *Kappa Biobrick Framework*, which was automated and extended to describe correctly the genome architecture, initiation of bacterial transcription, and incorporate metabolism. We developed *Atlas*, a software that is able to convert the many interactions and reactions encoded in biological networks in rule-based models. Rules are similar to chemical equations describing only the characteristics involved in a reaction. In doing that, a rule could represent thousands and even millions of individual reactions. The method was employed with known gene regulation data and metabolic reactions of the bacterium *Escherichia coli*.

Later, we developed *Pleione*, a software that employs a genetic algorithm to calibrate rule-based models. The tool gives support to four stochastic simulators (compatible with two rule languages). *Pleione* distributes simulations and calculations of the goodness of fit in high-performance computing infrastructures, and more importantly, harness equivalence statistical tests to determine the pertinence of stochastic simulations to experimental data. We also developed tools to estimate parameter uncertainty of calibrations and to estimate sensitivity indexes of user-selected parameters.

Modeling is viewed as a specialized task, inaccessible without proper knowledge of modeling frameworks. *Atlas* takes inspiration on available tools to reconstruct draft Genome-Scale Metabolic Models, and this thesis satisfactorily presents a software library to develop and analyze rule-based models for gene regulation and metabolism in bacteria. The developed tools allow the assess the impact of modifications like gene copy number, operon architecture, and other common genetic modifications to understand bacterial physiology, pathogenicity, and eventually, the engineering of bacterial cells for biotechnology and biomedicine applications.

The tools presented in this doctoral thesis are open-source and freely available for download from the Python Package Index and from github.com/networkbiolab/pleione and github.com/networkbiolab/PythonCyc.

Members of the Doctoral Thesis Committee:

Daniel Garrido

Timothy Rudge

María Rodríguez

Álvaro Olivera

Ernesto Pérez-Rueda

Santiago, May 2020

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA

DESARROLLO DE MODELOS BASADOS EN REGLAS DE LOS MECANISMOS DE REGULACIÓN DE LA EXPRESIÓN GÉNICA Y DEL METABOLISMO BACTERIANO

Tesis enviada a la Dirección de Investigación y Postgrado en cumplimiento parcial de los requisitos para el grado de Doctor en Ciencias de la Ingeniería.

RODRIGO ALBERTO SANTIBÁÑEZ PALOMINOS

RESUMEN

El modelado de sistemas es un amplio campo en ciencias e ingeniería enfocado en entender cómo los componentes de un sistema evolucionan a través del tiempo y espacio. Por siglos, el modelado ha sido usado para la descripción de una variedad de sistemas simples y con la llegada de la computación, hemos sido capaces de modelar sistemas altamente detallados, analizarlos y proponer hipótesis para posterior testeo experimental.

En esta tesis doctoral, la utilización de modelos basados en reglas fue propuesta para el rápido desarrollo de modelos borradores describiendo la regulación de la expresión génica bacteriana. La metodología desarrollada es capaz de modelar transcripción, traducción y la degradación de macromoléculas así como también el metabolismo bacteriano. Los procesos anteriormente mencionados son esenciales para la viabilidad celular, aunque el modelado es ampliamente adoptado para el metabolismo. Para sobrepasar el lento desarrollo de modelos, la proposición de modelado está acompaña por el desarrollo de herramientas computacionales para modelar automáticamente cada proceso, una técnica previamente disponible solo para modelos metabólicos.

La regulación de la expresión génica es esencial para la homeostasis celular y adaptación. Esta regulación depende de factores transcripcionales y otras proteínas que unen específicamente secuencias de ADN y controlan programas genéticos. Sin embargo, la complejidad de esta red regulatoria impide esfuerzos para modelar regulación génica a escala genómica. En primer lugar, nosotros proponemos una metodología construida sobre *Kappa BioBrick Framework*, la cual fue automatizada y extendida para describir correctamente la arquitectura del genoma, la iniciación de la transcripción bacteriana, e incorporar el metabolismo. Desarrollamos *Atlas*, un software que es capaz de convertir las muchas interacciones y reacciones codificadas en redes biológicas en modelos basados en reglas. Reglas son similares a ecuaciones químicas describiendo únicamente las características involucradas en una reacción. Al hacer eso, una regla puede representar miles o incluso millones de reacciones individuales. El método fue empleado con datos conocidos de regulación génica y reacciones metabólicas de la bacteria *Escherichia coli*.

Luego, desarrollamos *Pleione*, un software que emplea un algoritmo genético para calibrar modelos basados en reglas. La herramienta da soporte a cuatro simuladores estocásticos (compatible con dos lenguajes de reglas). *Pleione* distribuye simulaciones y cálculos de bondad de ajuste en infraestructuras de computación de alto rendimiento, y más importantemente, aprovecha pruebas estadísticas de equivalencia para determinar la pertinencia de simulaciones estocásticas a datos experimentales. También desarrollamos herramientas para estimar la incertidumbre en calibración de parámetros y para estimar índices de sensibilidad de parámetros seleccionados por el usuario.

El modelado es visto como una tarea especializada, inaccesible sin el conocimiento apropiado de métodos de modelado. *Atlas* toma inspiración en herramientas disponibles para reconstruir borradores de modelos metabólicos a escala genómica, y esta tesis satisfactoriamente presenta una biblioteca de software para desarrollar y analizar modelos basados en reglas para regulación génica y metabolismo en bacteria. Las herramientas desarrolladas permiten evaluar el impacto de modificaciones como variación del número de copias génicas, arquitectura del genoma, y otras modificaciones génicas comunes para entender la fisiología bacteriana, patogenicidad, y eventualmente, la ingeniería de células bacterianas para aplicaciones de biotecnología y biomedicina.

Las herramientas presentadas en esta tesis doctoral son de código abierto y disponibles libremente para descarga desde el Índice de Paquetes de Python y desde github.com/networkbiolab/pleione y github.com/networkbiolab/PythonCyc.

Miembros de la Comisión de Tesis Doctoral:

Daniel Garrido
Timothy Rudge
María Rodríguez
Álvaro Olivera
Ernesto Pérez-Rueda

Santiago, Mayo, 2020

1. INTRODUCTION

Mathematical and computational modeling are tools employed in systems biology, bioengineering, and other science fields with increasing use for the understanding of biological processes and their manipulation (Fisher & Henzinger, 2007). Given the technical advances in computing infrastructure and simulation methods, it is easier than ever model and simulates highly complex biological processes with high spatial and time resolution (Takahashi et al., 2005). Therefore, a variety of models have been developed and they are comprehensive compilations of knowledge and could be employed for further experimental design and hypotheses testing (Eng & Borenstein, 2016; Jahan et al., 2016; Khodayari et al., 2015). For instance, model complexity varies from simple enzyme kinetics (Wong et al., 2015) and cell growth (Wade et al., 2016) to the complex representation of metabolism at genome-scale (Simeonidis & Price, 2015) and phenotype (Carrera et al., 2014). Those models could make use of a simple system of equations that represent the dynamic of a few variables or complicated systems to reproduce the dynamics of thousands of cellular components.

Systems modeling could contextualize experimental data from diverse sources, which are common transcriptomics and proteomics (M. Kim et al., 2016; Sowa et al., 2017; Winter & Krömer, 2013). Those computational models seek to describe the phenotype as a function of the genotype, and once validated, its simulation is a cost-effective prediction tool of the outcome of experiments (Fisher & Henzinger, 2007). The evaluation of in silico interventions such as genetics modifications –single knockouts or copy number variation– has allowed rational strain design intending to improve the microorganism behavior in bioprocesses. For instance, Costa et al. (Costa et al., 2016) reviewed how the analysis of metabolic models has helped for the identification of genetic modifications able to optimize process parameters such as yield. Also and more importantly, new experimental data is incorporated into models as knowledge to improve sensitivity and specificity, or to develop new analysis methods (Costa et al., 2016; Zomorrodi et al., 2012). However, modeling of biological processes is a labor-intense task due to the structural complexity –understood for instance as a large number of components and reactions– of the considered systems (Vanlier

et al., 2013). The consequence of complexity in biological systems is the growth of reactions and their parameters, often exceedingly large for efficient analysis methods even with the incorporation of assumptions to reduce such complexity (Wade et al., 2016; Wong et al., 2015). In the case of large models, notably, there are two exponents: genome-scale metabolic models, GSMMs or GEMs (Terzer et al., 2009) and whole-cell models, WCMs (Sanghvi et al., 2013).

1.1 GENOME-SCALE METABOLIC MODELS (GSMMs).

Genome-scale metabolic models are an exhaustive recapitulation of cellular metabolism as they include all metabolic reactions in a compartment, cell, tissues, and communities. These GSMMs were firstly developed due to the computational limitation to identify sensible parameters for kinetic models based on ordinary differential equations (ODEs) (Hahl & Kremling, 2016; Terzer et al., 2009). The GSMMs describe the metabolic reactions as a stoichiometric matrix, where the columns are the reactions (of any type such as spontaneous, enzymatic, transport, and exchange) while rows represent the stoichiometric coefficient of every metabolite for each reaction (Terzer et al., 2009). A negative coefficient represents the consumption of the metabolite in the reaction and its synthesis by a positive coefficient. In the presented form, GSMMs are systems of linear equations without a single solution. The stoichiometric matrix has more reactions than metabolites, making them mathematically underdetermined. To solve a GSMM it is necessary the use of constraints to the reaction velocity (Kauffman et al., 2003). The constraints convert the GSMM into an overdetermined problem, reducing the feasible solution space, and allow the optimization of reaction rate (Costa et al., 2016), where flux balance analysis (FBA) is a commonly employed tool.

The first generation of constraint-based models is static and simulates a pseudo-stationary state. Later on, more complex models were developed to include changes in messenger RNAs (mRNAs), proteins, and/or metabolites concentration as constraints, allowing a more accurate and dynamic description of metabolism (Birch et al., 2014; Covert et al., 2008; Winter & Krömer, 2013). Besides the stoichiometric matrix and the constraints, and another important feature of GSMMs is the incorporation of genetic relationships known as gene-protein-reaction (GPR) associations (Dias et al., 2015; Machado et al., 2016) that

allows the predictive capability of the considered GSMM. For example, the GPR associations allow the identification of in silico essential genes (Terzer et al., 2009) and evaluate sensitivity and specificity concerning experimental data. Also, the GPR associations allow the evaluation of single, double, or multiple gene knock-outs and its impact on the optimal production rate, including biomass (B. Kim et al., 2015).

The GSMMs are the most successful methodology to describe and simulate metabolism at a large scale, for which is available a vast number of analysis tools (Zomorrodi et al., 2012, 2014). These models have allowed the accurate prediction of cellular growth in diverse genetic and environmental conditions (Edwards et al., 2001; Mccloskey et al., 2014).

1.2 WHOLE-CELL MODELS (WCMs).

The development of GSMMs can be viewed as an intermediate step towards a better understanding of every biological process that regulates metabolism (King et al., 2015; Smallbone et al., 2007). For instance, GSMMs are unable to evaluate the impact of gene deletions or amplifications of genes without metabolic function (enzymes or transporters) or evaluate metabolism rate under activation or repression caused by the concentration of metabolites (Carrera & Covert, 2015; Karr, Takahashi, et al., 2015). Regulatory information has been added to GSMMs as part of refinements outside the above description, known as ensemble or hybrid models such as GSMMs coupled to ODEs to describe cell growth during (fed-)batch (Gottstein et al., 2016). Whole-cell models are computational models aiming to include every gene product and their functions in every cell process (Purcell et al., 2013). The first WCM was published in 2012 and is a quantitative description of every mechanism controlled by and that control cell metabolism. Karr et al. (Karr et al., 2012) developed a WCM for the bacteria *Mycoplasma genitalium*, an intracellular human pathogen with a small genome of 580,070 bp and 525 genes (W. H. Chen et al., 2016). Due that the stationary state assumption made for GSMM does not apply to the totality of the cell processes, is impracticable the modeling using only mathematical formalism (Karr, Williams, et al., 2015). The Karr et al. WCM is an ensemble model of 28 different submodels, covering every cell process employing the best available method (Karr et al., 2012). The simulator is ad-hoc, simulating each model separately and combined at every second until cell division.

Including every gene function, the WCMs promise to predict with higher accuracy the phenotype and have broader applications in comparison with GSMMs (D. M. Morris & Jensen, 2008; Purcell et al., 2013). For instance, the *M. genitalium* WCM was able to determine with genes impact the most cell viability, and those genes been proposed as pharmacological targets for future treatment of *M. genitalium* infections (Karr et al., 2012).

1.3 STOCHASTIC MODELS AND RULE-BASED MODELS.

Although ODEs are frequently employed (Szigeti et al., 2018) to model from a simple system of equations to highly complex kinetic models, nowadays is recognized that some biological properties remain disregarded as long it is assumed their impacts are small enough. In specific, the deterministic simulation of ODE models disregards two fundamental facts: cellular processes are stochastics and involve thousands of different interactions and thousands of types of cell components which are discrete entities (J R Faeder et al., 2005; Fisher & Henzinger, 2007; Hahl & Kremling, 2016; Wilkinson, 2009).

Cell processes exhibit intrinsic and extrinsic noise without exception (Kærn et al., 2005; Raj & van Oudenaarden, 2008). For instance, extrinsic noise results in correlated gene expression and refers to the variation in the availability of factors that control gene expression (Shahrezaei et al., 2008; Swain et al., 2002), while intrinsic noise refers to the inherent variability and results in uncorrelated gene expression for identical genes (Raj & van Oudenaarden, 2008). To model stochastic processes, it is necessary to consider mathematical formalisms different from mainstream tools. Notably, the chemical master equation (CME) is the classical approach and dynamically define all possible states and the transition probability of them. The CME defines "the probability of a reaction R_j to happen somewhere in the volume Ω in the infinitesimal time interval $[t, t + dt)$ given the state $X(t)$ equal to x" (Engblom, 2008; Hahl & Kremling, 2016). For instance, in a simple model of gene expression, the CME solution relates the noise of mRNA and its encoded protein to the Poisson distribution (Wilkinson, 2009) and a decay process to the binomial distribution (Daniel T. Gillespie, 2007). However, rarely the CME has an analytical solution and numerical solutions were hard to obtain. Daniel Gillespie described in 1977 a practical algorithm for the numerical solution, the stochastic simulation algorithm (SSA), to simulate

reactive systems which its components are homogeneously distributed (the propensity of reaction is identical and does not depends on the spatial distribution) (Daniel T. Gillespie, 2007). Often, the modes of a model simulated with the SSA and with numerical integrators for ODEs do not differ, but when they do, the conclusion is that variability is high enough and an ODE does not provide an adequate description of the system true behavior (Daniel T. Gillespie, 2007; Wilkinson, 2009). Although the CME is the natural candidate to formulate stochastic models instead of stochastic differential equations (SDEs), the large size of biochemical systems precludes an exhaustive description using only a CME.

Rule-based models (RBMs) are an adequate alternative to describe comprehensively interactions and reactions for any system under consideration. Rules are similar to chemical equations that represent elementary reactions (Chylek et al., 2014; Hogg et al., 2014) in which only intervene objects with the necessary properties for the execution and describe only the modified properties after the execution (Feret et al., 2009). As Rules are similar to chemical equations, the "substrates" are described at the left (left-hand side, LHS) and the "products" are described at the right (right-hand side, RHS). Figure 1-1 shows the five basic Rules and exemplifies one using an "agent" (the abstract representation of a real entity) which has two sites (x and y). Optionally, each site could encode a particular state of the site (*orange* and *green*) with a meaningful property (figure adapted from Feret et al., 2009).

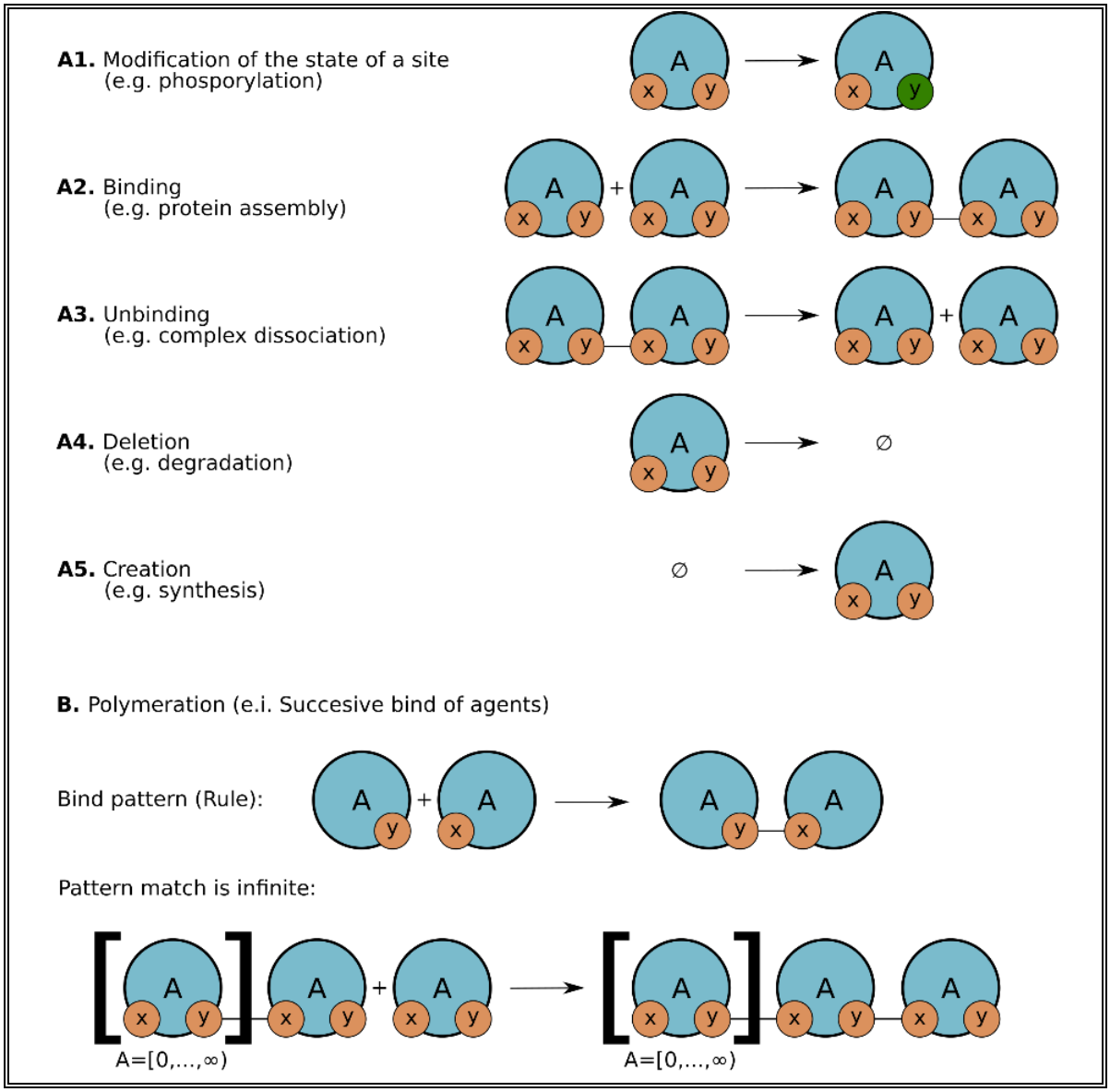


Figure 1-1. Basic patterns in rule-based modeling.
A. The five basic rules: Change of site state, defined as the change of the value of a *characteristic* (from *orange* to *green*) potentially representing phosphorylation of a protein residue; Binding and unbinding of agents, two rules that represent the assembly and separation of protein complexes; and the Deletion and Addition of agents, effectively modeling the introduction and subtraction of components from the considered system. An agent is the abstract representation of any component of the modeled system, in its free form (e.g. agent *A*) or in combination with others (e.g. agent *AA*).
B. Example of polymerization rule. Defining only the binding between the involved sites of identical agents, a single rule is only necessary to describe the lineal binding (and circular in a unimolecular reaction) of any number of agents into an arbitrarily long polymer.

It is common for the utilization of RBMs to describe and simulate (cellular) signaling pathways due to a large number of possible interactions (James R Faeder et al., 2003). For instance, the immunoglobulin IgE high-affinity receptor model is described by 354 different species. The counterpart RBM only required 15 rules (6 of them reversible) and 21 rates to model 3680 reactions (James R Faeder et al., 2003). Another example is the human insulin receptor which is composed of eight proteins that could be in 145 156 469 non-isomorphic conformations, without considering further post-translational modification (Koschorreck et al., 2007; Koschorreck & Gilles, 2008). Finally, the epidermal growth factor receptor/extracellular regulated kinase (EGFR/ERK) pathway model required only 70 rules, that reconstitute 10^{23} distinct molecular species (Danos et al., 2007a). In the case of numerical simulation employing ODEs, the model will require an explicit description of the many species and reactions. That is, the case of the IgE model (James R Faeder et al., 2003) requires the description of 354 ODEs with mass balance equations summing up 3680 reactions instead of the original 15 rules. The description of a large number of reactions that derive from a single rule is referred to as the *combinatorial explosion*. The three small, medium and large-size examples manifest explicitly how the rule-based modeling allows a compact description of the reactive system while employing ODE models could hinder analysis, calibration, inspection, correction, and the ensemble of models (J R Faeder et al., 2005; Koschorreck & Gilles, 2008).

Due to the large size of the equivalent ODE models, the Gillespie´s SSA has been implemented to simulate stochastically RBMs instead of the cumbersome deterministic simulation of ODE models (Daniel T. Gillespie, 2007; Hogg et al., 2014; McCollum et al., 2006). Although the original SSA is affected by the *combinatorial explosion*, network-free simulators such as NFsim (Sneddon et al., 2008) and KaSim (Danos et al., 2007a) discover the many species encoded through the simulation itself. The BioNetgen and kappa stand out as the most employed rule languages (Lopez et al., 2013), although exist other languages are less known such as *Allosteric Network Compiler* (Ollivier et al., 2010) and *Infobiotics* (Blakes et al., 2011). As noted, many rule languages have been proposed, while multiple simulation algorithms and software have been developed to increase the modeling spectrum and reduce calculation time, although sacrificing statistical accuracy (Wilkinson, 2009). A

useful property of rules languages is the ability to declare explicit compartments using an extended notation (Sorokina et al., 2013; Stewart & Wilson-Kanamori, 2011). Specialized software such as BioNetGen (James R. Faeder et al., 2009), *Spatial kappa* (Sorokina et al., 2013), and PISKaS (Bustos et al., 2018; Perez-Acle et al., 2018) have simplified the modeling process and reduced the computational cost of single simulations.

1.4 MODELING BIOLOGICAL PROCESSES

There are few consensuses on how to model biological processes due to the high structural complexity and interdependence of biological processes (Yordanov et al., 2016). For example, the most simple mathematical expression to model mRNA synthesis and degradation is $\emptyset \rightarrow RNA \rightarrow \emptyset$ (Bewick et al., 2009) and have being employed recurrently to simulate gene transcription (Hahl & Kremling, 2016; Qu et al., 2011). As expected, a more complex computational model should consider the availability of RNA polymerase, transcription factors, RNases, and nucleotides among other functional and structural components that affect the mRNA synthesis and degradation rates. Moreover, (bacterial) transcription is a succession of events involving the reversible binding of RNA polymerase to the promoter and the initiation, transcription, and termination of mRNA synthesis (A. Robinson & Van Oijen, 2013). Similarly, DNA replication and RNA and protein degradation also could be considered mechanisms composed of many successive events (Racle et al., 2015; A. Robinson & Van Oijen, 2013; Zouridis & Hatzimanikatis, 2007), and some authors have employed the presented idea to model transcription and translation from a theoretical point of view, e.g. (MacNeil & Walhout, 2011). The complexity of the aforementioned processes (number of components and interactions) hinders the development of WCMs from omics data and knowledge databases. Without a streamlined method to develop drafts WCMs, its adoption is hampered and precludes the analysis of cellular networks or the design of bacterial cells with applied purposes in biotechnology or biomedicine.

As aforementioned, the main reason to choose rules languages to model biological processes is the capacity to describe complex mechanisms in a concise and compact approach. Cells are the conjunction of distinct processes such as DNA replication (A. Robinson & Van Oijen, 2013), DNA transcription (A. Robinson & Van Oijen, 2013),

translation (A. Robinson & Van Oijen, 2013), RNA degradation (Strahl et al., 2015), protein degradation (Sekar et al., 2016; Striebel et al., 2009), metabolism (Costa et al., 2016; Srinivasan et al., 2015), cell division (Adiciptaningrum et al., 2015), signaling (Klipp & Liebermeister, 2006), protein translocation (J. K. Liu et al., 2014), assembly and degradation of protein complexes (e.g. RNA polymerase and ribosomes Davis et al., 2016; Gupta & Culver, 2014; Ishihama, 1981), post-translational modifications (Soufi et al., 2015), DNA modification (Casadesús & Low, 2013), DNA repair (Gowrishankar, 2015), as well as other general and specific bacterial processes such as sporulation or conjugation. As cells are complex enough systems, the thesis is centered at the molecular biology central dogma coupled to metabolism.

The central dogma was coined by Francis Crick and defines the information flow between biological molecules (not the mechanisms of such flow). The central dogma includes DNA replication, DNA transcription, and RNA translation. To obtain an adequate computational description of cell behavior, the central dogma should be coupled to the consumption of nucleotides, amino acids, and cofactor, while its production requires the synthesis of the corresponding enzymes (Lerman et al., 2012). Another necessary feature is the competitive and cooperative behavior exhibited by some transcription factors (Gutierrez et al., 2012). As a generalization, competitive behavior is expressed when two transcription factor DNA binding sites overlap at some extension. On the other hand, cooperative behavior when the binding of a transcription factor facilitates the binding of a second at an adjacent binding site (Shinar et al., 2006). Also, competition and cooperation have been described for replication, transcription, and translation (A. Robinson & Van Oijen, 2013), and metabolism (Notebaart et al., 2014).

Three advantages were identified for the proposed model of metabolism coupled to the five processes that regulate gene expression. The model will eliminate the dependence in omics data for the correlation of variables (e.g. transcriptomics to constraint further allowed reaction rates in GSMMs), potential incorporation of every gene which products are enzymes and regulators of metabolism or gene expression, and explicit bidirectional interaction of gene regulation and metabolism (Labhsetwar et al., 2013). The noted advantages have been explored previously, although not modeling explicitly the considered

mechanisms (Lerman et al., 2012; O´Brien et al., 2013; Pey et al., 2013). Consecutively, and in contrast to automatic tools that draft and refine GSMMs (semi-)automatically (Dias et al., 2015; N. E. Lewis et al., 2012; Magnúsdóttir et al., 2016), WCMs lack a unified methodology of development and the thesis will propose a software to develop such models for bacterial cells.

To develop a WCM limited to the processes for the central dogma coupled to the metabolism, a divide-and-conquer strategy will be developed from useful network representations of the many interactions. Networks (or graphs) encode any kind of data, and biological networks represent meaningful information, typically physical interactions, correlations, or directional effects. As networks can be arbitrarily large and complex, the method will employ data from the EcoCyc database (Keseler et al., 2017) to validate the modeling methodology. *Escherichia coli* strain K-12 MG1655 is the best-known microorganism from a genetic and cellular point of view (Baumstark et al., 2015; Bennett et al., 2009; S. Li et al., 2013; Mori, 2004), and the EcoCyc database recapitulates more than 32 000 publications (Keseler et al., 2017).

1.5 HYPOTHESIS AND OBJECTIVES

The main hypothesis of this thesis is that rule-based modeling in conjunction with tailored software allows automated reconstruction from biological networks into computational models at genome-scale for the mechanisms of gene expression regulation coupled to the bacterial metabolism, and the efficient stochastic simulation for model analyses.

General Objective

The general objective of this thesis is to develop software that allows the automatic reconstruction of rule-based models for the central dogma coupled to bacterial metabolism from biological networks such as genome graphs, and metabolic, gene regulatory, and interaction networks for protein-protein, RNA-protein, and mRNA-regulatory RNAs.

The specific objectives are:

1. To develop software for the automatic reconstruction of rule-based models that describe the regulation of gene expression and metabolism.

1.1 Automatic reconstruction of models employing the *Kappa BioBrick Framework*.

1.2 Extension to incorporate genome architecture of transcription factor binding sites, DNA polymerase binding sites, and genes.

1.3 Extension to couple metabolism to gene expression.

2. To propose a draft model for the central carbon metabolism of the bacterium *Escherichia coli* strain K-12 MG1655.
3. To develop computational tools to calibrate and analyze rule-based models.

1.6 APPROACH OF THIS THESIS

In this work, the central theme is to develop a computational tool that enables a complete reconstruction of an RBM for bacterial gene regulation coupled to cell metabolism. Data from the EcoCyc database for the bacterium *Escherichia coli* was employed to test the proposed methodology. Three additional computational tools were developed to calibrate, estimate the uncertainty of model parameters, and to perform sensitivity analysis.

- Chapter 1 discusses how to develop RBMs. The Chapter presents an informal derivation of the Gillespie´s Stochastic Simulation Algorithm and its use through a simple example. Consequently, it presents the κ-algorithm that performs the SSA to a site graph describing the modeled system, and through examples of increased complexity, it finally describes a Lotka–Volterra dynamics in spatially defined environments.
- Chapter 2 discusses through three examples the utilization of the PISKaS software to simulate κ language RBMs of explicit compartments with and without transport. The PISKaS software is an extension of the κ algorithm that simulates dependent SSA and perform corrections to the dynamics when transport reactions will change the system component quantities. It presents the three examples that describe genomic and plasmid gene regulation, disease outbreaks, and social prosperity in game theory.
- Chapter 3 discusses the formulation of a modeling framework able to transform a biological network into an RBM. The Chapter presents a divide-and-conquer strategy to convert four types of biological networks: genome graphs and gene regulatory, interaction, and metabolic networks. Consequently, it presents *Atlas*, a software that incorporates the formulated framework for the draft of RBMs and it describes how to curate and convert networks into a model of the bacterium *Escherichia coli*.
- Chapter 4 presents *Pleione*, a software for the calibration of any RBM employing a genetic algorithm. The software gives support for four stochastic software compatible with two rule languages. More importantly, the Chapter discusses the use of three equivalence statistical tests for the comparison of stochastic trajectories to experimental data.

- Chapter 5 presents *Sterope*, a software to perform a global sensitivity analysis of *kappa* RBM. The software is only compatible with the KaSim software due to the calculation of the Dynamic Influence Network. The Chapter compares, when appropriate, the results using *Sterope* for RBMs with flux control coefficients of exact ODEs and it shows the advantage to analyze RBMs in which their ODE counterparts suffer from excessive combinatorial explosion.

In addition to the main chapter, the thesis explores and describes bioinformatics approaches to analyze data and develop further the presented tools to analyze other systems.

- Annex Chapter 1 analyzes 16S sequence data from a placebo-treatment study. The microbiome composition at the phylum level was correlated to the consumption of sucralose in subgroups of healthy men adults in a two-week intervention.
- Annex Chapter 2 presents a rule-based model for the evaluation of theoretical societies with the translation of survey data into model parameters. It compares and discusses the evolution of two societies with disparate trust distributions, cooperation-deception probabilities, and interaction rates.
- Annex Chapter 3 presents *Alcyone*, a software developed around *Pleione* to determine the uncertainty in model parameters through the implementation of the one-leave-out Jackknife and the Bootstrapping methods. The Chapter shows results for a simple model analyzed with the two methodologies.
- Annex Chapter 4 shows how to extend *Pleione* to calibrate *antimony* models, a specialized programming language for deterministic and stochastic simulation (ODE solvers and SSA, respectively). The Chapter presents the calibration of a simple model with the Tellurium software, a package that reads and simulates *antimony* models.
- Annex Chapter 5 describes how to implement parallel computing workloads with four methodologies, all implemented in the software presented in the thesis main body.

2. CHAPTER I: RULE-BASED MODELS AND APPLICATIONS IN BIOLOGY.

Álvaro Bustos[1], Ignacio Fuenzalida[1], **Rodrigo Santibáñez**[1,2], Tomás Pérez-Acle[1,3], and Alberto J.M. Martin[1,4]

[1]Computational Biology Lab, Fundación Ciencia & Vida, Santiago, Chile.
[2]Escuela de Ingeniería, Pontificia Universidad Católica de Chile, Santiago, Chile.
[3]Centro Interdisciplinario de Neurociencias de Valparaiso, Valparaiso, Chile.
[4]Centro de Genomica y Bioinformatica, Facultad de Ciencias, Universidad Mayor, Santiago, Chile.

Book Chapter in Computational Cell Biology: Methods and Protocols book series, volume 1819 (2018). Springer New York, Ed. Louise von Stechow and Alberto Santos Delgado

2.1 SUMMARY

Complex systems are governed by dynamic processes whose underlying causal rules are difficult to unravel. However, chemical reactions, molecular interactions, and many other complex systems can be usually represented as concentrations or quantities that vary over time, which provides a framework to study these dynamic relationships. An increasing number of tools use these quantifications to simulate dynamically complex systems to understand better their underlying processes. The application of such methods covers several research areas from biology and chemistry to ecology and even social sciences.

In the following chapter, we introduce the concept of rule-based simulations based on the Stochastic Simulation Algorithm (SSA) as well as other mathematical methods such as Ordinary Differential Equations (ODE) models to describe agent-based systems. Besides, we describe the mathematical framework behind Kappa (κ), a rule-based language for the modeling of complex systems, and some extensions for spatial models implemented in PISKaS (Parallel Implementation of a Spatial Kappa Simulator). To facilitate the understanding of these methods, we include examples of how these models can be used to

describe population dynamics in a simple predator-prey ecosystem or to simulate circadian rhythm changes.

2.2 THE STOCHASTIC SIMULATION ALGORITHM (SSA)

The SSA, also known as Gillespie´s algorithm (Daniel T. Gillespie, 2007), is the basis of most stochastic simulation tools available. This algorithm and the tools based on it assume there is a homogeneous and "well-stirred" system of particles named agents. Agents can represent any type of entity within a system, i.e., molecules or individuals, and the interactions between agents are determined by a set of rules or equations taking place at in rates. These rules are ordered and divided into agents to which the rule applies and products (outcome agents). For instance, in a system of chemical reactions described by an equation or rule ($reactants \rightarrow products$), every set of particles matching the left side of the equation (or reactant agents) has an equal probability of being the subject of that rule; that is, to undergo the process described by the rule. To clarify, given a reaction of the form $A + B \rightarrow C$ in a system with 1000 particles of type A and 1000 of type B, this "well-stirring" assumption means that every pair of particles $\{A_1, B_1\}$, $\{A_1, B_2\}$, …, $\{A_i, B_j\}$, …, $\{A_{1000}, B_{1000}\}$ has equal probability of interacting to produce a particle of type C. Another important assumption made by the SSA is that the volume or area where the simulation takes place is fixed, and thus, concentrations of agents correspond to the discrete number of agents of each type.

To describe chemical systems, and this can be extended to any other type of system, a specific set of reactions is required. Reactions in this algorithm always match the following schema (Equation 2-1):

$$m_1 A_1 + \cdots + m_r A_r \rightarrow n_1 C_1 + \cdots + n_s C_s \qquad (2\text{-}1)$$

Whenever a reaction of this type takes place, a set of m reactant particles of types A_i are removed from the simulation (for $i = 1, \ldots, r$) and are in turn replaced by another set of n product particles of types C_j (for $j = 1, \ldots, s$). It should be noted that any of the products C_j could be of the same kind as one of the reactants, and that this schema covers reactions as simple as $A_i \rightarrow A_j$ to more complicated reactions requiring several types of different

agents. Other important reactions that follow the same schema are $A_i \rightarrow \emptyset$ to indicate degradation of agents and very similarly, $\emptyset \rightarrow A_i$ to model the addition of a new element in a system. Rules are applied according to reaction rates, which defines different behaviors of the system upon variations in the concentration of its reactants.

The quantity of each type of agent, or state of the system, at a given time t can be represented by a vector of non-negative integers, or state vector, in which each entry represents the amount of each agent type. The outcome of a given chemical reaction can also be represented by a state-change vector, with the same size as the state-vector at time t. The negative entries in the state-vector depict the consumption of an agent, positive values mean the creation of an agent, and 0 or *null* value indicates no change for a particular agent type. Therefore, if the state vector before a given reaction r is $\vec{X}$ and the associated state-change vector is $\vec{d}_r$, the state of the system changes from $\vec{X}$ to $\vec{X} + \vec{d}_r$ after the reaction occurs. For example, in a medium that has samples of three different chemicals A, B, and C, the following vector represents the existence of 1000 molecules of type A, 900 of type B, and 1200 of type C at time t:

$$\vec{X} = \begin{bmatrix} 1000 \\ 900 \\ 1200 \end{bmatrix}$$

In a similar way, the following two chemical reactions r_1 and r_2 correspond, respectively, to the two state-change vectors $\vec{d}_{r_1}$ and $\vec{d}_{r_2}$:

$$r_1 : 2A + B \rightarrow C \qquad\qquad r_2 : 2C \rightarrow 2A + B + C$$

$$\vec{d}_{r_1} = \begin{bmatrix} -2 \\ -1 \\ +1 \end{bmatrix} \qquad\qquad \vec{d}_{r_2} = \begin{bmatrix} +2 \\ +1 \\ -1 \end{bmatrix}$$

Note that in the second reaction one of the particles of type C acts as a catalyst and the outcome effect of the reaction r_2 is the same as of $C \rightarrow 2A + B$ with a $\vec{d}_{r_2}$ state-change vector identical to $-\vec{d}_{r_1}$. However, the relationship between the different probabilities of

reactions r_1 and r_2 happening and the number of particles of type C present leads to different long-term behaviors of the system.

A reaction r is fully specified by the state-change vector $\vec{d}_r$ and a propensity function a. This propensity function takes the state vector $\vec{X}$ as an argument and calculates the rate of every reaction r in the system; thus, if $a(r_1, \vec{X}) > a(r_2, \vec{X})$ reaction r_1 is more likely to occur than r_2. This is a discrete model; therefore, the function $a(r, \vec{X})$ is combinatorial in nature. For a fixed r, it should, theoretically, be directly proportional to the number of distinct sets of molecules that match the left side of the equation describing the reaction r and the physical properties of the medium being simulated. In this way, a probabilistic mathematical model of any set of reactions can be built given their state-change vectors $\vec{d}_r$ and a propensity function a. The function a reflects the constraints given by the chemical nature of the system being modeled and allows the description, at least indirectly, of the probability distribution of the possible future state of the system, given its initial state and a time-lapse:

$$P(\vec{x}, t|\vec{x}_0, t_0) \coloneqq \mathbb{P}(\vec{X}(t) = \vec{x}|\vec{X}(t_0) = \vec{x}_0) \tag{2-2}$$

Equation 2-2 is the Markovian condition that assumes the future state of the system relies exclusively on the present state $(\vec{x}_0, t_0)$ and the propensity function of each possible reaction. To be precise, to get to state $\vec{x}$ at time $t + dt$ for a dt small enough to ensure that the probability of two reactions occurring in that time interval is negligible, either the state at time t is also $\vec{x}$, or the state at time t is $\vec{x} - dr$, and reaction r takes place during the interval $[t, t + dt]$. Thus, we have the following approximate equality (in which R is the set of all reactions):

$$\begin{aligned} P(\vec{x}, t + dt|\vec{x}_0, t_0) \\ &\approx \sum_{r \in R} P\left(\vec{x} - \vec{d}_r, t\middle|\vec{x}_0, t_0\right)\mathbb{P}(\text{reaction } r \text{ happens in } [t, t\ +\ dt]) \\ &+\ P\left(\vec{x} - \vec{d}_r, t\middle|\vec{x}_0, t_0\right)\mathbb{P}(\text{no reaction happens during } [t, t\ +\ dt]) \\ &\approx \sum_{r \in R} P\left(\vec{x} - \vec{d}_r, t\middle|\vec{x}_0, t_0\right)a\left(r, \vec{x} - \vec{d}_r\right)dt + P(\vec{x}, t|\vec{x}_0, t_0)\left(1 - \sum_{r \in R} a(r, \vec{x})dt\right) \end{aligned}$$

From the last expression, moving the term $P(\vec{x}, t|\vec{x}_0, t_0)$ to the left side of the equality and dividing by dt, the following identity is obtained as $dt \to 0$:

$$\frac{d}{dt}P(\vec{x}, t|\vec{x}_0, t_0) = \sum_{r\in R}\left[P(\vec{x} - \vec{d}_r, t|\vec{x}_0, t_0)a(r, \vec{x} - \vec{d}_r)dt - \sum_{r\in R}P(\vec{x}, t|\vec{x}_0, t_0)a(r, \vec{x})dt\right] \qquad (2\text{-}3)$$

Equation 2-3, commonly known as the Chemical Master Equation (CME), can be rigorously formalized from the laws of probability and the theory of Markov processes (Daniel T. Gillespie, 1992), but for simplicity, we will use the informal derivation given above. Although the previous equation is theoretically enough to determine the probabilities involved in the simulation at any moment given an initial state (i.e., the function $P(\vec{x}, t|\vec{x}_0, t_0)$), determining an explicit form of $P(\vec{x}, t|\vec{x}_0, t_0)$ analytically from the CME is usually extremely hard. This difficulty is due to Equation 2-3 being a system of coupled differential equations with one function for each different state vector, and thus it can potentially have infinite unknown functions. Therefore, using this equation directly as a basis for a simulation is extremely impractical for systems composed of many different types of agents and/or with a large number of rules. However, it is possible to construct accurate numerical Markov simulations that follow the distribution given by the CME (Daniel T. Gillespie, 2007). To accomplish this and accurately simulate the future state of a system based on information of the current state, only two questions need to be answered:

• Which reaction will happen next? and

• How much time will pass from now until it happens?

Thus, for an accurate simulation, we only need information about the conditional probability distribution of the next reaction r and expected time τ. So, we define the function $p(r, \tau|\vec{x}, t)$ as follows:

$$p(r, \tau|\vec{x}, t)d\tau \approx \mathbb{P}(\text{the next reaction happens in the interval } [t + \tau, t + \tau + d\tau] \text{ and is of type } r\big|\vec{X}(t) = \vec{x})$$

If we assume the Markovian memoryless property, this probability should be independent of the current time t; thus, the definition can be simplified slightly by removing references to t:

$$p(r,\tau|\vec{x},t)d\tau$$
$$\approx P\big(\text{no reactions during } [0,\tau] \text{ and a reaction of type } r \text{ during } [\tau,\tau+d\tau] \,\big|\, \vec{X}(0)=\vec{x}\big)$$

Assuming that every reaction r takes place independently of all other reactions, the Markovian assumption tells us that the expected time T_r until a reaction of type r is an exponential variable of rate $a(r,\vec{x})$ [(Pardoux, 2010), chapters 6–7]. Thus, the time $T = \min_{r\in R} T_r$ until the next reaction is an exponential variable of rate $a_0(\vec{x}) \coloneqq \sum_{r\in R} a(r,\vec{x})$, and it is independent of the reaction r chosen. Therefore, an explicit value for the probability density p can be easily determined:

$$P(r,\tau|\vec{x},t) = a(r,\vec{x})\exp(-\tau a_0(\vec{x})) \qquad (2\text{-}4)$$

Equation 2-4 can be used to generate trajectories that follow the desired distribution since it implies that the probability of choosing a given reaction r is $a(r,\vec{x})/a_0(\vec{x})$ and it is independent of the expected time T, we get the following simple algorithm for generating valid trajectories given an initial state x_0:

1. Initialize the state $\vec{X}$ as $\vec{x}_0$ and the current time t to 0.

$$\vec{x}_0 = \vec{X}(t=t_0) = \begin{bmatrix}1000\\900\\1200\end{bmatrix} \times [A \quad B \quad C] \qquad \begin{matrix}2A+B \to C\\ 2C \to 2A+B+C\end{matrix}$$

2. Generate two random numbers p_1, p_2 in $[0,1]$ (uniform distribution), for example:
$$p_1 = 0.18 \text{ and } p_2 = 0.67$$
3. Determine the reactivity of the system as $a_0(\vec{x}) = \sum_{r\in R} a(r,\vec{x})$ and set δt as the value $\ln(1/p_1)/a_0(\vec{X})$. This ensures that the random variable δt has an exponential distribution with rate $a_0(\vec{X})$. For simplicity, each reaction occurs at the same frequency $r = r_1 = r_2 = 1.0s^{-1}$
$$a_0(\vec{x}) = r_1 \times A \times (A-1) \times B + r_2 \times C \times (C-1)$$
$$a_0(\vec{x}) = r_1 \times 1000 \times (1000-1) \times 900 + r_2 \times 1200 \times (1200-1)$$
$$\delta t = \ln(1/0.18)/900{,}538{,}800 = 1.90 \times 10^{-9}s \;=\; 1.90ns$$

4. Suppose the set of reactions is given by $R = \{r_1, \ldots, r_j, \ldots, r_n\}$. The probability to choose any reaction $r \in R$ is $a(r, \vec{X})/a_0(\vec{X})$. We choose the reaction r_j testing the following inequality:

$$\sum_{j=1}^{n-1} \frac{a(r_j, \vec{X})}{a_0(\vec{X})} \leq p_2 < \sum_{j=1}^{n} \frac{a(r_j, \vec{X})}{a_0(\vec{X})}$$

For example, given the two reactions r_1 and r_2, we test the following inequalities:

$$0 \leq 0.67 < \sum_{j=1}^{1} \frac{a(r_j, \vec{X})}{a_0(\vec{X})} = 0.9984$$

$$0.9984 \leq 0.67 < \sum_{j=1}^{2} \frac{a(r_j, \vec{X})}{a_0(\vec{X})} = 1.000$$

5. Replace the old value of t by the new value $t + \delta t$ and the old value of $\vec{X}$ with $\vec{X} + \vec{d}r$, where r is the reaction chosen in step (4).

$$\vec{x}_1 = \vec{X}(t = t_0 + \delta t) = \begin{bmatrix} 1000 \\ 900 \\ 1200 \end{bmatrix} \begin{bmatrix} -2 \\ -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 998 \\ 899 \\ 1201 \end{bmatrix}$$

6. Save the new values of t and $\vec{X}$ and go back to step (2) or finish if $a_0(\vec{X}) = 0$.

This is a basic form of the SSA, readers interested in a more in-depth analysis of the model may consult the review by Gillespie (Daniel T. Gillespie, 2007). Common methods for the simulation of rule-based models use adapted versions of this algorithm to generate accurate simulations, each approach making certain assumptions and often requiring a formal language to describe the models. Examples of such SSA-based implementations are BioNetGen (Chylek et al., 2015) and KaSim (https://kappalanguage.org), each with its formal language (BNGL, James R. Faeder et al., 2009; and *kappa*, Murphy et al., 2010, respectively).

2.3 INTRODUCTION TO ORDINARY DIFFERENTIAL EQUATIONS MODELS

Another common approach to the study of the dynamic behavior of complex systems employs ODEs or Partial Differential Equations (PDEs) based on the empirical law of mass action (Jost, 2011; Tu, 1994). This law states that the rate of a chemical reaction is proportional to the activity of each of its reactants. In order to simplify the model, it is often assumed that such activity values match the concentrations of each reactant. While this is generally not true, for elemental reversible reactions with no intermediate steps, it is a reasonable assumption and an acceptable approximation. For instance, given an elemental reversible reaction such as the following:

$$A + B \rightleftarrows C$$

the rate at which the forward reaction $A + B \rightarrow C$ occurs is proportional to the concentrations of A and B, with a similar remark applying to the backward reaction. This simple reversible equation prompts the following three ODEs systems as a candidate for modeling its evolution or dynamic behavior over time

$$\frac{d[A]}{dt} = -k_1[A][B] + k_2[C]$$
$$\frac{d[B]}{dt} = -k_1[A][B] + k_2[C]$$
$$\frac{d[C]}{dt} = k_1[A][B] - k_2[C]$$

in which $[X]$ stands for the concentration of the reactant X and k_1 and k_2 are rate constants usually determined from experimental data. The right-hand side of the equation represents that in the forward reaction ($A + B \rightarrow C$), one instance of A and one of B are replaced by one of C, with the opposite happening for the reverse reaction.

This small system of ODEs is usually nonlinear. The model has a very simple structure and allows both numerical and theoretical analyses. For instance, equilibrium can be calculated assuming that $k_1[A][B] - k_2[C] = 0$, which leads to Equation 2-5:

$$K = \frac{k_1}{k_2} = \frac{[C]}{[A][B]} \tag{2-5}$$

where K is called the equilibrium constant of the system and does not depend directly on the concentrations of the reactive substances but only on the rate constants k_1, k_2. K governs the asymptotic behavior of the system as time goes to infinity (Quack & International Union of Pure and Applied Chemistry. Physical and Biophysical Chemistry Division., 2007; Silberberg, 2015); more precisely, a system of chemical reactions eventually reaches a situation in which the concentration of each chemical involved remains unchanged, with this value being determined by the constant K [(Silberberg, 2015), chapter 17]. This can be seen mathematically by noticing that the system of equations above has a constant solution whose value depends on K, and any other positive solution converges to this value as $t \to \infty$ (Tu, 1994).

However, for more complex reactions and systems with more types of agents, the setup of the ODE system and the structure of the resultant reactions become very difficult to simulate using this type of equation (Daniel T. Gillespie, 2007). Chemical reactions such as electrolysis, which involves two or more instances of the same reactant, introduce higher-order terms that might induce unexpected and/or difficult-to-explain behavior in numerical simulations. In addition, non-elementary reactions have to be decomposed into a series of elementary reactions, which can greatly increase the number of terms and variables involved in the system. Thus, the ODE approach becomes impractical very quickly on sufficiently complex chemical systems. Another drawback of this approach is that low concentrations or quantifications of agents can lead to unrealistic simulations of the behavior of the system in the long term upon the extinction of these agents. This is particularly noticeable in small systems comprised of only hundreds or thousands of agents.

Another characteristic of the ODE-based approach is that it is purely deterministic. Given that in a real chemical system there are random fluctuations and non-deterministic phenomena, a deterministic model might not be able to fully represent all of the possible outcomes of the system. As in the previous paragraph, it is worth mentioning that random fluctuations usually have negligible long-term influence in large systems with sufficiently high concentrations of every species in the system. However, they become much more

evident in systems with a lower number of components. In such systems, there are potential alternative outcomes (different from the average behavior simulated by deterministic models) with large quantitative differences and non-negligible probabilities. Hence, taking into account this non-deterministic behavior becomes essential to understand small-scale systems (Daniel T. Gillespie, 2007).

Lastly, ODE-based models usually carry no spatial information, as the medium is assumed homogeneous and well-stirred, with a uniform distribution of all system components. Here, we describe several biological systems in which those assumptions are invalid. The most straightforward way to create models that take into account spatial information is by replacing the concentration value as a function of simulated time for each entity $[X](t)$ by a spatial density term $\rho_X(t; x, y, z)$, which represents the density of X in a small neighborhood of points in the area or volume comprised by the model. Also, additional terms in the differential equations above are required to model physical phenomena that may affect density. For example, the chemical entities in the simulated system are liquids capable of diffusion; a possible set of equations for the reaction $A + B \rightarrow C$ could be defined as

$$\frac{\partial \rho_A}{\partial t} = -\Delta\rho_A - k\rho_A\rho_B$$

$$\frac{\partial \rho_B}{\partial t} = -\Delta\rho_B - k\rho_A\rho_B$$

$$\frac{\partial \rho_C}{\partial t} = -\Delta\rho_C + k\rho_A\rho_B$$

where each ρ term corresponds to the following sum of partial derivatives (known as the Laplacian or Laplace operator):

$$\Delta\rho = \frac{\partial^2 \rho}{\partial x^2} + \frac{\partial^2 \rho}{\partial y^2} + \frac{\partial^2 \rho}{\partial z^2}$$

This conforms to the usual diffusion-reaction from physics (as stated in Evans, 2009, Chapter 2), $\partial\rho/\partial t + \Delta\rho = 0$ (with constant diffusion rates uniformly equal to 1), after adding the additional terms brought by the law of mass action considering that for a very small neighborhood of a point (x, y, z) the term $[X]$ is proportional to ρ_X and the chemical X may be assumed approximately homogeneous.

2.4 PARALLEL IMPLEMENTATION OF SPATIAL κ

2.4.1 The κ Algorithm

In this section, we introduce a modified version of the SSA that allows more complex simulations in a variety of contexts beyond the standard chemical applications. We do not go in-depth into the mathematical formalisms behind the modifications of the SSA introduced here; these details may be consulted in publications about the κ language such as the work of Danos *et al.* (Danos et al., 2007a; Danos et al., 2007b). The discussion below follows the theoretical framework setup by Danos, with a schematic graphical notation whenever possible. For the actual language and syntax used in standard implementations such as KaSim, please consult the KaSim reference manual (Boutillier, Feret, et al., 2018). Nevertheless, the examples in this and the following sections can be easily implemented in KaSim, which provides all of the standard κ framework. Further examples that involve spatial information are designed to be compatible with PISKaS (Perez-Acle et al., 2018), which is a spatially-enhanced fork of KaSim.

The classical Gillespie´s algorithm treats every kind of chemical compound (or, in general, a variation of an agent) as a separate type, no matter how similar it may be to a previously existent type of compound (Danos et al., 2007b). This becomes problematic when there is a large number of different compounds that are similar —but not identical— as there is no way to express this similarity properly in the classic SSA framework, even if these cannot participate in the same reactions. Thus, these results in state-vectors with a large number of entries and (usually) several almost-duplicate reactions or rules involving small variants of the same compound, requiring too many computational resources to simulate such systems. Another problem is that the described SSA framework ignores the internal structure of the compounds involved. This is a problem when dealing with complex molecules such as proteins or DNA since their internal structure can severely influence the outcome of a chemical process out of sheer geometrical positioning, let alone physical or chemical constraints caused by the size of the molecule. Last, biological systems and other complex systems are naturally compartmentalized (cellular compartments), a characteristic difficult to replicate into a model using an algorithm that assumes a homogeneously mixed

environment where all reactions take place.

The first two observations made above suggest that a modification of the data structure used to store the current status of the system, as well as a change in the idea of what constitutes an agent, might allow for a more flexible and robust framework. The concerns about information regarding the fixed structure of a compound suggest that an atom is probably a better model than a molecule for the concept of "agent". An atom interacts with other atoms in several ways, the covalent bond being among the simplest to understand conceptually: Each atom can form a finite number of pre-established links of a specific type with one or more other atoms, which in turn can also have links between themselves. For instance, an oxygen atom can form two covalent bonds, or in other words, it has two "open places" where other atoms can bind to, while a hydrogen atom can form a single covalent bond. When two hydrogen atoms bond to one oxygen atom by forming two covalent bonds, a water molecule is formed. Similarly, chemical reactions can be expressed as the formation or destruction of links between reactants or agents.

This motivates storing the current state of the system as a site graph (Danos et al., 2007b). This graph corresponds to a network in which the nodes or agents have a specific structure that limits the kinds of connections or bonds that can be formed. More specifically,

• Each agent has a type. Going with our chemical analogy, this would correspond to the specific element (hydrogen, helium, oxygen, etc.) of each atom.

• Each type has a set of sites associated with it. Every site has a set of possible internal states. In our example, these sites correspond to the places in the atom where covalent bonds can be formed, while the internal states may correspond to markers of phenomena like partial charges, or differentiators between distinct types of chemical bonds.

• Each link between two nodes (agents) connects exactly one site from one of the agents to one site of the other; reciprocally, a site from an agent can be involved with at most one link with another agent. In our chemical example, this means, for instance, that each of the two "open positions" from an oxygen atom can participate in only one covalent bond with another atom and thus this atom can be bound to at most two other atoms at once.

Reactions or rules can be also described via site graphs. A rule r is expressed via a site graph S_r and a set of transformations A_r, which corresponds to the addition or removal

of edges between sites of S_r, changing their internal states, or adding or removing agents from S_r.

As a simple example, let us consider the reaction of electrolysis $2H_2O \rightarrow 2H_2 + O_2$. To describe this reaction, only two types of agents are needed: H and O, each one representing the type of atom. Agents of type H have one site, h_1, while agents of type O have two sites o_1 and o_2. For our current purpose, neither of the sites has a specific internal state. The reactants can be represented by a graph with six nodes (agents), two of type O and the rest of type H; each of the four sites o_i, $i = 1,2$ is linked to a single h_1 site from one of the H agents. Furthermore, the set of transformations to be applied to this graph are as follows:

- Delete the four $o_i h_1$ links.
- Add a $h_1 h_1$ link to the two H agents corresponding to each water molecule.
- Add two links, $o_1 o_1$ and $o_2 o_2$, between the two O agents.

The effects of these operations on a set of agents can be observed graphically in Figure 2-1.

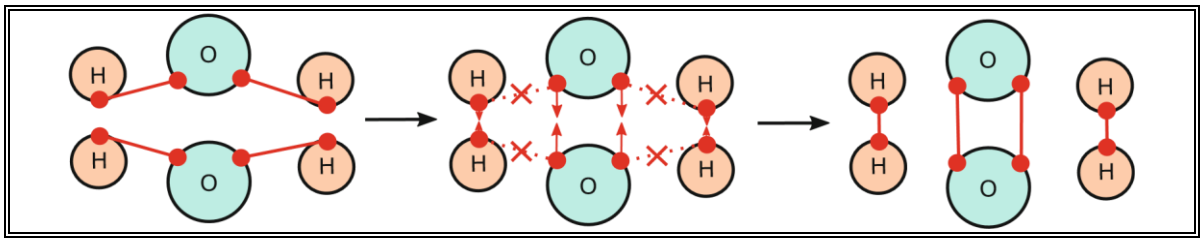


Figure 2-1. Rearrangement of agents over the application of a rule corresponding to the reaction of hydrolysis.

Observe that by the specific combinations of two types of agents, we are able to describe three different species participating in this reaction (H_2O, H_2, O_2). With the same two agents, other chemical species can be easily described. For example, ozone can be described using three O agents and different internal states to represent the hybridization of the chemical bonds involved. Another example is hydrogen peroxide, which uses two H and

two O agents, with a pattern of links mimicking the chemical structure of the molecule. By adding only a third type of agent with four sites c_1, ..., c_4, we can include carbon atoms in our model and thus represent the whole set of hydrocarbon species and other related types of compounds.

The κ framework allows declaring certain internal states or site links as "undefined," which allows applying the same rule to similar, but not identical, species. For example, the formation of alcohols from hydrocarbons corresponds to a set of very similar reactions, usually consisting of replacing a single H agent by a two-agent subgraph corresponding to the radical −OH. Therefore, specifying the whole structure of the hydrocarbon involved is usually superfluous.

2.4.2 Non-Chemical Models in κ: A Predator–Prey Ecosystem

While the original motivation for the SSA comes from the literal expression of chemical reactions, this framework can be used to model other types of systems where the agents involved do not represent chemical units but instead more complex entities. A simple example of this is the implementation of a predator–prey model, where agents represent a predator species that may consume other agents (prey). In this model, additional agents may also be used to indicate the availability of limited resources, such as plants or edible fruits for the sustenance of herbivorous prey.

A simple system involving two species A, B (prey and predator, respectively) can be modeled via a set of Lotka–Volterra equations (as seen in Kot, 2001, chapter 7, section B), which correspond to the following system of differential equations:

$$\frac{dA}{dt} = \alpha A - \beta AB$$

$$\frac{dB}{dt} = \gamma AB - \delta B$$

Here, α, β, γ, and δ are nonnegative rate constants. The two terms of the first equation are interpreted as follows: αA means that the rate of growth of the prey species is proportional to the number of extant members of the species (i.e., exponential growth); $-\beta AB$ represents the predation rate of members of A by the B species, which assuming a

homogeneous population is proportional to the product AB. With respect to the two terms of the second equation, these are interpreted as:

- The term γAB corresponds to the growth rate of the predator species, which is proportional to the number of extant members of B and A as well as to the number of available resources or the amount of prey population;
- The term $-\delta B$ is the rate of extinction of the predator, assumed to be proportional to the current population of B.

Note that the rate of natural death of A is neglected (technically, it can be represented by a diminished value of the constant α) as well as the dependence of A on other resources (for instance, available plants for an herbivorous animal). Moreover, the population density of both species is assumed constant. For instance, sexual reproduction is not considered, no age groups are taken into account (which makes this model inaccurate for predator species that target young members of the prey species), and no extinction of any of the involved species can be studied since the population densities are assumed to be constant.

The simple Lotka–Volterra model can be implemented as a κ model, allowing the inclusion of different parameters into the model such as natural dead, variable population densities, sexual reproduction, and age groups. In order to simplify notation, we expressed the model as chemical equations with internal states being represented via parentheses and linked sites via lines whenever necessary.

The rules of reproduction for A and extinction for B have a very simple format:

$$r_1 : A \rightarrow 2A$$

$$r_2 : B \rightarrow \emptyset$$

The reproduction rule for B has A as a catalyst, as the frequency at which reproduction of B occurs depends on the A population, as B will attempt to reproduce more often if there are more resources available, but this does not mean they should consume a member of A every time they attempt to reproduce. Thus, the rule is

$$r_3 : A + B \rightarrow A + 2B$$

Finally, the predation rule has B as a catalyst; for it to occur, a member of A needs to encounter a predator B. In this model, "hunger" or similar states are not considered. Thus, the rule appears as follows:

$$r_4: A + B \rightarrow B$$

The rates of each of those rules depend on the values of α, β, γ, and δ and they can be determined in the same way as if they were chemical reactions. Note that this model does not take into account internal states (e.g., hunger) or links between agents (e.g., two B agents acting together to capture prey). Next, we will discuss possible improvements to the model by using internal states or links to represent this type of situation.

One simple addition to this model would be the implementation of sexual reproduction. Of course, this will not apply to every type of species, and its effects might be negligible in simple ecological systems; however, for environments with a large disparity in sex distribution or acute sexual dimorphism, this approach might provide an accurate model.

To implement sexual reproduction into the model, we can use sites as a property of the agents. Sites are variables that can be used to store a finite set of values or states in the form of qualitative or quantitative descriptors. Thus, we can use a site g in each agent to represent the sex (e.g., ♀, ♂ for male and female, respectively, or Ψ for species with hermaphroditic individuals). Thus, the rules for sexual reproduction are as follows:

$$A(♀) + A(♂) \rightarrow A(♀) + A(♂) + A(♀)$$
$$A(♀) + A(♂) \rightarrow A(♀) + A(♂) + A(♂)$$
$$A(?) + B(♀) + B(♂) \rightarrow A(?) + B(♀) + B(♂) + B(♀)$$
$$A(?) + B(♀) + B(♂) \rightarrow A(?) + B(♀) + B(♂) + B(♂)$$

Note the "$A(?)$" term on the left side of the predator reproduction rules. As stated before, we allow for sites or links to be undefined so a single rule can be applied to every combination of internal states of A. In this case, what matters is that there are available resources (i.e., prey) and not the specific sex of the prey animals present. The "$A(?)$" term on the right-hand side means that the corresponding term on the left-hand side remains untouched. These rules add new agents of a specific type ($A(♂)$, $A(♀)$, $B(♂)$, $B(♀)$) without affecting the existing ones.

Age information or the stage of maturation of the agents can also be useful to improve the Lotka–Volterra basic model. For instance, we can suppose that predators more often capture young or elderly animals of the prey species due to inexperience, physical weakness, or illness. Similarly, only animals that have reached sexual maturity can reproduce, and in some species, elderly animals present diminished fertility.

To incorporate this information into the model, we include an additional internal state d, whose values correspond to the different stages of development of each species, for instance $\{D_c, D_y, D_a, D_e\}$ (child, young or adolescent, adult or sexually mature, elderly or senescent, respectively). We need to define rules of growth that make every agent transit through those internal states sequentially:

$$A(?, D_c) \rightarrow A(?, D_y)$$

For examples of sex- and age-segregated ecological models that served as the inspiration for the set of rules shown here, see *Fundamentals of Mathematical Ecology*, by Mark Kot (Kot, 2001).

Every rule introduced above could be reproduced in a relatively simple way in the usual Gillespie´s framework. However, the possibility to link agents through sites has not yet been covered in the κ language in this section. As an example, we will consider the formation of herds in both predator and prey species. A large group of prey animals can fend off a lone predator, while a prey animal can be more easily overwhelmed by a herd of social predators when alone or in a small group.

A potential way to implement this would be to add a few sites through which an agent can be linked to others of the same kind. Those links can represent social relations in the herd, and we can define rules to represent both herd protection and social hunting. For instance, we can add a few "relation sites," e.g., $p^{♀}$, $p^{♂}$, c, m (mother, father, child, mate) to represent a monogamous species with just one offspring per reproduction event and define the following state rules:

$$B(♀, D_a) \overset{m}{—} B(♂, D_a) \rightarrow B(♀, D_a) \underset{c, p^{♀}}{—} B(♂, D_c) \underset{p^{♂}, c}{—} B(♂, D_a)$$

Here, the notation $B\frac{s,t}{}B$ means site s from agent B is linked to site t of another agent B. Thus, this rule reads as follows: two agents who are in the "adult" stage of development and are a mating couple (there is a link between the m sites of both agents) engender a third agent (in this case, female) and the c site of each parent agent gets linked to the respective p site of the new $B(♀, D_c)$ agent (the former two agents get marked as parents to the new child agent).

2.4.3 Spatial Information in κ

Usually, the setup for the simulation algorithm of the κ framework in the standard implementations of κ such as KaSim (Boutillier, Feret, et al., 2018; Danos et al., 2007b) makes the same physical assumptions as the standard SSA, in particular, that the medium is homogeneous and well-stirred, which means that the agents are uniformly distributed in the environment. This simplifies the model defining the probability of two agents interacting as proportional to their respective population. While this assumption is valid for certain systems, e.g., chemical reactions in gases, it might not be applicable to systems that are not homogeneous or have spatial dependences. For instance, the cell membrane provides different chemical and physical properties to the intracellular and extracellular medium. Moreover, the transfer of certain substrates from one medium to the other is in itself a phenomenon of interest, which is entirely ignored by the κ framework. Thus, incorporating spatial information into an implementation of the κ language allows for more realistic models. However, the assumption of homogeneity cannot be completely eliminated, as we usually care only about large-scale tendencies and not individual agent behavior.

In the cell membrane system, there are two different and clearly defined media. Interactions between them consist of transportation of certain agents from one environment to the other through the membrane. We assume that both environments and cellular compartments are homogeneous; hence, the probability of a certain agent approaching the membrane depends on the quantity of that agent, which can be represented using rules.

One way to implement such a system of compartments is simply to add a new site w to every agent to represent each medium, which can have two states i and o corresponding

to inside and outside the cell. Each rule becomes a set of rules, one per compartment, to ensure that agents only interact with other agents in the same compartment:

$$A + B \to C \text{ becomes } \begin{cases} A(i) + B(i) \to C(i) \\ A(o) + B(o) \to C(o) \end{cases}$$

The transport rules change the states of agents from one compartment to another. In this case, we can represent this via rules like $A(i) \to A(o)$ and $A(o) \to A(i)$, each with a certain rate. In this way, to simulate osmosis through equal volumes, each of those rules should have an equal rate, such that the compartment with higher concentration has a higher rate of transportation.

For a more complete system with many more compartments, we can separate the cell space into a series of subspaces that we assume approximately homogeneous. This is analogous to the Riemann sums method to compute an integral:

$$\int_0^1 f(x)dx \approx \sum_{k=0}^{n-1} \frac{f(k/n)}{n}$$

To recall, this method allows to approximate the area under a curve given by the function f as a sum of the areas of small rectangles (Apostol, 1967). Here, we divide the interval $[0,1]$ into n equal subintervals (subspaces) of length $1/n$, and we assume that the value of $f(x)$ in $[k/n, (k+1)/n]$ is approximately $f(k/n)$, i.e., this assumption does not significantly affect the value of $f(x)$ and it is similar to the "approximately homogeneous" assumption from above. For Riemann summations, a large value of n, and, thus, smaller subintervals, gives a better estimation of $\int_0^1 f(x)dx$, at least when f is a continuous function. In a similar way, we can suppose that with smaller subspaces we should reach a better approximation.

If the space of agents has some kind of geometrical properties, we can represent them via the transport rules: Compartments that are geometrically adjacent should have a higher rate of transfer reactions. For instance, we could represent a cell as two different compartments, nucleus and cytoplasm (Figure 2-2), interconnected by several transport mechanisms, or build a more complex system where the nucleus is represented as a central

compartment surrounded by several other compartments representing subspaces of the cytoplasm (Figure 2-3).

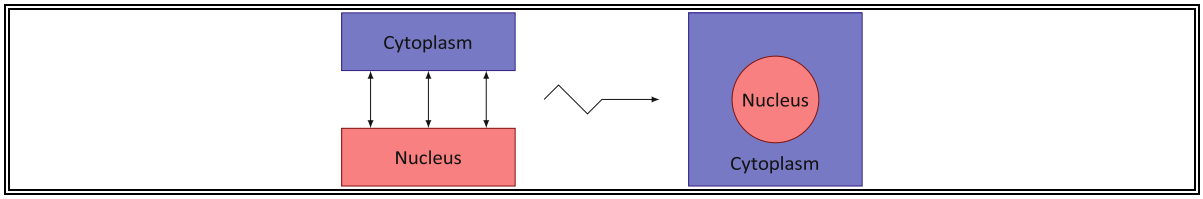


Figure 2-2. Internal representation and interpretation of a two-compartment model for a cell. Note that the geometry of the cell is not taken into account and thus the cytoplasm and the nucleus are taken as entirely homogeneous.

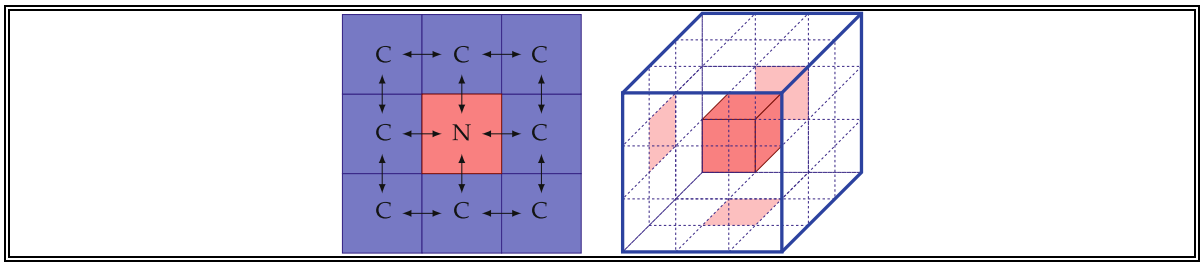


Figure 2-3. Potential compartment arrangements to represent the geometry of a cell. (Left) A 2D arrangement of nine compartments with eight representing the cytoplasm and the central one representing the nucleus. The arrows represent possible transport rules. (Right) A potential 3D version of the former 2D arrangement. The marked (central) compartment corresponds to the nucleus, while the other 26 represent the cytoplasm; there are transport rules among compartments that share a face.

To represent the geometry of the system, here we assume that there are only transport rules between adjacent compartments, automatically considering that the rate of any transport rule between nonadjacent compartments is zero.

As stated before, this does not require a special implementation of κ and can be done in the usual implementations such as KaSim by using a special site whose states correspond to the different compartments. However, since we need a copy of a rule for each compartment, this may result in several redundant rules that may severely affect execution time. Thus, an implementation allowing the explicit declaration of compartments can reduce the total number of rules needed and improve the performance of the simulation. In what follows, we will discuss one such implementation, namely Parallel Implementation of a

Spatial Kappa Simulator (PISKaS, Perez-Acle et al., 2018), which is a forked branch of KaSim. PISKaS focuses on the independent, parallel simulation of each compartment; however, most remarks below should apply to any spatial κ implementation.

2.4.4 Meta-population Dynamics in a Predator–Prey Model with Explicit Spatial Information

In the following section, we will introduce a model of population dynamics that attempts to reproduce the experimental results obtained in the work of Holyoak and Lawler (Holyoak & Lawler, 1996). The study subject here was the evolution of the population of a set of predator and prey species in an environment that allows spatial migration. The goal is to verify whether a prey species which is prone to extinction by predation can survive in a medium that allows for migration. The end result of the simulation appears to conform to the experimental results; further details of an implementation of this simulation can be consulted (Fuenzalida et al., 2015).

The experiment in Holyoak & Lawler (1996) studied two species of microorganisms, *Didinium nasutum* (predator) and *Colpidium striatum* (prey), which inhabit an environment consisting of several bottles (compartments) linked by four-way connectors in a specific configuration. Given that the prey species shows logistic behavior (i.e., short-term exponential growth, which eventually is stunted due to lack of resources, resulting in a stable population) in the absence of predators, it becomes natural to suggest a Lotka–Volterra model to represent the interaction between those two species, with rules similar to the ones introduced in Section 2.4.2.

This model incorporates transport rules to represent microorganisms from both species moving through the different bottles. A coarse geometry has to be introduced, representing the spatial arrangement of the system of bottles and connections. Each bottle can be assumed to be a homogeneous medium and, thus, can be taken as a compartment. In the original experiment, the configurations consisted of square grids of bottles joined by four-way connectors that linked each bottle to the bottles adjacent horizontally, vertically, and diagonally, as seen in Figure 2-4.

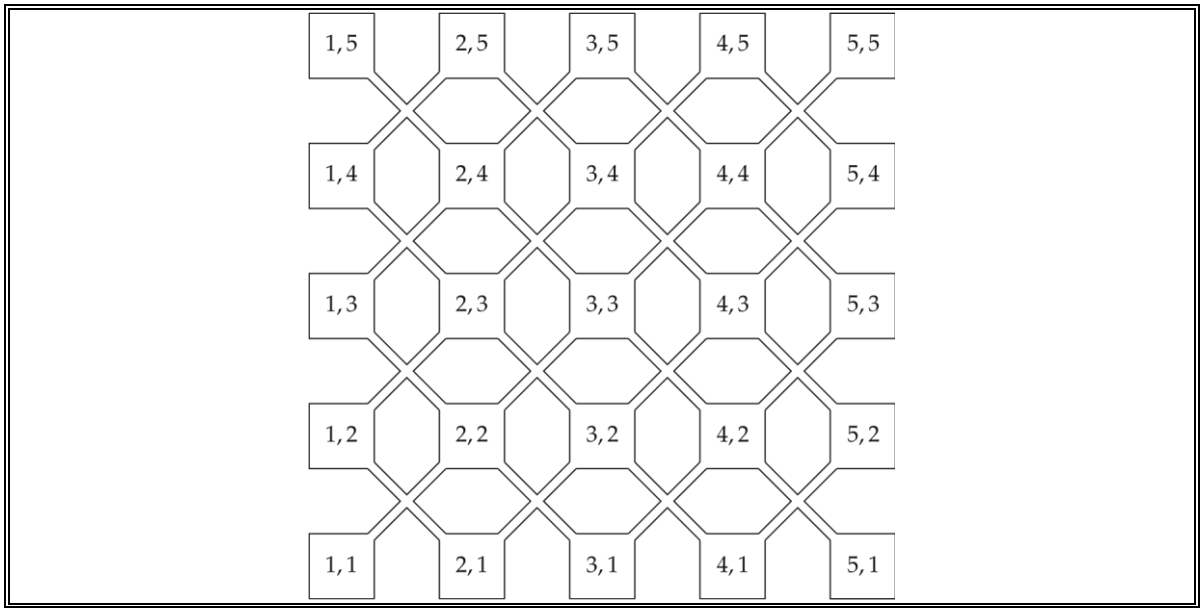


Figure 2-4. Representation of the bottle arrangement and the four-way connections linking each bottle to its neighbors.

Thus, this simulation is performed in an arrangement of 25 compartments, where each one represents a single bottle. The bottles have labels that represent their position in a square matrix of 5 rows and 5 columns; namely, a bottle labeled (i, j) is placed in the i-th column, j-th row. For example, the third bottle from the second row (from bottom to top) is labeled $(3,2)$. Transport rules allow the microorganisms from a bottle to move to other bottles but only to adjacent ones; for instance, a microorganism in the bottle $(3,4)$ can move only to the bottles with labels $(2,3)$, $(2,4)$, $(2,5)$, $(3,3)$, $(3,5)$, $(5,3)$, $(5,4)$, or $(5,5)$.

The rate of movement of the microorganisms between bottles is determined by analysis of the physical characteristics of the system. If no further information regarding the physical properties of the system is provided and there are no external factors affecting inter-bottle transportation, it is reasonable to assume that the movement of the prey species through the bottles corresponds to simple diffusion. In this way, diffusion happens at the same rate through each connection the bottles have, which is linked to the physical capacity of those connections (assumed equal in all directions). The predator species follows similar rules. This is summarized by the following set of rules, with equal rate λ_T for every non-boundary compartment:

$$\begin{gathered}\text{transport } A\colon (i,j) \to (i-1, j-1) \\ \text{transport } A\colon (i,j) \to (i, j-1) \\ \text{transport } A\colon (i,j) \to (i+1, j-1) \\ \vdots \\ \text{transport } A\colon (i,j) \to (i+1, j+1)\end{gathered}$$

with eight rules in total, one for every adjacent bottle. For bottles located in the border, the same reasoning applies, but with fewer rules (e.g., the bottle with a label $(1,2)$ should have five neighbors instead of eight, so there are five rules in that case). Given that some bottles may have more than one connection with another bottle (e.g., to get from $(2,2)$ to $(3,2)$ one can go up right first and then down right or down right first and then up right), we do not really need to assume all transport rates are equal and may adjust them accordingly if needed.

After stating the transport rules, we specify the behavior of both species. First of all, the agents will be of two types, prey (A) and predator (B); the prey species reproduce by simple mitosis, while the predator requires a certain minimal mass before it can undergo this process (Holyoak & Lawler, 1996). To implement this distinction, we need to introduce a way to count how many agents of the prey species have been consumed by a specific predator agent, and only allow "sated" predators to reproduce. We implement this by introducing agents with the following specifications:

• **Prey**: The associated agents have two sites, i and s. We use the second site to store the status of the prey agent (in this case, either alive, ℓ or dead, d), while the first site is used to link the prey agent to other agents to set up the aforementioned "counter."

• **Predators**: The corresponding agents have a single site a that is used to create links to a prey agent and store the state of the predator agent (h, f, d corresponding to hungry, sated or fed, and dead, respectively).

The rules are as follows:

• Prey species reproduction (mitosis):

$$A(?, \ell) \to A(?, \ell) + A(?, \ell)$$

• Predator feeding:

$$B(h) + A(?, \ell) \to B(h) \overset{a,\, i}{\text{———}} A(?, d)$$

$$? \overset{?,i}{\text{—}} A(?,d) + A(?,\ell) \rightarrow ? \overset{?,i}{\text{—}} A(?,d) \overset{s,i}{\text{—}} A(?,d)$$

The first rule is a rather straightforward statement of the situation that happens when a hungry predator meets an alive prey animal that then is consumed. The link formed between the B agent and the dead A agent indicates that the latter is incorporated into the mass of the predator.

The second rule, although less intuitive in the formulation, corresponds to the exact same statement; if an alive prey agent meets a dead prey agent that has its i site linked to some other agent, then the dead agent must be part of the mass of a predator, and thus this means the alive prey has encountered a predator. Once again, when this rule is applied, the alive prey is killed and consumed. Since both rules represent the same phenomenon, they should be assigned equal rates.

• Predator satiation:

$$B(h) \overset{a,i}{\text{—}} A(?,d) \overset{s,i}{\text{—}} A(?,d) \overset{s,i}{\text{—}} \ldots \overset{s,i}{\text{—}} A(?,d) \rightarrow B(f)$$

This rule specifies that a $B(h)$ agent with a sufficiently long chain of $A(?,d)$ agents linked to it (with "sufficiently long" corresponding to a specific number that represents the average amount of prey eaten before a predator undergoes mitosis) becomes a $B(f)$ agent (i.e., not hungry but sated) and the $A(?,d)$ agents are discarded and eliminated from the simulation. To ensure that the $A(?,d)$ agents are eliminated from the simulation immediately so that a predator does not continue feeding after reaching the satiation level, we give this rule a rate of ∞.

• Predator mitosis:

$$B(f) \rightarrow B(h) + B(h)$$

This is similar to prey mitosis, with the only difference being that we only allow $B(f)$ agents (i.e., predators that have gathered enough mass) to undergo this process. Since the agents representing the progeny of the parent agents will only have a fraction of the mass

of their parent, they need to gather additional mass themselves before they undergo reproduction, thus starting in the h (hungry) state.

• Predator unfed:

$$? \overset{?,i}{\text{---}} A(?,d) \overset{s,i}{\text{---}} A(?,d) \rightarrow ? \overset{?,i}{\text{---}} A(?,d)$$

This rule is the opposite of the second feeding rule. It states that a predator that has not been able to consume prey for long stretches of time has to utilize some of the mass it has already consumed and stockpiled for sustenance.

• Prey death:

$$A(?,\ell) \rightarrow \emptyset$$

• Predator death:

$$B(h) \rightarrow \emptyset$$

$$B(f) \rightarrow \emptyset$$

$$B(h) \overset{a,?}{\text{---}} ? \rightarrow \emptyset$$

The reason to have three distinct rules for predator death is to give different rates to each agent depending on how "hungry" they are. A $B(h)$ agent with no links does not have food reserves and is "starving" (the end result of a long period unfed), thus it has a higher rate of death than a $B(h)$ agent linked to others, i.e a predator that has been able to feed recently. Similarly, a $B(f)$ agent has a big reserve of mass and thus is not susceptible to starvation, hence having a lesser rate of death than other B agents.

• Prey cleanup:

$$A(?,d) \rightarrow \emptyset$$

This eliminates any $A(?,d)$ agents that do not have their i site linked to any other agent. Those $A(?,d)$ agents appear whenever a predator $B(h)$ linked to one or more $A(?,d)$ dies before reaching satiation, and since they serve no purpose anymore they are removed

from the simulation. As before, to ensure those agents are removed immediately we assign a rate ∞ to this rule.

With those rules, together with the transport scheme outlined above, we have set up a description for the predator–prey compartmentalized system as set up in the experiment by Holyoak and Lawler (Holyoak & Lawler, 1996). For a comparison between the results of the simulation and the experimental output, the reader may consult Fuenzalida *et al.* (Fuenzalida et al., 2015).

An equivalent ODE-based model applied in this specific example provides a fixed output: either both species always go extinct or both species manage to survive. This depends only on the parameters (rates) of the model, and thus it is impossible to see the effect that random fluctuations have on the output. The experiment by Holyoak and Lawler (Holyoak & Lawler, 1996) shows that these fluctuations are actually of importance in the current situation, as some iterations of the experiment showed total extinction while others result in survival under the same initial conditions. This can be interpreted as a result of the way in which the population of prey species migrates and distributes along with the bottles (which can be seen as random dispersal) and how the formation or dispersion of aggregates determines its survival, and by extension, the survival of the predator species as well. The usage of stochastic models in place of a deterministic ODE-based set of equations allows us to observe the effect that those random events have on the outcome of the simulations.

This model also allows us to observe the importance of spatial characterization in a simulation (see Figure 2-5). In this framework, we can easily observe how an isolated bottle usually reaches extinction events very quickly, which is analogous to the situation of a homogeneous medium with uniform population densities of the prey and predator species. Also, the complete system of linked bottles shows a much higher likelihood of survival of both species, also exhibiting the oscillatory behavior of both populations associated with the prey species favoring bottles with fewer population densities.

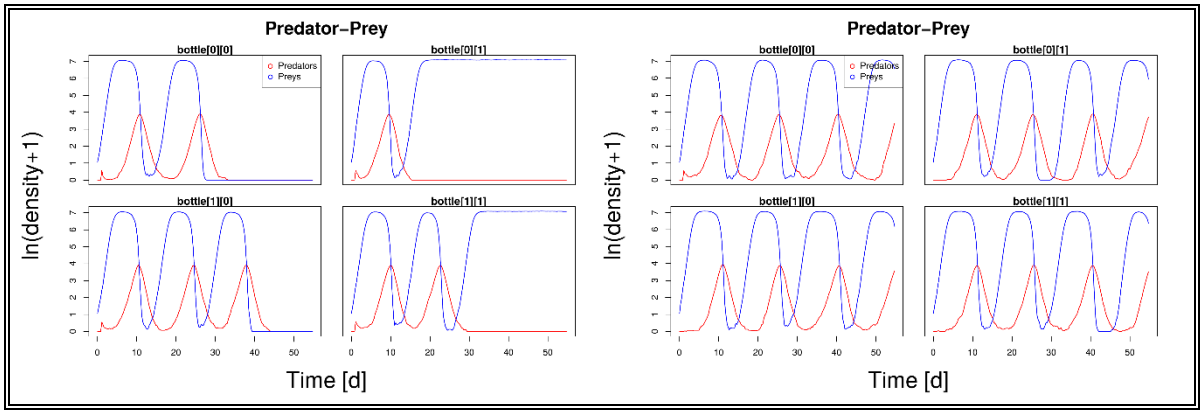


Figure 2-5. Population behavior of the species in a bottle arrangement of four cells. The figure shows the influence of spatial configuration, in particular, isolation on the left and migration on the right and extinction events.

2.4.5 A Circadian Clock Model

In this section, we discuss a simplified model of the mammalian circadian clock. Our goal is to represent how the day–night cycle affects the transcription processes inside the cell, resulting in a 24-h periodic behavior regarding the concentration of proteins, transcription factors, mRNA, and others. This is the second sample model described in (Fuenzalida et al., 2015) as an example of the usage of PISKaS for simulations, and it is based on a certain system of transcription factors regulated by the presence of sunlight and the molecular interactions and feedback loops initiated by them, as described in (Agostino et al., 2011).

The system modeled here consists of two compartments, corresponding to the nucleus and cytosol of the cell. Additional complexity can be added by dividing the cytosol into a set of compartments to represent the heterogeneity of the environment. For instance, the cell may be represented by a cube of $3 \times 3 \times 3 = 27$ compartments as shown in the right panel of Figure 2-3, with the central compartment representing the nucleus and the remaining 26 the cytosol, reflecting its geometric structure. However, for the sake of simplicity, this example uses the least complex two-compartment model.

We modeled the periodic behavior of five different genes in this model, *PER1*, *PER2*, *CRY1*, *CRY2*, and *NR1D1*, each coding a protein. In addition, there are two transcription factors (*CLOCK* and *BMAL1*) and a phosphatase (*CKI*), all of which are assumed to be at

constant concentrations within the cell. The model also considers an agent for each mRNA and its transportation to the cytoplasm. All protein components of this model are described in Table 2-1.

Table 2-1. Protein components of the circadian clock model and their respective UniProt IDs.

Type	**Name**	**UniProt Id**
Genes	*PER1*	O15534
	PER2	O15055
	CRY1	Q16526
	CRY2	Q49AN0
	NR1D1	P20393
Transcription factors	CLOCK	O15516
	BMAL1	O00327
Phosphatase	CKI	Q06486 or P49674

Genes and corresponding messenger RNA are represented by agents $G(i, s_1, \ldots, s_5)$ and $R(i)$, respectively, where the i site is used as an identifier of the encoded protein and, in the case of the gene agents, the s_1, ..., s_5 sites allow linking to other agents that represent transcription factors. Those agents, T_e and T_r, will have two sites, one to allow binding to DNA, while the other allows binding to certain proteins.

The proteins will be actually represented by different types of agents instead of a single one with an "identifier" site. The reason for this is different proteins interact in specific ways with the transcription factors and with each other, thus resulting in different numbers of binding sites and reactions. In this case, our protein agents will be declared as follows:

• $PER(i, p_1, p_2, s_{cry}, s_{cki})$: here, i is an identifier taking the values 1 or 2 (as we include two types of proteins that form similar kinds of bonds), p_1 and p_2 are phosphorylation sites (with states p phosphorylated and d not), and the remaining sites allow linking to other proteins and agents.

• $CRY(i, s_{per}, s_{clk})$: same as before, the i site is an identifier with two possible values, and the s sites are for interaction with other agents.
• $NR1D1(r)$: for this agent, we only include one site, which allows linking to a transcription factor T_r.

The rules describing the processes of phosphorylation and dephosphorylation that involve *PER* protein agents are handled by the phosphatase agents *CKI*.

Transport rules are to be limited to certain kinds of agents, since, for instance, we do not allow DNA to "leak" to the cytosol. Unlike the previous example, the linking between compartments is not symmetrical and has specific directions for each transport rule. For instance, we allow mRNA to move from the nucleus, where it is synthesized, to the cytosol where protein production occurs. However, there is no transport of mRNA from the cytosol back to the nucleus. In contrast, we allow certain *PER* and *NR1D1* proteins to move in both directions depending on the status of certain binding sites. A sample of those rules is as follows:

$$\text{transport } R: nucleus \rightarrow cytosol$$

$$\text{transport } PER\ \frac{s_{cry}, s_{per}}{}\ CRY: nucleus \rightarrow cytosol$$

$$\text{transport } PER(?, p, u): cytosol \rightarrow nucleus$$

$$\text{transport } NR1D1: cytosol \rightarrow nucleus$$

The *R* agents are only allowed to go in one direction, while *PER* agents are allowed to move in both directions, but only in certain arrangements (e.g., intranuclear transport is only allowed during a specific state of phosphorylation, while moving toward the cytosol is affected by the interaction between *PER* and *CRY* proteins).

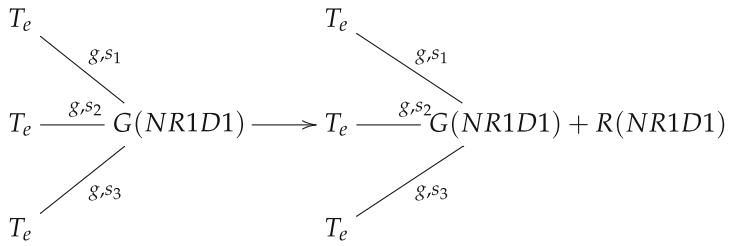
The remaining rules are also different in the two compartments. For example, since there is no DNA in the cytosol, there is no need to process rules pertaining to DNA transcription in the corresponding compartment, and similarly, there is no translation in the nucleus. PISKaS and similar rule-based compartmentalized software usually allow declaring some rules as exclusive for a subset of compartments. In this case, we have several rules to represent the phases of the encoding and expression process:

• **Translation rules**: Those rules govern the production of proteins with the information encoded in the corresponding messenger RNA, and thus are cytosol-exclusive. They take a very simple form, e.g.,

$$R(per_1) \rightarrow R(per_1) + PER(1, u, u, s_{cry}, s_{cki})$$

in which the identifier site of the mRNA agent R takes the value per_1 to indicate that it encodes a $PER(1, \dots)$ protein.

• **Transcription rules**: This kind of rule is nucleus-exclusive and manages the production of mRNA ($R(i)$ agents) from the corresponding DNA in the nucleus ($G(i, \dots)$ gene agents) in the presence of adequate transcription factors. Those rules are stated in ways similar to this example:



in which an *NR1D1*-encoding gene bonded through its first three sites to adequate transcription factors produces a $R(NR1D1)$ agent, representing the corresponding messenger RNA. This agent is afterward moved to the cytosol compartment using the respective transport rules, where it induces the creation of a $NR1D1(r)$ agent, representing the phenomenon of protein translation. Note that, as it is to be expected, $G(NR1D1)$ agents act only as catalysts with no modifications either to themselves or to the associated transcription factors.

• **RNA and protein degradation**: Rules to represent degradation, usage, or elimination of mRNA and proteins are included, in a similar way as the death rules in the previous example:

$$R(?) \rightarrow \emptyset$$

RNA degradation rules are deemed exclusive to the cytosol, while protein degradation rules are applied in both compartments.

• **Phosphorylation rules**: These are limited to *PER* agents and managed by *CKI* agents:

$$CKI\ \frac{\ell, s_{cki}}{}\ PER(1,u,u) \rightarrow CKI\ \frac{\ell, s_{cki}}{}\ PER(1,p,u)$$

Different rates can be given to other configurations involving distinct kinds of proteins or with other states of phosphorylation.

• **Protein reactions**: A set of rules to determine the interaction between proteins and transcription factors both inside and outside the nucleus. These rules are either exclusive to the nucleus or to the cytosol. For example, the interaction of phosphatase agents with *PER* proteins and their further linking to *CRY* proteins only happens in the cytosol:

$$PER(?) + CKI \rightarrow PER\ \frac{s_{cki}, \ell}{}\ CKI$$

$$PER(?, p) + CRY \rightarrow PER(?, p)\ \frac{s_{cry}, s_{per}}{}\ CRY$$

Note that the second reaction has to follow the first as phosphorylation of the PER agent has to be achieved.

• **Light-dependent transcription**: To express the dependence of this phenomena to the day–night cycle, we incorporate an additional set of transcription rules, which do not depend on T_e or T_r agents. Those rules have a variable rate, with much higher activity during the "daytime"; otherwise, they are similar to previously shown transcription rules (Agostino et al., 2011; Fuenzalida et al., 2015):

$$G(per_1) \rightarrow G(per_1) + R(per_1)$$

Only *PER*(1) and *PER*(2) agents are generated by those rules, as these are the proteins intended to be dependent on the circadian clock.

2.4.6 Perturbations

In this section, we discuss a final feature of PISKaS and other rule-based simulation environments: the possibility to incorporate perturbations. Perturbations correspond to changes in the status of the system attributed to external factors; in κ, they can be manifested as variations on the rate of certain rules, for instance.

In the current example, perturbations are implemented as if-then-else statements, which verify the timer of the simulation and assign values to certain variables accordingly:

if $0 \leq T \bmod 24 < 12$ **then**
    set $\lambda L \leftarrow 0.2$
else if $12 \leq T \bmod 24 < 24$ **then**
    set $\lambda_L \leftarrow 0.01$

Here, T is the timer of the simulation, thus "*T mod 24*" represents the value shown in a 24-h clock. Daylight corresponds to the time when the clock is between 0 and 12, while nighttime corresponds to the remaining values in this time frame. The variable λ_L is then assigned as a rate to the rules from the Light-dependent transcription section above. Other kinds of perturbation rules exist and may be applied to different contexts.

This model is a very simple example of an application of rule-based simulation of a biological process without explicit reference to the underlying rules of chemistry. It also shows a simulation in which there is a natural requirement for two compartments (as the nucleus and cytosol show vastly different phenomena) and how having an explicit framework for those compartments separation benefits the description of the model, as it allows for simpler expressions of the rules and other factors involved. It also shows how other features of the simulation environment, i.e., how the possibility of incorporating perturbations, allow us to describe many phenomena of interest.

As stated above, it is also possible to generate complex models of this phenomenon that incorporate more fine-grained spatial information regarding the cytosol, together with the mentioned two-environment situation. This allows for a much richer simulation but does not notably affect the expression of the rules, as we only need to distinguish nucleus and non-nucleus compartments. Thus, this gives us a model with good readability as the rules match closely the observed phenomena and we do not need to restate rules for each compartment. In addition, this model allows faster execution in a practical situation as the number of rules affects the execution time, and, for standard κ environments, this number scales at least linearly with the number of compartments involved. Analysis performed and described in (Fuenzalida et al., 2015) shows that the results in an environment that allows for explicit compartment declaration such as PISKaS do not suffer from severe losses of accuracy when the synchronization time is small, and thus follow the behavior of the observed biological system.

2.5 CONCLUSIONS AND OUTLOOK

In this chapter, we have described and introduced rule-based stochastic simulation. We highlighted the characteristics of this type of modeling and compared it with deterministic approaches widely employed in the modeling literature. In doing so, we explained several example models that will help the reader to understand the strengths and limitations of this approach. Simulation engines such as KaSim and its fork PISKaS to allow explicit spatial declaration are freely available in public repositories (KaSim can be obtained at github.com/Kappa-Dev/KaSim and PISKaS at github.com/DLab/PISKaS). The models describing the predator–prey ecosystem with explicit spatial information and the circadian clock for both simulation engines can be obtained at github.com/DLab/models.

2.6 NOMENCLATURE

Term	Definition
Agent	Abstract representations of entities on a system. An agent can bind each other´s agents through its *sites*. Optionally, each *site* could harbor a *state*, a label that recapitulates a feature of the mentioned *site*, or a numeric property of the agent.
Bond	A representation of binding between two *sites* of two different agents.
Compartment	A declaration that represents a physical or logical space or volume, which is part of a system.
Rule	Chemical equations that represent elemental reactions where reactants are *agents* with a set of features necessary and sufficient for a transformation to occur (*Left-Hand Side*) and the resulting pattern for each participating *agent* (*Right-Hand Side*). κ *rules* declare reactions that change the value of a *site*, create or destroy *bonds* between *agents*, and create or remove *agents* on the modeled system.
Site	An abstract representation of a physical or logical interface where an *agent* binds another *agent* or where different *states* are declared.
Specie	Each of the individual instances of an *agent*.
State	An abstract representation of a qualitative or quantitative characteristic that recapitulates a feature of the declared *site*.
Transport	A declaration that states the *link* that uses an *agent* to travel from one compartment to another. Additionally, it may declare the frequency and time employed to move the agent between compartments.

3. CHAPTER II: STOCHASTIC SIMULATION OF MULTISCALE COMPLEX SYSTEMS WITH PISKAS: A RULE-BASED APPROACH

Tomas Perez-Acle[1,2], Ignacio Fuenzalida[1], Alberto J.M. Martin[1,2], **Rodrigo Santibañez**[1,3], Rodrigo Avaria[1], Alejandro Bernardin[1,2], Alvaro M. Bustos[1], Daniel Garrido[3], Jonathan Dushoff[4,5], James H. Liu[6]

[1]Computational Biology Lab, Fundacion Ciencia & Vida, Santiago, Chile

[2]Centro Interdisciplinario de Neurociencia de Valparaiso, Universidad de Valparaíso, Valparaíso, Chile

[3]Department of Chemical and Bioprocess Engineering, School of Engineering, Pontificia Universidad Catolica de Chile, Santiago, Chile

[4]Department of Biology, McMaster University, Hamilton, ON, Canada

[5]Institute of Infectious Disease Research, McMaster University, Hamilton, ON, Canada

[6]Massey University, Palmerston North, New Zealand

Published in *Biochemical and Biophysical Research Communications* (2018). Vol. 498, Issue 2, Pages 342-351

3.1 SUMMARY

Computational simulation is a widely employed methodology to study the dynamic behavior of complex systems. Although common approaches are based on either ordinary differential equations or stochastic differential equations, these techniques make several assumptions that, when it comes to biological processes, could often lead to unrealistic models. Among others, model approaches based on differential equations entangle kinetics and causality, failing when complexity increases, separating knowledge from models and assuming that the average behavior of the population encompasses any individual deviation. To overcome these limitations, simulations based on the Stochastic Simulation Algorithm (SSA) appear as a suitable approach to model complex biological systems. In this work, we review three different models executed in PISKaS: a rule-based framework to produce multiscale stochastic simulations of complex systems. These models span multiple time and spatial scales ranging from gene regulation up to Game Theory. In the first example, we describe a

model of the core regulatory network of gene expression in *Escherichia coli* highlighting the continuous model improvement capacities of PISKaS. The second example describes a hypothetical outbreak of the Ebola virus occurring in a compartmentalized environment resembling cities and highways. Finally, in the last example, we illustrate a stochastic model for the prisoner´s dilemma; a common approach from social sciences describing complex interactions involving trust within human populations. As a whole, these models demonstrate the capabilities of PISKaS providing fertile scenarios were to explore the dynamics of complex systems.

3.2 INTRODUCTION

Complex Systems (CSs) encompass a variety of phenomena where the interaction between constituent elements produces emergent properties. Among other characteristics, CSs exhibit degeneracy being highly robust to random failures (Kitano, 2002). Such systems are ubiquitous in nature and society and, when it comes to their analysis, they are usually represented as networks. In these networks, edges represent the interactions occurring between the entities composing the system, which are typically depicted as nodes. Nevertheless, networks are static pictures of CSs: they disregard its dynamic behavior precluding the study of some of its fundamental properties such as evolvability (Mayer & Hansen, 2017). Among other methods, two main approaches for the dynamic modeling of CSs prevail: deterministic methods based on ordinary differential equations (ODEs) or agent-based modeling, and stochastic approaches based either on stochastic differential equations (SDEs) or on the Stochastic Simulation Algorithm (SSA).

Approaches based on ODEs and SDEs require the definition of an equation for each different reaction or interaction in the model. Despite their broad applicability (Fisher & Henzinger, 2007), ODEs models rely on several unrealistic assumptions. For instance, ODEs require a homogeneous distribution of components, and furthermore, a continuous distribution of interactions and quantities that, in real systems, are of discrete nature. On the other hand, SDEs assume that the stochasticity of the systems can be modeled as a source of noise acting as a modulator of the average dynamic of the population. In contrast, ODE models assume a deterministic behavior, focusing on the mean distribution of the system

under study. On top of this, ODE- and SDE-based models usually entangle kinetics and causality. This characteristic excels when unrelated simultaneous processes produce an apparent causality. Last but not least, approaches based on differential equations require as many equations as variables, hindering continuous model improvement, and therefore separating domain knowledge from the models.

Although both ODEs and SDEs are well-established methods, SSA-based methods are gaining momentum among other approaches as both reliable and versatile strategies to model a variety of CSs (Barrio et al., 2006). An SSA model occurs in a reactor where a set of species or agents interact by a set of reactions or rules. While the reactor is a well-mixed and homogeneous environment containing the initial quantities of every agent, the application of the set of rules will produce the dynamic of the system. Within the reactor, stochastically selected rules are applied as discrete events generating trajectories through time to produce solutions representing feasible temporal paths of the system. Importantly, as time depends on the execution of rules, time intervals are asymmetric over the trajectory of the system and between parallel reactors. Current variants of SSA use their own language to describe the systems under simulation. Some examples are the BioNetGen language (James R. Faeder et al., 2009) and the Kappa language (Murphy et al., 2010). Importantly, for each one of these languages, a proper simulation engine should be used: NFSim (Chylek et al., 2015) executes models written in BioNetGen and KaSim (Boutillier, Feret, et al., 2018) executes models written in Kappa. Several simulation engines based on the SSA have been developed. While some implementations account for a large number of species (Danos et al., 2007b; James R. Faeder et al., 2009; Gibson & Bruck, 2000; Lok & Brent, 2005), some others deal with systems formed by numerous particles (Cao et al., 2006; D. T. Gillespie, 2001), or with slow-scale systems (Cao et al., 2005; Rathinam et al., 2003), and some others were developed to add computational power to the simulation (Dematté & Prandi, 2010; Dittamo & Cangelosi, 2009).

Despite the increasing relevance of SSA and the expanded features included in several of its implementations, these fail to deal properly with the combinatorial diversity and spatial heterogeneity of biological systems. To overcome these issues, we developed PISKaS, a parallel implementation of a spatial Kappa simulator (Fuenzalida et al., 2015;

Fuenzalida, Bustos, et al., 2017; Fuenzalida, Martin, et al., 2017; Nuñez et al., 2012). PISKaS is a multiscale simulation tool suitable to perform stochastic simulations on distributed memory computing architectures. PISKaS expands the Kappa language by allowing the explicit declaration of compartments interconnected by links to simulate heterogeneous environments. In PISKaS, these links account for different types of transport between compartments. Each PISKaS compartment is executed in parallel, running its own SSA. The execution of a communication rule between compartments is treated as a perturbation modifying the number of agents in both the source and the destination compartment. Due to these new features, Kappa models are easier and more compact to write and to understand than those employed by other implementations. While the original motivation for both the SSA and the Kappa language comes from the modeling of chemical reactions, PISKaS can be used to model systems unrelated to chemistry, or where the species or agents involved do not represent chemical units but instead more complex entities (v.g. genes, individuals or communities).

In the following pages, three models of CSs will be used to demonstrate the versatility and capabilities of PISKaS to deal with multiscale models. In doing so, we present a partial model for the transcriptional regulation of *Escherichia coli*; a highly compartmentalized model to study the spread of Ebola virus disease; and a well-known model from Game Theory (GT): the Prisoner´s dilemma (PD) which is aimed at the study of cooperation and competition between individuals in a society. While in the *E. coli* model we exploit PISKaS capabilities to deal with continuous model improvement, on the Ebola model we rely on a highly compartmentalized environment, and in the PD, we follow an approach where rules operate considering the value of certain agent properties. As a whole, we propose PISKaS as a suitable tool to produce stochastic simulations of multiscale CSs using a rule-based approach

3.3 METHODS

PISKaS was developed from a forked branch of the Kappa language simulator, KaSim v3.5 (Boutillier, Feret, et al., 2018), adding new features to improve performance by implementing parallelism, and to allow spatial declaration of the model environment by

using compartments and links. KaSim is written in Ocaml (http://www.ocaml.org), a programming language following both the functional and object-oriented paradigms. PISKaS employs OcamlMPI 1.01 (https://forge.ocamlcore.org/ projects/ocamlmpi/), taking advantage of the MPI library to implement the parallel execution of compartments. To execute a simulation, PISKaS starts an iterative process determining the probability that the next event on the simulation occurs at time τ, selecting the rule R_j, given the state of the system x in time t (Equation 3-1):

$$p(\tau, R_j | x, t) = a_j(x) e^{-a_0(x)\tau} \tag{3-1}$$

where $\tau = t + \Delta t$, a_0 is the total reactivity of the system and a_j is the reactivity of rule j. Importantly, τ is calculated from a random number r_1 distributing uniformly between 0 and 1 according to Equation 3-2:

$$\tau = \frac{1}{a_0(x)} \ln \frac{1}{r_1} \tag{3-2}$$

By doing so, each temporal step over the simulation randomly selects a single rule that creates, destroys, binds, unbinds agents, or modifies some property of an agent: the five primitives of the Kappa language (Murphy et al., 2010). Importantly, PISKaS proposes a new algorithm to deal with the heterogeneous spatiality of CSs. This modified version of the traditional SSA partitions the initial state of the system into several compartments. Each compartment is considered as a homogeneous volume following the SSA fundamental assumption; reactions operating over agents are considered as collisions between particles and, no collisions occur between agents belonging to different compartments. As a consequence, every compartment in the model executes its own SSA. Transport between compartments is produced by rules executing perturbations to modify the number of agents in both linked compartments: whereas a transport rule removes an agent from a compartment, it will then create that agent in the linked compartment. To produce the system dynamics, PISKaS implements a parallel algorithm that supports, at the cost of some accuracy that is tunable by the modeler, the scalability of the model by adding compartments, links, and transport rules. These features provide a more accurate physical description of the

environment, allowing the execution of millions of agents in thousands of compartments, depending on hardware capabilities.

The performance of PISKaS implementation depends on several parameters of the model. The main parameter regulating performance is the synchronization step h, i.e. the elapsed time between synchronization processes occurring in different compartments. A small value for h produces a slower but more accurate simulation compare to that of a larger h. Unfortunately, finding the optimum h value requires a trial-and-error approach because it is tightly linked to both the spatial complexity of the model and its rules. Nevertheless, our divide-and-conquer approach allows creating orthogonal reactors in the simulation where each SSA runs in parallel, using its own agents and rules.

To deal with the problem of finding a proper h parameter, at least 1000 simulations were run for every model varying the value of h, considering 10 time-intervals equally distributed, starting from the smallest time ticking between all linked compartments, up to the largest one. To do so, an initial simulation was run for every model, monitoring the behavior of time ticking in all different compartments. To compute statistics, trajectory ensembles were obtained by averaging over time and calculating the standard deviation. Importantly, as the SSA can be defined as a state algorithm dedicated to solving numerically a continuous first-order Markov chain, there is no need to use a more complex process such as the block average, to compute the error over the average trajectory of the system.

All simulations were run in a distributed memory cluster composed of two computing nodes of 64 cores with 128 GB RAM interconnected by a Mellanox Infiniband switch. Each node has 4 AMD Opteron 6376 processors running at 2.3 GHz.

The development of PISKaS is under the GNU license and can be downloaded from GitHub at https://github.com/DLab/PISKaS/wiki. All models used in this work can be downloaded from GitHub at https://github.com/DLab/PISKaS/wiki/Models.

3.4 RESULTS: APPLICATION CASES

3.4.1 Simulation of transcriptional networks in *E. coli*

The regulation of gene expression occurring in living organisms is an epitome of biological CSs. Even though it can be argued that gene regulation is an interconnected process highly

dependent on gene expression, whole-genome control is usually modeled by dividing it into more or less independent processes (Lim et al., 2013; Nishimura et al., 2015). Partitioning has two main purposes; to facilitate knowledge generation and to simplify parameter optimization. To demonstrate PISKaS capabilities to support continuous improvement of models, two systems were studied: the core gene regulatory network of *E. coli* and the replication of the ColEI plasmid. Both models were subsequently combined into a more detailed system, taking advantage of the natural compartmentalization of biological systems (Figure 3-1A). By using the gene regulatory network (GRN) of *E. coli* MG1655 and a custom-made script written with the Python PySB library (Lopez et al., 2013), all Kappa models were written automatically following the Kappa Biobrick Framework (Stewart & Wilson-Kanamori, 2011). The GRN was created combining literature (Cho et al., 2014) and the EcoCyc database (Keseler et al., 2017) in a format that permitted the creation of models where agents, rules, parameters, and the initial condition of the system were predefined. Specific rules created and employed in all three models are described using a graphical notation in Figure S 3-1.

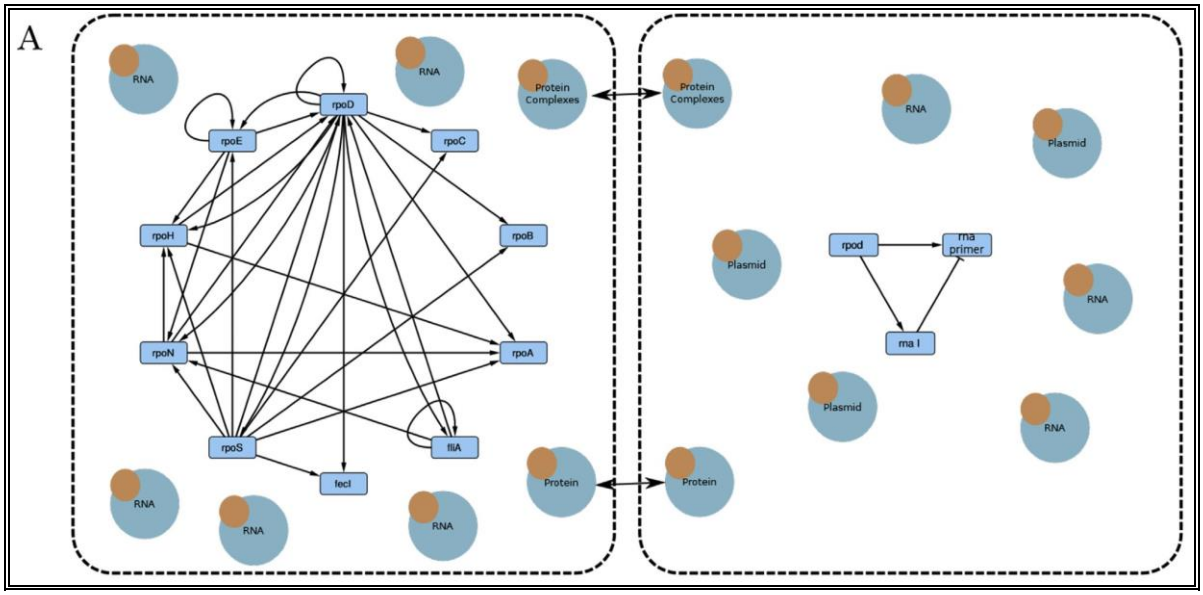


Figure 3-1. The gene regulatory network simulated in both the core regulatory network and in the ColEI replication models.
(A) The regulation of transcription (upper-left) and plasmid replication (upper-right) were compartmentalized and allowed to share free proteins and protein complexes (arrows).

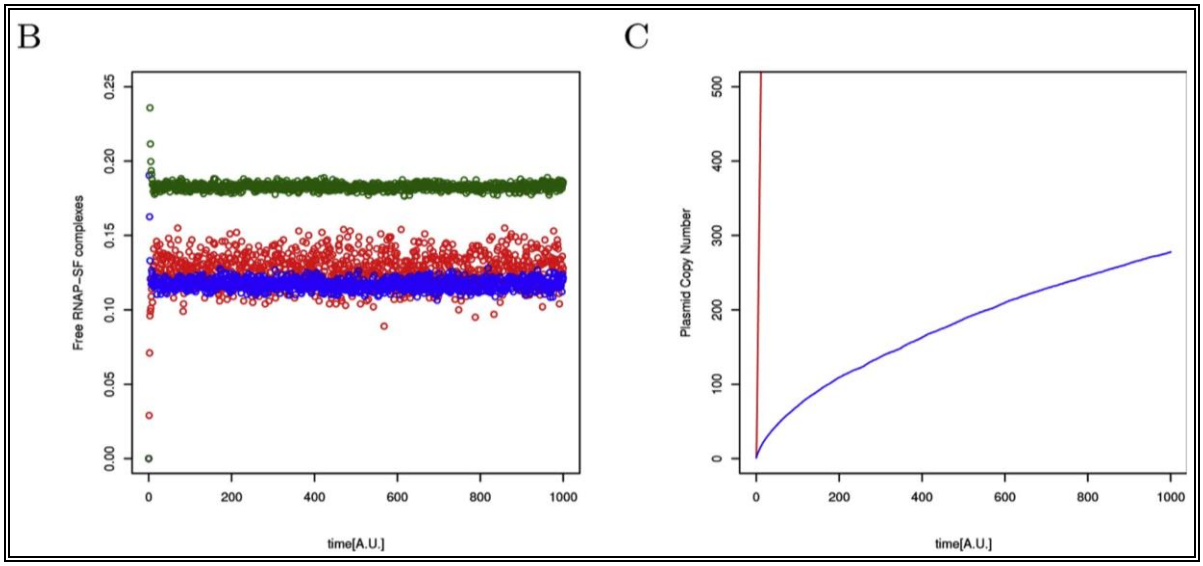


Figure 3-2. The gene regulatory network simulated in both the core regulatory network and in the ColEI replication models (continued).
The outcome of the simulation of transcriptional networks in *E. coli*. (B) Presence of RNAP-SF complexes in the three settings of the core regulatory network model: Red dots represent the percentage of free RNAP-SF complexes, while blue and green represent one and seven RNAP-SF complexes, respectively. (C) Two simulations of ColEI replication model: red line represents the unconstrained situation where RNA I and RNA primer show no interaction, and the blue line indicates RNA interaction inhibits replication.

3.4.1.1 Core regulatory network: Sigma factors

The core regulatory network in *E. coli* is controlled by seven proteins called Sigma Factors (SFs) (Cho et al., 2014). An SF binds to the RNA Polymerase (RNAP) core enzyme giving specificity and causing competition for different transcription factor binding sites (Mauri & Klumpp, 2014). This model is composed of 10 agents representing genes, three of them encoding for each RNAP subunits, and the remaining seven agents encoding for each SFs. The assembly of every possible complex RNAP-SF was modeled as a reversible process as reported previously (Ishihama, 1981; Mauri & Klumpp, 2014). Gene regulation conducted by the binding of each RNAP-SF complex to each of the promoters was also modeled as a reversible process, whereas transcription and termination were irreversible events. Our model also accounted for the transcription units present in this system. Specifically, rpoC is

the only unit transcribed right after rpoB, and both genes coding for RNAP subunits are regulated by the same RNAP-SF.

Three different settings were explored with 1000 simulations each: in the first setting, only one RNAP could be assembled, while seven were possible in the second and 28 in the third setting: one RNAP for each non-redundant regulation according to Figure 3-1. Moreover, in both the first and second settings, only one copy of each SF was available, while in the third setting, there was a copy of every regulation present in the GRN. The setting with a higher number of SFs resulted in better stability of the number of RNAPSF complexes, showing an average slightly lower than 20% (Figure 3-1B). Notably, having a higher number of copies of the SFs, this setting resulted in the closest one to previously reported experimental data (Patrick et al., 2015) where the fraction of free RNAP-SF complexes was 30%, rendering a more realistic version of the model. The fraction of free RNAP-SF is regulated by additional layers of regulatory elements and the rate of interaction between components. Several antagonists regulate the *E. coli* SF activity. Of note is the rsd protein that binds rpoD and controls the transition to a stationary state (Mitchell et al., 2007). Other Sigma antagonists are the FlgM that binds FliA and controls flagellar synthesis, and the rseAB protein that regulates rpoE activity, necessary for stress response. Eleven components have sigma factor antagonist activity and anti-sigma factor antagonist activity in *E. coli*. Therefore, the incorporation of the aforementioned elements in the core regulatory model could reduce the notorious distance between simulated and experimental values, prior to parameter adjustment.

3.4.1.2 ColEI replication model

The regulatory network controlling the replication of the ColEI plasmid is formed by two non-coding RNAs (Polisky, 1988). The first is an RNA that binds tightly to the DNA, making a hybrid DNA-RNA that after processed by RNase E serves as a primer for the DNA Polymerase (A. Robinson & Van Oijen, 2013). To control its replication, this plasmid encodes an antisense RNA, called RNA I, which binds the 50 ending nucleotides of the RNA primer, destabilizing the hybrid and effectively inhibiting DNA replication. To explore further PISKaS capabilities, our ColEI replication model simulates the formation of an

RNA-RNA pairing to inhibit DNA replication (Figure 3-1C). Even though arbitrary rates were employed, a pseudo-stationary state was reached with near 300 plasmid copies. On the other hand, without the RNA-RNA interaction, the plasmid reached approximately 48,000 copies over the simulation, which is far beyond the numbers obtained from the model with both non-coding RNAs.

3.4.1.3 Combined model

Given that, both previous models share common components (SFs, RNAP, and RNase E), a more realistic setting was developed by joining them, demonstrating the continuous improvement strategy supported by PISKaS. During the combined simulation, despite the explicit communication between compartments, each type of agent behaves in a very similar way to that observed in the separated simulations. However, the dynamic of the plasmid replication was heavily impacted; reaching only 20 copies of the plasmid instead of the 200 obtained in the separated model, with some simulations stalled after only one copy was produced.

3.4.2 A hypothetical outbreak of the Ebola virus

Other interesting models of CSs are those employed in the study of the spread of infectious diseases. Of note, the accurate modeling of infectious diseases leads to several significant consequences such as predictions on the impact of an outbreak over the population; determining the number of infected, exposed, and total cases. Moreover, a proper model could lead to identifying the rate of disease growth, and to the definition of the most effective policies aimed at avoiding the spread of the disease, such as vaccination strategies. One of the earliest mathematical models ever developed to study the propagation of an infectious disease was the SIR model developed by Kermack and McKendrick in 1927 (Kermack & McKendrick, 1927). In this seminal work, the authors defined a compartmentalized ODE model where people could be categorized into three groups. As a consequence, the SIR model defines susceptibles (S), i.e. those who can be infected; infected (I), i.e. those who have been infected; and removed (R), those who cannot further participate in the dynamics either by acquiring immunity or by being dead. This simple but effective model predicts the number of infected people over time by finding both the basic reproductive number R_0, i.e.

the number of individuals that each infected person could infect over time, and the exponential growth rate of the disease r_0, i.e. a number defining how quickly the growth of the disease follows the exponential trend.

Despite its broad applicability (Hethcote, 2000; Newman, 2002; Shulgin et al., 1998), this model lacks important features of some contagious diseases such as the transmission of viruses from dead people, as was recently observed for the Ebola virus (Aylward et al., 2014). A model explicitly dealing with this type of transmission was recently developed (Weitz & Dushoff, 2015). In this SEIRD model, individuals are classified into susceptible (S), exposed (E), infected (I), removed (R), and dead (D) (Figure 3-2A). To assess the capabilities of PISKaS to deal with such compartmentalized models, we developed a SEIRD model to simulate an outbreak of the Ebola disease in a hypothetical country composed by four cities (Figure 3-2B), named C1 to C4, and having a bidirectional highway system interconnecting them. In doing so, we produced a country-level model by including compartments resembling four large cities with population densities of 9200, 3912, 3867, and 3958 habitants per km^2 for C1, C2, C3 and C4, respectively. Moreover, our model allows the free travel of agents between cities from all S, E and I states. This is particularly important when considering that both E and I agents could spread the disease by travelling along the hypothetical country. Rates of exposition from infected people (β_i), exposition from dead people (β_D), the time needed to transit from the exposed to the infected state (T_E), and the time (T_I) in which infected people remain until the application of the rate of either death (f) or removal ($1 - f$), were all obtained from the ODEs adjustment of the SEIRD model to actual data coming from the last Ebola outbreak in Africa, as proposed in (Weitz & Dushoff, 2015). Kappa rules employed to generate the set of stochastic SEIRD simulations are shown in Figure S 3-2.

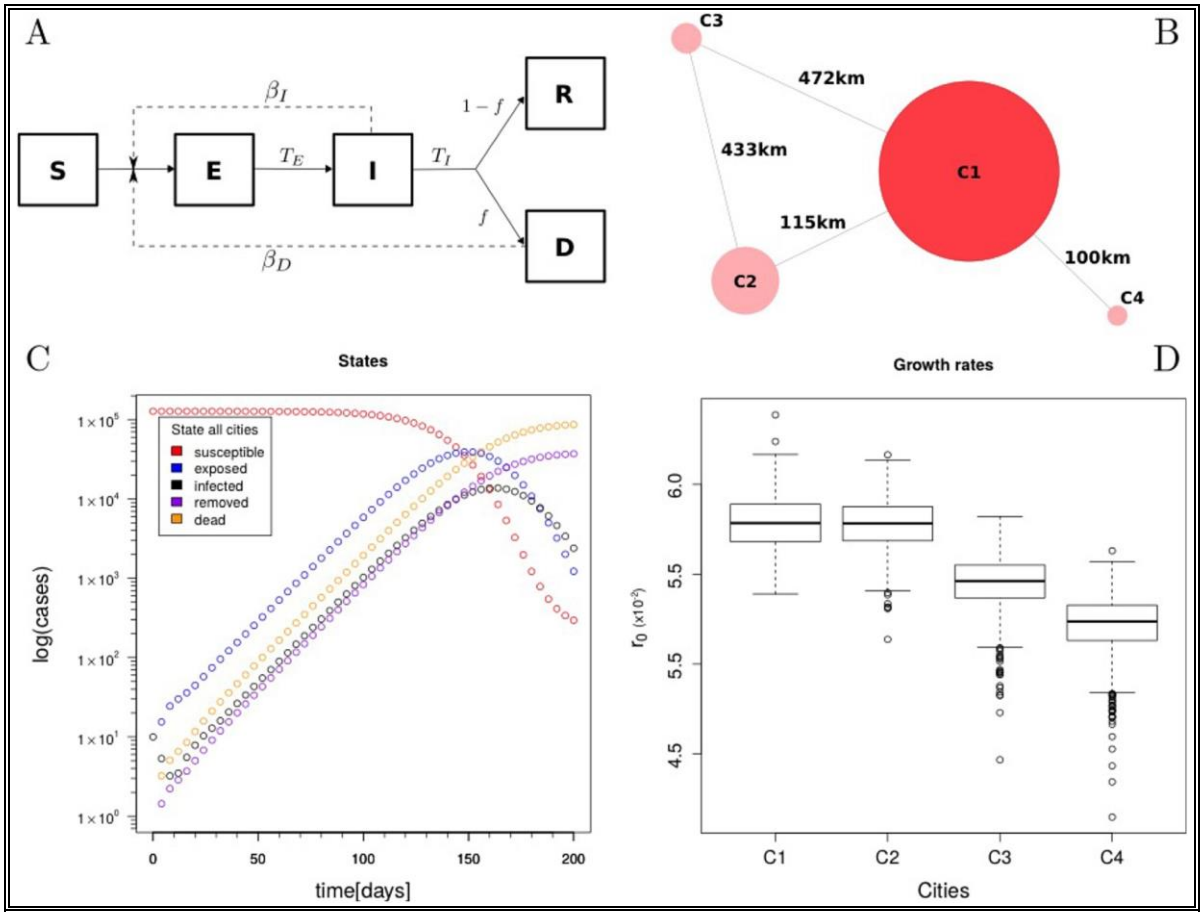


Figure 3-3. A hypothetical outbreak of the Ebola virus.
(A) Graphical representations of the SEIRD model. Black boxes describe each state that can be achieved by any agent of the simulation: Susceptible, Exposed, Infected, Removed and Dead. Solid arrows represent how these states evolve over time, while dashed arrows indicate the transmission of the virus. (B) The topology of connectivity between cities of our model. The size of each city is proportional to the population density normalized by the population density of C1. (C) and (D), results from the simulations of the spread of Ebola. (C), averaged occurrence of the five different states for agents participating in the SEIRD model using a log scale. (D), exponential growth rates calculated for each city in the model according to the 1000 independent simulations. Data is shown using a box-plot where the central black line is the average, the limits of the boxes represent the standard deviation and the whiskers represent the variance.

As an initial condition, ten infected persons appeared in city C1 on day 0. Of note, this relatively large number of initially infected people ensured the expansion of the disease since the disease did not spread in the majority of the simulations initialized with fewer infected individuals (data not shown). These simulations were employed to characterize the

dynamic properties of the disease spreading in each city, including the exponential growth rate r_0 averaged after 1000 simulations. City-level r_0 (Figure 3-2D) were calculated using Equation 3-3, where $I(t)$ is the number of infected people at time t.

$$I(t) = \exp(r_0 t) \tag{3-3}$$

As expected, the average trajectory over an ensemble of 1000 stochastic simulations shows the typical exponential spread of the Ebola disease (Figure 3-2C). It is worth noting that in the first day of the disease spread, the number of infected people decreases due to two factors: first, the transit of infected people to either the dead or the removed states, and because of the time that people spend in the susceptible state, around 6 days after the exposition to the virus. This can be noted in Figure 3-2C, by considering that people of types R, D, and E, appear around day 6 from the simulation start. Previously to this day, only infected and susceptible people are present in the simulation. When considering the growth rates for different cities (Figure 3-2D), our results indicate that in each city the disease spreads at different rates. This behavior could be the consequence of both differences between the population density of each city, and the highway connectivity between cities. Of note, while it is expected that city C1 being the largest city will exhibit the highest r_0 (Figure 3-2D), it is interesting that the second highest r_0 belongs to city C2 instead of city C4, which is the closest city to the origin of the disease (city C1). Importantly, population densities of cities C2, C3, and C4 are similar therefore, the important parameter influencing r_0 in city C2 seems to be the topology of connectivity. Despite the degree of connectivity between cities C2, and C3 is the same ($D = 2$), the larger value of r_0 for C2 must be influenced by the close proximity of C2 to the origin of the infection. Notably, city C4 is the closest city to the origin of the infection and is the smallest city in the model (Figure 3-2D). On the contrary to what we expected, r_0 for city C4 is the smallest one, denoting the importance of the degree of connectivity for the spread of the disease. As a whole, our data indicate that both proximity and degree of connectivity should be key factors to consider when developing contingency policies to avoid the spread of contagious diseases such as Ebola.

3.4.3 Exploring human collaboration using the Prisoner´s dilemma

Social Dilemmas (SDs) encompass different situations in which people may cooperate or defect, thus obtaining a personal benefit and producing a detriment to their counterpart, represented by other individuals, and by extension to society at large (Capraro, 2013). In general, SDs include a variety of problems spanning economics, political sciences, anthropology, and cultural evolution. To formally treat these SDs, mathematical approaches from Game Theory (GT) provide a theoretical framework suitable to produce computational models such as the Prisoner´s Dilemma (PD) or the investor´s game, among others (Veloz et al., 2014). These games have guided quantitative research on social sciences for decades, predicting under which circumstances cooperation or defection may surge, providing a rich scaffold to understand the evolution of collaboration in human societies (El-Seidy et al., 2016; Nowak & Sigmund, 1990; Rogers & Ashlock, 2010; Veloz et al., 2014). In each matrix-based game, such as the PD, there are two players: each player, or agent, has two possible moves or actions interpreted as Cooperation (C) or Defection (D), resulting in four possible outcomes of the game (Table 3-1). The possible outcomes from this game are 1) reward from mutual cooperation (R); 2) punishment arising from mutual defection (P); 3) suckers outcome, obtained by the player who cooperates against a defecting partner (S); and 4) temptation outcome, achieved by defecting against a cooperating partner (T).

In its simplest form, a PD is a matrix-based game described by the following relations (Equations 3-4 and 3-5):

$$T > R > P > S \tag{3-4}$$

$$2R > T + S \tag{3-5}$$

where Equation 3-4 ensures that exploiting a cooperating partner is preferred over mutual cooperation, which is preferred to mutual defection, and the latter is preferred over being exploited. Importantly, Equation 3-5 ensures that mutual cooperation is preferred to both unilateral cooperation and defection (Capraro, 2013; Kuhn, 2019; Veloz et al., 2014).

Despite GT provides a versatile theoretical framework to tackle several SDs, usual approaches do not consider the relationship between trust among agents participating in the game and the probability of cooperation or defection. Of note, the creation and production

of social capital in human societies depends on the level of trust between people belonging to the same social class (i.e. intra-class trust), and also between people belonging to different social classes (i.e. inter-class trust) (Evers, 2001). Moreover, societies with higher trust levels usually exhibit higher social capital and, as a consequence, a higher tendency to collaborate (Fukuyama, 1995). Therefore, we decided to integrate trust as part of the social variables belonging to the agents. In doing so, the economic prosperity of a society, expressed in the form of gross domestic product (GDP), will depend on the creation and development of social capital, which in our simulation is in direct relationship with both intra- and inter-class trust.

To explore PISKaS capabilities to produce a model where the operation of rules depends on the value of certain properties belonging to the agents, we produced a PD where the probability of defection or cooperation will depend on the level of trust held among the agents. Consequently, reward or punishment will be received by the agents of the simulation according to the pay-off matrix of the PDG (Table 3-1). This behavior will increase or decrease individual wealth over the simulation and summing over the agents of the simulation, which will produce the GDP of the society.

Table 3-1. The pay-off matrix used for our simulation of the PD.
Temptation (T) produces +2 units of goods, reward (R) produces +1 units, punishment (P) produces –1, and sucker´s pay-off produces –2 units.

		Player 1	
		C	D
Player 2	C	+1 (R), +1 (R)	–2 (S), +2 (T)
	D	+2 (T), –2 (S)	–1 (P), –1 (P)

To define the players participating in the PD, our agents have the following interface: $\mathrm{Person}(X, T{\sim}1{\sim}2{\sim}3{\sim}4, W)$, where *Person* is the name of the agent, X corresponds to the interaction site between agents, T defines the level of trust, and W represents a wallet where every agent accumulates the reward obtained from the last interaction, according to the pay-off matrix. To explore the hypothesis of whether the size of the population is directly related to the creation of GDP, we decided to model two isolated cities, City 1 and City 2, with 800

and 1000 agents, representing the urban and rural population of New Zealand, respectively. The distribution of trust for both cities was obtained from the trust profiles of the urban population of New Zealand, as revealed by the "The Global Trust Inventory" (J. H. Liu et al., 2018). In brief, the trust profile of the population is composed by a 28.3% of High Trust (HT), 46.4% Moderate Trust (MT), 19.0% Low Institutional Trust (LIT), and 6.2% Low Trust (LT). Therefore, the number of agents belonging to each class of trust was obtained by multiplying the number of total agents by the trust distribution. Kappa rules to run the simulation were written for each trust profile (Figure S 3-3) describing how players engage in interactions with other players during the PD games. Importantly, independent of the level of trust, all agents have the same probability of interaction over the simulation. However, as the number of agents corresponding to each trust profile is different, generally speaking, the probability of interactions will be higher for those larger groups compared to that of the smaller groups.

To produce an ensemble trajectory of our PD model, each simulation was run for 200 arbitrary units (A.U.) of time, and their results were averaged between 1000 independent simulations. Results shown in Figure 3-3 correspond to a model composed of two isolated compartments, named City 1 (Figure 3-3A, 3-3B, 3-3E, and 3-3F) and City 2 (Figure 3-3C, 3-3D, 3-3G, and 3-3H). The normalized social wealth (GDP) was calculated by summing up to over the W site of the individual agents and dividing by the number of agents (Figure 3-3A and 3-3C). We also computed the normalized contribution to the GDP of each trust profile, as shown in Figure 3-3B and 3-3D. The percentage of different outcomes coming from the interactions between trust profiles is shown in Figure 3-3E and 3-3H. In general, the average normalized GDP generated over the simulation by City 2 is higher compared to that of City 1. Moreover, the slope of the curve showing the creation of the GDP over time is higher in City 2 compared to that of City 1.

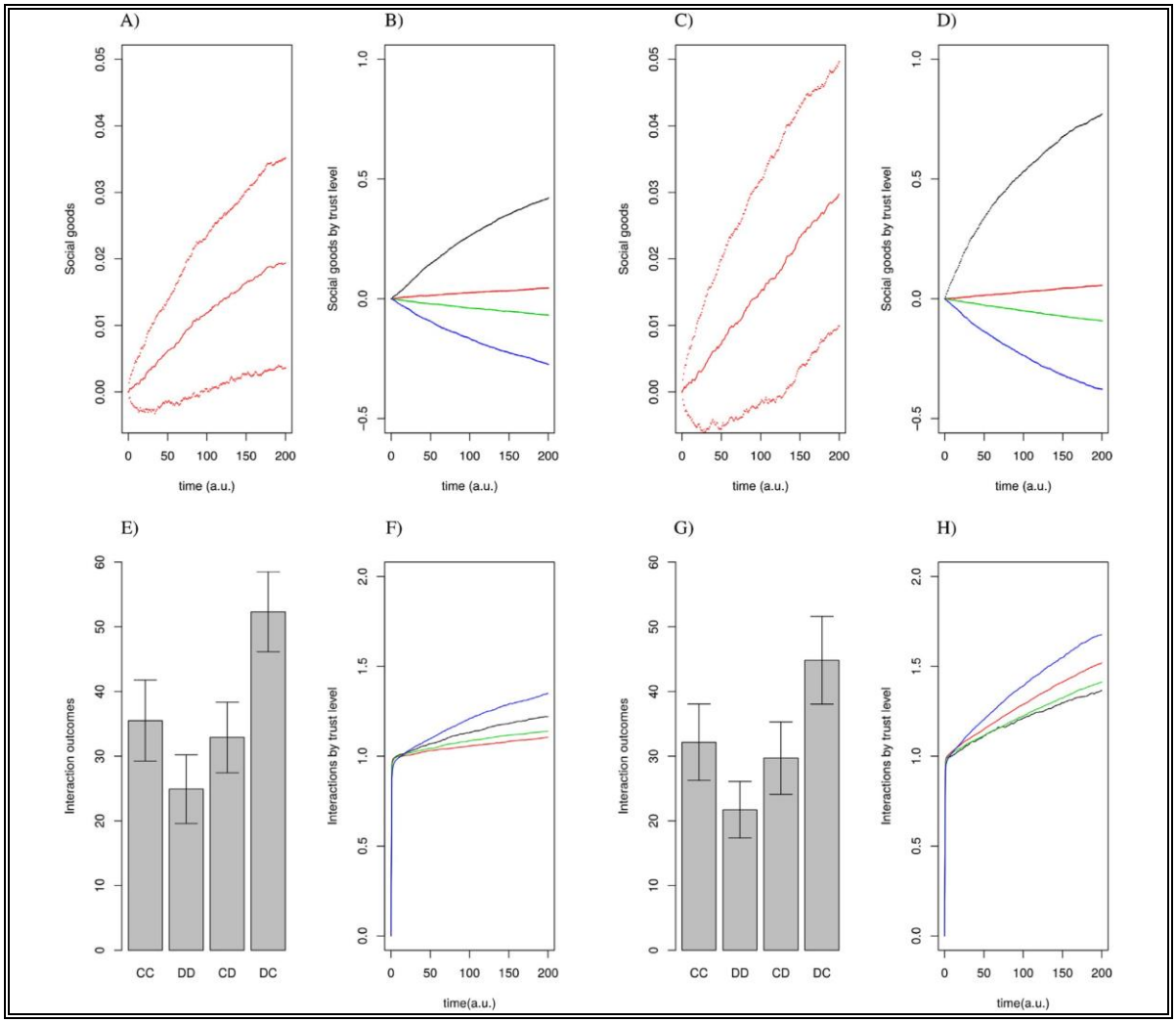


Figure 3-4. Results of the PD model.

The outcome of 1000 independent simulations for City 1 and for City 2 are presented in panels A, B, E, and F; and panels C, D, G, and H, respectively. The accumulation of social goods as a measurement of the average normalized GDP (solid red line) for City 1 and City 2 is shown in panels A and C, respectively. Dotted lines represent the standard deviation of the GDP over 1000 simulations. The normalized individual accumulation of goods for agents belonging to different trust profiles of City 1 and City 2, is presented in panels B and D, respectively. The total outcome of the PD interactions between agents from City 1 and City 2 over the simulation is shown using percentages in panels E and G, respectively. The four possible outcomes of the PD matrix-game are denoted as CC (cooperate-cooperate), DD (defect-defect), CD (cooperate-defect) and DC (defect-cooperate). Total interactions from the PD occurring between agents of the same trust profile in City 1 and City 2 are shown in panels F and H, respectively. For panels B, D, F, and H, the black line represents HT, the red line represents MT, the green line represents LIT and the blue line represents LT.

When we consider the standard deviation (Figure 3-3A and 3-3C) generated by 1000 independent simulations (see methods), both GDP curves seem to be generated from the same distribution. Therefore, no significant differences can be observed between both curves. However, a slightly significant difference between the standard deviation of the GDP curve of City 2 compared to that of City 1, at both the lower and higher extremes, can be seen in Figure 3-3A and 3-3C. This behavior implies that at the beginning of some simulations –between 0 and 80 a. u. of time– City 2 will generate a lower GDP. On the other hand, for some other simulations, City 2 will generate a higher GDP at the end of the simulation, between 110 and 220 A.U. of time. Consequently, these differences could be explained by the accumulation of individual wealth of the population.

To explore further the role of wealth accumulation by different trust profiles over the creation of GDP, we calculated the distribution of wealth by trust level, normalized by the number of agents of each profile, as seen in Figure 3-3B and 3-3D. As a whole, the average individual accumulation of wealth by both MT and LIT is similar when comparing both cities. When considering the average individual accumulation of wealth for both HT and LT at the end of the simulation, City 2 exhibits both the largest and the lowest values compared to that of City 1. This behavior could explain the differences shown in Figure 3-3A and 3-3C denoting the importance of individual wealth accumulation. Despite a higher GDP for City 2 appears as the obvious result –considering that City 1 has a 20% lower number of agents–, unexpectedly, the accumulation of individual wealth per trust profile in City 2 did not follow a proportional increase. Notably, only the accumulation of individual wealth of both HT and LT are the ones that seem to be affected by the number of agents. To determine whether this effect is generated by the interactions between agents, we calculated the percentage of interactions coming from the PD, as seen in Figure 3-3E and 3-3G. As a whole, the percentage of interactions of types cooperate-cooperate (CC), defect-defect (DD), and cooperate-defect (CD) from both cities is similar, with higher differences shown for interactions of type defect-cooperate (DC), which is significantly higher in City 1 compared to that of City 2. When considering the number of interactions normalized per trust profile (Figure 3-3F and 3-3H), is interesting to note that the total number of interactions for every trust profile in City 2 is higher than that of City 1. Moreover, the ranking of interactions

between trust profiles is different when comparing both cities: while in both cities the profile showing the largest interaction is LT, in City 2 the profile showing the lowest interaction is HT, whereas in City 1 the lowest one is MT. As the probability of interactions between all profiles is the same (see above), the most important factor influencing the increase of the GDP of City 2 compared to that of City 1, is the number of interactions occurring in the larger profiles named MT and LIT which in City 2 appear at the middle of the interaction ranking. In contrast, these two larger groups appear at the bottom of the interactions ranking in City 1. Therefore, the higher GDP of City 2 compared to that of City 1 could be the result of an elevated number of interactions of the two larger trust profiles of the society.

3.5 DISCUSSION

Through the description of three multiscale examples representing CSs from different domains of knowledge, we have explored the versatility of PISKaS. Despite we have explored different features from our models, an exhaustive parameter exploration, including the topology of connectivity between compartments, the sensibility of the models to changes on rates of rules, adjustment of rates to real data, among other, are all efforts we are currently executing and whose results will be published elsewhere. Of note, PISKaS could scale-up compared to the original implementation by adding distributed memory in computational resources and, in doing so, allows for the definition of more realistic models with increased complexity. Due to the explicit declaration and usage of compartments, models in PISKaS overcome the limitation of running the SSA only on homogeneously mixed volumes, using a divide-and-conquer approach to speed-up the calculations.

As denoted by the simulation of the transcriptional networks of *E. coli*, non-compartmentalized approaches to conduct computational simulations may render unrealistic or over-simplified results: a limitation that is overcome by the declaration of compartments in PISKaS. It is also well known that transcription and other biological processes at the molecular level are stochastic; therefore, simulations based on differential equations oversimplify the inherent dynamics of these biological processes disregarding the importance of individual components. Our models, first simulated independently and then combined, reproduced properties of their natural counterpart. Of note, this simple model

could be further expanded following the same modular procedure depicted above: separated modules, representing different parts of the whole system, can be modeled independently to then be added to produce a global model. In addition, once the parameters of the smaller models are determined, their re-optimization in the final model should be less expensive, requiring only smaller fine-tuning. This is particularly relevant when considering that by mixing the two compartments given the simple scheme that enables PISKaS to simulate compartments and transportation of agents, we are expanding this model to include the regulation of gene expression for the entire genome of *E. coli* (to be published elsewhere).

By studying a hypothetical outbreak of the Ebola virus in Chile, we demonstrated how explicit compartmentalization permits the identification of important properties of a system that may be overlooked otherwise. Traditional modeling of disease spread calculates a single exponential growth rate r_0 for each disease. In contrast, our simulations identified the different r_0 of each city, demonstrating the importance of both the population density and the connectivity pattern between cities. Despite the obvious idea that an infectious agent is transmitted at higher rates in densely populated cities than in the countryside, a compartmentalized model combining both population density and highway topology is very hard to produce using differential equations. In the case of our PISKaS model, to produce such a model is worth only a dozen lines of Kappa rules.

Assessing human collaboration scenarios by producing a computational model of the PD, highlights a completely different level of abstraction when modeling CSs. Interactions between humans follow complex rules, difficult to understand, and quantify, where the relevance of an individual could be of radical importance for the whole (Asimov, 1951). To simplify the complexity of these interactions, we have produced a PISKaS model of the PD where, apart from the classical GT approach, we introduced trust as a proxy to the creation of social wealth. As a consequence, we quantified the GDP or our artificial societies that is generated by the dynamics of the PD. Notably, our simulations suggest that to increase the wealth of a society, the most relevant actor is the accumulation of goods by the middle class, usually the largest segment in western societies. Moreover, to support the advancement of societies, a relevant factor is the creation and development of trust between individuals

belonging to the same social class but more importantly, is the creation and development of trust between different social classes.

3.6 CONCLUSION

In this work, we illustrate an alternative to the deterministic simulation of CSs based on the stochastic simulation paradigm. Our three examples introduce the most relevant characteristics of PISKaS, our rule-based stochastic simulation engine. As a whole, PISKaS versatility provides a suitable framework for the study of complex multiscale systems with explicit spatial definitions. As discussed above, these features are particularly important to model biological systems where the spatial heterogeneity and the stochasticity generated by individual components are both key elements to understand the dynamics of the system.

4. CHAPTER III: AUTOMATIC RECONSTRUCTION OF RULE-BASED MODELS FOR REGULATION OF BACTERIAL GENE EXPRESSION AND METABOLISM.

Rodrigo Santibáñez[1,2], Daniel Garrido[2], Alberto J.M. Martin[1]

[1]Network Biology Lab, Centro de Genómica y Bioinformática, Facultad de Ciencias, Universidad Mayor, Santiago, Chile

[2]Department of Chemical and Bioprocess Engineering, School of Engineering, Pontificia Universidad Católica de Chile, Santiago, Chile

Manuscript under review in the Bioinformatics journal.

4.1 SUMMARY

Cells are complex systems composed of hundreds of genes whose products interact to produce elaborated behaviors. To control such behaviors, cells rely on transcription factors to regulate gene expression and gene regulatory networks (GRNs) are employed to describe and understand such behavior. However, GRNs are static models and dynamic models are difficult to obtain due to their size, complexity, stochastic dynamics, and interactions with other cell processes.

We developed Atlas, a python software that converts genome graphs and gene regulatory, interaction, and metabolic networks into dynamic models. The software employs these biological networks to write rule-based models for the PySB framework. The underlying method is a divide-and-conquer strategy to obtain sub-models and combine them later into an ensemble model. To exemplify the utility of Atlas, we used networks of varying size and complexity of *Escherichia coli* and evaluated in silico modifications such as gene knockouts and the insertion of promoters and terminators. Moreover, the methodology could be applied to the dynamic modeling of natural and synthetic networks of any bacteria. Code, models, and tutorials are available online (https://github.com/networkbiolab/atlas).

4.2 INTRODUCTION

Recent technological advances have allowed the inquiry and understanding of biological systems at unprecedented detail (e.g., Regev et al., 2017). From such developments, the impact of stochastic dynamics in living systems has been corroborated, measured, and modeled (Raj & van Oudenaarden, 2008). To date, most of the available models look for the reproduction of cell metabolism at genome-scale using constraint-based models (Szigeti et al., 2018). However, constraint-based models disregard the dynamic and stochastic nature of metabolism (Costa et al., 2016) and the prediction of the impact of genetic modifications remains challenging (e.g., Foster et al., 2019, or Long & Antoniewicz, 2019). Dynamic modeling of metabolism has been proposed to circumvent the drawbacks of constraint-based models (e.g., Hädicke & Klamt, 2017) despite specific issues such as the need for calibration, extensive validation, and showing a time-consuming development. Also, it is necessary to consider that metabolism is only one aspect of cellular behavior and models are desired encompassing all cellular processes (Karr et al., 2012; Karr, Takahashi, et al., 2015; Sanghvi et al., 2013). If available, those models would help in understanding complex cell dynamics, with applications in biotechnology or biomedicine (Carrera & Covert, 2015), e.g. designing minimal cells (Rees-Garbutt et al., 2020) or synthetic genomes (Fredens et al., 2019). Although there are available whole-cell models for *Mycoplasma genitalium* (Karr et al., 2012) and recently for *Escherichia coli* (Macklin et al., 2020) and there is a clear pathway to develop integrative and larger models (Covert et al., 2008; Szigeti et al., 2018), whole-cell models are still not widely developed neither adopted (Szigeti et al., 2018).

4.3 APPROACH

Gene expression regulates metabolism, which in turn modulates transcription, translation, and degradation rates as well as the activity of transcription factors (Covert et al., 2008). These processes interplay in networks of molecular interactions between DNA, RNA, proteins, and metabolites (Grimbs et al., 2019; Hernández-Prieto et al., 2014). Here, we aimed to perform the integrative modeling of transcription, translation, regulation of gene expression, metabolism, and genome architecture, which is considered a prototype whole-cell model (Szigeti et al., 2018). We developed Atlas, a software that facilitates the dynamic

modeling of gene regulation and bacterial metabolism by using biological networks to develop Rule-Based Models (RBMs) employing the PySB framework (Lopez et al., 2013) for later simulation, curation, and analysis. The developed software takes inspiration in available tools that automate the reconstruction of draft constraint-based models such as Merlin (Dias et al., 2015), RAVEN (Agren et al., 2013; H. Wang et al., 2018), ModelSEED (Henry et al., 2010), KBase (Arkin et al., 2018), and other software (reviewed in Faria et al., 2018).

An RBM employs an abstract language very similar to chemical equations capable to encode millions of individual reactions (Danos, Feret, Fontana, Harmer, et al., 2007) depending on the strictness of rule definitions. Further, we chose to develop RBMs because of their modularity, they allow deterministic simulations through network generation (Blinov et al., 2004, 2006; Hlavacek et al., 2006), they do not require the modeling of mass balances for all molecular species (network-free simulations, Sneddon et al., 2011), and permit stochastic simulations employing the Gillespie´s Stochastic Simulation Algorithm (SSA, Daniel T. Gillespie, 1976) or modifications (Danos et al., 2008; Danos, Feret, Fontana, Harmer, et al., 2007; Sneddon et al., 2011). In addition to the mentioned features, RBMs are more readable than counterparts such as ODE-based models, making RBMs easier to review, inspect, and correct collaboratively using version control tools, e.g. Git (Perez-Riverol et al., 2016). Finally, Rule-based languages were used previously to model automatically signaling pathways, e.g. with the software INDRA (Gyori et al., 2017) and KAMI (Harmer et al., 2019).

4.4 MATERIAL AND METHODS

4.4.1 Biological networks

Primary data employed was obtained from the EcoCyc database (Karp, Ong, et al., 2018; Keseler et al., 2017) with help of an updated version of PythonCyc (github.com/latendre/PythonCyc) and PathwayTools version 24 (Karp et al., 2019). The modified API is distributed freely from github.com/networkbiolab/PythonCyc and the Python Package Index with examples of use at github.com/networkbiolab/pythoncyc notebooks.

Data was formatted as biological networks. For instance, genes are connected to their regulators in a canonical GRN and we modified the network to connect transcription factors to DNA binding sites and RNA Polymerase-Sigma Factors (RNAP-σ) to promoters to obtain a sigma-specificity network. In the case of metabolic networks, we employed tripartite networks where a reaction connects to the associated enzyme and metabolite(s) instead of the more common bipartite representation of reactions and metabolites. Finally, the protein-protein, the protein-DNA binding sites (a GRN), and the protein-metabolite interaction networks were formatted as collapsed hyper-graphs (Klamt et al., 2009) to encode complexes, i.e., networks where (a group of) nodes connect to a group of nodes. We employed brackets to denote complexes, e.g. "[crp,crp]" representing the CRP homodimer and "[crp,CAMP,crp,CAMP]" to define the CRP-cyclic AMP heterodimer. The software *Atlas* disregards the order of components: "[crp,CAMP,crp,CAMP]" and "[crp,crp,CAMP,CAMP]" are equivalents. The networks employed in this work are in the Supplementary Tables and in the *examples* directory at github.com/ networkbiolab/atlas.

4.4.2 Natural and synthetic GRNs used as examples

Natural GRNs representing data from *E. coli* were employed as examples and include the lactose, arabinose, and fucose degradation operons (LacI, AraC, FucR regulons), the central carbon metabolism (Millard et al., 2017), and all *E. coli* transporters and enzymes from the BioCyc database (Karp, Billington, et al., 2018; Keseler et al., 2017). In addition, we employed the regulation of gene expression for the *E. coli* sigma factors (Sigma Factors Model, Perez-Acle et al., 2018). Primary data was completed with available information on genome architecture from Cho et al., (2009) and sigma factor specificity from Cho et al., (2014).

We modified the Sigma Factors Model (Perez-Acle et al., 2018) to exemplify the modeling of synthetic designs prior to experimentation. These modifications include the knockout of each sigma factor modeled and the incorporation of a promoter and/or a terminator to modify the rpoBC operon (Cho et al., 2014). The two types of in silico modifications were made modifying the genome graph used as input for Atlas, adding a promoter or a terminator between the rpoB coding DNA sequence (CDS) and the rpoC

ribosome binding site (RBS) or removing the CDS of each sigma factor preserving the natural promoters, RBS, and terminators. In the case of the insertion of a rpoC promoter, the GRN was modified to incorporate the RNAP-σ specificity of the rpoB promoter.

4.4.3 Draft, simulation, curation, and analysis of RBMs

Draft sub-models were obtained using biological networks as input and combined later in a divide-and-conquer modeling strategy. These ensemble models were employed for simulation, curation, and calibration.

Models were simulated with the PySB interfaces for the SciPy ODE integrator (Virtanen et al., 2020) and the Kappa Simulator v4.0 (KaSim, Boutillier, Maasha, et al., 2018). The ODE integration requires the enumeration of all components and individual reactions (network generation, Blinov et al., 2004, 2006; Hlavacek et al., 2006). In any situation where the network generation procedure took excessive time to finalize (set as a 5 minutes threshold), network-based simulations were replaced by network-free simulations employing KaSim.

Models were exported to kappa language and analyzed with the Kappa Static Analyzer (KaSA) from the Kappa platform (Boutillier, Maasha, et al., 2018) to perform reachability analysis (Danos et al., 2008; Feret, 2007), after their reconstruction or any manual curation. In brief, RBMs describe a network of reactions, and some of them could be dead rules due to the unavailability of preceding rules that synthesize reactants in the required form. Curation of the data was carried out manually, for instance, to remove duplications (e.g., gene products with two identical reactions, but different metabolite names), ambiguities (e.g., names referring to a family of metabolites), lack of compartmental information (e.g., transport reactions which substrate(s) and product(s) are the same metabolite, but located in different compartments), the incorrect stoichiometry of reaction per enzymatic complex, missing gene regulations, and others.

In the case of the Sigma Factor Model and its in silico genetic modifications, we performed the following analysis. Dynamics of 1000 stochastic simulations for 100 units of time performed with KaSim were contrasted employing the software edgeR (Y. Chen et al., 2014; M. D. Robinson et al., 2009). Simulations were carried on with arbitrary rates at one

event per unit time (also arbitrary). In the case of the addition of a promoter and/or a terminator to modify the rpoBC operon, the three resulting models were subject to calibration with transcriptomics data of cold stress from Jozefczuk et al. (2010) assuming the new networks describe the correct genome architecture. We calibrated the binding and unbinding rates of the RNAP-σ complexes to promoters and the RNA decay rates of the new models and the reference model employing the software Pleione and the described strategy 3 with the chi-square fitness function (Santibáñez et al., 2019): 100 iterations, 100 models per iteration, selecting two models to recombine with a probability inverse to the ranking. After calibration, co-expression networks were constructed with ExpressionCorrelation (www.baderlab.org/Software/ExpressionCorrelation) for the average values of 20 simulations. The ExpressionCorrelation employs the Pearson´s correlation coefficient and we selected absolute values higher than 0.95 for visualization.

The co-expression and other networks were visualized with the software Cytoscape v3.7.2 (Shannon et al., 2003; Su et al., 2014) and models were visualized within Jupyter notebooks with the software pyViPR (Ortega & Lopez, 2020).

4.5 RESULTS AND DISCUSSION

4.5.1 Software overview and basic workflow

An overview of the Atlas software is depicted in Figure 4-1. The main module is able to reconstruct independent models from genome graphs and protein-protein, protein-metabolites, and protein-DNA interaction and metabolic networks. In addition, a specialized function could employ simultaneously data from the genome graph and from a sigma-specificity network (RNAPσ-promoters interaction network) to produce a model of bacterial regulation of gene expression. As models are independent, the module also provides a function to combine them, and functions to add regulatory relationships to gene expression rules; to get, remove, modify, and add rules; and remove and get the current value of a parameter. After reconstruction, models require to set their parameters (if they were not provided as metadata in the networks) and to define the initial condition.

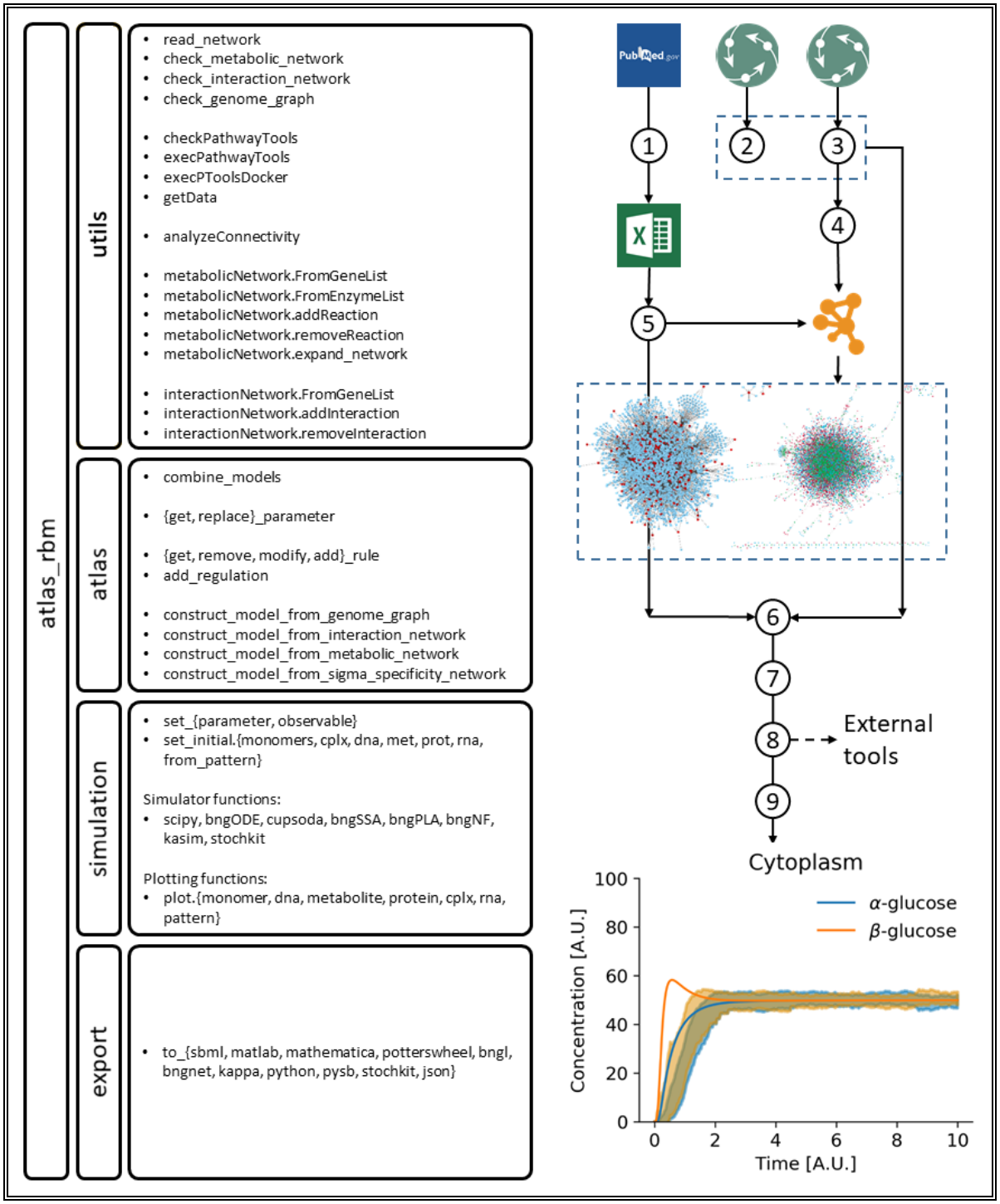


Figure 4-1. Overview of the Atlas software and a typical workflow from gathering data to plot simulation results.

Left. Atlas is a python3 software divided into four modules: The main module (*atlas*) has functions to reconstruct rule-based models from biological networks in plain text. The *utils* module has functions to read and check networks (uniqueness of reactions and uniqueness

of interactions), analyze the models produced by Atlas with KaSA, and get information from locally installed BioCyc databases with help of the PathwayTools software. The *simulation* module has functions to set parameters, observables, and initials, simulate the model with a variety of software, and to plot the results. Finally, the *export* module has functions to export the model through any supported format in the PySB framework. Right. A typical workflow is divided into the following steps: 1) Review data from the literature, 2) Obtain protein complex composition from PathwayTools with ‘utils.interactionNetwork.FromGeneList‘, 3) Obtain metabolic data from PathwayTools with ‘utils.metabolicNetwork.FromGeneList‘ or ‘utils.metabolicNetwork.FromEnzymeList‘, 4) Expand the metabolic network (Enzymes, Substrates, Products) to a source-target format, 5) Compile data from any source employing a spreadsheet software or a text editor, 6) Reconstruct models matching the type of network and execute ‘combine_models‘, add regulatory interactions of protein-DNA interactions to gene expression rules or correct rules if necessary, 7) Set parameters, observables, and the initial condition of model components, 8) Optionally, export the model for simulation, analysis, or curation with external tools, and 9) Simulation and plot of variables. A.U.: Arbitrary Unit.

The user of Atlas could choose from a variety of simulators and finally, to plot the results of simulations. In the case of stochastic simulations, results include every simulation along with the mean and the standard deviation. The user could export the model at any stage in a variety of formats, and employ external tools to simulate, curate, or analyze the reconstructed model. Complementary, Atlas provides utilitarian functions able to read and check the different networks, analyze the connectivity of the model, and obtain data from the BioCyc databases (Caspi et al., 2016; Karp, Billington, et al., 2018). Data could be transformed and exported for visualization with Cytoscape (Shannon et al., 2003; Su et al., 2014) and models could be visualized within Jupyter notebooks with pyViPR (Ortega & Lopez, 2020).

An important note is the definition of the different components of the model (or agents). We defined five distinct agents: Proteins (’prot’), Metabolites (’met’), DNA (’dna’), RNA (’rna’), and Complexes (’cplx’). A ‘Complex‘ agent is an alias for complexes such as the RNAP or the bacterial ribosome. All agents have a ‘name‘ and a ‘location‘ sites for identification purposes. In addition, all components have interaction sites named as ’dna’, ’met’, ’prot’, and ’rna’ that allow interaction with another agent of the matching type. The DNA agents have an additional identification site called ’type’ to define their nature: promoter, RBS, CDS, terminator, or binding site. Finally, proteins, DNA, and RNA agents

have two sites named ’up’ and ’dw’ that allow the automated description of complexes of the same type (e.g. two proteins interacts in their ’up’ and ’dw’ sites, instead of the ’prot’ site). Following the definition of agents, Atlas is capable to write complexes of any size and determine the correct internal links of components.

4.5.2 The lactose operon: Modeling regulation of gene expression, transcription, translation, and metabolism

We modeled a variety of metabolic networks of different size and complexity. The lactose model is composed of three enzyme-coding genes and one regulator, the arabinose-fucose model is composed of 13 enzyme-coding genes and 2 regulators, the *E. coli* central carbon metabolism model is composed of 200 enzyme-coding genes, and the genome-scale metabolic model of *E. coli* with 3596 transport and enzymatic reactions. To highlight the capabilities of *Atlas*, we describe in detail the modeling of the lactose metabolism because it is a common model of gene regulation with more than 50 years of biochemical information (M. Lewis, 2011).

The lactose operon from *E. coli* consists of three genes: the *β*-galactosidase gene lacZ, the lactose permease gene lacY (also known as lactose-proton symporter), and the galactoside O-acetyltransferase gene lacA. The EcoCyc database informs that LacY is able to incorporate *α*lactose, melibiose, lactulose, 3-O-galactosylarabinose, and melibionate into the cell cytoplasm. Interestingly, the common synthetic activator IPTG (o-nitrophenyl-*β*-galactoside) is mentioned in the description for the lactose transport, but there is no inclusion of the reaction for LacY. Next, LacZ could metabolize lactose into β-galactose and glucose, lactulose into β-galactose and fructofuranose, and 3-O-galactosylarabinose into βgalactose and arabinose. Data from literature (e.g. Huber et al., 1981; Juers et al., 2012) was used to complete the data derived from the EcoCyc database and was added manually to the network (Table S 4-1, Table S 4-2, and Figure S 4-1) and the final network is depicted in Figure 4-2A (labels were omitted for visualization purposes). The modeling of similar corrections for other enzymes could be useful to understand the dynamic properties of metabolic pathways before experimental validation of the kinetics properties of each enzyme. For the case of lactose degradation, simulations of the curated metabolic network are shown in Figure 4-2B

for the two anomers of glucose, galactose, and allolactose produced from a source of 100 molecules (or an arbitrary concentration unit) of β-lactose. As expected, the degradation of lactose into glucose and galactose is complete, while mutarotation allows equilibrium of anomers. Although sugar mutarotations are very slow reactions, they are spontaneous and we included in Atlas the capacity to model non-enzymatic reactions as EcoCyc reports 145 "spontaneous" and three transport reactions without an identified gene.

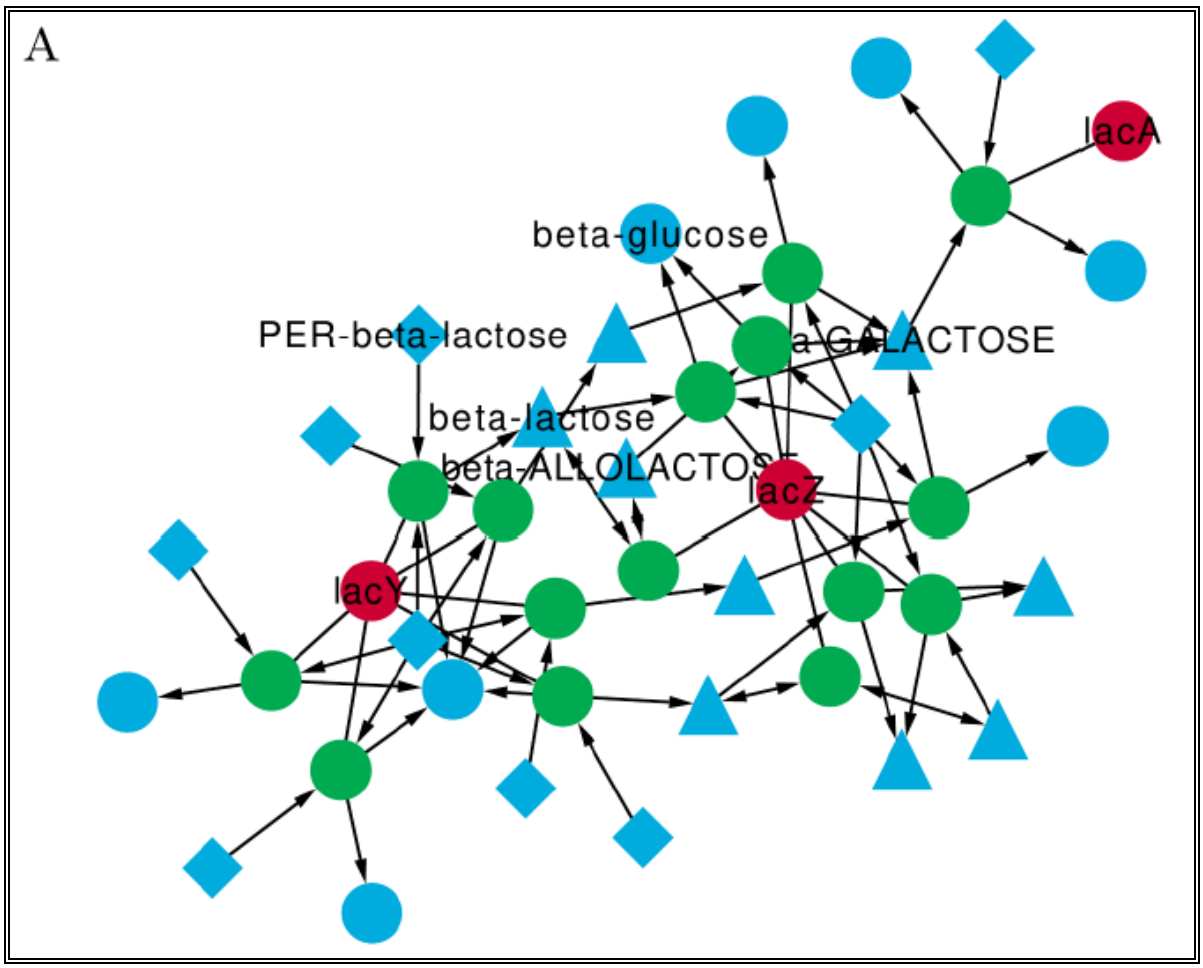


Figure 4-2. Simulation of RBMs for the lactose degradation pathway.
Panel A shows a visualization of the curated metabolic network from the EcoCyc database. Nodes represent enzymes (red), reactions (green), and metabolites (cyan). Shapes represent substrates (diamonds), intermediates (triangles), and products (circles). Arrows show the reaction reversibility.

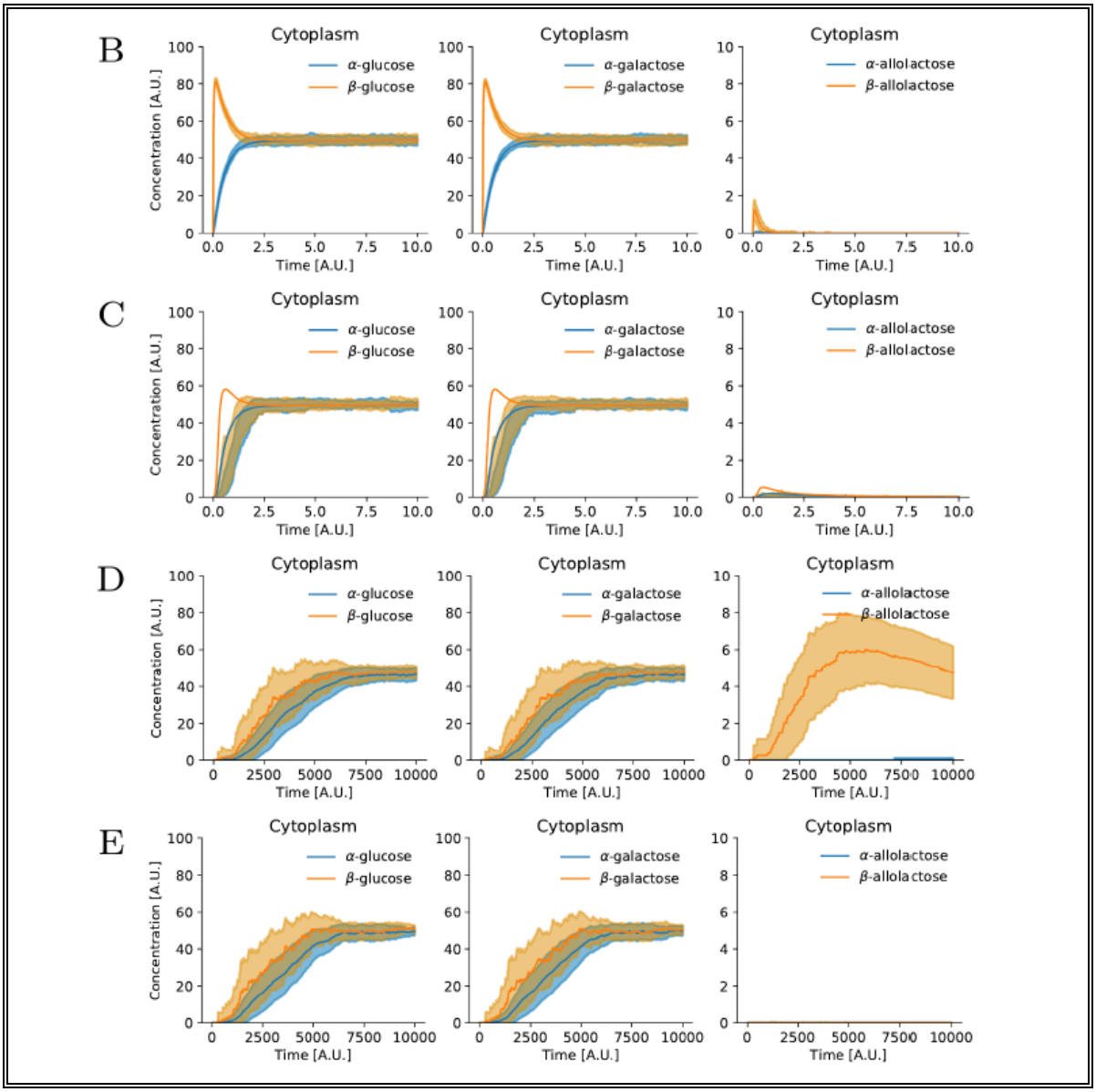


Figure 4-2. Simulation of RBMs for the lactose degradation pathway (continued)
(B-E) The total concentration of glucose, galactose, and allolactose produced from 100 molecules of lactose with hypothetical parameters. The continuous lines represent a deterministic simulation (SciPy, panels B and C) or the mean of 100 stochastic simulations (KaSim, D and E) with the area showing one standard deviation. (B) Simulation of the metabolic network reconstructed from network in panel A. (C) Simulation of the metabolic and protein-protein interaction networks. (D, E) Simulation of the metabolism, protein-protein interactions, transcription, translation, and gene expression regulation: Panel D depicts the natural situation where allolactose binds a lacI protein and is protected from degradation, and panel E shows a hypothetical situation where allolactose cannot bind the lacI protein. Models at github.com/networkbiolab/atlas/tree/master/examples/lactose. A.U.: Arbitrary Unit.

Once we curated the metabolic network, we modeled next the protein-protein interaction network, which connects gene expression to the metabolism for reactions performed by protein complexes (Figure S 4-2, Table S 4-4). For the *E. coli* lactose metabolism, the *β*-galactosidase is a homotetramer, the galactoside O-acetyltransferase is a homotrimer, and the lactose-proton symporter acts as a monomer. We employed the collapsed hyper-network representation to describe the protein-protein interactions from literature or assumptions and automated the modeling of assembly processes. For instance, the assembly process for the *β*-galactosidase tetramer comes from dimers (Matsuura et al., 2011), and we supposed the existence of a galactoside O-acetyltransferase dimer as pre-complex (Fowler et al., 1985). Additionally, we took into consideration the reaction stoichiometry for each enzymatic complex. In this curation step, we identified if reactions could happen independently of complex assembly (i.e., monomers are catalytically active) or if the protein complex is necessary for the catalytic activity *in vivo* (i.e., monomers are inactive). For the *β*-galactosidase complex, each subunit is catalytically active only when the tetramer is assembled (X. Li et al., 2018). Similarly, the galactoside O-acetyltransferase active sites act independently of each other and since they are formed with residues from two adjacent monomers (Lewendon et al., 1995; X. G. Wang et al., 2002) the trimer was assumed as the only active catalytic form. Figure 4-2C shows deterministic and stochastic simulations for an RBM including the assembly of protein complexes and the metabolic reactions. Interestingly, the deterministic and stochastic simulations disagree at the beginning of the dynamics, although they reached a similar steady-state. As the stochastic simulation requires the assembly of enzyme complexes before performing any metabolic reaction, they show a lag-phase in contrast to the deterministic simulation.

Next, the model was coupled to a representation for transcription and translation in addition to the activity of transcription factors. We employed the Kappa BioBrick Framework (KBF, Stewart & Wilson-Kanamori, 2011) and automated the modeling of rules describing bacterial transcription and translation. The KBF describes transcription and translation as a succession of rules: The reversible docking of RNAP (ribosome) to a promoter (RBS), the sliding of the RNAP through the DNA and sliding of the ribosome through the RNA, and fall off from the terminator (RNAP) or the stop codon (ribosome).

Atlas considers all promoters and terminators to write the rules described in the KBF. The transcription from the lactose operon is initiated at four promoters and terminated by two Rho-independent terminators (Figure 4-3). Moreover, we modeled an internal promoter that drives transcription from the lacYA operon, although its importance *in vivo* is not clear (Zaslaver et al., 2006). Employing the rules defined in KBF, we reconstructed a model for transcription and translation that considers the genomic architecture of the lactose operons and we coupled it to the metabolic model presented previously (including protein assembly). Therefore, the resulting ensemble model requires only DNA (and the transcription-translation machinery) to produce the necessary proteins for metabolic activity.

Finally, the RBM representing the lactose metabolism was completed with a representation of transcriptional control. To model gene regulation, we added to the interaction network the LacI-allolactose and LacI-DNA binding site complexes. In the case of LacI, each dimer binds in tandem to one DNA binding site, and two dimers could dimerize, forming a DNA loop that impedes the binding of RNAP-σ to promoters or initiates transcription (Rutkauskas et al., 2009). To inactivate LacI, free proteins bind allolactose that seems to impede the binding of LacI-allolactose to DNA binding sites (M. Lewis, 2005). In principle, the modeling of DNA binding protein interactions requires one rule per transcription factor, as we could ignore differences in the rates of DNA-protein kinetics. However, the different affinities, the genomic architecture, and the transcription factor mechanisms (reviewed in van Hijum et al., 2009) encouraged the development of another approach. To do so, overlapping DNA binding sites and other genomic features were defined as a collapsed hyper-network similar to was done for protein complexes (Figure 4-3, Table S 4-5). Figure 4-2D shows the results of simulations where allolactose binds free lacI proteins, while Figure 4-2E shows the simulation from a hypothetical situation where free lacI proteins cannot bind allolactose. The difference from both situations was modest and showed an earlier rise of the glucose and galactose concentration of near 100 units of time when allolactose could bind lacI proteins (Figure S 4-3). Because of allolactose binds free lacI proteins, the release of lacI proteins freeing the promoter occurred in both models.

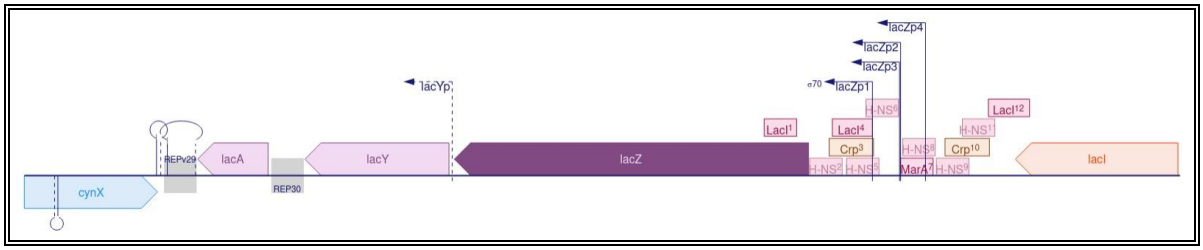


Figure 4-3. Genomic organization of the *Escherichia coli* lactose operon.
The lactose operon shows four promoters controlled independently by the repressor LacI, the activator-repressor CRP, the repressor H-NS, and the repressor MarA. An internal promoter and two terminators contribute to the expression dynamics of enzymes and the transporter. Image reproduced from the EcoCyc website (https://www.ecocyc.org/gene?orgid=ECOLI&id=EG10527#tab=TU; Keseler et al., 2017).

Although system parameters could be found in databases or calibrated (e.g. with pyBioNetFit, Mitra et al., 2019, or Pleione, Santibáñez et al., 2019), the results show that modeling of RBMs for metabolism, protein complex /.assembly, transcription, translation, and regulation of gene expression can be done in an automated manner, facilitating deterministic and stochastic simulation. Parameters employed for simulation are detailed in Table S 4-2, S 4-3, S 4-4, and S 4-5, and a benchmark is detailed in Table S 4-6.

4.5.3 Modeling natural and synthetic transcriptional control: The sigma factors model.

We later addressed the modeling of RNAP-σ assembly and transcriptional control of its expression mediated by the activity of sigma factors. Compared to eukaryotes, bacteria have only one RNAP and different sigma factors that confer promoter specificity (Mauri & Klumpp, 2014). *E. coli* has seven sigma factors that interact physically with the core RNAP to form holoenzymes. The purpose here is to present how to model transcription control as the RBM is presented and calibrated elsewhere (Perez-Acle et al., 2018; Santibáñez et al., 2019) and to employ it to model synthetic transcriptional control. Also, Atlas models a molecular step in bacterial transcription disregarded in KBF (Stewart & Wilson-Kanamori, 2011): the sigma factor is released from holoenzymes when transcription is initiated (Mauri & Klumpp, 2014).

We modeled the holoenzymes binding to promoters as if those interactions were the binding of any transcription factor to their cognate DNA binding sites. To do so, the RNAPσ specificity (Table S 4-7) and the genome architecture (Table S 4-8) were considered simultaneously, two features employed separately for the modeling of DNA-protein interactions and transcription. Both networks are represented in Figure 4-4A (a canonical GRN) and Figure 4-4B (an extended network to show the considered genome architecture). The resulting model describes holoenzymes explicitly as a complex of five proteins instead of a unique agent modeling the RNAP complex employed in the lactose model. Results for the dynamics of the described GRN are shown in Figure 4-4C and Figure 4-4D for a hypothetical case of only RNA synthesis without mRNA degradation. It can be seen that gene expression shows similar rates, though results are influenced by the parameter values and the initial condition for proteins.

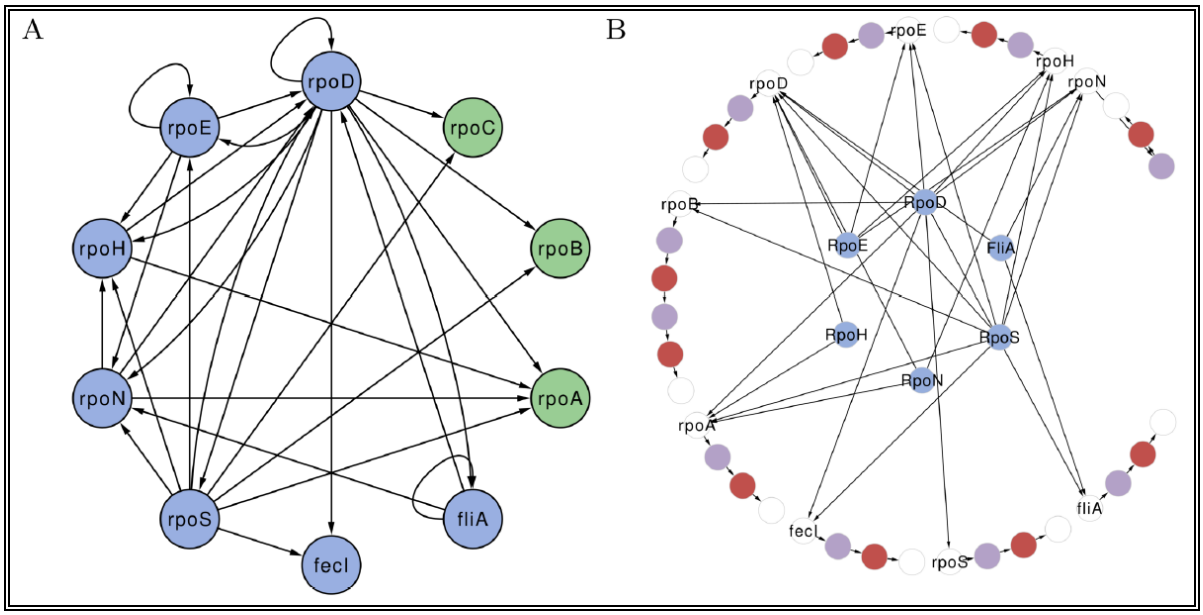


Figure 4-4. Stochastic simulation of the *Escherichia coli* sigma factor GRN.
(A) Visualization of the curated GRN from the EcoCyc database. Light blue nodes represent the seven sigma factor and the green nodes represent the three RNAP subunits encoding genes. Arrows represent the positive regulation of transcription determined from sigma factor specificity for promoters.

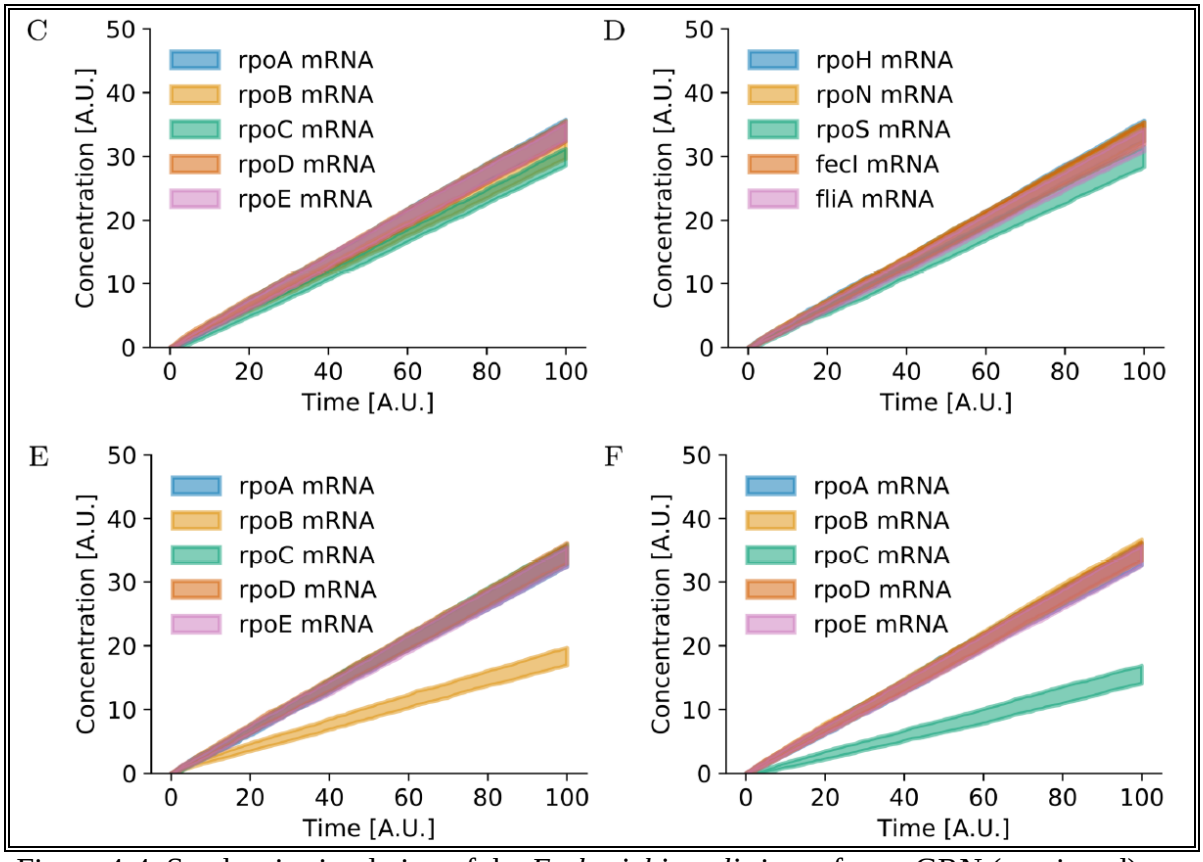


Figure 4-4. Stochastic simulation of the *Escherichia coli* sigma factor GRN (continued). (B-D) Mean of 100 stochastic simulations (KaSim) and one standard deviation from the mean. (B) Extension of the GRN to encode the genomic architecture of the ten considered genes. The rpoB and rpoC (left side of the outer ring) form a single operon. Labeled white nodes are the promoters, purple nodes are the ribosome binding sites, red nodes are the coding DNA sequence, and unlabeled white nodes are the terminators. (C, D) Stochastic simulation of the natural genomic architecture and regulatory interactions. (E) Stochastic simulation for the network modified with an in silico internal rpoC promoter. (F) Stochastic simulation for the network modified with an in silico internal rpoB terminator. Models at github.com/networkbiolab/atlas/tree/master/examples/sigma-model. A.U.: Arbitrary Unit.

The use of Atlas is not restricted to natural networks and allows the modeling of different genomic arrangement of genes. One purpose of such procedure is to assess differences in mRNA and other cell component dynamics with the final goal of the computational-aided design before experimental evaluation. For the sigma model, we modeled three variants that modified the rpoBC operon architecture. Those variants were: a) the incorporation of an internal promoter between rpoB and rpoC genes allowing the

interaction of an RNAP-σ complex, b) the incorporation of an internal terminator allowing falloff of the RNAP, and c) the incorporation of both. Our simulations showed that the incorporation of a promoter for rpoC reduced the synthesis rate for rpoB due to reduced RNAP availability for its promoter (Figure 4-4E), which in turn is determined by the model parameters and the initial condition. On the contrary, the addition of the internal terminator reduced the synthesis rate for rpoC (Figure 4-4F) due to the reduced probability to continue RNA elongation from rpoB into rpoC. Finally, both modifications showed no changes in RNA synthesis rates due to the compensation of falling off RNAPs from the rpoB terminator and interaction of RNAP-σ holoenzymes to the synthetic rpoC promoter (Figure S 4-4B). In addition, the model showed similar expression rates as the situation of independent rpoB and rpoC operons (Figure S 4-4D).

More realistic stochastic simulations were performed with the Sigma Factors Model extended to model RNA degradation as unimolecular decay (Perez-Acle et al., 2018). In contrast with the simulation results shown in Figure 4-4E, Figure 4-4F, and Figure S 4-4, the in silico variants were calibrated as if the new models represent the natural genomic architectures. Also, we performed an indirect comparison of mRNA quantities using Pearson´s correlation coefficient to compare mRNA dynamics for the average of simulations. We report correlations higher than 0.95 as an absolute value in a co-expression network (Figure S 4-5). The expression profiles for rpoB and rpoC remained correlated in all variants, in contrast to a correlation coefficient of 0.57 determined from the original data. A complete explanation is tailored to the ability of the performed calibration to find parameter values that resemble the experimental data the best possible for an unnatural transcriptional network and delineates the need for (cell-free) experiments to accurately measure RNA synthesis rates in modified genomic contexts.

Finally, we performed in silico knockout experiments. Comparisons employing the edgeR software (Y. Chen et al., 2014; M. D. Robinson et al., 2009) and a threshold for the False Discovery Rate (FDR) of 0.05 showed that the deletion of rpoD and rpoS impacted the most in mRNA synthesis, while the other deletions did not show differential expressed genes. The knockout of rpoD impedes the expression of rpoS (Figure 4-4B) while we observed lower expression for fliA and fecI and higher expression for rpoA, rpoE, rpoH, and

rpoN compared to the reference model. In turn, the knockout of rpoS showed lower expression for rpoB and fecI, and higher expression for fliA. Determined fold change and FDR values are in Table S 4-9 and Table S 4-10, respectively. Consider that, simulations were done to highlight the capability of Atlas to model different genetic modifications and parameters did not reflect any experimentally determined rate. Also, the models did not incorporate degradation rules for mRNAs and an extension of the model to synthesize and degrade proteins will allow the detailed modeling of in silico designs and the comparison of simulations to experimental data from synthetic constructs employing cell-free translation-transcription technologies (Borkowski et al., 2018).

4.6 CONCLUSIONS

Mathematical and computational modeling is often viewed as a specialized task. To facilitate modeling, we automated the development of RBMs as these types of models show simulation flexibility, a reasonable degree of readability, modularity for integrative modeling, and good simulation scalability.

Atlas produces sub-models from genome graphs, protein-protein, protein-metabolites, and protein-DNA interaction, and metabolic networks. We developed in this work a divide-and-conquer strategy supported by the modularity of RBMs as it is the pathway for the development of whole-cell models (Szigeti et al., 2018). The software produces RBMs for the PySB framework (Lopez et al., 2013) and rules can be added in any order while PySB checks if new rules are compatible with the current model. In addition, PySB could export to *kappa* language and we employed the KaSA software (Boutillier, Maasha, et al., 2018) to assess further the coherence of the developed RBMs. Simulation of RBMs could be done within PySB and calibration of exported models could be performed with pyBioNetFit (Mitra et al., 2019, only BNGL models) or *Pleione* (Santibáñez et al., 2019, BNGL and *kappa* models) to compare the reconstructed models with experimental data or available models.

Atlas contrast to available software because it lacks a graphical interface (e.g. RuleBender, Smith et al., 2012, and VirtualCell, Blinov et al., 2017) although the user could employ Atlas within a Jupyter notebook and use pyViPR (Ortega & Lopez, 2020) to

visualize the model structure. Also, Atlas relies on the user to obtain formatted data to model interactions in contrast to INDRA (Gyori et al., 2017) that could use Natural Language Processing to read information and reconstruct models. In turn, Atlas could model metabolism, transcription, and translation, besides protein-protein interactions widespread found in signaling pathways that INDRA (Gyori et al., 2017) and KAMI (Harmer et al., 2019) could model.

Finally, the models and the Atlas software are extensible, for instance, to model cooperative behavior currently not supported. The utilization of the law of mass action for the metabolic network (and other reactions) limits the utility of the resulting RBMs in the current form, but export to BNGL or kappa leverages this imposition as they support mathematical expressions as reaction rate. However, we expect to extend Atlas to consider enzyme-metabolites interactions and describe detailed mechanisms of enzyme reactions (Saa & Nielsen, 2017), allosteric regulations of metabolic activity, and to model the assembly of ribosomes (Davis et al., 2016; Gupta & Culver, 2014; Shajani et al., 2011). Notably, Atlas is already compatible with metabolic and interaction data from eukaryotes and we obtained a model from data for 1991 metabolic reactions of the yeast *Saccharomyces cerevisiae* from BioCyc. In addition, we expect further interoperability with INDRA models of signaling pathways to model protein modifications such as phosphorylation and the collaboration from researchers. With collaboration in mind, we shared the developed models in this work at https://github.com/networkbiolab/Atlas/examples.

5. CHAPTER IV: *PLEIONE*: A TOOL FOR STATISTICAL AND MULTI-OBJECTIVE CALIBRATION OF RULE-BASED MODELS

Rodrigo Santibáñez[1,2], Daniel Garrido[2], Alberto J.M. Martin[1]

[1]Network Biology Lab, Centro de Genómica y Bioinformática, Facultad de Ciencias, Universidad Mayor, Santiago, Chile

[2]Department of Chemical and Bioprocess Engineering, School of Engineering, Pontificia Universidad Católica de Chile, Santiago, Chile

Published in *Scientific Reports* (2019). Vol. 9, Issue 1, 15104

5.1 SUMMARY

Mathematical models based on Ordinary Differential Equations (ODEs) are frequently used to describe and simulate biological systems. Nevertheless, such models are often difficult to understand. Unlike ODE models, Rule-Based Models (RBMs) utilize formal language to describe reactions as a cumulative number of statements that are easier to understand and correct. They are also gaining popularity because of their conciseness and simulation flexibility. However, RBMs generally lack tools to perform further analysis that requires simulation. This situation arises because exact and approximate simulations are computationally intensive. Translating RBMs into ODEs is commonly used to reduce simulation time, but this technique may be prohibitive due to the combinatorial explosion. Here, we present the software called *Pleione* to calibrate RBMs. Parameter calibration is essential given the incomplete experimental determination of reaction rates and the goal of using models to reproduce experimental data. The software distributes stochastic simulations and calculations and incorporates equivalence tests to determine the fitness of RBMs compared with data. The primary features of *Pleione* were thoroughly tested on a model of gene regulation in *Escherichia coli*. *Pleione* yielded satisfactory results regarding calculation time and error reduction for multiple simulators, models, parameter search strategies, and computing infrastructures.

5.2 INTRODUCTION

Systems biology studies the behavior of biological systems by determining and quantifying all of the molecular interactions that characterize them (Endy & Brent, 2001). This area of science relies on different experimental, mathematical, and computational tools to address system generalities such as robustness and specific details such as bi-stability (Breitling, 2010; Fisher & Henzinger, 2007). Notably, these computational approaches can be classified into two primary types: those that aim to determine cell component interactions (e.g., methods to infer Gene Regulatory Networks GRNs, from expression data Marbach et al., 2012) and those used to study the dynamical properties that define such systems (Endy & Brent, 2001; Fisher & Henzinger, 2007).

In the post-genomic era, elucidating gene regulation remains one of the primary challenges of systems biology (Martin et al., 2016). This information is highly relevant to understanding metabolism (Fischer & Sauer, 2003; Fuhrer et al., 2017), cell responses (Jozefczuk et al., 2010; M. Kim et al., 2015), and cell-to-cell interactions (Shoaie et al., 2013; Sung et al., 2017) and for developing industrial applications of microorganisms (Johns et al., 2018). The increasing availability of omics data has facilitated the modeling of biological systems with the goal of understanding and predicting their behavior (Bartocci & Lió, 2016). Frequently modeled experimental data include genomics, transcriptomics, proteomics, and metabolomics (Szigeti et al., 2018). These datasets can be modeled in single-omics or integrated, multi-omics representations of cell behavior (M. Kim & Tagkopoulos, 2018). Historically, Ordinary Differential Equations (ODEs)-based models have been extensively used for modeling biological systems (Szigeti et al., 2018). Nowadays, Rule-Based Models (RBMs) are gaining popularity because of their advantages over their ODE counterparts (Chylek et al., 2015; Danos et al., 2007a; Lopez et al., 2013). For example, RBMs are best suited to modeling large systems that may be composed of millions of different types of components and transformations (Danos et al., 2007a; Lopez et al., 2013). To simulate RBMs, most of the available tools use the Gillespie´s Stochastic Simulation Algorithm (SSA), a method to retrieve an exact numerical solution from a Chemical Master Equation (Daniel T. Gillespie, 1977). The SSA and its derivatives are implemented in RBM simulators

such as KaSim (Danos et al., 2007a), BioNetGen (BNG) (Blinov et al., 2006; James R Faeder et al., 2003), and others (Gibson & Bruck, 2000; Hogg et al., 2014; McCollum et al., 2006; Schaff et al., 2016; Sneddon et al., 2008). Unfortunately, tools to perform calibration and determine parameter uncertainty of RBMs are generally lacking; they are available only for BioNetGen Language (BNGL) and Systems Biology Markup Language (SBML) models (Mitra et al., 2019; Thomas et al., 2015). Without proper calibration, analyses of model perturbations and predictions beyond experimental data are impossible.

One significant difficulty of modeling biological systems is robustly estimating parameter values. This calibration procedure is especially relevant considering that the purpose of models is to reproduce observed data or phenomena (Sun et al., 2012). Notably, this problem is computationally expensive in the case of RBMs, where conventional approaches rely on an equivalent ODE model to reduce the calculation time (Aguilera et al., 2017; Kozer et al., 2013). In general, multiple simulations of an RBM converge on a numerical solution to its ODE counterpart. However, both modeling frameworks entail different assumptions (Chylek et al., 2015). As a calibration example, Kozer *et al.* (Kozer et al., 2013) employed BNG to simulate multiple RBMs that differed only in parameter values, solving each model in a deterministic fashion with the CVODE software (Hindmarsh et al., 2005). In contrast to Kozer *et al.* (Kozer et al., 2013), Aguilera *et al.* (Aguilera et al., 2017) developed and performed a calibration of a stochastic model employing first deterministic simulations. These authors argued that if a deterministic stationary state closely matches the modes of the experimental data, the employed parameters are good candidates for fitting stochastic simulations to the same data (Aguilera et al., 2017). However, their approach fails if the stationary state is remarkably different from the simulated pseudo-stationary state of stochastic simulations, as was argued by Halh & Kremling (Hahl & Kremling, 2016), or in situations where a deterministic simulation is not possible. For instance, KaSim (Danos et al., 2007a) does not provide an ODE solver, and while KaDE (Camporesi et al., 2017) can export a *kappa* model to ODEs in a variety of compatible software, the size of the generated model is affected by combinatorial complexity, the explosion in the number of ODEs due to the numerous interactions and modifications modeled (Hogg et al., 2014). For example, the EGFR/ERK pathway model contains 70 rules that are equivalent to approximately 10^{23}

ODEs (Danos et al., 2007a). Despite the availability of robust methods to calibrate and analyze ODE models, using these methods to calibrate RBMs may disregard the stochastic behavior of a system and accordingly result in a loss of useful information. To circumvent these problems, Thomas *et al.* (Thomas et al., 2015) developed BioNetFit (BNF) and recently Mitra *et al.* (Mitra et al., 2019) developed PyBioNetFit (pyBNF). Both of these tools harness computational load schedulers to parallelize simulations and cut down on the time necessary to calibrate an RBM. Although BNF (and pyBNF) address the need for a calibration tool, they support only RBMs written in BNGL and SBML (James R Faeder et al., 2003) and rely on algebraic equations to compare experimental data and simulation that may be of special concern depending on the nature of the modeled phenomena.

In this study, we present and describe the features of *Pleione*, an open resource to calibrate RBMs. *Pleione* encodes a Genetic Algorithm (GA), a robust and general methodology that searches the parameter space with operations that select, recombine, and mutate models with increased fitness (Whitley, 1994). *Pleione* was developed to perform three primary tasks: calibrate RBMs regardless of their underlying formal language, statistically assess models against experimental data, and distribute calculations with minimal user intervention. *Pleione* supports BNG2 (James R Faeder et al., 2003), NFsim (Sneddon et al., 2008), KaSim (Blinov et al., 2006; Danos et al., 2007a), and PISKaS (Perez-Acle et al., 2018) to perform simulations of RBMs either in BNG or kappa language. SBML models can be transformed with BNG into BNGL models (James R Faeder et al., 2003) or with PySB and exported to a myriad of formats (Lopez et al., 2013). Furthermore, *Pleione* can evaluate models employing unique or combined fitness functions (Konak et al., 2006) referred to hereafter as single-objective GA (SOGA) or multi-objective GA (MOGA). Additionally, it incorporates parametric and non-parametric equivalence tests such as the two one-sided t-tests (Schuirmann, 1987), the Double Mann-Whitney U-test (Cornell, 1990), and the Wellek´s test (Wellek, 1996, 2010) as measurements of the fit between the experimental data and the stochastic simulations. *Pleione* can accordingly determine significant equivalences between experimental and simulated data. Lastly, we parallelized calculations employing the SLURM software, the de-facto standard for high-performance

computing infrastructure. We also support parallelized calculations without SLURM using the python *multiprocessing* package.

We tested *Pleione* in a variety of settings and report its behavior. We employed multiple search strategies with algebraic functions to calibrate 79 free parameters of the core GRN model of *E. coli* (Perez-Acle et al., 2018). We also report the uncertainty in parameter values using jackknife and bootstrap procedures. Subsequently, we calibrated the Aguilera´s simple model of gene regulation (Aguilera et al., 2017) employing the equivalence tests alone and in combination with a second fitness function. Finally, we provide a comparison of the developed method against BNF (Thomas et al., 2015), and we perform a calibration of an example RBM with six free parameters.

5.3 METHODS

5.3.1 Software implementation

Pleione is open software written in python3 and it is cross-platform compatible. The GA implemented is a simple iterative process of selection, recombination, and mutation of parameter values (Whitley, 1994). Before performing any optimization, *Pleione* reads an RBM and identifies the free parameters in the model, i.e., user-selected variables that are going to be calibrated and their allowed search space. After building the first population, *Pleione* writes as many models as defined by the user with each parameter set; it then queues the simulation jobs using SLURM or the python *multiprocessing* API. After the first population is simulated, the models are ranked using one or more of the nine algebraic and two statistical fitness functions. Then, two selection strategies can be used. The first is a uniform probability selection that selects two parents from the best ("elite of") models. The second strategy employs a distribution that is inversely proportional to the rank (and optional elitism). The latter is a selection strategy previously implemented in BNF (Thomas et al., 2015). The user can control crossover and select between single and multiple crossover points. Finally, each parameter value can be mutated, and a new value can be selected from a uniform or log-uniform distribution or multiplied by a random factor centered on the old parameter value. The simulation, selection, and mutation procedures are repeated until the number of iterations reaches the user-defined value. *Pleione´*s default is to perform elitism

with multiple crossing points and parameter mutations from a uniform distribution. *Pleione* is able to perform calibrations employing BNG2 (James R Faeder et al., 2003), NFsim (Sneddon et al., 2008), KaSim (Blinov et al., 2006; Danos et al., 2007a), and PISKaS (Perez-Acle et al., 2018), regardless of differences in model configuration, command-line interface, or format of the reported simulation results. Other (stochastic) simulators can be incorporated into *Pleione* if they provide a command-line interface. *Pleione* is freely available at Python Package Index and Github (see the *Pleione* Manual for installation and user instructions at https://pleione.readthedocs.io).

5.3.2 Fitness functions

5.3.2.1 Software implementation of iterative equivalence tests.

Equivalence tests aim to determine the significance of the extent to which two distributions differ or are equivalent to practical purposes (Cornell, 1990). The tests determine the rejection of one of the two null hypotheses that the difference lies beyond the equivalence range. The first of these implemented tests were the "two one-sided t-tests" (Schuirmann, 1987) from the python *statsmodels* package with a predefined 5% significance level. The fitness function is selected by its acronym TOST (see the *Pleione* Manual for details at https://pleione.readthedocs.io). TOST shifts simulated values to the left and the right and test whether the resulting distribution is statistically smaller or larger than unaltered data. Only when both unpaired *t*-tests reject their null hypotheses does TOST reject its own.

The second test, the Double Mann-Whitney U-test, is a straightforward adaptation of the Mann-Whitney U-test as mentioned by Cornell (Cornell, 1990) and discussed by Wellek (Wellek, 1996). The fitness function is selected by its acronym DUT (see the *Pleione* Manual for details at https://pleione.readthedocs.io). Like TOST, DUT shifts the simulated variables toward the left and the right and compares them with unaltered data using a one-sided U-test. The U-test determines whether a random variable is stochastically larger than another random variable (Marshall, 1951); it is the non-parametric counterpart of the unpaired *t*-test. To calculate the U-test, we first determined how many experimental values were smaller and greater than the shifted simulated values using Algorithm 1. Here, *exp* stands for an experiment replication; *shifted sim* stands for shifted stochastic simulations of an RBM.

Lower and *upper* are the threshold limits. After comparing the experimental data and simulations, the observed difference U_{exp} and U_{sims} were compared against a critical value using Algorithm 2.

Algorithm 1. Count how many times experimental data is larger than shifted simulated values.

1: **for** *i* **in** range(0, len(*exp*)) **do**
2: **for** *j* **in** range(0, len(*shifted sim*)) **do**
3: **if** $exp_i > shifted\ sim_j$ **then**
4: *Uexp* = *Uexp* + 1.0
5: **else if** $exp_i < shifted\_sim_j$ **then**
6: *Usim* = *Usim* + 1.0
7: **else**
8: *Uexp* = *Uexp* + 0.5
9: *Usim* = *Usim* + 0.5
10: **end if**
11: **end for**
12: **end for**

Algorithm 2. Double Mann-Whitney U-test. Determine if the difference is statistically significant for each variable.

1: $U_{max} = U_{model}$ = len(*exp*) × len(*shifted sim*)
2: **for** *i* **in** range(0, len(*exp*)) **do**
3: **for** *j* **in** range(0, len(*shifted sim*)) **do**
4: **test** H_0: *exp* > *sim* − *lower*
5: **if** $U_{max} - U_{exp} \leq U_{critic}$ **then** *null* hypothesis, H_0, is rejected
6: $U_{lower} = 1.0$
7: **else**
8: $U_{lower} = 0.0$
9: **end if**
10: **test** H_0: exp < *sim* + *upper*
11: **if** $U_{max} - U_{sim} \leq U_{critic}$ **then** *null* hypothesis, H_0, is rejected
12: $U_{upper} = 1.0$
13: **else**
14: $U_{upper} = 0.0$
15: **end if**
16: $U_{model} = U_{model} - U_{lower} \times U_{upper}$

The third equivalence test is the Wellek´s test (Wellek, 1996, 2010) that determines whether if the probability of the difference of two random variables lies within a (small) threshold around 50%. We implemented the mawi.R routine from the EQUIVNONINF R package (https://rdrr.io/cran/EQUIVNONINF/) in python. *Pleione* uses WMWET as an acronym for the fitness function.

All three equivalence tests were calculated for each variable and time point, and the fitness functions minimize the total sum of successful equivalence tests subtracted from the total number of performed tests (e.g. U_{max} in Algorithm 2).

5.3.2.2 Algebraic functions.

Pleione includes nine algebraic fitness functions, which are described below with their acronyms in parentheses. As noted previously, exp stands for an experiment replication, sim stands for a simulated value, $\overline{exp}$ refers to the average, and σ_{exp} is the standard deviation of experimental values.

• Squared Difference of two Averages (SDA):

$$\left(\frac{1}{m}\sum_{i=1}^{m} exp_i - \frac{1}{n}\sum_{j=1}^{n} sim_j\right)^2 \tag{5-1}$$

• Absolute value of the Difference of two Averages (ADA):

$$\left|\frac{1}{m}\sum_{i=1}^{m} exp_i - \frac{1}{n}\sum_{j=1}^{n} sim_j\right| \tag{5-2}$$

• Pair-Wise Square Deviation (PWSD):

$$\frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}\left(exp_i - sim_j\right)^2 \tag{5-3}$$

• Absolute Pair-Wise Deviation (APWSD):

$$\frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}\left|exp_i - sim_j\right| \tag{5-4}$$

• Normalised Pair-Wise Square Deviation (NPWSD):

$$\frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}\left(\frac{exp_i - sim_j}{exp_i}\right)^2 \tag{5-5}$$

• Normalised Absolute Pair-Wise Deviation (ANPWSD):

$$\frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}\left|\frac{exp_i - sim_j}{exp_i}\right| \quad (5\text{-}6)$$

• Sum of Squares (SSQ)(Thomas et al., 2015):

$$\sum_{i=1}^{m}\sum_{j=1}^{n}\left(exp_i - sim_j\right)^2 \quad (5\text{-}7)$$

• Chi-Square (CHISQ)(Thomas et al., 2015):

$$\sum_{i=1}^{m}\sum_{j=1}^{n}\left(\frac{exp_i - sim_j}{\sigma_{exp}}\right)^2 \quad (5\text{-}8)$$

• Mean Normalised Square Error (MNSE)(Thomas et al., 2015):

$$\sum_{i=1}^{m}\sum_{j=1}^{n}\left(\frac{exp_i - sim_j}{\overline{exp}}\right)^2 \quad (5\text{-}9)$$

5.3.3 Models and experimental data

5.3.3.1 Calibration of transcriptional dynamics.

To test *Pleione*, we employed a simple model of gene regulation (Aguilera et al., 2017) and the core GRN model of *E. coli* that we published previously (Perez-Acle et al., 2018). The former is a four-equation model developed and used by Aguilera *et al.* (Aguilera et al., 2017) with known parameters to create a synthetic dataset. The equations describe the synthesis and degradation of an mRNA and its encoded protein. We randomly selected ten simulations using the "true" parameters and then set all rates as free parameters. The latter model resembles the GRN, protein-protein interactions, and genome architecture of *E. coli* K-12 (see Figure S 5-8). The core GRN is composed of ten genes, of which seven encode sigma factors (Cho et al., 2014). The other genes correspond to the α, β, and β´ subunits of the *E. coli* RNAP. This GRN was built from EcoCyc (Keseler et al., 2017) and complemented with literature data (Cho et al., 2014). It resulted in a network of 30 positive regulations (see Figure S 5-8). The experimental data used to calibrate the model correspond to microarrays of *E. coli* gene expression after stress performed by Jozefczuk *et al.* (Jozefczuk et al., 2010)

(GEO accession GSE20305). These data were processed as in Marbach *et al.* (Marbach et al., 2012), and the resulting values were assumed to represent the absolute quantity of mRNA per cell for each available time point after stress induction (see Supplementary File S1 online). We selected the binding and unbinding of each sigma factor to the core RNAP (14 parameters), the binding and unbinding of the seven RNAP holoenzymes to their cognate promoters (56 parameters), and the decay of RNA molecules (9 parameters) as free parameters.

We employed six different GA configurations, which are described below. All strategies included 100 models simulated ten times through 100 iterations. We repeated the calibration from the same initial population choosing the same random number generator seed to test the fitness functions included in *Pleione*. All of the simulations for the Aguilera´s simple model were done with BNG v2.2.6 (https://github.com/RuleWorld/bionetgen/releases/tag/BioNetGen-2.2.6-stable). For the core GRN model, the simulations were done with KaSim v4.0 (https://github.com/Kappa-Dev/KaSim/releases/tag/v4.0). The scripts to repeat the calibration are in Supplementary File S1 online. The differences between strategies to calibrate are as follows:

• Strategy 1, elitist GA: After each iteration, the ten best models were selected and left unaltered until the next iteration. From this elite of models, two parents were selected with a uniform probability and crossed in a single, random point. We allowed the self-recombination of parents. The mutation of parameters resulted in a new value from the original range with a 30% probability. We did not repeat simulations from the elite population.

• Strategy 2, non-elitist GA. After each iteration, all models were subjected to selection with an inverse to the rank probability distribution. For instance, the model i with a rank r_i had a probability of selection p_i equal to $1/r_i \sum_{j=1}^{R} 1/r_j$. After selecting two different parents, they were recombined in a single, random point. The mutation of parameter values was the same as in Strategy 1.

• Strategy 3. The same as Strategy 2, but the mutation of parameters yielded a new value with a 20% probability from a random factor within a ±10% interval centered at the old value. This strategy was applied using BNF.
• Strategies 4, 5, and 6 (MOGA). These strategies were similar to Strategy 1, but we selected the CHISQ (Equation 5-8) and WMWET; the ANPWSD (Equation 5-6), WMWET, and PWSD (Equation 5-3); and the CHISQ, WMWET, SDA (Equation 5-1), and NPWSD (Equation 5-5) simultaneously as fitness functions for the rank models.

5.3.3.2 Comparison with BioNetFit.

Pleione was also used to calibrate an equivalent model to "example 6" reported by Thomas *et al.* (Thomas et al., 2015) in kappa language. Although Thomas *et al.* published another two models, some of the BNGL syntax within them do not have a simple or equivalent expression in *kappa* language. The "example 6" is a toy model with synthetic experimental data that resemble the ligand-induced autophosphorylation of a receptor (Thomas et al., 2015). The model was rewritten employing PySB version 1.5.0 (Lopez et al., 2013) and exported to *kappa* and BNGL. The latter was simulated with BNG2 and compared with the original model to discard syntax misinterpretation while rewriting the model. The *kappa* model was further modified to enable an equilibration step similar to that of the original model. To perform the comparison with BNF, both GAs were run employing strategy 3. The first population was drawn from a log-uniform distribution between 10^{-2} and 10^{+2}. The same options were used when calibrating with BNF, and we selected equivalent objective functions to fit with BNF and *Pleione*. The simulators were BNG v2.2.6 https://github.com/RuleWorld/bionetgen/releases/tag/BioNetGen-2.2.6-stable and NFsim v1.12.1, which was compiled from source obtained at the BNF GitHub repository (https://github.com/PosnerLab/BioNetFit). BNF was obtained from https://github.com/ RuleWorld/BioNetFit/releases/tag/v1.01.

5.3.4 Complementary analysis and statistical tests

The core GRN model simulates the availability of RNAP, also referred to as the free RNAP fraction. To determine whether the simulated free fraction was in a pseudo-stationary state, we employed the one-way ANOVA test and the Kruskal-Wallis H-test, both from the python

SciPy package (Jones et al., 2015). Additionally, to determine the probability of a range of simulated free RNAP, we repeatedly employed the one-sample *t*-test with a 95% confidence level in a one-tail test; we adjusted the threshold of free RNAP fraction over the entire time interval until the *t*-test rejected the null hypothesis at least once. Finally, to determine which fitness functions to employ in a multi-objective GA, we calculated the Pearson´s, Spearman´s rank (ρ), and Kendall´s rank (τ) correlation coefficients, which were also implemented within the SciPy package (Jones et al., 2015). Parameter uncertainty for the core GRN model was assessed using *Alcyone* https://github.com/networkbiolab/alcyone/tree/master/example employing the leave-one-out jackknife and the bootstrap resampling methods. The latter was performed with 20 GA runs, which enabled a maximum 90% confidence interval. Animations of the dynamics were prepared from 1000 simulations with KaSim. We plotted the average and one standard deviation for a total of 40 variables during the first minute at intervals of one second; the remaining simulated times had intervals of 0.1 minutes (i.e., 6 seconds).

5.4 RESULTS AND DISCUSSION

5.4.1 Single-objective genetic algorithm

5.4.1.1 Algebraic fitness functions.

Pleione calculates nine algebraic equations commonly used as fitness functions (see the Methods section for the definition of these equations). We provide all fitness functions in a single file and separately from the *Pleione* main code for three reasons. First, the supported stochastic simulators report their results in different formats. Second, the separation enables transparent parallelization of fitness calculations and its utilization to validate models with new or independent data not used to calibrate the model. And third, the separation allows a straightforward and easy way to add new fitness functions following specific guidelines (https://pleione.readthedocs.io) and to give support to new deterministic or stochastic simulators.

To test *Pleione*, we employed an elitist GA (see the Methods section, strategy 1) and transcriptomic data from Jozefczuk *et al.* (Jozefczuk et al., 2010) to calibrate the core GRN model (Perez-Acle et al., 2018). Jozefczuk *et al.* used a variety of conditions to evaluate the

changes in mRNA expression using microarrays. We selected cold stress to exemplify the use of *Pleione*, although any experimental procedure that determines the abundance of mRNA molecules is suitable for matching with the model utilized in this work. We used *Pleione* with the same seed for the random number generator so all GAs started from the same population of models. The resulting error convergence for each fitness function is shown in Figure 5-1A for the ten best models. To provide a fair comparison between each of the fitness functions, we report the fractional error that corresponds to the averaged error normalized by the average error at the first iteration for the elite of models. This procedure makes it possible to determine the fitness function characterized by the largest error reduction compared with its first iteration. For the tested model, calibration strategy, and fitness functions, the squared difference of averages (SDA, Equation 5-1), the normalized pair-wise squared deviation (NPWSD, Equation 5-5), the chi-square error (CHISQ, Equation 5-8), and the mean normalized standard error (MNSE, Equation 5-9) exhibited the best performances. These four fitness functions reduced the average error for the elite of models to nearly 20% of their original value (Figure 5-1A).

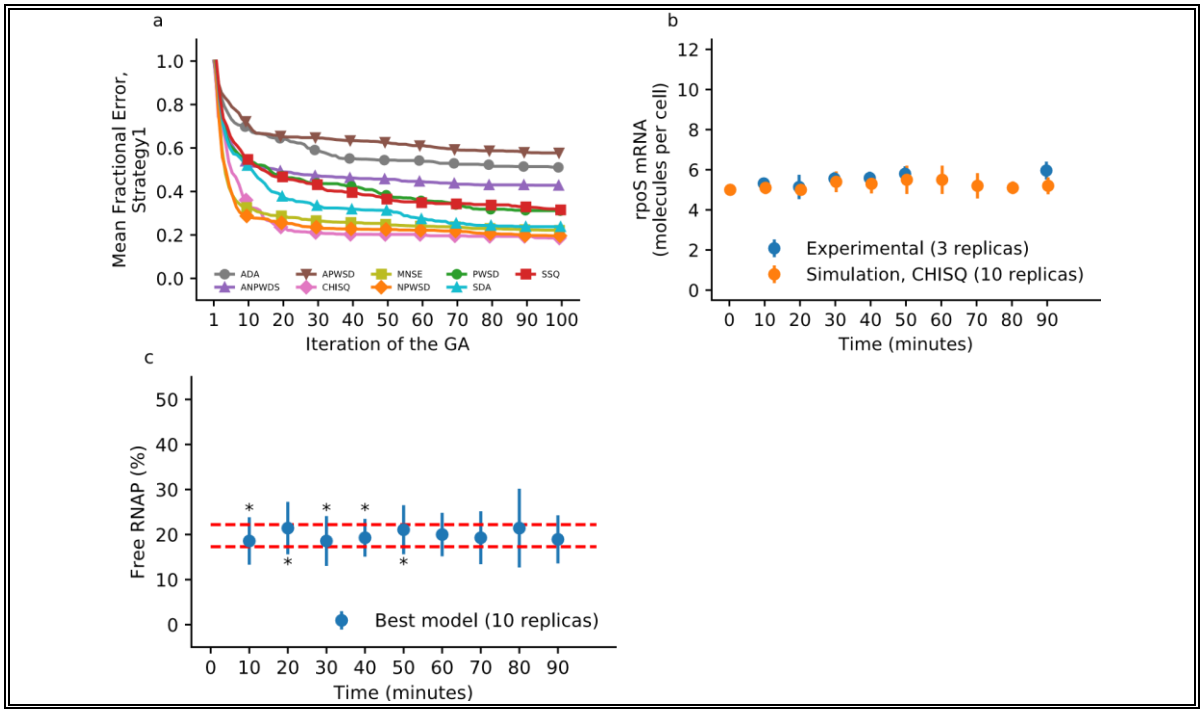


Figure 5-1. Parameter calibration using algebraic fitness functions.
The RBM representing the core GRN of *E. coli* was calibrated against transcriptome data and evaluated algebraically. (a) Error convergence. The traces correspond to the mean of the 10 best models (the elite population) per iteration, normalized by the mean error at the first iteration (fractional error). (b) Comparison of the best model. The 10 simulations used to evaluate the best model at the end of the GA employing the chi-squared fitness function are plotted along with experimental data for the rpoS mRNA. The symbols correspond to the mean and one standard deviation. Data were plotted purposely with an offset to prevent the error bars from overlapping. (c) Independent validation. The calibrated model predicts free RNAP in a pseudo-stationary state in the 17–22% range; stars denote where a one-sample t-test concludes the model simulates a larger value (at 10 minutes, p-value ≈ 0.041; at 30 minutes, p-value ≈ 0.049; at 40 minutes, p-value ≈ 0.036) and a smaller value (at 20 minutes, p-value ≈ 0.038; at 50 minutes, p-value ≈ 0.040) compared with the 17–22% interval.

Overall, we obtained good agreement between most of the experimental mRNA observations compared with their simulated values. Figure 5-1B shows the dynamics of the rpoS mRNA for the best-fit model based on CHISQ, and for all ten simulated mRNAs dynamics in Figure S 5-1. The remarkably poor fit for rpoA, rpoB, and rpoD mRNAs (Figure S 5-1 C, D, and F) may be explained by the lack of negative regulatory mechanisms in the model. There is a second regulatory layer in *E. coli* composed of antagonist proteins of sigma factor activity and a third layer that includes the antagonists of sigma-factor antagonists36. Incorporating those regulatory proteins into the model would benefit the calibration of mRNA responses that exhibit increased degradation and recovery, or vice versa. For instance, the fecI mRNA is rapidly synthesized and then degraded throughout the cold stress experiment (see Figure S 5-1A), however, the model predicted positive net synthesis. Moreover, the model that we used as an example has constant levels of proteins. Extending the model beyond gene regulation to incorporate protein translation, degradation, and metabolism would increase the repertoire of responses and regulatory mechanisms that are active in a given condition. Finally, the model represents a small portion of the complete *E. coli* GRN and ignores all transcription factors and small regulatory RNAs with known function (Hershberg et al., 2003; Keseler et al., 2017) that directly or indirectly affect the dynamics of the considered mRNA. For instance, the Rsd protein accumulates throughout exponential growth and sequesters the RpoD protein, enabling higher activity of the sigma factor RpoS that drives gene expression in the stationary growth phase (Piper et al., 2009). Another example is the

rpoH mRNA that responds to high temperatures by restructuring its folding and increasing its translation rate (Kamath-Loeb & Gross, 1991).

Complementary to the results, we also estimated the uncertainty on parameter values using leave-one-out jackknife and bootstrap resampling (Table S 5-1 and Table S 5-2). Furthermore, we provide the predicted values for all mRNAs, free and bound proteins, and protein complexes (40 variables) in Supplementary Video S1 (before calibration) and Video S2 (after calibration) online. The videos show that the model simulates a fast regime dominated by the formation of protein complexes during the first minute and then a slow regime dominated by the synthesis and degradation of mRNA later on. Small multiple snapshots of the model dynamics at 0, 1, 10, and 90 minutes are shown in Figure S 5-2. To validate with independent data, we compared the free fraction of RNA Polymerase (RNAP) with that quantified experimentally by Patrick *et al.* (Patrick et al., 2015). The free RNAP simulated fraction was in close agreement with the reported value. In this case, the best model selected by the CHISQ function predicted a free RNAP fraction in the 17–22% range (Figure 5-1C); an ANOVA test revealed no statistical difference among the nine simulated time points ($F(8, 81) \approx 0.4185$, p-value ≈ 0.9067). For comparison, Patrick *et al.* showed that the free fraction of RNAP in *E. coli* depends on the growth rate, and they reported a free fraction of 28% in fast-growing cells (Patrick et al., 2015).

5.4.1.2 Iterative equivalence tests as fitness functions.

Statistical tests are commonly used to determine significant differences between treatments. Usually, researchers use parametric tests such as the paired and unpaired t-tests or their non-parametric counterparts, the Wilcoxon rank-sum and the Mann-Whitney U-test, to determine if responses are different (Lauzon & Caffo, 2009). However, the applicability of those statistical tests for calibrating models simulated stochastically is questioned in the base of their null hypotheses. While common statistical tests determine whether two particular values such as the mean or variance from two distribution are statistically different, equivalence tests determine the opposite hypotheses. Equivalence tests aim to determine the significance of the extent to which two distributions can differ and also be equivalent for practical purposes (Cornell, 1990).

We incorporated three equivalence tests in *Pleione*: the parametric two one-sided *t*-tests (Schuirmann, 1987) and the non-parametric Double Mann-Whitney U-test (Cornell, 1990) and Wellek´s test (Wellek, 1996, 2010). The two one-sided t-tests and the Double Mann-Whitney U-test are straightforward implementations of the t-test and U-test employed to determine whether a distribution shifted to the left and the right is now smaller and greater respectively compared to the other distribution. If one of the shifted distributions lies outside the predefined equivalence range, the equivalence test cannot reject its null hypothesis. On the other hand, Wellek´s test determines whether the probability of one random variable being greater than another is between a (small) range centered at 50%40. *Pleione* calculates the selected test for each time point and variable, a procedure conducted to build a rejection matrix. Then, the iterative equivalence test is defined as the sum of the non-rejected equivalence tests (i.e., tests that do not conclude the distributions are equivalent). A characteristic of the above definition is that the iterative equivalence tests are discrete functions with known limits: A perfect model has a score equal to zero, and a completely wrong model score equals the number of variables times the number of experimental time points. For instance, if the experimental setup measured ten variables for ten time points, the maximum score is 100. Additionally, defined in a way that counts how many failed tests were, the iterative equivalence tests are fitness functions suitable to be minimized, therefore useful to improve stochastic models through an evolutionary algorithm or similar procedure.

To determine the applicability of these equivalence tests, we used Aguilera *et al.*´s simple model of gene regulation (Aguilera et al., 2017). These authors employed the model with known parameters to test their calibration method. Thanks to these known or "true" parameters, we constructed large batches of synthetic data. We randomly selected ten replications and calibrated the model using strategy 1. In general, the GA recovered the four "true" parameters. In the case of the two one-sided t-tests, the percent error between the found and "true" parameters was no greater than 13.5% (31.2% cumulated percent error for the four parameters) when using one standard deviation as symmetric equivalence range. Similarly, employing the Double Mann-Whitney U-test, the GA recovered all parameters reasonably well, with a cumulated percent error close to 40%. Finally, employing the Wellek´s test with equivalence probabilities in the range of 19–76% (as similarly done in

Wellek40) yielded two parameters with percent errors less than 2.0%. However, the other two parameters were approximately 88% of the value of the "true" parameters. All of the results are listed in Table 5-1. Next, we will show that combining the equivalence test with other search strategies and fitness functions greatly improved the calibration of the considered model. Also, results for the core GRN model calibrated with the Wellek´s test are shown in Figure 5-2 and Figure S 5-1. As depicted in Figure 5-2A, calibration with the Wellek´s test yielded a low reduction in error, despite a good agreement for rpoS mRNA (Figure 5-2B) and free RNAP (Figure 5-2C).

Table 5-1. Results of employing each equivalence test for the calibration of the Aguilera´s simple model.
Each strategy and fitness function (see the Methods section) was run once and the table shows the parameters of the best model at the end of the calibration. A comparison is shown for the percentual error compared to the "true" value of the parameters (r1_v = 5, r2_k1 = 0.03, r3_k1 = 0.1, r4_k1 = 0.03) and the cumulative error of all parameters.

	Fitness function	**Score**	**Parameters**				**Percent error**				**Cumulative error**
			r1_v	**r2_k1**	**r3_k1**	**r4_k1**	**r1_v**	**r2_k1**	**r3_k1**	**r4_k1**	
strategy1	TOST	4	5.672	0.034	0.097	0.029	13.43%	12.60%	−2.92%	−2.26%	31.21%
	DUT	3	5.672	0.034	0.094	0.028	13.43%	12.60%	−6.30%	−6.81%	39.14%
	WMWET	5	9.355	0.056	0.099	0.029	87.10%	87.77%	−1.47%	−1.71%	178.05%
strategy2	TOST	3	5.264	0.031	0.092	0.026	5.28%	3.22%	−7.79%	−12.49%	28.77%
	DUT	5	5.616	0.033	0.092	0.028	12.32%	10.99%	−7.79%	−6.93%	38.02%
	WMWET	4	5.275	0.032	0.094	0.028	5.49%	6.53%	−5.98%	−6.77%	24.77%
strategy3	TOST	5	8.493	0.050	0.216	0.015	69.85%	68.30%	116.40%	−49.29%	303.85%
	DUT	6	7.242	0.046	0.236	0.029	44.83%	52.36%	136.36%	−3.99%	237.54%
	WMWET	6	5.792	0.035	0.297	0.017	15.84%	17.43%	196.97%	−43.96%	274.20%
strategy3	TOST + SDA	8	5.065	0.032	0.099	0.029	1.29%	8.30%	−1.25%	−3.94%	14.78%
	DUT + SDA	7	6.117	0.037	0.091	0.027	22.34%	22.58%	−9.11%	−11.66%	65.70%
	WMWET + SDA	4	4.829	0.028	0.097	0.029	−3.42%	−6.89%	−3.46%	−3.71%	17.48%

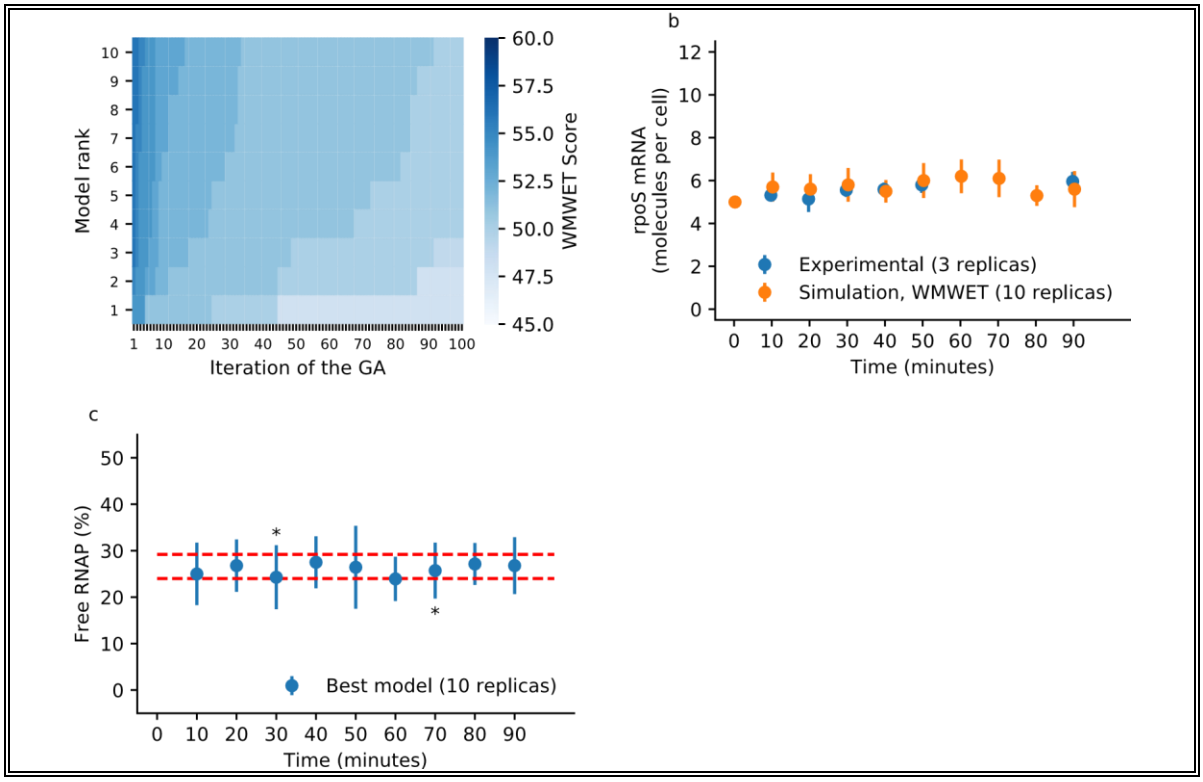


Figure 5-2. Parameter calibration using the non-parametric Wellek´s equivalence test. The RBM representing the core GRN of *E. coli* was calibrated against transcriptome data and evaluated algebraically. (a) Error convergence. The heat map corresponds to the value of the 10 best models (the elite population) per iteration. (b) Comparison of the best model. The 10 simulations used to evaluate the best model at the end of the GA are plotted along with experimental data for the rpoS mRNA. The symbols correspond to the mean and one standard deviation. Data were plotted purposely with an offset to prevent the error bars from overlapping. (c) Independent validation. The calibrated model predicts free RNAP in a pseudo-stationary state in the 24–29% range; stars denote where a one-sample t-test concludes the model simulates a larger value (at 30 minutes, p-value ≈ 0.039) and a smaller value (at 70 minutes, p-value ≈ 0.037) compared with the 24–29% interval.

5.4.2 Multiple-objective genetic algorithm

Discrepancies between fitness functions can be resolved using multi-objective calibration. We present the fitness calculated by one of each fitness functions included in *Pleione* for the core GRN model (Figure S 5-3). For nearly all of the fitness functions, the model with the lowest error was noted with the employed objective function as well with other fitness functions. This result reveals the strong correlation between the employed metrics (see Figure S 5-4). The included fitness functions accordingly consider and measure the

variability of data and simulations in different ways and justify the developed multi-objective capability in *Pleione*. There are multiple procedures for implementing such MOGAs, which were reviewed by Konak *et al.* (Konak et al., 2006). We included the sum of individual rankings per fitness function (without weighting) to determine the contribution of each metric to the overall error. To exemplify the use of the multi-objective capability of *Pleione*, we selected multiple fitness functions based on the Spearman´s ρ correlation coefficient calculated between all fitness at the first iteration (see Figure S 5-4). We could accordingly exclude fitness functions that would yield the same result after the calibration. In other words, the use of two highly correlated fitness functions is redundant. For instance, calibrations with the SDA (Equation 5-1) and with the sum of squares function (SSQ, Equation 5-7; see the Methods section) were highly correlated for the core GRN model; differences in the final error would be marginal if the user decided to use SDA instead of the other to calibrate the model. In the tested strategy 1, the fractional error for the elite of models employing SDA as fitness function was 23.8% while for SSQ was 31.6%.

We expect that combining fitness functions with low correlation among them will lead to better calibrations compared with using a single function. We found that the convergence behavior of some fitness functions in a MOGA was comparable or better to those in a SOGA (see Figure S 5-5). However, other fitness functions exhibited erratic behavior consistent with a search strategy that attempts to satisfy multiple objectives at the same time. One remarkable example was the CHISQ (Equation 5-8), where the error diverged compared with the single-objective optimization in the tested configuration. The fit of rpoS mRNA is shown online in Figure S 5-6 for two of the three tested calibration settings (see the Methods section, strategies 4 and 5). Comparisons between the MOGA and SOGA strategies are shown in Figure 5-3 and Figure S 5-7. Figures 5-3A, B, and S 5-6 show the mean ratio between a fitness function in a MOGA for the same fitness function in a SOGA for strategies 4, 5, and 6, respectively. In general, utilizing a MOGA reduced the calibration error (i.e., the ratios were below unity), but modest or no improvement was observed for the Wellek´s test compared with the situation where the Wellek´s test was used as a single objective. Although, the primary rationale for using a MOGA is that the resulting model is more robust because it minimizes multiple fitness functions simultaneously.

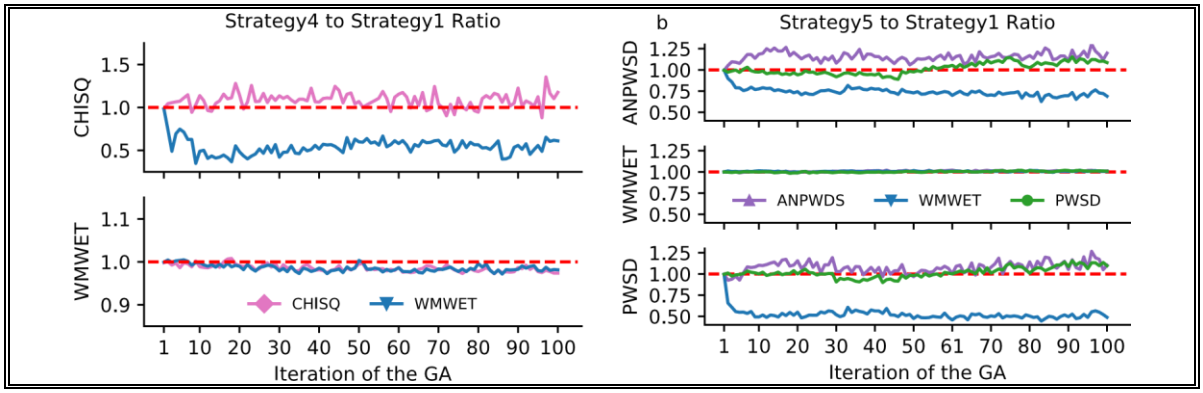


Figure 5-3. Comparison of MOGAs with their corresponding SOGAs.
The RBM representing the core GRN of *E. coli* was calibrated against transcriptome data. (a) Strategy 4 mean ratio. The panels show the behavior of the chi-squared (CHISQ, Equation 8) and the Wellek´s test (WMWET) fitness functions in a multi-objective optimization ratioed by the same error in the respective single-objective optimization (minimizing CHISQ, pink lines; or the Wellek´s test, blue lines). (b) Strategy 5 mean ratio. The panels show the behavior of ANPWSD (Equation 6), WMWET, and PWSD (Equation 3) in a MOGA divided by their corresponding error in a SOGA (purple lines: minimizing ANPWSD; blue lines: minimizing WMWET; green lines: minimizing PWSD). Values below 1.0 correspond to MOGA found an average error lower that SOGA. The legend indicates the objective function employed to calibrate the model.

5.4.3 Comparison with BioNetFit: The "example 6" model

Recently, Thomas *et al.* (Thomas et al., 2015) developed BNF intended for BNGL models, and Mitra *et al.* (Mitra et al., 2019) presented an improved version of Thomas´ software with more calibration algorithms and analysis (e.g., determination of the uncertainty of parameters values Daly et al., 2018). To perform a comparison between *Pleione* and BNF, we translated the "example 6" model (https://github.com/RuleWorld/BioNetFit) to PySB (Lopez et al., 2013) and then exported it to BNGL and kappa. The resulting models were calibrated with both BNF employing the network-free simulator NFsim and *Pleione* with the network-free simulators KaSim and NFsim, and the network-based Gillespie´s SSA within BNG2. Figure 5-4A shows the square root of the SDA (Equation 5-1) for the best model achieved at each iteration. Our calibrations reveal that the error after calibrating the kappa "example 6" model is similar to the calibration with BNF. A comparison of the first and last

iterations is shown in Figure 5-4B, where each dot represents an independent calibration. The calibration with BNF yielded the smallest error. However, BNF only calibrates BNGL models; *Pleione* introduces the necessary methods to calibrate *kappa* RBMs. Moreover, we tested the equality of the independent runs using the Kruskal-Wallis H-test. We concluded that there was no significant difference between the lowest model errors at the last iteration ($H(4) \approx 4.128$, p-value ≈ 0.248). This result further supports the notion that *Pleione* can perform as well as available tools for finding a candidate model. Notably, *Pleione* extends the use of GAs to a second RBM language as well to other stochastic simulators such as KaSim, PISKaS, and NFsim.

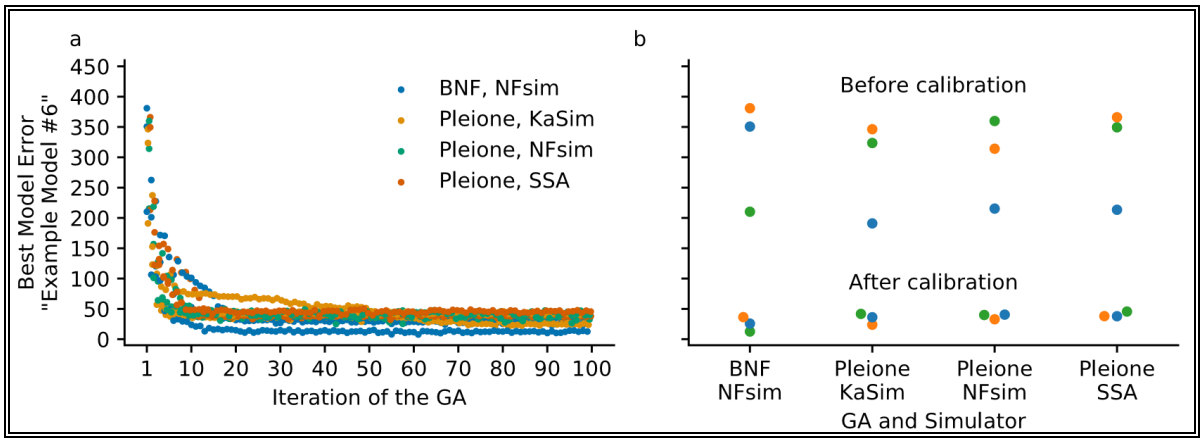


Figure 5-4. Comparison of calibration with BNF and Pleione.
The "example 6" model from Thomas *et al.* (Thomas et al., 2015) was calibrated with synthetic data provided by the authors. Left. The traces correspond to the square root of the SDA (Equation 5-1) for the best model per iteration. Each GA was run three times with the same initial population of models for each stochastic simulator. Differences at the first iteration are due to the stochastic simulations of each model. Each model was simulated with the network-free simulators KaSim and NFsim and with the Gillespie´s SSA within the BNG software. Right. The dots indicate the error achieved at the last iteration (lower group) and the initial error at the first (blue, seed = 0), second (orange, seed = 2), and third independent calibration (green, seed = 2596283685).

5.4.4 Calibration of the core GRN model without elitism

Multiple strategies have been developed to select individuals and generate new solutions from them with GAs (Whitley, 1994). To calibrate RBMs, BNF selects individuals to recombine from the entire population; *Pleione* chooses individuals from an elite of models. Furthermore, BNF prohibits self-recombination by default while *Pleione* allows it. Moreover, another difference between the approaches pertains to the probability distribution used to select a model as a parent. In BNF, the selection is inversely proportional to the rank (see the BNF Manual, Thomas et al., 2015); *Pleione* selects individuals from a uniform distribution within the elite of models. To analyze thoroughly the developed method, we also calibrated the core GRN model with *Pleione* following an inverse rank strategy to select parents within the GA. This technique is referred to as strategy 2 (see the Methods section). For instance, from a population of 100 individuals, the first-ranked model has a probability of 19.28% of being selected as a parent; the second-best model has a probability of 9.64%.

The calibration results of the core GRN model with all fitness functions are shown in Figure 5-5A for the elitist GA (see the Methods section; strategy 1), in Figure 5-5B for the non-elitist GA (strategy 2), and in Figure 5-5C for strategy 3, the selection and recombination strategy most similar to BNF. Notably, the results revealed that the selection of parents implemented in *Pleione* (referred to as strategy 1) outperformed the selection of parents implemented in BNF (strategy 2) for the core GRN model (Figure 5-5B). That is, the selection of individuals from an elite of models was more beneficial to find better solutions than selecting individuals from all models. These results must be specifically interpreted for the considered model in which the number of parameters is nearly 13 times larger than in the "example 6" model. Moreover, the parameter search benefits from other features in BNF that recombines two individuals in multiple points, not just one as initially tested. We performed a calibration employing an inverse to the rank selection, a multiple crossing-over, and a mutation strategy within a ±10% range of the original parameter value (strategy 3). The results, shown in Figure 5-5C, expose that the strategy used in BNF performed better for the considered model. For instance, the fractional error at the final iteration with the CHISQ error was roughly 5% of the original error for all models. That is

nearly a six-fold improvement compared with strategy 1, where the fractional error was 29% for all models.

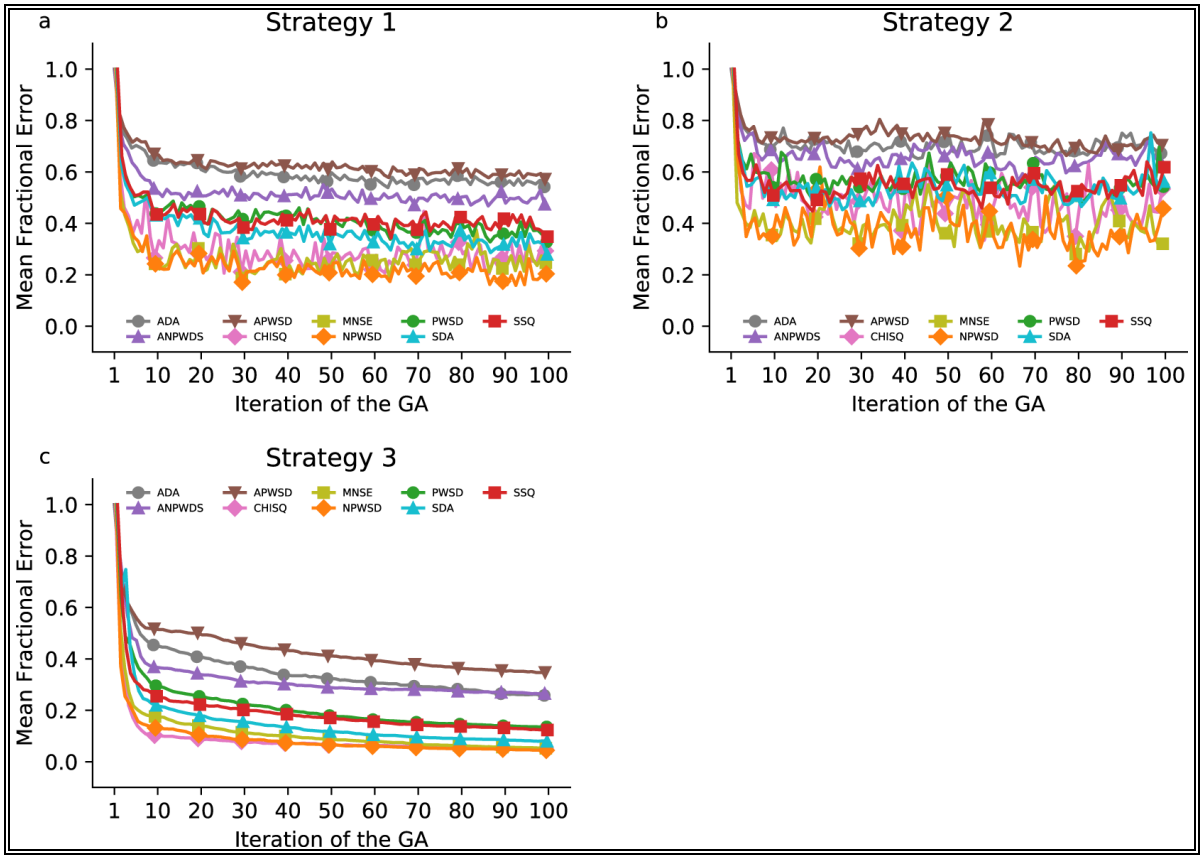


Figure 5-5. Calibration of the core GRN model with strategies 1, 2, and 3.
The RBM representing the core GRN of *E. coli* was calibrated against transcriptome data and evaluated individually. (a–c) Error convergence in a single-objective calibration. The traces correspond to the mean of all models per iteration, normalized against the maximum value achieved at the first iteration (fractional error).

Complementary to the calibration of the core GRN model, we calibrated the simple model of gene regulation with each one of the equivalence tests using strategy 3. In the case of the Wellek´s test, the percent error for each parameter was as low as 3.42% and as high as 6.89%, with a total percent error of 17.48%, but only when combined with SDA (Equation 5-1). In general, strategy 3 exhibited the worst performance than strategy 1 at recovering the "true" parameters using equivalence tests. In the case of Wellek´s test, the total percent error

was increased from 178% (strategy 1) to 274% (strategy 3) and in all cases reduced by employing a second fitness function (e.g. SDA, Equation 5-1) for calibration employing strategy 3 (see the Methods section). All of the results are listed in Table 5-1.

5.5 CONCLUSION

Parameter estimation is a common problem when developing predictive models in systems biology. In the case of stochastic simulations, the problem is magnified when data variability is disregarded using deterministic simulations to reduce simulation time or when fitness functions disregard simulation variability even when the model is simulated stochastically. Our method, *Pleione*, solved these two issues. *Pleione* takes less time to calibrate as the increasing availability of CPUs reduces the burden of multiple stochastic simulations, leveraging the need for deterministic simulations of RBMs. Parallelization permits *Pleione* could use a sufficient number of stochastic simulations to be compared with experimental data. Regarding fitness functions, we propose using equivalence tests to determine the similarity of data and simulations. Although we included parametric and non-parametric tests, the Wellek´s test makes it possible to render a confident assessment of the pertinence of a few stochastic simulations to reproduce a small number of experiment replications. Moreover, our approach provides support to kappa RBMs and can select models according to multiple metrics that overcomes the drawbacks of each fitness function.

One limitation of our method is the source of noise as *Pleione* does not currently consider experimental variability. Also, *Pleione* does not perform identifiability and parameter uncertainty determination on its own. We developed *Alcyone* (see the Methods section) to supplement this deficiency using jackknife and bootstrapping methods. However, Approximate Bayesian Computation (ABC, reviewed by Warne et al., 2019) has strengths in terms of identifiability and parameter uncertainty. ABC methods are preferred, but they take exceedingly long computational times as the discrepancy threshold is lower (as shown by Warne et al., 2019), and selecting the discrepancy metric may be non-trivial.

Features within *Pleione* are extensible. The incorporation of a noise model into simulated data, new fitness functions, and other statistical approaches like the ones introduced by Liu and Faeder (B. Liu & Faeder, 2017) and others (Warne et al., 2019) are

considered and will take advantage of parallelization of simulations and calculations. Finally, *Pleione* is part of a larger project that will introduce common procedures to analyze RBM like swarm particle optimization (Benuwa et al., 2016) (already present in Mitra´s pyBNF), maximum likelihood optimization (Daigle et al., 2012), and sensitivity analysis (Campolongo et al., 2007; Kent et al., 2013) of parameter values, all based on parallelization of stochastic simulations. We also expect that *Pleione* will facilitate broader adoption and development of RBMs to reproduce the structure and dynamics of complex biological systems.

6. CHAPTER V: *STEROPE*: AN OPEN-SOURCE PACKAGE FOR GLOBAL SENSITIVITY ANALYSIS OF RULE-BASED MODELS

Rodrigo Santibáñez[1,2], María Rodríguez-Fernández[3], Daniel Garrido[2], Alberto J.M. Martin[1]

[1]Network Biology Lab, Centro de Genómica y Bioinformática, Facultad de Ciencias, Universidad Mayor, Santiago, Chile

[2]Department of Chemical and Bioprocess Engineering, School of Engineering, Pontificia Universidad Católica de Chile, Santiago, Chile

[3]Instituto de Ingeniería Biológica y Médica, Pontificia Universidad Católica de Chile, Santiago, Chile

Manuscript under preparation.

6.1 SUMMARY

The development and use of computational models is a foundation pillar in systems biology. More and more models have been employed to describe metabolism, gene regulation, and other processes ranging from few genes to complete genomes in a myriad of conditions and cell types. Among modeling techniques, Rule-Based Models have been proposed as tools able to model and simulate complex systems unattainable by other modeling approaches. Despite this, rule-based models are not widely adopted and one reason for this is the lack of analysis tools that are common to other techniques and allow further understanding of the model structure and dynamics.

In this article, we present *Sterope*, a tool for the efficient calculation of global sensitivity indices of rule-based model parameters. *Sterope* relies on the stochastic simulator KaSim to obtain rule execution counts and influences to estimate elementary, interaction, and total-effects sensitivities. In addition, *Sterope* distributes simulations and calculations enabling the analysis of complex models. Using models of varying complexity, we describe how to use *Sterope* to determine parameters that control the dynamics of a model and the practical consequences for experimental determination, calibration, and process

optimization. We attained satisfactory results even considering tens of thousands of samples and for models with long simulation times, a large number of parameters, and rules. Finally, *Sterope* is easy to configure and run, and we expect to contribute with it to the wider adoption of rule-based modeling in systems biology.

6.2 INTRODUCTION

Systems biology relies on the development and use of computational models to understand the role of the different components of a biological system and to provide hypotheses for further experimental testing (Endy & Brent, 2001). Most of the models developed use Ordinary Differential Equations (ODEs), a technique for which a body of simulators and robust analysis methods exists (Szigeti et al., 2018). However, ODE models are not appropriated in all situations and Rule-Based Models (RBMs) have been proposed to solve those issues (Chylek et al., 2015). This kind of models describes biochemical reactions by successive statements called *rules* that closely resemble chemical equations (Daniel T. Gillespie, 2007, 1977), but they encode the sufficient requirements for a reaction to happen (Danos et al., 2007a). In doing that, RBMs compress into a few *rules* thousands or even millions of individual reactions (Danos et al., 2007a; James R Faeder et al., 2003b).

Despite the fact that RBMs could be transformed into ODE models and simulated (Aguilera et al., 2017; Kozer et al., 2013), this transformation is not always attainable and RBMs must rely on stochastic simulation through the Gillespie´s Stochastic Simulation Algorithm and derivatives (Daniel T. Gillespie, 1977; Hogg et al., 2014; McCollum et al., 2006). The critical reason for unsuitable deterministic simulations of RBMs is the combinatorial explosion. Firstly, the combinatorial explosion refers to the excessively large number of complexes where a single modeled component could participate. This issue could impact negatively on deterministic simulations and analyses of the dynamics (Hogg et al., 2014). For example, an RBM proposed for the EGFR signaling pathway required 70 *rules* that describe the dynamics of approximately 10^{23} different components (Danos et al., 2007a). In addition, a second reason to avoid the deterministic simulation of RBMs is that biological systems show stochastic dynamics (Raj & van Oudenaarden, 2008; Wilkinson, 2009) that a deterministic simulation will necessarily disregard. However, doing a stochastic simulation

and subsequent analysis imperatively requires tailored methods and software to simulate and analyze efficiently many repetitions of the same model. The lack of core analysis methods impedes the application of RBMs for the further understanding of biological systems besides the simple reproduction of observed or experimental data.

We present *Sterope*, an open-source software to perform a global sensitivity analysis (Kent et al., 2013) of RBMs. Methods for global sensitivity analysis define samples in a selected region of interest (ROI) and analyze the output of all samples to estimate sensitivity indices (Saltelli et al., 2005). *Sterope* relies on KaSim (Danos et al., 2007a) to retrieve a numeric representation of the *rules* execution counts and influences through a defined time frame. The analysis is formally called Dynamic Influence Network (DIN, Forbes et al., 2018) and the sensitivity indexes of *rules* influences are comparable with the determination of flux control coefficients in Metabolic Control Analysis (MCA, Tomar & De, 2013), although they come from different modeling and algorithmic approaches. *Sterope* contrasts with R4Kappa (Sorokin et al., 2019) which allows the analysis of snapshots, a graph of modeled components with their non-zero quantities at a certain simulated time, but potentially suffer from combinatorial explosion. Another feature of *Sterope* is its ease of set up configuration and the distribution of simulation and calculations, a feature that also contrasts with R4Kappa that do not easily distribute simulations and calculation of sensitivity indices. Also, R4Kappa do not generalize the analysis, while *Sterope* calculate sensitivity indices for all defined rules in the model.

Through example models of varying complexity degree, we describe the benefits of doing global sensitivity analysis of RBMs using *Sterope*. We analyzed two main cases: when parameters are known and sensitivity indices gave insights to hypothesize interventions for process optimization and allow the determination of the calibration quality; and a second case where the analysis of unknown parameters determines priority for experimental efforts and subsequent computational calibration.

6.3 MATERIAL AND METHODS

6.3.1 Implementation

Sterope is an open-source python package. It performs a global sensitivity analysis through four steps (Figure 6-1). In the initial step, the user sets up the problem definition inside the text file that encodes the RBM and executes *Sterope* with mandatory options: model and time to simulate. In the following step, *Sterope* submits samples for repeated and distributed simulation that follow one of seven methods: Sobol´s method (Sobol, 2001), Random Balance Designs-Fourier Amplitude Sensitivity Test and Fourier Amplitude Sensitivity Test (Cukier et al., 1973), Delta Moment-Independent Measure (Borgonovo, 2007), Derivative-based Global Sensitivity Measure (Sobol´ & Kucherenko, 2009), Fractional Factorial Sensitivity Analysis (Saltelli et al., 2008), or the Morris´ method (M. D. Morris, 1991). Next, the DINs of each sample are bootstrapped to reduce the bias of having few stochastic simulations. Finally, *Sterope* performs parallelized analysis of the bootstrapped DINs (one per parameter for *rule* execution, and the square of the parameter number for *rule* influences) and reports the sensitivity indices (including confidence intervals if the selected method calculates them) in text files. The Sobol´s method is the default analysis method and with it, *Sterope* reports first order (elementary effects), second order (interaction effects), and total effects. Parameter samples and sensitivity indices are calculated through the Sensitivity Analysis Library (SALib) python package (Herman & Usher, 2017), and *Sterope* bridges the samples to sensitivity calculation through the simulation and bootstrapping of the samples. Distributed simulation, bootstrapping, and indexes calculation is achieved through the Dask python package (Rocklin, 2015), and therefore, *Sterope* is compatible with High-Performance Computing infrastructures if the SLURM Workload Manager is installed. We also provide support for sensitivity analysis of the flux control coefficients from a Metabolic Control Analysis (Tomar & De, 2013), through the use of the libRoadRunner python package (Somogyi et al., 2015). For further details, please see the Manual at https://sterope.readthedocs.io.

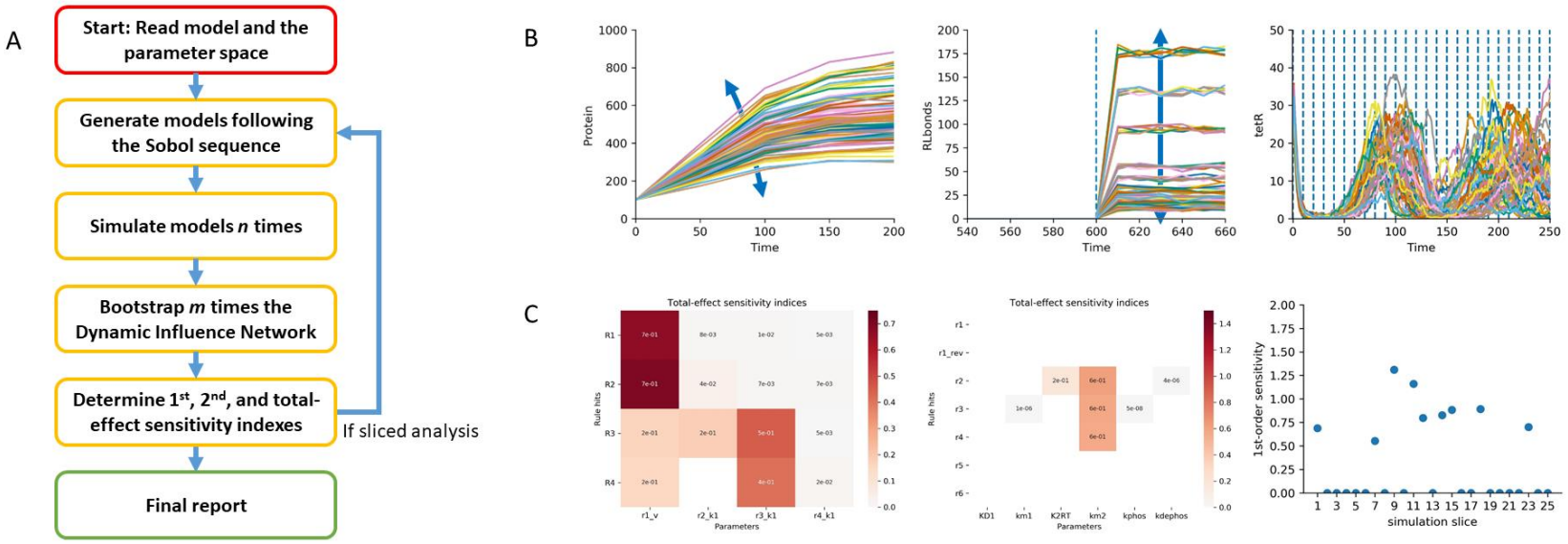


Figure 6-1. Workflow employing Sobol´s method and results from three global sensitivity schemes.
Panel A describes the main steps performed by *Sterope* from reading the model to finalize reporting the sensitivity indexes. Panel B shows three different schemes to calculate sensitivity indexes. From left to right, the first scheme employs the complete trajectory from the simulation; the second employs a fraction of the simulation, useful to understand the importance of parameters in a relevant period; and the third, a generalization of the second case, useful when the full dynamics could bias the sensitivity index. Each line represents the average of ten simulations. Panel C shows the total-effect sensitivity indexes obtained from analyzing the corresponding DINs. In the case of the last example, the windowed analysis corresponds to periods of ten time units, with steps of ten units of time.

6.3.2 Example models and configuration

Models analyzed in this work include the following:

1. The Aguilera´s simple model (Aguilera et al., 2017), a representation of constitutive gene expression modeled by four *rules* and four parameters of known value that represent the synthesis and degradation of an mRNA and its encoded protein.

2. The oscillatory *repressilator* model (BIOMD0000000012). The SBML model was imported and exported to *kappa* employing the PySB python package (Lopez et al., 2013). It is a model of three mRNAs and their encoded proteins which repress the synthesis of one mRNA forming a repression cycle that oscillates indefinitely with dynamics modeled by 12 *rules* (six synthesis rules and six degradation rules for mRNA and proteins) controlled by seven parameters of known value.

3. The Thomas´ example6 model obtained from the BioNetFit GitHub repository (github.com/RuleWorld/BioNetFit). The model reproduces ligand-induced phosphorylation of a receptor modeled by seven *rules* and six parameters of unknown values. The model was previously exported to *kappa* (Santibáñez et al., 2019) employing the PySB python package (Lopez et al., 2013).

4. The core Gene Regulatory Network (GRN) of *Escherichia coli* (Perez-Acle et al., 2018). The model reproduces the synthesis and degradation of ten genes controlled by 28 gene regulations of seven RNA Polymerases in 142 *rules* and 79 parameters of unknown value that were selected for analysis. The parameters control protein-protein interaction rates, RNA Polymerase binding and unbinding rates to promoters, and the degradation rates of the modeled mRNAs.

The first two models were analyzed in the ±20% range of the known parameters, while the Thomas´ and the core GRN models were analyzed in the 10^{-2}–10^{+2} range due to uncertainty of the true parameter values. The models were analyzed with the Sobol´s method which requires $N(2k + 2)$ samples were k is the number of selected parameters to analyze, to calculate first and second order, and total-effect sensitivity indices. For all models, N was equal to 1000, except for the core GRN model were N was 100 due to RAM limitations. Simulation time was set accordingly to each model and the DIN analysis comprised the

whole length of the simulation. In the case of the *repressilator* model, the sensitivity analysis was performed further in framed windows of 10 units of time. To compare the obtained sensitivity indexed to the ones obtained with MCA, flux control coefficients were obtained with the *getUnscaledFluxControlCoefficientMatrix* method of the libRoadRunner python package (Somogyi et al., 2015).

Further information about the configuration of models and analysis is presented in Supplementary File S1 online at https://figshare.com/articles/File_S1_zip/9941927

6.4 RESULTS

We exemplify the utilization of *Sterope* for two applications of sensitivity analysis; process optimization by identifying parameters with important sensitivity indices, and second, to simplify calibration by fixing parameters with low sensitivity indices and prioritizing them for experimental determination.

6.4.1 Sensitivity analysis of known parameter values

The sensitivity analysis for the Aguilera´s model is shown in Figure 6-2 and for the *repressilator* model in Figure 6-3. Both analyses were done for selected parameters, determining the sensitivity indices with respect to the number of *rules* executions (*hits*) and for the *rule* influence on all *rules* (including itself). In the case of the Aguilera´s model of gene expression, the sensitivity analysis revealed that the synthesis rate of the mRNA (*r1_v* parameter) affects the most over the execution of every modeled *rule* (Figure 6-2A), particularly to those *rules* referred to the synthesis and degradation of the mRNA (*rules* R1 and R2). Additionally, the degradation rate of the mRNA (*r2_k1* parameter) affects the synthesis (and degradation) of the encoded protein (*rules* R3 and R4). Similarly was the case of the synthesis rate of the protein (*r3_k1* parameter) that is sensitive and controls its own execution (*rule* R3) and degradation (*rule* R4). Interestingly, all second-order indices were not significant as they were estimated with a 95% confidence interval that includes zero, although, all total effects were significant (Figure 6-2B). In the case of *rules* influence, the sensitivity analysis indicated that few influences are significantly controlled by parameter values (Figure 6-2C). In particular, the synthesis and degradation rates of mRNA (*r1_v* and *r2_k1* parameters) are sensitive parameters determining the influence of the synthesis of

mRNA (*rule* R1) and its degradation (*rule* R2) over the execution of mRNA synthesis and protein synthesis (*rule* R3). Similarly, the first-order sensitivity indexes of protein synthesis and degradation rates (*r3_k1* and *r4_k1* parameters) affected locally the influence of protein synthesis and degradation over its degradation. In turn, total-effect indexes revealed that both all parameters contributed to the influence of all *rules* (Figure 6-2D). Comparatively, an MCA analysis (Figure 6-2E) showed an agreement of first-order sensitive parameters (Figure 6-2A) and with the most important total-effect sensitive parameters (Figure 6-2B).

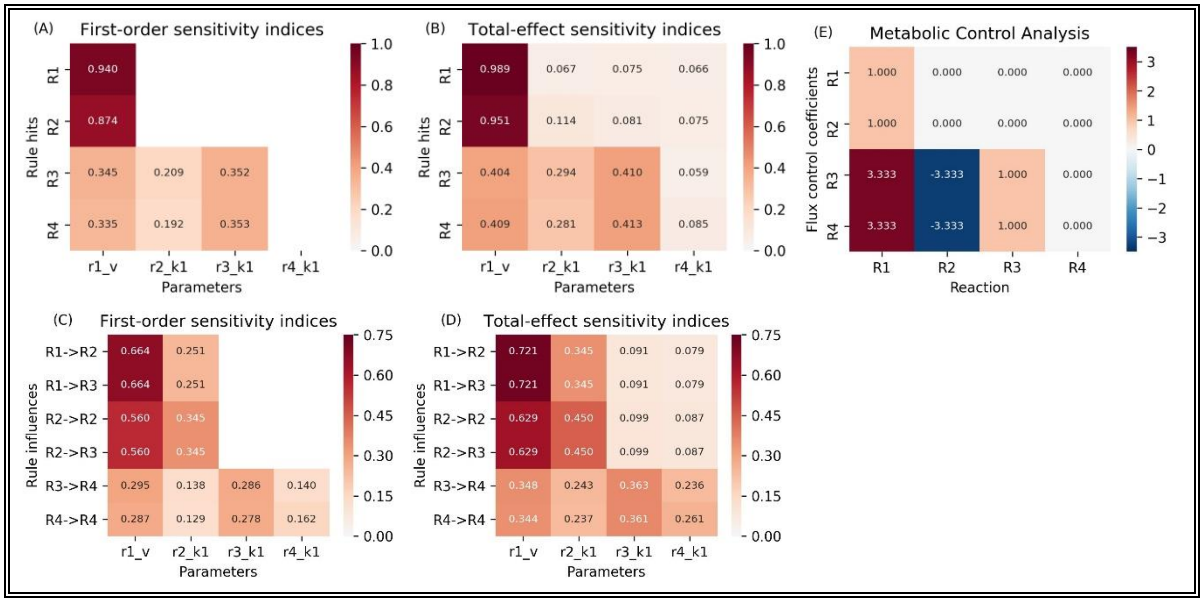


Figure 6-2. Global Sensitivity Analysis for Aguilera´s simple model of gene expression. Panels A and C show the elementary effects of the four parameters corresponding to the synthesis (*r1_v*, *r3_k1*) and degradation (*r2_k1*, *r4_k1*) rates of the mRNA and its encoded protein, respectively. Both figures omit non-significant sensitivities as reported by Sobol´s method (with a 95% confidence level). Similarly, panels B and D show total-effects sensitivity indices. None of the interaction effects (second-order sensitivity indices) were significant. Panel E shows the flux control coefficients from an equivalent model analyzed with the libRoadRunner python library.

The sensitivity analysis of the *repressilator* model gave insights into which parameters control the observed behavior of the model. In this case, the Hill coefficient (the parameter *n*) is the most important parameter as it affects every *rule* execution (Figure 6-3A). Also, the sensitivity analysis revealed that the degradation rate of proteins also impacts

the execution count of each *rule*, except for the synthesis and degradation of the cI mRNA (*Reaction3* and *Reaction6*) and the cI protein (*Reaction9* and *Reaction12*). However, the sensitivity indexes for mRNA and protein synthesis were closer to zero compared to the Hill coefficient sensitivities, indicating the low importance of those parameters. In the case of the second-order sensitivities, the analysis revealed that affected the synthesis and degradation of the cI mRNA and protein (Supplementary File S1). Also and similarly to the Aguilera´s simple model, all total-effect sensitivity indices were significant (Figure 6-3B). In the case of influences, the Hill coefficient (*n*) was again the most important parameter controlling the observed dynamics (24 of the 144 total *rules* influences, Figure 6-3C), followed by the degradation rate of the mRNAs (12 of the 144 influences) and then by the degradation rate of the proteins (2 of the 144 influences). Finally, the magnitudes of the sensitivity indices are close to zero (Figure 6-3C), revealing the low importance of mRNA and protein degradation rates to control *rules* influences.

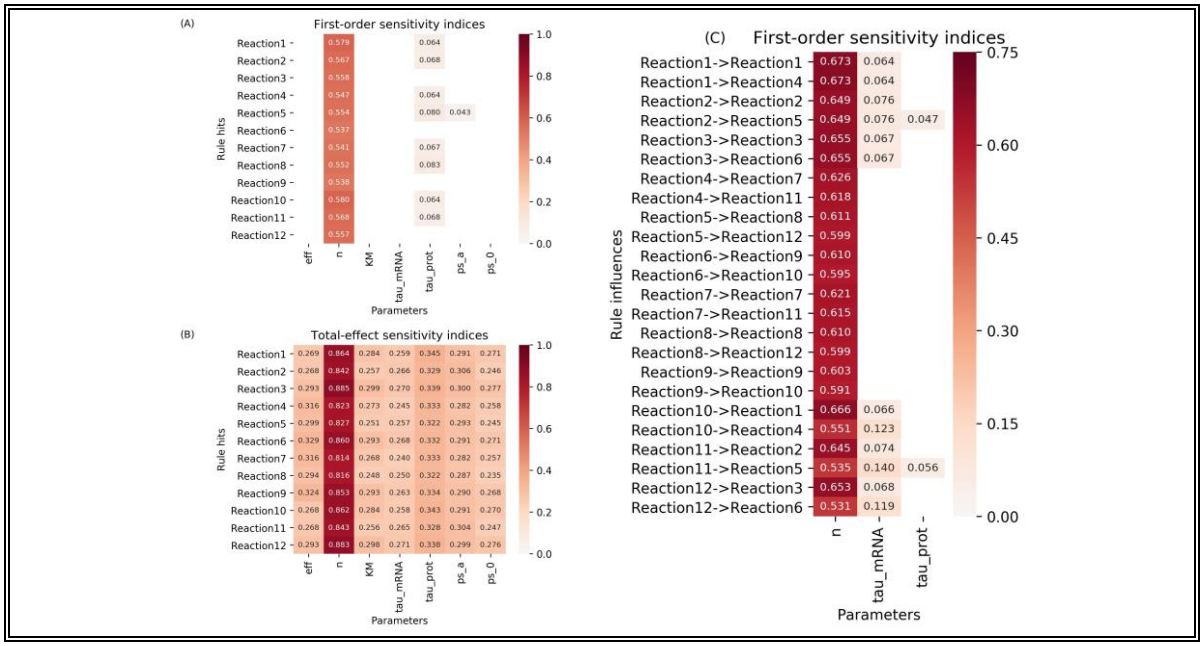


Figure 6-3. Global Sensitivity Analysis for the *repressilator* model.
Panels A and B show first-order sensitivity indices for the execution of Rules, while Panel C shows the same index for Rules influences over the execution of other Rules. Panel A and C omit indices that are non-significant.

6.4.2 Sensitivity analysis of unknown parameter values

Sensitivity analysis can also be used before model parameterization to prioritize parameters for calibration (those with large sensitivity indices) or experimental determination of those with low sensitivity. To demonstrate the use of *Sterope* for this purpose, we analyzed two models: the Thomas´ model consisting of six parameters originally calibrated in the 10^{-2}–10^{+2} ROI (Thomas et al., 2015), and the core GRN model analyzed within the same ROI.

For the Thomas´ model, the sensitivity analysis revealed that the *KD1* parameter that corresponds to the reversible binding of the ligand to a free receptor and the *km2* parameter that controls the unbinding of the receptor dimer are the most sensitive parameters. The *KD1* and the *km2* parameters are followed in importance by the *kphos* and *kdephos* parameters that control the phosphorylation and dephosphorylation of the receptor, respectively (Figure 6-4A). Similarly, the same parameters showed the greatest total-effects, although all parameters contribute to the observed dynamics (Figure 6-4B). The sensitivity analysis revealed only two parameter interactions (for instance, the *kphos* and *kdephos* parameters over the phosphorylation and dephosphorylation rules, respectively) with a confidence range above zero (Supplementary File S1 online). In the case of the influence of *rules*, the *KD1* and the *K2RT* parameters, the last corresponding to the receptor-receptor interaction, are the most important parameters with the larger sensitivity indices (Figure 6-4C and 6-4D). Again, analysis of the *kphos* and *kdephos* parameters showed their values control the self-influence of the dephosphorylation and phosphorylation rules, respectively.

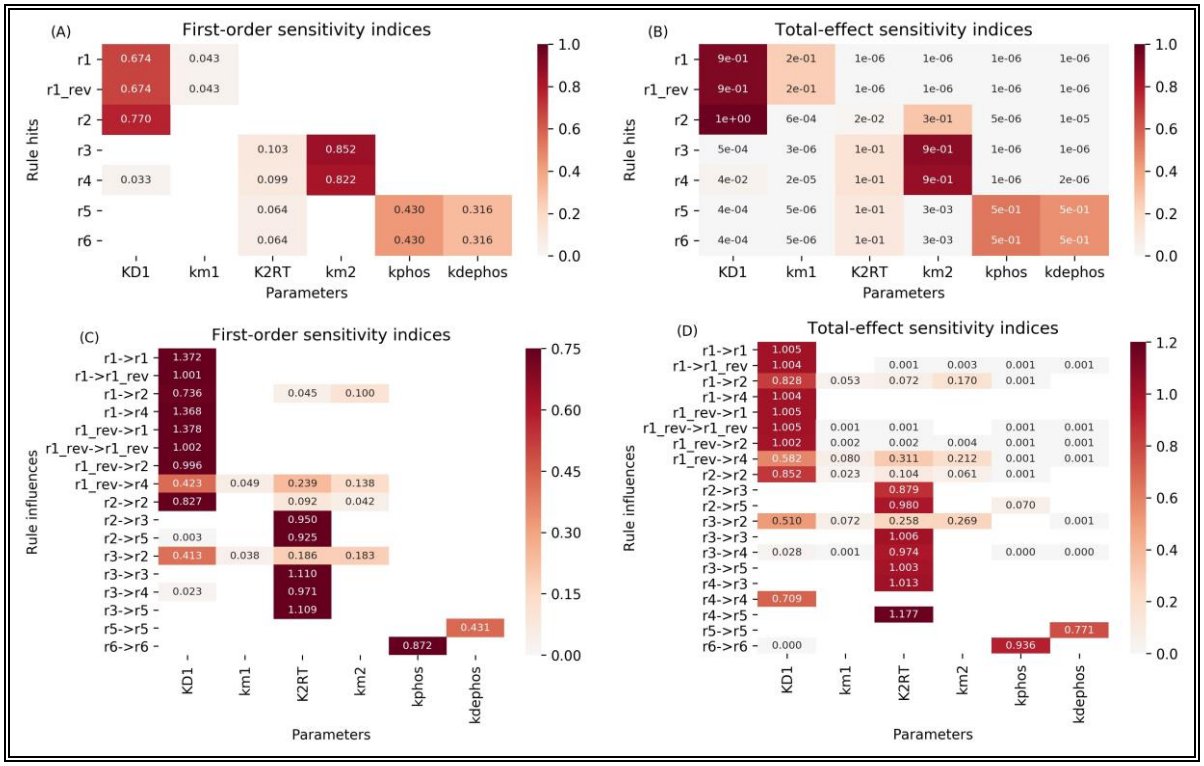


Figure 6-4. Global Sensitivity Analysis for the Thomas´ example6 model of ligand-induced phosphorylation of a receptor.
Panels A and C show the elementary effects of the six selected parameters that correspond to ligand-receptor binding and unbinding, receptor dimerization in the presence and absence of its ligand, and phosphorylation of the ligand-receptor and dephosphorylation of the receptor. Both figures omit non-significant sensitivities as reported by Sobol´s method (with a 95% confidence level). Similarly, panels B and D show total-effects sensitivity indices. None of the interaction effects (second-order sensitivity indices) were significant.

With respect to the sensitivity analysis of the core GRN model, it revealed that parameters act locally on the execution of *rules*. Also, the analysis revealed *rules* are affected significantly by the contribution of all parameters (Figure 6-5). In addition, we found 2852 interaction effects (of 3081, 92.5%) that showed at least one sensitivity index with a 95% confidence range excluding zero. Those 2852 interactions control the execution of 114 *rules* (80.3% of the total). Interestingly, sensitivity analysis for the *rules* influences showed that all parameters have at least one significant first-order sensitivity index, but they control 4.6% of the 20164 possible influences.

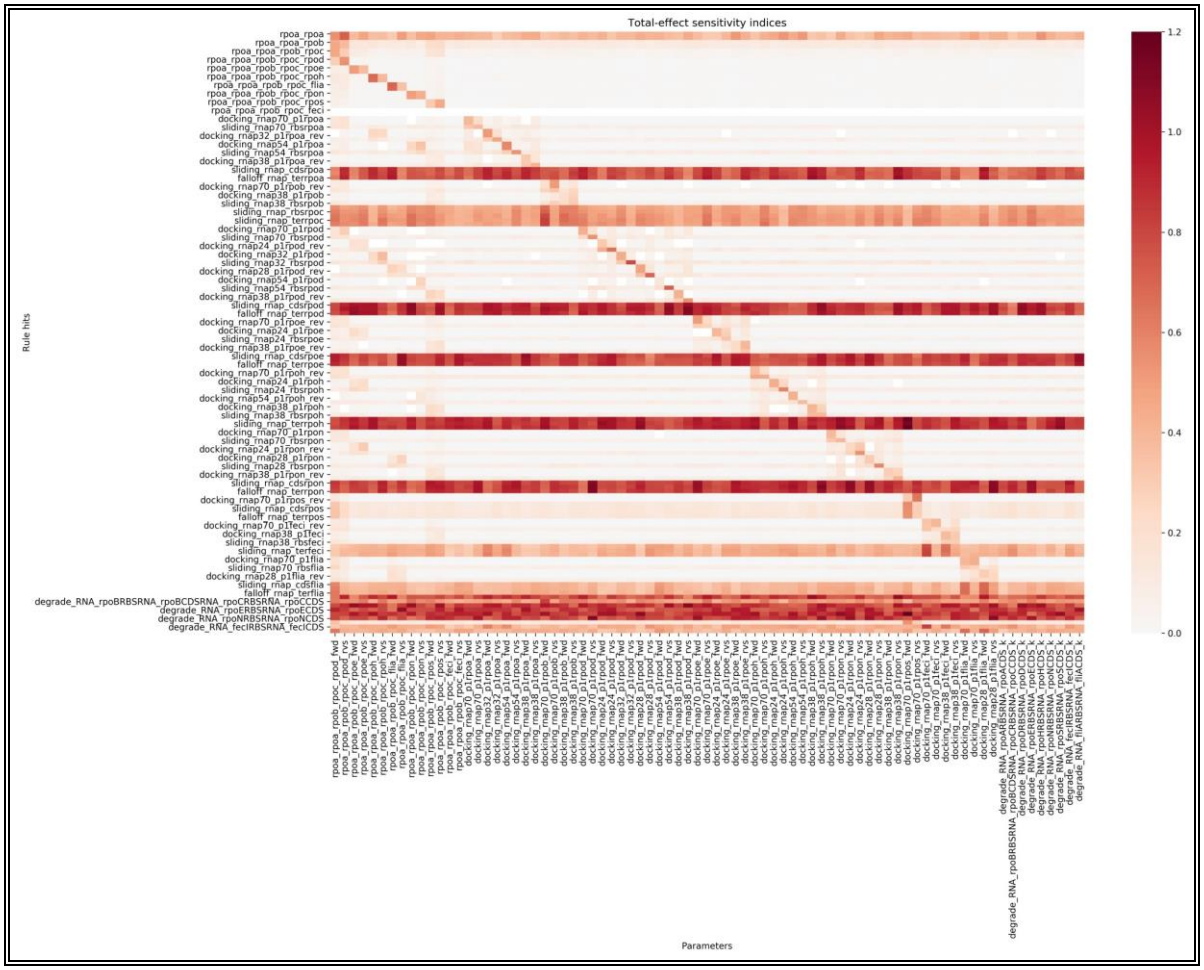


Figure 6-5. Global Sensitivity Analysis for the core Gene Regulatory Network.
The figure shows the total-effect sensitivity indices for the 79 analyzed parameters, that correspond to 14 protein-protein interaction rates (binding and unbinding), 56 RNA Polymerase and promoter interactions (binding and unbinding), and 9 degradation rules (the ten genes are organized in 9 operons). Sensitivity indices were calculated over the range from 10^{-2} to 10^{+2}.

6.5 DISCUSSION

Sterope is an open-source software package able to determine global sensitivity indices of RBMs. It solves the necessity of having an easy to set up software while achieving parallel simulations and calculations of global sensitivity indices. Although there are several software tools that perform a global sensitivity analysis of deterministic simulations, as far as we know, R4Kappa is the only method specially designed to analyze RBMs. While

R4Kappa allows sensitivity analysis of the model components and their dynamics through the simulation (employing the *sensitivity* R package), in contrast, *Sterope* analyzes the interactions of parameters and *rules* execution counts and influences. The selected approach for *Sterope* differs greatly and is more appropriate to analyze the complete structure of the RBMs, due that employing R4Kappa, the combinatorial explosion forces the selection of one modeled component at a time to analyze sensitivity indexes of parameters. *Sterope* uses state-of-the-art methods to determine the sensitivities of RBMs parameters. The method allows the determination of non-sensitive parameters through the analysis of a few samples but requires many samples to narrow the confidence interval, for what parallel computing is indispensable.

We analyzed four RBMs with varying degrees of complexity and the number of selected parameters. We achieved fast analyses through the implementation of parallel computing approaches, allowing the analysis of tens of thousands of samples to determine the parameter sensitivities indices (log files describing total time are in the Supplementary File S1 online). *Sterope* was able to analyze and find which parameters control the observed dynamics in a reasonable time, but not memory. The analyzed models correspond to two applications of sensitivity analysis; when parameter values are known with low uncertainty, those parameters with high sensitivity could be selected and intervened to modify the dynamics in order to improve the modeled process. In the case of Aguilera´s model, the observed dynamics were strongly affected by the synthesis rate of the mRNA and, therefore, a target for experimental intervention. Subsequently, in the case of the *repressilator* model, the dynamics are affected the most by the Hill coefficient, however, its intervention could not be feasible experimentally. The second application of sensitivity analysis is used when models require calibration. Relative high sensitive parameters are preferred and parameters with low sensitivity indices should be prioritized for fixing with experimentally determined values. For the Thomas´ model, the dissociation rate of the ligand-receptor complex seems to be a candidate for experimental measurement, while for the core GRN model, the promoter affinities seem good candidates for experimental determination and then, to reduce the model dependence on parameter calibration.

7. CONCLUSIONS AND FUTURE PERSPECTIVES

In the first place, this thesis demonstrates the feasibility of the automatic reconstruction of RBMs to recover a computational representation of cellular processes. The main technique developed, *Atlas*, was inspired by tools available for the automated metabolic modeling of constraint-based models at the genome-scale. Of note, the tool additionally models three aspects of cell behavior. Firstly, it models gene expression covering transcription, translation, and the degradation of macromolecules as these processes regulate the availability of enzymes and therefore the activity of the metabolic network. Secondly, it models the regulation of gene expression governed by the differential activity of transcription factors and other cell components such as RNA polymerases. Finally but not least, it models explicitly the bacterial genome architecture including promoters, ribosome binding sites, coding DNA sequences, terminators, and importantly, transcription factors binding sites. In contrast, other projects that are aimed to reconstruct models from information (e.g., INDRA, Gyori et al., 2017, and KAMI, Harmer et al., 2019) are circumscribed to protein-protein interactions and post-translational modifications.

We utilized real data from the bacterium *Escherichia coli* and we sought the developed method to be employed for other bacteria, such as the industrial relevant *Bacillus subtilis* or *Pseudomonas* and Cyanobacteria strains. A notable characteristic of the work is the ability to develop models for different cellular processes under a unique modeling framework. We expect all remaining cellular processes should be compatible in a WCM developed in a single rule language. Of note, this single-language model is a difference from the Karr *et al.* work (Karr et al., 2012) which combined different models (ODE, stochastic, GSMM, etc.) to simulate the bacterium *Mycoplasma genitalium*. As for disadvantages, the method relies on the user´s skills and familiarity with the python programming language. Also, we do not provide a graphical user interface, but Jupyter (Kluyver et al., 2016) and pyViPR (Ortega & Lopez, 2020) could be employed for interactive programming and visualization of the model, respectively. As a minor note, we developed integration with the BioCyc databases and potentially able to model 11090 organisms.

More in detail, we developed a divide-and-conquer strategy to develop RBMs. Those models offer advantages over natural counterparts such as ODE-based models. The developed models required stochastic simulation due to excessive combinatorial explosion, for which simulation tools are fast enough to not perceive the burden raised from the necessity to simulate hundreds or even thousands of trajectories. For the developed models (Chapter III), we simulated them 100 times employing a single core (i.e., no parallelized) and found total time to simulate less than ten minutes (see Table S 4-6). However, the calibration, uncertainty analysis, and sensitivity analysis of models required tailored methods able to harness high-performance computing infrastructures and simulate tens of thousands of models. We developed open-source tools to perform such tasks compatible with clusters of computing servers, encouraged also because of limitations of the available methods. For instance, BNF calibrates only BNGL models and R4kappa suffers from the combinatorial explosion and constrains the sensitivity analysis to one component at a time.

To calibrate models, we developed *Pleione*. The software encodes a genetic algorithm and it supports two rule languages and four stochastic simulation software. Moreover, it utilizes equivalence statistical tests besides the ability to calculate classical fitness functions. We demonstrated the utility of using those tests to calibrate a variety of models and the results relied on the type of statistical comparison made. Then, to perform uncertainty analysis, we developed *Alcyone*. The software could perform a one-leave-out Jackknife or bootstrapping to estimate uncertainty in parameter values. *Alcyone* utilizes *Pleione* to calibrate RBMs, and the required time to perform the analysis depends on the type and number of experimental replications (n). For Jackknife, *Alcyone* requires to calibrate $n + 1$ times, while bootstrapping requires a confidence level (α): for a confidence level of 90% are required 20 calibrations; for 95% are required 40; for 99% are required 200 calibrations ($\max \alpha =$ (calibrations-2)/calibrations). Of note, pyBNF employs a Bayesian approach to determine uncertainty (Mitra et al., 2019). Finally, for global sensitivity analysis, we developed *Sterope*. The software employs KaSim (Boutillier, Maasha, et al., 2018) to retrieve the number of rule executions and the influences of rules over other rules called Dynamic Influence Network (a heuristic approach that calculates activity and control coefficients from simulations) and from that results, estimates first-order, second-order, and

total sensitivity indexes. The influences are a constant number equal to the square number of rules, and therefore, the method does not suffer from the combinatorial explosion. Additionally, the method determines all sensitivity indexes at once. The mentioned software could harness a high-performance computing infrastructure to reduce the needed time to analyze models. The reduction is proportional to the total number of available CPUs, but other constraints could affect the performance such as is the network and file read/write speeds and the exact parallelization approach.

Besides the technical aspects of the developed software tailored to the specific analyses, it is ensured the extensibility of them to utilize other (stochastic) simulation software, rule languages, fitness functions, etc. In the case of *Pleione*, there is a functional interface for the simulation software Tellurium (deterministic and stochastic, K. Choi et al., 2018) and there is a plan to develop a tool to perform an Approximate Bayesian Computation to calibrate and perform uncertainty analysis at the same time. Also, there are plans to systematize the analysis of genomic modifications (gene copy numbers and genome rearrangements) and extend the method incorporated in *Atlas* to model simple bacterial communities for the understanding, simulation, design, and use of models to propose biotechnological applications.

In conclusion, this thesis proposes software, distributed freely in the python meta-package called *Pleiades,* to tackle specifically the problem of modeling, calibration, and complementary analyses of RBMs for the development of WCM. We sought that cellular processes not covered explicitly in this thesis, such as allosteric metabolic control, cooperative behavior and competition in metabolic reactions, and protein or RNA modifications could be added in a similar strategy developed for metabolism, physical interactions, transcription, translation, and genome architecture.

8. REFERENCES

Adiciptaningrum et al. (2015). Stochasticity and homeostasis in the *E. coli* replication and division cycle. *Scientific Reports*, *5*(November), 18261.

Adjaye-Gbewonyo et al. (2018). High social trust associated with increased depressive symptoms in a longitudinal South African sample. *Social Science and Medicine*, *197*, 127–135.

Agostino et al. (2011). Unwinding the molecular basis of interval and circadian timing. *Frontiers in Integrative Neuroscience*, *5*.

Agren et al. (2013). The RAVEN Toolbox and Its Use for Generating a Genome-scale Metabolic Model for Penicillium chrysogenum. *PLoS Computational Biology*, *9*(3), e1002980.

Aguilera et al. (2017). A new efficient approach to fit stochastic models on the basis of high-throughput experimental data using a model of IRF7 gene expression as case study. *BMC Systems Biology*, *11*(1), 26.

Andersen et al. (2003). Applied Latent Class Analysis. *Canadian Journal of Sociology / Cahiers Canadiens de Sociologie*, *28*(4), 584.

Apostol. (1967). *Calculus*. J. Wiley.

Arkin et al. (2018). KBase: The United States department of energy systems biology knowledgebase. *Nature Biotechnology*, *36*(7), 566–569.

Asimov. (1951). *Foundation*. Doubleday.

Axelrod & Hamilton. (1981). The evolution of cooperation. *Science*, *211*(4489), 1390–1396.

Aylward et al. (2014). Ebola virus disease in West Africa - The first 9 months of the epidemic and forward projections. *New England Journal of Medicine*, *371*(16), 1481–1495.

Barrio et al. (2006). Oscillatory regulation of hes1: Discrete stochastic delay modelling and simulation. *PLoS Computational Biology*, *2*(9), 1017–1030.

Bartocci & Lió. (2016). Computational Modeling, Formal Analysis, and Tools for Systems Biology. *PLoS Computational Biology*, *12*(1), 1–22.

Baumstark et al. (2015). The propagation of perturbations in rewired bacterial gene networks. *Nature Communications*, *6*, 1–5.

Bennett et al. (2009). Absolute metabolite concentrations and implied enzyme active site occupancy in *Escherichia coli*. *Nature Chemical Biology*, *5*(8), 593–599.

Benuwa et al. (2016). A comprehensive review of Particle swarm optimization. *International Journal of Engineering Research in Africa*, *23*(5), 141–161.

Bewick et al. (2009). Complex mathematical models of biology at the nanoscale. *Wiley Interdisciplinary Reviews: Nanomedicine and Nanobiotechnology*, *1*(6), 650–659.

Bian et al. (2017). Gut microbiome response to sucralose and its potential role in inducing liver inflammation in mice. *Frontiers in Physiology*, *8*(JUL).

Birch et al. (2014). Incorporation of flexible objectives and time-linked simulation with flux balance analysis. *Journal of Theoretical Biology*, *345*, 12–21.

Blakes et al. (2011). The infobiotics workbench: An integrated in silico modelling platform for systems and synthetic biology. *Bioinformatics*, *27*(23), 3323–3324.

Blinov et al. (2004). BioNetGen: Software for rule-based modeling of signal transduction based on the interactions of molecular domains. *Bioinformatics*, *20*(17), 3289–3291.

Blinov et al. (2006). A network model of early events in epidermal growth factor receptor signaling that accounts for combinatorial complexity. *BioSystems*, *83*(2-3 SPEC. ISS.), 136–151.

Blinov et al. (2017). Compartmental and Spatial Rule-Based Modeling with Virtual Cell. *Biophysical Journal*, *113*(7), 1365–1372.

Borgonovo. (2007). A new uncertainty importance measure. *Reliability Engineering and System Safety*, *92*(6), 771–784.

Borkowski et al. (2018). Cell-free prediction of protein expression costs for growing cells. *Nature Communications*, *9*(1457), 1–11.

Boulangé et al. (2016). Impact of the gut microbiota on inflammation, obesity, and metabolic disease. *Genome Medicine*, *8*(1).

Boutillier, Feret et al. (2018). *Kappa tools reference manual*.

Boutillier, Maasha et al. (2018). The Kappa platform for rule-based modeling. *Bioinformatics*, *34*(13), i583–i592.

Breitling. (2010). What is systems biology? *Frontiers in Physiology*, *1 MAY*(May), 1–5.

Bustos et al. (2018). Rule-based models and applications in biology. In *Methods in Molecular Biology* (Vol. 1819, pp. 3–32).

Callahan et al. (2017). Exact sequence variants should replace operational taxonomic units in marker-gene data analysis. *ISME Journal*.

Callahan et al. (2016). DADA2: High-resolution sample inference from Illumina amplicon data. *Nature Methods*, *13*(7), 581–583.

Campolongo et al. (2007). An effective screening design for sensitivity analysis of large models. *Environmental Modelling and Software*, *22*(10), 1509–1518.

Camporesi et al. (2017). KaDE: A tool to compile kappa rules into (reduced) ODE models. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *10545 LNBI*, 291–299.

Cao et al. (2005). The slow-scale stochastic simulation algorithm. *Journal of Chemical Physics*, *122*(1).

Cao et al. (2006). Efficient step size selection for the tau-leaping simulation method. *Journal of Chemical Physics*, *124*(4).

Capraro. (2013). A Model of Human Cooperation in Social Dilemmas. *PLoS ONE*, *8*(8).

Carrera & Covert. (2015). Why Build Whole-Cell Models? *Trends in Cell Biology*, *25*(12), 719–722.

Carrera et al. (2014). An integrative, multi-scale, genome-wide model reveals the phenotypic landscape of *Escherichia coli*. *Molecular Systems Biology*, *10*(7), 735.

Casadesús & Low. (2013). Programmed heterogeneity: Epigenetic mechanisms in bacteria. *Journal of Biological Chemistry*, *288*(20), 13929–13935.

Caspi et al. (2016). The MetaCyc database of metabolic pathways and enzymes and the BioCyc collection of pathway/genome databases. *Nucleic Acids Research*, *44*(D1), D471–D480.

Chen, W. H. et al. (2016). Integration of multi-omics data of a genome-reduced bacterium: Prevalence of post-transcriptional regulation and its correlation with protein

abundances. *Nucleic Acids Research*, *44*(3), 1192–1202.
Chen, Y. et al. (2014). Differential Expression Analysis of Complex RNA-seq Experiments Using edgeR. In *Statistical Analysis of Next Generation Sequencing Data* (pp. 51–74).
Cho et al. (2014). Genome-scale reconstruction of the sigma factor network in *Escherichia coli*: Topology and functional states. *BMC Biology*, *12*.
Cho et al. (2009). The transcription unit architecture of the *Escherichia coli* genome. *Nature Biotechnology*, *27*(11), 1043–1049.
Choi, J. K. & Bowles. (2007). The coevolution of parochial altruism and war. *Science*, *318*(5850), 636–640.
Choi, K. et al. (2018). Tellurium: An extensible python-based modeling environment for systems and synthetic biology. *BioSystems*, *171*, 74–79.
Chylek et al. (2015). Modeling for (physical) biologists: An introduction to the rule-based approach. *Physical Biology*, *12*(4).
Chylek et al. (2014). Rule-based modeling: A computational approach for studying biomolecular site dynamics in cell signaling systems. *Wiley Interdiscip. Rev. Syst. Biol. Med.*, *6*(1), 13–36.
Coleman. (1988). Social capital in the creation of human capital. *American Journal of Sociology*, *94*, 95–120.
Comeau et al. (2017). Microbiome Helper: a Custom and Streamlined Workflow for Microbiome Research. *MSystems*, *2*(1).
Cornell. (1990). The evaluation of bioequivalence using nonparametric procedures. *Communications in Statistics - Theory and Methods*, *19*(11), 4153–4165.
Costa et al. (2016). Kinetic modeling of cell metabolism for microbial production. *Journal of Biotechnology*, *219*, 126–141.
Covert et al. (2008). Integrating metabolic, transcriptional regulatory and signal transduction models in *Escherichia coli*. *Bioinformatics*, *24*(18), 2044–2050.
Cukier et al. (1973). Study of the sensitivity of coupled reaction systems to uncertainties in rate coefficients. I Theory. *J. Chem. Phys.*, *59*(8), 3873–3878.
Daigle et al. (2012). Accelerated maximum likelihood parameter estimation for stochastic biochemical systems. *BMC Bioinformatics*, *13*(1).
Daly et al. (2018). Inference-based assessment of parameter identifiability in nonlinear biological models. *Journal of the Royal Society Interface*, *15*(144).
Danos et al. (2008). Abstract interpretation of cellular signalling networks. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *4905 LNCS*, 83–97.
Danos, Feret, Fontana, & Krivine. (2007). Scalable simulation of cellular signaling networks. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *4807 LNCS*, 139–157.
Danos, Feret, Fontana, Harmer et al. (2007). Rule-based modelling of cellular signalling. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *4703 LNCS*, 17–41.
Davis et al. (2016). Modular Assembly of the Bacterial Large Ribosomal Subunit. *Cell*, *167*(6), 1610-1622.e15.

de Laguna. (1962). The Role of Teleonomy in Evolution. *Philosophy of Science*, *29*(2), 117–131.

Dematté & Prandi. (2010). GPU computing for systems biology. *Briefings in Bioinformatics*, *11*(3), 323–333.

Dias et al. (2015). Reconstructing genome-scale metabolic models with merlin. *Nucleic Acids Research*, *43*(8), 3899–3910.

Dittamo & Cangelosi. (2009). Optimized parallel implementation of Gillespie's first reaction method on graphics processing units. *Proceedings - 2009 International Conference on Computer Modeling and Simulation, ICCMS 2009*.

Edwards et al. (2001). In silico predictions of *Escherichia coli* metabolic capabilities are consistent with experimental data. *Nature Biotechnology*, *19*(2), 125–130.

El-Seidy et al. (2016). Two population three-player prisoner's dilemma game. *Applied Mathematics and Computation*, *277*, 44–53.

Endy & Brent. (2001). Modelling cellular behaviour. *Nature*, *409*(6818), 391–395.

Eng & Borenstein. (2016). An algorithm for designing minimal microbial communities with desired metabolic capacities. *Bioinformatics*, *32*(13), 2008–2016.

Engblom. (2008). Time-parallel simulation of stochastic chemical kinetics. *AIP Conference Proceedings*, *1048*, 174–177.

Evans. (2009). *Partial differential equations*. Orient BlackSwan.

Everard et al. (2013). Cross-talk between Akkermansia muciniphila and intestinal epithelium controls diet-induced obesity. *Proceedings of the National Academy of Sciences of the United States of America*, *110*(22), 9066–9071.

Evers. (2001). Social capital: Measurement and consequences. *Canadian Journal of Policy Research*, *2*(1), 41–51.

Faeder, J R et al. (2005). Rule-based modeling of biochemical networks. *Complexity*, *10*(4), 22–41.

Faeder, James R. et al. (2009). Rule-based modeling of biochemical systems with BioNetGen. *Methods in Molecular Biology*, *500*(1), 113–167.

Faeder, James R et al. (2003). Investigation of Early Events in FcεRI-Mediated Signaling Using a Detailed Mathematical Model. *The Journal of Immunology*, *170*(7), 3769–3781.

Faria et al. (2018). Methods for automated genome-scale metabolic model reconstruction. *Biochemical Society Transactions*, *46*(4), 931–936.

Feret. (2007). Reachability analysis of biological signalling pathways by abstract interpretation. *AIP Conference Proceedings*, *963*(2), 619–622.

Feret et al. (2009). Internal coarse-graining of molecular systems. *Proceedings of the National Academy of Sciences*, *106*(16), 6453–6458.

Fischer & Sauer. (2003). Metabolic flux profiling of *Escherichia coli* mutants in central carbon metabolism using GC-MS. *European Journal of Biochemistry*, *270*(5), 880–891.

Fisher & Henzinger. (2007). Executable cell biology. *Nature Biotechnology*, *25*(11), 1239–1249.

Forbes et al. (2018). Dynamic Influence Networks for Rule-Based Models. *IEEE Transactions on Visualization and Computer Graphics*, *24*(1), 184–194.

Foster et al. (2019). From *Escherichia coli* mutant 13C labeling data to a core kinetic model: A kinetic model parameterization pipeline. *PLoS Computational Biology*, *15*(9), e1007319.

Fowler et al. (1985). The amino acid sequence of thiogalactoside transacetylase of *Escherichia coli*. *Biochimie*, *67*(1), 101–108.

Fredens et al. (2019). Total synthesis of *Escherichia coli* with a recoded genome. *Nature*, *569*(7757), 514–518.

Fuchs & Archer. (1997). Realist Social Theory: The Morphogenetic Approach. *Canadian Journal of Sociology / Cahiers Canadiens de Sociologie*, *22*(3), 385.

Fuenzalida, Bustos et al. (2017). PISKaS: a HPC tool for stochastic agent/rule-based modeling of spatially explicit biological systems [slides, version 1; not peer reviewed]. *F1000Research (ISCB Comm J)*, *6*(922).

Fuenzalida, Martin et al. (2017). Modeling multiscale complex biological systems using PISKa [poster, version 1; not peer reviewed]. *F1000Research (ISCB Comm J)*, *6*(914).

Fuenzalida et al. (2015). PISKa: A Parallel Implementation of Spatial Kappa [version 1; not peer reviewed]. *International Society for Computational Biology Community Journal*, *4*.

Fuhrer et al. (2017). Genomewide landscape of gene–metabolome associations in *Escherichia coli*. *Molecular Systems Biology*, *13*(1), 907.

Fukuyama. (1995). *Trust: The Social Virtues and the Creation of Prosperity*. Free Press.

Galgani et al. (2013). Enhanced skeletal muscle lipid oxidative efficiency in insulin-resistant vs insulin-sensitive nondiabetic, nonobese humans. *Journal of Clinical Endocrinology and Metabolism*, *98*(4).

Gibson & Bruck. (2000). Efficient exact stochastic simulation of chemical systems with many species and many channels. *Journal of Physical Chemistry A*, *104*(9), 1876–1889.

Gilbert. (2008). *Agent-based models*. SAGE.

Gillespie, D. T. (2001). Approximate accelerated stochastic simulation of chemically reacting systems. *Journal of Chemical Physics*, *115*(4), 1716–1733.

Gillespie, Daniel T. (1976). A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *Journal of Computational Physics*, *22*(4), 403–434.

Gillespie, Daniel T. (1992). A rigorous derivation of the chemical master equation. *Physica A: Statistical Mechanics and Its Applications*, *188*(1–3), 404–425.

Gillespie, Daniel T. (2007). Stochastic Simulation of Chemical Kinetics. *Annual Review of Physical Chemistry*, *58*(1), 35–55.

Gillespie, Daniel T. (1977). Exact stochastic simulation of coupled chemical reactions. *Journal of Physical Chemistry*, *81*(25), 2340–2361.

Goldsmith. (2000). Acute and subchronic toxicity of sucralose. *Food and Chemical Toxicology*, *38*(SUPPL.2), 53–69.

Gottstein et al. (2016). Constraint-based stoichiometric modelling from single organisms to microbial communities. *Journal of the Royal Society Interface*, *13*(124).

Gowrishankar. (2015). End of the Beginning: Elongation and Termination Features of

Alternative Modes of Chromosomal Replication Initiation in Bacteria. *PLoS Genetics*, *11*(1).

Grice & Goldsmith. (2000). Sucralose - An overview of the toxicity data. *Food and Chemical Toxicology*, *38*(SUPPL.2).

Grimbs et al. (2019). A system-wide network reconstruction of gene regulation and metabolism in *Escherichia coli*. *PLoS Computational Biology*, *15*(5), 1–29.

Grotz et al. (2003). Lack of effect of sucralose on glucose homeostasis in subjects with type 2 diabetes. *Journal of the American Dietetic Association*, *103*(12).

Grotz et al. (2017). A 12-week randomized clinical trial investigating the potential for sucralose to affect glucose homeostasis. *Regulatory Toxicology and Pharmacology*, *88*, 22–33.

Guiso et al. (2004). The role of social capital in financial development. *American Economic Review*, *94*(3), 526–556.

Gupta & Culver. (2014). Multiple in vivo pathways for *Escherichia coli* small ribosomal subunit assembly occur on one pre-rRNA. *Nature Structural and Molecular Biology*, *21*(10), 937–943.

Gutierrez et al. (2012). Cooperative Binding of Transcription Factors Promotes Bimodal Gene Expression Response. *PLoS ONE*, *7*(9).

Gyori et al. (2017). From word models to executable models of signaling networks using automated assembly. *Molecular Systems Biology*, *13*(11), 954.

Hädicke & Klamt. (2017). EColiCore2: A reference network model of the central metabolism of *Escherichia coli* and relationships to its genome-scale parent model. *Scientific Reports*, *7*, 39647.

Hahl & Kremling. (2016). A comparison of deterministic and stochastic modeling approaches for biochemical reaction systems: On fixed points, means, and modes. *Frontiers in Genetics*, *7*(AUG), 157.

Harmer et al. (2019). Bio-curation for cellular signalling: The KAMI project. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *16*(5), 1562–1573.

Henrich et al. (2010). The weirdest people in the world? *Behavioral and Brain Sciences*, *33*(2–3), 61–83.

Henry et al. (2010). High-throughput generation, optimization and analysis of genome-scale metabolic models. *Nature Biotechnology*, *28*(9), 977–982.

Herman & Usher. (2017). SALib: An open-source Python library for Sensitivity Analysis. *J. Open Source Softw.*, *2*(9), 97.

Hernández-Prieto et al. (2014). Toward a systems-level understanding of gene regulatory, protein interaction, and metabolic networks in cyanobacteria. *Frontiers in Genetics*, *5*(JUL), 1–18.

Hershberg et al. (2003). A survey of small RNA-encoding genes in *Escherichia coli*. *Nucleic Acids Research*, *31*(7), 1813–1820.

Hethcote. (2000). Mathematics of infectious diseases. *SIAM Review*, *42*(4), 599–653.

Hindmarsh et al. (2005). SUNDIALS: Suite of nonlinear and differential/algebraic equation solvers. *ACM Transactions on Mathematical Software*, *31*(3), 363–396.

Hlavacek et al. (2006). Rules for modeling signal-transduction systems. *Science's STKE : Signal Transduction Knowledge Environment*, *2006*(344), re6.

Hogg et al. (2014). Exact Hybrid Particle/Population Simulation of Rule-Based Models of Biochemical Systems. *PLoS Computational Biology*, *10*(4).

Holyoak & Lawler. (1996). Persistence of an extinction-prone predator-prey interaction through metapopulation dynamics. *Ecology*, *77*(6), 1867–1879.

Huber et al. (1981). The anomeric specificity of β-galactosidase and *lac* permease from *Escherichia coli*. *Canadian Journal of Biochemistry*, *59*(2), 100–105.

Ishihama. (1981). Subunit of assembly of *Escherichia coli* RNA polymerase. *Adv Biophys*, *14*, 1–35.

Jahan et al. (2016). Development of an accurate kinetic model for the central carbon metabolism of *Escherichia coli*. *Microbial Cell Factories*, *15*(1), 112.

Janssen & Ostrom. (2006). Empirically Based, Agent-based models. *Ecology and Society*, *11*(2).

Johns et al. (2018). Metagenomic mining of regulatory elements enables programmable species-selective gene expression. *Nature Methods*, *15*(5), 323–329.

Jones et al. (2015). *SciPy: Open Source Scientific Tools for Python*, *2001 (http://www.scipy.org/)*.

Jost. (2011). *Partial differential equations*. Springer.

Jozefczuk et al. (2010). Metabolomic and transcriptomic stress response of *Escherichia coli*. *Molecular Systems Biology*, *6*(1), 1–16.

Juers et al. (2012). LacZ β-galactosidase: Structure and function of an enzyme of historical and molecular biological importance. *Protein Science*, *21*(12), 1792–1807.

Kærn et al. (2005). Stochasticity in gene expression: from theories to phenotypes. *Nature Reviews Genetics*, *6*(6), 451–464.

Kamath-Loeb & Gross. (1991). Translational regulation of $σ$32 synthesis: Requirement for an internal control element. *Journal of Bacteriology*, *173*(12), 3904–3906.

Karp, Billington et al. (2018). The BioCyc collection of microbial genomes and metabolic pathways. *Briefings in Bioinformatics*, *20*(4), 1085–1093.

Karp et al. (2019). Pathway Tools version 23.0 update: software for pathway/genome informatics and systems biology. *Briefings in Bioinformatics*, bbz104.

Karp, Ong et al. (2018). The EcoCyc Database. *EcoSal Plus*, *8*(1), 1–19.

Karr et al. (2012). A whole-cell computational model predicts phenotype from genotype. *Cell*, *150*(2), 389–401.

Karr, Takahashi et al. (2015). The principles of whole-cell modeling. *Current Opinion in Microbiology*, *27*, 18–24.

Karr, Williams et al. (2015). Summary of the DREAM8 Parameter Estimation Challenge: Toward Parameter Identification for Whole-Cell Models. *PLoS Comput. Biol.*, *11*(5), 1–21.

Kasai et al. (2015). Comparison of the gut microbiota composition between obese and non-obese individuals in a Japanese population, as analyzed by terminal restriction fragment length polymorphism and next-generation sequencing. *BMC Gastroenterology*, *15*(1).

Kauffman et al. (2003). Advances in flux balance analysis. *Current Opinion in Biotechnology*, *14*(5), 491–496.

Kent et al. (2013). What can we learn from global sensitivity analysis of biochemical

systems? *PLoS ONE*, *8*(11).
Kermack & McKendrick. (1927). A contribution to the mathematical theory of epidemics. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, *115*(772), 700–721.
Keseler et al. (2017). The EcoCyc database: Reflecting new knowledge about *Escherichia coli* K-12. *Nucleic Acids Research*, *45*(D1), D543–D550.
Khodayari et al. (2015). Succinate Overproduction: A Case Study of Computational Strain Design Using a Comprehensive *Escherichia coli* Kinetic Model. *Frontiers in Bioengineering and Biotechnology*, *2*(January), 76.
Kim, B. et al. (2015). Applications of genome-scale metabolic network model in metabolic engineering. *J Ind Microbiol Biotechnol*, *42*, 339–348.
Kim, M. et al. (2016). Multi-omics integration accurately predicts cellular state in unexplored conditions for *Escherichia coli*. *Nature Communications*, *7*, 13090.
Kim, M. & Tagkopoulos. (2018). Data integration and predictive modeling methods for multi-omics datasets. *Molecular Omics*, *14*(1), 8–25.
Kim, M. et al. (2015). Microbial Forensics: Predicting Phenotypic Characteristics and Environmental Conditions from Large-Scale Gene Expression Profiles. *PLoS Computational Biology*, *11*(3), 1–21.
King et al. (2015). Next-generation genome-scale models for metabolic engineering. *Current Opinion in Biotechnology*, *35*, 23–29.
Kitano. (2002). Systems biology: A brief overview. *Science*, *295*(5560), 1662–1664.
Klamt et al. (2009). Hypergraphs and cellular networks. *PLoS Computational Biology*, *5*(5), e1000385.
Klipp & Liebermeister. (2006). Mathematical modeling of intracellular signaling pathways. *BMC Neuroscience*, *7*(SUPPL. 1), 1–16.
Kluyver et al. (2016). Jupyter Notebooks—a publishing format for reproducible computational workflows. *Positioning and Power in Academic Publishing: Players, Agents and Agendas - Proceedings of the 20th International Conference on Electronic Publishing, ELPUB 2016*.
Knack & Keefer. (1997). Does social capital have an economic payoff? A cross-country investigation. *Quarterly Journal of Economics*, *112*(4), 1251–1288.
Konak et al. (2006). Multi-objective optimization using genetic algorithms: A tutorial. *Reliability Engineering and System Safety*, *91*(9), 992–1007.
Koschorreck et al. (2007). Reduced modeling of signal transduction - A modular approach. *BMC Bioinformatics*, *8*, 1–24.
Koschorreck & Gilles. (2008). ALC: Automated reduction of rule-based models. *BMC Systems Biology*, *2*, 1–24.
Kot. (2001). *Elements of Mathematical Ecology*. Cambridge University Press.
Kozer et al. (2013). Exploring higher-order EGFR oligomerisation and phosphorylation - A combined experimental and theoretical approach. *Molecular BioSystems*, *9*(7), 1849–1863.
Krzyzanski & Jusko. (1998). Integrated functions for four basic models of indirect pharmacodynamic response. *Journal of Pharmaceutical Sciences*, *87*(1), 67–72.
Kuhn. (2019). Prisoner's Dilemma. In Zalta (Ed.), *The Stanford Encyclopedia of*

Philosophy (Winter 201). Metaphysics Research Lab, Stanford University.
Labhsetwar et al. (2013). Heterogeneity in protein expression induces metabolic variability in a modeled *Escherichia coli* population. *Proc. Natl. Acad. Sci.*, *110*(34), 14006–14011.
Lauzon & Caffo. (2009). Easy multiplicity control in equivalence testing using two one-sided tests. *American Statistician*, *63*(2), 147–154.
Lerman et al. (2012). In silico method for modelling metabolism and gene product expression at genome scale. *Nature Communications*, *3*(May), 910–929.
Lewendon et al. (1995). Structural and mechanistic studies of galactoside acetyltransferase, the *Escherichia coli* LacA gene product. *Journal of Biological Chemistry*, *270*(44), 26326–26331.
Lewis, M. (2005). The lac repressor. *Comptes Rendus - Biologies*, *328*(6 SPEC. ISS.), 521–548.
Lewis, M. (2011). A tale of two repressors. In *Journal of Molecular Biology* (Vol. 409, Issue 1, pp. 14–27). Elsevier Ltd.
Lewis, N. E. et al. (2012). Constraining the metabolic genotype-phenotype relationship using a phylogeny of in silico methods. In *Nature Reviews Microbiology* (Vol. 10, Issue 4, pp. 291–305).
Li, S. et al. (2013). Directional RNA-seq reveals highly complex condition-dependent transcriptomes in *E. coli* K12 through accurate full-length transcripts assembling. *BMC Genomics*, *14*(1), 520.
Li, X. et al. (2018). Bottom-up single-molecule strategy for understanding subunit function of tetrameric β-galactosidase. *Proceedings of the National Academy of Sciences of the United States of America*, *115*(33), 8346–8351.
Lim et al. (2013). Design Principles of Regulatory Networks: Searching for the Molecular Algorithms of the Cell. *Molecular Cell*, *49*(2), 202–212.
Liu, B. & Faeder. (2017). Parameter estimation of rule-based models using statistical model checking. *Proceedings - 2016 IEEE International Conference on Bioinformatics and Biomedicine, BIBM 2016*, 1453–1459.
Liu, J. H. et al. (2014). Symbologies, technologies, and identities: Critical junctures theory and the multi-layered nation-state. *International Journal of Intercultural Relations*, *43*(PA), 2–12.
Liu, J. H. et al. (2018). The Global Trust Inventory as a "Proxy Measure" for Social Capital: Measurement and Impact in 11 Democratic Societies. *Journal of Cross-Cultural Psychology*, *49*(5), 789–810.
Liu, J. K. et al. (2014). Reconstruction and modeling protein translocation and compartmentalization in *Escherichia coli* at the genome-scale. *BMC Systems Biology*, *8*, 110.
Lobach et al. (2019). Assessing the in vivo data on low/no-calorie sweeteners and the gut microbiota. *Food and Chemical Toxicology*, *124*, 385–399.
Lok & Brent. (2005). Automatic generation of cellular reaction networks with Moleculizer 1.0. *Nature Biotechnology*, *23*(1), 131–136.
Long & Antoniewicz. (2019). Metabolic flux responses to deletion of 20 core enzymes reveal flexibility and limits of *E. coli* metabolism. *Metabolic Engineering*, *55*, 249–

257.
Lopez et al. (2013). Programming biological models in Python using PySB. *Molecular Systems Biology*, *9*(1), 646.
Lozupone, C. A. et al. (2007). Quantitative and qualitative β diversity measures lead to different insights into factors that structure microbial communities. *Applied and Environmental Microbiology*, *73*(5), 1576–1585.
Lozupone, C. et al. (2011). UniFrac: An effective distance metric for microbial community comparison. *ISME Journal*, *5*(2), 169–172.
Machado et al. (2016). Stoichiometric Representation of Gene–Protein–Reaction Associations Leverages Constraint-Based Analysis from Reaction to Gene-Level Phenotype Prediction. *PLoS Computational Biology*, *12*(10), 1–24.
MacNeil & Walhout. (2011). Gene regulatory networks and the role of robustness and stochasticity in the control of gene expression. *Genome Research*, *21*(5), 645–657.
Magnúsdóttir et al. (2016). Generation of genome-scale metabolic reconstructions for 773 members of the human gut microbiota. *Nature Biotechnology*, *35*(1), 81–89.
Magnuson et al. (2016). Biological fate of low-calorie sweeteners. *Nutrition Reviews*, *74*(11), 670–689.
Magnuson et al. (2017). Critical review of the current literature on the safety of sucralose. *Food and Chemical Toxicology*, *106*, 324–355.
Marbach et al. (2012). Wisdom of crowds for robust gene network inference. *Nature Methods*, *9*(8), 796–804.
Marshall. (1951). A Large-Sample Test of the Hypothesis that one of two Random Variables is Stochastically Larger than the Other. *Journal of the American Statistical Association*, *46*(255), 366–374.
Martin et al. (2016). Graphlet based metrics for the comparison of gene regulatory networks. *PLoS ONE*, *11*(10), 1–13.
Martyn et al. (2018). Low-/no-calorie sweeteners: A review of global intakes. *Nutrients*, *10*(3).
Matsuda & DeFronzo. (1999). Insulin sensitivity indices obtained from oral glucose tolerance testing: Comparison with the euglycemic insulin clamp. *Diabetes Care*, *22*(9), 1462–1470.
Matsuura et al. (2011). Kinetic analysis of β-galactosidase and β-glucuronidase tetramerization coupled with protein translation. *Journal of Biological Chemistry*, *286*(25), 22028–22034.
Matthews et al. (1985). Homeostasis model assessment: insulin resistance and β-cell function from fasting plasma glucose and insulin concentrations in man. *Diabetologia*, *28*(7), 412–419.
Mauri & Klumpp. (2014). A Model for Sigma Factor Competition in Bacterial Cells. *PLoS Computational Biology*, *10*(10), e1003845.
Mayer & Hansen. (2017). Evolvability and robustness: A paradox restored. *Journal of Theoretical Biology*, *430*, 78–85.
Mccloskey et al. (2014). A model-driven quantitative metabolomics analysis of aerobic and anaerobic metabolism in *E. coli* K-12 MG1655 that is biochemically and thermodynamically consistent. *Biotechnology and Bioengineering*, *111*(4), 803–815.

McCollum et al. (2006). The sorting direct method for stochastic simulation of biochemical systems with varying reaction execution behavior. *Computational Biology and Chemistry*, *30*(1), 39–49.

Medina et al. (2017). Distinct patterns in the gut microbiota after surgical or medical therapy in obese patients. *PeerJ*, *2017*(6).

Millard et al. (2017). Metabolic regulation is sufficient for global and robust coordination of glucose uptake, catabolism, energy production and growth in *Escherichia coli*. *PLoS Computational Biology*, *13*(2), e1005396.

Mishler & Rose. (2001). What are the origins of political trust? Testing institutional and cultural theories in post-communist societies. *Comparative Political Studies*, *34*(1), 30–62.

Mitchell et al. (2007). The *Escherichia coli* regulator of sigma 70 protein, Rsd, can up-regulate some stress-dependent promoters by sequestering sigma 70. *Journal of Bacteriology*, *189*(9), 3489–3495.

Mitra et al. (2019). PyBioNetFit and the Biological Property Specification Language. *IScience*, *19*, 1012–1036.

Mori. (2004). From the sequence to cell modeling: Comprehensive functional genomics in *Escherichia coli*. *Journal of Biochemistry and Molecular Biology*, *37*(1), 83–92.

Morris, D. M. & Jensen. (2008). Toward a biomechanical understanding of whole bacterial cells. *Annual Review of Biochemistry*, *77*, 583–613.

Morris, M. D. (1991). Factorial Sampling Plans for Preliminary Computational Experiments. *Technometrics*, *33*(2), 161–174.

Most et al. (2017). Gut microbiota composition strongly correlates to peripheral insulin sensitivity in obese men but not in women. *Beneficial Microbes*, *8*(4), 557–562.

Murphy et al. (2010). Rule Based Modeling and Model Refinement. In *Elements of Computational Systems Biology* (pp. 83–114). John Wiley & Sons, Inc.

Newman. (2002). Spread of epidemic disease on networks. *Physical Review E - Statistical Physics, Plasmas, Fluids, and Related Interdisciplinary Topics*, *66*(1).

Nishimura et al. (2015). Stochasticity in Gene Expression in a Cell-Sized Compartment. *ACS Synthetic Biology*, *4*(5), 566–576.

Notebaart et al. (2014). Network-level architecture and the evolutionary potential of underground metabolism. *Proc. Natl. Acad. Sci.*, *111*(32), 11762–11767.

Nowak & Sigmund. (1990). The evolution of stochastic strategies in the Prisoner's Dilemma. *Acta Applicandae Mathematicae*, *20*(3), 247–265.

Nuñez et al. (2012). A Rule-based Model of a Hypothetical Zombie Outbreak: Insights on the role of emotional factors during behavioral adaptation of an artificial population. *Arxiv*.

O'Brien et al. (2013). Genome-scale models of metabolism and gene expression extend and refine growth phenotype prediction. *Molecular Systems Biology*, *9*(693).

Ollivier et al. (2010). Scalable rule-based modelling of allosteric proteins and biochemical networks. *PLoS Computational Biology*, *6*(11).

Omran, Ahearn et al. (2013). Metabolic effects of sucralose on environmental bacteria. *Journal of Toxicology*, *2013*.

Omran, Baker et al. (2013). Differential Bacteriostatic Effects of Sucralose on Various

Species of Environmental Bacteria. *ISRN Toxicology*, *2013*, 1–6.
Ortega & Lopez. (2020). Interactive Multiresolution Visualization of Cellular Network Processes. *IScience*, *23*(1), 100748.
Pardoux. (2010). *Markov Processes and Applications: Algorithms, Networks, Genome and Finance*. Wiley.
Patrick et al. (2015). Free RNA polymerase in *Escherichia coli*. *Biochimie*, *119*, 80–91.
Perez-Acle et al. (2018). Stochastic simulation of multiscale complex systems with PISKaS: A rule-based approach. *Biochemical and Biophysical Research Communications*, *498*(2), 342–351.
Perez-Riverol et al. (2016). Ten Simple Rules for Taking Advantage of Git and GitHub. *PLoS Computational Biology*, *12*(7), e1004947.
Pey et al. (2013). Integrating gene and protein expression data with genome-scale metabolic networks to infer functional pathways. *BMC Systems Biology*, *7*.
Piper et al. (2009). A global view of *Escherichia coli* Rsd protein and its interactions. *Molecular BioSystems*, *5*(12), 1943–1947.
Polisky. (1988). ColE1 replication control circuitry: Sense from antisense. *Cell*, *55*(6), 929–932.
Purcell et al. (2013). Towards a whole-cell modeling approach for synthetic biology. *Chaos*, *23*(2).
Putnam. (2000a). *Bowling Alone: The Collapse and Revival of American Community*. Simon & Shuster.
Putnam. (2000b). Social capital: One or many? Definition and measurement. *Journal of Economic Surveys*, *14*(5), 629–653.
Qu et al. (2011). Multi-scale modeling in biology: How to bridge the gaps between scales? *Progress in Biophysics and Molecular Biology*, *107*(1), 21–31.
Quack & International Union of Pure and Applied Chemistry. Physical and Biophysical Chemistry Division. (2007). *Quantities, units, and symbols in physical chemistry*. RSC Pub.
Quast et al. (2013). The SILVA ribosomal RNA gene database project: Improved data processing and web-based tools. *Nucleic Acids Research*, *41*(D1).
Racle et al. (2015). Noise analysis of genome-scale protein synthesis using a discrete computational model of translation. *The Journal of Chemical Physics*, *143*(4), 044109.
Raj & van Oudenaarden. (2008). Nature, Nurture, or Chance: Stochastic Gene Expression and Its Consequences. *Cell*, *135*(2), 216–226.
Rand & Nowak. (2013). Human cooperation. *Trends in Cognitive Sciences*, *17*(8), 413–425.
Rathinam et al. (2003). Stiffness in stochastic chemically reacting systems: The implicit tau-leaping method. *Journal of Chemical Physics*, *119*(24), 12784–12794.
Rees-Garbutt et al. (2020). Designing minimal genomes using whole-cell models. *Nature Communications*, *11*(836).
Regan & Archer. (1992). Culture and Agency: The Place of Culture in Social Theory. *Social Forces*, *70*(3), 828.
Regev et al. (2017). The human cell atlas. *ELife*, *6*, e27041.

Richerson et al. (2016). Cultural group selection plays an essential role in explaining human cooperation: A sketch of the evidence. *Behavioral and Brain Sciences*, *39*, e30.

Richerson & Henrich. (2012). Tribal Social Instincts and the Cultural Evolution of Institutions to Solve Collective Action Problems. *SSRN Electronic Journal*.

Roberts et al. (2000). Sucralose metabolism and pharmacokinetics in man. *Food and Chemical Toxicology*, *38*(SUPPL.2), 31–41.

Robinson, A. & Van Oijen. (2013). Bacterial replication, transcription and translation: Mechanistic insights from single-molecule biochemical studies. *Nature Reviews Microbiology*, *11*(5), 303–315.

Robinson, M. D. et al. (2009). edgeR: A Bioconductor package for differential expression analysis of digital gene expression data. *Bioinformatics*, *26*(1), 139–140.

Rocklin. (2015). Dask: Parallel Computation with Blocked algorithms and Task Scheduling. In Huff & Bergstra (Eds.), *Proceedings of the 14th Python in Science Conference* (pp. 130–136).

Rodriguez-Palacios et al. (2018). The Artificial Sweetener Splenda Promotes Gut Proteobacteria, Dysbiosis, and Myeloperoxidase Reactivity in Crohn's Disease-Like Ileitis. *Inflammatory Bowel Diseases*, *24*(5), 1005–1020.

Rogers & Ashlock. (2010). The Impact of Long-Term Memory in the Iterated Prisoner's Dilemma. In *Intelligent Engineering Systems through Artificial Neural Networks* (pp. 245–252).

Romano et al. (2017). Parochial trust and cooperation across 17 societies. *Proceedings of the National Academy of Sciences of the United States of America*, *114*(48), 12702–12707.

Rousseau et al. (1998). Not so different after all: A cross-discipline view of trust. *Academy of Management Review*, *23*(3), 393–404.

Rutkauskas et al. (2009). Tetramer opening in LacI-mediated DNA looping. *Proceedings of the National Academy of Sciences of the United States of America*, *106*(39), 16627–16632.

Saa & Nielsen. (2017). Formulation, construction and analysis of kinetic models of metabolism: A review of modelling frameworks. *Biotechnology Advances*, *35*(8), 981–1003.

Saltelli et al. (2008). *Global Sensitivity Analysis. The Primer*. John Wiley & Sons, Ltd.

Saltelli et al. (2005). Sensitivity analysis for chemical models. *Chemical Reviews*, *105*(7), 2811–2827.

Sanghvi et al. (2013). Accelerated discovery via a whole-cell model. *Nature Methods*, *10*(12), 1192–1195.

Santibáñez et al. (2019). Pleione: A tool for statistical and multi-objective calibration of Rule-based models. *Scientific Reports*, *9*, 15104.

Schaff et al. (2016). Rule-based modeling with Virtual Cell. *Bioinformatics*, *32*(18), 2880–2882.

Schiffman & Rother. (2013). Sucralose, a synthetic organochlorine sweetener: Overview of biological issues. *Journal of Toxicology and Environmental Health - Part B: Critical Reviews*, *16*(7), 399–451.

Schuirmann. (1987). A comparison of the Two One-Sided Tests Procedure and the Power Approach for assessing the equivalence of average bioavailability. *Journal of Pharmacokinetics and Biopharmaceutics*, *15*(6), 657–680.

Sekar et al. (2016). N-terminal-based targeted, inducible protein degradation in *Escherichia coli*. *PLoS ONE*, *11*(2), 1–17.

Shahrezaei et al. (2008). Colored extrinsic fluctuations and stochastic gene expression. *Molecular Systems Biology*, *4*.

Shajani et al. (2011). Assembly of bacterial ribosomes. *Annual Review of Biochemistry*, *80*, 501–526.

Shannon et al. (2003). Cytoscape: A software Environment for integrated models of biomolecular interaction networks. *Genome Research*, *13*(11), 2498–2504.

Shearer & Swithers. (2016). Artificial sweeteners and metabolic dysregulation: Lessons learned from agriculture and the laboratory. *Reviews in Endocrine and Metabolic Disorders*, *17*(2), 179–186.

Shinar et al. (2006). Rules for biological regulation based on error minimization. *Proceedings of the National Academy of Sciences of the United States of America*, *103*(11), 3999–4004.

Shoaie et al. (2013). Understanding the interactions between bacteria in the human gut through metabolic modeling. *Scientific Reports*, *3*, 1–10.

Shulgin et al. (1998). Pulse vaccination strategy in the SIR epidemic model. *Bulletin of Mathematical Biology*, *60*(6), 1123–1148.

Silberberg. (2015). *Chemistry: The Molecular Nature of Matter and Change- Ebook*. McGraw-Hill.

Simeonidis & Price. (2015). Genome-scale modeling for metabolic engineering. *Journal of Industrial Microbiology and Biotechnology*, *42*(3), 327–338.

Sircana et al. (2018). Altered Gut Microbiota in Type 2 Diabetes: Just a Coincidence? *Current Diabetes Reports*, *18*(10).

Smallbone et al. (2007). Something from nothing Bridging the gap between constraint-based and kinetic modelling. *FEBS Journal*, *274*(21), 5576–5585.

Smith et al. (2012). RuleBender: integrated modeling, simulation and visualization for rule-based intracellular biochemistry. *BMC Bioinformatics*, *13*(Suppl 8), S3.

Sneddon et al. (2011). Efficient modeling, simulation and coarse-graining of biological complexity with NFsim. *Nature Methods*, *8*(2), 177–183.

Sneddon et al. (2008). NFsim : Managing Complexity in Stochastic Simulations of Reaction Networks. *Physical Review*, 4–4.

Sobol. (2001). Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. *Math. Comput. Simul.*, *55*(1–3), 271–280.

Sobol & Kucherenko. (2009). Derivative based global sensitivity measures and their link with global sensitivity indices. *Mathematics and Computers in Simulation*, *79*(10), 3009–3017.

Somogyi et al. (2015). LibRoadRunner: A high performance SBML simulation and analysis library. *Bioinformatics*, *31*(20), 3315–3321.

Sorokin et al. (2019). RKappa: Software for Analyzing Rule-Based Models. In *Methods in Molecular Biology* (Vol. 1945, pp. 363–390). Springer New York.

Sorokina et al. (2013). A simulator for spatially extended kappa models. *Bioinformatics*, *29*(23), 3105–3106.

Soufi et al. (2015). Characterization of the *E. coli* proteome and its modifications during growth and ethanol stres. *Frontiers in Microbiology*, *6*(FEB), 1–11.

Sowa et al. (2017). Integrative FourD omics approach profiles the target network of the carbon storage regulatory system. *Nucleic Acids Research*, *45*(4), 1673–1686.

Srinivasan et al. (2015). Constructing kinetic models of metabolism at genome-scales: A review. *Biotechnology Journal*, *10*(9), 1345–1359.

Stewart & Wilson-Kanamori. (2011). Modular modelling in synthetic biology: Light-based communication in *E. coli*. *Electronic Notes in Theoretical Computer Science*, *277*(1), 77–87.

Strahl et al. (2015). Membrane recognition and dynamics of the RNA degradosome. *PLoS Genetics*, *11*(2), e1004961.

Striebel et al. (2009). Controlled destruction: AAA+ ATPases in protein degradation from bacteria to eukaryotes. *Current Opinion in Structural Biology*, *19*(2), 209–217.

Su et al. (2014). Biological Network Exploration with Cytoscape 3. *Current Protocols in Bioinformatics*, *2014*(47), 8.13.1-8.13.24.

Suez et al. (2014). Artificial sweeteners induce glucose intolerance by altering the gut microbiota. *Nature*, *514*(7521), 181–186.

Suez et al. (2015). Non-caloric artificial sweeteners and the microbiome: Findings and challenges. *Gut Microbes*, *6*(2), 149–155.

Sun et al. (2012). Parameter estimation using metaheuristics in systems biology: A comprehensive review. *IEEE/ACM Trans. Comput. Biol. Bioinforma.*, *9*(1), 185–202.

Sung et al. (2017). Global metabolic interaction network of the human gut microbiota for context-specific community-scale analysis. *Nature Communications*, *8*, 1–12.

Swain et al. (2002). Intrinsic and extrinsic contributions to stochasticity in gene expression. *Proceedings of the National Academy of Sciences of the United States of America*, *99*(20), 12795–12800.

Swithers. (2013). Artificial sweeteners produce the counterintuitive effect of inducing metabolic derangements. *Trends in Endocrinology and Metabolism*, *24*(9), 431–441.

Sylvetsky et al. (2017). Plasma concentrations of sucralose in children and adults. *Toxicological and Environmental Chemistry*, *99*(3), 535–542.

Szigeti et al. (2018). A blueprint for human whole-cell modeling. *Current Opinion in Systems Biology*, *7*, 8–15.

Takahashi et al. (2005). Space in systems biology of signaling pathways - Towards intracellular molecular crowding in silico. *FEBS Lett.*, *579*(8), 1783–1788.

Terzer et al. (2009). Genome-scale metabolic networks. *Wiley Interdisciplinary Reviews.Systems Biology and Medicine*, *1*(3), 285–297.

Thomas et al. (2015). BioNetFit: A fitting tool compatible with BioNetGen, NFsim and distributed computing environments. *Bioinformatics*, *32*(5), 798–800.

Tomar & De. (2013). Comparing methods for metabolic network analysis and an application to metabolic engineering. *Gene*, *521*(1), 1–14.

Tu. (1994). Review of Ordinary Differential Equations. In *Dynamical Systems* (pp. 5–38). Springer Berlin Heidelberg.

Turnbaugh et al. (2006). An obesity-associated gut microbiome with increased capacity for energy harvest. *Nature*, *444*(7122), 1027–1031.
Vallacher et al. (2010). Rethinking intractable conflict: The perspective of dynamical systems. *American Psychologist*, *65*(4), 262–278.
van Hijum et al. (2009). Mechanisms and Evolution of Control Logic in Prokaryotic Transcriptional Regulation. *Microbiology and Molecular Biology Reviews*, *73*(3), 481–509.
Vanlier et al. (2013). Parameter uncertainty in biochemical models described by ordinary differential equations. *Mathematical Biosciences*, *246*(2), 305–314.
Veloz et al. (2014). Reaction networks and evolutionary game theory. *Journal of Mathematical Biology*, *68*(1–2), 181–206.
Virtanen et al. (2020). SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature Methods*, *17*(3), 261–272.
Wade et al. (2016). Perspectives in mathematical modelling for microbial ecology. *Ecological Modelling*, *321*, 64–74.
Wang, H. et al. (2018). RAVEN 2.0: A versatile toolbox for metabolic network reconstruction and a case study on Streptomyces coelicolor. *PLoS Computational Biology*, *14*(10), e1006541.
Wang, Q. et al. (2007). Naïve Bayesian classifier for rapid assignment of rRNA sequences into the new bacterial taxonomy. *Applied and Environmental Microbiology*, *73*(16), 5261–5267.
Wang, Q. P. et al. (2018). Non-nutritive sweeteners possess a bacteriostatic effect and alter gut microbiota in mice. *PLoS ONE*, *13*(7).
Wang, X. G. et al. (2002). Structure of the lac operon galactoside acetyltransferase. *Structure*, *10*(4), 581–588.
Warne et al. (2019). Simulation and inference algorithms for stochastic biochemical reaction networks: From basic concepts to state-of-the-art. *Journal of the Royal Society Interface*, *16*(151).
Weitz & Dushoff. (2015). Modeling post-death transmission of Ebola: Challenges for inference and opportunities for control. *Scientific Reports*, *5*.
Wellek. (1996). A new approach to equivalence assessment in standard comparative bioavailability trials by means of the Mann-Whitney statistic. *Biometrical Journal*, *38*(6), 695–710.
Wellek. (2010). *Testing Statistical Hypotheses of Equivalence and Noninferiority*. Chapman and Hall/CRC.
Whitley. (1994). A genetic algorithm tutorial. *Statistics and Computing*, *4*(2), 65–85.
Wilkinson. (2009). Stochastic modelling for quantitative description of heterogeneous biological systems. *Nature Reviews Genetics*, *10*(2), 122–133.
Winter & Krömer. (2013). Fluxomics - connecting 'omics analysis and phenotypes. *Environmental Microbiology*, *15*(7), 1901–1916.
Wong et al. (2015). A generalised enzyme kinetic model for predicting the behaviour of complex biochemical systems. *FEBS Open Bio*, *5*, 226–239.
Woolcock & Narayan. (2000). Social capital: Implications for development theory, research, and policy. *World Bank Research Observer*, *15*(2), 225–249.

Yamagishi et al. (1999). Trust, gullibility, and social intelligence. *Asian Journal of Social Psychology*, *2*(1), 145–161.

Yilmaz et al. (2014). The SILVA and "all-species Living Tree Project (LTP)" taxonomic frameworks. *Nucleic Acids Research*, *42*(D1).

Yordanov et al. (2016). A method to identify and analyze biological programs through automated reasoning. *NPJ Systems Biology and Applications*, *2*(August 2015), 16010.

Zak & Knack. (2001). Trust and growth. *The Economic Journal*, *111*(470).

Zhang et al. (2019). The structure of trust as a reflection of culture and institutional power structure: Evidence from four East Asian societies. *Asian Journal of Social Psychology*, *22*(1), 59–73.

Zhao et al. (2017). Akkermansia muciniphila improves metabolic profiles by reducing inflammation in chow diet-fed mice. *Journal of Molecular Endocrinology*, *58*(1), 1–14.

Zomorrodi et al. (2014). D-OptCom: Dynamic Multi-level and Multi-objective Metabolic Modeling of Microbial Communities. *ACS Synthetic Biology*, *3*(4), 247–257.

Zomorrodi et al. (2012). Mathematical optimization applications in metabolic networks. *Metabolic Engineering*, *14*(6), 672–686.

Zouridis & Hatzimanikatis. (2007). A model for protein translation: polysome self-organization leads to maximum protein synthesis rates. *Biophys. J.*, *92*(3), 717–730.

9. SUPPLEMENTARY MATERIAL

S 3-1 Schematic representation of rules to model gene expression and plasmid replication control.

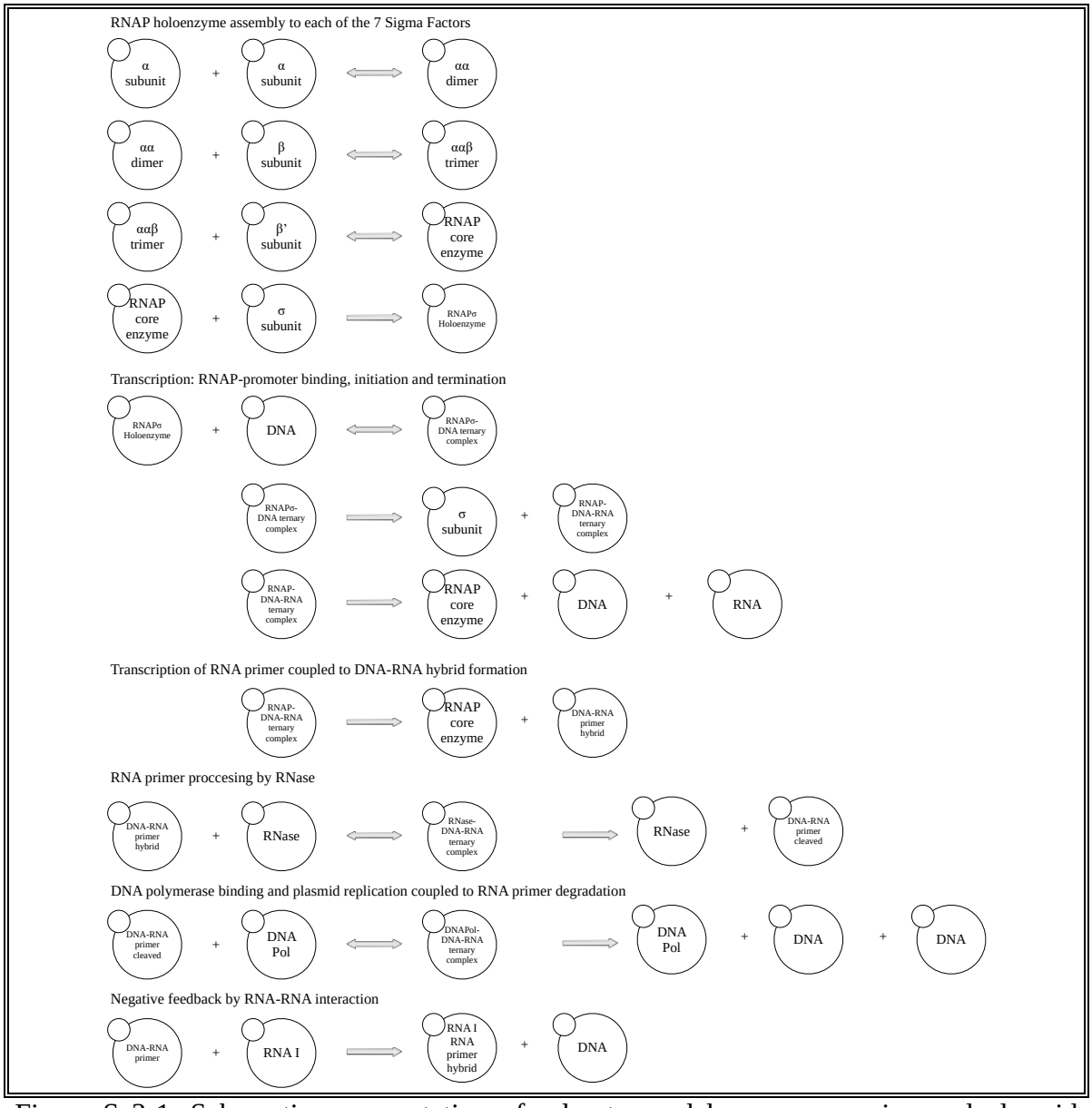


Figure S 3-1. Schematic representation of rules to model gene expression and plasmid replication control.

The top two groups of rules recapitulate the assembly of the RNAP and the holoenzymes, and the reversible binding of RNAP holoenzyme to promoters. Gene expression is modeled in the following sets of rules, while the remaining rules depict the control of plasmid replication.

S 3-2 Graphical description of the rules used to implement the SEIRD model.

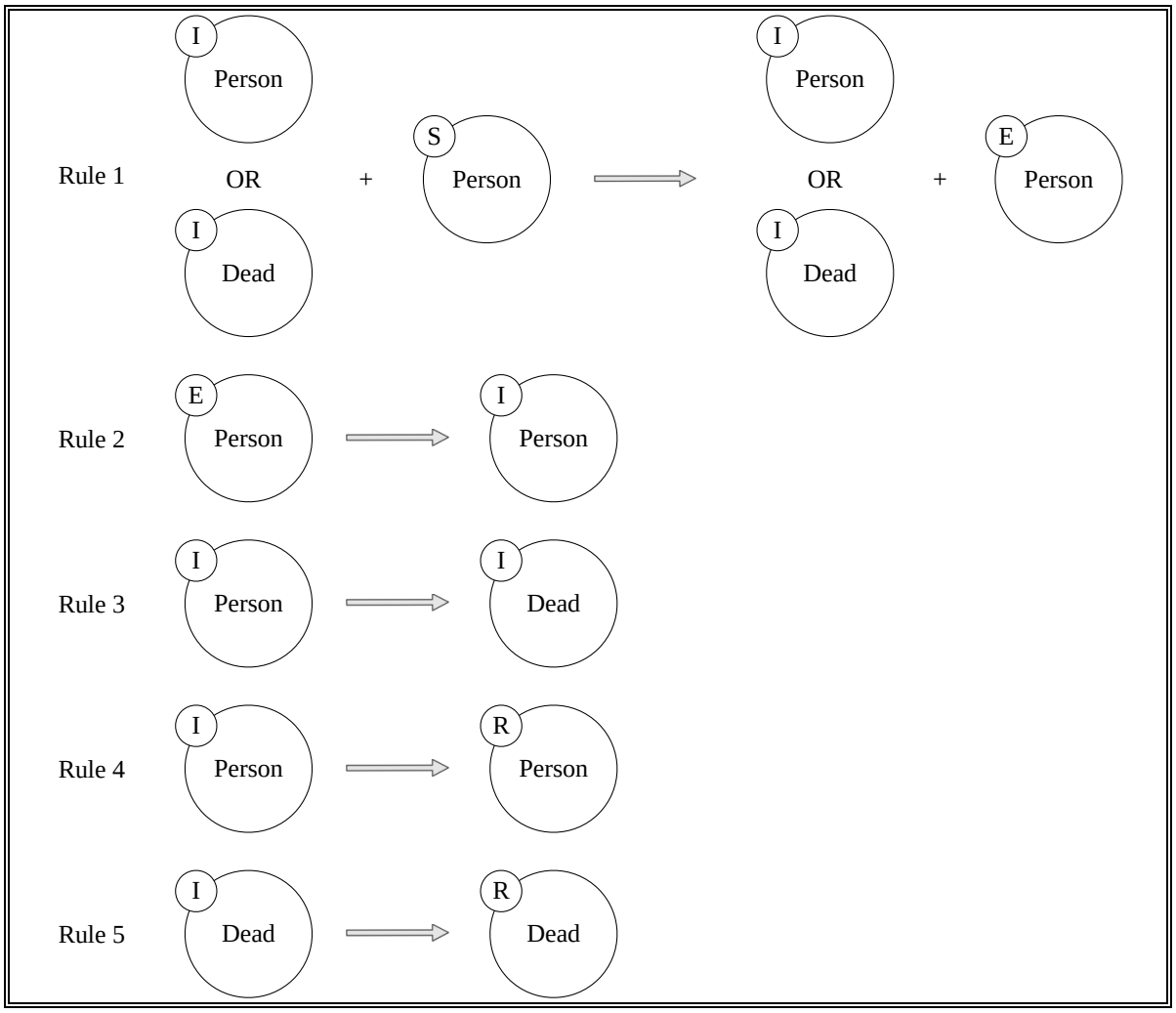


Figure S 3-2. Graphical description of the rules used to implement the SEIRD model.
Each big circle represents an agent and the small circles represent the site used to store the class to which the agent belongs. Therefore, agents named Person belonging either to the class susceptible or infected are denoted by Person (S) and Person (I), respectively. Rule 1; a susceptible person becomes an exposed person. This exposure can be the result of interaction either with an infected or a dead person. Rule 2; an exposed person becomes infected after an average incubation period of 7 days. Rule 3; an infected person dies at a certain rate. Rule 4; an infected person is removed from the simulation by acquiring immunity at a certain rate. Rule 5; a dead infected person can be removed from the simulation by being buried/incinerated at a certain rate. All transition rates were obtained from an ODE model of the Ebola disease (Weitz & Dushoff, 2015).

S 3-3 Graphical description of the rules of the PD model.

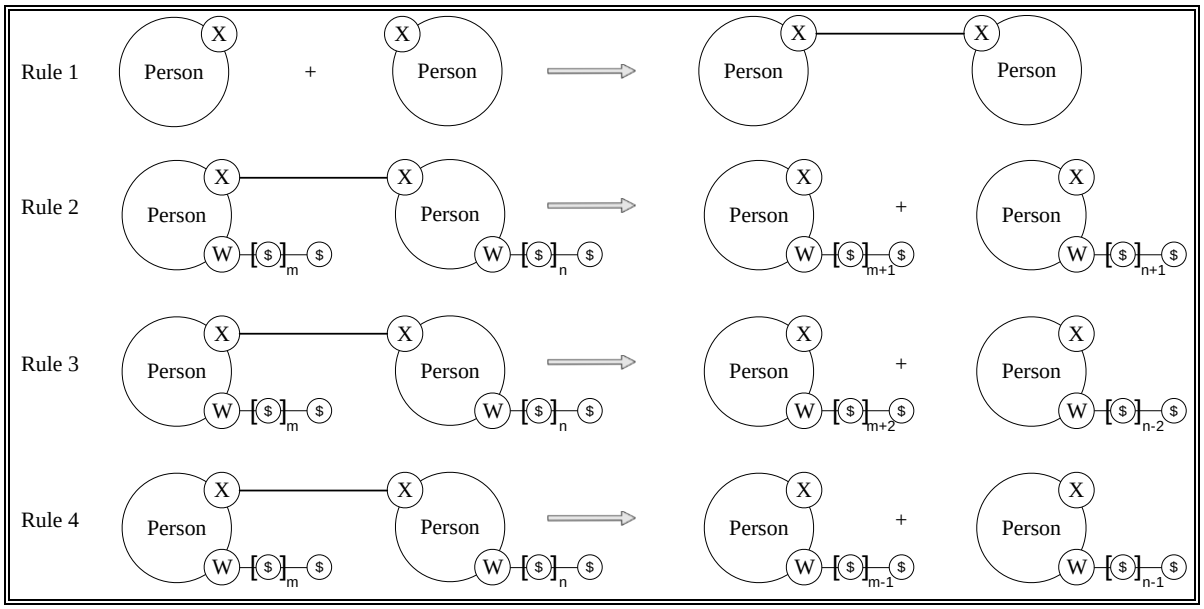


Figure S 3-3. Graphical description of the rules of the PD model.
Rule 1; a rule of entanglement/interaction by which two free agents (individuals) become a complex through their interaction site *X*. Rule 2; disentanglement/dissociation by cooperation, a complex breaks apart and every agent receives a payoff that is stored on its own wallet site (W). Rule 3; disentanglement-dissociation by temptation, a complex breaks apart leaving one agent with the temptation payoff on its *W* site and the other with the sucker´s payoff on its W site. Rule 4, disentanglement/dissociation by mutual defection, a complex of agents break apart and each agent receives the punishment payoff on its W site.

S 4-1. Lactose metabolic network.

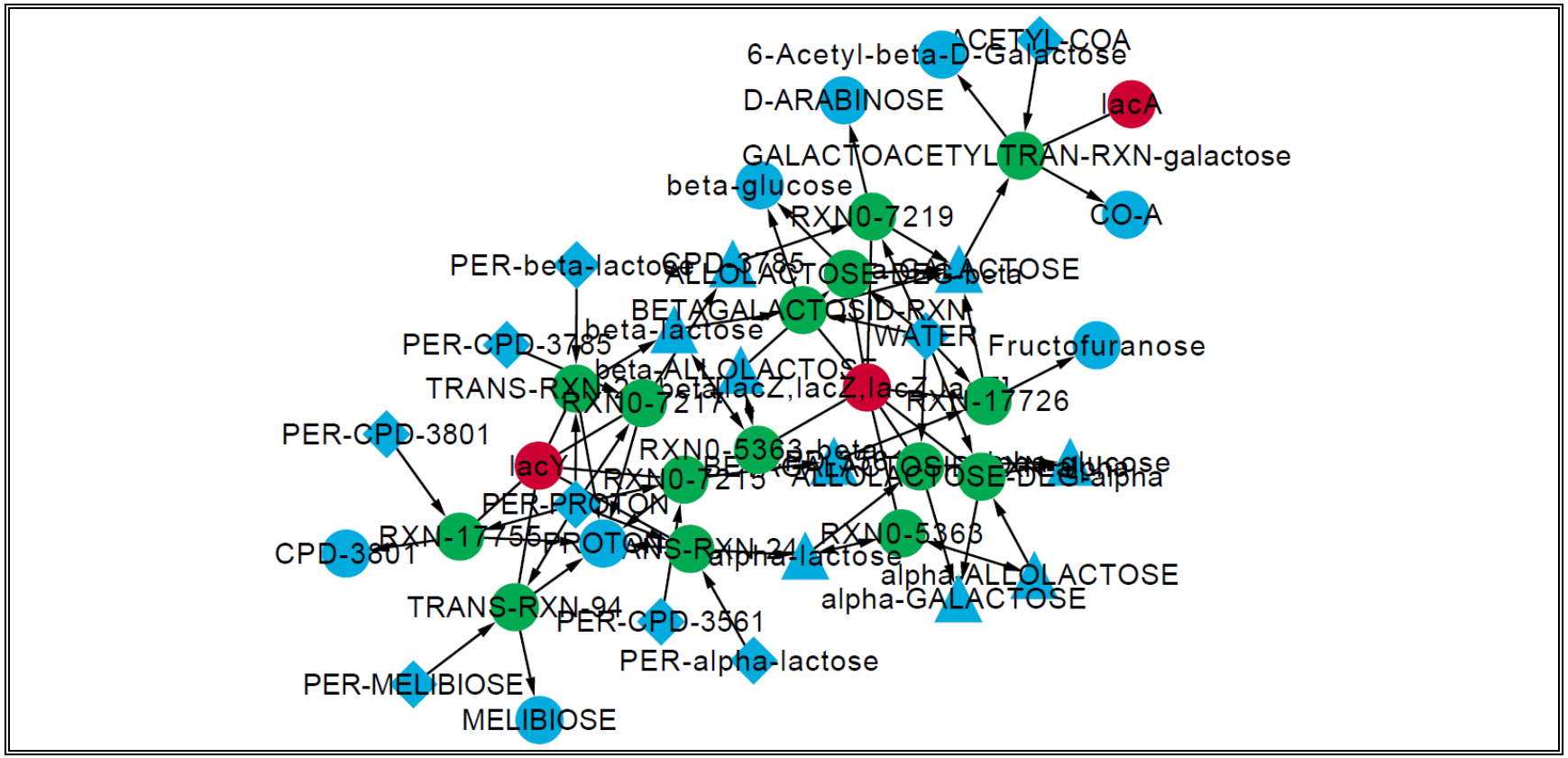


Figure S 4-1. Lactose metabolic network.
The figure shows the metabolic network constructed with data from the EcoCyc Database v24. Reactions are shown in green, gene products that catalyze those reactions are shown in red (undirected edges), and metabolites are shown in cyan (directed edges). Node shapes for metabolites represent substrates (diamonds), intermediates (triangles), and products (circles). Reaction reversibility is shown from the available information. The image was created from Supplementary Table S 4-2 with the Cytoscape software v3.7.2. Visualization from https://github.com/networkbiolab/atlas/blob/master/cytoscape-sessions/ecoli-lactose.cys.

S 4-2. Two representations of the protein-protein interaction network for the *E. coli* lactose consumption.

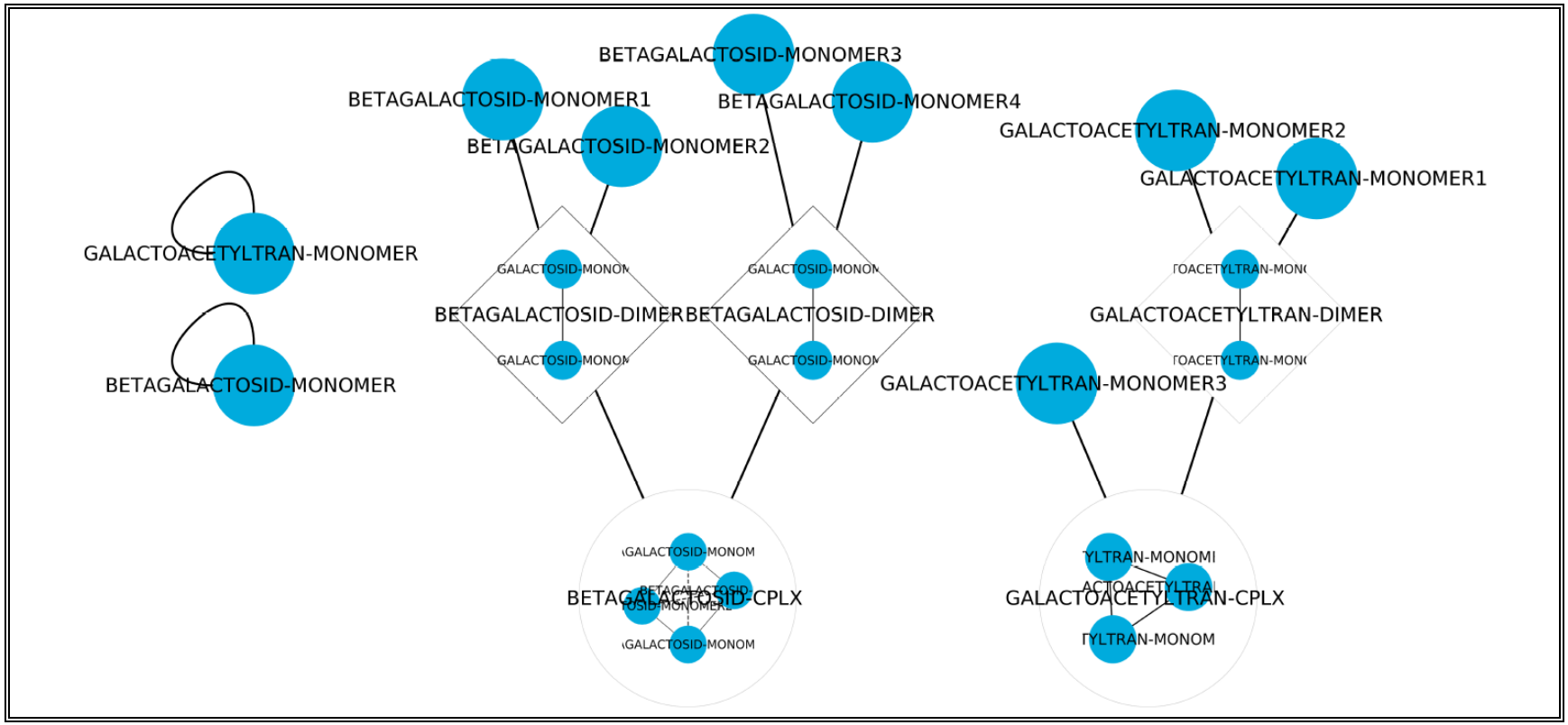


Figure S 4-2. Two representations of the protein-protein interaction network for the *E. coli* lactose consumption.
The left side of the figure shows the canonical representation of a protein-protein interaction network without any further encoded information. In the canonical form, both monomers interact with molecules of the same kind but disregarding stoichiometry and assembly pathways. The right side shows a collapsed hypergraph (nodes encoding subnetworks) representing how each protein complex (white nodes) is formed by the interaction of protein monomers and showing the stoichiometry for each complex. In addition, the collapsed hypergraph encodes the assembly steps, from monomers to fully assembled complexes. The image was created manually with help of the Cytoscape software v3.7.2.

S 4-3. Magnification of the glucose dynamics in response to allolactose activity.

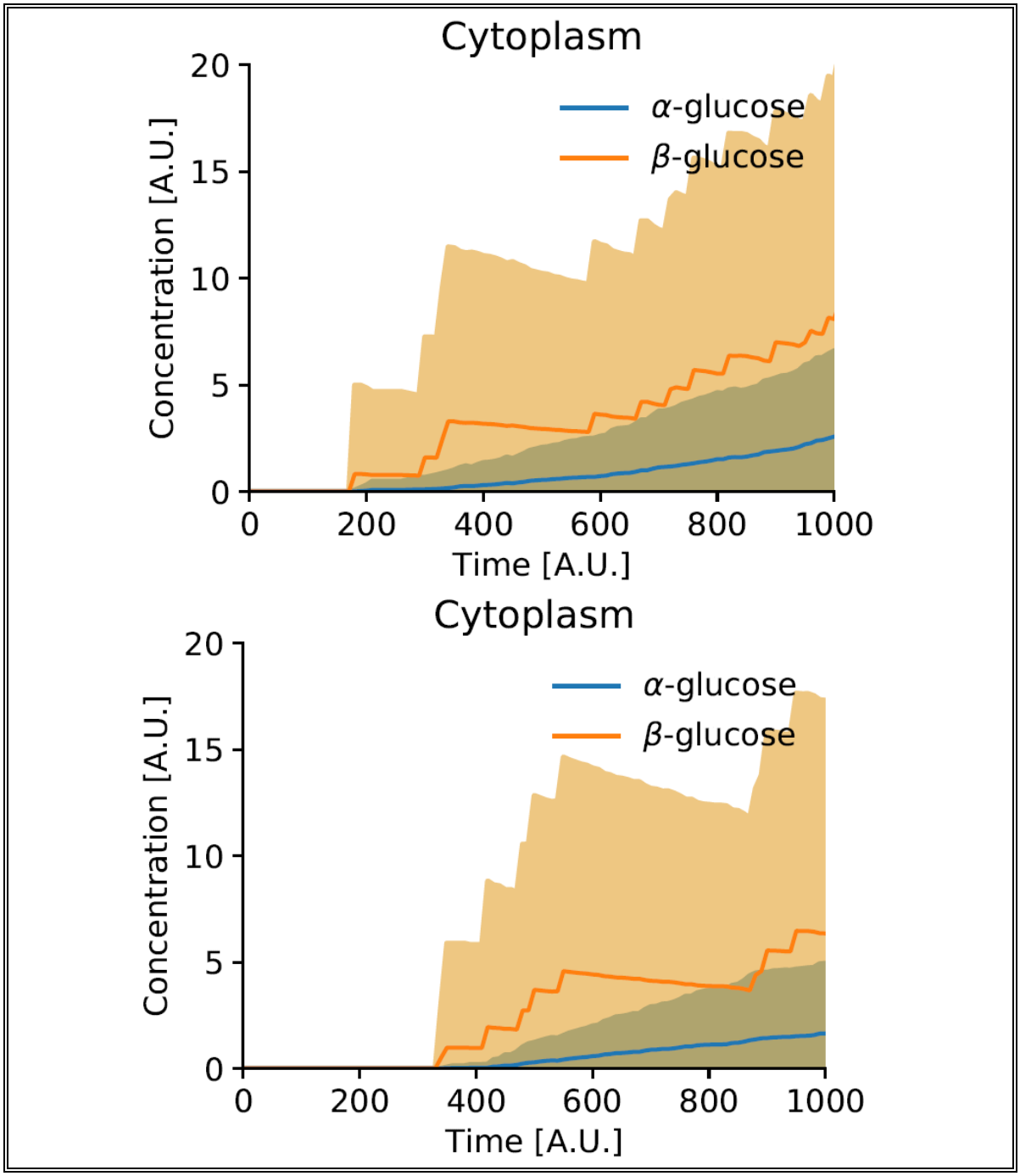


Figure S 4-3. Magnification of the glucose dynamics in response to allolactose activity. Plots depict the early dynamics of glucose, the product of lactose degradation. Allolactose binding to lacI proteins allowed an earlier response of glucose (upper panel) in comparison with a hypothetical situation where allolactose could not bind lacI proteins (bottom panel).

S 4-4. Simulation of in silico genetic modifications.

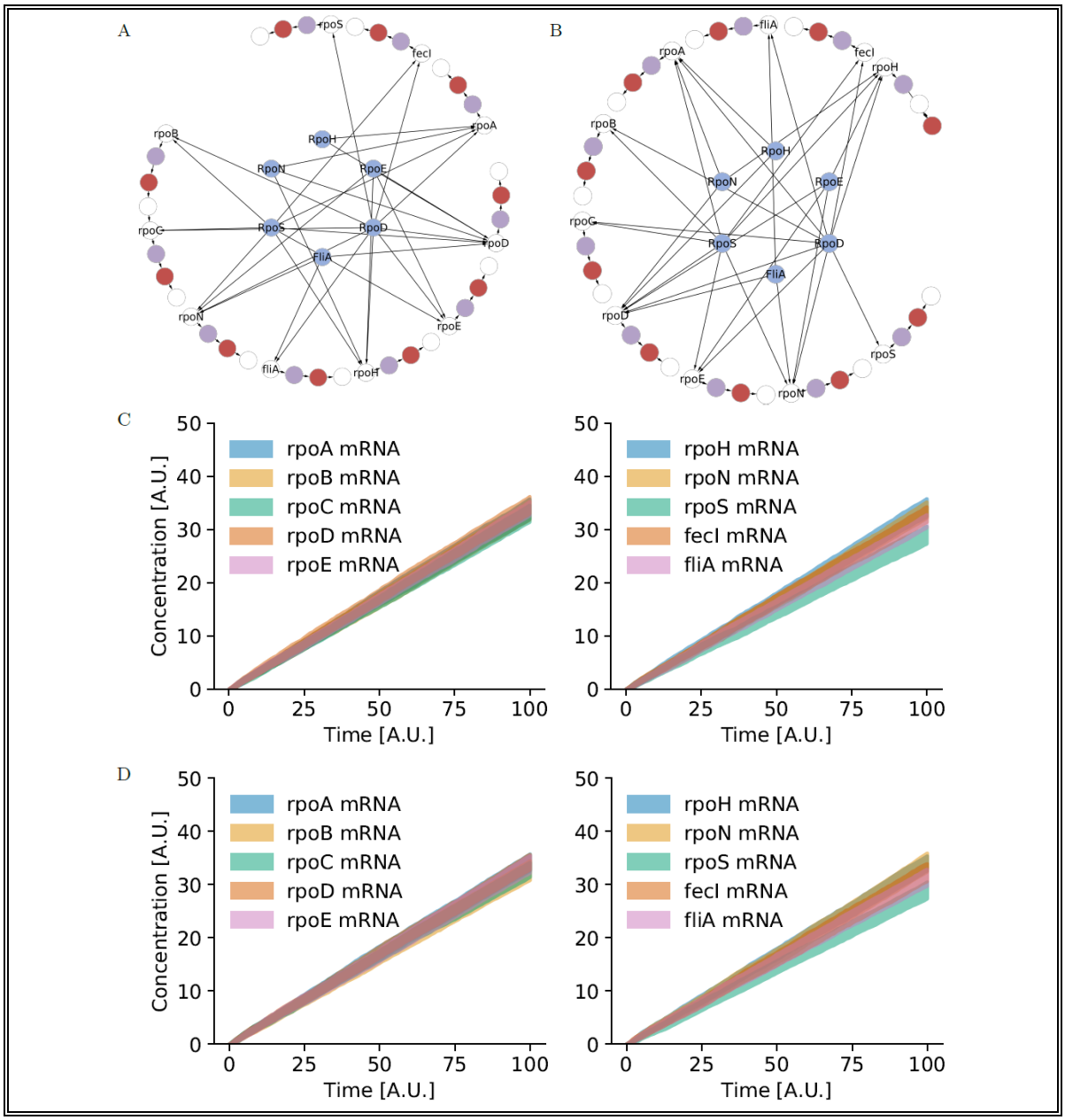


Figure S 4-4. Simulation of in silico genetic modifications.
The model for the regulation of the sigma factors was reconstructed from two genomic architectures. The panel A shows the in silico modification to add an rpoB terminator (unlabeled white node) and an rpoC promoter (labeled white node). The panel B shows the in silico modification to split the rpoBC operon into independent operons. Panel C shows the stochastic simulation of the modifications depicted in panel A. In contrast, panel D depicts the dynamics for the modifications B. The area shows the average and one standard deviation from 100 stochastic simulations.

S 4-5. Co-expression networks of in silico genetic modifications.

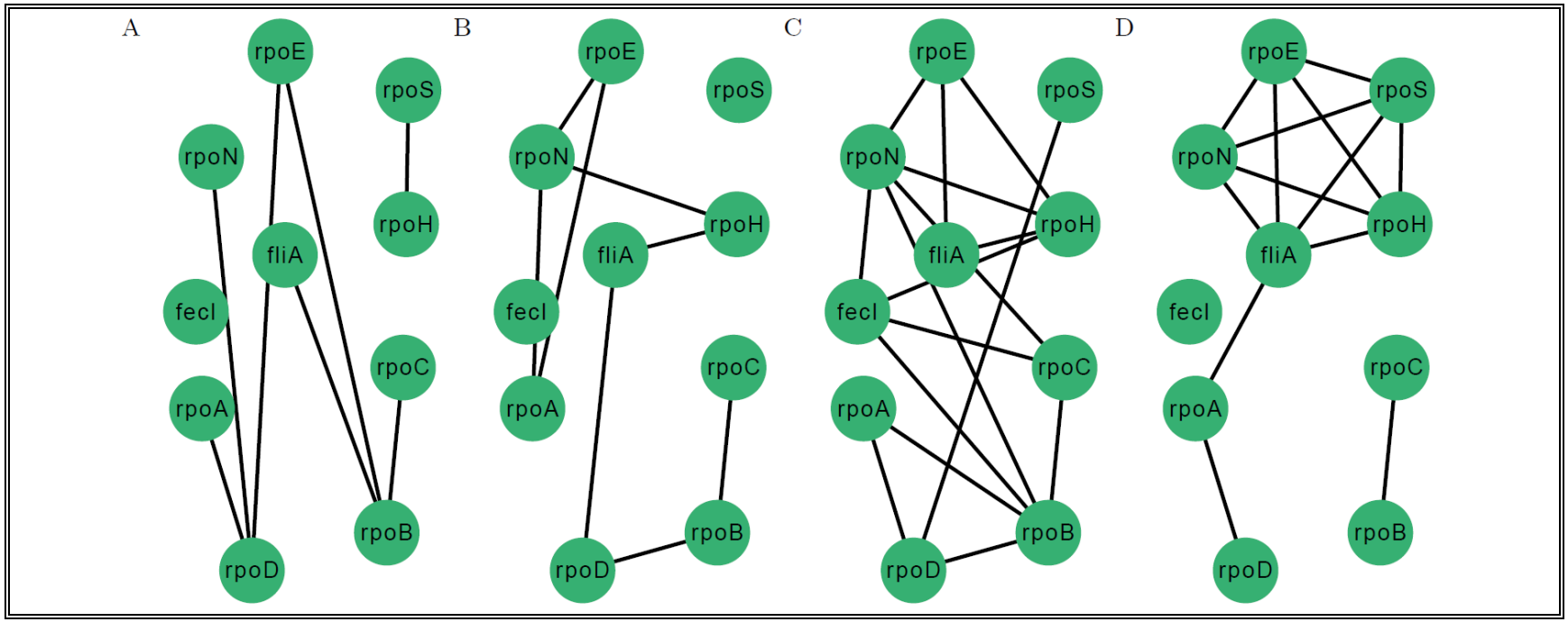


Figure S 4-5. Co-expression networks of in silico genetic modifications.
From left to right, each network represents the Pearson's correlation coefficient for two mRNA dynamics obtained from simulation greater than 0.95. Nodes represent the dynamics of the corresponding gene and edges exist between two dynamics if they are significantly correlated (p-value < 0.05). Networks were prepared from the average values of the ten simulations employed to determine the goodness of fit during parameter calibration with Pleione (Santibáñez et al., 2019).
(A) Co-expression network showing correlations for a model representing the natural genome architecture.
(B) Co-expression network for the model including the rpoC in silico promoter.
(C) Co-expression network for the model including the rpoB in silico terminator.
(D) Co-expression network showing correlations for the dynamics of a calibrated model for both in silico modifications, the internal rpoB terminator, and rpoC promoter.

S 4-6. Lactose metabolism from the EcoCyc database version 24.

Table S 4-1. Lactose metabolism from the EcoCyc database version 24.

	GENE OR COMPLEX	ENZYME LOCATION	REACTION	SUBSTRATES	PRODUCTS	FWD RATE	RVS RATE
1	BETAGALACTOSID-CPLX	cytosol	BETAGALACTOSID-RXN	CPD-15972,WATER	GALACTOSE,Glucopyranose	1.0	0.0
2	BETAGALACTOSID-CPLX	cytosol	RXN0-5363	Alpha-lactose	ALLOLACTOSE	1.0	1.0
3	BETAGALACTOSID-CPLX	cytosol	RXN-17726	CPD-3561,WATER	GALACTOSE,Fructofuranose	1.0	0.0
4	BETAGALACTOSID-CPLX	cytosol	RXN0-7219	CPD-3785,WATER	GALACTOSE,D-ARABINOSE	1.0	0.0
5	GALACTOACETYLTRAN-CPLX	cytosol	GALACTOACETYLTRAN-RXN	Beta-D-Galactosides,ACETYL-COA	6-Acetyl-Beta-D-Galactosides,CO-A	1.0	0.0
6	LACY-MONOMER	inner membrane	TRANS-RXN-24	PROTON,Alpha-lactose	PROTON,Alpha-lactose	1.0	1.0
7	LACY-MONOMER	inner membrane	TRANS-RXN-94	PROTON,MELIBIOSE	PROTON,MELIBIOSE	1.0	1.0
8	LACY-MONOMER	inner membrane	RXN0-7215	PROTON,CPD-3561	PROTON,CPD-3561	1.0	1.0
9	LACY-MONOMER	inner membrane	RXN0-7217	PROTON,CPD-3785	PROTON,CPD-3785	1.0	1.0
10	LACY-MONOMER	inner membrane	RXN-17755	PROTON,CPD-3801	PROTON,CPD-3801	1.0	1.0

S 4-7. Curated lactose metabolism from the EcoCyc database version 24 and literature.

Table S 4-2. Curated lactose metabolism from the EcoCyc database version 24 and literature.
Note: The locations for the ”spontaneous” reactions refer to the locations where the reactions occur.

	GENE OR COMPLEX	ENZYME LOCATION	REACTION	SUBSTRATES	PRODUCTS	FWD RATE	RVS RATE
1	spontaneous	cytosol	LACTOSE-MUTAROTATION	alpha-lactose	beta-lactose	1.0	1.0
2	spontaneous	cytosol	GALACTOSE-MUTAROTATION	alpha-GALACTOSE	beta-GALACTOSE	1.0	1.0
3	spontaneous	cytosol	GLUCOSE-MUTAROTATION	alpha-glucose	beta-glucose	1.0	1.0
4	LACY-MONOMER	inner membrane	TRANS-RXN-24	PER-PROTON,PER-alpha-lactose	PROTON,alpha-lactose	1.0	0.0
5	LACY-MONOMER	inner membrane	TRANS-RXN-24-beta	PER-PROTON,PER-beta-lactose	PROTON,beta-lactose	1.0	0.0
6	LACY-MONOMER	inner membrane	TRANS-RXN-94	PER-PROTON,PER-MELIBIOSE	PROTON,MELIBIOSE	1.0	0.0
7	LACY-MONOMER	inner membrane	RXN0-7215	PER-PROTON,PER-CPD-3561	PROTON,CPD-3561	1.0	0.0
8	LACY-MONOMER	inner membrane	RXN0-7217	PER-PROTON,PER-CPD-3785	PROTON,CPD-3785	1.0	0.0
9	LACY-MONOMER	inner membrane	RXN-17755	PER-PROTON,PER-CPD-3801	PROTON,CPD-3801	1.0	0.0
10	BETAGALACTOSID-CPLX	cytosol	BETAGALACTOSID-RXN	beta-lactose,WATER	beta-GALACTOSE,beta-glucose	1.0	0.0
11	BETAGALACTOSID-CPLX	cytosol	BETAGALACTOSID-RXN-alpha	alpha-lactose,WATER	alpha-GALACTOSE,alpha-glucose	1.0	0.0
12	BETAGALACTOSID-CPLX	cytosol	RXN0-5363	alpha-lactose	alpha-ALLOLACTOSE	1.0	1.0
13	BETAGALACTOSID-CPLX	cytosol	RXN0-5363-beta	beta-lactose	beta-ALLOLACTOSE	1.0	1.0
14	BETAGALACTOSID-CPLX	cytosol	ALLOLACTOSE-DEG-alpha	alpha-ALLOLACTOSE	alpha-GALACTOSE,alpha-glucose	1.0	0.0
15	BETAGALACTOSID-CPLX	cytosol	ALLOLACTOSE-DEG-beta	beta-ALLOLACTOSE	beta-GALACTOSE,beta-glucose	1.0	0.0
16	BETAGALACTOSID-CPLX	cytosol	RXN-17726	CPD-3561,WATER	beta-GALACTOSE,Fructofuranose	1.0	0.0
17	BETAGALACTOSID-CPLX	cytosol	RXN0-7219	CPD-3785,WATER	beta-GALACTOSE,D-ARABINOSE	1.0	0.0
18	GALACTOACETYLTRAN-CPLX	cytosol	GALACTOACETYLTRAN-RXN-galactose	beta-GALACTOSE,ACETYL-COA	6-Acetyl-beta-D-Galactose,CO-A	1.0	0.0

S 4-8. Curated lactose metabolism from the EcoCyc database version 24 and literature. New parameters.

Table S 4-3. Curated lactose metabolism from the EcoCyc database version 24 and literature. New parameters.
New parameters for the simulation of gene expression and regulation of gene expression.
Note: The locations for the ”spontaneous” reactions refer to the locations where the reactions occur.

	GENE OR COMPLEX	ENZYME LOCATION	REACTION	SUBSTRATES	PRODUCTS	FWD RATE	RVS RATE
1	spontaneous	cytosol	LACTOSE-MUTAROTATION	alpha-lactose	beta-lactose	0.001	0.001
2	spontaneous	cytosol	GALACTOSE-MUTAROTATION	alpha-GALACTOSE	beta-GALACTOSE	0.001	0.001
3	spontaneous	cytosol	GLUCOSE-MUTAROTATION	alpha-glucose	beta-glucose	0.001	0.001
4	lacY	inner membrane	TRANS-RXN-24	PER-PROTON,PER-alpha-lactose	PROTON,alpha-lactose	0.100	0.000
5	lacY	inner membrane	TRANS-RXN-24-beta	PER-PROTON,PER-beta-lactose	PROTON,beta-lactose	0.100	0.000
6	lacY	inner membrane	TRANS-RXN-94	PER-PROTON,PER-MELIBIOSE	PROTON,MELIBIOSE	0.100	0.000
7	lacY	inner membrane	RXN0-7215	PER-PROTON,PER-CPD-3561	PROTON,CPD-3561	0.100	0.000
8	lacY	inner membrane	RXN0-7217	PER-PROTON,PER-CPD-3785	PROTON,CPD-3785	0.100	0.000
9	lacY	inner membrane	RXN-17755	PER-PROTON,PER-CPD-3801	PROTON,CPD-3801	0.100	0.000
10	[lacZ,lacZ,lacZ,lacZ]	[cytosol,cytosol,cytosol,cytosol]	BETAGALACTOSID-RXN	beta-lactose,WATER	beta-GALACTOSE,beta-glucose	0.100	0.000
11	[lacZ,lacZ,lacZ,lacZ]	[cytosol,cytosol,cytosol,cytosol]	BETAGALACTOSID-RXN-alpha	alpha-lactose,WATER	alpha-GALACTOSE,alpha-glucose	0.100	0.000
12	[lacZ,lacZ,lacZ,lacZ]	[cytosol,cytosol,cytosol,cytosol]	RXN0-5363	alpha-lactose	alpha-ALLOLACTOSE	10.000	0.100
13	[lacZ,lacZ,lacZ,lacZ]	[cytosol,cytosol,cytosol,cytosol]	RXN0-5363-beta	beta-lactose	beta-ALLOLACTOSE	10.000	0.100
14	[lacZ,lacZ,lacZ,lacZ]	[cytosol,cytosol,cytosol,cytosol]	ALLOLACTOSE-DEG-alpha	alpha-ALLOLACTOSE,WATER	alpha-GALACTOSE,alpha-glucose	0.100	0.000
15	[lacZ,lacZ,lacZ,lacZ]	[cytosol,cytosol,cytosol,cytosol]	ALLOLACTOSE-DEG-beta	beta-ALLOLACTOSE,WATER	beta-GALACTOSE,beta-glucose	0.100	0.000
16	[lacZ,lacZ,lacZ,lacZ]	[cytosol,cytosol,cytosol,cytosol]	RXN-17726	CPD-3561,WATER	beta-GALACTOSE,Fructofuranose	1.000	0.000
17	[lacZ,lacZ,lacZ,lacZ]	[cytosol,cytosol,cytosol,cytosol]	RXN0-7219	CPD-3785,WATER	beta-GALACTOSE,D-ARABINOSE	1.000	0.000
18	[lacA,lacA,lacA]	[cytosol,cytosol,cytosol]	GALACTOACETYLTRAN-RXN-galactose	beta-GALACTOSE,ACETYL-COA	6-Acetyl-beta-D-Galactose,CO-A	1.000	0.000

S 4-9. Protein-protein, protein-metabolites, and Transcription Factors DNA binding sites interactions.

Table S 4-4. Protein-protein, protein-metabolites, and Transcription Factors DNA binding sites interactions.
The prefix ”BS-” defines a DNA binding site with coordinates relative to the lacZ transcription start site. Brackets denote a single complex. For the case of allolactose could not bind lacI proteins, parameters in rows 6-8 were set to zero

	SOURCE	TARGET	FWD RATE	RVS RATE	LOCATION
1	lacZ	lacZ	1.0	0.0000	cytosol
2	[lacZ,lacZ]	[lacZ,lacZ]	1.0	0.0000	cytosol
3	lacA	lacA	1.0	0.0000	cytosol
4	lacA	[lacA,lacA]	1.0	0.0000	cytosol
5	lacI	lacI	1.0	0.0000	cytosol
6	[lacI,lacI]	SMALL-alpha-ALLOLACTOSE	10.0	0.0001	cytosol
7	[lacI,lacI,SMALL-alpha-ALLOLACTOSE]	SMALL-alpha-ALLOLACTOSE	10.0	0.0001	cytosol
8	[lacI,lacI]	SMALL-beta-ALLOLACTOSE	10.0	0.0001	cytosol
9	[lacI,lacI,SMALL-beta-ALLOLACTOSE]	SMALL-beta-ALLOLACTOSE	10.0	0.0001	cytosol
10	[lacI,lacI]	BS-lacI-422-402	1.0	0.0100	cytosol
11	[lacI,lacI]	BS-lacI-21-1	1.0	0.0100	cytosol
12	[lacI,lacI]	BS-lacI-72-92	1.0	0.0100	cytosol

S 4-10. Physical interactions between the RNA Polymerase holoenzymes and promoters in the Lactose degradation Model.

Table S 4-5. Physical interactions between the RNA Polymerase holoenzymes and promoters in the Lactose degradation Model. The table shows the architecture of the three genes that were considered for the development of the lactose degradation Model. The prefix ”BS-” defines a DNA binding site with coordinates relative to the lacZ transcription start site. Brackets denote a single complex. Figure 4-2 shows the information represented in this table.

	UPSTREAM	DOWNSTREAM	RNAP FWD DOCK RATE	RNAP RVS DOCK RATE	RNAP FWD SLIDE RATE	RNAP FWD FALL RATE	RIB FWD DOCK RATE	RIB RVS DOCK RATE	RIB FWD SLIDE RATE	RIB FWD FALL RATE
1	[lacZ-pro4	lacZ-pro3	1.0	1.0	1.0					
2	lacZ-pro3	lacZ-pro2	1.0	1.0	1.0					
3	lacZ-pro2	BS-lacI-72-92	1.0	1.0	1.0					
4	BS-lacI-72-92	BS-lacI-21-1			1.0					
5	BS-lacI-21-1	lacZ-pro1			1.0					
6	lacZ-pro1	lacZ-rbs	1.0	1.0	1.0					
7	lacZ-rbs	lacZ-cds			1.0		1.0	1.0	1.0	1.0
8	lacZ-cds	BS-lacI-422-402			1.0					
9	BS-lacI-422-402	lacY-pro1			1.0					
10	lacY-pro1	lacY-rbs	0.0	0.0	1.0					
11	lacY-rbs	lacY-cds			1.0		1.0	1.0	1.0	1.0
12	lacY-cds	lacA-rbs			1.0					
13	lacA-rbs	lacA-cds			1.0		1.0	1.0	1.0	1.0
14	lacA-cds	lacA-ter1			1.0	1.0				
15	lacA-ter1	lacA-ter2]			1.0	1.0				

S 4-11. Benchmarks.

Table S 4-6. Benchmarks.
The table shows the components of each model developed in this work. The time to retrieve data from BioCyc was measured for the case of the central carbon and the genome-scale models, while the latter model provides a limit case. The time of ”simulation” is the total time of 100 stochastic simulations with KaSim v4 including the time to calculate the average and the standard deviation of simulations. A.U.: Arbitrary units.

	Lactose models				**Sigma Factors Model**					**Other models**		
	Metabolic	**Interactions**	**Gene expression**	**No binding**	**Reference**	**rpoC pro**	**rpoB ter**	**both**	**independent**	**Arabinose**	**Central Carbon Metabolism**	**Genome-Scale Metabolism**
Genes		3	4	4	10	10	10	10	10	7	200	
Enzymes	2	2	2	2								1380
Transporters	1	1	1	1								286
Metabolic reactions	18	18	18	18						9	458	3595
Metabolites	20	20	20	20						19	354	2072
Operons			1	1	9	9	9	9	10	4		
Protein-protein		5	5	5	10	10	10	10	10	10		
Protein-metabolites			4	4						11		
TFs-DNA			3	3						18		
RNAP-promotor					28	30	28	30	30			
Obtain data BioCyc											141 s	291 s
Construct model	0.74 s	1.062 s	2.231 s	2.164 s	2.897 s	3.89 s	3.73 s	3.13 s	5.11 s	2.57 s	53.9 s	44.4 s*
Combine models		0.857 s	3.150 s	3.980 s	2.300 s	3.29 s	3.13 s	3.63 s	3.51 s	3.37 s		
# Monomers	3	2	5	5	4	4	4	4	4	5	2	3
# Rules	18	23	62	62	125	130	127	132	130	106	332 (72.48%**)	2193 (61.00%**)
# Initials	26	32	49	49	38	39	39	40	40	52	520	0
# Observables	22	30	48	48	48	49	49	50	50	52	354	2072
Simulation Time to simulate	15.9 s 10 A.U.	17.8 s 10 A.U.	324 s 10000 A.U.	308 s 10000 A.U.	186 s 100 A.U.	256 s 100 A.U.	233 s 100 A.U.	255 s 100 A.U.	163 s 100 A.U.	527 s 1000 A.U.		

* Model reconstructed without initials for metabolites.
** Percentage of the total metabolic reactions in the network due to identical names.

S 4-12. Physical interactions between the RNA Polymerase holoenzymes and promoters in the Sigma Model.

Table S 4-7. Physical interactions between the RNA Polymerase holoenzymes and promoters in the Sigma Model.
The table shows the physical interactions for the RNA Polymerase of *Escherichia coli* and the associated promoters. Brackets denote a single protein complex.

	SOURCE	TARGET	FWD DOCK RATE	RVS DOCK RATE	FWD SLIDE RATE
1	[rpoA,rpoA,rpoB,rpoC,rpoD]	rpoA-pro1	1.0	1.0	1.0
2	[rpoA,rpoA,rpoB,rpoC,rpoD]	rpoB-pro1	1.0	1.0	1.0
3	[rpoA,rpoA,rpoB,rpoC,rpoD]	rpoD-pro1	1.0	1.0	1.0
4	[rpoA,rpoA,rpoB,rpoC,rpoD]	rpoE-pro1	1.0	1.0	1.0
5	[rpoA,rpoA,rpoB,rpoC,rpoD]	rpoH-pro1	1.0	1.0	1.0
6	[rpoA,rpoA,rpoB,rpoC,rpoD]	rpoN-pro1	1.0	1.0	1.0
7	[rpoA,rpoA,rpoB,rpoC,rpoD]	rpoS-pro1	1.0	1.0	1.0
8	[rpoA,rpoA,rpoB,rpoC,rpoD]	fliA-pro1	1.0	1.0	1.0
9	[rpoA,rpoA,rpoB,rpoC,rpoD]	fecI-pro1	1.0	1.0	1.0
10	[rpoA,rpoA,rpoB,rpoC,rpoE]	rpoD-pro1	1.0	1.0	1.0
11	[rpoA,rpoA,rpoB,rpoC,rpoE]	rpoE-pro1	1.0	1.0	1.0
12	[rpoA,rpoA,rpoB,rpoC,rpoE]	rpoH-pro1	1.0	1.0	1.0
13	[rpoA,rpoA,rpoB,rpoC,rpoE]	rpoN-pro1	1.0	1.0	1.0
14	[rpoA,rpoA,rpoB,rpoC,rpoH]	rpoA-pro1	1.0	1.0	1.0
15	[rpoA,rpoA,rpoB,rpoC,rpoH]	rpoD-pro1	1.0	1.0	1.0
16	[rpoA,rpoA,rpoB,rpoC,rpoN]	rpoA-pro1	1.0	1.0	1.0
17	[rpoA,rpoA,rpoB,rpoC,rpoN]	rpoD-pro1	1.0	1.0	1.0
18	[rpoA,rpoA,rpoB,rpoC,rpoN]	rpoH-pro1	1.0	1.0	1.0
19	[rpoA,rpoA,rpoB,rpoC,rpoS]	fecI-pro1	1.0	1.0	1.0
20	[rpoA,rpoA,rpoB,rpoC,rpoS]	rpoA-pro1	1.0	1.0	1.0
21	[rpoA,rpoA,rpoB,rpoC,rpoS]	rpoB-pro1	1.0	1.0	1.0
22	[rpoA,rpoA,rpoB,rpoC,rpoS]	rpoD-pro1	1.0	1.0	1.0
23	[rpoA,rpoA,rpoB,rpoC,rpoS]	rpoE-pro1	1.0	1.0	1.0
24	[rpoA,rpoA,rpoB,rpoC,rpoS]	rpoH-pro1	1.0	1.0	1.0
25	[rpoA,rpoA,rpoB,rpoC,rpoS]	rpoN-pro1	1.0	1.0	1.0
26	[rpoA,rpoA,rpoB,rpoC,fliA]	rpoD-pro1	1.0	1.0	1.0
27	[rpoA,rpoA,rpoB,rpoC,fliA]	rpoN-pro1	1.0	1.0	1.0
28	[rpoA,rpoA,rpoB,rpoC,fliA]	fliA-pro1	1.0	1.0	1.0

S 4-13. Genomic architecture for the RNA Polymerase genes.

Table S 4-8. Genomic architecture for the RNA Polymerase genes.
The table shows the architecture of the ten genes that were considered for the development of the Sigma Model. Figure 4-3B shows in the outer ring the information contained in this table.

	UPSTREAM	DOWNSTREAM	RNAP FWD DOCK RATE	RNAP RVS DOCK RATE	RNAP FWD SLIDE RATE	RNAP FWD FALL RATE	RIB FWD DOCK RATE	RIB RVS DOCK RATE	RIB FWD SLIDE RATE	RIB FWD FALL RATE
1	[rpoA-pro1	rpoA-rbs	1.0	1.0	1.0					
2	rpoA-rbs	rpoA-cds			1.0		1.0	1.0	1.0	1.0
3	rpoA-cds	rpoA-ter1]			1.0	1.0				
4	[rpoB-pro1	rpoB-rbs	1.0	1.0	1.0					
5	rpoB-rbs	rpoB-cds			1.0		1.0	1.0	1.0	1.0
6	rpoB-cds	rpoC-rbs			1.0					
7	rpoC-rbs	rpoC-cds			1.0		1.0	1.0	1.0	1.0
8	rpoC-cds	rpoC-ter1]			1.0	1.0				
9	[rpoD-pro1	rpoD-rbs	1.0	1.0	1.0					
10	rpoD-rbs	rpoD-cds			1.0		1.0	1.0	1.0	1.0
11	rpoD-cds	rpoD-ter1]			1.0	1.0				
12	[rpoE-pro1	rpoE-rbs	1.0	1.0	1.0					
13	rpoE-rbs	rpoE-cds			1.0		1.0	1.0	1.0	1.0
14	rpoE-cds	rpoE-ter1]			1.0	1.0				

S 4-13. Genomic architecture for the RNA Polymerase genes (continued).

Table S 4-8. Genomic architecture for the RNA Polymerase genes (continued).
The table shows the architecture of the ten genes that were considered for the development of the Sigma Model. Figure 4-3B shows in the outer ring the information contained in this table.

	UPSTREAM	DOWNSTREAM	RNAP FWD DOCK RATE	RNAP RVS DOCK RATE	RNAP FWD SLIDE RATE	RNAP FWD FALL RATE	RIB FWD DOCK RATE	RIB RVS DOCK RATE	RIB FWD SLIDE RATE	RIB FWD FALL RATE
15	[rpoH-pro1	rpoH-rbs	1.0	1.0	1.0					
16	rpoH-rbs	rpoH-cds			1.0		1.0	1.0	1.0	1.0
17	rpoH-cds	rpoH-ter1]			1.0	1.0				
18	[rpoN-pro1	rpoN-rbs	1.0	1.0	1.0					
19	rpoN-rbs	rpoN-cds			1.0		1.0	1.0	1.0	1.0
20	rpoN-cds	rpoN-ter1]			1.0	1.0				
21	[rpoS-pro1	rpoS-rbs	1.0	1.0	1.0					
22	rpoS-rbs	rpoS-cds			1.0		1.0	1.0	1.0	1.0
23	rpoS-cds	rpoS-ter1]			1.0	1.0				
24	[fliA-pro1	fliA-rbs	1.0	1.0	1.0					
25	fliA-rbs	fliA-cds			1.0		1.0	1.0	1.0	1.0
26	fliA-cds	fliA-ter1]			1.0	1.0				
27	[fecI-pro1	fecI-rbs	1.0	1.0	1.0					
28	fecI-rbs	fecI-cds			1.0		1.0	1.0	1.0	1.0
29	fecI-cds	fecI-ter1]			1.0	1.0				

S 4-14. Effect of the in silico deletions of DNA coding sequences.

Table S 4-9. Effect of the in silico deletions of DNA coding sequences.
The table shows the logarithm of the fold change compared to a reference model. Negative values represent lower observed expression and positive values greater observed expression in comparison with the reference condition. Columns show the in silico deletion and rows the effect of the deletion on the expression levels at the end of 1000 simulations of 100 units of time.

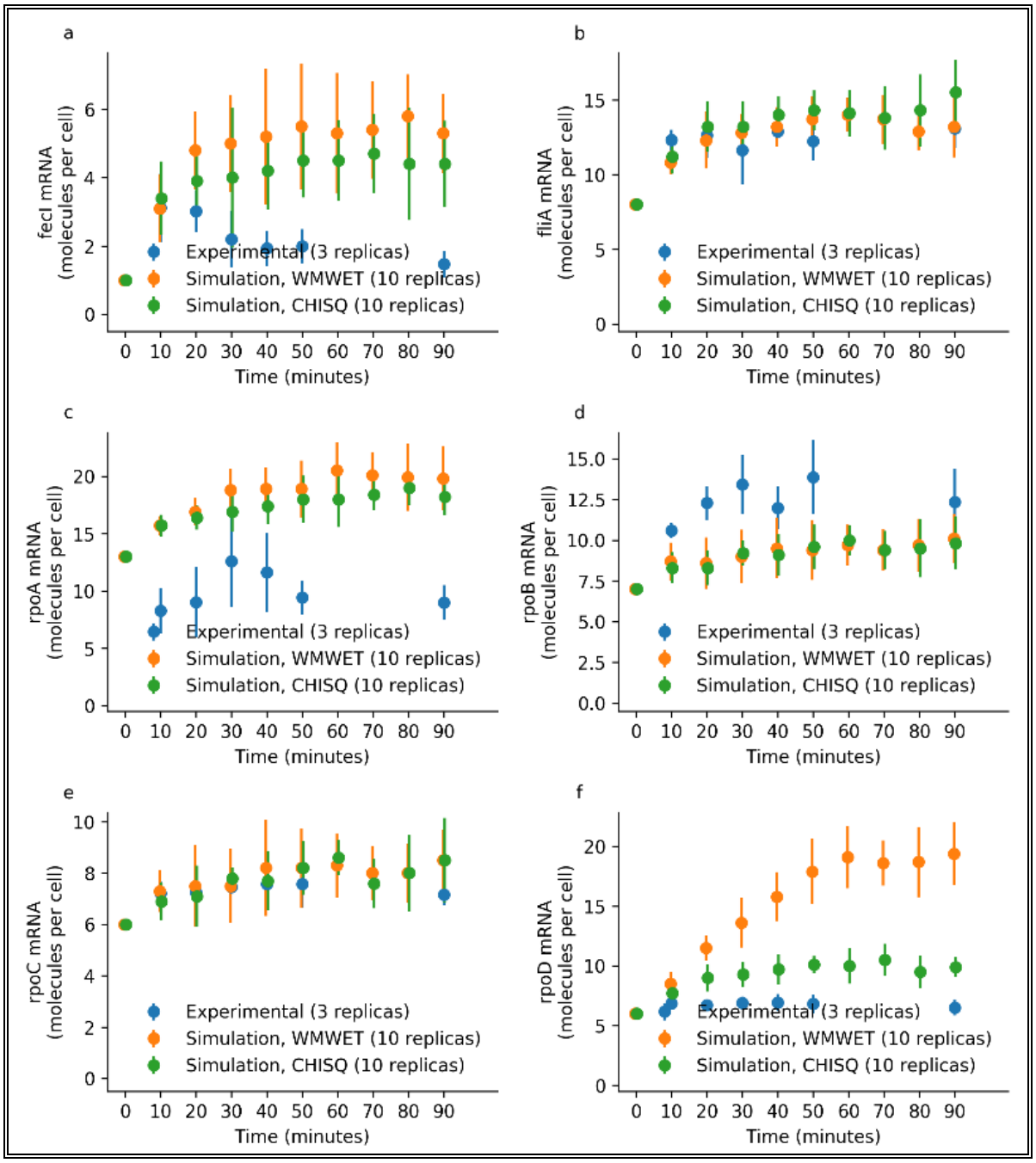
	ΔrpoD	**ΔrpoE**	**ΔrpoH**	**ΔrpoN**	**ΔrpoS**	**ΔfliA**	**ΔfecI**
rpoS mRNA	-7.88e+00	3.17e-02	1.02e-02	2.96e-02		2.93e-02	9.41e-03
fliA mRNA	-1.32e-01	1.55e-02	-3.27e-03	6.85e-03	5.01e-02		-1.02e-03
rpoH mRNA	5.20e-02	-1.55e-02		-2.17e-02	1.65e-02	8.69e-03	-2.13e-03
rpoA mRNA	4.70e-02	2.35e-04	-1.01e-02	-7.21e-03	1.59e-03	8.50e-04	5.10e-03
rpoN mRNA	4.58e-02	-1.33e-02	-2.66e-03		1.53e-02	-1.30e-02	-6.51e-03
fecI mRNA	-3.27e-02	-7.07e-03	-7.58e-03	-5.13e-03	-7.14e-02	-7.03e-03	
rpoE mRNA	3.10e-02		4.96e-03	8.35e-04	-6.12e-03	1.35e-03	-2.86e-03
rpoC mRNA	7.67e-03	7.47e-03	8.91e-03	4.91e-03	-2.72e-02	1.08e-02	4.11e-03
rpoB mRNA	6.29e-03	7.80e-03	9.98e-03	5.13e-03	-2.90e-02	1.03e-02	9.65e-04
rpoD mRNA		-1.59e-02	-1.25e-02	-7.84e-03	1.36e-02	-1.89e-02	-7.29e-03

S 4-15. False Discovery Rates of the effect of the in silico deletions of coding DNA sequences.

Table S 4-10. False Discovery Rates of the effect of the in silico deletions of coding DNA sequences.
The table shows the False Discovery Rate determined with the edgeR software (FDR < 0.05). Columns show the in silico deletion and rows show the FDR of the change of expression levels at the end of 1000 simulations of 100 units of time.

	ΔrpoD	**ΔrpoE**	**ΔrpoH**	**ΔrpoN**	**ΔrpoS**	**ΔfliA**	**ΔfecI**
rpoS mRNA	0.00e+00						
fliA mRNA	6.56e-29				3.77e-05		
rpoH mRNA	6.76e-06						
rpoA mRNA	4.54e-05						
rpoN mRNA	5.81e-05						
fecI mRNA	5.74e-03				2.49e-09		
rpoE mRNA	6.69e-03						
rpoC mRNA							
rpoB mRNA					4.03e-02		
rpoD mRNA							

S 5-1 Simulation of the best parameter set after calibration of the Core GRN Model



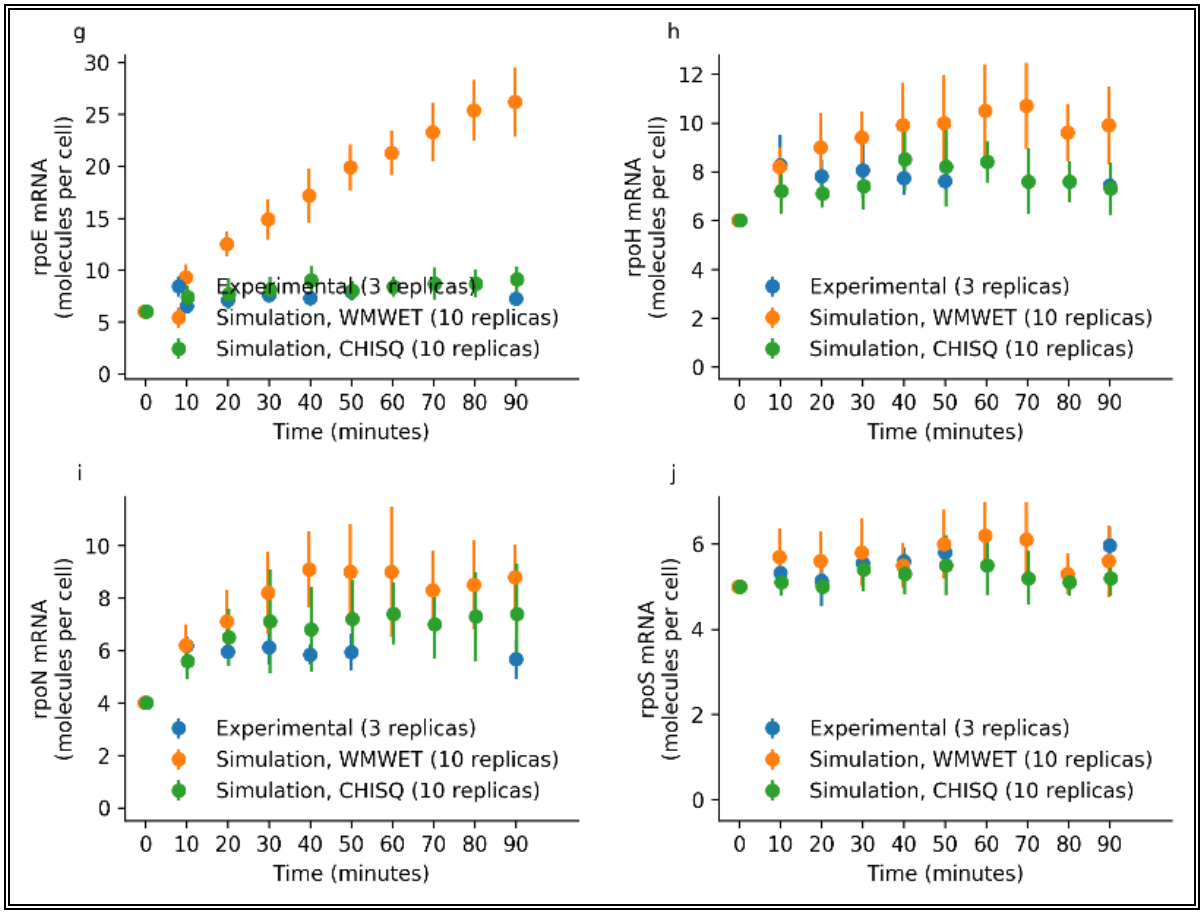
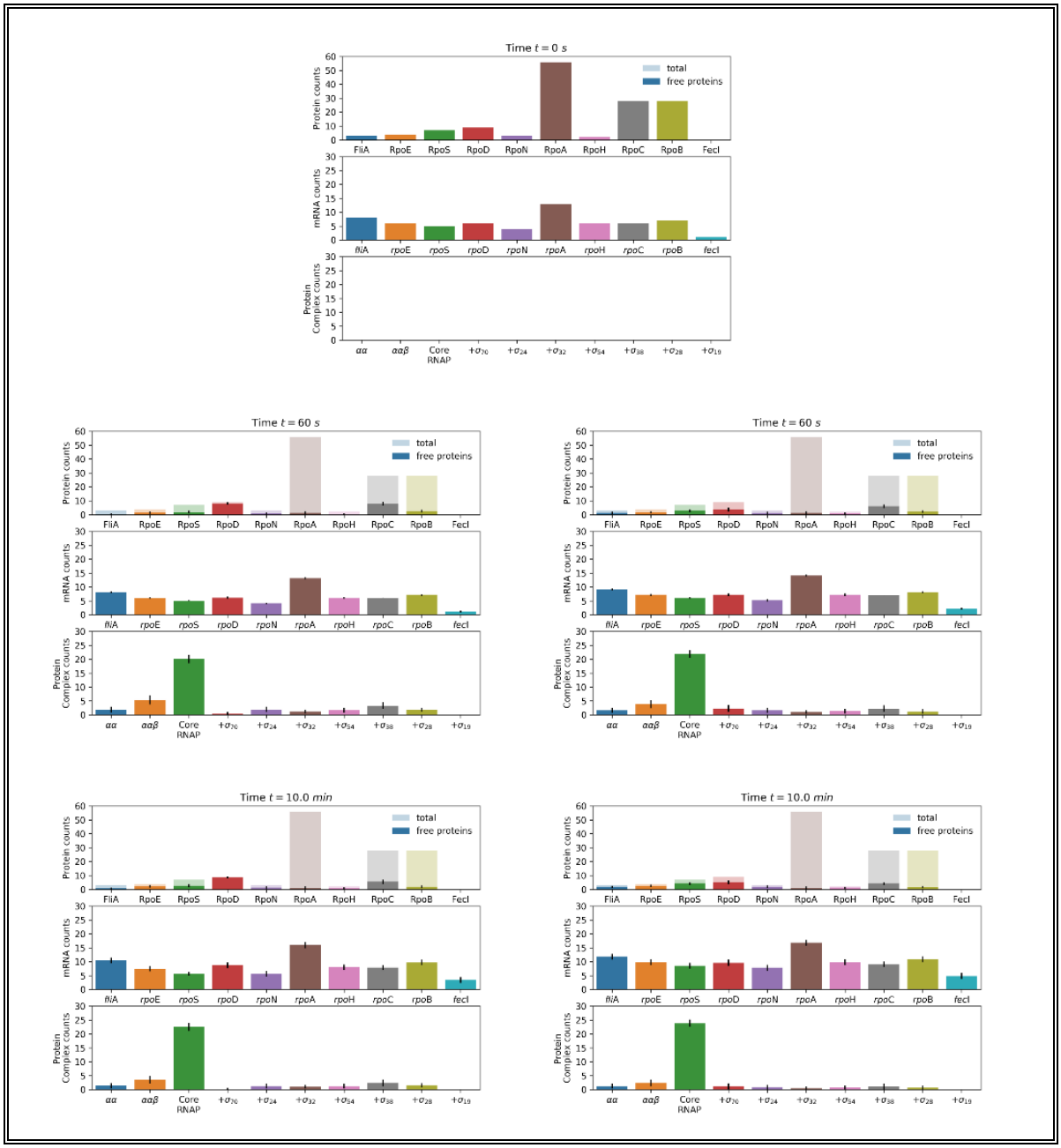


Figure S 5-1. Simulation of the best parameter set after calibration of the Core GRN Model. Simulation of the best parameter set after calibration of the Core GRN Model with the iterative Wellek´s test (WMWET) and the chi-square (CHISQ) fitness functions employing Strategy1. The panels show one of the ten mRNA which dynamics were reproduced by the Core GRN Model to some extent. There are 4 main outcomes: good agreement for both metrics, like the case of rpoC and rpoS mRNA; good agreement with only one fitness function, for instance, the rpoB and rpoE mRNAs; and complete lack of fit with both, as is seen for the rpoD mRNA and others.

S 5-2 Small multiple of simulations through the time.



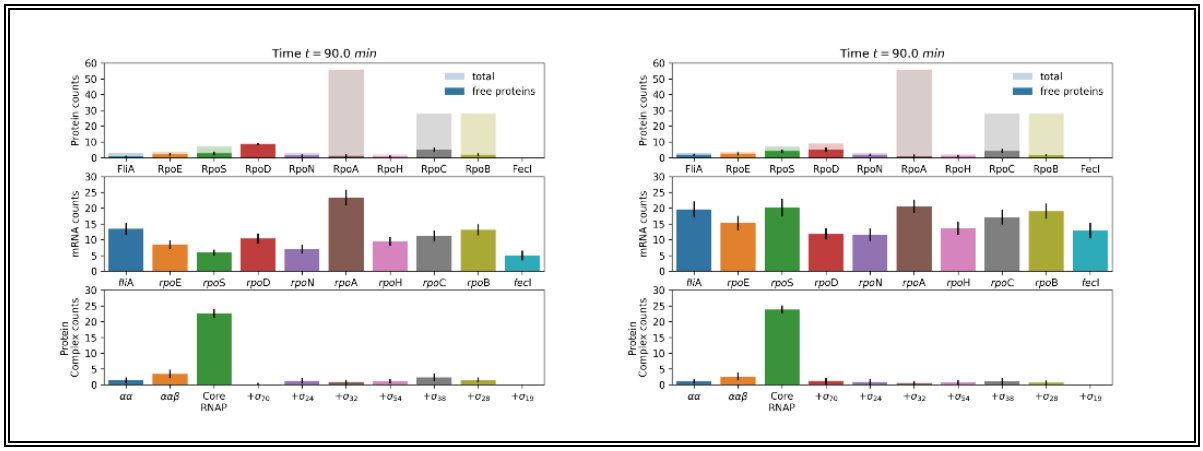
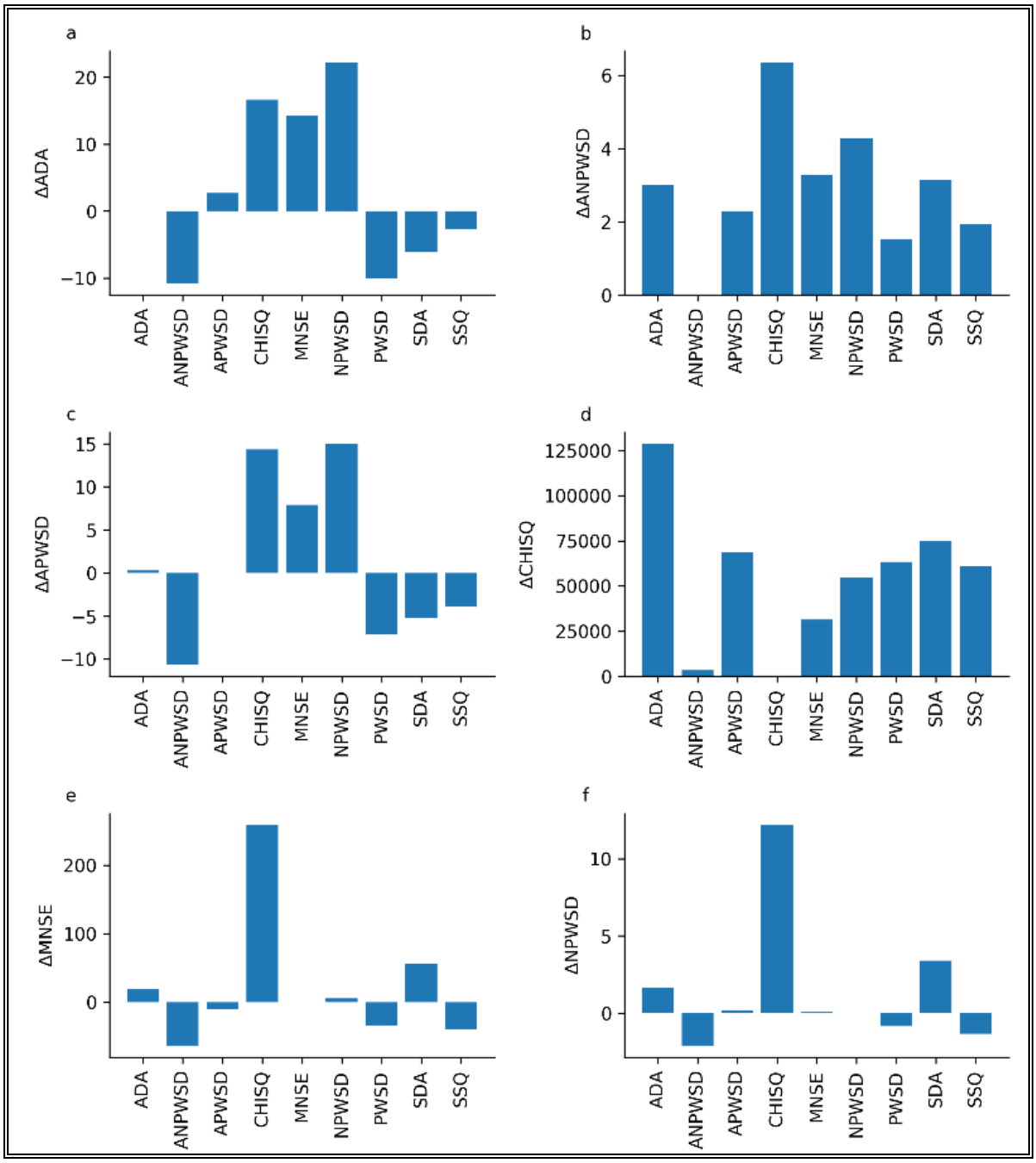


Figure S 5-2. Small multiple of simulations through the time.
Small multiple depicting the initial condition (top panel) and how it evolved through the simulated time (1, 10, and final time of 90 minutes) with model parameters before (panels at the left) and after calibration (panels at the right).

S 5-3 Difference for the error of the best models calibrated with Strategy1 at each single fitness function.



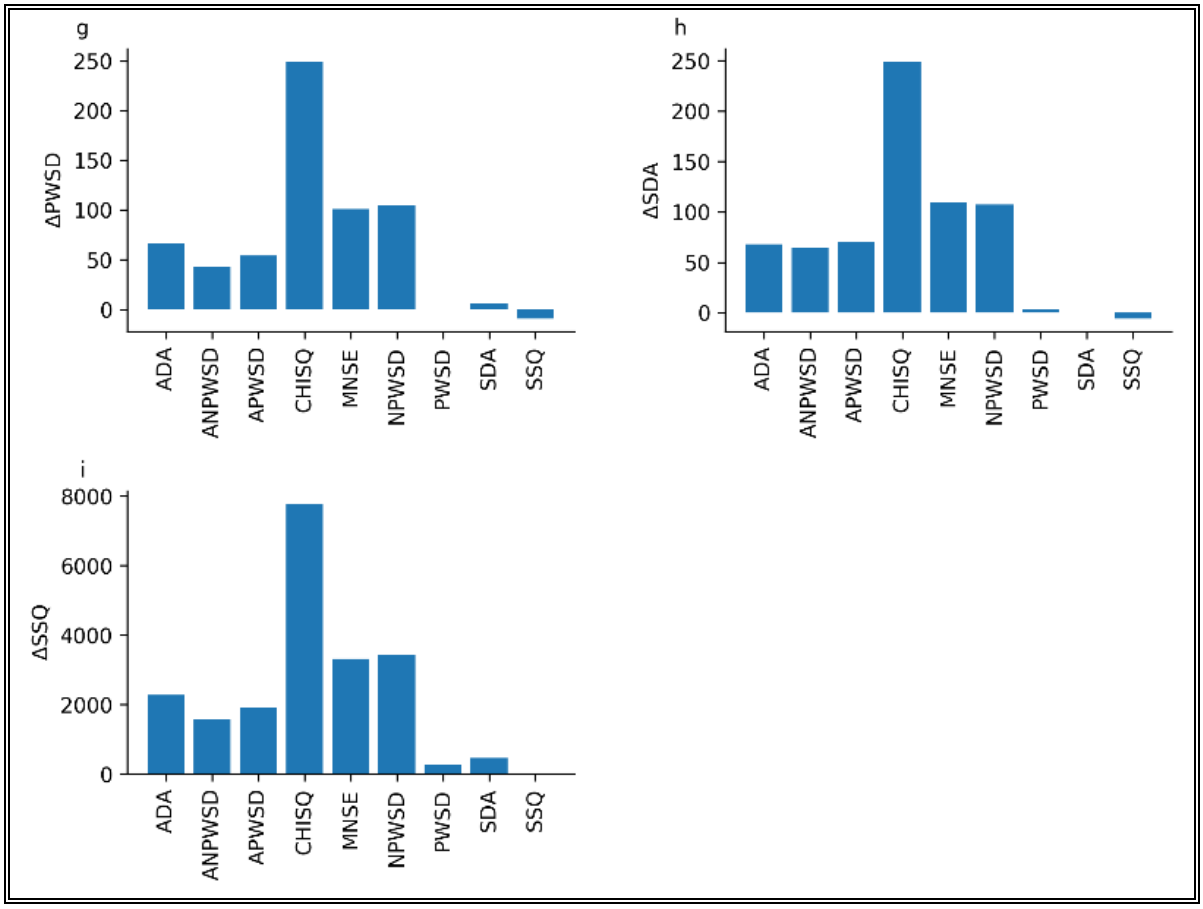


Figure S 5-3. The difference for the error of the best models calibrated with Strategy1 at each single fitness function.

The error determined for the first rank model at the end of the calibration is shown subtracted by the error found when the same fitness function was used as the objective function. Most of the fitness functions showed equal or lower capability to reduce other fitness errors, indicated by the positive difference. Interestingly, the absolute value of the difference of two averages (ADA, Equation 5-2, panel A) as well as other fitness functions were minimized even further employing another function as the objective.

S 5-4 Correlation coefficients (left) and p-values (right) for the ten fitness functions included in *Pleione*.

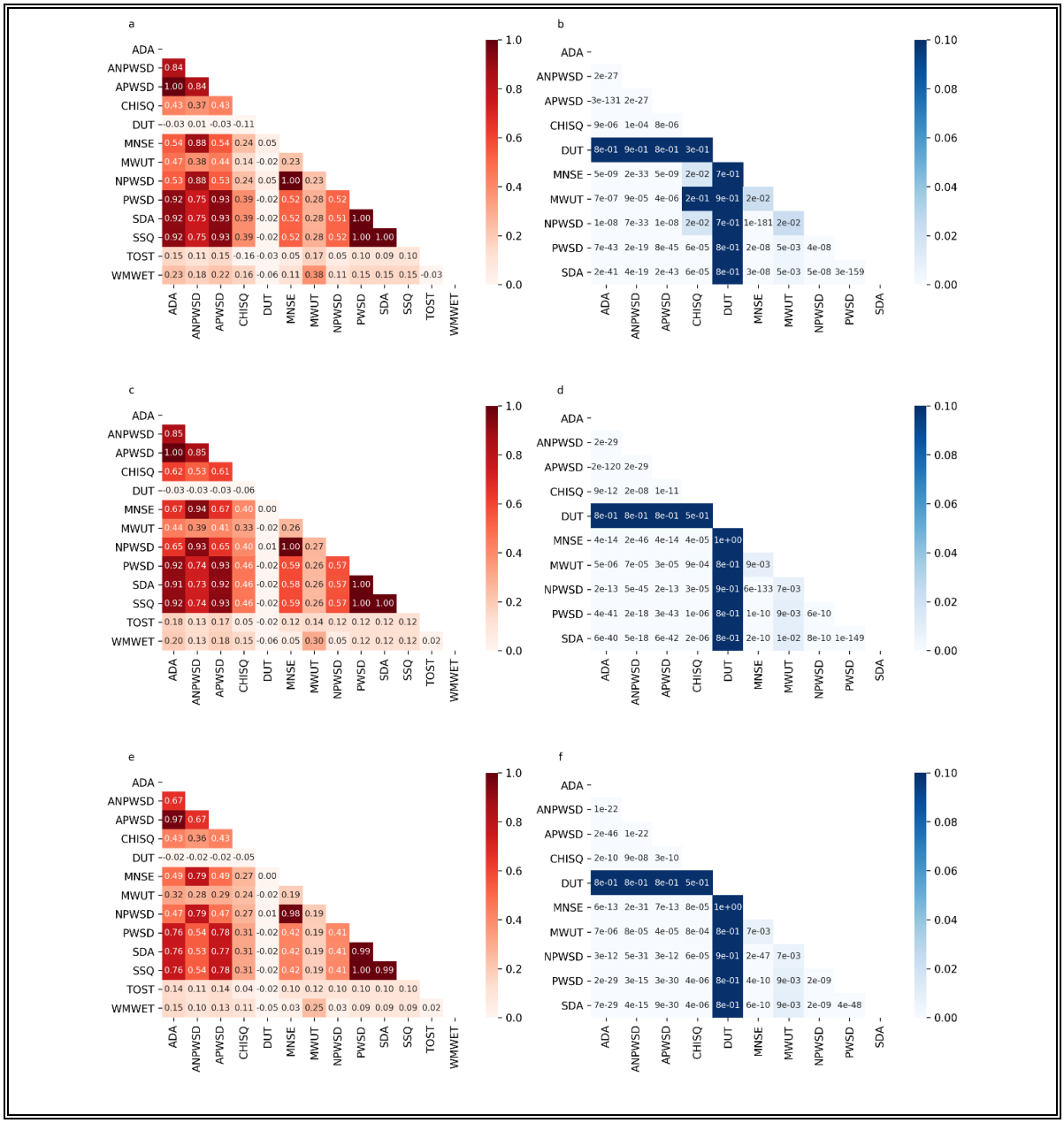
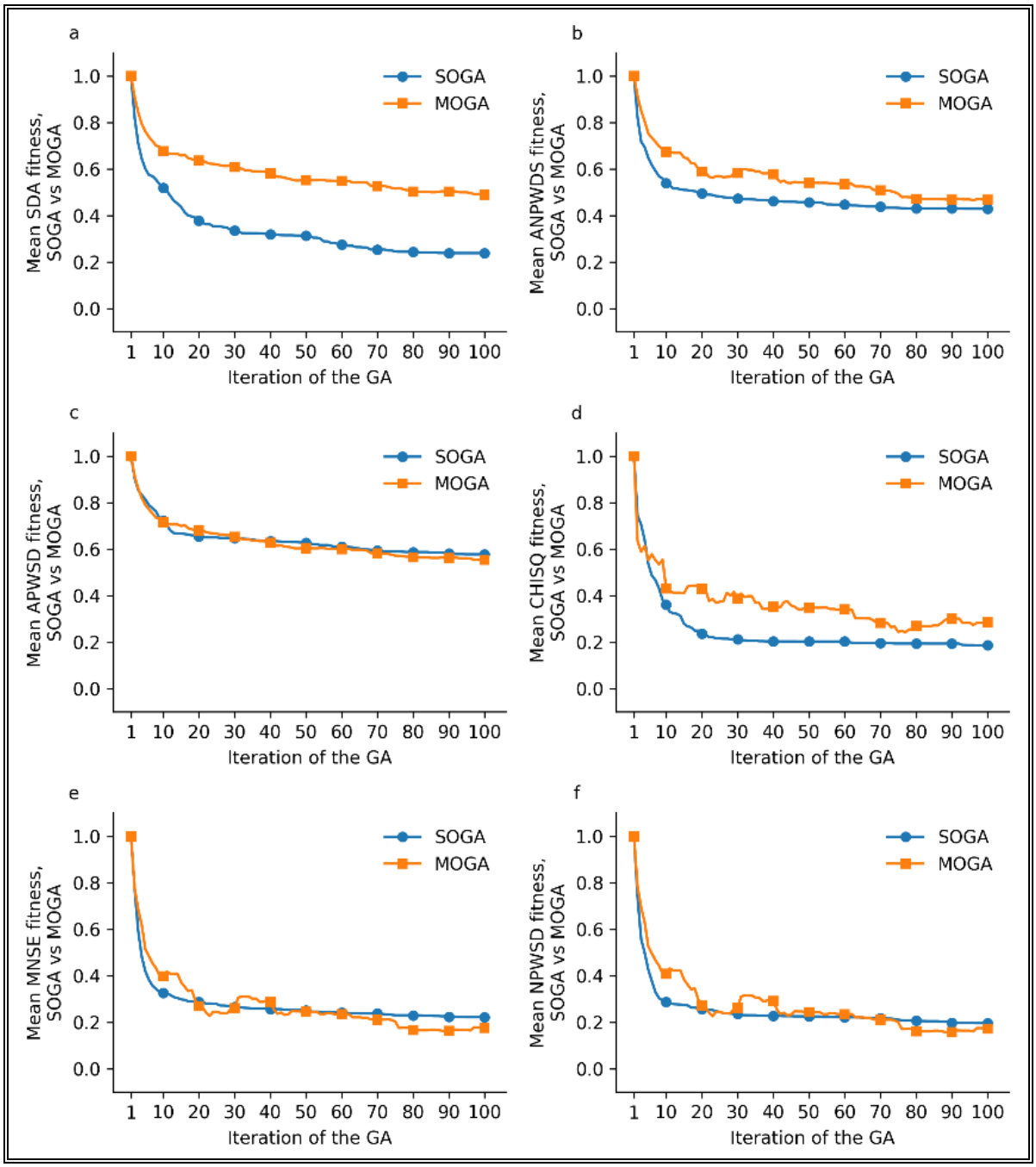


Figure S 5-4. Correlation coefficients (left) and p-values (right) for the ten fitness functions included in Pleione.

A, B. Pearson´s correlation coefficient; C, D. Spearman ρ correlation coefficient; and E, F. Kendall τ correlation coefficient. All coefficients were calculated with the python SciPy package at the first iteration of 100 individuals to calibrate the Core GRN Model.

S 5-5 Fractional mean error convergence.



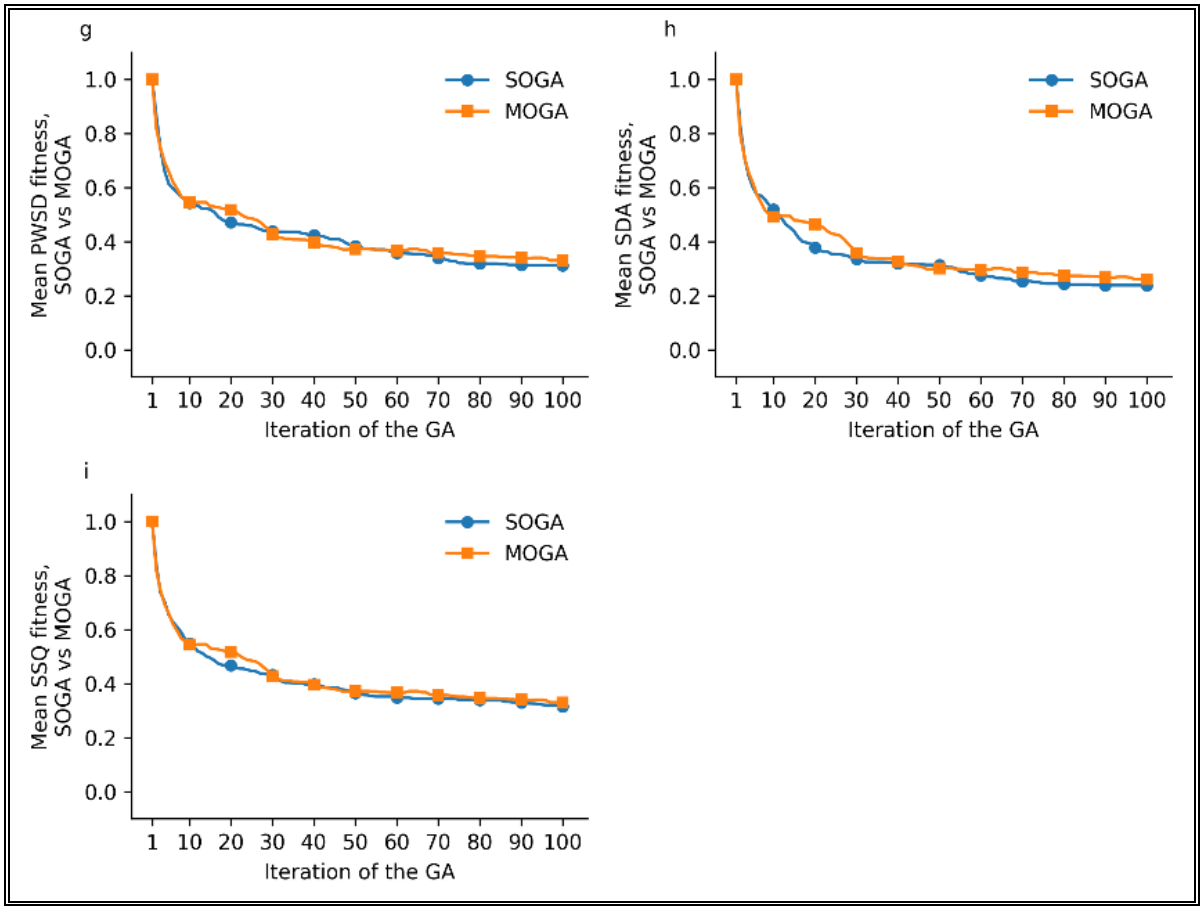


Figure S 5-5. Fractional mean error convergence.
Fractional mean error convergence for a calibration employing Strategy1 (blue circles) and Strategy4 (orange squares). Each panel shows the convergence for a single fitness function while it was selected to calibrate the Core GRN Model, and comparatively with the same error while was selected simultaneously the ANPWSD (first row, right panel), the Wellek´s test, and PWSD (the fourth row, left panel) fitness functions.

S 5-6 Best fits for the rpoS mRNA.

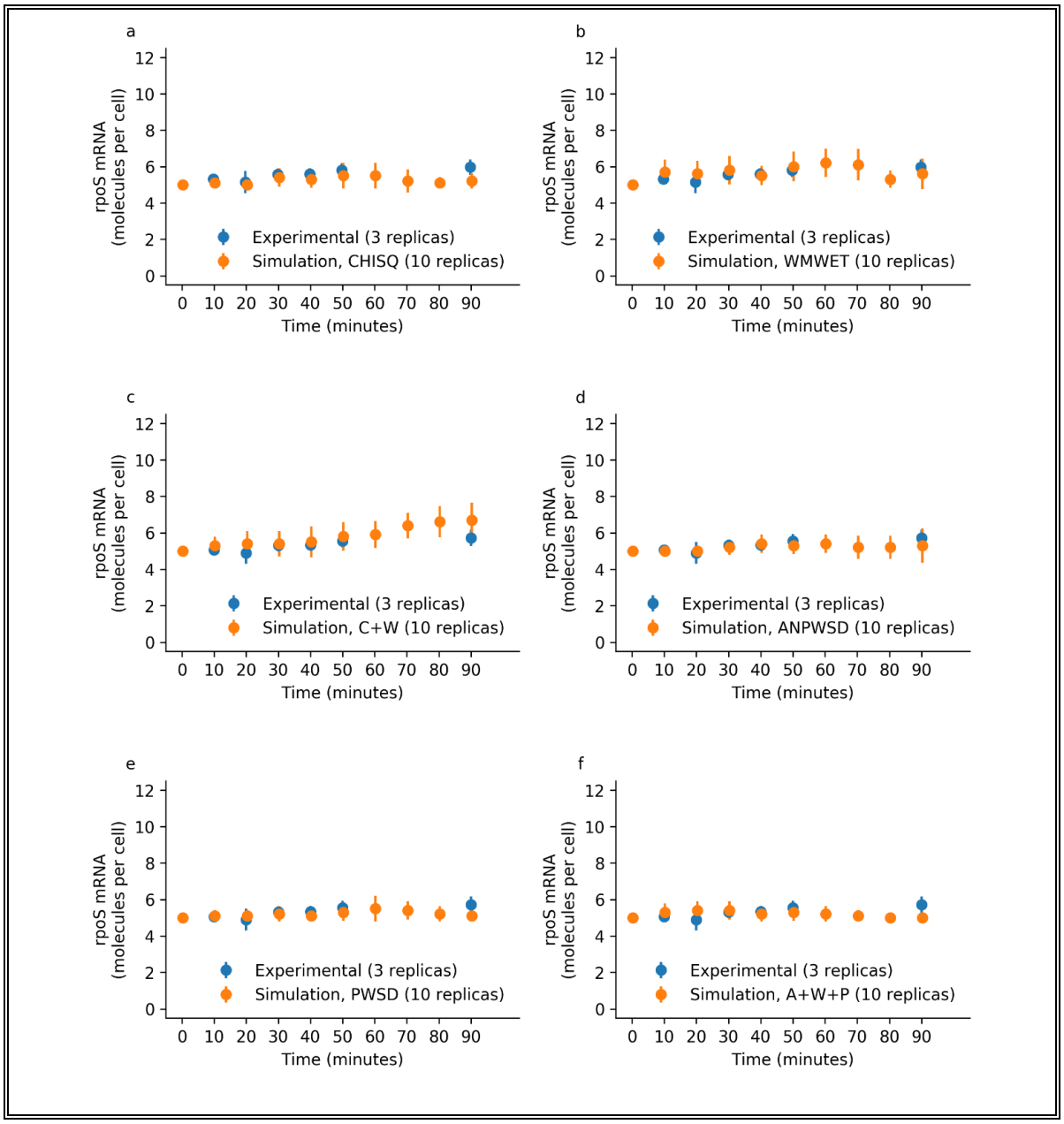


Figure S 5-6. Best fits for the rpoS mRNA.
Best fits for the rpoS mRNA employing a single (left panels) and multiple fitness functions (right panels) simultaneously. The two Multi-Objective calibrations were performed employing the chi-square and the Wellek´s test (C), and the other with the absolute pair-wise deviation, the Wellek´s test, and the normalized absolute pair-wise deviation fitness functions (F). Differences are marginal between each calibration.

S 5-7 Calibration with *Pleione* with the Strategy3.

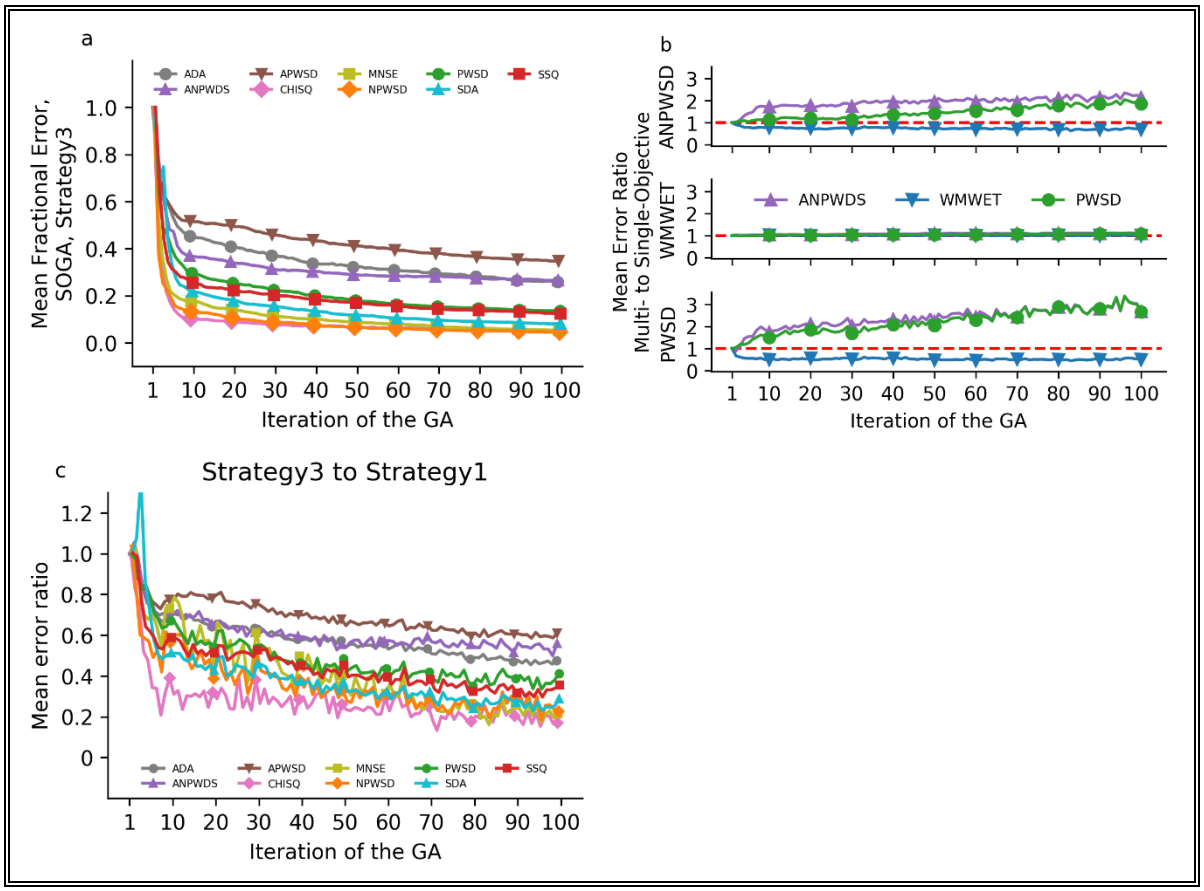


Figure S 5-7. Calibration with Pleione with the Strategy3.
The RBM representing the Core GRN of *E. coli* was calibrated against transcriptomic data and evaluated individually with all ten fitness functions. A. Error convergence in a Single-Objective Calibration. The traces correspond to the mean of all models per iteration, normalized against the maximum value achieved at the first iteration (fractional error) B. Multi-Objective Performance. The traces correspond to the ratio between a single fitness function in a Multi-Objective calibration against itself when selecting one of three fitness functions in a Single-Objective GA. Ratios below the unity mean that the fitness function in a Single-Objective calibration achieved greater errors than itself in the Multi-Objective optimization. C. Comparison to elitist GA. The figure shows the ratio between the fractional errors to compare the inverse and the elitist strategies. Ratios below the unity mean that the selection and mutation implemented in Strategy3 achieved a lower error reduction compared to the initial population than the elitist strategy to select parents.

S 5-8 Biological networks employed to develop the Core Gene Regulatory Network Model.

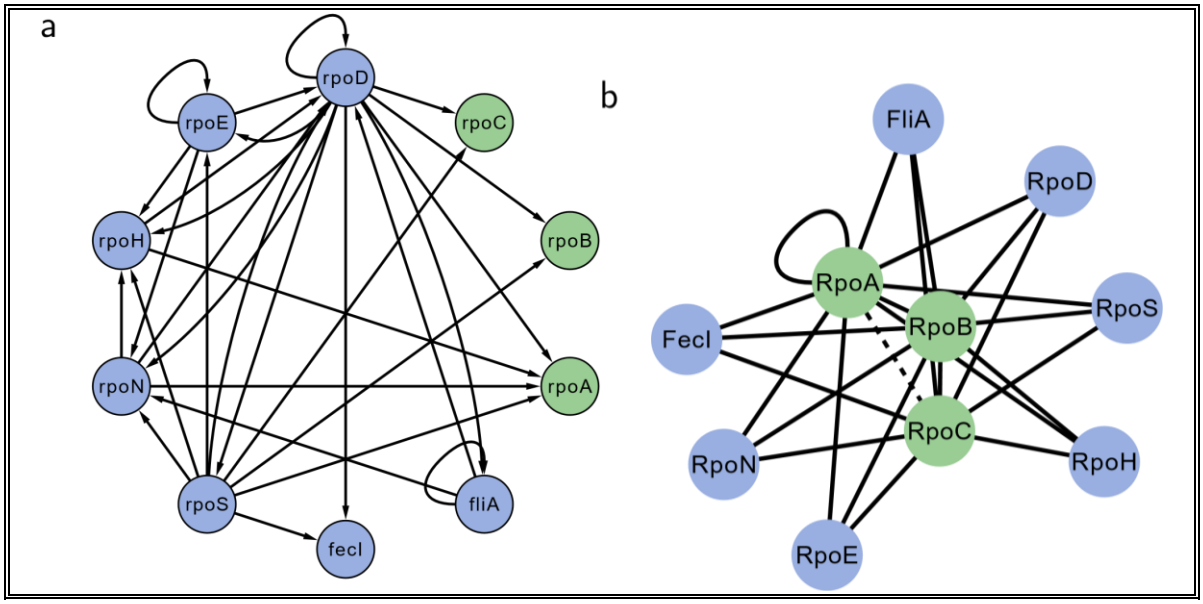


Figure S 5-8. Biological networks employed to develop the Core Gene Regulatory Network Model.

A: The Gene Regulatory Network is a directed network composed of 30 positive regulations and 10 genes, seven of which are sigma factors (light blue) and the remaining encode the α, β, and β' subunits of the RNA Polymerase Core Enzyme (RpoA, B, and C). B: The Protein-Protein Interaction Network encoding the physical interactions between all proteins. The core enzyme is a tetramer composed of two α subunits, and one of each β and β0 subunits (green nodes). The RNAP holoenzyme is completed with one of the seven sigma factors. The hard lines are physical interactions while the dashed line between RpoA and RpoC proteins denotes an indirect interaction.

S 5-9 Inference of parameter uncertainty throughout one-leave-out Jackknife for the core GRN model.

Table S 5-1. Inference of parameter uncertainty throughout one-leave-out Jackknife for the core GRN model.
”Jack” refers to the Jackknife estimator corrected after bias consideration. ”Mean” refers to the average of all subsamples, ”SE” the standard error of subsamples, and ”bias” refers to the Jackknife bias.

Parameter	Jack	Mean	SE	bias
degrade_RNA_fecIRBSRNA_fecICDS_k	-0,03359	0,075229	0,01532	0,072546
degrade_RNA_fliARBSRNA_fliACDS_k	0,074741	0,053473	0,004697	-0,01418
degrade_RNA_rpoARBSRNA_rpoACDS_k	0,131001	0,082656	0,026083	-0,03223
degrade_RNA_rpoBRBSRNA _rpoBCDSRNA_rpoCRBSRNA_rpoCCDS_k	0,071978	0,081669	0,013136	0,006461
degrade_RNA_rpoDRBSRNA_rpoDCDS_k	0,127786	0,079709	0,026003	-0,03205
degrade_RNA_rpoERBSRNA_rpoECDS_k	0,113551	0,084511	0,009664	-0,01936
degrade_RNA_rpoHRBSRNA_rpoHCDS_k	0,102653	0,092558	0,011236	-0,00673
degrade_RNA_rpoNRBSRNA_rpoNCDS_k	0,097976	0,085013	0,010174	-0,00864
degrade_RNA_rpoSRBSRNA_rpoSCDS_k	0,108318	0,06769	0,026675	-0,02709
docking_rnap24_p1rpod_fwd	1,445771	0,582415	0,288675	-0,57557
docking_rnap24_p1rpod_rvs	0,173118	0,53066	0,253974	0,238362
docking_rnap24_p1rpoe_fwd	-0,57589	0,315666	0,527432	0,59437
docking_rnap24_p1rpoe_rvs	0,696355	0,54036	0,456712	-0,104
docking_rnap24_p1rpoh_fwd	-1,06251	0,605495	0,584072	1,112003
docking_rnap24_p1rpoh_rvs	-0,75318	0,594018	0,546206	0,898129
docking_rnap24_p1rpon_fwd	-0,43668	0,602523	0,410908	0,692804
docking_rnap24_p1rpon_rvs	1,569712	0,600113	0,206707	-0,6464
docking_rnap28_p1flia_fwd	-0,36672	0,659787	0,132764	0,684337
docking_rnap28_p1flia_rvs	0,070267	0,347218	0,31347	0,184634
docking_rnap28_p1rpod_fwd	0,199553	0,895794	0,135881	0,464161
docking_rnap28_p1rpod_rvs	-0,19469	0,175885	0,175418	0,247047
docking_rnap28_p1rpon_fwd	-1,04314	0,69326	0,47154	1,1576
docking_rnap28_p1rpon_rvs	1,218488	0,560888	0,488593	-0,4384
docking_rnap32_p1rpoa_fwd	1,267096	0,363877	0,483783	-0,60215
docking_rnap32_p1rpoa_rvs	2,104009	0,329714	0,389054	-1,18286
docking_rnap32_p1rpod_fwd	0,669233	0,350031	0,411611	-0,2128
docking_rnap32_p1rpod_rvs	1,391538	0,453387	0,259699	-0,62543
docking_rnap38_p1feci_fwd	1,945819	0,277128	0,249491	-1,11246
docking_rnap38_p1feci_rvs	0,352451	0,508967	0,466181	0,104344
docking_rnap38_p1rpoa_fwd	2,079586	0,357589	0,557333	-1,148
docking_rnap38_p1rpoa_rvs	-0,27816	0,730746	0,223807	0,672603
docking_rnap38_p1rpob_fwd	0,283344	0,386651	0,35836	0,068871
docking_rnap38_p1rpob_rvs	-0,33383	0,79434	0,286838	0,752116
docking_rnap38_p1rpod_fwd	-0,02745	0,67524	0,397401	0,468459
docking_rnap38_p1rpod_rvs	-0,17494	0,268199	0,278194	0,295427
docking_rnap38_p1rpoe_fwd	-0,14534	0,659037	0,630924	0,536253
docking_rnap38_p1rpoe_rvs	0,047703	0,529356	0,173448	0,321102
docking_rnap38_p1rpoh_fwd	-0,8298	0,506609	0,235179	0,890936
docking_rnap38_p1rpoh_rvs	0,555719	0,434632	0,329177	-0,08072
docking_rnap38_p1rpon_fwd	1,56973	0,343754	0,29229	-0,81732

docking_rnap38_p1rpon_rvs	1,575125	0,458534	0,199591	-0,74439
docking_rnap54_p1rpoa_fwd	-0,50234	0,331603	0,343427	0,555961
docking_rnap54_p1rpoa_rvs	0,548096	0,501163	0,44061	-0,03129
docking_rnap54_p1rpod_fwd	1,403161	0,371097	0,334056	-0,68804
docking_rnap54_p1rpod_rvs	0,379299	0,581213	0,362477	0,134609
docking_rnap54_p1rpoh_fwd	-0,2703	0,265867	0,207013	0,357441
docking_rnap54_p1rpoh_rvs	-0,61039	0,358586	0,537333	0,645986
docking_rnap70_p1feci_fwd	-0,39592	0,368121	0,338683	0,509362
docking_rnap70_p1feci_rvs	0,883405	0,638015	0,216305	-0,16359
docking_rnap70_p1flia_fwd	1,616003	0,45535	0,254541	-0,77377
docking_rnap70_p1flia_rvs	-0,10751	0,287335	0,216224	0,263228
docking_rnap70_p1rpoa_fwd	0,632337	0,345819	0,425477	-0,19101
docking_rnap70_p1rpoa_rvs	0,597349	0,686827	0,366964	0,059652
docking_rnap70_p1rpob_fwd	0,306648	0,428711	0,26828	0,081375
docking_rnap70_p1rpob_rvs	1,211016	0,453764	0,205415	-0,50483
docking_rnap70_p1rpod_fwd	-0,92441	0,593744	0,485754	1,012101
docking_rnap70_p1rpod_rvs	1,178694	0,126788	0,03708	-0,70127
docking_rnap70_p1rpoe_fwd	-0,05674	0,142428	0,14174	0,132776
docking_rnap70_p1rpoe_rvs	-0,77975	0,613189	0,305522	0,928623
docking_rnap70_p1rpoh_fwd	-1,53598	0,77517	0,123273	1,540769
docking_rnap70_p1rpoh_rvs	-0,49271	0,583872	0,159728	0,717723
docking_rnap70_p1rpon_fwd	0,297563	0,804841	0,138248	0,338186
docking_rnap70_p1rpon_rvs	-0,85372	0,591613	0,43974	0,963555
docking_rnap70_p1rpos_fwd	0,091168	0,452659	0,444664	0,240994
docking_rnap70_p1rpos_rvs	-0,02759	0,44272	0,477901	0,313537
rpoa_rpoa_rpob_rpoc_feci_fwd	197,1614	46,75603	52,63992	-100,27
rpoa_rpoa_rpob_rpoc_feci_rvs	111,2242	40,73381	46,77473	-46,9936
rpoa_rpoa_rpob_rpoc_flia_fwd	-7,86049	56,3545	43,37604	42,80999
rpoa_rpoa_rpob_rpoc_flia_rvs	199,3186	42,13268	48,42403	-104,791
rpoa_rpoa_rpob_rpoc_rpod_fwd	-11,6538	8,633392	11,531	13,52482
rpoa_rpoa_rpob_rpoc_rpod_rvs	22,95695	47,70991	43,90544	16,50197
rpoa_rpoa_rpob_rpoc_rpoe_fwd	-103,89	53,4257	10,76507	104,8774
rpoa_rpoa_rpob_rpoc_rpoe_rvs	-47,1222	30,03322	32,35312	51,43695
rpoa_rpoa_rpob_rpoc_rpoh_fwd	-43,5907	68,99582	24,76471	75,05766
rpoa_rpoa_rpob_rpoc_rpoh_rvs	4,132941	54,25928	54,25458	33,41756
rpoa_rpoa_rpob_rpoc_rpon_fwd	189,1148	31,4646	37,85269	-105,1
rpoa_rpoa_rpob_rpoc_rpon_rvs	153,2202	58,58658	35,33477	-63,0891
rpoa_rpoa_rpob_rpoc_rpos_fwd	-25,4132	14,10009	24,37087	26,3422
rpoa_rpoa_rpob_rpoc_rpos_rvs	17,20758	31,1371	14,56806	9,286347

S 5-10 Inference of parameter uncertainty using 20 bootstraps.

Table S 5-2. Inference of parameter uncertainty using 20 bootstraps.
The procedure allows a confidence interval of 90.0%.

	Minimum	Maximum
degrade_RNA_fecIRBSRNA_fecICDS_k	0,015418	0,093511
degrade_RNA_fliARBSRNA_fliACDS_k	0,011663	0,074076
degrade_RNA_rpoARBSRNA_rpoACDS_k	0,01	0,094953
degrade_RNA_rpoBRBSRNA _rpoBCDSRNA_rpoCRBSRNA_rpoCCDS_k	0,029113	0,098912
degrade_RNA_rpoDRBSRNA_rpoDCDS_k	0,03878	0,096884
degrade_RNA_rpoERBSRNA_rpoECDS_k	0,031806	0,097227
degrade_RNA_rpoHRBSRNA_rpoHCDS_k	0,063836	0,09933
degrade_RNA_rpoNRBSRNA_rpoNCDS_k	0,02077	0,097391
degrade_RNA_rpoSRBSRNA_rpoSCDS_k	0,008518	0,091329
docking_rnap24_p1rpod_fwd	0,095833	0,942509
docking_rnap24_p1rpod_rvs	0,103773	0,993867
docking_rnap24_p1rpoe_fwd	0,01094	0,881966
docking_rnap24_p1rpoe_rvs	0,0603	0,906298
docking_rnap24_p1rpoh_fwd	0,037422	0,943149
docking_rnap24_p1rpoh_rvs	0,018008	0,82974
docking_rnap24_p1rpon_fwd	0,093583	0,916892
docking_rnap24_p1rpon_rvs	0,060983	0,847561
docking_rnap28_p1flia_fwd	0,05987	0,895893
docking_rnap28_p1flia_rvs	0,009676	0,924858
docking_rnap28_p1rpod_fwd	0,019887	0,961636
docking_rnap28_p1rpod_rvs	0,339933	0,91945
docking_rnap28_p1rpon_fwd	0,015624	0,844661
docking_rnap28_p1rpon_rvs	0,138558	0,904499
docking_rnap32_p1rpoa_fwd	0,019595	0,827524
docking_rnap32_p1rpoa_rvs	0,217859	0,956201
docking_rnap32_p1rpod_fwd	0,021209	0,835443
docking_rnap32_p1rpod_rvs	0,049484	0,920967
docking_rnap38_p1feci_fwd	0,017611	0,915298
docking_rnap38_p1feci_rvs	0,051926	0,948303
docking_rnap38_p1rpoa_fwd	0,009059	0,892295
docking_rnap38_p1rpoa_rvs	0,03737	0,961912
docking_rnap38_p1rpob_fwd	0,132154	0,956783
docking_rnap38_p1rpob_rvs	0,215169	0,993703
docking_rnap38_p1rpod_fwd	0,088594	0,976686
docking_rnap38_p1rpod_rvs	0,150762	0,981178
docking_rnap38_p1rpoe_fwd	0,050735	0,89449
docking_rnap38_p1rpoe_rvs	0,018819	0,94098
docking_rnap38_p1rpoh_fwd	0,05578	0,975849
docking_rnap38_p1rpoh_rvs	0,241838	0,966549
docking_rnap38_p1rpon_fwd	0,257689	0,968392
docking_rnap38_p1rpon_rvs	0,027071	0,892891
docking_rnap54_p1rpoa_fwd	0,018159	0,74595
docking_rnap54_p1rpoa_rvs	0,038679	0,866578
docking_rnap54_p1rpod_fwd	0,03544	0,848812
docking_rnap54_p1rpod_rvs	0,176343	0,962994

docking_rnap54_p1rpoh_fwd	0,006197	0,775382
docking_rnap54_p1rpoh_rvs	0,090734	0,974517
docking_rnap70_p1feci_fwd	0,019493	0,92872
docking_rnap70_p1feci_rvs	0,180631	0,896933
docking_rnap70_p1flia_fwd	0,130893	0,966198
docking_rnap70_p1flia_rvs	0,144216	0,987815
docking_rnap70_p1rpoa_fwd	0,016978	0,972855
docking_rnap70_p1rpoa_rvs	0,3019	0,952864
docking_rnap70_p1rpob_fwd	0,048642	0,780183
docking_rnap70_p1rpob_rvs	0,055855	0,941712
docking_rnap70_p1rpod_fwd	0,202404	0,965757
docking_rnap70_p1rpod_rvs	0,175459	0,819108
docking_rnap70_p1rpoe_fwd	0,042815	0,931646
docking_rnap70_p1rpoe_rvs	0,046631	0,858377
docking_rnap70_p1rpoh_fwd	0,014443	0,940546
docking_rnap70_p1rpoh_rvs	0,087262	0,857371
docking_rnap70_p1rpon_fwd	0,390051	0,951194
docking_rnap70_p1rpon_rvs	0,188165	0,935545
docking_rnap70_p1rpos_fwd	0,118926	0,978016
docking_rnap70_p1rpos_rvs	0,073125	0,947389
rpoa_rpoa_rpob_rpoc_feci_fwd	8,337535	95,10719
rpoa_rpoa_rpob_rpoc_feci_rvs	1,715186	71,85666
rpoa_rpoa_rpob_rpoc_flia_fwd	22,55262	87,54622
rpoa_rpoa_rpob_rpoc_flia_rvs	1,43309	99,53465
rpoa_rpoa_rpob_rpoc_rpod_fwd	0,070744	81,59405
rpoa_rpoa_rpob_rpoc_rpod_rvs	28,6883	94,72592
rpoa_rpoa_rpob_rpoc_rpoe_fwd	1,552307	88,211
rpoa_rpoa_rpob_rpoc_rpoe_rvs	0,470571	93,89551
rpoa_rpoa_rpob_rpoc_rpoh_fwd	6,008725	96,79683
rpoa_rpoa_rpob_rpoc_rpoh_rvs	1,057004	98,13897
rpoa_rpoa_rpob_rpoc_rpon_fwd	3,466332	77,08649
rpoa_rpoa_rpob_rpoc_rpon_rvs	7,007182	75,74328
rpoa_rpoa_rpob_rpoc_rpos_fwd	0,210678	78,48835
rpoa_rpoa_rpob_rpoc_rpos_rvs	29,67281	90,96462

10. ANNEX CHAPTER I: SHORT-TERM IMPACT OF SUCRALOSE CONSUMPTION ON THE METABOLIC RESPONSE AND GUT MICROBIOME OF HEALTHY ADULTS

Pamela Thomson[1,*], **Rodrigo Santibáñez**[1,*], Carolina Aguirre[2], José E. Galgani[2,3], Daniel Garrido[1]

[1]Department of Chemical and Bioprocess Engineering, School of Engineering, Pontificia Universidad Católica de Chile, Santiago, Chile

[2]Departamento Ciencias de la Salud, Carrera de Nutrición y Dietética, Facultad de Medicina, Pontificia Universidad Católica de Chile, Santiago, Chile

[3]Departamento de Nutrición, Diabetes y Metabolismo, Facultad de Medicina, Pontificia Universidad Católica de Chile, Santiago, Chile

*Both authors contributed similarly in this study

Published in *British Journal of Nutrition* (2019). Vol. 122, Issue 8, Pages 856-862

10.1 SUMMARY

Sucralose is an artificial non-nutritive sweetener used in foods aimed to reduce sugar and energy intake. While thought to be inert, the impact of sucralose on metabolic control has shown to be the opposite. The gut microbiome has emerged as a factor shaping metabolic responses after sweetener consumption. We examined the short-term effect of sucralose consumption on glucose homeostasis and the gut microbiome of healthy male volunteers. We performed a randomized, double-blind study in 34 subjects divided into two groups, one that was administered sucralose capsules (780 mg/day for seven days; n=17), and a control group receiving a placebo (n=17). Before and after the intervention, glycemic and insulinemic responses were assessed with a standard oral glucose load (75 g). Insulin resistance was determined using HOMA-IR and Matsuda indexes. The gut microbiome was evaluated before and after the intervention by 16S rRNA sequencing. During the study, body weight remained constant in both groups. Glycemic control and insulin resistance were not affected during the 7-day period. At the phylum level, the gut microbiome was not modified

in any group. We classified subjects according to their change in insulinemia after the intervention, to compare the microbiome of responders and non-responders. Independent of consuming sucralose or placebo, individuals with a higher insulinemic response after the intervention had lower Bacteroidetes and higher Firmicutes abundances. In conclusion, the consumption of high doses of sucralose for 7 days does not alter glycemic control, insulin resistance, or gut microbiome in healthy individuals. However, it highlights the need to address individual responses to sucralose.

10.2 INTRODUCTION

Sucralose (1,6-dichloro 1,6-dideoxy β-D-fructofuranosyl 4-chloro 4-deoxy α-D-galactopyranoside) is a non-caloric artificial sweetener (NAS) synthesized by the selective halogenation of sucrose (Magnuson et al., 2016). Approved by the FDA for use in humans, it is 600 times sweeter than sucrose. Due to its low production cost, high thermostability, and solubility, sucralose has emerged as an important sugar substitute in foods and drinks. The acceptable daily intake (ADI) of sucralose has been established at 15 mg/kg body weight (Martyn et al., 2018).

The concept that replacing sucrose with NAS in foods and drinks improves metabolic control has been challenged (Shearer & Swithers, 2016). In mice, sucralose added to drinking water for 11 weeks impaired oral glucose tolerance when compared with water alone or water with sucrose or glucose (Suez et al., 2014). Such deleterious effect was prevented when mice were treated with broad-spectrum antibiotics against Gram-Negative or Gram-Positive bacteria. The fact that sucralose displays bacteriostatic action on several gut microbes (Omran, Ahearn, et al., 2013; Q. P. Wang et al., 2018), and that most of the sucralose is not absorbed in the intestine (Magnuson et al., 2016; Roberts et al., 2000; Sylvetsky et al., 2017), gives support to observations showing that sucralose can alter gut microbiome composition (Rodriguez-Palacios et al., 2018; Q. P. Wang et al., 2018). Taken together, the notion that sucralose influences glucose control through alterations in intestinal microbiota has emerged.

In humans, the consumption of high doses of sucralose for three months has been assessed in nondiabetic (Grotz et al., 2017) and type-2 diabetic (Grotz et al., 2003)

individuals. Those studies showed no influence of sucralose on glycemic control when compared with placebo. However, between-subject variability in given markers of glycemic control appears higher after sucralose vs. placebo, particularly in non-diabetic individuals. Eventually, the glycemic response to sucralose in humans, as in mice, is also mediated through changes in gut microbiota (Lobach et al., 2019).

Considering the relevance of chronic diseases and the wide availability of sucralose in foods and drinks, it is critical to determine the effect of sucralose on metabolic responses and the gut microbiome. The goal of this study was to evaluate the short-term effect of sucralose on glycemic control and its interaction with the microbiota in healthy subjects.

10.3 EXPERIMENTAL METHODS

10.3.1 Subjects

This study was conducted in accordance with the Declaration of Helsinki and approved by the Ethics Committee of the Faculty of Medicine, Pontificia Universidad Católica de Chile. All participants provided written informed consent. Thirty-four healthy men between 18 and 50 years with stable weight (variation < 2 kg in the last three months) and body mass index (BMI) between 20–30 kg/m^2 were recruited. None of them carried out intense physical activity regularly or received any drug treatment during the last three months. The fulfillment of all the inclusion criteria was evaluated in an initial screening, and individuals not meeting the requirements were excluded. Only males were included in order to avoid potential menstrual cycle-related changes in insulin sensitivity.

10.3.2 Study design

A parallel, double-blind, placebo-controlled study was performed. Selected participants were requested to fast overnight and instructed to avoid intense physical activity the day before the evaluation, in addition to smoking, alcohol, and energy drinks consumption 12 h previous to the evaluation. On the evaluation day, subjects were requested to bring or collect in the clinical facility a fecal sample in a 15 ml anaerobic container. The fecal sample was immediately stored at –80 °C for gut microbiome analysis. Body weight was measured followed by the insertion of an intravascular cannula in a peripheral vein of the arm. After

resting for 30 min, two 10-min apart blood samples were taken followed by the administration of an oral glucose load (75 g in 290 ml solution). Blood samples were obtained after 30, 60, 90, and 120 min of glucose ingestion. Once finished this procedure, volunteers were randomly assigned to an intervention group with sucralose (n = 17) or placebo (n = 17). For this, we used an online resource (https://www.randomizer.org/). Randomization was conducted by one of the members of our group. This person was not involved in recruiting, selecting, or performing any of the measurements of this study, and broke the labels after data analysis. Four subjects did not finish the study, resulting in 16 subjects in the sucralose group and 14 in the placebo group. Individuals were instructed to ingest one capsule containing sucralose or placebo, three times a day for 7 days. Sucralose was purchased from VitaSweet (Beijing, China). Each sucralose capsule contained 260 mg sucralose and 70 mg calcium carbonate; thus the three capsules consumed daily were equivalent to the 75% of the Acceptable Daily Intake (ADI) (Martyn et al., 2018) (15 mg/kg/d) of a subject weighing 70 kg. Each placebo capsule contained 250 mg of calcium carbonate. Each subject was instructed to register and report any adverse events. After 7 days, subjects were requested to attend the clinical facility to repeat the same procedures. Volunteers were instructed to record any adverse events or changes in their customary diet.

10.3.3 Glucose and insulin tests

Plasma concentrations of glucose were determined by the glucose oxidase method, in a dry chemistry equipment (Vitros 4600, Ortho Clinical Diagnostics, Raritan, NJ). Serum concentrations of insulin were quantified by a capture chemiluminescent immunoassay (Centaur XPT, Siemens, Henkestr, Germany). Insulin resistance was calculated as the HOMA-IR index (Homeostasis Model Assessment of Insulin Resistance), using the formula by Matthews et al. (Matthews et al., 1985): (fasting insulin [µU/ml] × fasting glucose [mmol/l])/22.5. Alternatively, the ISI-Composite index was also determined, using the formula proposed by Matsuda and De Fronzo (Matsuda & DeFronzo, 1999): 10,000/(fasting glycemia [mg/dl] × fasting insulin [µU/ml] × average glycemia in the oral test (30 – 120 min) [mg/dl] × average insulin in the oral test (30 – 120 min) [µU/ml])$^{0.5}$. Total Area Under

the Curve (AUC) for insulinemic and glycemic responses were also calculated, using the trapezoidal rule (Krzyzanski & Jusko, 1998).

10.3.4 Statistical analysis

Clinical data were expressed as the average ± SD. Statistical analyses were run on SAS v9.2 (SAS Institute. Cary. NC). Subject´s characteristics at screening were compared between groups using the Student´s *t*-test. Differences in the AUC for the glycemic and insulinemic responses were assessed through the Wilcoxon´s rank sums test and the Mann-Whitney U-test. Differences in clinical variables before and after the intervention were determined using a repeated-measure ANOVA including group, time (repeated), and the group × time interaction. The significance level for all analyses was set at a p-value < 0.05.

10.3.5 Sample size

The sample size needed to detect a difference in glycemia (i.e., mean of glycemia between 30–120 min after OGTT) after one-week sucralose ingestion was estimated from two 4-weeks apart OGTTs performed in healthy, young, non-obese individuals (Galgani et al., 2013). Considering a within-subject SD of ±11 mg/dL as the expected variability without intervention, and a between-subject SD of 21 mg/dL, 15 subjects will allow detecting a difference of 13 mg/dL in glycemia after one-week sucralose ingestion, with a power of 82% for paired samples two-sided and a significance level of 5%.

10.3.6 Gut microbiome analysis

Fecal samples stored at –80 ºC were first thawed and 150 mg were used for total DNA extraction (Quick-DNA Fecal/Soil Microbe Miniprep Kit; Zymo Research, Irvine, CA) by using a Disruptor Genie device (Scientific Industries, USA) (Medina et al., 2017). DNA samples were quantified (NanoDrop 2000c; Thermo Fisher Scientific USA) and diluted to 20 ng/µl in nuclease-free water (IDT; Coralville, IA). DNA samples were submitted for Illumina MiSeq sequencing at Integrated Microbiome Resource (Halifax, NS). The 16S rRNA gene V3-V4 variable region was amplified using V3-V4 primers (Comeau et al., 2017), which also included a barcode in the forward primer. The DADA2 v1.10 R package was used to analyze the 16S rRNA gene sequences, following a modified procedure

(Callahan et al., 2016). Briefly, the sequences were filtered by the quality and trimmed to remove the barcode and low-quality nucleotides prior to estimating the sequencing error. Then, sequences were denoised to identify Amplicon Sequence Variants (Callahan et al., 2017), then merged and used to assign microbial taxonomy following a Naïve Bayesian classifier (Q. Wang et al., 2007), employing the SILVA database version 132 (Quast et al., 2013; Yilmaz et al., 2014). Raw data paired-end reads obtained from the MiSeq platform were stored in the European Nucleotide Archive database (ENA), under study accession number PRJEB27704.

10.3.7 Bioinformatics analysis

Microbiome composition similarity at the phylum level was determined employing Principal Coordinates Analysis with distances calculated with the weighted UniFrac method (C. Lozupone et al., 2011; C. A. Lozupone et al., 2007). The weighted UniFrac quantitatively estimates the beta-diversity taking into account phylogenetic distances between microbial taxa and their relative abundance. The similarity between microbiome compositions was calculated with the Unweighted Pair Group Method with Arithmetic Mean (UPGMA) Algorithm. Fold changes in the relative microbiome composition at the phylum level (values after/before intervention), for each group of subjects and subgroups were determined using the Mann-Whitney U-test. The significance level for all analyses was set at $p < 0.05$.

10.4 RESULTS

10.4.1 Subjects characteristics at the screening

Age, weight, and height were similar between groups (Table 10-1), while BMI was higher in the placebo compared with the sucralose group (*p*-value = 0.04). Such difference in BMI was accompanied by higher blood cholesterol concentration (*p*-value < 0.01), but similar blood glucose concentration (Table 10-1).

Table 10-1. Clinical parameters at screening (mean ± SD [range])

	Placebo (*n* = 14)	**Sucralose (*n* = 16)**	***t*-test**
Age (years)	23.5 ± 2.9 [18.2 – 29.3]	22.8 ± 3.0 [18.7 – 30.2]	0.51
Weight (kg)	77.0 ± 8.3 [57.9 – 88.0]	73.2 ± 6.9 [60.9 – 83.7]	0.19
Height (m)	1.73 ± 0.04 [1.67 – 1.80]	1.75 ± 0.07 [1.63 – 1.90]	0.31
Body mass index (kg/m^2)	25.7 ± 2.9 [20.8 – 28.9]	23.8 ± 1.7 [21.1 – 26.6]	0.04
Glycemia (mg/dl)	84 ± 8 [67 – 95]	85 ± 6 [72 – 94]	0.50
Cholesterol (mg/dl)	173 ± 23 [134 – 219]	147 ± 22 [106 – 184]	<0.01

10.4.2 Metabolic responses to sucralose

Body weight remained stable throughout the study, with an average change of 0.16 ± 0.74 kg (*p*-value = 0.44) and 0.21 ± 1.17 kg (*p*-value = 0.49) in the placebo and sucralose groups, respectively. Volunteers in the placebo and sucralose groups did not report adverse events or changes in their usual dietary patterns. As observed at screening, fasting plasma glucose concentration was similar between groups, and not affected by the consumption of placebo or sucralose (Table 10-2). In turn, a borderline higher fasting serum insulin concentration was observed in the placebo vs. sucralose group (*p*-value = 0.07; Table 10-2). Similar to fasting plasma glucose, consumption of sucralose or placebo did not affect fasting serum insulin concentration (Table 10-2).

Table 10-2. Metabolic response to intervention (mean ± SD).

	Placebo (n = 14)			Sucralose (n = 16)			p-value		
	Before	After	Change*	Before	After	Change*	Group	Time	Group×Time
Fasting									
Glycemia (mg/dl)	82 ± 5	79 ± 4	–2.2 ± 5.0	82 ± 5	82 ± 5	0.0 ± 6.1	0.22	0.31	0.29
Insulinemia (µU/ml)	12 ± 5	11 ± 4	–1.0 ± 3.3	9 ± 4	8 ± 4	–0.9 ± 4.6	0.07	0.22	0.93
HOMA-IR	2.4 ± 1.1	2.1 ± 0.8	–0.3 ± 0.7	1.9 ± 0.9	1.7 ± 0.9	–0.2 ± 1.1	0.13	0.21	0.77
After oral glucose									
Glycemia (mg/dl)**	115 ± 17	112 ± 21	–3.0 ±17.4	107 ± 21	113 ± 21	6.2 ± 18.6	0.65	0.63	0.17
Insulinemia (µU/ml)**	81 ± 38	87 ± 51	5.6 ± 38.1	63 ± 40	78 ± 41	15.4 ±30.5	0.35	0.11	0.44
ISI-Composite	4.1 ± 2.1	4.7 ± 3.0	0.5 ± 2.4	7.8 ± 10	5.9 ± 4.2	–1.9 ± 7.7	0.21	0.52	0.29

*Calculated as values after minus before intervention.
**Mean of the respective response over the 30 to 120 min period after glucose ingestion.

Upon glucose ingestion, the average glycemic and insulinemic responses were similar between groups and not affected by the consumption of sucralose or placebo (Table 10-2). Regarding insulin resistance (sensitivity) markers, no differences by group and intervention were detected (Table 10-2). One further assessment included the analysis of the changes in the aforementioned variables after both interventions. Consistent with ANOVA, none of the changes were different from zero (Table 10-2). We calculated the area under the curve for glycemic and insulinemic responses for each subject (Figure 10-1). This analysis did not show alterations in glycemia or insulinemia in any group.

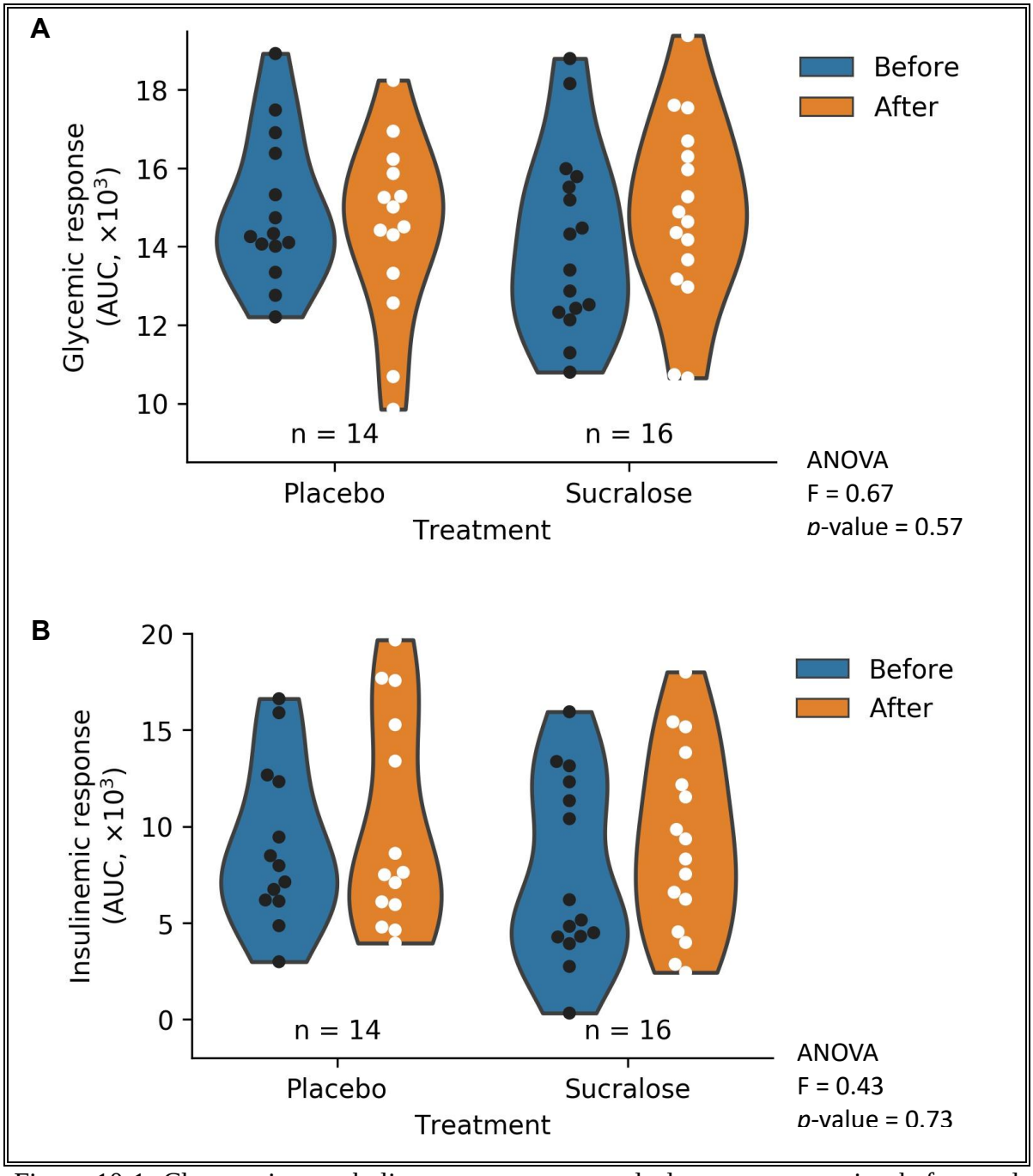


Figure 10-1. Changes in metabolic responses upon oral glucose consumption before and after the intervention.

A: Total Glycemic Area Under the Curve (AUC) for each group; B: insulinemic total AUC. Subjects with insufficient data to calculate the AUC were not included. The kernel density estimation shows the probability of the values.

10.4.3 Changes in gut microbiome composition

The gut microbiome of all subjects in each group, both before and after the 7-day intervention period, was analyzed. As expected, the Firmicutes and Bacteroidetes phyla were dominant in the microbiome of these subjects (Figure S 10-1). Smaller representations of Actinobacteria, Verrucomicrobia, and Proteobacteria were also observed. On average, a higher relative abundance of phylum Firmicutes in the placebo vs. sucralose group was observed before initiating the treatments (Figure 10-2). However, microbiome composition remained stable throughout both interventions (Figure 10-2).

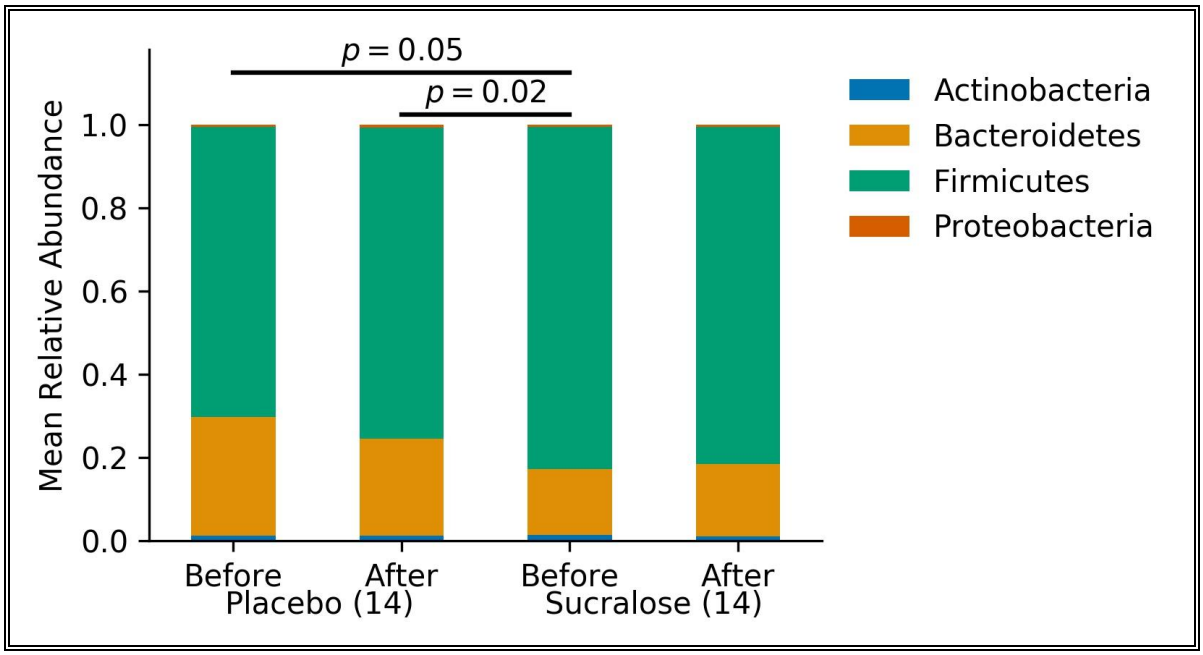


Figure 10-2. Gut microbiome compositions for each group before and after each treatment. Bars show the average relative abundance of the four dominant phyla of the human gut microbiome.

We performed a principal component analysis (PCA) to identify variations in the gut microbiome composition for each individual before and after interventions (Figure 10-3A). This PCA represents the normalized absolute abundance for all 14 phyla identified in every microbiome sample evaluated. Consistent with the aforementioned analysis, most subjects displayed modest variations in their microbiome composition (Figure 10-3A).

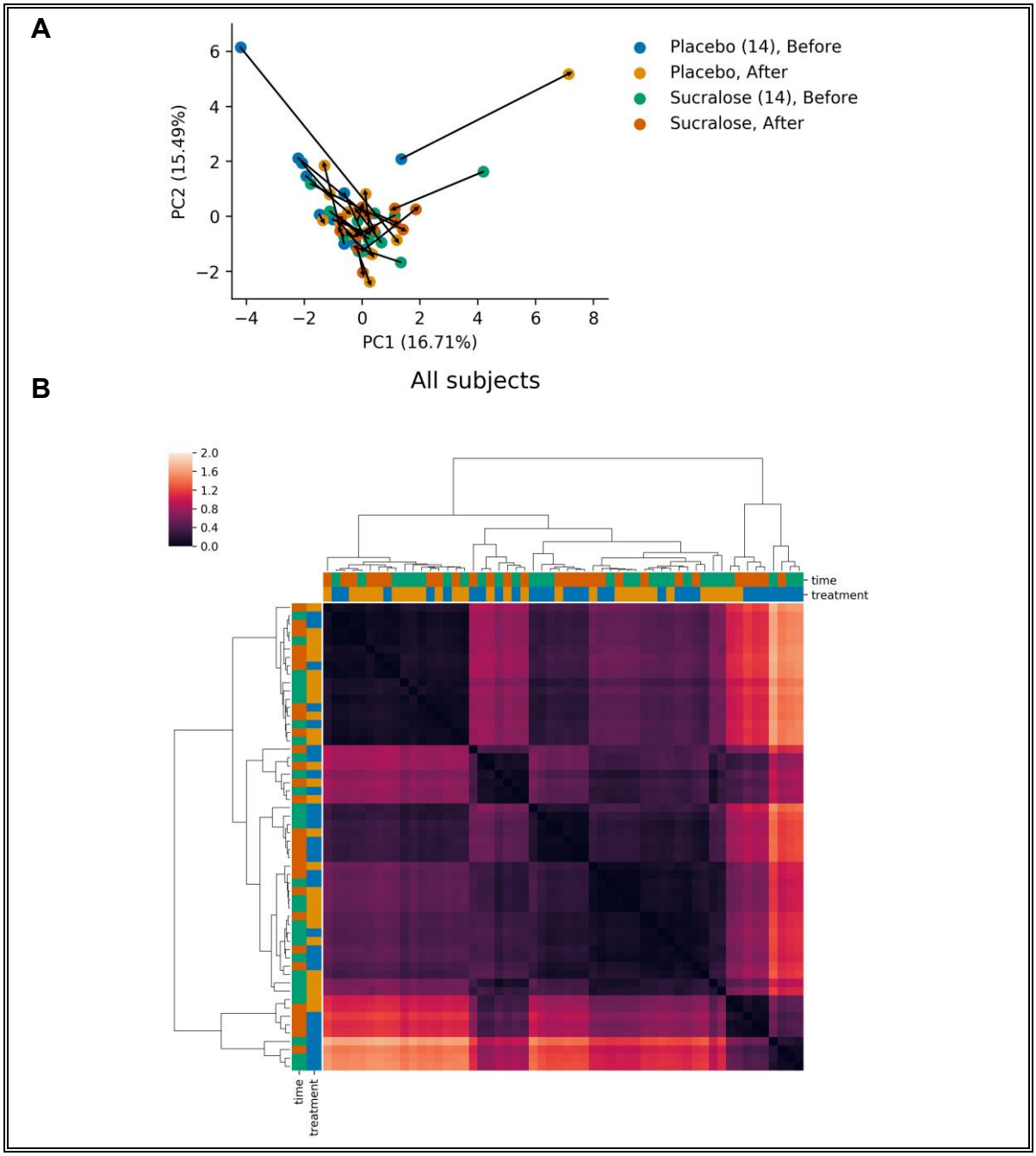


Figure 10-3. Comparisons of gut microbiome composition between subjects.
A: Principal Coordinates Analysis of all phyla identified showing close similarity for groups including both Placebo and Sucralose at both intervention times. Arrows show the trajectories of changes in microbiome composition for each individual in the study. B: Heatmap of distances calculated as the weighted UniFrac and clustering of closest distances using the UPMGA Algorithm.

We also calculated the weighted UniFrac distances for each gut microbiome (Figure 10-3B), in order to determine biological diversity among the microbiomes. This analysis showed that microbiomes in the sucralose group before and after the intervention (S1/S2) tend to cluster together, as well as microbiomes from subjects in the placebo group before and after the intervention (P1/P2). Again, this analysis indicates that both treatments did not substantially modify the microbiome of these subjects, while differences detected before interventions remained.

10.4.4 Correlations between the gut microbiome and metabolic markers

Next, we classified individuals according to their metabolic responses, by calculating the after-to-before ratio for serum insulin and glucose AUCs, BMI, and insulin resistance (sensitivity) markers. This was aimed to determine if within the same treatment certain subjects responded differently, identifying "*responders*" (subjects with a ratio higher than 1), and "*non-responders*" (a ratio smaller or equal to 1) (Suez et al., 2014).

Independent of the treatment received, *responders* vs. *non-responders* according to insulin AUC had higher Firmicutes and lower Bacteroidetes abundances in their microbiomes (Figure 10-4A, Figure 10-4B). Such differences were also noted when classifying subjects according to their change in glucose AUC, albeit with borderline significance (Figure S 10-2). Bacteroidetes relative abundance was also lower in *responders* according to changes in HOMA (Figure S 10-3). Finally, classifying individuals by their BMI (overweight or normal weight) did not show significant changes in their gut microbiomes regardless of the treatment received. Only for individuals in the placebo group, we observed that overweight individuals had higher Firmicutes and Actinobacteria compared to those with normal weight (Figure S 10-4).

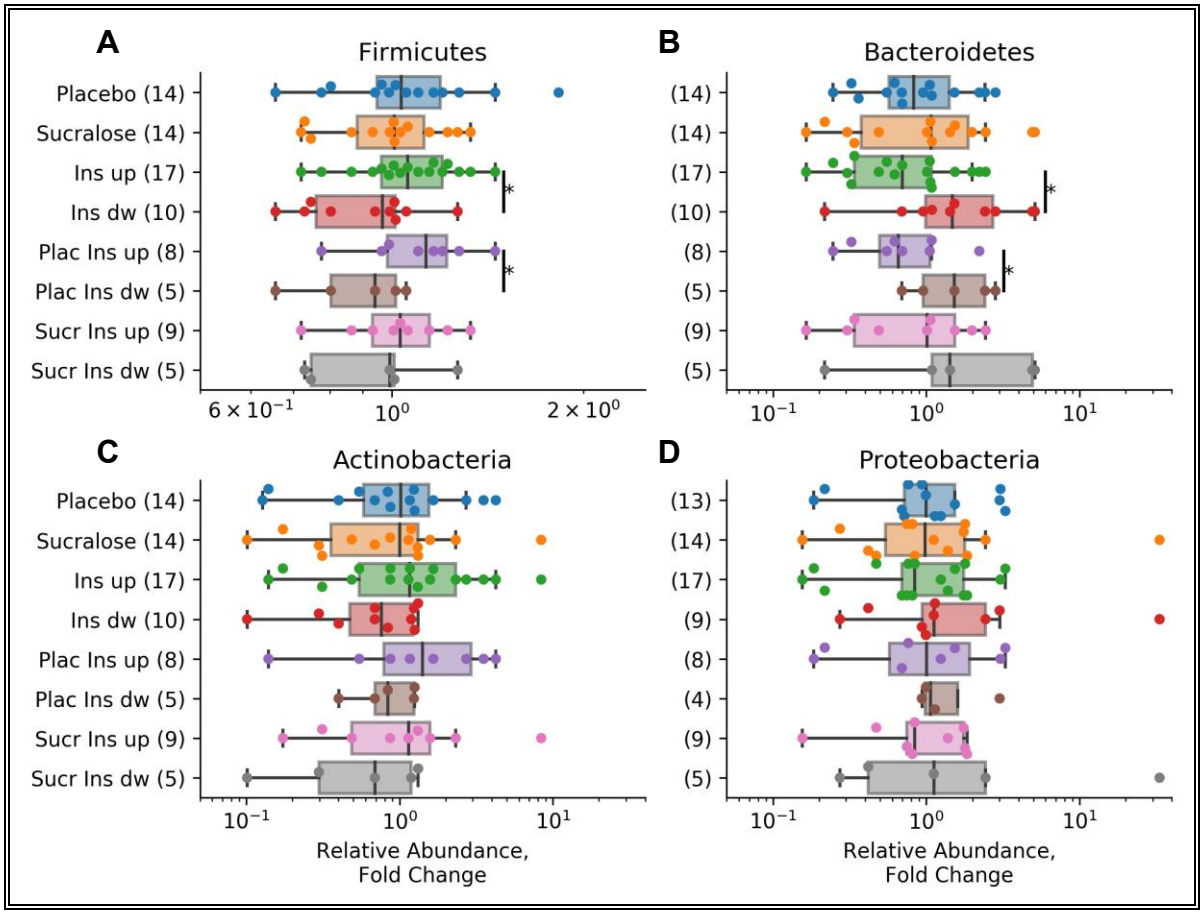


Figure 10-4. Pairwise correlations between fold changes in microbiome phyla and insulinemia responder status.
The figure shows the fold change in all major four phyla before and after the intervention, correlated with subgroups in the x-axis. Insulinemia AUC ratios were calculated with AUC values after/before the intervention. A: Fold changes in Firmicutes; B: Bacteroidetes; C: Actinobacteria; D: Proteobacteria. Boxplots indicate the median and the interquartile range, with whiskers determined as the 1.5 range of the box. An asterisk denotes significant differences (p-value < 0.05) determined with the Mann-Whitney U-test.

10.5 DISCUSSION

10.5.1 Evidence of metabolic impairments for NAS

Sucralose is one of the most consumed NAS in the world (Schiffman & Rother, 2013), with several foods and beverages being supplemented with this sweetener. While recognized as safe by several studies, recent evidence has shown that sucralose among other NAS may

promote weight gain and metabolic disturbances such as glucose intolerance (Swithers, 2013). Therefore, it is critical to understand the actual impact of NAS in our metabolism.

Previous findings in mice showed no adverse effects of sucralose on inflammatory markers or fasting glucose levels, even at large doses for up to 2 years (Goldsmith, 2000; Grice & Goldsmith, 2000). A study in diabetic subjects administrating sucralose at a dose of 667 mg (~7.5 mg/kg/day) for 13 weeks did not observe changes in glycated hemoglobin, fasting glycemia, or fasting C-peptide against placebo (Grotz et al., 2003), which are similar observations compared to this work. Another study conducted in non-diabetic individuals who consumed 1000 mg/day of sucralose (~13.2 mg/kg/day) for 12 weeks did not detect differences relative to placebo in fasting glycemia, insulinemia or glycated hemoglobin (Grotz et al., 2017). Sucralose consumption at the population level may reach up to 15% ADI, which is evidently lower compared with the amount provided in these previous reports (Magnuson et al., 2017). None of these studies evaluated the gut microbiome composition of these subjects. Here we performed a short-term study with a small sample size, finding that sucralose consumption at high doses does not alter the glycemic response of healthy individuals.

Recently the gut microbiome has emerged as a factor that could contribute to the biological effects of NAS, in particular sucralose (Suez et al., 2015). Early environmental studies revealed a strong bacteriostatic effect of several NAS, including sucralose and saccharin (Omran, Ahearn, et al., 2013; Omran, Baker, et al., 2013; Suez et al., 2015; Q. P. Wang et al., 2018). Wang *et al.* also showed increases in Firmicutes in mice exposed to sucralose (Q. P. Wang et al., 2018). Thus, alterations in the gut microbiome have been shown to be associated with NAS exposure, and these alterations could be causative of metabolic impairments such as glucose intolerance and insulin resistance (Suez et al., 2014). In C57/BL6 mice sucralose altered microbiome composition, increased pro-inflammatory fecal metabolites, and induced hepatic pro-inflammatory markers (Bian et al., 2017). Suez *et al.* (Suez et al., 2014) reported that in addition to saccharin, sucralose induced glucose intolerance in mice, and importantly this metabolic effect is transmittable to germ-free mice through their gut microbiome, indicating causality (Suez et al., 2014).

The Suez´s study was a pioneer in showing that after short-term saccharin exposure, individuals display contrasting responses in terms of glucose tolerance and microbiome composition. Some of the observations raised have however been questioned by other studies (Lobach et al., 2019). Interestingly, that contrasting glucose tolerance was emulated in germ-free mice transplanted with the microbiome of the human donors. That study also provided insights into what microorganisms are being altered in the complex gut microbiome. Saccharin consumption in the short-term resulted in enrichment in Bacteroides phyla, especially *B. vulgatus* and *B. fragilis* (Suez et al., 2014). Interestingly this exposure in mice also resulted in decreased abundance of *Akkermansia muciniphila*, a key gut microbe that has been associated with anti-inflammatory properties (Everard et al., 2013; Zhao et al., 2017). While short-term saccharin consumption resulted in an increase in Bacteroidales and decrease in Clostridiales in *responders* (Suez et al., 2014), here we observed that sucralose administration at 75% of the maximum ADI for a similar period of time resulted in no major changes in the gut microbiome composition in healthy subjects. Evidently, metabolic and microbiome responses to saccharin could be very different compared to those of sucralose, considering their chemical properties and their differences in absorption and metabolism. The absence of effect of sucralose consumption on gut microbiome composition could be explained for powering the study on glycemia and not on gut microbiome data.

We observed that at the beginning of the study, subjects in the placebo group had a different gut microbiome (higher Firmicutes and lower Bacteroidetes) compared to the sucralose group. Interestingly this correlated with these subjects having higher initial BMI and cholesterol compared to the sucralose group. These differences were not intentional and could be expected from the randomization of a population of healthy subjects. Moreover, subjects in the placebo group with higher insulinemia, HOMA, or BMI presented significant changes in their gut microbiomes (Figure 10-4, Figure S 10-3, and Figure S 10-4). These changes were a higher abundance of the Firmicutes phylum and lower Bacteroidetes. These observations indicate that the placebo group had substantial differences regarding glycemic response and gut microbiome compared to the sucralose group, and even inside the placebo group there were interesting differences between these individuals. In spite of these baseline

differences, neither group presented changes in glycemic response or gut microbiome associated with the treatment received.

In addition, we did not observe differences in the gut microbiome between *responders* and *non-responders* associated with sucralose or placebo consumption, using criteria changes in serum insulin AUC. However, subjects who had a higher insulin AUC after the intervention, and regardless of the treatment received, had a higher Firmicutes/Bacteroidetes ratio (Figure 10-4). This indicates that this metabolic difference could be more relevant than the intervention itself. This also correlates with the idea of this cohort being so healthy that only minimal or very sensitive changes in the gut microbiome were obtained. Changes in these phyla are probably relevant and implicate a considerable rearrangement of the community, considering they represent more than 90% of the total gut microbiome. This same phenotype (higher Firmicutes and lower Bacteroidetes, or a higher Firmicutes/Bacteroidetes ratio) has been observed in several studies reporting alterations in the gut microbiome in obese subjects and especially in type 2 diabetes (Kasai et al., 2015; Most et al., 2017; Sircana et al., 2018; Turnbaugh et al., 2006). Mechanistically, these microbiomes have been associated with low-grade inflammation, higher gut permeability, and higher circulating lipopolysaccharide (Boulangé et al., 2016).

10.6 CONCLUSION

This study shows that the consumption of high doses of sucralose for 7 days in healthy subjects does not alter glycemic control. There were no changes in the gut microbiomes of these subjects with respect to the consumption of sucralose or placebo. Independent of the intervention, subjects displaying an increase vs. decrease in insulinemia after either intervention had different gut microbiome compositions. Thus, initial metabolic differences could have been more important than the intervention itself in terms of altering the gut microbiome. Further studies should study the impact of other important non-caloric sweeteners, but including potential responder-non responder differences among subjects.

10.7 SUPPLEMENTARY FIGURES

S 10-1 Relative abundance of identified phyla in each sample.

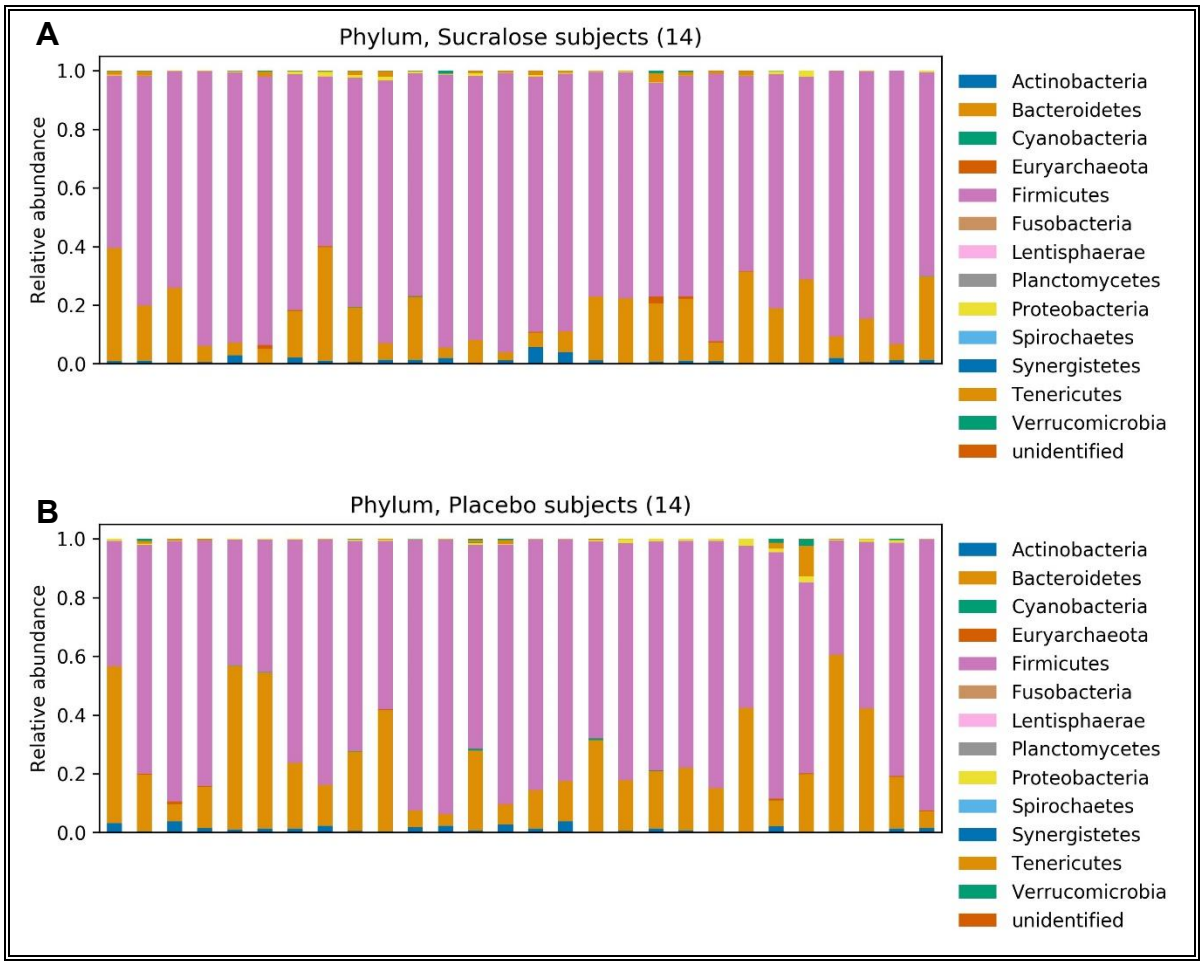


Figure S 10-1. Relative abundance of identified phyla in each sample.
A: The 28 taxonomic assignment and determined abundance (before and after intervention) for the subjects in the sucralose treatment. B: Taxonomic assignment and abundance for the subjects in the placebo treatment.

S 10-2 Pairwise correlations between fold changes in microbiome phyla and glycemia responder status.

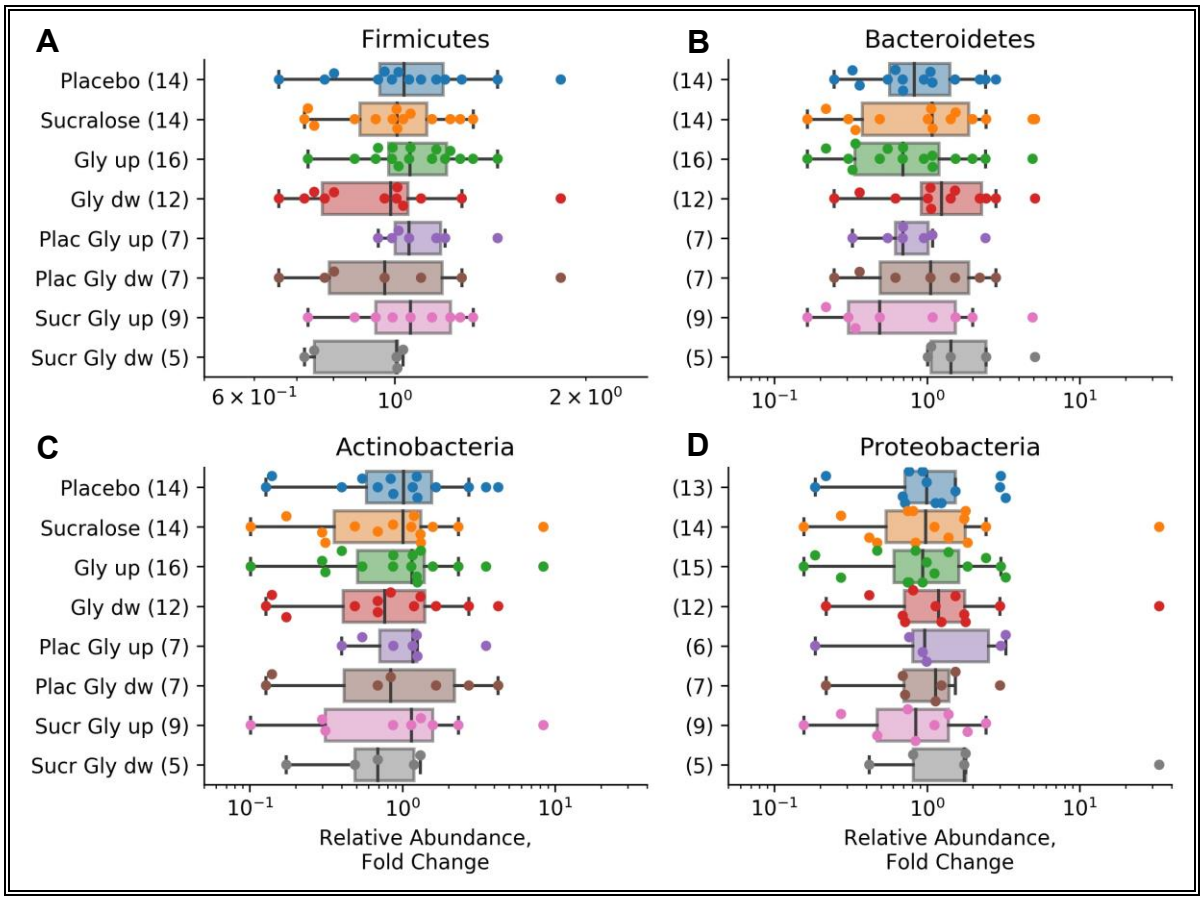


Figure S 10-2. Pairwise correlations between fold changes in microbiome phyla and glycemia responder status.

The figure shows the fold change in all major four phyla before and after the intervention, correlated with subgroups in the x-axis. Glycemia AUC ratios were calculated with AUC values after/before the intervention. A: Fold changes in Firmicutes; B: Bacteroidetes; C: Actinobacteria; D: Proteobacteria. Boxplots indicate the median and the interquartile range, with whiskers determined as the 1.5 range of the box. The asterisk denotes significant differences (p-value < 0.05) determined with the Mann-Whitney U-test.

S 10-3 Pairwise correlations between fold changes in microbiome phyla and HOMA responder status.

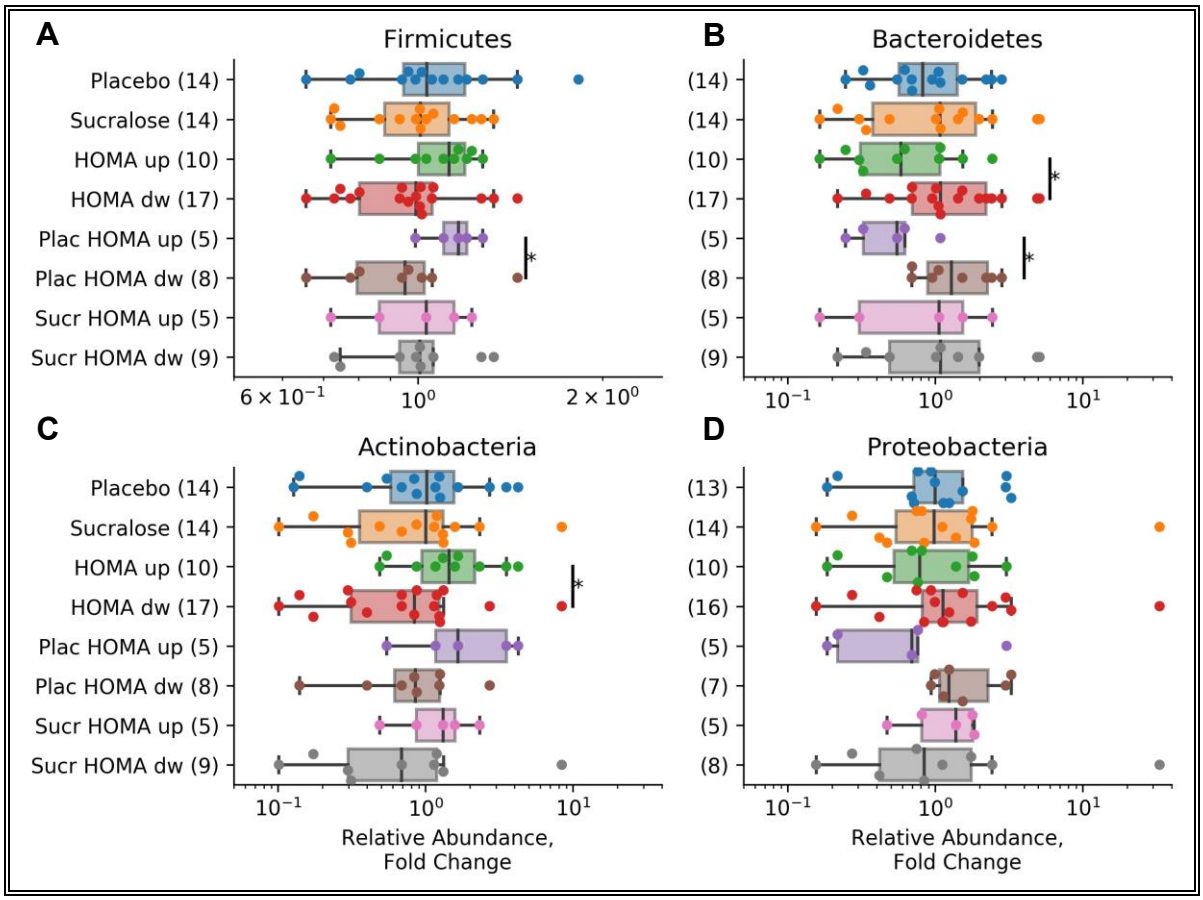


Figure S 10-3. Pairwise correlations between fold changes in microbiome phyla and HOMA responder status.

HOMA ratios were calculated with AUC values after/before the intervention. A: Fold changes in Firmicutes; B: Bacteroidetes; C: Actinobacteria; D: Proteobacteria. Boxplots indicate the median and the interquartile range, with whiskers determined as the 1.5 range of the box. An asterisk denotes significant differences (p-value < 0.05) determined with the Mann-Whitney U-test.

S 10-4 Pairwise correlations between fold changes in microbiome phyla and BMI.

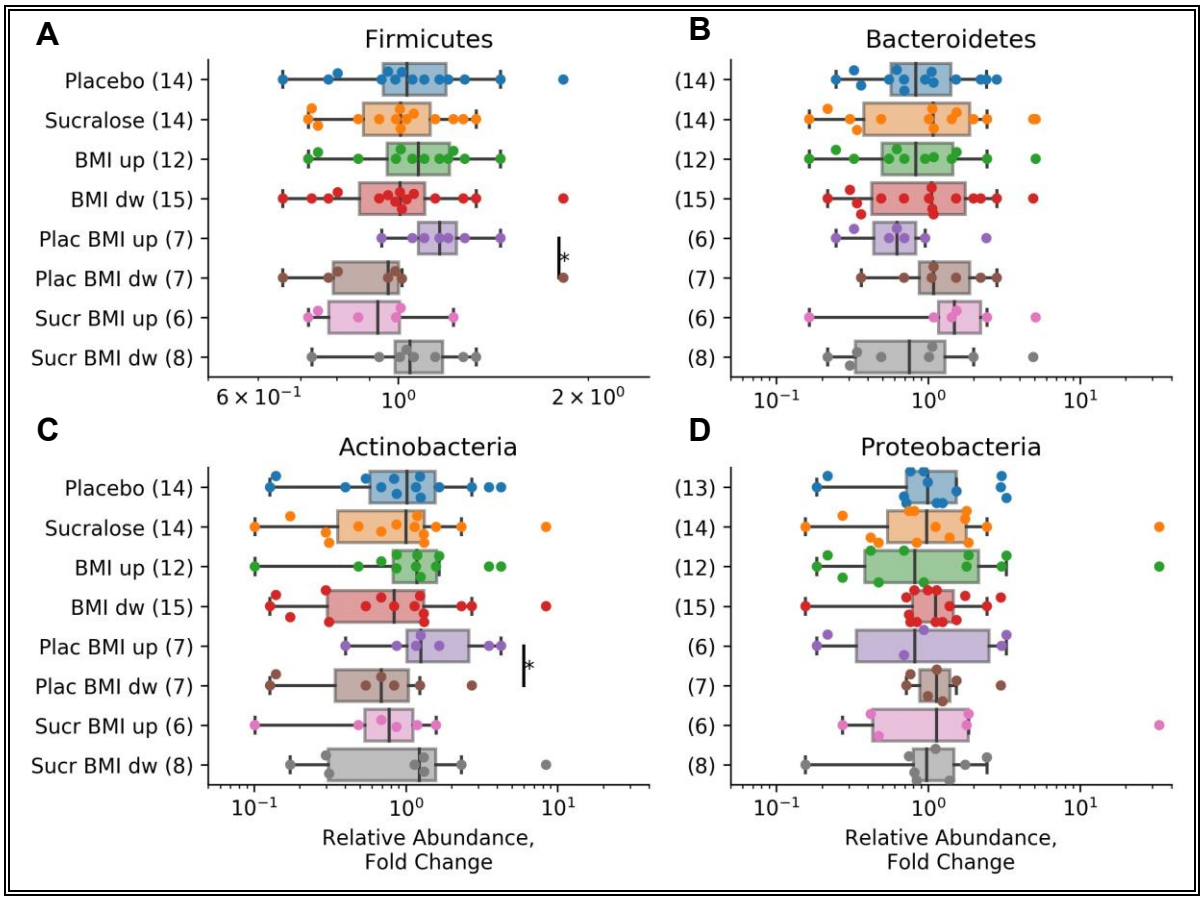


Figure S 10-4. Pairwise correlations between fold changes in microbiome phyla and BMI. BMI values considered were only on day 1 of the study, and subjects were classified as overweight (BMI > 25) or normal weight (BMI < 25). Then the abundances of the four gut microbiomes were correlated with these groups. A: Fold changes in Firmicutes; B: Bacteroidetes; C: Actinobacteria; D: Proteobacteria. Boxplots indicate the median and the interquartile range, with whiskers determined as the 1.5 range of the box. An asterisk denotes significant differences (p-value < 0.05) determined with the Mann-Whitney U-test.

11. ANNEX CHAPTER II: ASSESSING THE ROLE OF TRUST PROFILES FOR THE ECONOMIC GROWTH OF SOCIETIES: A STOCHASTIC RULE-BASED SIMULATION USING THE PRISONER´S DILEMMA GAME

Pablo Monares[1], James H. Liu[2], **Rodrigo Santibáñez**[1], Alejandro Bernardin[1], Ignacio Fuenzalida[1], Tomás Pérez-Acle[1,3,4], Robert Jiqi Zhang[2]

[1]Computational Biology Lab, Fundación Ciencia & Vida, Santiago, Chile.

[2]School of Psychology, Massey University, Auckland, New Zealand.

[3]Centro Interdisciplinario de Neurociencia de Valparaíso (CINV), Universidad de Valparaíso, Valparaíso, Chile.

[4]Facultad de Ciencias de la Salud, Universidad San Sebastián (USS), Santiago, Chile

Published in *IEEE Transactions on Computational Social Systems* (2020). Vol. 7, Issue 4, Pages 849-857

11.1 SUMMARY

According to Robert Putnam, trust can be a proxy for social capital. Thus, a higher societal trust could be related to economic growth. To test this hypothesis, we simulated the association between trust and economic growth in two artificial societies. One artificial society (New Zealand) exhibited higher levels of initial trust, and the other (Argentina) had lower levels of trust. Initial starting points for simulations were set using representative survey data (using the global trust inventory). Computational simulation relied on a rule-based model (RBM), integrating time through a stochastic simulation algorithm implemented in PISKaS. Agents in the artificial societies were distributed according to proportions of four trust profiles, with more high trusters (HTs) in New Zealand. In each iteration, the agents played a prisoner´s dilemma, earning or losing money according to different payoff matrices, cooperation probabilities, and interaction frequencies, modeling different conditions for economic exchange. We analyzed the economic performance of each country, together with the performance of each trust profile. Results support the notion that

societies with high trust perform economically better, on average, than those with low trust, but only if interaction frequency is held constant. Despite the relevance of HTs for economic development, their performance is tightly linked to the type of society in which they interact: they prosper more in a Rule of Law society, and where HTs are more common, compared with a predators´ paradise, where the sucker´s payoff is more punitive.

11.2 INTRODUCTION

One of the more difficult challenges in social sciences is to develop diachronic accounts of how processes and structures unfold in time and across space (Fuchs & Archer, 1997; Regan & Archer, 1992). Typically, empirical data in the social sciences (e.g., surveys, experiments, and observations) focus on particular times (like the present) and places (i.e., western societies, see Henrich et al., 2010). By contrast, computer simulations can easily examine the consequences of social processes unfolding over a thousand or a hundred thousand iterations, simulating extended passages of time. Hence, these have been part of evolutionary theorists´ toolkit since the classic computer simulation tournament on the evolution of cooperation (Axelrod & Hamilton, 1981).

The tradition of using computer simulations to examine the evolution of cooperation has continued to extraordinarily sophisticated models, such as Choi and Bowles´ simulation (2007) of the evolution of parochial altruists. It would seem that there should be a thriving interplay between social science theory and its empirical tests through computer simulation. However, to date, this study is thin. There are, for example, much less advanced studies on simulating the growth of economic prosperity compared with the evolution of cooperation even though they both involve processes of social exchange.

The evolution of cooperation is rooted in biology, whereas the growth of prosperity is rooted in material conditions involving the economics of a society. Simulating the growth (or decline) or prosperity therefore potentially requires many more factors that could be hard to simulate. The purpose of this article is to describe a rule-based simulation of the growth of prosperity using a model of trust derived from real, nationally representative survey data (J. H. Liu et al., 2018) as the first step in this direction. To do so, New Zealand and Argentina were treated as prototypes of a developed (OECD) and a developing economy, respectively.

We demonstrate that the basic processes involved in simulating the evolution of cooperation can be applied to derive insights into the growth of prosperity across different contemporary societies.

11.2.1 Growth of Prosperity from a Trust/Social Capital Perspective

In recent decades, there has been a surge of interest across the social sciences in examining the functions of social capital in producing prosperity (Coleman, 1988; Putnam, 2000a). According to Woolcock and Narayan (2000), social capital refers to the norms and networks that enable people to act collectively. It is all the values that flow from people´s inclinations to form associations with one another and from the individual to the group level (Evers, 2001). Such a vast concept is notoriously difficult to measure (see Paldam, 2000), especially across time and across cultures. Therefore, for the purposes of this article, we focus on a more easily measured concept that is theoretically closely related to social capital, that of trust. Putnam (Evers, 2001) conceptualized trust as a proxy for measuring social capital. While other theorists differ as to the exact relationship between the two, most agree with Simmel (1950) that trust is one of the synthetic forces capable of integrating micro-level social processes with group dynamics and macrolevel institutions (see Rousseau et al., 1998).

Where there is social capital, is also likely there is trust. The basic idea is that trust opens up opportunities to form social capital (Coleman, 1988), and social capital builds trust (Evers, 2001). Many poorer communities lack trust beyond small and immediate relations, so they cannot benefit from the prosperity an open economy brings (Fukuyama, 1995). They tend to stay within small, tightly sealed communities with a limited radius of trust. They remain poor because the dilemma of trust is one in which opening oneself up to possibilities of prospering by trusting strangers also opens oneself up to the possibility of being exploited (see Fukuyama, 1995; or Rousseau et al., 1998).

Furthermore, previous research has shown that wealthy countries typically have higher levels of trust than poorer countries (as illustrated, for instance, in J. H. Liu et al., 2018, with the notable exception of China, Zhang et al., 2019). Liu *et al.* (2018) developed a new measure of trust, i.e., the global trust inventory (GTI) that conceptualizes trust as a

system of meaning with “subcomponents (factors) and an overall configuration that makes sense of social interactions that link the individual to different levels of society.” The GTI measures the factors of trust with reasonable scalar invariance across 11 countries varying from developed to developing, allowing the functions of trust to be tested across economies at different stages of development.

The particular factor that we focus on here is the Rule of Law (see data.worldjusticeproject.org/). A liberal (open) economy needs the Rule of Law to thrive. What Rule of Law does it is to protect a person or company from incurring maximum damage from being cheated by some unscrupulous other during social/economic exchange. Typically, poorer countries with shorter histories of democracy have weaker legal institutions: they tend to be more corrupt than more advanced economies (Fukuyama, 1995; Mishler & Rose, 2001). Putting these two patterns together, a theory of trust/social capital would predict that higher trusters should do better in a society with a stronger Rule of Law, whereas low trusters (LTs) (or cheaters) should do better where the Rule of Law is weak (and they are less likely to be punished for cheating others). This hypothesis connects the growth of prosperity with evolutionary theories, where higher-level institutions (i.e., group-based selection in this case Rule of Law) impact on the individual-level characteristics (e.g., propensity to trust, see J. K. Choi & Bowles, 2007 for an evolutionary simulation the effects of group-level selection).

11.2.2 A Rule-Based Simulation of New Zealand and Argentina

The abovementioned hypothesis is consistent with evolutionary theory but could be criticized as being too simple for simulating a modern economy. In this article, we argue to the contrary that simulations are useful in projecting the outcomes of simple rules of trust and social exchange because they enable inferences about real-life situations.

The key finding of Axelrod and Hamilton´s classic simulation (Axelrod & Hamilton, 1981) (replicated many times subsequently, see Rand & Nowak, 2013 for a review) was that tit-for-tat (reciprocal altruism, a high trust strategy) is evolutionarily stable against the rational, egoistic strategy of always-defect (a low trust strategy) as long as the computer agent is able to learn from prior encounters. That is, over repeated interactions, the computer

agent in their simulations learned whom to trust and whom not to trust, and as long as there were enough other trusters in the simulation, their mutually rewarding interactions would win the day and compensate for the "sucker´s payoff" of being cheated after trusting someone who betrays their trust. However, the requirements for a high trust strategy to be successful could depend on other things, such as the reward-cost environment (e.g., Rule of Law, or other normative factors reducing the cost of being cheated in the PDG), in addition to the preponderance of other trusters in the local environment. Furthermore, a major question in the field is: What are the minimum conditions required for high trusters (HTs) to thrive against defectors? (Rand & Nowak, 2013). This is precisely what we intend to examine.

As mentioned earlier, it is a challenge, especially in the initial steps of a long-term research project, to decide what factors to simulate in the growth of prosperity. In this research, we use the same paradigm of Axelrod and Hamilton (1981), the prisoner´s dilemma game (PDG), to model social interactions in our simulation, where avoiding or, at least, mitigating the costs of the sucker´s payoff is critical to the growth of cooperation and by an extension (we argue) to the growth of prosperity. We simulate social exchange between different agents with different levels of trust [e.g., HT, medium truster (MT), LT, and low institutional truster (LIT)] over repeated interactions. This research is the first step of what it is planned to be a series of increasingly more sophisticated simulations. We begin our program of research with naïve agents, that is, agents who are unable to learn from their previous encounters and unable to adjust their behavior to avoid being repeatedly cheated by defectors. The literature from evolutionary theory suggests that such a cooperative strategy is unlikely to survive competition against defecting strategies over the course of evolution (Axelrod & Hamilton, 1981; Rand & Nowak, 2013). Since our simulation is focused on a different question, whether HTs can thrive under specific initial societal conditions surveyed in two countries, we reasoned that beginning our program with naïve or "dumb" trusters is the logical starting point to look at for minimum conditions required for co-operators to prosper.

Our simulation further differs from evolutionary theory-based simulations in two crucial ways. The first is in the meaning of the dependent variable. In the abstract, this is

points gained after a round of PDG exchanges. In evolutionary theory-based simulations, these are interpreted as "survival scores" because they are used to determine how many offspring a given computer agent/strategy spawns in the next round. Thus, points gained in the social exchange are used as the metric to decide the survivability of a given strategy (Axelrod & Hamilton, 1981) or personality type (J. K. Choi & Bowles, 2007).]. Each round or iteration of the computer simulation can be interpreted as the success or failure of a generation of living organisms competing with one another to survive and pass their genetic material (i.e., their computer algorithms) onto the next generation. This interpretation does not apply to our simulation, as points are treated as money earned, not as survival scores. Each iteration of our simulation can be abstractly interpreted as a certain period in the agent´s life. In a 100-time-step simulation, each round could be interpreted as six months, whereas in a 1000-time-step simulation, each iteration could be treated as a fortnight— this is not important. What is crucial is that in our simulation, points are interpreted as the overall financial success of a given computer agent during a significant span of time in their lives.

Second, we innovate by varying the outcome matrix or rewards and costs associated with the social exchange in the context of real survey data related to actual economies, using a rule-based modeling approach. As mentioned previously, highly developed economies tend to have a Rule of Law that attenuates the costs of being cheated and lowers the rewards of cheating. For example, the cheater would be more likely to be caught by police or successfully sued in the court for unscrupulously taking advantage of someone in New Zealand compared with Argentina (see Corruption Perceptions Index, 2018, or the Rule of Law measure at https://worldjusticeproject.org/). We used data from a multi-country survey on trust and trusting behavior as the starting point of our simulations of the growth of prosperity in these two societies. This principle of normative (or group-based) cooperation in punishing defectors is also crucial in the evolution of cooperation (see Choi & Bowles, 2007; or Richerson et al., 2016), but to the best of our knowledge, varying the outcome matrix has not been prioritized in simulating environmental conditions surrounding dyadic interaction in evolutionary PDGs.

Liu *et al.* (2018) used nationally representative samples to demonstrate that the structure of trust was consistent across 11 democratic societies, including both developed

and developing societies. Aggregating across the seven factors of trust (from close relationships all the way up to trust in government), they used latent profile analysis (Andersen et al., 2003) to ascertain the overall number of people with high trust (HT), moderate trust (MT), low trust (LT), and low institutional trust (LIT). In New Zealand, the percentage of HTs was 28.3%, versus 7.3% in Argentina, whereas LTs were 6.2% of the survey population in New Zealand and 12.7% in Argentina. Furthermore, Romano *et al.* (Romano et al., 2017) used a behavioral experiment (the trust game) to report the likelihood of actually trusting another person from these same samples (and other non-democratic and nonwestern societies) in economic exchange. HTs were found to trust another person in a social dilemma of trust 57% of the time compared with only 47% of the time for LT. MTs were in between these two categories in terms of the likelihood of trusting another person. Furthermore, we found that HT self-reported more social interactions per week than LT and Argentinians reported far more social interactions than New Zealanders. We use these data as the starting point to define the computer agents in our prosperity simulation.

Starting points are particularly important according to critical junctures theory (J. H. Liu et al., 2014), a dynamical system theory-based approach to study the rise and fall of societies. Within this approach, small differences in initial settings can end up producing large differences in a system´s trajectory, known as the "butterfly effect" (for an overview, see Vallacher et al., 2010). Hence, we seek to examine whether initial starting points of states and the distribution and tendencies of individuals within states determine both the relative individual prosperity and the wealth of the state as a whole. This is of particular interest in attempting to identify critical junctures characterizing the growth of prosperity in a developing (Argentina) versus a developed (New Zealand) state.

11.2.3 Rule-based Models: a novel approach to model human societies

Human societies are an exemplar of complex systems: the diversity of interactions as well as the intricacy of their connectivity produce emergent behavior over time (Perez-Acle et al., 2018). In studying their dynamics, they are often modeled by using agent-based model (ABM) approaches. A particularity of these models is that interacting agents follow a set of internal rules (Gilbert, 2008), imposing a teleonomic nature on these agents (de Laguna,

1962). Hence, ABM models are interesting for simulating human beings because they do not necessarily follow a strict biologically bound reproduction framework. Moreover, from both the philosophical and biological points of view, it is arguable that living systems have a general-purpose beyond reproduction (de Laguna, 1962; Perez-Acle et al., 2018).

Rule-based models (RBMs) offer an alternative to ABM because these do not require interacting agents to have an internal purpose nor teleonomy. RBM agents express properties that allow them to interact with each other by following rules that are environmentally coded. While some properties may account for internal states of the agents (e.g., trust level, individual prosperity, and sociability), others afford interaction between agents [e.g., population density might increase interaction rates (IRs)]. Following these premises, our simulations were performed by using PISKaS, a parallel computational platform to execute RBMs using the stochastic simulation algorithm (SSA, Daniel T. Gillespie, 1976). PISKaS is capable of interpreting and executing models written in the Kappa language (Perez-Acle et al., 2018). While RBMs and SSA have been applied to study system dynamics in a variety of fields (Chylek et al., 2015; James R. Faeder et al., 2009), applications to social systems are less common and more recent. Their fundamentals have been extensively reviewed in Bustos et al., 2018.

A particularity of our RBMs is that the system dynamics are generated directly from the interaction between agents rather than through an imposed teleonomic mechanism. In other words, the system´s dynamics is the consequence of the continuous interaction between agents over time. Hence, by following this modeling framework, we treat human societies as complex systems exhibiting emergent behavior as a consequence of the nonlinear and highly correlated interaction between agents in the simulation. Importantly, modeling human societies using RBMs allows us to follow an approach without preconceived notions of its global behavior, nor how the studied phenomenon emerges. In doing so, we are studying social phenomena by gradually expressing all the known facts governing the interactions between agents using incrementally complex rules along with simulation time (Forbes et al., 2018). Of note, RBMs are also ideal to perform causal mechanistic analyses, that is, to identify the chain of events necessary to obtain a particular result in a given system (Forbes et al., 2018). This characteristic is very appealing to our

study because we want to disentangle the intimate relationship between trust and prosperity, shedding light on the causality between trust/social capital, system characteristics (such as the presence of Rule of Law or not), and economic development.

11.3 MATERIALS AND METHODS

11.3.1 Simulations

All simulations were based on initial starting points obtained from surveys of Argentina and New Zealand reported by Liu *et al.* (2018). As mentioned earlier, contrary to classical works linking empirical methods with ABM (Janssen & Ostrom, 2006), we followed an RBM approach. In doing so, we focused on analyzing the economic performance of the computational social systems that we developed, intended to resemble human societies at different stages of social and economic development. In both cases (New Zealand and Argentina), agents were categorized according to the trust profiles identified by J. H. Liu *et al.* (2018).

In our RBMs, interactions between agents consider the particularities of each trust profile and the consequences of these interactions, as expressed by a precise formal statement or rule. Each rule has a certain reactivity or probability to be executed, which is defined by specific parameters of each trust profile: the size of the profile, their IRs, and their cooperation probabilities (CPs). Importantly, while IR determines how often agents interact, CP denotes how often agents defect or cooperate while playing the PDG during their interactions.

All simulations were performed using the PISKaS stochastic simulation engine (Perez-Acle et al., 2018). Each simulation is defined by a set of parameters (see below), considering the country, the distribution of the trust profiles within this country, the IR, the CP, and the payoff matrix. Each simulation was executed 1000 times, randomly changing the initial seed of the SSA each time, to collect data and compute statistics.

Our RBM is generally defined by the following premises:

1) all the agents belonging to the same country inhabit in the same compartment;

2) interactions occurring between pairs of agents involve a positive or negative outcome, increasing or decreasing the agents´ prosperity;

3) all agents, irrespective of their trust profile, are homogeneously distributed in the compartment;

4) when agents interact, they do it at the lowest IR of the interacting agents;

5) money is earned (or lost) by agents in each particular interaction;

6) each country has 200000 agents distributed according to the percentages of each trust profile (defined by the GTI using representative national samples); and

7) agents are naïve in the sense that they do not learn from previous interactions and cannot remember information about previous transactions. A representative set of simulation rules is shown in Figure 11-1.

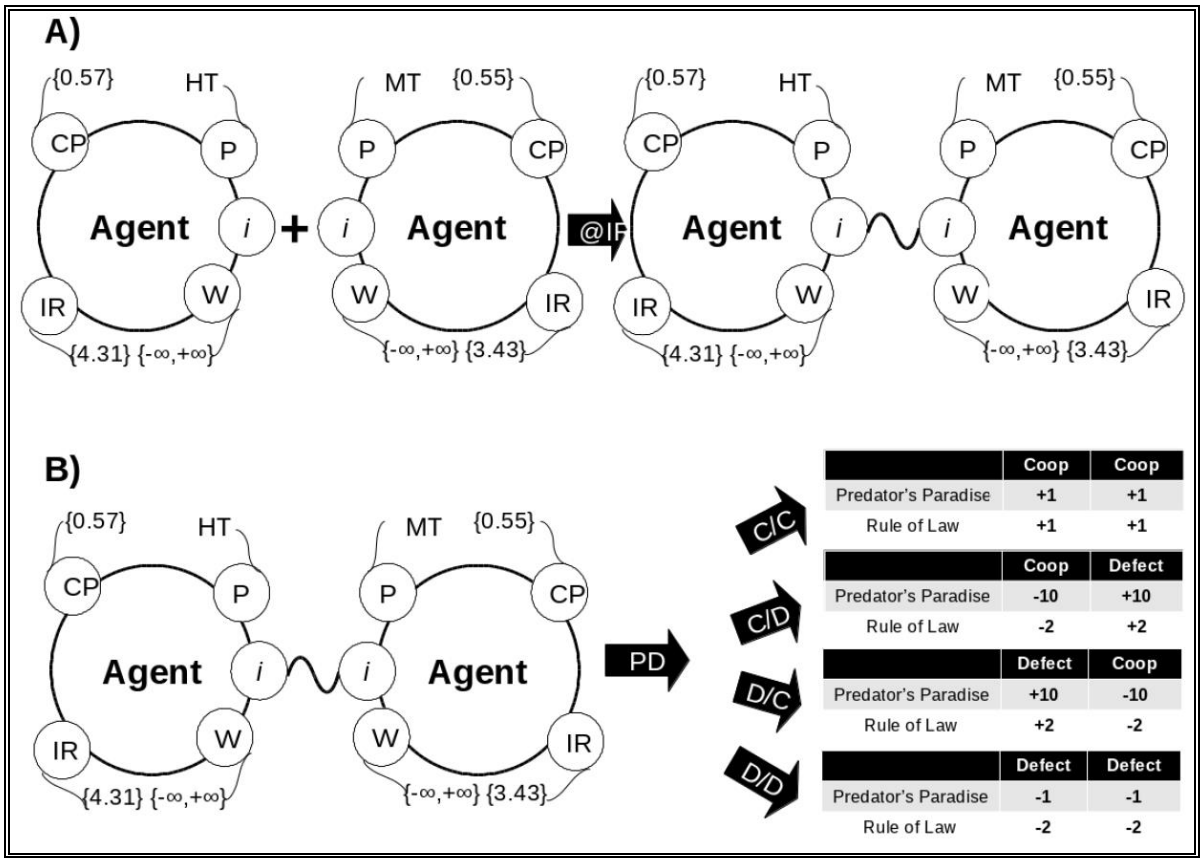


Figure 11-1. Main simulation steps.
(a) Example of two agents belonging to New Zealand, one HT and one MT, interacting along with the simulation. As a result of the interaction at rate IR between both agents, a complex is formed. (b) Two interacting agents forming a complex play the PD game during an iteration of the simulation. As a result of the PD game, one of the four outcomes of the payoff matrix is obtained, either increasing or decreasing the amount of money that it is stored on

the agents´ wallet (W). Note that each agent has sites identified as properties denoting their trust profile (P), IR, CPs, and the interaction site (i). Numerical values for all the properties belonging to the interacting agents are extracted from Table 11-1. Both agents are immediately separated after playing the PDG game, becoming available to participate in new iterations of the algorithm.

11.3.2 General parameters

For each country, we considered the four trust profiles established by the GTI according to Liu *et al.* (2018): HTs, MTs, LITs, and LTs. Of note, due to the characteristics of the applied instrument, these parameters are both empirically based and nationally representative. For each trust profile, we defined a set of parameters, including profile distribution, IRs, and CPs (see Table 11-1). IRs for New Zealand and Argentina were obtained from the GTI instrument designed and applied by our collaborators (J. H. Liu et al., 2018). Therefore, these values were extracted from representative samples of each society, reflecting, in a certain way, the character of the social dynamics in each country.

Table 11-1. General parameters according to the Trust profiles per country.
Note that we defined cooperation probabilities to be the same for the four profiles across the two countries.

Profiles	**Profile distribution**		**Interaction rates**		**Cooperation probabilities**
	Arg	**Nz**	**Arg**	**Nz**	
HT	7.30%	28.30%	14.09	4.31	0.57
MT	29.80%	46.40%	10.75	3.43	0.55
LIT	50.20%	19.00%	9.86	3.21	0.55
LT	12.70%	6.20%	8.32	1.46	0.47

11.3.3 Payoff matrices

In order to evaluate the effects of societal organizations that to a different degree punish (or reward) defection in economic transactions between individuals, we evaluated the effect of two different payoff matrices: A Predator´s Paradise and a Rule of Law matrix. The

Predator´s Paradise is conceptualized as a society where there is a high benefit for cheating and a high cost for being cheated. By contrast, a Rule of Law society is conceptualized as one in which there is a higher chance of being caught and being punished for cheating, so the benefit for cheating is less and the cost of cooperating with a defector is also lower.

Hence, the Predator´s Paradise matrix consists of a higher payoff to defecting versus cooperating in the PDG. In the Predator´s Paradise, the payoff matrix is T: +10; R: +1; P: −1; and S: −10, where T represents the temptation, R represents the (mutual) reward for cooperation, P represents the (mutual) punishment (for mutual defection), and S represents the sucker´s payoff (being cheated). In the case of the Rule of Law payoff matrix, defecting is less encouraged, as it only pays 2. Hence, the payoff matrix is T: +2; R: +1; P: −1; and S: −2. Note that with these payoff matrices, the only interactions that contribute to increasing prosperity of the society overall are those of mutual reward (R/R), whereas mutual defection (P/P) produces an overall loss of prosperity to society.

11.4 RESULTS

11.4.1 Gross Domestic Product

As a first approach to study the relationship between trust and economic performance, we analyzed the overall economic performance of the two countries during the simulation. Hence, we considered the total accumulated money per country [or gross domestic product (GDP)] at the end of every simulation. To do so, earnings per agent per iteration were added up and the accumulated per trust profile summed up to compute the country´s GDP. Using the parameters obtained from Table 11-1, independently of the selected payoff matrix, the overall economic outcome of Argentina was better than that of New Zealand under either Predator´s Paradise or Rule of Law (see Figure 11-2A and 11-2B). This better performance is likely to be due to the higher IR for Argentina that was derived from Liu *et al.* (2018), compared with that of New Zealand (see Table 11-1). To confirm this hypothesis, for both countries, we divided the GDP by the total number of interactions in every time step (see Figure 11-2C and 11-2D). Now, independently of the type of societal organization, the economic performance of New Zealand (controlling for the number of dyadic interactions per time step) was better than that of Argentina. This denotes that the higher trust rates in

New Zealand lead to more mutually beneficial interactions (which are the only ones that increase overall GDP).

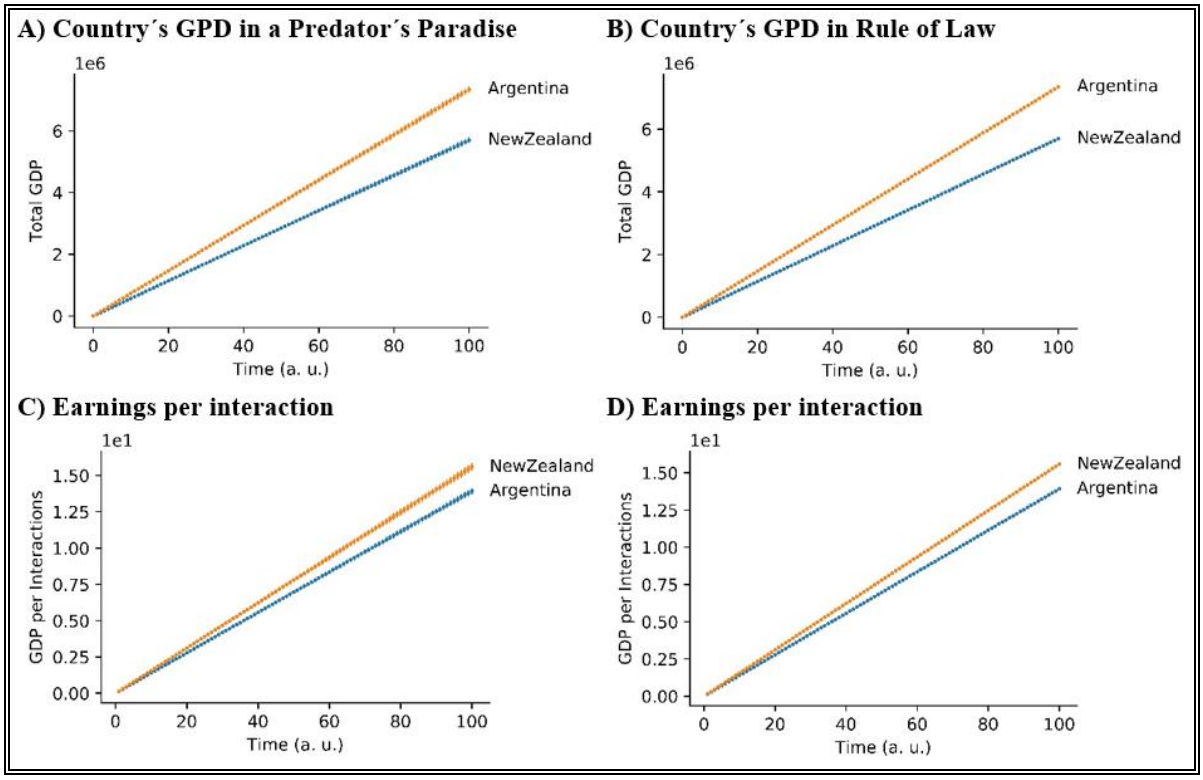


Figure 11-2. Cumulative and per interaction GDP.
Cumulative GDP for the payoff matrices representing high sucker´s payoff (Predator´s Paradise, A and C), and low sucker´s payoff (Rule of Law, B and D). Panels A and B show the cumulative GDP achieved along with the simulation. Panels C and D show the cumulative GDP when this amount is divided by the total of interactions within each country.

11.4.2 Dissecting economic performance per trust profile in the two societies.

To explore the dynamics of economic performance for the different trust profiles, we examined the per capita GDP per trust profile contrasting the Predator´s Paradise versus the Rule of Law payoff matrices for Argentina and New Zealand (see Figure 11-3).

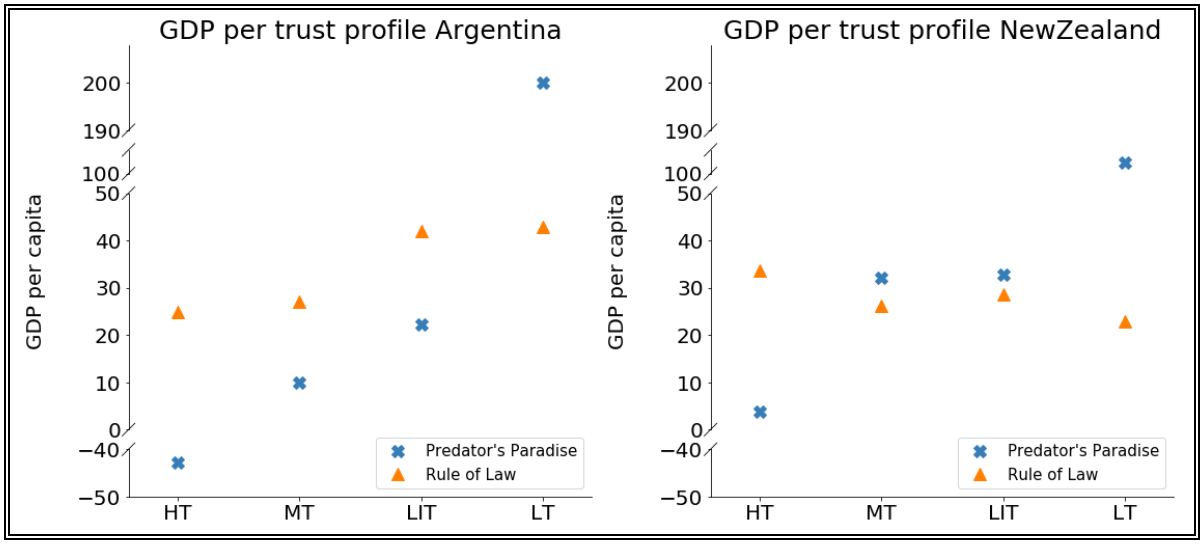


Figure 11-3. Economic performance per trust profile in Predator´s Paradise and Rule of law societies using GTI parameters.
The GDP per capita for HT (High), MT (Medium), LIT (Low Institutional) and LT (Low Trust) profiles for both countries is displayed for the two payoff matrices.

In the Predator´s Paradise payoff matrix, where there is a high benefit for cheating and a high cost for being cheated, agents belonging to the MT, LIT, and LT profiles all earned more money in Argentina (per capita), whereas agents belonging to the HT profile lost money during the simulation. As consequence, the cumulative GDP of the HT profile was negative at the end of the simulation. Of note, even though the LIT profile was numerically the largest in Argentina, the economic growth within the country (GDP) was largely attributable to the performance of the LT profile. Other profiles linearly decreased in per capita prosperity compared with this profile of agents most likely to defect. Thus, if the payoff matrix is modeled as a Predator´s Paradise for a simulated society, such as Argentina, cheaters prosper and naïve HTs suffer.

On the other hand, in contrast to Argentina, all trust profiles in New Zealand earned money over the course of the Predator´s Paradise simulation. In agreement with Argentina, the LT profile was also the best performer per capita in New Zealand, and both MT and LIT profiles exhibited similar per capita economic performance. Although cheaters also prospered in New Zealand, the HT profile did as well, having a positive GDP per capita, contrasting with the case of Argentina, where they lost money. This is probably due to the

greater preponderance of HT and MT profiles in New Zealand, as well as a less exaggerated tendency for the higher trust profiles to interact more per round in New Zealand compared with Argentina.

For the Rule of Law payoff matrix, irrespective of their trust profile, all agents earned money during the simulation. The far less punitive sucker´s payoff was responsible for this result. The per capita economic performance of the LT and LIT profiles in Argentina was very similar. Of note, the per capita economic performance of the HT and MT profiles in Argentina was also very similar and lower than that for the two less trustful profiles. Notably, the difference in prosperity between the four profiles was far less pronounced under the Rule of Law compared with the Predator´s Paradise payoff matrix in Argentina.

In marked contrast to Argentina, using the Rule of Law payoff matrix, the best per capita economic performance in New Zealand was achieved by the HT profile, followed by LIT, MT, and, in the last position, the LT profile. Of note, despite the limitations of our simple model that used naïve agents incapable of discriminating between cheating and co-operating interaction partners, the HT profile exhibited the best per capita economic performance. Even without any social intelligence (Yamagishi et al., 1999), more cooperative and more socially interactive agents (HT) prospered more than a defecting type of agent (operationalized by LT), given the Rule of Law payoff matrix. Differences between the four profiles were relatively small in this scenario.

Independent of the societal organization as modeled by the payoff matrix, the per capita economic performance of the trust profiles in Argentina was always the same: the best performers are LT, followed by LIT, MT, and, in the last position, HT. The stability of this order suggests that the key element for per capita economic performance in the Argentina simulation is the interaction strategy of the trust profiles. The poor per capita economic performance of the HT profile in Argentina in the Predator´s Paradise payoff matrix—losing money—contrasts with the performance of the HT profile in the Rule of Law payoff matrix, where all agents earned money. The dependence of the economic performance of the HT profile on the type of institutions prevalent in a specific society highlights the critical role of institutions in growing prosperity for a country (see J. H. Liu et al., 2018; Richerson & Henrich, 2012).

11.5 DISCUSSION

Insights into the growth of prosperity were uncovered through an RBM stochastic simulation using real survey data from two societies as its initial starting point. First, the minimum conditions for the growth of individual-level prosperity through cooperation in society were identified. It was found that even without any social intelligence (Yamagishi et al., 1999), a highly interactive, cooperative agents (HT) were capable of outperforming a less interactive, less cooperative agents (LT) when there were more cooperative agents in the social environment (28% HT in New Zealand versus 7% in Argentina). In addition, these agents interacted more often than less cooperative agents, under conditions where the sucker´s payoff for being cheated was less punitive (−2 versus −10). The vast majority of the studies that simulated cooperation between computer agents have used an evolutionary interpretation of the PDG payoff matrix and, hence, have largely neglected to theorize about details of the numerical settings for these payoffs. In our simulations, many of these parameters were derived from real survey results (J. H. Liu et al., 2018), and, therefore, are meaningfully concrete numbers rather than abstract indicators. The interpretation of the payoff matrix, therefore, also deserves theoretical attention.

In the evolutionary literature, one Prisoner´s Dilemma is more or less the same as any other, whereas, in our prosperity simulation, the Predator´s Paradise payoff matrix yielded qualitatively different outcomes compared with the Rule of Law matrix. When the sucker´s payoff was set at +10 for temptation and −10 for being cheated, the LT profile, with a 47% cooperation rate (8%–10% lower than the other three profiles), prospered massively (making 10 times more than the next profile in Argentina and three times more than the next in New Zealand). When the sucker´s payoff was set at ±2, this resulted in far more equitable results between the four trust profiles in Argentina even though the rank order of their per capita GDP earnings remained unchanged ($LT > LIT > MT > HT$). In New Zealand, a qualitatively different set of results appeared, where the more cooperative, trusting profiles prospered more ($HT > LIT > MT > LT$), even though the computer agents could not distinguish cheaters from co-operators.

These results highlight the importance of the institutional environment in deciding which types of individuals would thrive in a given society. Trust and social capital have been predominantly theorized as positive or perhaps even necessary attributes for societal prosperity (e.g., Fukuyama, 1995) and this received considerable empirical support (e.g., Knack & Keefer, 1997; Zak & Knack, 2001). However, the literature has not emphasized how many institutional conditions weigh into this as an important contextual constraint. This is probably because trust and institutional performance are often tangled together in real-life situations. For example, a high trust society is often one with high institutional performance as well; therefore, it is difficult to examine whether a change of institutional conditions would alter the impact of trust (see also Guiso et al., 2004). Here, we were able to disentangle the interplay of trust and the institutional environment by setting them as separate parameters in simulations. Our results indicate that, in contradiction to hardline social capital theorists, there are negative implications of being a trustful person in the wrong context. A high trustor ends up in a most disadvantaged position if their cooperative tendencies are not safeguarded by normative protections, such as a rule of law being observed by other people around them. This insight accords with survey research from South Africa, a developing country with tough living conditions in many places (Adjaye-Gbewonyo et al., 2018), research reported that HTs living in a district with lower generalized trust showed more depressive symptoms in the second wave of a cross-lagged panel study.

Even though modeling a modern economy as the sum of dyadic interactions between individuals is a gross simplification, the parameters that we used in these initial simulations are still instructive. In accord with insights from evolutionary theory (Rand & Nowak, 2013), the number of other co-operators in society was influential in deciding the overall earning of a given trust profile. In New Zealand, where more than a quarter of the population is HT, the social environment is more conducive to an HT thriving if institutional conditions are favorable (e.g., if there is Rule of Law). In Argentina, even with the Rule of Law, HT was disadvantaged because LTs were twice as common as in New Zealand. Making the situation more complicated were the survey results that made Argentina agents about 3-fold more socially active than New Zealand agents. The Argentina HTs could be viewed as gluttons for punishment as they not only were more likely to cooperate than any other profile, but

also, they interacted more per round and, hence, were constantly being exploited by LT. By contrast, the New Zealand LTs were like the alligator snapping turtles, ready to take a bite out of naïve HTs, but slow to do so (they only interacted 1.46 times per round compared with 14.09 times for Argentina HT). The differential social interaction rate was undoubtedly part of the reason for the results obtained. Future research might want to take this factor out, or make it less salient, as it may be that our Latin American survey respondents were reporting their frequency of social rather than economic interactions.

The high level of interaction simulated for Argentina produced the unanticipated result that Argentina had better performance overall than New Zealand. Only when interaction rates were controlled for (Figure 11-2, C and D) made New Zealand have better economic performance compared with that of Argentina, as was anticipated due to its high number of HT and MT (in New Zealand the two profiles constitute 74.7% of the total population compared with only 37.1% in Argentina). Thus, in our simulations, although New Zealand society had fewer interactions between agents, these interactions were more successful on average in contributing to GDP than those for Argentina.

11.6 CONCLUSION

Whether we think of payoff matrices in the relative terms of economic prosperity or in absolute terms of evolutionary survival, the distribution of trust types makes a difference, just as predicted by the literature (Axelrod & Hamilton, 1981; Rand & Nowak, 2013). HT does better in New Zealand, where there are more like-minded and like-acting people. A highly versus slightly punitive sucker´s payoff makes a difference in terms of the relative performance of different trust types. If HT is dispositional rather than learned (as implied in our simulation, where there was no social learning by the agents), it will be punished more (or eliminated) in a developing society, such as Argentina, but not in a developed society, such as New Zealand. We saw this in the case of the Predator´s Paradise in Argentina, where the HT profile alone became impoverished, while all the other three profiles prospered. In future simulations, we plan to allow agents to learn and switch their trust type or acquire social intelligence at discriminating between trustworthy versus untrustworthy interaction

partners (see Yamagishi et al., 1999). This might produce a more realistic simulation and reduce the robustness of the advantage of LT in a Predator´s Paradise.

Notwithstanding these limitations, it was an impressive result for HT to earn the most money per capita in New Zealand under a Rule of Law societal organization, even with zero social intelligence. This suggests that there may be ranges of societal conditions within which co-operators may prosper economically that extend beyond those previously identified in the evolutionary literature. It suggests that there are some societies that make it easier to be a "good person" (a co-operator) than others, and this can be attributed to the situation, not the individual. Although we have interpreted our results as the growth or decline of prosperity in contemporary society, mathematically, our simulations are not that different from previous simulations with evolutionary interpretations. More systematically exploring/varying initial simulation parameter settings (e.g., the distribution of trust types, the form of societal organization and interaction tendencies of different trust types) may be a fruitful direction for expanding our understanding of the conditions under which cooperative actors may thrive in future research.

12. ANNEX CHAPTER III: UNCERTAINTY OF PARAMETER VALUES

To determine the uncertainty of parameter values, *Alcyone* serves as a *wrapper* function around *Pleione* (Chapter IV). It calculates uncertainty in parameter values from a one-leave-out Jackknife and Bootstrapping procedures.

12.1 IMPLEMENTATION

12.1.1 One-leave-out Jackknife

Jackknife works taking out one experimental value from the measurements and calculating the parameter of interest from the remaining values. It is a general procedure, and in the case of calibration, the calculation proceeds as:

First, take out one observation and perform calibration with the remaining values. For instance, if there are available three replications, *Alcyone* prepares three sets of two observations each. Then, *Alcyone* calls *Pleione* to calibrate with them (hence, the qualification as a *wrapper* function). Optionally, to determine the bias and determine an unbiased estimation of parameter values, *Alcyone* calls *Pleione* to calibrate the model with the totality of observations. Once finished the calibrations, the parameters for each best model are employed to calculate an estimation with equations 12-1 to 12-4. Equation 12-1 determines a biased estimation considering each parameter θ_i, where i is one of the Jackknifed calibrations. Then, *Alcyone* determines the variance with equation 12-2. To correct the estimation, equation 12-3 estimates the bias (the difference between the expected value and the true value) using the parameter values determined using the totality of observations ($\hat{\theta}$) and the unbiased estimation is calculated with equation 12-4 and 12-5.

$$\bar{\theta} = \frac{1}{n}\sum_{i=1}^{n}\theta_i \tag{12-1}$$

$$\mathrm{Var}(\bar{\theta}) = \frac{n-1}{n}\sum_{i=1}^{n}(\theta_i - \bar{\theta})^2 \tag{12-1}$$

$$\widehat{\mathrm{Bias}}_{(\theta)} = (n-1)\left(\bar{\theta} - \hat{\theta}\right) \tag{12-3}$$

$$\hat{\theta}_{\mathrm{Jack}} = \hat{\theta} - \widehat{\mathrm{Bias}}_{(\theta)} = n\hat{\theta} - (n-1)\bar{\theta} \tag{12-4}$$

$$\hat{\theta}_{\mathrm{Jack}} \pm t_{\left(1-\frac{\alpha}{2},n-1\right)} * \sqrt{\mathrm{Var}(\bar{\theta})} \tag{12-5}$$

12.1.2 Bootstrapping

Bootstrapping is the generalization of Jackknife and works simply as a random sampling with replacement. After calibration, the uncertainty is calculated as the percentile range. However, to estimate uncertainty with high confidence, many calibrations must be done to have enough samples to determine closed percentile ranges. For instance, to determine a 95% confidence range, *Alcyone* should calibrate at least 40 samples.

12.2 RESULTS

The simple model of gene regulation presented by Aguilera *et al.* (2017) was used as a testbed for its simplicity (four *Rules*, four parameters) and known parameters. From this model, ten stochastic simulations were used as synthetic experimental values. *Alcyone* was employed to determine uncertainty employing Jackknife (Table 12-1), and bootstrapping (Table 12-2) in a similar setting to those employed in strategy 3 for *Pleione* (Chapter IV). The function used was the two one-sided t-tests (TOST) and the Squared Deviations of two Averages (SDA, Equation 5-1). Deterministic simulation of the biased parameters estimated from Jackknifed calibrations is shown in Figure 12-1.

Table 12-1. Uncertainty in parameter values for the Aguilera´s model employing Jackknife.

	mRNA synthesis rate	**mRNA degradation rate**	**Protein synthesis rate**	**Protein degradation rate**
Jackknife 1	5,893897	0,03613419	0,1003642	0,02874476
Jackknife 2	6,334189	0,03962465	0,09436991	0,02713551
Jackknife 3	5,046561	0,0308266	0,1024753	0,02919243
Jackknife 4	5,165796	0,0303539	0,09126425	0,02704501
Jackknife 5	4,588181	0,02719895	0,1000036	0,02906013
Jackknife 6	5,977492	0,03517676	0,09115435	0,02769493
Jackknife 7	6,994644	0,04365462	0,09127987	0,02783525
Jackknife 8	6,129591	0,03798136	0,0961096	0,02862402
Jackknife 9	5,414017	0,03330024	0,1014478	0,02948169
Jackknife 10	5,025488	0,02913781	0,09970754	0,02980731
Complete dataset	6,107979	0,03837579	0,09381348	0,02661513
biased estimates	5,656986	0,034339	0,096818	0,028462
Jackknife estimates	10,166920	0,074708	0,066776	0,009992
Standard errors	2,088007	0,014674	0,012890	0,002781
90% CI	±4,723400	±0,033194	±0,029158	±0,006292

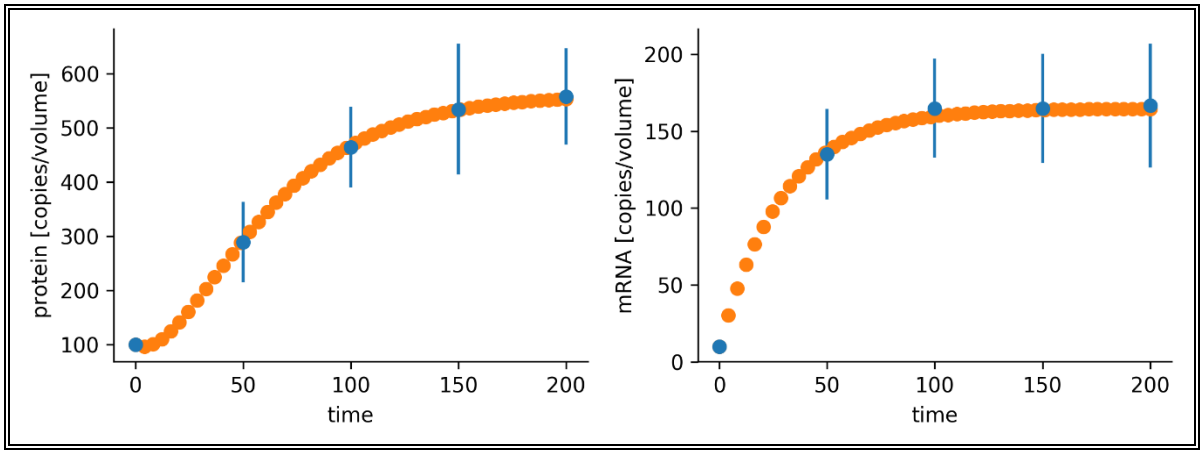


Figure 12-1. Deterministic simulation with the biased parameters estimated from Jackknifed calibrations.
Bars show one standard deviation of 10 stochastic simulations used as synthetic experimental data.

Table 12-2. Uncertainty in parameters values for the Aguilera´s model employing Bootstrapping with an 80% confidence interval (10 bootstrapping runs)

	True value	**minimum**	**maximum**
mRNA synthesis rate	5,0	4,910416	5,984259
mRNA degradation rate	0,03	0,0289544	0,0360525
Protein synthesis rate	0,10	0,0930859	0,1009508
Protein degradation rate	0,03	0,026989	0,0294768

12.3 OUTLOOK

Jackknife and Bootstrapping are common, general, and straightforward methods to determine uncertainty for any calculation derived from samples. In the case of parameter values, the procedure was able to determine ranges with close or no deviation from the “real” parameter values. Jackknife is fast as is complexity is only the number of measurements plus one calibration to estimate bias. However, it is an approximation and bootstrapping could be employed to determine uncertainty linked to confidence ranges at expense of slower calculation time due to the calibration of many subsamples. Moreover, both methods could be replaced with Bayesian and Approximate Bayesian Calculation methods.

13. ANNEX CHAPTER IV: TOWARDS *PLEIONE* 2.0

Pleione (Chapter IV) was developed with Rule-Based Models simulators in mind. Of the many simulators, *Pleione* supports four of them (covering two languages) and three main simulations methods: the original Stochastic Simulation Algorithm, the network-free method, and the deterministic method implement inside BioNetGen. Moreover, one of the claims was the ability to extend the method to support other simulation software. Here, it is explained the procedure using the Tellurium python package. The package allows simulation and analysis of SBML models, a standard used to disseminate models. However, SBML is not human-readable and Tellurium supports a model specificity language called Antimony. The model is human-readable and consistent as a programming language, and an important feature to determine how to encode inside the model the calibration specifications. Line numbers refer to the KaSim interface (https://github.com/networkbiolab/Pleione/blob/master/Pleione/kasim.py) while *Pleione* transits to be an object-oriented software.

13.1 SET THE REGULAR EXPRESSION

Pleione determines which parameters to calibrate from a regular expression. The variable is encoded on lines 261 to 267. The regular expression captures parameter names (using "`(\w+)`") and numbers (“`([-+]?(?:(?:\d*\.\d+)|(?:\d+\.?))(?:[Ee][+-]?\d+)?)`”). As complex enough, the regular expression captures numbers if they are integer or float numbers, in either exponential form or not, written explicitly with the sign or not. KaSim encodes parameters as “`%var: name value`” and use a double dash (“`//`”) and pound (“`#`”) symbols to determine inline comments. Antimony models resemble python structure (as well as many other programming languages) and parameters are defined as simple as “`variable_name = number`”. Line 261 was modified to “`(\w+) = number […]`” (for details see line 259 in github.com/glucksfall/Pleione/blob/master/*Pleione*/tellurium.py)

13.2 WRITE A SYSTEM CALL

Pleione calls simulation software with a system call (optionally wrapped inside a SLURM *sbatch* call). In this step, the variable on line 358 should be modified to match the programming language specification and line 380 to call the simulation software with the

model and other arguments. In the case of Tellurium, the *simulation software* is python itself, and therefore, an ad-hoc script was written to read a model, simulate it, and write results in an appropriate format, script executed with the "`python –m pleione.sim-tellurium model time step output`" system call:

```
import os, sys
import pandas, tellurium

rr = tellurium.loada(sys.argv[1])
rr.integrator = 'rk4'
results = rr.simulate(0, float(sys.argv[2]), int(sys.argv[3]) + 1)

with open(sys.argv[4], 'w') as outfile:
      pandas.DataFrame(data = results,
            columns = results.colnames).fillna(0).to_csv(outfile,
            sep = '\t', index = False)
```

13.3 RESULTS

The Aguilera´s model (Aguilera et al., 2017) was used to test the *Pleione´s* Tellurium interface. In contrast to stochastic simulations, the Tellurium interface simulates ODEs, although the model specifications are reactions. The model was calibrated in similar settings to the description in *Pleione* (Chapter IV) employing the Chi-Square (Equation 5-8) fitness function. Figure 13-1 and 13-2 show the best fit found in two settings that differ only in the number of models per iteration. The initial condition for mRNA synthesis in the linear range 1 to 10, mRNA degradation in the linear range 0.01 to 0.06, protein synthesis in the linear range 0.09 to 0.3, and protein degradation in the linear range 0.01 to 0.06. Parameter mutation was in the ±10% range done with a probability of 20%. Recombination had a 50% probability per parameter. General options were 100 iterations, the recombination of models followed an inverse to the rank probability, and doing 100 or 1000 models per iteration.

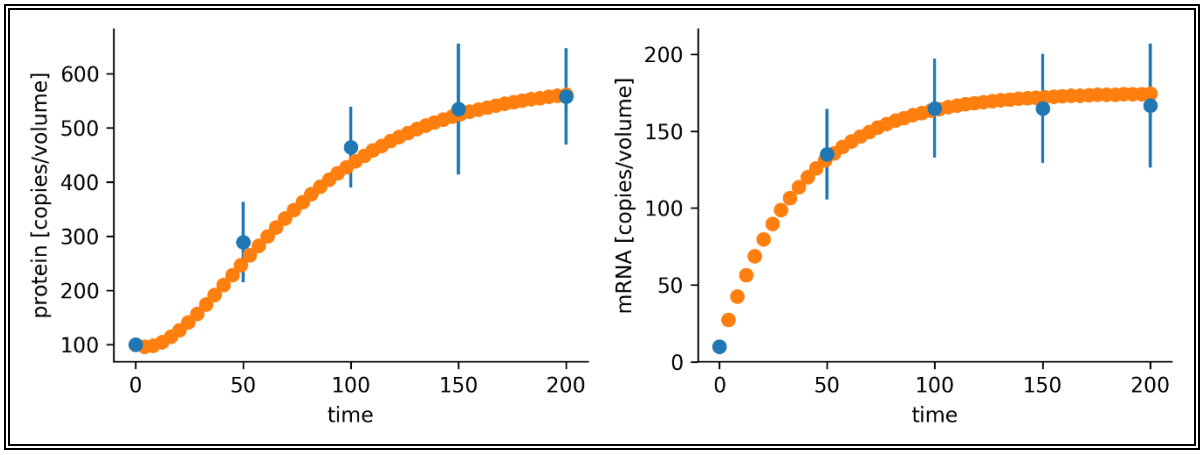


Figure 13-1. Best fit for the Aguilera´s model employing the *Pleione* Tellurium interface. One hundred models per iteration. Bars show one standard deviation of 10 stochastic simulations used as synthetic experimental data.

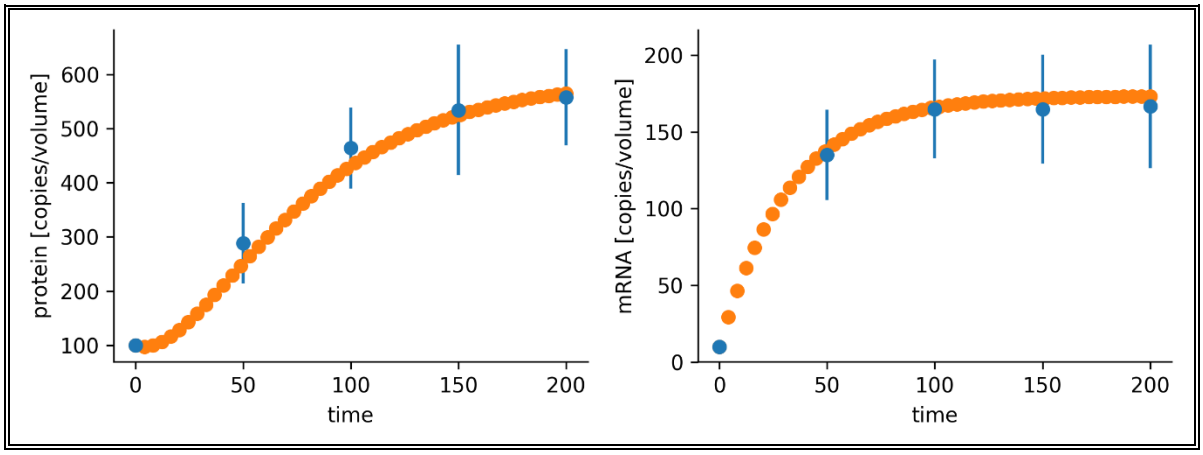


Figure 13-2. Best fit for the Aguilera´s model employing the *Pleione* Tellurium interface. One thousand models per iteration. Bars show one standard deviation of 10 stochastic simulations used as synthetic experimental data.

13.4 OUTLOOK

Pleione was firstly developed to support Rule-Based Models written in *kappa* language and simulated with KaSim only. Further development added support for PISKaS, and the necessary comparison with BioNetFit encouraged the support for BioNetGen and NFsim. The developed method supporting two rule languages and four simulation software was presented in Chapter IV.

Three modeling frameworks –E-cell, Tellurium, and PySB– are written in python and were developed to be unified platforms for model specification, simulation, plotting, and reporting. However, none of them provides calibration methods by itself and the simple two-steps method to extend *Pleione* to support them, as it was done for Tellurium, will allow users to keep their modeling tools instead of changing them for another that provides the required analysis methods.

14. ANNEX CHAPTER V: PARALLEL COMPUTING IN PYTHON

Most of the thesis was possible through the implementation of parallel computing. Although the complexity of problems varied in degree, here will be explained how to implement parallel computing for *embarrassingly* parallel workloads. This class of problems includes those simple enough to be divided into small tasks that are independent of each other, and therefore, do not require distributed shared memory or complex communication between tasks, such as molecular simulation software. We parallelized tasks, for instance, *Pleione* (Chapter IV) simulates models and calculates fitness in parallel, while *Alcyone* (Chapter V) simulates models and calculates each sensitivity index in parallel. In a perfect infrastructure, the total calculation time would be a fraction inversely dependent on the number of available computing cores, or the total time employing single or multiple cores would be equal.

14.1 MULTIPROCESSING

The *multiprocessing* library is part of the python standard library and is a collection of functions and methods for parallel computing. The following example with *multiprocessing* library creates a batch of subprocesses and each of them runs in sequence tied to a single core. The approach contrast with the *threading* standard library as the later create threads, but they are executed one at a time because of the Global Interpreter Lock, even if one thread is executed per core. The GIL is a lock implemented as safety, but make threads no faster than serial execution. The following example will return the square of a number in two steps: first, create a list of numbers and then map the *square* function to a pool of processors. The result is a list and the code could be generalized, as in *Pleione* for the execution of external software through the *subprocess* standard library.

```
import multiprocessing

def sq(x):
	return x**2

lst = range(0, 1001) # a list of 1001 number from 0 to 1000
processes = multiprocessing .cpu_count()-1 # count available cores

with multiprocessing.Pool(processes = processes) as pool:
	data = pool.map(sq, lst) # runs in 0.183s
```

A limit of the former approach is that the *map* method only accepts one iterable as an argument. To employ functions with many arguments effectively, the iterable must be a tuple (an immutable list) of lists:

```
import multiprocessing

def sq(x):
	return x[0]**x[1]

tuple1 = tuple(range(0, 1001))
tuple2 = tuple(range(0, 1001))
tuple = zip(tuple1, tuple2) # tuple = ([tuple1[0], tuple2[0]], …)

processes = multiprocessing .cpu_count()-1 # count available cores

with multiprocessing.Pool(processes = processes) as pool:
	data = pool.map(sq, tuple) # runs in 0.180s
```

14.2 SBATCH

In any case, *multiprocessing* is limited to the physical CPU where python is running. To leverage High-Performance Computing infrastructures such as clusters, federations, and grids, it is needed a specialized software that works as a scheduler. A scheduler controls the use of the resources of those infrastructures by any other software and impersonates the user to run timely any submitted job when resources are available. Resources can be CPUs, cores, memory, GPUs, among others. In the case of the National Laboratory for High-Performance Computing, the scheduler software is SLURM, a common scheduler installed in the most powerful supercomputers. As noted, SLURM impersonates the user to execute a certain software (in a broad understanding, from single commands to complex scripts) and therefore, our simple example requires modification. Here, a helper script will be used to determine the square of a number:

```
import sys

x = sys.args[1]) # sys.args[0] = helper.py
print(x**2)
```

While the main script will be:

```
import re, subprocess

lst = range(0, 1001)

for I in lst:
	cmd = 'sbatch --wrap="python3 helper.py {:s}"'.format(I)
	cmd = re.findall(r'(?:[^\s,"]|"+(?:=|\\.|[^"])*"+)+', cmd)
	out, err = subprocess.Popen(cmd,
			stdout = subprocess.PIPE,
			stderr = subprocess.PIPE).communicate
```

Although the example is similar to the previously presented, the capability of using *sbatch* is limited to software that writes their results to the standard output or files. In the case of *Pleione*, the simulation software write the results to text files that are read later in the execution of the genetic algorithm. In addition, the monitoring of submitted jobs was necessary (checking the job ID in the out variable or errors) before submitting the next job. The limitation could be overcome with specialized software such as Dask (Rocklin, 2015).

14.3 DASK

14.3.1 Single nodes: Dask delayed

Dask delayed is similar to the approach presented previously for multiprocessing, as in fact, it uses the *multiprocessing* standard library. However, it optionally works inside a predefined cluster of workers (an equivalent to the multiprocessing pool). The following example implements Dask delayed to compute the square of a list of numbers:

```
import dask

def sq(x):
	return x**2

results = [] # empty list
lst = range(0, 1001)
for I in lst:
	y = dask.delayed(sq)(I)   # map I to a “delayed” sq function
	results.append(y)

data = dask.compute(*results) # runs in 0.706s
```

14.3.2 Multiple nodes: Dask futures

To use multiple nodes as a cluster of workers, first, it is needed a cluster tied to the scheduler installed in the system. Again, the choice is SLURM and the example is the square function. After a few modifications with respect to Dask *delayed*, a working example is:

```
import dask_jobqueue

def sq(x):
      return x**2

if __name__ == '__main__': # important to create the cluster
      lst = range(0, 1001)

      cluster = dask_jobqueue.SLURMCluster(
                  queue = os.environ['SLURM_JOB_PARTITION'],
                  cores = 1, walltime = '1:00:00', memory = '1GB',
                  local_directory = os.getenv('TMPDIR', '/tmp'))

      cluster.scale(250) # start the client with 250 workers
      client = Client(cluster)
      futures = client.map(sq, lst)
      data = client.gather(futures)   # wait until all evaluations are done
      cluster.scale(0) # stop the cluster
```

An advantage of Dask *delayed* with respect to low-level *sbatch* calls is the ability to gather the results of pure python code without an intermediate file. A disadvantage is shown in shared systems with many users is that the approach is greedy by creating a cluster of fixed size. To explain further, *Pleione* creates and submits one job per simulation and evaluation, and SLURM controls if other user´s jobs gain enough priority to be executed first. In contrast, *Sterope* creates a SLURMCluster and reserves resources in advance, delaying execution of other user´s jobs in favor of *Sterope* execution.

14.4 OUTLOOK

Here was presented four simple methods to develop parallel computing workloads. Although the examples are simple enough evaluations without shared memory or communication between tasks, more complex tasks could be developed from the code exemplified. The user should be able to identify the best fit for their needs, with *multiprocessing* the first choice for physically bound calculations (laptops, desktop PCs, and single nodes in a cluster), and

Dask *delayed* as an improved version, but not necessary in most cases. To leverage multiple nodes (such as clusters, federations, and grids) a scheduler or a Dask cluster must be used for parallel computing of *embarrassingly* parallel workload. Please note that all examples represented costly methods as they create subprocesses before to perform any calculation on them, and the user should evaluate first is the overhead (time to create and close a subprocess) is enough small compared to the calculation time.