Modeling, Computation, and Characterization to Accelerate the Development of Synthetic Gene Circuits in Cell-Free Extracts

Thesis by

Vipul Singhal

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

California Institute of Technology

Pasadena, California

2019

(Defended June 11, 2018)

Vipul Singhal

ORCID: 0000-0003-1670-1824

Dedicated to my Family

Acknowledgments

I want to start by thanking my advisor, Richard Murray. Richard, among the many things that I am grateful for, the biggest has been your patience while I struggled to develop my scientific voice. I also want to thank you for the incredible generosity you have shown with your time, and for always providing me with any resource or advice I needed.

I am grateful to Professors Lea Goentoro, Erik Winfree and Matthew Thomson for their support, and for being on my committee. Erik and Lea, you provided me with insightful feedback, which my research has benefited from greatly. Matt, though we have just met, I am looking forward to picking your brain about what mathematics can contribute to biology.

I want to express my heartfelt thanks to the faculty of the Computation and Neural Systems program, and in particular Thanos Siapas, Erik Winfree, Pietro Perona and Christof Koch, for giving me the chance to pursue my graduate studies as part of this program. I could not have asked for a better experience. I must also thank Eduardo Sontag for always being willing to discuss theory work at a moment's notice. Your enthusiasm is contagious!

I must thank my lab mates, Sam Clamons, Andy Halleran, James Parkin, Mark Prator, Reed McCardell, Shaobin Guo, Anu Thubagere, Andrey Shur, Yong Wu, Yutaka Hori, Zoltan Tuza and Abel Chiao for the fantastic working environment they created. I also want to thank Victoria Hsiao for being a shining example of how to maintain balance during grad school; Enoch Young for the long mathematical and philosophical digressions we took; Zach Sun for all that work we did on the project that never made it here, yet informed so much of my subsequent thinking; William Poole, Sam Clamons and Wolfgang Halter for extremely useful discussions on the identifiability theory work; and Jongmin Kim for being (much) better than Google at answering all sorts of research questions I had during my

early years. It would probably not be too much of an exaggeration to say that discussions with Anandh Swaminathan have probably helped me avoid about half of the bad ideas I was able to avoid, and adopt half of the good ones I adopted. Last but not least, I would like to thank Miryong Yun, our lab manager, whose efficiency and professionalism have been enabling forces to be reckoned with.

A big part of the summers here were spent mentoring students, and I would be remiss if I did not mention them. Flora Meng, Cody Dunn, Ronnie Rodrigues Pereira , Miroslav Gasparek, Anushka Rau, Enrique Amaya, Tiffany Zhou, Pulkit Malik and Anton Frisk, you were an absolute inspiration to work with. Co-mentors Clare Hayes and Ania Baetica: this task would have been impossible without you. Clare, I must also thank you for teaching me lab work, which I imagine was not easy.

Tristan McKinney, Stephen Perry, Howard Hui and Anandh Swaminathan, you guys have been my support system away from home, and the countably infinite adventures we have had will forever be with me. The ski/snowboard crew, Karthik, Eric, Bassam, Anandh, thanks for abandoning me on the mountain during those early days, and forcing me to up my skills. Your hands-off approach worked, and so did the knee pads. I must also mention all the other people who have helped make grad school fun: Brian Brophy and the storytelling class of 2014, OASIS, and the badminton club.

Most importantly, I want to thank my family. Mom, Dad, Niki, Tashu, Neha Didi, and Sanchit Jijaji. Without you, none of this would be possible.

Abstract

Synthetic biology may be defined as an attempt at using engineering principles to design and build novel biological functionalities. An important class of such functionalities involves the bottom up design of genetic networks (or 'circuits') to control cellular behavior. Performing design iterations on these circuits *in vivo* is often a time consuming process. One approach that has been developed to address these long design times is to use *E. coli* cell extracts as simplified circuit prototyping environments. The analogy with similar approaches in engineering, such as prototyping using wind tunnels and breadboards, may be extended by developing accompanying computer aided design tools. In this thesis, we discuss the development of computational and mathematical tools to accelerate circuit prototyping in the TX-TL cell free prototyping platform, and demonstrate some applications of these tools.

We start by discussing the problem of reducing circuit behavior variability between different batches of TX-TL cell extracts. To this end, we demonstrate a model-based methodology for calibrating extract batches, and for using the calibrations to 'correct' the behavior of genetic circuits between batches. We also look at the interaction of this methodology with the phenomenon of parameter non-identifiability, which occurs when the parameter identification inverse problem has multiple solutions. In particular, we derive conditions under which parameter non-identifiability does not hinder our modeling objectives, and subsequently demonstrate the use of such non-identifiable models in performing data variability reduction.

Next, we describe `txtlsim`, a MATLAB® Simbiology® based toolbox for automatically generating models of genetic circuits in TX-TL, and for using these models for part characterization and circuit behavior prediction. Large genetic circuits can have non-negligible

resource usage needs, leading to unintended interactions between circuit nodes arising due to the loading of cellular machinery, transcription factors or other regulatory elements. The usage of consumable resources like nucleotides and amino acids can also have non-trivial effects on complex genetic circuits. These types of effects are handled by the modeling framework of `txtlsim` in a natural way.

We also highlight mcmc_simbio, a smaller toolbox within `txtlsim` for performing concurrent Bayesian parameter inference on Simbiology® models. Concurrent inference here means that a common set of parameters can be identified using data from an ensemble of different circuits and experiments, with each experiment informing a subset of the parameters. The combination of the concurrence feature with the fact that Markov chain Monte Carlo (MCMC) based Bayesian inference methods allow for the direct visualization of parameter non-identifiability enables the design of ensembles of experiments that reduce such non-identifiability.

Finally, we end with a method for performing model order reduction on transcription and translation elongation models while maintaining the ability of these models to track resource consumption. We show that due to their network topology, our models cannot be brought into the two-timescale form of singular perturbation theory when written in species concentration coordinates. We identify a coordinate system in which singular perturbation theory may be applied to chemical reaction networks more naturally, and use this to achieve the desired model reduction.

Published Content and Contributions

X. F. Meng, A.-A. Baetica, V. Singhal, and R. M. Murray. (2017). Recursively constructing analytic expressions for equilibrium distributions of stochastic biochemical reaction networks. *J. R. Soc. Interface* 2017 14 20170157; *10.1098/rsif.2017.0157*. AB and VS conceived the project. XFM, AB and VS performed the mathematical analysis. XFM did the computational experiments. All authors wrote and edited the manuscript.

Z. Sun, J. Kim, V. Singhal, R. M. Murray. (2015). Protein degradation in a TX-TL cell-free expression system using ClpXP protease. *bioRxiv* 019695; *10.1101/019695*. ZS and JK did the experiments. VS and JK did the mathematical and computational modeling. ZS wrote the manuscript, and all authors edited it.

M. Takahashi, J. Chappell, C. A. Hayes, Z. Z. Sun, J. Kim, V. Singhal, K. Spring, S. Al-Khabouri, C. P. Fall, V. Noireaux, R. M. Murray, J. B. Lucks. (2014). Rapidly Characterizing the Fast Dynamics of RNA Genetic Circuitry with Cell-Free Transcription–Translation (TX-TL) Systems. *ACS Synthetic Biology*. 4. *10.1021/sb400206c*. MT and JBL conceived the project, and intitial experiments were done by VS, KS, SAK and CP under the guidance of MT and JBL at the first Cold Spring Harbor Laboratory synthetic Biology summer course. Subsequent experiments were performed by MT with help from JC, CH, ZS and JK. MT wrote the manuscript, with input from all the authors.

Z. Tuza, V. Singhal, J. Kim, and R. M. Murray. (2013). An in silico modeling toolbox for rapid prototyping of circuits in a biomolecular "breadboard" system. *Proceedings of the IEEE Conference on Decision and Control*. 1404-1410. *10.1109/CDC.2013.6760079*. RMM conceived the project. The software was written by ZT and VS. The experiments were performed by ZT and JK. The manuscript was written by VS and ZT, and edited by all the authors.

Contents

Chapter 1

Introduction

The vision of synthetic biology was first laid out as early as 1974 by the Polish geneticist Wacław Szybalski [38]: "the real challenge will start when we enter the synthetic phase of research in our field. We will then devise new control elements and add these new modules to the existing genomes or build up wholly new genomes. This would be a field with an unlimited expansion potential and hardly any limitations to building 'new better control circuits' and 'new better lambdas', or finally other organisms, like a 'new better mouse'". These early words have been shown to be surprisingly close the the mark, as evidenced by the great strides made in synthetic biology in the last eighteen years.

We may define synthetic biology as the attempt at using principles from engineering disciplines to design and build novel functionalities out of biological components. One of the major paradigms within synthetic biology is the bottom up design of novel genetic circuits, defined as networks of gene regulatory elements and other molecular machinery, to control cellular behavior. Examples of genetic circuits include the seminal work on oscillators [15] and bistable modules [23] and the subsequent work on logic gate circuits [45], feedforward circuits [24], spatial control of expression [64], and circuits leveraging non-coding RNAs for regulation [65], among others.

One of they key paradigms in engineering is the use of simplified prototyping environments for testing system designs. Examples include wind tunnels [3] in aeronautics and breadboards [52] for circuit prototyping in electrical engineering. In synthetic biology, this role is increasingly being played by cell-free protein expression systems [60]. Constituted of cell extracts or purified cellular machinery mixed with buffers containing energy

molecules and other resources, these test tube based platforms provide numerous advantages for genetic circuit prototyping. Firstly, since the DNA encoding the genetic circuit is not constrained by the need for DNA replication, restrictions due to plasmid selection markers and antibiotic compatibility are lifted. This allows for the rapid exploration of genetic circuit variants by adding circuit component containing DNA species in different combinations. Furthermore, time-consuming cloning and transformation steps may be bypassed by using linearized DNA in the form of polymerase chain reaction (PCR) products or *de novo* synthesized fragments, which speeds up the design-build-test cycle time considerably. These advantages of combinatorial multiplexing and quicker cycle times are further compounded by using liquid handling robots [55] and microfluidic devices [47], which cell-free gene expression is particularly well suited for.

We consider the cell-free transcription-translation system developed by Noireaux [60], or TX-TL for short, as our prototyping platform. TX-TL uses an *E. coli* cell extract as a source of the gene expression machinery and may be implemented in batch or continuous flow modes [47]. In either mode, the extract used must be harvested by the lysis of *E. coli* cells, and different batches of these extracts can show significant variability in gene expression capacity. In Chapter 2 we introduce the calibration-correction method, a model-based methodology for calibrating extract batches, and correcting circuit behavior based on these calibrations. We find that naive implementations of this methodology have certain performance limitations, and that the method may be improved by considering parameter non-identifiability of the models used, and designing modifications to the method based on these considerations. In Chapter 3 we observe that the analogy to prototyping platforms in traditional engineering disciplines may be extended by using computer aided design (CAD) software to model genetic circuits before any physical prototyping is attempted. To this end, we describe `txtlsim`, a MATLAB® Simbiology® based toolbox for prototyping genetic circuits in the TX-TL cell-free expression system. This chapter also discussed the capabilities of a sub-toolbox for performing Bayesian parameter inference using ensembles of experimental data. In Chapter 4, we discuss the theoretical justifications for certain modeling choices made for modeling transcription and translation reactions in `txtlsim`.

Specifically, we discuss the model order reduction of detailed multi-step elongation models of transcription and translation, while maintaining the ability of the reduced models to track nucleotide and amino acid usage. In the process of performing the model reduction, we show that reaction extent coordinates form a more natural coordinate system compared to the traditionally used species concentration coordinates to bring our model into the two timescale form of singular perturbation theory [39]. Finally, in Chapter 5, we end with some concluding remarks.

Chapter 2

A Model-Based Calibration Methodology for Cell-Free Extract Variability Reduction

2.1 Introduction

Cell-free extracts have been proposed as a potential tool for the rapid prototyping of genetic circuits in synthetic biology [47]. One of the challenges in the development of this technology is that there is significant variability across different batches of extracts, which limits our ability to reliably generalize the results of any one extract. The study performed by Takahashi et al. [65] showed large variation in the constitutive expression of a fluorescent protein between batches, and Hu et al. [33] went one step further, showing that the variability in expression could be mapped to variability in the parameter estimates. Interestingly, Garamella et al. [22] showed minimal variability in constitutive gene expression between four extract batches. However, these batches were produced by the same expert personnel under strictly controlled conditions and supervision (personal communication), and such reproducibility has not been demonstrated in other labs. Furthermore, Garamella et al. did not demonstrate the lack of variability in the behavior of more complex circuits.

Our main goal in this chapter is to describe a method for computationally correcting for the variability between extracts. We frame the reduction in batch-to-batch variability in terms of what we call the *data correction problem*. This involves finding a method for

transforming the behavior of a circuit from a *candidate* extract into what it would look like had it been collected in a *reference* extract. The idea is that whenever data are collected in a batch of extract, they should be transformed into their reference extract version, making them directly comparable with other similarly transformed data.

The data correction problem may be solved by a model-based methodology for calibrating extract batches and subsequently using these calibrations to correct measured genetic circuit behavior. We call this procedure the *calibration-correction* method, after an analogous procedure developed for correcting wind tunnel data in the 1940s [63]. The assumption underlying the method is that there are certain features of extracts that vary from batch to batch, and the variation of these features can be captured as the variation of certain *extract specific parameters* (ESPs). Furthermore, we assume the parameters associated with the circuit do not change when implemented in different extract batches. In general, this assumption should hold for sufficiently fine grained circuit models, becoming more approximate when coarser models are used.

The calibration-correction method involves first performing a set of calibration experiments on both the reference and candidate extracts, and using models corresponding to these experiments to estimate the ESPs associated with each extract. Subsequently, the behavior of a circuit of interest is measured in the candidate extract, and its *circuit specific parameters* (CSPs) are estimated from this data and a corresponding model. This estimation step is performed with the ESPs for the candidate extract fixed at the values obtained at the calibration stage. Finally, the prediction for the circuit behavior in the reference extract is generated using the circuit model with these CSPs, along with the ESPs for the reference extract.

The choice of ESPs and CSPs is hypothesis driven, and must be verified experimentally. In this work we assume that extract specific parameters generally correspond to parameters associated with cellular machinery like RNA polymerases and ribosomes, while the CSPs correspond to parameters like transcription factor dimerization constants, which are associated primarily with circuit parts.

The implementation of this method is complicated by the fact that the parameters

of the models we use to describe biochemical systems are rarely completely identifiable. Roughly stated, parameter non-identifiability refers to the situation when the parameter estimation inverse problem is underdetermined, leading to non-unique solutions. It occurs when the data are not sufficiently informative for the level of detail present in the model. Set based parameter estimation methods like that in [32] or MCMC allow for the computation of equivalence classes [66] of parameter values that fit the model behavior to the data. The fact that the parameter sets obtained are equivalence classes with respect to a model-data set pair simply means that one may sample an *arbitrary* point from the identified parameter set, and it will be a point at which the model fits the data. This notion leads to the question of whether these sets of parameters can be treated as equivalence classes with respect to any method that depends on solving the inverse problem, which will be a major theme in this chapter.

The main conceptual contribution of this chapter will be to show that with respect to the calibration-correction problem, and under some consistency conditions on the parameter sets, these sets can indeed be treated as equivalence classes. The statement of the conditions on the parameter sets will lead to a prescription of how to design the calibration experiments, and in Section 2.6, even lead to a refinement of the calibration-correction method itself.

The framework presented in this work is not limited to correcting the behavior of genetic circuits across cell extracts, and may be applied to the correction of behavior between different cell strains, between the *in vitro* and *in vivo* environments, and even to applications in other engineering disciplines [63]. As such, even though we continue to refer to circuits and extracts, we note that replacing these with *process* and *environment* respectively allows this framework to be used elsewhere.

We start by showing that extracts prepared using the same protocol by different individuals display large variability in gene expression. We then define some notation in Section 2.3. In Section 2.4 we describe the data correction problem and the calibration-correction method in formal terms, and demonstrate the method using a simple example. In Section 2.5, we discuss a set of conditions that are required to hold for the method

to work in general, and discuss the limitations of the method in light of these conditions. Section 2.6 introduces a refinement of the method that improves its performance, and discuss its effect on our example system. In Section 2.7, we demonstrate how the refinement works on artificial data, and end with some concluding remarks in Section 2.8.

2.2 Extracts Display Significant Variability Across Batches

We start with a demonstration of the variability in extract behavior in three batches of extracts. The extracts used were created using the protocol in [60] using the BL21 Rosetta bacterial strains, with cell lysis performed using a French press instead of the bead-beating method described there. Example data in these extracts are shown in Figure 2.1, which shows the results of expressing six constitutive transcriptional units in the three extracts, expressed with linear DNA and GamS protein for protection from nucleases. The buffer used for the experiments in Figure 2.1 was the same. The experiments were performed with five technical repeats, and the mean and standard deviation (shown as solid lines and shaded regions in corresponding colors in the figure) were computed.

We note that the extracts show different levels of expression across different promoters. In particular, eJP shows the highest expression across all the constructs, followed by eVS, and finally eSG.

2.3 Notation and Preliminary Ideas

2.3.1 Experiments, Systems, Models and Parameters

We consider systems $\mathcal{S} = (\mathcal{E}, \mathcal{C})$ described as a combination of an extract $\mathcal{E}$ and a circuit $\mathcal{C}$, and define an experiment $\mathcal{H} = (\mathcal{S}, x_0, \overline{y})$ to be the execution of a system under initial conditions x_0 and output measurements $\overline{y}$. The bar symbol ($\bar{}$) over y is used to denote the fact that the experimental data are assumed to reflect the true behavior of the system, as opposed to the model output trajectories $\hat{y}$ generated at a particular parameter point and set of initial conditions. The definitions of the data correction problem and the

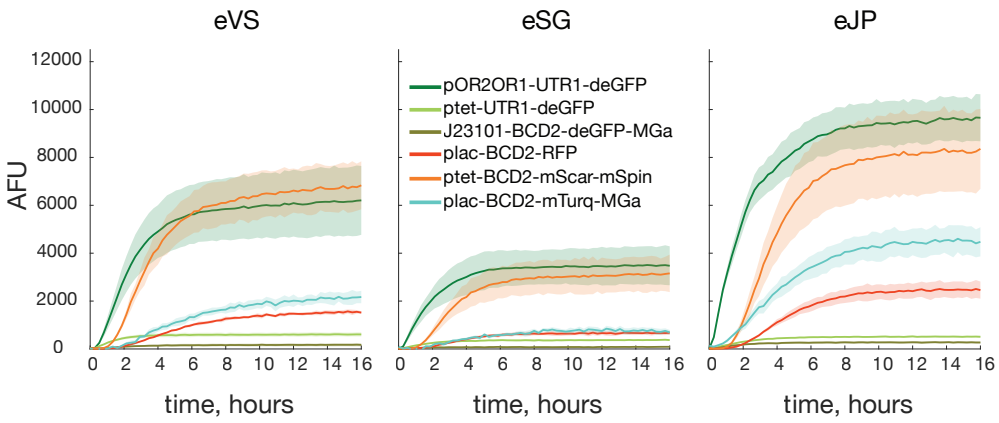

Figure 2.1: There is significant batch-to-batch variability in extracts. We expressed six constitutive reporter constructs (n = 5, technical repeats, shaded region = standard deviation) in three extract batches prepared by different scientists. Each of the constructs was expressed on linear DNA.

calibration-correction method, along with the conditions for the method to work will be given in the context of a model universe, where artificial data will be produced by models with known parameters. Here, the true data variable will also be denoted by $\overline{y}$, and will be distinguished from sample output trajectories $\hat{y}$ and the generic symbol y. For simplicity, we only allow for inputs in the form of initial conditions to the systems, though inputs at other times may be included without significant change to the framework or the mathematical results derived in this chapter.

The parameter vector θ of a model M associated with a given experiment will be partitioned into *extract specific parameter* (ESP) coordinates $e \in \mathbb{R}^{q_E}$, and *circuit specific parameter* (CSP) coordinates $c \in \mathbb{R}^{q_C}$. We do not restrict these parameters to be in the positive orthant, since any positive parameters may be log transformed to exist in the entire space. In subsequent definitions and proofs, we will declutter notation by dropping the explicit specification of the spaces these parameters live in, but these will always be assumed to be as defined here.

The partition of $\theta = (e, c)$ into ESPs and CSPs may be made using the following guidelines: ESPs are parameters associated primarily with species that are present in the system regardless of the the circuit implemented. Examples of ESPs are the concentration of transcriptional and translational machinery, elongation rates for transcription and translation

and the concentration of RNA degradation machinery. CSPs are parameters associated with species that may no longer exist in the system when the circuit is changed. Examples include promoter-transcription factor binding parameters or transcription factor dimerization parameters.

We define an initialized parametrized model with the equations

$$\begin{aligned} \dot{x} &= f(x,\theta), \\ y(\theta,x_0) &= h(x,\theta), \qquad x(0) = x_0(\theta). \end{aligned} \tag{2.1}$$

Here the state vector and its initialization are $x, x_0 \in \mathbb{R}^n_+$ respectively, the solutions are assumed to exist for all $t \geq 0$, the parameter vector symbol is $\theta = (e,c) \in \Omega$, where Ω is the set of all parameter values of interest. The hat symbol (ˆ) over a parameter denoting an estimated value of the parameter, as in $\hat{\theta}$ or $\hat{e}$. We shall reserve the tilde (˜) symbol for miscellaneous purposes, such as picking an arbitrary point in a set while proving an assertion. The output is denoted $y(t,\theta,x_0) \in \mathbb{R}^r$. For simplicity, we do not explicitly model inputs to the system, finite intervals of existence of solutions, or restrictions of the state and parameter spaces to sets smaller than the non-negative orthant. The overall mathematical framework and arguments we develop do not depend on these simplifications, and the general case can be included if needed. The functions f and h are assumed to be analytic vector fields with respect to x in some neighborhood of any attainable x [69]. Lastly, time dependence of the vector fields can be modeled by including t in the state variables. We will use the shorthand $y(\theta,x_0) = M(\theta,x_0)$ to refer to a model in Equation (2.1). We will almost always simplify this notation by dropping the explicit dependence on x_0, which will be assumed to be implicit and appropriately defined in every model. Furthermore, we will often replace θ with (e,c), as in stating $y(e,c) = M(e,c)$ or $M(e,c)$, instead of $y(\theta) = M(\theta)$ where appropriate.

2.3.2 Model Universe

Our analytical results will be stated and proved in a virtual *model universe*, where artificial data $\overline{y}$ is generated using models (denoted by $\overline{M}$) with known nominal parameter values

($\overline{\theta}$). The primary reason for this is that our theoretical results can only be proven when one has access to the true values of the parameters that generated the data.

Furthermore, we will limit ourselves to the case where the models used to estimate parameters from the data are the very models used to generate the data in the first place. In particular, both models will have the same dynamical equations f specifying them. In this sense, the models we use to work with the data are correct, and this assumption will be denoted by $M = \overline{M}$. Working in a model universe, along with this additional correctness assumption, allows us to look at the interaction of non-identifiability with our method in isolation, i.e., without also having to be concerned with whether our models are good models of the system that generated the data. Issues associated with model correctness or the use of increasingly approximate models (that often arise due to model order reduction) are left as future extensions of this work. Furthermore, it is worth explicitly stating that even though the nominal parameter vector used to generate the output trajectory is a single point in the parameter space, the non-identifiability of parameters when their identification is attempted using the output trajectory and the nominal models arises because of the structure of the dynamics function f and that of the output function h. Thus, when stating and proving our main results in Section 2.5, we will always use single points to specify nominal parameter values, even when we can only identify sets of parameter values from the output trajectories.

More precisely, when we refer to a model universe, we identify an experiment $\mathcal{H} = (\mathcal{S}, x_0, \overline{y})$ with a model with known nominal parameters, $\overline{y} = \overline{M}(\overline{\theta}, x_0)$. Here, $\overline{y}$ denotes the measurements from these virtual experiments in our model universe.

2.3.3 Parameter Non-Identifiability

In this subsection, we follow Walter and Lecourtier [69] in defining the notion of parameter non-identifiability.

Definition 1 (Output-Indistinguishable)**.** Let $M(\theta_A)$ be a parametrized model, and let $M(\theta_B)$ be a model with the same structure. $M(\theta_A)$ and $M(\theta_B)$ are said to be *output-indistinguishable*

if

$$\begin{aligned} &\theta_A,\ \theta_B \in \Omega, \\ &y(\theta_A, x_0) = y(\theta_B, x_0) \quad \forall t \geq 0,\ \forall x_0 \in \mathbb{R}^n_+. \end{aligned} \tag{2.2}$$

Definition 2 (Structural Global Identifiability (parameter))**.** The i^{th} coordinate of θ_A, denoted $\theta_{A,i}$, is *structurally globally identifiable* (SGI) if for almost any $\theta_A \in \Omega$, Equation (2.2) has a unique solution for $\theta_{B,i}$.

This means that the i^{th} coordinate of the parameter vector being SGI is equivalent to the set of parameter points θ_A in the parameter space that differ in their i^{th} coordinate and still give output indistinguishable trajectories having measure zero. Stated differently, for an SGI coordinate, output indistinguishable trajectories almost always lead to a unique estimate of the coordinate.

Definition 3 (Structural Global Identifiability (model))**.** The model $M(\theta)$ is called *structurally globally identifiable* (SGI) if all its parameters θ_i, for $i = 1, 2, \ldots, q_E + q_P$, are SGI.

The key point to note is that in the absence of global identifiability, multiple points in the parameter space give rise to the same output behavior. In biological applications, this situation tends to be common due to a limited number of measurements and a large number of state variables. Our main goal is to demonstrate that it is not always necessary to achieve global identifiability for every parameter to achieve a modeling objective such as ours. To this end, we shall consider models with non-SGI parameters, and thus allow e and c to exist in sets of output-indistinguishable parameters, denoted by E and C respectively.

2.3.4 Reference and Candidate Extracts, Calibration and Test Circuits

We define two extracts, the *reference extract* ($\mathcal{E}_1$), and a *candidate extract* ($\mathcal{E}_2$). Let $\mathcal{H}_{i,\text{cal}}$ be an experiment performed with a *calibration circuit*, $\mathcal{C}_{\text{cal}}$, on an extract $\mathcal{E}_i$ to determine the extract specific parameters, and let $\mathcal{H}_{i,\text{test}}$ be an experiment carried out with a *test circuit*, $\mathcal{C}_{\text{test}}$. The goal of the data transformation is to carry out the test circuit experiment in the candidate extract, $\mathcal{H}_{2,\text{test}} = (\mathcal{S}_{2,\text{test}}, x_{0,\text{test}}, \overline{y}_{2,\text{test}})$, and transform output measurements

towards what they would have looked like had they been collected in the reference extract, i.e., transform $\overline{y}_{2,\text{test}}$ into $\hat{y}_{1,\text{test}} \approx \overline{y}_{1,\text{test}}$ where $\overline{y}_{1,\text{test}}$ is the output of $\mathcal{H}_{1,\text{test}}$. Our overall strategy will be to use the estimates of the ESPs obtained at the calibration step in the test circuit models at the correction step. This will require that the models used to describe the calibration and test circuits be similar in the sense that they model the core processes at the same level of abstraction. In the example in Section 2.4.3, protein production is modeled using a single enzymatic reaction.

2.4 A Calibration-Correction Methodology Can be Used to Reduce Extract Variability

In this section, we define the calibration-correction methodology, and demonstrate it using an example. As mentioned in the previous section, we are interested in framing the methodology in a manner that allows for the non-identifiability of model parameters. In Sections 2.4.1 and 2.4.2, we give formal definitions of the data correction problem, the parameter identification operation and the calibration-correction method, and in Section 2.4.3, we demonstrate it on a simple example of correcting the behavior of a tetR mediated repression system.

2.4.1 Framing Extract Variability Reduction as the Data Correction Problem

We begin by framing the variability reduction problem in terms of the data correction problem, defined below and shown schematically in Figure 2.2.

Definition 4 (The Data Correction Problem)**.** Let $\mathcal{H}_{i,\text{test}} = ((\mathcal{E}_i, \mathcal{C}_{\text{test}}), x_{0,\text{test}}, \overline{y}_{i,\text{test}})$, $i = 1, 2$, be the experiments describing the test circuit in the reference and candidate extracts respectively. Assume that we have the freedom to design and perform calibration experiments $\mathcal{H}_{i,\text{cal}}$, $i = 1, 2$, in both the reference and candidate extracts, and collect the resulting data, $\overline{y}_{1,\text{cal}}$ and $\overline{y}_{2,\text{cal}}$. Solving the *data correction problem* involves finding a method that takes as input the tuple $(M_{\text{cal}}, M_{\text{test}}, \overline{y}_{1,\text{cal}}, \overline{y}_{2,\text{cal}}, \overline{y}_{2,\text{test}})$ and returns a trajectory $\hat{y}_{1,\text{test}}$, such that $\hat{y}_{1,\text{test}} = \overline{y}_{1,\text{test}}$.

Remark 1. In general, the data correction problem will only be solvable in the model universe, where the data will be generated as follows. Let $\bar{e}_1$ and $\bar{e}_2$ be the ESPs for $\mathcal{E}_1$ and $\mathcal{E}_2$ respectively. Let $\bar{c}_{\text{cal}}$ and $\bar{c}_{\text{test}}$ be the CSPs for the calibration and test experiments respectively. Then the output data we discuss in the model universe is

$$\overline{y}_{i,\text{cal}} \triangleq \overline{M}_{\text{cal}}(\bar{e}_i, \bar{c}_{\text{cal}}), \tag{2.3}$$

$$\overline{y}_{i,\text{test}} \triangleq \overline{M}_{\text{test}}(\bar{e}_i, \bar{c}_{\text{test}}), \tag{2.4}$$

for $i = 1, 2$. ◇

Remark 2. With real data, the equality $\hat{y}_{1,\text{test}} = \overline{y}_{1,\text{test}}$ in the definition must be replaced with the approximate equality $\hat{y}_{1,\text{test}} \approx \overline{y}_{1,\text{test}}$, or perhaps merely even a requirement of a decrease in the distance (under some metric d) between the predicted and reference trajectories relative to the distance between the reference and candidate extract trajectories, $d(\overline{y}_{1,\text{test}}, \hat{y}_{1,\text{test}}) < d(\overline{y}_{1,\text{test}}, \overline{y}_{2,\text{test}})$. ◇

2.4.2 The Calibration-Correction Method as the Solution to the Data Correction Problem

We begin by describing the parameter identification as a set valued operation on a data-model pair, and subsequently use this as a basis for defining a sequence of steps that together constitute the calibration-correction method.

Definition 5 (Parameter Identification)**.** Let the set Γ be the set of all pairs $(y, M(\theta))$ for which there exists a parameter $\hat{\theta} \in \Omega$ such that $y = M(\hat{\theta})$. Let $\mathcal{P}(\Omega)$ be the power set of Ω. We define the *parameter identification* of the θ coordinates of the model M as an operation $\text{ID}_\theta : \Gamma \to \mathcal{P}(\Omega)$, with $\text{ID}_\theta(y, M(\theta)) = \{\hat{\theta} \in \Omega \mid y = M(\hat{\theta})\}$

In the definition above we have included θ as a subscript to the parameter identification operator to make explicit exactly which parameter coordinates within the model M are being identified. This helps with stylistic uniformity in the usage of this operator, because we also define a *conditional* version for it, where certain parameter coordinates

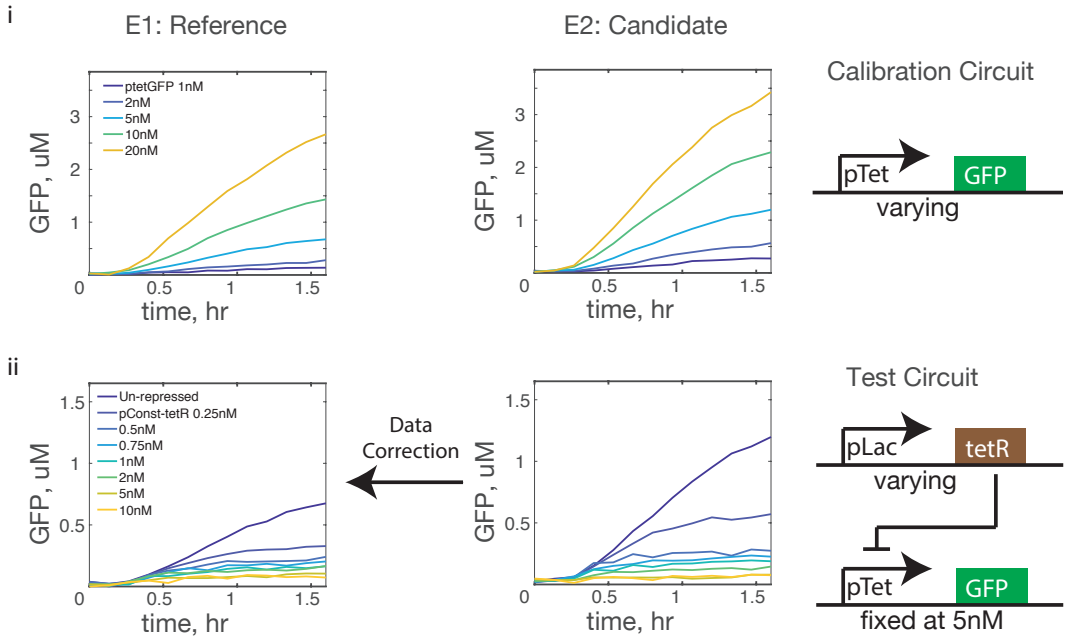

Figure 2.2: The Data Correction Problem. The data correction problem of Definition 4 involves the transformation of the behavior of a genetic circuit, which we refer to as the *test* circuit (ii), from a *candidate* extract to a *reference* extract. We have the freedom to design and implement a set of *calibration* experiments ($\mathcal{H}_{i,\text{cal}}$, $i = 1, 2$) on the two extracts (i), and collect the resulting data ($\overline{y}_{1,\text{cal}}$ and $\overline{y}_{2,\text{cal}}$). In this figure, the test circuit is the repression of the pTet promoter by constitutively expressed tetR transcription factor. For the calibration experiments, we will demonstrate that simply using constitutive GFP expression is sufficient to transform the data when the parameter non-identifiability is addressed using the tools developed in this work.

are held at fixed values, and sets of values of the remaining coordinates are estimated. This is discussed in Remark 3 next.

Remark 3. We define two minor modifications to the use of the ID_θ operator. First, we allow for the identification of a subset of parameter coordinates, such as the circuit specific parameters c, with values for the remaining parameter coordinates fixed at a specified value. We use the notation $\mathrm{ID}_{c|e=\hat{e}}(y, M(e,c))$ or $\mathrm{ID}_{c|e=\tilde{e}}(y, M(e,c))$ to describe this version of the operator. Here, the hat or tilde is used to denote the fact that the ESP coordinates are set to a specific value ($\hat{e}$ or $\tilde{e}$), while the set of values for the remaining parameter coordinates (the CSP coordinates in this example) is free to be estimated by the operator. We also note that in this case, the domain and codomain of this operator are slightly different from those shown in the definition above. Indeed, the domain for the $\mathrm{ID}_{c|e=\tilde{e}}(y, M(e,c))$ case is the set of all pairs $(y, M(\tilde{e},c))$ for which there exists a parameter $\hat{c}$ such that $y = M(\tilde{e},\hat{c})$. Similarly, the codomain can be $\mathcal{P}(\mathbb{R}^{q_c})$ or $\mathcal{P}(\mathrm{proj}_c\,\Omega)$, where proj_c denotes the projection operator from the full coordinate space to the CSP coordinates, c.

In the rest of this chapter, we will simplify notation by shortening $\mathrm{ID}_{c|e=\tilde{e}}(y, M(e,c))$ to $\mathrm{ID}_c(y, M(\tilde{e},c))$. This should not cause any ambiguity, since both the c subscript and the tilde over the ESP coordinates e are being used to denote the fact that we are estimating the CSP coordinates c, while holding the ESP coordinates at $\tilde{e}$. In both cases, we call this the conditional ID operator.

A second method of identifying values for some subset of parameter coordinates (say c once again) is to identify values over all the parameter coordinates, and then to project the resulting set down to the coordinates of interest. An example of the notation we will use to describe this operation is $\mathrm{proj}_c\,\mathrm{ID}(y, M(\theta))$, where $\theta = (e,c)$ and the ID operator works on the full parameter vector θ, as defined in Definition 5. ◇

Next, we define the calibration-correction method as a sequence of steps involving parameter identification and prediction. Along with stating each step of the method in terms of single parameter points identified or used, and single trajectories generated, we also give descriptions of the sets of all such points and trajectories. The definitions of these sets allow for the investigation of the idea of whether the non-identifiable parameter

sets can be treated as equivalence classes with respect to this method. In particular, in Section 2.5, we will derive a set of conditions for the method to work when *arbitrary* points in the parameter sets are picked at the various stages of the method. Figure 2.3 shows a schematic description of this procedure.

Definition 6 (The Calibration-Correction Method)**.** Consider the data correction problem in the context of the model universe. We define the *calibration-correction method* as a sequence of steps that takes as input the tuple $(M_{\text{cal}}, M_{\text{test}}, \overline{y}_{1,\text{cal}}, \overline{y}_{2,\text{cal}}, \overline{y}_{2,\text{test}})$ and returns a prediction of the behavior of the test circuit in the reference extract, denoted by $\hat{y}_{1,\text{test}}$. The steps are:

1. *Calibration Step.* Find extract specific parameters that fit the calibration model to corresponding data for each of the extracts, while sharing a common estimate of the circuit specific parameter vector. I.e., find $\hat{e}_{1,\text{cal}}$ and $\hat{e}_{2,\text{cal}}$ such that the tuple $(\hat{e}_{1,\text{cal}}, \hat{e}_{2,\text{cal}}, \hat{c}_{\text{cal}})$ satisfies $\overline{y}_{1,\text{cal}} = M_{\text{cal}}(\hat{e}_{1,\text{cal}}, \hat{c}_{\text{cal}})$ and $\overline{y}_{2,\text{cal}} = M_{\text{cal}}(\hat{e}_{2,\text{cal}}, \hat{c}_{\text{cal}})$ for some $\hat{c}_{\text{cal}}$. Note that the set of all such ESP points is constructed as follows: first, the set of all valid $(\hat{e}_{1,\text{cal}}, \hat{e}_{2,\text{cal}}, \hat{c}_{\text{cal}})$ tuples is defined as

$$\tilde{\Theta}_{\text{cal}} \triangleq \Big\{ (e_1, e_2, c) \,\Big|\, \overline{y}_{i,\text{cal}} = M_{\text{cal}}(e_i, c),\, i = 1, 2 \Big\},$$

   and then, the ESP sets are defined as

$$E_{i,\text{cal}} \triangleq \text{proj}_{e_i} \tilde{\Theta}_{\text{cal}}, \qquad i = 1, 2. \tag{2.5}$$

2. *Correction Step One.* Identify circuit specific parameters of the test circuit in the candidate extract while holding the extract specific parameters at the value estimated at the previous step. I.e., find $\hat{c}_{2,\text{test}}$ such that $\overline{y}_{2,\text{test}} = M_{\text{test}}(\hat{e}_{2,\text{cal}}, \hat{c}_{2,\text{test}})$. Note that the set of all such points is given by

$$C'_{2,\text{test}} \triangleq \bigcup_{\hat{e} \in E_{2,\text{cal}}} \text{ID}_{c|e=\hat{e}} \Big(\overline{y}_{2,\text{test}}, M_{\text{test}}\big(e, c\big) \Big), \tag{2.6}$$

   where we have used the full notation for the conditional ID operator.

3. *Correction Step Two.* Predict test circuit behavior in the reference extract using the circuit specific parameters estimated in the first correction step, and extract specific parameters estimated in the calibration step. I.e., generate the prediction $\hat{y}_{1,\text{test}} = M_{\text{test}}(\hat{e}_{1,\text{cal}}, \hat{c}_{2,\text{test}})$. Note that the set of all predictions that can be generated is given by

$$Y_1 \triangleq \bigcup_{\hat{e}\in E_{1,\text{cal}}} \bigcup_{\hat{c}\in C'_{2,\text{test}}} \hat{y}_1(\hat{e},\hat{c}), \tag{2.7}$$

where individual predictions of the reference trajectories are given by $\hat{y}_1(\hat{e},\hat{c}) = M_{\text{test}}(\hat{e},\hat{c})$.

Remark 4. If the ESP sets from the calibration step were to be estimated, the version of the calibration step defined above would be straightforward to implement computationally. This is because the estimation of $\tilde{\Theta}_{\text{cal}}$ can be done in a single step (see Chapter 3, Section 3.5 for concurrent parameter inference tools), and the sets $E_{i,\text{cal}}$, for $i = 1, 2$, are simple projections computed from the estimated set.

We also give an equivalent, but less computationally tractable definition here that allows for the estimation of the parameters for the two extracts separately, followed by a restriction procedure that enforces agreement between the CSPs estimated in the two extracts. We start with estimating the joint ESP-CSP sets for individual extracts, $\Theta_{i,\text{cal}} \triangleq \text{ID}_\theta\left(\overline{y}_{i,\text{cal}}, M_{\text{cal}}(\theta)\right)$, $i = 1, 2$, and then compute the set of CSPs where these agree, $C_{\text{cal}} \triangleq \text{proj}_c\,\Theta_{1,\text{cal}} \cap \text{proj}_c\,\Theta_{2,\text{cal}}$. Finally, the ESP sets are generated by restricting the $\Theta_{i,\text{cal}}$ by C_{cal},

$$E_{i,\text{cal}} \triangleq \left\{ e \;\middle|\; \exists c \in C_{\text{cal}} : (e,c) \in \Theta_{i,\text{cal}} \right\}, \qquad i = 1, 2.$$

The fact that the sets $\Theta_{i,\text{cal}}$, $i = 1, 2$, are estimated separately can be useful in cases where the dimension of the spaces e and c live in (i.e., q_E and q_C) are large enough that estimating $\tilde{\Theta}_{\text{cal}} \in \mathbb{R}^{2q_E+q_C}$ might be much more difficult compared to $\Theta_{i,\text{cal}} \in \mathbb{R}^{q_E+q_C}$. The tradeoff here is that intersections and restrictions of sets represented by point clouds can be computationally difficult. Finally, the lemma in Appendix 2.A establishes the equivalence of this definition to the one given in Definition 6 (Equation 2.5). ◇

Remark 5. Note that the set $C'_{2,\text{test}}$ is a subset of the larger set $C_{2,\text{test}} \triangleq \text{proj}_c \, \text{ID}_\theta(\overline{y}_{2,\text{test}}, M_{\text{test}})$. Indeed, $C'_{2,\text{test}}$ is obtained from $C_{2,\text{test}}$ by only keeping the points whose corresponding e coordinate values were in the calibration set $E_{2,\text{cal}}$. We use $C'_{2,\text{test}}$ because in the first correction step, we identify c only after fixing the value of e to an arbitrary point within $E_{2,\text{cal}}$. ◇

Remark 6. We can define two *failure conditions* for the calibration-correction method that will be useful in deriving the main theoretical results of this chapter. Both the conditions must be avoided for the calibration-correction method to solve the data correction problem.

The first condition (FC1) occurs if a parameter identification step is attempted when no parameter exists such that the model fits the data. This means that the data-model pair (y, M) under consideration is not in the domain, Γ, of the operator ID. For example, in the first correction step, if $\hat{e}_{2,\text{cal}}$ is such that there is no $\tilde{c}$ that satisfies $\overline{y}_{2,\text{test}} = M_{\text{test}}(\hat{e}_{2,\text{cal}}, \tilde{c})$, then the parameter estimation step fails at this point. In terms of Equation (2.6), this failure condition occurs if it occurs for *any* point e in $E_{2,\text{cal}}$.

The second failure condition (FC2) occurs if correction step two is able to produce a trajectory not equal to the true trajectory, i.e., $\hat{y}_{1,\text{test}} \neq \overline{y}_{1,\text{test}}$. In terms of the set Y_1 defined in Equation (2.7), this means that Y_1 contains at least one element that is not equal to $\overline{y}_{1,\text{test}}$. ◇

Before we state and prove the conditions that need to hold for this method to work, we illustrate its use with a simple example.

2.4.3 A Simple Example

To illustrate the calibration-correction method, we use tetR mediated repression as our test circuit experiment, constitutive GFP expression as our calibration circuit experiment, and model protein production directly from DNA using an enzymatic reaction. Figure 2.4 shows the data, and the results from the calibration and correction steps. The test circuit experiment involves fixing the tetR repressible ptet-UTR1-deGFP DNA at $5\,\text{nM}$, and varying the constitutive tetR DNA concentration from 0–$0.75\,\text{nM}$. The calibration experiment

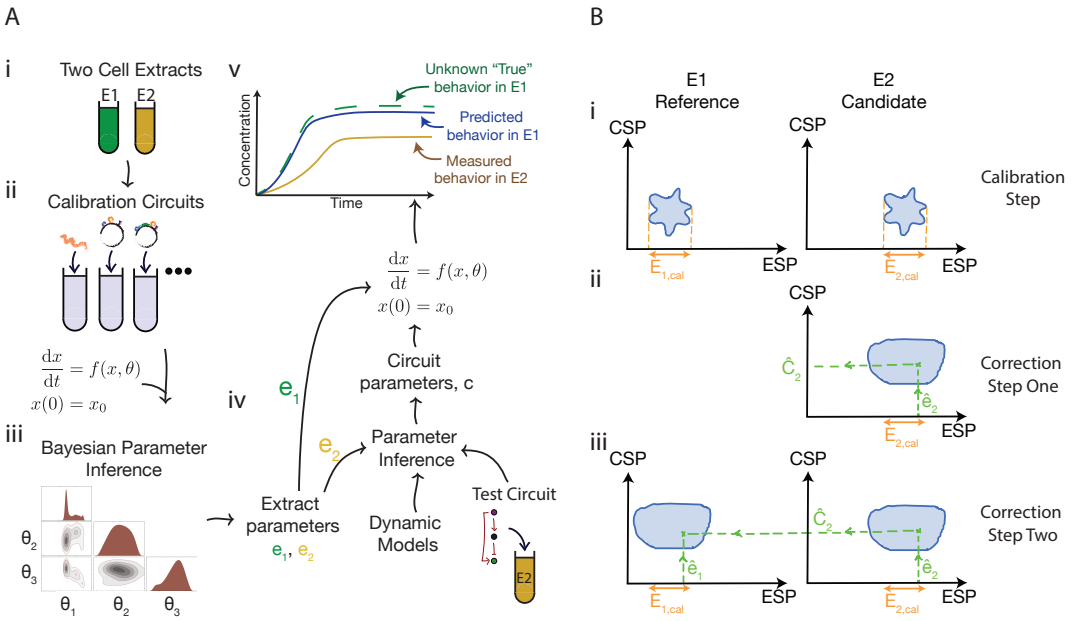

Figure 2.3: (A) Schematic describing the calibration-correction method of Definition 6. Calibration Step: Given two cell extracts (Ai), a reference extract $\mathcal{E}_1$ and a candidate extract $\mathcal{E}_2$, perform a set of calibration experiments (Aii) on each of the two extracts, and collect the corresponding data. Use parametrized models describing these experiments, along with parameter estimation tools (Aiii), to estimate the extract specific parameters (e_1 and e_2) as described in the calibration step of Definition 6. (Aiv) Correction Step One: Collect data for a test circuit in $\mathcal{E}_2$. The goal is to transform this into what it would look like had it been collected in $\mathcal{E}_1$. Use a model of the test circuit to estimate the CSPs for this circuit with the ESPs fixed to a value obtained for $\mathcal{E}_2$'s ESPs in the calibration step. The model used here must be at a similar level of detail as the models used for the calibration step. This allows for the ESPs estimated at that step to be used here. (Av) Correction Step Two: Finally, plug in the ESPs for $\mathcal{E}_1$ and the CSPs just estimated into the test circuit model to generate the desired transformed data (blue solid line) in the time-course schematic shown. (B) Interplay of parameter non-identifiability with the calibration-correction method. When the parameter estimation procedure returns sets of parameters that all fit the model to the data, we say that the parameters of the model are non-identifiable. (Continued below)

Figure 2.3: (Continued from above) The calibration-correction method in the presence of non-identifiability involves treating the sets of estimated parameters as equivalence classes, allowing for arbitrary points in the sets to be used for the purposes of the method. (Bi) Calibration step: The calibration step is still performed with the CSPs shared across the two extracts, and ESPs estimated individually, resulting in a set of points ((e_1, e_2, c)). In the schematic, the projections of this set onto the (e_1, c) and (e_2, c) coordinate axes are shown as the shaded regions. The ESPs that are obtained at the calibration step are now sets in the ESP coordinates, $E_{1,\text{cal}}$ and $E_{2,\text{cal}}$, corresponding to the projection of the sets in the full parameter space onto the e_1 and e_2 coordinate axes. (Bii) The first correction step involves picking an arbitrary point in the set $E_{2,\text{cal}}$ and estimating the set of CSPs that fit the test circuit model to the data at this point, and then treating this set as an equivalence class in turn and picking an arbitrary point from this set. The shaded region denotes the set of all parameters in the full coordinate space that fit the test circuit to the data. (Biii) Correction Step Two: An arbitrary point from the ESP set for the reference extract, $E_{1,\text{cal}}$, is picked, along with the arbitrarily picked point from the first correction step, and used to parameterize the test circuit model and generate the desired correction.

involves varying this reporter construct in isolation from $1\text{–}20\,\text{nM}$. The calibration circuit M_{cal} is modeled as

$$\mathrm{D_G} + \mathrm{Enz} \underset{k_{rG}}{\overset{k_{fG}}{\rightleftharpoons}} \mathrm{D_G{:}Enz} \xrightarrow{k_c} \mathrm{D_G} + \mathrm{Enz} + \mathrm{G}, \tag{2.8}$$

where $\mathrm{D_G}$ is the GFP DNA, Enz is an enzyme species denoting a lumped description of the machinery that implements the conversion of DNA into protein, and G is the GFP protein. The test circuit is modeled using the equations M_{test},

$$\begin{aligned}
\mathrm{D_T} + \mathrm{Enz} &\underset{k_{rT}}{\overset{k_{fT}}{\rightleftharpoons}} \mathrm{D_T{:}Enz} \xrightarrow{k_c} \mathrm{D_T} + \mathrm{Enz} + \mathrm{T}, \\
\mathrm{D_G} + \mathrm{Enz} &\underset{k_{rG}}{\overset{k_{fG}}{\rightleftharpoons}} \mathrm{D_G{:}Enz} \xrightarrow{k_c} \mathrm{D_G} + \mathrm{Enz} + \mathrm{G}, \\
2\mathrm{T} &\underset{k_{r,dim}}{\overset{k_{f,dim}}{\rightleftharpoons}} \mathrm{T_2}, \\
\mathrm{D_G} + \mathrm{T_2} &\underset{k_{r,rep}}{\overset{k_{f,rep}}{\rightleftharpoons}} \mathrm{D_G{:}T_2},
\end{aligned} \tag{2.9}$$

where $\mathrm{D_T}$ is the DNA that codes for the tetR repressor protein (under the control of a constitutive promoter), T and $\mathrm{T_2}$ are the tetR protein monomer and dimer respectively. Note that the tetR dimer sequesters the GFP expressing DNA, $\mathrm{D_G}$, and in doing so, represses GFP.

Recall that the models used for the circuits at the calibration and correction stages have to be at the same levels of modeling to allow for ESPs estimated at the calibration stage to be used in the test model at the correction stage. In the example above, both the models produce protein using a single step enzymatic reaction, with the parameters θ partitioned into ESPs $e = (k_c, [\mathrm{Enz}]_0)$ and the CSPs $c = (k_{fT}, k_{rT}, k_{fG}, k_{rG}, k_{f,dim}, k_{r,dim}, k_{f,rep}, k_{r,rep})$. Here, $[\mathrm{Enz}]_0$ denotes the initial concentration of Enz. The main reason for picking this simple model for protein expression is that at this level of modeling, the number of parameters is small enough that the theoretical conditions we discuss can be visualized in three dimensions before being generalized to models with higher dimensional parameter spaces.

Continuing with our example, we next perform the calibration step of the method using an MCMC method (see Section 3.5) to estimate the posterior distribution of the parameters given the data, $\mathbb{P}\left(e_1, e_2, c_{cal} \mid \overline{y}_{1,cal}, \overline{y}_{2,cal}, M_{cal}\right)$. We note that the calibration circuit CSPs are estimated jointly over the two extract batches, i.e., the distribution above is that of the vector $(e_1, e_2, c_{\mathrm{cal}})$ such that the model $M_{\mathrm{cal}}(e_i, c_{\mathrm{cal}})$ fits the data $y_{i,\mathrm{cal}}$ simultaneously for both values of $i = 1, 2$. Figure 2.4 shows the model fits from this step, and the corner-plots showing pairwise projections of the joint parameter distributions of the (e_i, c_{cal}) coordinates for both $\mathcal{E}_1$ and $\mathcal{E}_2$.

To perform the first correction step we fixed the candidate extract ESP value at a single point drawn from $E_{2,\mathrm{cal}}$ and estimated $C_{2,\mathrm{test}}$. The model fits are shown in Figure 2.4. Fixing the ESP value to a point in $E_{1,\mathrm{cal}}$ and drawing 500 points from $C_{2,\mathrm{test}}$ to generate the corrected trajectories implements correction step two, and the results are shown in the third column in Figure 2.4 (iii).

To conclude this section, we compute the degree of variability reduction achieved by our procedure on this test circuit data. We define two metrics to measure the variability reduction. The first metric measures the the ratio of the sum of the deviations between the corrected and reference trajectories to the sum of the deviations between the original reference and candidate trajectories. Formally, we write the metric as,

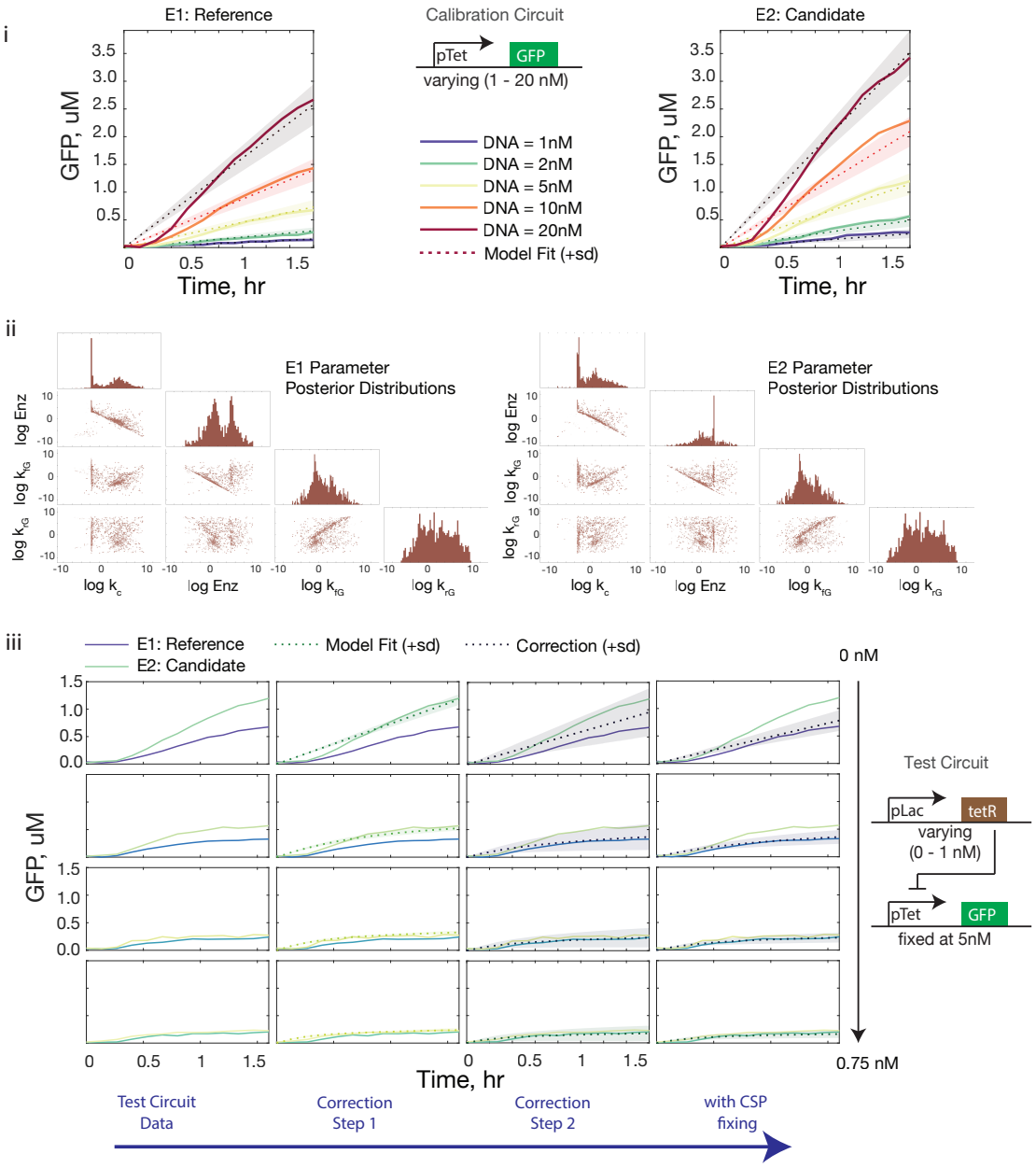

Figure 2.4: Demonstration of the calibration-correction method on the experimental data described in Figure 2.2. The calibration data are the constitutive expression of the pTet promoter at various DNA concentrations. The test data are the repression of a fixed concentration of the pTet promoter with varying concentrations of repressor DNA. (Continued below)

Figure 2.4: (continued from above) (i) Model fits to the calibration dataset using a Bayesian parameter inference approach. The joint parameter posterior distributions obtained using this approach are used as proxies for the parameter sets described in the main text. The model in Equation (2.8) is used with the calibration data to infer the joint posterior distributions of the ESPs and CSPs, denoted by the set $\tilde{\Theta}_{\text{cal}}$ in Definition 6. The solid lines depict the experimental data trajectories, and the dashed lines and shaded regions denote the means and standard deviations (resp.) of trajectories simulated using points drawn from the posterior distribution. (ii) Parameter posterior distributions from the calibration step. The ESPs are $e = (k_c, [\mathrm{Enz}]_0)$, where we let $[\mathrm{Enz}]_0$ denote the initial amount of the Enz species. The CSPs are $c = (k_{fG}, k_{rG})$, the binding-unbinding rate constants of the DNA to the Enz species. The parameter inference was performed with the CSPs shared across the extract, i.e., in a joint space with points (e_1, e_2, c). Here we show the posterior distributions of the parameter vector (e_i, c) for the two extracts $i = 1, 2$. The distributions are shown as corner plots of the pairwise projections on the off diagonal plots and the marginal distributions on the diagonal. (iii) The two correction steps on the tetR repression test circuit data. The reporter DNA is fixed at $5\,\mathrm{nM}$, and the repressor DNA varies from from 0–$0.75\,\mathrm{nM}$ down the rows (0, 0.25, 0.5 and $0.75\,\mathrm{nM}$). The solid lines in all the plots are experimental data, the dashed lines and shaded regions are the mean and standard deviations of simulated trajectories corresponding to parameters drawn from the respective parameter sets as described by the calibration-correction method in Definition 6. The first three columns, starting from the left, are: test circuit data in the two extracts, correction step one, where the model in Equation (2.9) is fit to the candidate extract data, and the second correction step. We see that the model fits the candidate extract data quite well in the first correction step, and correction step two is able to move the model prediction trajectories towards the reference extract data trajectories at all repressor DNA concentrations. The standard deviation of the predicted trajectories in the third column is much larger than that of the fitted trajectories in the second column. In Sections 2.6 and 2.7, we discuss ESP-CSP *covariation* as a possible reason for this type of increase in the standard deviation, and propose a modification to the calibration-correction method, called *CSP fixing*, that addresses this type of covariation. The fourth column shows the result of applying this modified version of the method to this data, and shows that the standard deviations tighten up considerably.

$$R_1 = \frac{\sum_{i=1}^{n_{IC}} \|\hat{y}_{1,\text{test}}(x_{0,i}) - \overline{y}_{1,\text{test}}(x_{0,i})\|_2}{\sum_{i=1}^{n_{IC}} \|\overline{y}_{2,\text{test}}(x_{0,i}) - \overline{y}_{1,\text{test}}(x_{0,i})\|_2}, \tag{2.10}$$

where the sum is taken over the n_{IC} experimental conditions (which, in this case, are the four tetR DNA concentrations). We have added an argument $\left(x_{0,i}\right)$ to the output trajectory variable y to reflect this fact explicitly. For our dataset, we compute the value of this metric to be $R_1 = 0.42$.

The second metric computes, for each of the n_{IC} initial conditions, the ratio of the deviation between the corrected trajectory and the reference extract trajectory, and the deviation between the original candidate extract trajectory and the reference extract trajectory. It then takes the mean of these individual ratios to give a score for the average correction. Formally, it is defined as

$$R_2 = \frac{1}{n_{IC}} \sum_{i=1}^{n_{IC}} \frac{\|\hat{y}_{1,\text{test}}(x_{0,i}) - \overline{y}_{1,\text{test}}(x_{0,i})\|_2}{\|\overline{y}_{2,\text{test}}(x_{0,i}) - \overline{y}_{1,\text{test}}(x_{0,i})\|_2}, \tag{2.11}$$

and gives a value of 0.48 when computed for our dataset.

2.5 Identifiability Conditions

In this section, we show that the SGI property is not necessary for the data correction problem to be solved by the calibration-correction method. This will be stated as a corollary of the main result of this section (Theorem 1), which gives conditions on the sets of non-identifiable parameters obtained during the calibration-correction method such that the method solves the data correction problem.

The key insight underlying the theory developed in this chapter is that since the correction only needs to be applied to the output trajectories, and not to the full state vector trajectories, we do not need the parameters to be fully identifiable. Roughly speaking, this is due to the fact that the non-identifiability occurs because the output trajectories are not informative enough to identify the parameters to a degree that allows the state tra-

jectories to be reconstructed. Indeed, the identified parameter sets only contain enough information to reconstruct the outputs. However, since we are only attempting to correct the outputs, and not the state trajectories, the method continues to work in the presence of the non-identifiability.

This idea of using parameters estimated using only the outputs to in turn correct only the output behavior, and not the entire state vector trajectories is closely related to the idea of the sets of output-indistinguishable parameters being equivalence classes with respect to the inputs and outputs of a model. While these sets may be equivalence classes with respect to individual estimations performed using model data pairs, some additional restrictions need to be placed on these sets if they are to be treated as equivalence classes with respect to the calibration-correction method. The main goal of this section will be to derive these conditions.

Theorem 1 (Parameter consistency)**.** *Consider the data correction problem (Definition 4) in the model universe, i.e., when the experimental data are generated by nominal parametrized initialized models, as described in Remark 1. Furthermore, consider the calibration-correction method of Definition 6, and the sets* $\tilde{\Theta}_{\text{cal}}$, $E_{1,\text{cal}}$, $E_{2,\text{cal}}$ *and* $C'_{2,\text{test}}$ *as defined there. Define* $\Theta_{i,\text{test}} \triangleq \text{ID}_\theta\left(\overline{y}_{i,\text{test}}, \overline{M}_{\text{test}}(\theta)\right)$ *for* $i = 1, 2$*. Then, the conditions,*

$$\tilde{\Theta}_{\text{cal}} \neq \emptyset, \tag{2.12}$$

$$E_{2,\text{cal}} \subseteq \text{proj}_e\, \Theta_{2,\text{test}}, \tag{2.13}$$

$$E_{1,\text{cal}} \times C'_{2,\text{test}} \subseteq \Theta_{1,\text{test}}, \tag{2.14}$$

are necessary and sufficient for the calibration-correction method to solve the data correction problem.

Proof. We note that solving the data correction problem using the calibration-correction method simply involves avoiding the failure conditions FC1 and FC2 described in Remark 6. Avoiding FC1 wherever it may occur ensures that the method can be implemented in the first place, and avoiding FC2 means that the method returns the desired result. Thus, we must show that the conditions (2.12-2.14) are necessary and sufficient for avoiding FC1 and

FC2.

The necessity of condition (2.12) follows from the fact that if $\tilde{\Theta}_{\text{cal}} = \emptyset$, then there does not exist a vector (e_1, e_2, c) such that $\overline{y}_{i,\text{cal}} = M_{\text{cal}}(e_i, c)$ for $i = 1, 2$, leading to FC1 being met at the calibration step. While not needed for the proof, we note in passing that in the model universe, where $M_{\text{cal}}(\theta) = \overline{M}_{\text{cal}}(\overline{\theta})$ and $\overline{y}_{i,\text{cal}} = \overline{M}_{\text{cal}}(\overline{e}, \overline{c})$, condition (2.12) always holds.

Next, we prove the necessity of $E_{2,\text{cal}} \subseteq E_{2,\text{test}}$, where $E_{2,\text{test}} \triangleq \text{proj}_e \Theta_{2,\text{test}}$. Assume that there exists an $\tilde{e} \in E_{2,\text{cal}}$ such that $\tilde{e} \notin E_{2,\text{test}}$. Thus, there does not exist a $\tilde{c}$ such that $M_{\text{test}}((\tilde{e}, \tilde{c})) = \overline{y}_{2,\text{test}}$. Since the operator $\text{ID}_{c|e=\tilde{e}}$ is only defined on the set $\{(y, M) \mid \exists c : M((\tilde{e}, c)) = y\}$, we see that the map $\text{ID}_{c|e=\tilde{e}}(\overline{y}_{2,\text{test}}, M_{\text{test}}(e, c))$ is not well defined, leading to FC1 at the first correction step.

We prove the necessity of condition (2.14) as follows. Assume that there exists a $(\tilde{e}, \tilde{c}) \in E_{1,\text{cal}} \times C'_{2,\text{test}}$ such that $(\tilde{e}, \tilde{c}) \notin \Theta_{1,\text{test}}$. Since we use points $\hat{e} \in E_{1,\text{cal}}$ and $\hat{c} \in C'_{2,\text{test}}$ to generate the prediction $\hat{y}_{1,\text{test}}$ in the second correction step, it is possible that $\hat{e} = \tilde{e}$ and $\hat{c} = \tilde{c}$. Furthermore, since $\Theta_{1,\text{test}}$ is the set of all points (e, c) that give the correct trajectory $\overline{y}_{1,\text{test}}$, we have the possibility that $\hat{y}_{1,\text{test}} \neq \overline{y}_{1,\text{test}}$. This is the second failure condition.

Finally, sufficiency is a simple consequence of the fact that conditions (2.12-2.14) address both the points in the method where FC1 could be met, and the point in the method where FC2 could occur. Explicitly, condition (2.12) allows the calibration step to avoid FC1, condition (2.13) allows correction step one to avoid FC1, since it implies that for all $\tilde{e} \in E_{2,\text{cal}}$, there exists a $\tilde{c}$ such that $(\tilde{e}, \tilde{c}) \in \Theta_{2,\text{test}}$. Condition (2.14) enables correction step two to avoid FC2, since it implies that for all $\tilde{e} \in E_{1,\text{cal}}$ and for all $\tilde{c} \in C'_{2,\text{test}}$ we have that $\overline{y}_{1,\text{test}} = M_{\text{test}}(\tilde{e}, \tilde{c})$, implying that the set of all possible predicted trajectories only has the correct trajectory in it, $Y_1 = \{\overline{y}_{1,\text{test}}\}$. □

Remark 7. We can give some physical interpretations of the conditions (2.12-2.14). To do this, we first note that condition (2.14) implies (see Lemma 3 in Appendix 2.B)

$$E_{1,\text{cal}} \subseteq \text{proj}_e \Theta_{1,\text{test}}, \tag{2.15}$$

$$C'_{2,\text{test}} \subseteq C'_{1,\text{test}}, \tag{2.16}$$

where $C'_{1,\text{test}}$ is defined in a similar way to $C'_{2,\text{test}}$,

$$C'_{1,\text{test}} \triangleq \bigcup_{\hat{e}\in E_{1,\text{cal}}} \text{ID}_{c|e=\hat{e}}\left(\overline{y}_{1,\text{test}}, M_{\text{test}}(e,c)\right).$$

Condition (2.12) and (2.15) may be interpreted to mean that the calibration experiments must be more informative about the ESPs than the test circuit experiments. This follows from the fact that the sets of output-indistinguishable ESPs obtained from the calibration step are subsets of the corresponding sets from the test circuits, $\text{proj}_e\,\Theta_{i,\text{test}}$.

Condition (2.16) says that the CSP sets for the test circuit, if estimated by first fixing the ESPs to values obtained at the calibration stage, must agree. Agreement here is defined to be unidirectional, with one set being a subset of another. This is only because the correction being performed is from the candidate extract to the reference extract. If bidirectional correction (Corollary 2, below) were required, then we would have equality in condition (2.16).

Finally, condition (2.14) says that the ESP and CSP coordinates in the set $\Theta_{1,\text{test}}$ can only *covary* outside $E_{1,\text{cal}} \times C'_{2,\text{test}}$, i.e., all the points within this set must belong to $\Theta_{1,\text{test}}$. Covariation is defined in Section 2.6. ◇

Next, we state a few corollaries of the theorem.

Corollary 1 (SGI Sufficiency)**.** *SGI models are sufficient for the calibration-correction method to solve the data correction problem in the model universe.*

Proof. Recall from Remark 1 that in the model universe, the data are generated by nominal parameters, $\overline{e}_1$, $\overline{e}_2$, $\overline{c}_{\text{cal}}$, $\overline{c}_{\text{test}}$. We observe that since the models are SGI, these parameters uniquely fit the model to the data, and therefore the sets in conditions (2.12-2.14) only have single entries, leading to these conditions being trivially satisfied:

$$\tilde{\Theta}_{\text{cal}} = \{(\overline{e}_1, \overline{e}_2, \overline{c}_{\text{cal}})\} \neq \emptyset,$$
$$E_{2,\text{cal}} = \{\overline{e}_2\} \subseteq \text{proj}_e\{(\overline{e}_2, \overline{c}_{\text{test}})\} = \text{proj}_e\,\Theta_{2,\text{test}},$$
$$E_{1,\text{cal}} \times C'_{2,\text{test}} = \{\overline{e}_1\} \times \{\overline{c}_{\text{test}}\} \subseteq \{(\overline{e}_1, \overline{c}_{\text{test}})\} = \Theta_{1,\text{test}}.$$

□

Corollary 2 (Bidirectional Correction)**.** *To be able to correct the test data from either extract to the other requires that:*

$$\begin{aligned} \tilde{\Theta}_{\text{cal}} &\neq \emptyset, \\ E_{i,\text{cal}} &\subseteq \text{proj}_e\, \Theta_{i,\text{test}}, \qquad i = 1,\, 2, \\ E_{1,\text{cal}} \times C'_{2,\text{test}} &\subseteq \Theta_{1,\text{test}}, \\ E_{2,\text{cal}} \times C'_{1,\text{test}} &\subseteq \Theta_{2,\text{test}}. \end{aligned}$$

Proof. The proof is a simple union of the sets of conditions implied by Theorem 1 for each direction of correction. □

Remark 8. We note that the condition $C'_{2,\text{test}} \subseteq C'_{1,\text{test}}$ discussed in Remark 7 gets transformed into $C'_{2,\text{test}} = C'_{1,\text{test}}$. ◇

Next we discuss the case of correcting the calibration data itself. This will be important in the next section when we examine the effect of a phenomenon called parameter covariation on the calibration-correction method. There, we will prove that a modified version of the method is able to solve the problem at least for this case, even in the presence of parameter covariation.

Corollary 3 ('Test = Calib' Case)**.** *Consider the data correction problem for the case where the test data and models are the same as the calibration data and models, i.e.,* $\overline{y}_{i,\text{test}} = \overline{y}_{i,\text{cal}}$ *and* $\overline{M}_{\text{test}} = \overline{M}_{\text{cal}}$ *for* $i = 1,\, 2$*. Furthermore, let* $\Theta_{i,\text{cal}} \triangleq \text{ID}_\theta\left(\overline{y}_{i,\text{cal}}, M_{\text{cal}}(\theta)\right)$ *for* $i = 1,\, 2$*, and*

$$C'_{2,\text{cal}} \triangleq \bigcup_{\tilde{e} \in E_{2,\text{cal}}} \text{ID}_c\left(\overline{y}_{2,\text{cal}}, M_{\text{cal}}\left(\tilde{e}, c\right)\right). \tag{2.17}$$

Then, the conditions

$$\tilde{\Theta}_{\text{cal}} \neq \emptyset, \tag{2.18}$$

$$E_{2,\text{cal}} \subseteq \text{proj}_e \, \Theta_{2,\text{cal}}, \tag{2.19}$$

$$E_{1,\text{cal}} \times C'_{2,\text{cal}} \subseteq \Theta_{1,\text{cal}}, \tag{2.20}$$

are necessary and sufficient for the calibration correction method to solve this problem.

Proof. Simply specialize the conditions in Theorem 1 to this case. □

2.6 Covariation Between ESP and CSP Parameter Coordinates Introduces Error into the Method

In this section, we describe covariation (Figure 2.5), and show that it causes the calibration correction method to fail. We then discuss an improvement to the method that addresses this issue. We start by defining a device that will be useful for taking slices of parameter sets.

Definition 7 (Cutting Plane)**.** Consider the space of parameters $\mathbb{R}^q$, the vector $\theta \in \mathbb{R}^q$ partitioned into two sets of coordinates $\theta = (\theta_a, \theta_b) \in \mathbb{R}^{q_a} \times \mathbb{R}^{q_b}$ and the subspaces $A \triangleq \mathbb{R}^{q_a} \times \{0\}$ and $B \triangleq \{0\} \times \mathbb{R}^{q_b}$ corresponding to the θ_a and θ_b coordinates respectively. Let $\tilde{\theta}_a \in A$. Then, we denote the *cutting plane* generated by shifting the origin of B to $(\tilde{\theta}_a, 0)$ with the notation $\text{cut}_{\theta_b}(\tilde{\theta}_a)$.

Definition 8 (Parameter Covariation)**.** Consider the space of parameters $\mathbb{R}^q$ and the vector $\theta \in \mathbb{R}^q$ partitioned into two sets of coordinates $\theta = (\theta_a, \theta_b) \in \mathbb{R}^{q_a} \times \mathbb{R}^{q_b}$. Consider some set of parameters $\Theta \subseteq \mathbb{R}^q$. If there exist $\tilde{\theta}_{a1}, \tilde{\theta}_{a2} \in \text{proj}_{\theta_a} \Theta$ such that $\text{proj}_{\theta_b}\left(\Theta \cap \text{cut}_{\theta_b}(\tilde{\theta}_{a1})\right) \neq \text{proj}_{\theta_b}\left(\Theta \cap \text{cut}_{\theta_b}(\tilde{\theta}_{a2})\right)$, then Θ is said to have *parameter covariation* of its θ_b coordinates with respect to its θ_a coordinates.

Remark 9. We will often abbreviate parameter covariation to just covariation, and say that parameter coordinates can *covary*. ◇

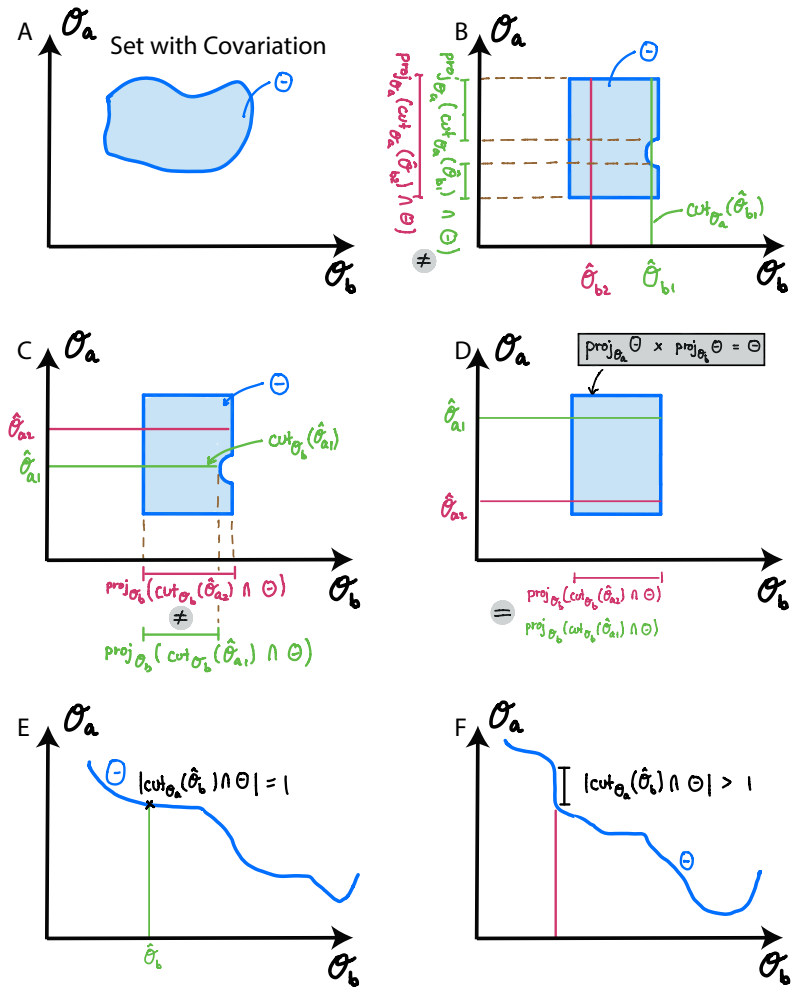

Figure 2.5: Schematic descriptions of parameter covariation and associated results. (A) The arbitrarily shaped set Θ shows parameter covariation. (B, C) Two ways of defining parameter covariation for a given set: Covariation of the θ_a coordinates with respect to the θ_b coordinates (B) and covariation of the θ_b coordinates with respect to the θ_a coordinates (C). The lines represent the cutting planes, and the intersection of these planes and the set Θ is projected onto the appropriate axes. Lemma 1 shows that covariation is equivalent to the Cartesian product condition in (D) not holding. This in turn can be used to show that the two ways of defining covariation (B and C) are equivalent, and therefore the definition of covariation is symmetric. (E, F) Thin covariation. (E) Thin covariation in the θ_a coordinates with respect to the θ_b coordinates. (F) The covariation in the θ_a coordinates is not thin with respect to the θ_b coordinates.

Lemma 1. *Let $\theta = (\theta_a, \theta_b) \in \Theta \subseteq \mathbb{R}^q$ be a partition of the coordinates of $\mathbb{R}^q$. Then, the set Θ has covariation of its θ_b coordinates with respect to its θ_a coordinates if and only if $\mathrm{proj}_{\theta_a} \Theta \times \mathrm{proj}_{\theta_b} \Theta \neq \Theta$.*

Proof. First, we prove the ($\Rightarrow$) direction. Covariation implies that for some $\theta_{a1}, \theta_{a2} \in \mathrm{proj}_{\theta_a} \Theta$ there exists a point $\tilde{\theta}_b \in \mathrm{proj}_{\theta_b} \Theta$ such that

$$\tilde{\theta}_b \in \left(\mathrm{proj}_{\theta_b}\left(\Theta \cap \mathrm{cut}_{\theta_b}(\tilde{\theta}_{a1})\right) \triangle \left(\mathrm{proj}_{\theta_b}\left(\Theta \cap \mathrm{cut}_{\theta_b}(\tilde{\theta}_{a2})\right), \tag{2.21}$$

where $\triangle$ is the symmetric difference set operation. It further implies that there exists a point $\tilde{\theta}_a \in \{\tilde{\theta}_{a1}, \tilde{\theta}_{a2}\} \subseteq \mathrm{proj}_{\theta_a} \Theta$ such that $(\tilde{\theta}_a, \tilde{\theta}_b) \notin \Theta$. Thus, $\mathrm{proj}_{\theta_a} \Theta \times \mathrm{proj}_{\theta_b} \Theta \neq \Theta$.

Next, we prove the ($\Leftarrow$) direction. Let $(\tilde{\theta}_{a1}, \tilde{\theta}_b) \in \mathrm{proj}_{\theta_a} \Theta \times \mathrm{proj}_{\theta_b} \Theta$ be such that $(\tilde{\theta}_{a1}, \tilde{\theta}_b) \notin \Theta$. Since $\tilde{\theta}_b \in \mathrm{proj}_{\theta_b} \Theta$, there exists a $\tilde{\theta}_{a2} \in \mathrm{proj}_{\theta_a} \Theta$ such that $(\tilde{\theta}_{a2}, \tilde{\theta}_b) \in \Theta$. Thus we have $\tilde{\theta}_b \in \mathrm{proj}_{\theta_b}\left(\Theta \cap \mathrm{cut}_{\theta_b}(\tilde{\theta}_{a2})\right)$ but $\tilde{\theta}_b \notin \mathrm{proj}_{\theta_b}\left(\Theta \cap \mathrm{cut}_{\theta_b}(\tilde{\theta}_{a1})\right)$, which proves the assertion. □

Corollary 4. *The set Θ has covariation of its θ_b coordinates with respect to its θ_a coordinates if and only if it has covariation of its θ_a coordinates with respect to its θ_b coordinates.*

Proof. The proof of Lemma 1 can be repeated with straightforward modifications (essentially swapping the roles of θ_a and θ_b) to show the equivalence of the condition $\mathrm{proj}_{\theta_a} \Theta \times \mathrm{proj}_{\theta_b} \Theta \neq \Theta$ to the set Θ having covariation of its θ_a coordinates with respect to its θ_b coordinates. □

Remark 10. This equivalence will allow us to refer to sets having covariation with respect to a given partition. Specifically, we will consider Θ having covariation with respect to the (e, c) partition. ◇

Next, we show that in the presence of covariation, the calibration-correction method is unable to solve the data correction problem even in the case when the test data are the calibration data themselves. In particular, we will assume that the restriction of $\Theta_{1,\mathrm{cal}}$ to $E_{1,\mathrm{cal}} \times \mathrm{proj}_c \Theta_{2,\mathrm{cal}}$ has covariation with respect to the (e, c) partition.

Proposition 1. *Consider the 'Test = Calib' case of the data correction problem described in Corollary 3, along with the definitions of the various sets given there. Assume the condi-*

tions

$$\tilde{\Theta}_{\text{cal}} \neq \emptyset, \tag{2.22}$$

$$C'_{2,\text{cal}} \subseteq \text{proj}_c\, \Theta_{1,\text{cal}}, \tag{2.23}$$

$$E_{i,\text{cal}} \subseteq \text{proj}_e\, \Theta_{i,\text{cal}}, \qquad i = 1,\, 2, \tag{2.24}$$

hold, but the set

$$\Theta'_{1,\text{cal}} \triangleq \Theta_{1,\text{cal}} \cap \left(E_{1,\text{cal}} \times \text{proj}_c\, \Theta_{2,\text{cal}}\right) \tag{2.25}$$

has covariation in its e coordinates with respect to its c coordinates. Then, the calibration-correction method fails to solve this problem.

Proof. Condition (2.23), along with the fact that for the 'Test = Calib' case, $C'_{2,\text{cal}} = \text{proj}_c\, \Theta_{2,\text{cal}}$, implies that $\text{proj}_c\, \Theta'_{1,\text{cal}} = C'_{2,\text{cal}}$. Condition (2.24) implies $\text{proj}_e\, \Theta'_{1,\text{cal}} = E_{1,\text{cal}}$. Covariation implies that $\text{proj}_e\, \Theta'_{1,\text{cal}} \times \text{proj}_c\, \Theta'_{1,\text{cal}} \neq \Theta'_{1,\text{cal}}$. Thus, the proper subset relation $\Theta'_{1,\text{cal}} \subsetneq E_{1,\text{cal}} \times C'_{2,\text{cal}}$ holds, and therefore there exists $(\tilde{e}, \tilde{c}) \in E_{1,\text{cal}} \times C'_{2,\text{cal}}$ such that $(\tilde{e}, \tilde{c}) \notin \Theta'_{1,\text{cal}} \subseteq \Theta_{1,\text{cal}}$. This implies that $E_{1,\text{cal}} \times C'_{2,\text{cal}} \nsubseteq \Theta_{1,\text{cal}}$, which violates condition (2.20). □

Next, we define a specific type of covariation, which we call *thin* covariation, and show that a modification to the calibration-correction method is able to solve the data correction problem for the 'Test = Calib' case when the CSP coordinates covary in this way with respect to the ESP coordinates. In Section 2.7.1, we will show that even the simplest models show non-identifiability with this type of covariation. We will also show that the variance blow up seen in the third column of Figure 2.4 decreases significantly when this modified version of the calibration-correction method is used.

Definition 9 (Thin Covariation)**.** Let $\Theta \subset \mathbb{R}^q$ be a set of parameters and let $(\theta_a, \theta_b) \in \mathbb{R}^q$ be a partition of the coordinates of $\mathbb{R}^q$. If Θ covaries with respect to this partition and if for all $\tilde{\theta}_b \in \text{proj}_{\theta_b}\, \Theta$, we have $\left|\text{cut}_{\theta_a}(\tilde{\theta}_b) \cap \Theta\right| = 1$, then we say that the covariation of the θ_a coordinates of Θ is thin with respect to the θ_b coordinates.

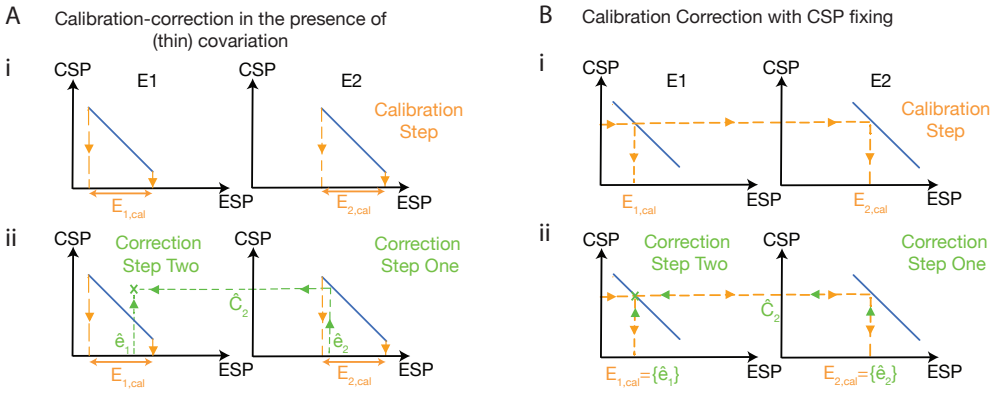

Figure 2.6: (A) A schematic description of how thin covariation between the ESP-CSP coordinates in the estimated joint parameter sets can cause calibration-correction to fail at correcting even the calibration data ('Test = Calib' special case described in Corollary 3). Columns correspond to extracts $\mathcal{E}_1$ and $\mathcal{E}_2$, rows to the calibration and correction steps, as labeled. The blue lines in all the plots are the joint ESP-CSP sets of all the parameter values that fit the calibration model to data. Covariation here is depicted by the fact that the blue line is not vertical, horizontal or a rectangle, i.e., as the CSP value changes, so does the ESP value, and so the ESP and CSPs cannot be picked independently from the respective projections onto the ESP and CSP coordinate axes. 'Thinness' of this covariation (of the CSP coordinates with respect to the ESP coordinates) corresponds to the fact that for each fixed ESP value in the set of possible values it can take, there is one and only one corresponding CSP value. (i) Under this setup, the calibration step leads to the ESP sets shown as projections of the blue lines. (ii) The first correction step fixes the ESP value to a point $\hat{e}_2 \in E_{2,\text{cal}}$, and estimates the only possible CSP value $\hat{c}_2$. The second correction step picks an arbitrary point $e_1 \in E_{1,\text{cal}}$, and uses the CSP value $\hat{c}_2$ to give the parameter point that will be used to generate the final predicted trajectory. It is clear that in general, due to covariation, this point will not lie on the blue line, which is the set of all points that will give the correct prediction. Indeed, this leads to the second failure condition (FC2) described in Remark 6. (B) How the CSP fixing modification (Definition 10) to the calibration step helps solve this issue. Consider the same setup as in (A), with the following exception: The ESP sets estimated at the calibration step are now generated by first intersecting the parameter sets (blue lines) with a line parallel to the ESP axis ('cutting plane' parallel to the ESP subspace in higher dimensions) centered at an arbitrary CSP value that can be attained (i.e., a value in the set $\mathbf{proj}_c\,\tilde{\Theta}_{\text{cal}}$), and secondly projecting these intersections to the ESP coordinates for both extracts. This CSP fixing modification is formally stated in Definition 10. It is clear that with this modification, following a logical procedure similar to the one in (A), the second correction step uses a parameter point on the blue line, avoiding FC2.

Remark 11. We note that if $\Theta \triangleq \mathrm{ID}_\theta(\overline{y}, M(\theta))$, then the condition that for all $\tilde{\theta}_b \in \mathrm{proj}_{\theta_b} \Theta$, we have $\left|\mathrm{cut}_{\theta_a}(\tilde{\theta}_b) \cap \Theta\right| = 1$ is equivalent to the θ_a coordinates of the model $M(\theta_a, \theta_b)$ being SGI for each fixed θ_b. ◇

Remark 11 says that this type of covariation is essentially a statement about the some coordinates being conditionally structurally globally identifiable, despite covarying with respect to the remaining coordinates.

Definition 10 (CSP Fixing)**.** Consider the sets $\Theta_{i,\mathrm{cal}} \triangleq \mathrm{ID}_\theta\left(\overline{y}_{i,\mathrm{cal}}, M_{\mathrm{cal}}(\theta)\right)$, $i = 1, 2$ and let $\tilde{c} \in \mathrm{proj}_c \Theta_{1,\mathrm{cal}} \cap \mathrm{proj}_c \Theta_{2,\mathrm{cal}}$. Then, we define CSP fixing as a modification to the calibration step in which the sets $E_{i,\mathrm{cal}} \triangleq \mathrm{proj}_e\left(\mathrm{cut}_e(\tilde{c}) \cap \Theta_{i,\mathrm{cal}}\right)$ for $i = 1, 2$.

Proposition 2. *Consider the sets $\Theta_{i,\mathrm{cal}} \triangleq \mathrm{ID}_\theta\left(\overline{y}_{i,\mathrm{cal}}, M_{\mathrm{cal}}(\theta)\right)$ for $i = 1, 2$, and the partition $\theta = (e, c)$. Assume that the $\Theta_{i,\mathrm{cal}}$ have thin covariation in their c coordinates with respect to their e coordinates. Then, the calibration-correction method with CSP fixing is able to solve the data correction problem for the 'Test = Calib' case of Corollary 3.*

Proof. Let $\tilde{c} \in \mathrm{proj}_c \Theta_{1,\mathrm{cal}} \cap \mathrm{proj}_c \Theta_{2,\mathrm{cal}}$ and $\tilde{e}_2 \in E_{2,\mathrm{cal}} \triangleq \mathrm{proj}_e\left(\mathrm{cut}_e(\tilde{c}) \cap \Theta_{2,\mathrm{cal}}\right)$. We note that the sets $\mathrm{proj}_c\left(\mathrm{cut}_c(\tilde{e}_2) \cap \Theta_{2,\mathrm{cal}}\right) = \mathrm{ID}_c(\overline{y}_{2,\mathrm{cal}}, M_{\mathrm{cal}}(\tilde{e}_2, c))$ are equal by definition. Now, pick an arbitrary point $\tilde{c}' \in \mathrm{proj}_c\left(\mathrm{cut}_c(\tilde{e}_2) \cap \Theta_{2,\mathrm{cal}}\right)$. It follows that $\tilde{c}' = \tilde{c}$ from the fact that $\tilde{c} \in \mathrm{proj}_c\left(\mathrm{cut}_c(\tilde{e}_2) \cap \Theta_{2,\mathrm{cal}}\right)$ and that the element in $\left|\mathrm{cut}_{\theta_a}(\tilde{\theta}_b) \cap \Theta\right| = 1$ is unique. Thus, the only possible CSP value that can be returned by the first correction step is $\tilde{c}$.

Next, we look at the second correction step. Pick an arbitrary $\tilde{e}_1 \in E_{1,\mathrm{cal}} \triangleq \mathrm{proj}_e\left(\mathrm{cut}_e(\tilde{c}) \cap \Theta_{1,\mathrm{cal}}\right)$. Since the point $(\tilde{e}_1, \tilde{c}) \in \Theta_{1,\mathrm{cal}}$, we have that $\overline{y}_{1,\mathrm{cal}} = \hat{y}_{1,\mathrm{cal}} \triangleq M(\tilde{e}_1, \tilde{c})$, and FC2 is avoided. □

2.7 Computational Investigation of Covariation and CSP fixing

In this section we investigate the effect of covariation on the calibration-correction method computationally, and show that CSP fixing helps reduce the error introduced by covariation. The general approach will be to generate artificial data using the models in Equations (2.8) and (2.9) with a fixed set of parameters, and then to use these same models to perform the calibration-correction method. In this way, we implement the model universe

setting for the investigation, and are able to study the effects of non-identifiability without having to also consider issues of model correctness.

2.7.1 The 'Test = Calib' case of Corollary 3

We show that even the simplest models, such as that in Equation (2.8), show non-identifiability and (thin) covariation in this non-identifiability, and that the calibration-correction method of Definition 6 fails in the 'Test = Calib' special case of the data correction problem (Corollary 3) precisely in the way we expect from the theoretical framework developed in Section 2.6. We also show that with the CSP fixing modification to the calibration step, this type of failure is avoided.

We begin by generating artificial calibration data for extracts $\mathcal{E}_1$ and $\mathcal{E}_2$ using the calibration model in Equation (2.8) with the parameters in Table 2.1. The true trajectories are shown as dotted lines in Figure 2.7 (ii-iii). We have added a small amount of Gaussian noise to these trajectories for visualization purposes; however the trajectories used as data in the calibration-correction method do not contain this added noise. The calibration step was performed with $k_{fG} = 5$ fixed at its true value, reducing the number of parameters in the model to three (the sole CSP k_{rG}, and the pair of ESPs $[\mathrm{Enz}]_0$ and k_c) allowing for the visualization of the full joint distribution of the parameter samples that result from performing the MCMC parameter inference. This visualization is the most direct method of seeing the existence of non-identifiability and of thin covariation in the set of parameters

Table 2.1: Parameters Used to Generate Artificial Data

Type	Parameter	Extract 1 Value	Extract 2 Value	Model(s)
ESP	$[[\mathrm{Enz}]_0]$	100	200	M_{cal}, M_{test}
ESP	k_c	0.012	0.024	M_{cal}, M_{test}
CSP	k_{fG}	5	5	M_{cal}, M_{test}
CSP	k_{rG}	300	300	M_{cal}, M_{test}
CSP	k_{fT}	5	5	M_{test}
CSP	k_{rT}	300	300	M_{test}
CSP	$k_{f,dim}$	20	20	M_{test}
CSP	$k_{r,dim}$	10	10	M_{test}
CSP	$k_{f,rep}$	20	20	M_{test}
CSP	$k_{r,rep}$	10	10	M_{test}

that result from the parameter estimation.

The fitting of the model to the data (Figure 2.7 (ii, iii)) in the calibration step results in an estimate of the joint distribution of the parameter vector $(e_1, e_2, c) \in \tilde{\Theta}_{\text{cal}}$. The three dimensional scatter plots of empty blue circles in Figure 2.7 (iv, v) show the results of this estimation marginalized to the coordinates $(e_2, c) = ([\text{Enz}]_{0,2}, k_{c2}, k_{rG})$ and (e_1, c) respectively for the two extracts. We also fit a surface to the scattering of these points (translucent green gridded surface plot), which helps visualize the fact that these points essentially lie on a two dimensional surface within the three dimensional space of parameters, and that this surface displays thin covariation in its CSP coordinates with respect to its ESP coordinates. The calibration concludes with the projection of the points onto the ESP axes for $\mathcal{E}_1$ and $\mathcal{E}_2$, as shown by the filled in blue circles in Figure 2.7 (iv, v).

The red point in Figure 2.7 (iv) shows the result of the first correction step, where the ESPs $\left([\text{Enz}]_0, k_c\right)$ were fixed to one of the points estimated in the calibration step (red point on the ESP plane), and the CSP was estimated. We see that the CSP value estimated is such that the full parameter point lies in the joint ESP-CSP set (red point lifted up to the green surface). The fitted trajectories from this stage are shown in Figure 2.7 (vii).

We observe from the position of the red point in Figure 2.7 (v) that picking an arbitrary point from the set of ESPs, and using the CSPs from the first correction step leads to a point that does not lie on the joint ESP-CSP surface for extract $\mathcal{E}_1$. The corresponding predicted correction and the true behavior of the artificial data are shown in Figure 2.7 (viii).

Figure 2.7 (vi, ix) show the result of repeating the procedure with the CSP fixing modification applied at the calibration step. In particular, the CSP was fixed at the value that was estimated at the first calibration step (lifted red point in Figure 2.7 (iv)), though any value in the set $\text{proj}_c\, \Theta_{1,\text{cal}} \cap \text{proj}_c\, \Theta_{2,\text{cal}}$ is allowed. The key insight here is that now the ESP sets are much smaller, and in correction step one, the ESPs can only be picked so that the very CSP value that was fixed gets estimated, and subsequently, in correction step two, the only ESP values that can be picked are such that when they are used with this CSP value, the resulting point lies in the set of parameters $\Theta_{1,\text{cal}}$ that fit the true $\mathcal{E}_1$ data to the model. In Figure 2.7 (ix), we see that this leads to the desired correction.

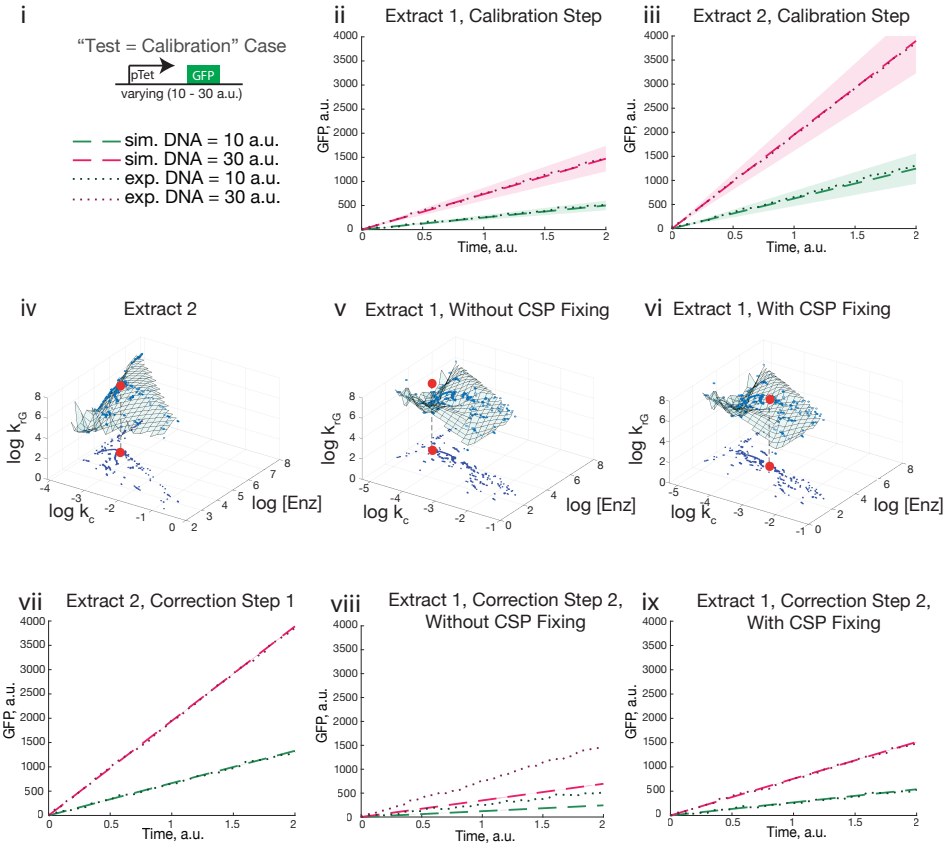

Figure 2.7: *In silico* (model universe) demonstration of the effect of thin covariation and CSP fixing for the 'Test = Calib' case of the data correction problem. (i) Dotted lines: Artificial experimental data, with a small amount of Gaussian noise added for easier visualization (fits were performed on the noise free trajectories). The data was generated using the constitutive expression model of Equation (2.8) at DNA concentrations of 10 and 30 arbitrary units (a.u.). Dashed lines and shaded regions: means and standard deviations of simulated trajectories using parameter points drawn from the estimated posterior distributions. (ii - iii) Artificial data generated using known parameters for two extracts. The CSPs used were the same for the models in both extracts $\bar{c} = (\bar{k}_{fG}, \bar{k}_{rG}) = (5, 300)$, while the ESPs differed for the two extracts, $\bar{e}_1 = (\bar{k}_{c1}, \overline{\mathrm{Enz}}_{0,1}) = (0.012, 100)$, $\bar{e}_2 = (\bar{k}_{c2}, \overline{\mathrm{Enz}}_{0,2}) = (0.024, 200)$. The model fits to the data are the dashed lines and shaded regions, and the parameter distribution for the three parameters estimated ($k_{fG} = 5$ fixed) is shown as the blue empty circles and the fitted translucent green surface in (vi, v). We note that this model shows thin covariation as is visible from the two dimensional surface fit (embedded in 3D) to the scattering of points. Solid blue circles are the projection of the parameter points onto the ESP subspace. (continued below)

Figure 2.7: (continued from above) The plot (iv) also depicts the first correction step, with the red point on the $k_c - [\mathrm{Enz}]_0$ plane denoting the point $\hat{e}_{2,\mathrm{cal}} \in E_{2,calib}$ and the red point on the surface showing the corresponding estimated CSPs. (vii) The fits corresponding to the CSP estimation of correction step one. (v) Correction step two showing an arbitrary ESP point $\hat{e}_1 \in E_{1,\mathrm{cal}}$ with the estimated CSP from (iv) leading to a point that is off the surface of points denoting the set of all parameters that fit extract 1 data to the model. (viii) Corresponding corrections fail. (vi) Correction step two with CSP fixing at the calibration step, and the corresponding corrected trajectories (ix).

2.7.2 Application of CSP Fixing in the General Setting

We conclude this section by demonstrating that when CSP fixing is used in the general case when the test circuit is not the same as the calibration circuit, the CSP fixing modification to the method still leads to significant improvements in the performance of the method (Figure 2.8). The calibration data used was the same as in Section 2.7.1, and the test circuit model (Figure 2.8 (i)) used was the one in Equations (2.9), with parameters used to generate the artificial data given in Table 2.1. As before, dotted lines denote artificial data with a small amount of noise added for ease of visualization only (all the fitting was done on noise free data). The calibration stage with and without CSP fixing was identical to that in Section 2.7.1. To reduce the dimension of the space that the parameter inference algorithm would need to explore, we fixed the forward rates k_{fG}, k_{fT},$k_{f,dim}$, and $k_{f,rep}$, and limited the CSPs to only the reverse rates, k_{rG}, k_{rT},$k_{r,dim}$, and $k_{r,rep}$. In this setting, performing the first correction step gave a set of parameter estimates for the CSPs, and the resulting fits to the $\mathcal{E}_2$ test circuit data are shown in Figure 2.8 (ii). Performing the second correction step led incorrect prediction of the corrected trajectories (failure condition two), as shown in Figure 2.8 (iii). Significantly, applying the CSP fixing modification to the calibration step led to good prediction of the circuit in $\mathcal{E}_2$, as shown in Figure 2.8 (iv).

2.8 Discussion and Future Work

Cell-free extract *in vitro* systems are becoming a useful prototyping tool in synthetic biology, yet the intrinsic variability between the batches of these extracts places limitations on the comparability of results obtained in different batches. Indeed, users currently plan

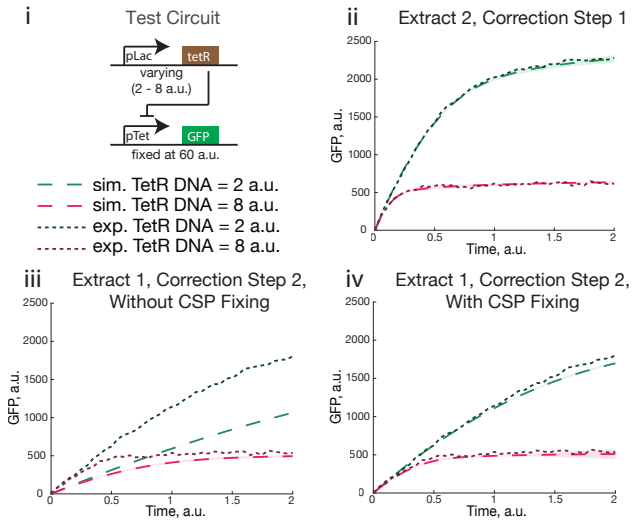

Figure 2.8: The effect of CSP fixing at the calibration step on the correction of novel test circuit data. (i) The test circuit was the repression of the pTet promoter, modeled by Equations (2.9). The pTet-GFP DNA was held fixed at 60 a.u., while the constitutive tetR DNA was varied between 2 a.u. and 8 a.u.. The dotted lines were the artificial experimental data generated using the parameters in Table 2.1. The calibration step was performed as in Figure 2.7, both with and without CSP fixing. The first correction step leads to the fits shown in (ii), and the second correction step leads to the poor corrections shown in (iii). When CSP fixing is employed at the calibration step, the second correction step performs well, as shown in (iv).

their investigations so that all their experiments may be completed before the batch of extract runs out. For this reason, they are limited in the number of things they are able to compare under identical experimental conditions.

We have demonstrated a model-based methodology for calibrating extracts that allows for genetic circuit behavior to be normalized or *corrected*. This methodology is organized into two steps, a *calibration step*, where a set of calibration circuits is used to estimate extract specific parameters of a particular extract, and a *correction step*, in which the calibrations are used to transform a novel circuit's behavior from what it was in a given extract into what it would have been in a *reference* extract. The general idea is that whenever a new extract batch is made, a predefined set of calibration experiments may be performed on that extract to measure its extract specific parameters. These, along with similarly estimated parameters for the reference extract may be used to transform any data collected in the new extract into the reference extract form, and thus be made directly comparable with all other data also transformed into its reference extract form.

We have developed this calibration-correction method for normalizing behavior across extract batches that are assumed to only differ in the values of the parameters of the biochemical reaction network for a given circuit, and not in the *topology* of the networks. The framework here should be applicable to any scenario where only this type of differences exist. One example of this situation is when correcting for run-to-run variability in data, which would require (perhaps a limited number of) calibration experiments to be performed with each run.

Correcting for behavior between topologically different environments, which may arise when *in vitro*-to-*in vivo* prediction of behavior is attempted, or when correcting for variability between different bacterial strains is required, may also be achieved if this method is generalized in a manner outlined next. Briefly stated, the method would still assume that as long as the modeling framework and environment specific parameters are chosen well enough to capture most of the environment specific influences on the circuit (in each environment), then the circuit specific parameters should be largely independent of the environment they are estimated in. This should allow for an *environment specific* set of

calibration experiments to be designed and used in the calibration step, followed by the first and second correction steps that are largely similar to those in the methodology outlined here. The appropriateness of the choice of the level of detail in the modeling, the partition of parameters into extract specific versus circuit specific and the choice of calibration experiments in each of the different environments may be achieved in an iterative, empirical, hypothesis driven manner.

We have also developed theoretical results for when this methodology is expected to work in the presence of parameter non-identifiability. Due to the large discrepancy between the size of biochemical networks and the number of species that can be measured as outputs, parameter non-identifiability is a ubiquitous property of these models. The general prescription in modeling studies [2] is to perform a greater number of experiments to eliminate non-identifiability, reduce the order of the model to reduce the number of parameters, or to fix some parameters to effectively reduce the number of non-identifiable parameters. However, in many cases, more experiments may not be feasible due to cost, time or technological constraints. Model order reduction may not be desirable if, for example, certain mechanisms in the model need to be kept for independent reasons (one example being the explicit modeling of nucleotide binding and consumption during transcription and translation to keep track of resources). The fixing of some parameters, while reducing the number of effective parameters may not remove non-identifiability completely.

The main insight behind our theoretical results is that since we are only trying to correct the trajectories of the very species that we are able to measure in the first place, perhaps the sets of values the non-identifiable parameters can take can be treated as equivalence classes with respect to their usage in our modeling framework. This is a general idea that, even though developed and demonstrated in this specific framework, should apply to a broader class of applications of parametric models, as long as those applications depend on using only the observable outputs. A future direction of this work would be to develop these ideas at this level of generality, starting with the linear systems framework found in control theory.

We can identify a few other directions of investigation for future work. Firstly, condition 2.14 in Theorem 1 might be generalizable to a similar result which gives conditions under which part models with parameter non-identifiability can be combined to predict the behavior of an entire system. In the simplest case, this could be a simple Cartesian product condition, though we suspect that this would be too restrictive, since covariation between the parameters of different parts may exist, requiring a more careful analysis. For example, we may have to prescribe precisely which parameters must be identified, and to what extent, before the remaining non-identifiability does not matter for the output prediction problem. Secondly, we believe that it should be possible to use the result from the theory of differential equations that specifies the continuous dependence of model outputs on parameters to show that the direction of movement, when the outputs are varied under a fixed set of experimental conditions, of a non-identifiable parameter set must be orthogonal to the direction of the non-identifiability, and indeed the non-identifiability must be 'thin', in some geometric sense, in the direction of movement. Lastly, we may wish to generalize these results to the case when there is noise in the data, the parameter sets are replaced by probability distributions, and notions of practical identifiability [53] are incorporated into our analysis.

Appendices

2.A Equivalence of the Two Definitions of the Calibration Step

In this section, we prove two identities that establish the equivalence of the two definitions of the calibration step given in Definition 6 and Remark 4.

Lemma 2. *Let* $\tilde{\Theta}_{\text{cal}}$, $\Theta_{1,\text{cal}}$ *and* $\Theta_{2,\text{cal}}$ *be as defined in Definition 6 and Remark 4. Then, the identities*

$$\text{proj}_c\, \tilde{\Theta}_{\text{cal}} \equiv \text{proj}_c\, \Theta_{1,\text{cal}} \cap \text{proj}_c\, \Theta_{2,\text{cal}}, \tag{2.26}$$

$$\text{proj}_{e_i}\, \tilde{\Theta}_{\text{cal}} \equiv \left\{ e \;\middle|\; \exists c \in \left(\text{proj}_c\, \Theta_{1,\text{cal}} \cap \text{proj}_c\, \Theta_{2,\text{cal}} \right) : (e, c) \in \Theta_{i,\text{cal}} \right\}, \qquad i = 1,\, 2, \tag{2.27}$$

hold.

Proof. First, we prove (2.26) using a series of equivalences. Let $\tilde{c} \in \text{proj}_c\, \tilde{\Theta}_{\text{cal}}$. This is equivalent to

$$\exists e_1, e_2 \,:\, (e_1, e_2, \tilde{c}) \in \tilde{\Theta}_{\text{cal}} \tag{2.28}$$

$$\Leftrightarrow \exists e_1, e_2 \,:\, \overline{y}_{i,\text{cal}} = M_{\text{cal}}(e_i, \tilde{c}), \qquad i = 1, 2 \tag{2.29}$$

$$\Leftrightarrow (e_i, \tilde{c}) \in \Theta_{i,\text{cal}}, \qquad i = 1, 2 \tag{2.30}$$

$$\Leftrightarrow \tilde{c} \in \text{proj}_c\, \Theta_{1,\text{cal}} \cap \text{proj}_c\, \Theta_{2,\text{cal}}, \tag{2.31}$$

which proves the assertion.

Next, we prove (2.27) for e_1 by showing that the left and right hand sides are subsets of each other. The proof for the e_2 case is similar. Denote the set on the left hand side with L, and the one on the right with R. Let $\tilde{e}_1 \in L = \text{proj}_{e_1}\, \tilde{\Theta}_{\text{cal}}$. Then, $\exists \tilde{e}_2, \tilde{c}$ such that

$(\tilde{e}_1, \tilde{e}_2, \tilde{c}) \in \tilde{\Theta}_{\text{cal}}$, which implies $\tilde{c} \in \text{proj}_c \tilde{\Theta}_{\text{cal}}$ and $\overline{y}_{1,\text{cal}} = M_{\text{cal}}(\tilde{e}_1, \tilde{c})$. By the identity (2.26), we have that $\tilde{c} \in \text{proj}_c \Theta_{1,\text{cal}} \cap \text{proj}_c \Theta_{2,\text{cal}}$ and $(\tilde{e}_1, \tilde{c}) \in \Theta_{1,\text{cal}}$, which shows that $L \subseteq R$.

We conclude the proof by showing that $R \subseteq L$. Let $\tilde{e}_1 \in R$, which means that there exists a $\tilde{c} \in \text{proj}_c \Theta_{1,\text{cal}} \cap \text{proj}_c \Theta_{2,\text{cal}}$ such that $\overline{y}_{1,\text{cal}} = M_{\text{cal}}(\tilde{e}_1, \tilde{c})$. Furthermore, since $\tilde{c} \in \text{proj}_c \Theta_{2,\text{cal}}$, there also exists an $\tilde{e}_2$ such that $\overline{y}_{2,\text{cal}} = M_{\text{cal}}(\tilde{e}_2, \tilde{c})$. Together these imply that $(\tilde{e}_1, \tilde{e}_2, \tilde{c}) \in \tilde{\Theta}_{\text{cal}}$, which gives $\tilde{e}_1 \in \text{proj}_{e_1} \tilde{\Theta}_{\text{cal}}$, proving the assertion. □

2.B Equivalence of the Two CSP Subset Conditions Given in Remark 7

The Cartesian product condition given in Equation (2.14) implies two further conditions, which we state in Lemma 3 below. The first of these follows simply by projecting both sides of Equation (2.14) onto the ESP coordinates. The second condition, on the other hand, is stronger than simply projecting (2.14) onto the CSP coordinates. This condition states that the CSP points generated at the first correction step, $C'_{2,\text{test}}$, must be a subset of the set of CSP points generated by fitting $\overline{y}_{1,\text{test}}$ to the model when the ESP points are restricted to be in the set $E_{1,\text{cal}}$.

Lemma 3. *Condition* (2.14)*, which states that* $E_{1,\text{cal}} \times C'_{2,\text{test}} \subseteq \Theta_{1,\text{test}}$*, implies that*

$$E_{1,\text{cal}} \subseteq \text{proj}_e \Theta_{1,\text{test}}, \tag{2.32}$$

$$C'_{2,\text{test}} \subseteq C'_{1,\text{test}}, \tag{2.33}$$

where $C'_{1,\text{test}}$ *is defined in a similar way to* $C'_{2,\text{test}}$*,*

$$C'_{1,\text{test}} \triangleq \bigcup_{\hat{e} \in E_{1,\text{cal}}} \text{ID}_{c|e=\hat{e}}\left(\overline{y}_{1,\text{test}}, M_{\text{test}}(e, c)\right).$$

Proof. Condition (2.32) follows simply by applying the proj_e operator to both sides of condition (2.14). To prove condition (2.33), we note that condition (2.14) implies that for an arbitrary $\tilde{c} \in C'_{2,\text{test}}$, we have that for all $\tilde{e} \in E_{1,\text{cal}}$, the model fits the data, $\overline{y}_{1,\text{test}} = M_{\text{test}}(\tilde{e}, \tilde{c})$.

This in turn implies that

$$\tilde{c} \in \bigcup_{\hat{e} \in E_{1,\text{cal}}} \text{ID}_{c|e=\hat{e}} \left(\overline{y}_{1,\text{test}}, M_{\text{test}}(e, c) \right) = C'_{1,\text{test}}. \tag{2.34}$$

Thus, $C'_{2,\text{test}} \subseteq C'_{1,\text{test}}$. □

Chapter 3

A MATLAB® Simbiology® Toolbox for Circuit Behavior Prediction in TX-TL and Concurrent Bayesian Parameter Inference

3.1 Introduction and Background

The use of computer-aided design (CAD) tools, such as SPICE (Simulation Program with Integrated Circuit Emphasis, [46]) for electric circuit design, decreases the design iteration time in engineering disciplines. We have developed an analogue of such tools for the TX-TL prototyping platform, in the form of a MATLAB® toolbox called `txtlsim` that allows for easy specification, characterization and simulation of genetic circuits.

The use of CAD tools in systems and synthetic biology is not a novel idea. Some examples of simulation software include the TABASCO simulator [40], COPASI [31], ProMot [44], Cello [48] and bioscrape [62]. TABASCO allows for fast stochastic simulation of gene regulatory circuits at the single molecule and single base pair resolution while not trading off too much speed. It does this by employing a dual architecture that allows for switching between modeling base pair resolution reactions and species level reactions. COPASI is a general purpose simulator that allows for the simulation of both stochastic and deterministic models, and even for hybrid models where low copy number species are simulated stochastically, and all other species deterministically. The Process Modeling Tool (ProMoT) employs a unique method for formulating genetic circuits in a *composable* format, with

well defined biochemical signal carriers between the parts [16,44]. Signal carriers take the form of polymerase per second (PoPS), ribosomes per second (RiPS), transcription factor per second (FaPS) and inducer or cofactor signal per second (SiPS). Each part has a defined set of inputs and output terminals, and can only be composed with another part with a corresponding set of terminals. Cello is an example of an electronic design automation tool that takes a desired function as an input, and draws upon a library of Boolean logic gates to generate candidate circuits that perform that function. Finally, bioscrape is a tool developed for performing fast stochastic simulations with time delays, cell lineage tracking and Bayesian parameter inference for general genetic circuit models.

Due to the often complementary nature of the tools available for simulation, inference and analysis, it is desirable to have a way to transfer models between these tools. The Systems Biology Markup Language (SBML) is a widely adopted XML (eXtensible Markup Language) based format for representing biochemical networks. The use of such an information standard for specifying biochemical networks has other advantages: it reduces the chance of old models being lost when the simulator they were written for are no longer supported, and makes it easier for users to parse and understand models written by other researchers, possibly using other tools [34]. The SBML specification is divided into 'levels', where level one specifies a hierarchy of objects that can be used to specify a biochemical network: a model, the comprising compartments, reactions, species (reactants and products in the reactions), parameters and rules. The subsequent levels are intended to implement other functionalities associated with the base network, such as support for MathML and metadata.

In this chapter we describe `txtlsim` in enough detail for the reader to get a sense of the main capabilities of the toolbox. `txtlsim` is written using MATLAB® Simbiology®, which in turn is modeled after SBML. Indeed Simbiology® defines models, compartments, reactions, species, parameters, rules and events as classes, and provides a rich set of methods and properties associated with them. In `txtlsim`, DNA and individual species to be added to TX-TL can be specified using a set of symbolic specification rules, and the toolbox generates a deterministic mass action model of the gene regulatory network ex-

pected to exist in TX-TL under the specified conditions. A typical TX-TL model, specified at the resolution of whole DNA, mRNA and protein species is composed of transcription, translation, RNA degradation, regulatory mechanisms and the inactivation of the ability of TX-TL to express genes. Optionally, linear DNA and protein degradation can be included. Furthermore, other special mechanisms, like sigma factor action or RNA-mediated transcriptional attenuators, may also be included [65].

We highlight several features of `txtlsim`. Firstly, this toolbox requires only a few lines of code to generate a complex chemical reaction network that models the reactions in TX-TL. In lower level specifications, such as Simbiology®, bioscrape [62] or even simply raw ODEs, this would amount to several tens to over a hundred equations that would need to be manually specified and processed. The key reason for the need for this complexity is that the toolbox is able to model the consumption of limited nucleotides and amino acid species, and the loading of the finite catalytic machinery (RNA polymerases, ribosomes, RNases, transcription factors etc). The consumption and degradation of nucleotides and amino acids is thought to underlie the inactivation of the gene expression capability, and is therefore important to model for capturing the full curves of TX-TL reactions. Coupling between different parts of a circuit, via the loading of enzymatic resources [29] or regulatory elements has been shown to introduce unintended interactions between parts of genetic circuits in both TX-TL and *in vivo* [13]. These types of retroactivity or loading effects are automatically and simply incorporated into `txtlsim`, at least with respect to the species that exist in the toolbox, by virtue of the fact that we use mass action models. In fact, a more general property holds: the models built using the toolbox are extensible in the sense that once a species exists, if a new type of interaction is added that relates to that species, none of its previous interactions need to be modified explicitly. Another feature of `txtlsim` worth noting is that the models generated with it can be converted into SBML, and may be exported into any other SBML compatible environment for analysis. The final feature is the MCMC based Bayesian parameter inference capabilities incorporated into `txtlsim` in the form of a sub-toolbox called `mcmc_simbio`. The main feature distinguishing this from existing parameter inference tools is the ability to perform concurrent

Bayesian parameter inference on a *set* of model-experimental data pairs that contain parameters with common identities. This feature is built around a MATLAB® implementation [28] of the affine invariant ensemble MCMC sampler [25] for generating the parameter posterior distributions given the model, data and experimental setup. Our wrapper adopts this sampler to estimate parameter distributions for models written generically in Simbiology®, which in turn is able to import SBML models, and can therefore be used for parameter inference with a large class of models. We note that the study in [35] looked at the issue of concurrent parameter inference (referred to as *consensus* inference there), but only used optimization procedures to estimate point estimates of parameters. Thus, their method does not provide the main advantage of Bayesian inference: insight into parameter identifiability. Indeed, with the concurrence feature, it becomes possible to inform parameters from multiple model-data pairs, potentially improving identifiability. This, in turn, increases the value of using Bayesian inference to study the identifiability properties of model parameters.

In this chapter, we describe the modeling framework, usage and architecture of the toolbox, along with the parameter inference capabilities included in the mcmc_simbio subtoolbox. In Section 3.2, we describe the user end code for setting up a TX-TL model for tetR mediated negative autoregulation. In Section 3.2.1, we elaborate on the choice of the chemical reactions implemented in the toolbox. In Section 3.3 we characterize the parts of an incoherent feedforward loop motif and compare model predictions to experimental data. Next, in Section 3.4, we discuss the software architecture that enables the automatic generation of the biochemical network. Finally, we discuss multi-experiment concurrent parameter inference capabilities of mcmc_simbio in Section 3.5.

3.2 An Overview of the `txtlsim` Toolbox

In this section, we describe the `txtlsim` toolbox in some detail. The code snippet shown below depicts what a user would write to set up the negative autoregulation circuit, in which a repressor represses its own expression, along with that of a reporter.

```
% set up extract and buffer tubes (Simbiology `Model' objects) with parameters
    from a configuration file identified to a particular extract batch.
tube1 = txtl_extract('E1');
tube2 = txtl_buffer('E1');
tube3 = txtl_newtube('negative_autoregulation');
% add DNA specifying a negative autoregulation circuit
txtl_add_dna(tube3, 'ptet(50)', 'UTR1(20)','deGFP(1200)', 1, 'plasmid');
txtl_add_dna(tube3, 'ptet(50)', 'UTR1(20)','tetR(1200)', 0.2, 'plasmid');
% combine tubes, add inducer, 'run' the experiment and visualize results
Mobj = txtl_combine([tube1, tube2, tube3]); % Simbiology Model object
txtl_addspecies(Mobj, 'aTc', 500); % add inducer
simData = txtl_runsim(Mobj, 12*60*60); % Simulate 12 hours of trajectories
txtl_plot(simData, Mobj);
```

The set of commands above closely mimic the actual experimental protocol of setting up the reaction. The functions `txtl_extract` and `txtl_buffer` access extract and buffer parameter configuration files, specified by the input string `'E1'` here, to set up two Simbiology model objects called `tube1` and `tube2` respectively, which are model objects containing extract and buffer specific parameters and species. The configuration files are user defined, and the parameters they contain can come from the literature, or from parameter inference performed on experimental data.

Next, the `txtl_newtube` and `txtl_add_dna` commands are used to initialize a new model object and add different DNAs to the tube respectively. In its most common use case, the `txtl_add_dna` command allows for specification of promoter, untranslated region and coding sequence to form a transcriptional unit on the specified DNA, along with the concentration of the DNA added, and whether it is a linear fragment or plasmid DNA. For example, in the first call to `txtl_add_dna`, the promoter, ribosome binding site (RBS) and coding sequence (CDS) are specified by the strings `'pOR2OR1'`, `'UTR1'` and `'tetR'` respectively. These strings, each describing a component of the transcriptional unit, are used to access a library containing code and parameter files associated with the respective components. These component files specify all the reactions and species associated with the component, and allow for the modular composition of these components into

circuits.

The `txtl_combine` command is used to combine the three tubes into a single model object, `Mobj`, which is subsequently simulated using the `txtl_runsim` command.

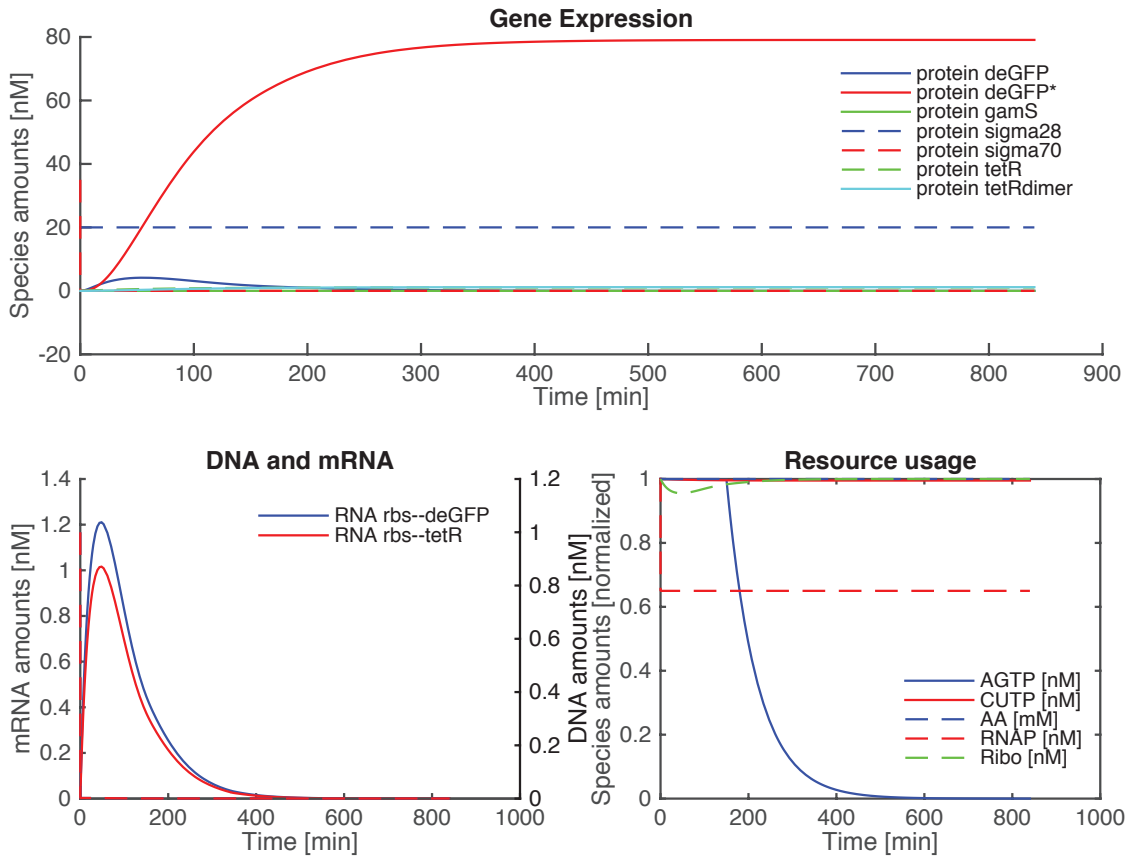

Figure 3.1: Standard output of the TXTL toolbox.

Figure 3.1 shows the result of the `txtl_plot` command, which is arranged into three panels. The top panel shows the protein species that exist within the system. The `protein deGFP*` is the folded GFP. Bottom left plot shows RNA (solid) and DNA (dashed) dynamics. RNA rises before repression by TetR causes transcription to stop. The bottom right plot (normalized to 1) shows that the `AGTP` species degrades after about 3 hours ([49] Figure 1B). The other species we can observe are `CUTP`, ribosomes, amino acids and RNA polymerases.

3.2.1 The Modeling Framework of the `txtlsim` Toolbox

Here we describe the typical reaction network generated by `txtlsim` when a transcriptional unit (TU) is expressed. More complex networks made out of multiple TUs interacting via transcription factor (TF) mediated regulation are simply iterations of this canonical network, but coupled via catalytic and consumable resources, and the relevant regulatory interactions.

We begin with a description of the species naming convention used in the toolbox. The species in the toolbox may be divided into five broad categories: DNA, mRNA, proteins, miscellaneous species like inducers or nucleotides, and the biochemical complexes formed by combining these in defined ways. To avoid a combinatorial explosion, not every possible species that can exist is created, and instead, the toolbox uses the user inputs and corresponding reactions to define the set of species to be created. The species follow a strict naming convention, allowing for the use of regular expressions in parsing the name stings, and for making the decisions required for the creation of the chemical reaction network underlying a given model. Example conventions for DNA, RNA and proteins are given in Table 3.1

Table 3.1: Species naming conventions

Species Type	Convention	Example
DNA	`DNA <promspec>--<utrspec>--<cdsspec>`	`'DNA thio-junk-ptet--utr1--tetR'` `'DNA ptet--utr1--tetR-lva'`
RNA	`RNA <utrspec>--<cdsspec>`	`'RNA utr1--tetR'` `'RNA att1-utr1--tetR-lva'`
protein	`protein <cdsspec>`	`'protein tetR'` `'protein tetR-lva'`

Here, `promspec`, `utrspec` and `cdsspec` are the promoter, untranslated region (UTR) and coding sequence specifications respectively. Some examples of the variations of these specifications are shown in Table 3.1. The specifications are separated by the long hyphen '–', and within each specification, we may have various types of modifiers, such as `junk` DNA on the promoter to protect against DNA degradation, attenuator RNA in the untranslated region [65] or `lva` protein degradation tags on the coding sequence. The

miscellaneous species include inducers like anhydrotetracycline (aTc) or Isopropyl beta-D-1-thiogalactopyranoside (IPTG), core species like ribosomes (`ribo`), RNA polymerases (`RNAP`), `RecBCD` and `RNase` nucleases, etc., and resources like amino acids (`AA`) and grouped nucleotide species (`AGTP`, `CUTP`).

We now turn to a discussion of the reactions set up by the toolbox. The three main processes that almost always get set up for every DNA specified by the user are transcription, translation and RNA degradation. DNA degradation via the RecBCD nuclease happens only to linear DNA fragments, and can be reduced by adding the protein GamS to the system, which sequesters RecBCD [21, 61]. Protein degradation is only active when the ClpXP protease is present in the system, and the protein to be degraded is tagged with a degradation tag. Other conditional behaviors include TF mediated promoter occlusion or activation, protein dimerization, maturation, binding to small molecules like inducers, and non-coding RNA based regulation. These behaviors are included in the biochemical reaction network when the relevant DNA or individual molecule species are specified as inputs to the system. Figure 3.2 shows the general set of reactions associated with each DNA that is specified as an input using the `txtl_add_dna` command.

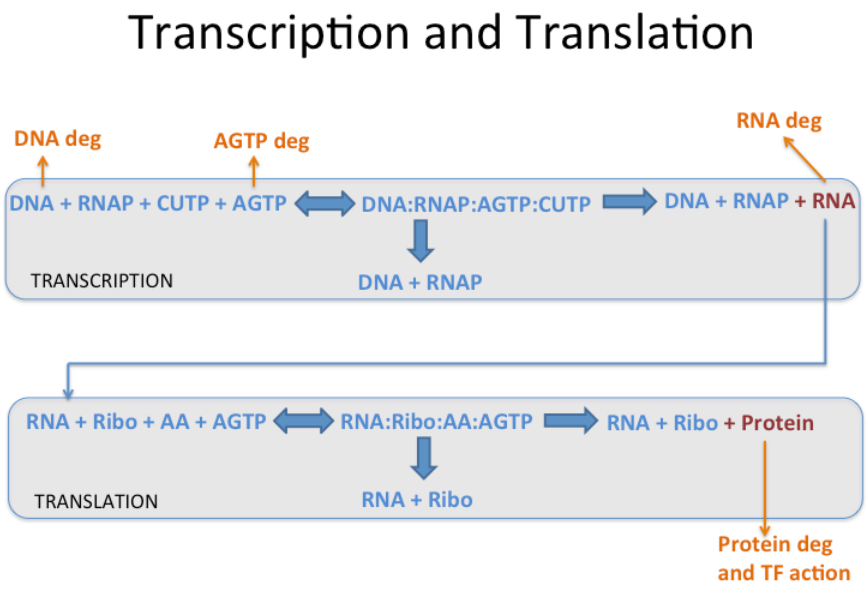

Figure 3.2: A high level description of the mechanics present in the toolbox for each transcriptional unit.

Transcription is modeled using the equations

$$\begin{aligned}
\mathrm{RNAP} + \mathrm{DNA} &\rightleftharpoons \mathrm{RNAP{:}DNA}, && \text{TX machinery binding to DNA,}\\
\mathrm{RNAP{:}DNA} + \mathrm{AGTP} &\rightleftharpoons \mathrm{AGTP{:}RNAP{:}DNA}, && \text{nucleotide binding,}\\
\mathrm{RNAP{:}DNA} + \mathrm{CUTP} &\rightleftharpoons \mathrm{CUTP{:}RNAP{:}DNA}, && \text{nucleotide binding,}\\
\mathrm{AGTP{:}RNAP{:}DNA} + \mathrm{CUTP} &\rightleftharpoons \mathrm{CUTP{:}AGTP{:}RNAP{:}DNA}, && \text{nucleotide binding,}\\
\mathrm{CUTP{:}RNAP{:}DNA} + \mathrm{AGTP} &\rightleftharpoons \mathrm{CUTP{:}AGTP{:}RNAP{:}DNA}, && \text{nucleotide binding,}\\
\mathrm{CUTP{:}AGTP{:}RNAP{:}DNA} &\xrightarrow{k_{tx}} \mathrm{RNAP{:}DNA}_{\mathrm{term}} + \mathrm{mRNA}, && \text{mRNA production,}\\
\mathrm{CUTP{:}AGTP{:}RNAP{:}DNA} &\xrightarrow{\left(\frac{L_m}{4}-1\right)k_{tx}} \mathrm{RNAP{:}DNA}, && \text{consumption, } L_m \text{ = RNA length,}\\
\mathrm{RNAP{:}DNA}_{\mathrm{term}} &\longrightarrow \mathrm{RNAP} + \mathrm{DNA}, && \text{termination.}
\end{aligned} \tag{3.1}$$

The catalytic machinery of transcription is lumped into a single species, denoted RNAP. It is assumed to encompass RNA Polymerases, sigma factors, and other cofactors, but not transcription factors, whose binding will be modeled explicitly. The consumable nucleotide species ATP and GTP are lumped into a single species AGTP, and CTP and UTP are lumped into a species denoted CUTP. After the binding of the catalytic and consumable species, the production of mRNA itself is divided into two reactions, an mRNA production reaction and a nucleotide consumption reaction. As its name suggests, the consumption reaction simply uses up the nucleotide species AGTP and CUTP, without producing mRNA. The rate of this reaction is a multiple of the transcription reaction rate, with a scaling determined by the mRNA length in bases, L_m, so that the stoichiometry of nucleotide consumption and mRNA production is correct. This modeling choice is discussed at length in Chapter 4, and briefly in Appendix 3.A. At the end of mRNA production, a termination complex $\mathrm{RNAP{:}DNA}_{\mathrm{term}}$ forms, which then dissociates into RNAP and DNA in a separate reaction.

The reduced equations for translation,

$$\begin{aligned}
\mathrm{Ribo} + \mathrm{mRNA} &\rightleftharpoons \mathrm{Ribo{:}mRNA}, && \text{ribosome binding to mRNA,}\\
\mathrm{Ribo{:}mRNA} + \mathrm{AA} &\rightleftharpoons \mathrm{AA{:}Ribo{:}mRNA}, && \text{resource binding,}\\
\mathrm{AA{:}Ribo{:}mRNA} + 2\,\mathrm{AGTP} &\rightleftharpoons \mathrm{AA{:}AGTP^2{:}Ribo{:}mRNA}, && \text{resource binding,}\\
\mathrm{AA{:}AGTP^2{:}Ribo{:}mRNA} &\xrightarrow{k_{tl}} \mathrm{Ribo{:}mRNA_{term}} + \mathrm{protein}, && \text{protein production,}\\
\mathrm{AA{:}AGTP^2{:}Ribo{:}mRNA} &\xrightarrow{(L_p-1)k_{tl}} \mathrm{Ribo{:}mRNA}, && \text{consumption, } L_p \text{ = protein length,}\\
\mathrm{Ribo{:}mRNA_{term}} &\longrightarrow \mathrm{Ribo} + \mathrm{mRNA}, && \text{termination,}
\end{aligned} \tag{3.2}$$

look similar to those for transcription. We note that on average it takes two ATP and two GTP per amino acid (AA) residue, leading to the binding and consumption reactions shown below.

RNA degradation is mediated by RNases, and is implemented as an enzymatic reaction,

$$\begin{aligned}
\mathrm{RNase} + \mathrm{mRNA} &\rightleftharpoons \mathrm{RNase{:}mRNA}, && \text{RNase binding to mRNA,}\\
\mathrm{RNase{:}mRNA} &\longrightarrow \mathrm{RNase}, && \text{degradation.}
\end{aligned} \tag{3.3}$$

Similar binding and degradation reactions are set up for mRNA in its various bound forms, such as $\mathrm{Ribo{:}mRNA}$, $\mathrm{AA{:}Ribo{:}mRNA}$, $\mathrm{AA{:}AGTP{:}Ribo{:}mRNA}$, $\mathrm{AA{:}AGTP^2{:}Ribo{:}mRNA}$ and $\mathrm{Ribo{:}mRNA_{term}}$, which result in the degradation of the mRNA and return of the remaining complexed species to the species pool.

Apart from these three main mechanisms, we also model RecBCD mediated linear DNA degradation as an enzymatic reaction, the sequestration of RecBCD by the GamS protein, ClpXP mediated degradation of tagged proteins and transcription factor mediated regulation. Other interactions, for example kinase-phosphatase action, RNA attenuator mediated transcriptional regulation and explicit sigma factor function, can also be included if desired. For brevity, we only list the transcriptional repression and induction reactions here. Repression by the dimerizable protein TetR and its sequestration by the inducer

anhydrous tetracycline (aTc) is modeled as

$$\begin{aligned} 2\,\mathrm{TetR} &\rightleftharpoons \mathrm{TetR}_{\mathrm{dimer}}, && \text{repressor dimerization,} \\ \mathrm{DNA} + \mathrm{TetR}_{\mathrm{dimer}} &\rightleftharpoons \mathrm{DNA{:}TetR}_{\mathrm{dimer}}, && \text{DNA sequestration,} \\ 2\,\mathrm{aTc} + \mathrm{TetR}_{\mathrm{dimer}} &\rightleftharpoons \mathrm{aTc}_2{:}\mathrm{TetR}_{\mathrm{dimer}}, && \text{DNA sequestration.} \end{aligned} \tag{3.4}$$

In Section 3.4, we discuss the software architecture that allows for the automatic generation of these reactions and the interactions between them without the need for the user to specify them explicitly.

3.3 Part Characterization and Circuit Behavior Prediction

In this section, we discuss an example involving the characterization of the parts of a type one incoherent feedforward loop (IFFL), followed by the prediction of the behavior of the IFFL in TX-TL using `txtlsim`, and comparison to experimental data. We begin by parameterizing the model's core mechanics using parameters drawn from the literature. We then decompose the behavior of the IFFL into five distinct parts, and estimate part parameters by fitting models of each part to corresponding experimental data. Finally, we use the characterized parts to predict the behavior of the IFFL under a variety of experimental conditions, and compare the computational predictions with experimental data.

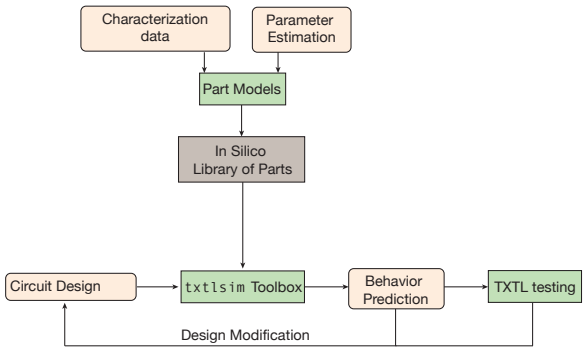

Figure 3.3: The general workflow of using CAD software like `txtlsim` for circuit prototyping. After a library of characterized parts is built, circuit designs can be tested *in silico* and *in vitro*, and modified to fit the design needs. This process can also help refine the models by comparing the model behavior to *in vitro* behavior.

3.3.1 Core Parameters

The parameters in the system come from the literature, and from parameter estimation carried out using experimental data collected in our lab. For parameters from the literature, the main sources are [37] and [60] . Reference [37] gives us the transcription elongation rate of about $1\,\mathrm{nt\,s^{-1}}$, and a $4\,\mathrm{AA\,s^{-1}}$ lower bound on the translation elongation rate. It finds an mRNA degradation half life of $12\text{–}14\,\mathrm{min}$ (which we reproduce in Figure 3.5 (i)), and notes that the degradation machinery does not get saturated even when there is $200\,\mathrm{nM}$ of mRNA in the system. Furthermore, the following features are observed, which we reproduce in the toolbox for characterization purposes: $30\,\mathrm{nM}$ of plasmid DNA gives an approximate steady state of $30\,\mathrm{nM}$ of mRNA, $1\,\mu\mathrm{M}$ of protein is accumulated in $1\,\mathrm{h}$, and the accumulation rate decays exponentially over the next 9 hours, with an eventual maximum expression level of about $10\,\mu\mathrm{M}$ (Figure 3.4).

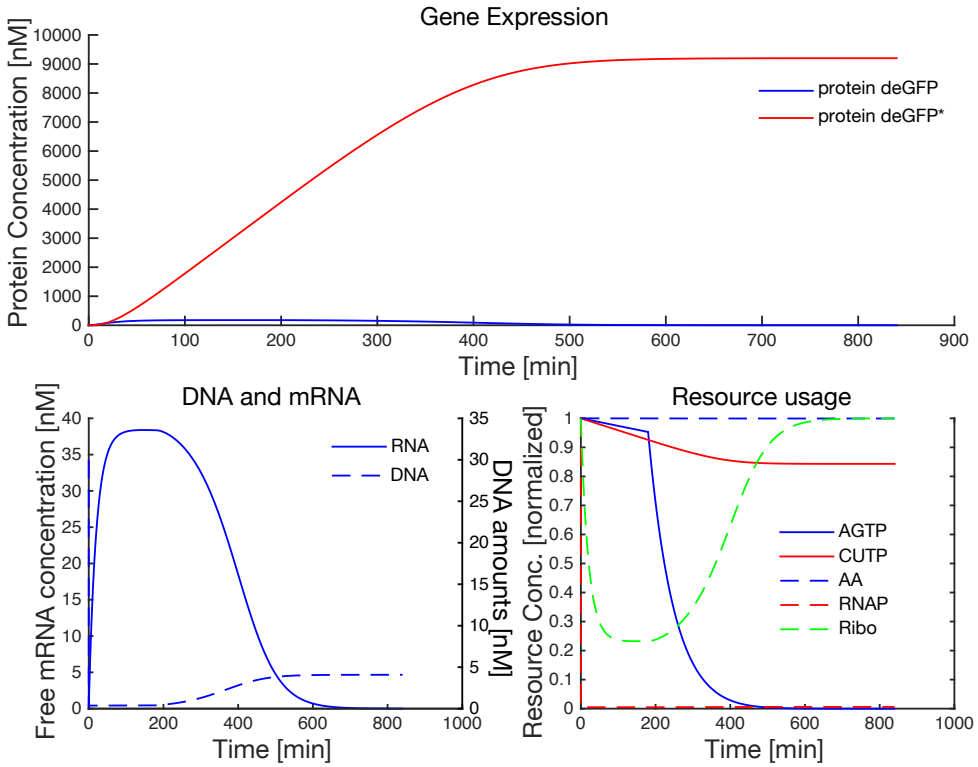

Figure 3.4: Constitutive GFP expression after core parameters were set to values from the literature. When $30\,\mathrm{nM}$ of constitutively expressing deGFP reporter plasmid DNA is expressed in TX-TL, about $10\,\mu\mathrm{M}$ of deGFP produced in $10\,\mathrm{h}$, about $30\,\mathrm{nM}$ of mRNA steady state is reached, and AGTP starts degrading at about three hours.

The concentration of RNAP and Ribosomes and the Michaelis-Menten constant for transcription is given in Table 1 of Karzbrun et al. [37] as $30\,\mathrm{nM}$, >$30\,\mathrm{nM}$ and 1-$10\,\mathrm{nM}$ respectively. Reference [49] shows that ATP levels start to fall exponentially at about three hours, giving us the degradation dynamics of ATP in the system. This is implemented in the toolbox using a Simbiology® event. Reference [60] provides the concentrations of ATP and GTP at $1.5\,\mathrm{mM}$ in TX-TL, those of UTP and CTP at $0.9\,\mathrm{mM}$ and an AA concentration of $1.5\,\mathrm{mM}$. Figure 3.5 (ii) shows a comparison of experimental results form [37] and the simulation results from the toolbox, for the constitutive expression of GFP when plasmid DNA is varied from $5\,\mathrm{nM}$ to $30\,\mathrm{nM}$. Some parameters, like the forward and reverse rates of the binding of amino acids and nucleotides, are difficult to design characterization experiments for, and so we simply fixed them to values that allowed the model to give good agreement with the literature and the experimental data that we collected. These parameters are generally non-identifiable, and the behavior of the model tends to be insensitive to variations in their values over a broad range of values.

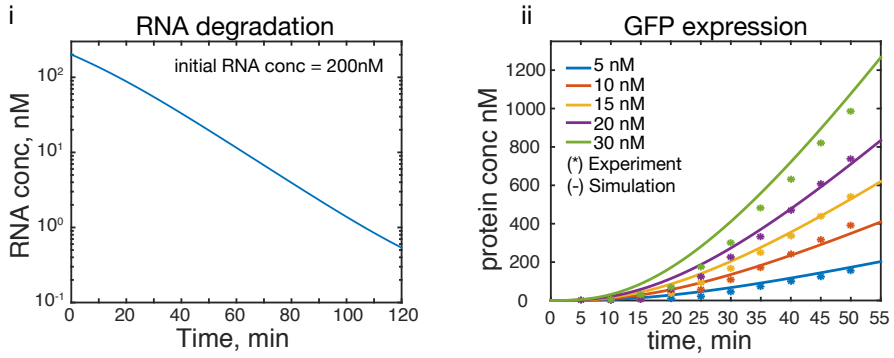

Figure 3.5: (i) RNA degradation half life of about 17 minutes agrees with the numbers in [37] ($12\,\mathrm{min}$) and [12] ($20\,\mathrm{min}$). (ii) Constitutive GFP expression after core parameters were set to values from the literature. The simulation results compared to the data from [37].

In the next section, we describe how we estimated parameters for the parts of an IFFL, before using the part models to predict the behavior of the IFFL in various experimental conditions.

3.3.2 IFFL Part Specific Parameters

The IFFL is a circuit in which an activator transcription factor simultaneously activates a reporter protein and a repressor transcription factor. The reporter protein is also repressed by the repressor. Owing to the fact that activation only requires the production of one protein (the activator), and repression requires the production of two proteins (the activator, followed by the repressor), repression of the reporter is delayed with respect to activation. In cells, where there is dilution present, this mechanism leads to the reporter concentrations showing a pulse. In TX-TL, without active protein degradation, one simply observes a cessation of reporter protein accumulation that occurs sooner than that which would be expected due to the inactivation of TX-TL. Figure 3.6 (Bi) shows a schematic of the IFFL, where the circles represent proteins, pointed arrows show activation (lasR to tetR and lasR to deGFP) and blunt arrows show repression (tetR to deGFP). The inducers 3OC12 (a type of N-acyl homoserine lactone, abbreviated AHL) and anhydrous tetracycline (aTc) activate lasR and sequester tetR respectively, and are shown with green and red arrows (respectively). The lasR protein is under the control of the constitutively expressing pLac promoter, the tetR protein is under the control of the pLas promoter, which is activated by LasR in the presence of 3OC12. The deGFP reporter protein is under the control of a combinatorial promoter, which is only active when activated lasR is present and tetR is absent (or sequestered by aTc). The characterization of the parts of the IFFL was performed using five experiments: the constitutive expression behavior of the pTet and pLac promoters, tetR mediated repression of the pTet promoter, aTc induction, and finally induction via activated lasR. These experiments are summarized in Table 3.2 and the results of the experiments, along with model fitting, are shown in Figure 3.6 (A). All the experiments in Figure 3.6 were performed using plasmid DNA.

Each of the five characterization experiments have subsets of parameters in the model that are naturally associated with them. For instance, the constitutive expression of pTet Figure 3.6 (Ai) informs the dissociation constant for the pTet DNA and RNAP. The Ribosome to RNA binding dissociation constant and the amino acid binding constant were not taken from any literature source, so we chose to estimate these using the first estimation data

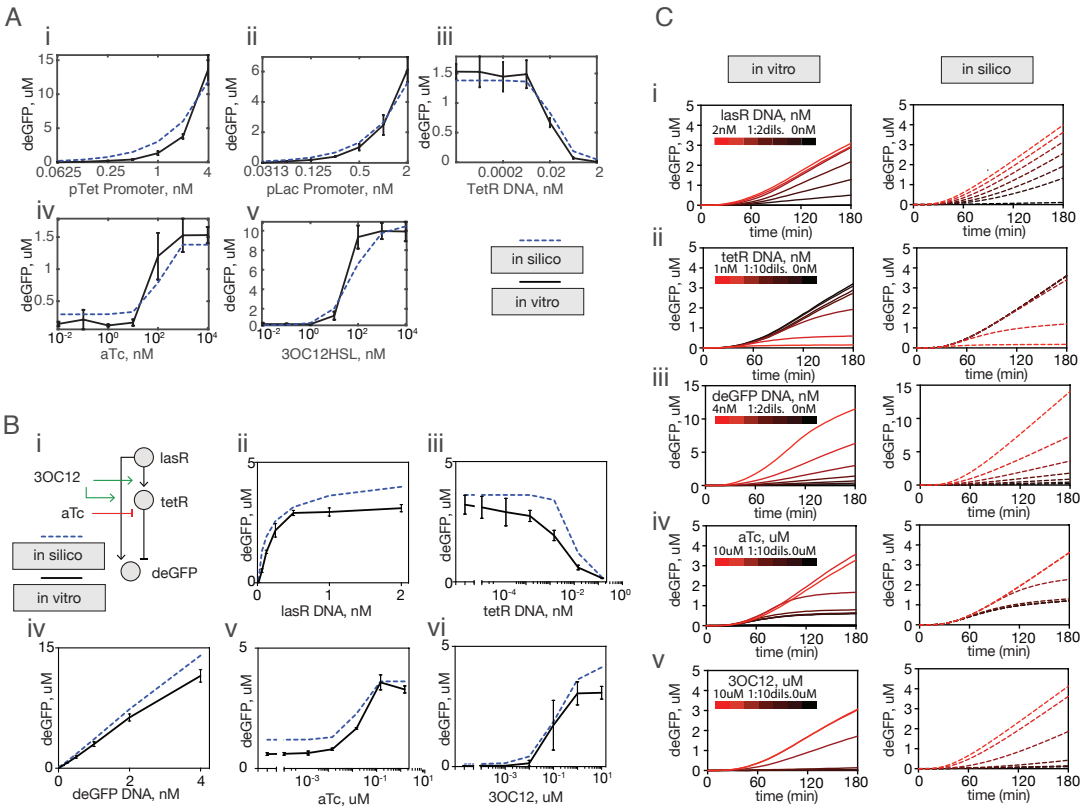

Figure 3.6: Incoherent feedforward loop (IFFL) characterization and behavior prediction. (A) Experiments were performed on parts of the IFFL, and the reporter expression time course data were fit to corresponding models to estimate parameters. Panels (i - v) show the endpoints of the experimental data and the fitted trajectories plotted on the same axes for comparison. Vertical lines show error bars from replicate data. The experimental conditions are described in Table 3.2. (B, C) Comparing predictions from the characterized model to experimental data. (B) shows the endpoints of the model trajectories and the experimental data plotted on the same axes for the five experimental variations tested, and (C) shows the same experiments and model predictions, but for the full time course trajectories. Details of the experimental conditions are in the main text.

Table 3.2: Description of panels in Figure 3.6 (A)

Panel	Experiment	Parameter(s) Estimated	Associated Reactions and Notes
i	Constitutive expression deGFP under a TetR responsive promoter: pTet-UTR1-deGFP at 4, 2, 1, 0.5, 0.25, 0.125 and 0.0625 nM to sequester any native LacI.	$K_{d,pTet}$, $K_{d,ribo}$, $K_{d,AA}$	RNA polymerase binding to the promoter (DNA), Ribo binding to ribosome binding site (RNA) and Amino acids and AGTP binding to the ribosome - RNA complex. The first parameter can be used as a measure of promoter strength, and the other two parameters were estimated here, and fixed to the estimated values during the estimations in panels ii - v.
ii	Constitutive expression of deGFP under a LacI responsive promoter: pLacO1-UTR1-deGFP at 2, 1, 0.5, 0.25, 0.125, 0.0625 and 0.0313 nM. IPTG at 1 mM to sequester any native LacI.	$K_{d,pLac}$	RNA polymerase binding to the promoter (DNA). This parameter can be used as a measure of promoter strength.
iii	pTet repression; jointly with (iv). pTet-UTR1-deGFP at 1 nM. pLac-UTR1-deGFP varied at 2, 0.2, 0.02, 0.002, 0.0002, 0.000 02 and 0.000 002 nM. IPTG at 1 mM to sequester any native LacI.	$K_{d,tdim}$, $K_{d,trep}$, $K_{d,aTc}$	tetR dimerization, ptet DNA sequestration and aTc binding to tetR dimer. Estimation performed jointly with the data in panel iv.
iv	tetR induction (jointly with iii). pTet-UTR1-deGFP at 1 nM. pLac-UTR1-deGFP at 0.1 nM. aTc varied at 10, 1, 0.1, 0.01, 0.001, 0.0001 and 0.000 01 µM. IPTG at 1 mM to sequester any native LacI.	$K_{d,tdim}$, $K_{d,trep}$, $K_{d,aTc}$	tetR dimerization, ptet DNA sequestration and aTc binding to tetR dimer. Estimation performed jointly with the data in panel iii.
v	3OC12 induction of pLas: pLac-UTR1-LasR at 1 nM, pLas-UTR1-deGFP at 1 nM, 3OC12 varied at 10, 1, 0.1, 0.01, 0.001, 0.0001 and 0.000 01 µM. IPTG at 1 mM to sequester any native LacI.	$K_{d,OC12}$, $K_{d,LasLeak}$, $K_{d,LasAct}$, $K_{d,pLas}$	3OC12 binding to lasR, RNAP binding to plas DNA, [OC12:lasR] binding to plas DNA and RNAP binding to activated plas DNA

from this set, and fix these for all subsequent fitting and simulation. Estimation was carried out using the Simbiology® toolbox for MATLAB®, where we used the Levenberg-Marquardt algorithm to solve a non-linear least squares fitting problem to perform the parameter fitting.

3.3.3 Model Predictions

Using these estimated parameters, we built an IFFL *in-silico* and compared its behavior to its *in vitro* analogue. The results are shown in Figure 3.6 (B, C), which show the endpoint expression and the full time course trajectories as a function of experimental conditions

respectively. The nominal experimental conditions for the IFFL in Figure 3.6 (Bi) were as follows. IPTG was added at $1\,\mathrm{mM}$, making the pLac promoter constitutive by sequestering native LacI in the extract. The LasR inducing AHL, 3OC12, was at $1\,\mu\mathrm{M}$. The constitutive activating plasmid pLac-UTR1-lasR was at $1\,\mathrm{nM}$. The repressor DNA pLas-UTR1-tetR was at $0.1\,\mathrm{nM}$ and the reporter DNA plastetO-UTR1-deGFP. The tetR inducer aTc was at $10\,\mu\mathrm{M}$, over which it is toxic to TX-TL. This is the reason why the tetR DNA concentration was kept at a low value of $0.1\,\mathrm{nM}$. At this concentration, there is enough tetR produced to repress pTet almost completely (Figure 3.6 (Aiii)) in the absence of aTc, while still keeping the tetR levels low enough for $10\,\mu\mathrm{M}$ of aTc to fully sequester it. With these nominal conditions, the perturbations shown in the panels (Bii-vi) and (Ci-v) in Figure 3.6 were applied, with the results as shown. We note that the model of the IFFL generated by `txtlsim` and characterized as described in Section 3.3.2 was able to predict the the *in vitro* behavior of the IFFL well.

3.4 Automated Reaction Network Generation

In Section 3.2, we gave an overview of how a user may set up a simple circuit using a few lines of code in `txtlsim`. In this section we go into further details of how the software sets up the model. We start with a walk-through of what each command does, along with a discussion of some of the architectural features of the toolbox, and conclude with a discussion of how the nucleotide and amino acid consumption is modeled for the single step mRNA or protein production models used in the toolbox.

3.4.1 Software Architecture Walk-Through

In this section we cover how the toolbox sets up a model object, and in doing so, highlight various features of the toolbox. Figure 3.7 shows the basic flow of the user level code. We will discuss the function of each of of the commands in the user level code, and in doing so provide an overview of the structure of the toolbox.

The main directory of the toolbox is called `trunk`. The key subdirectories in this di-

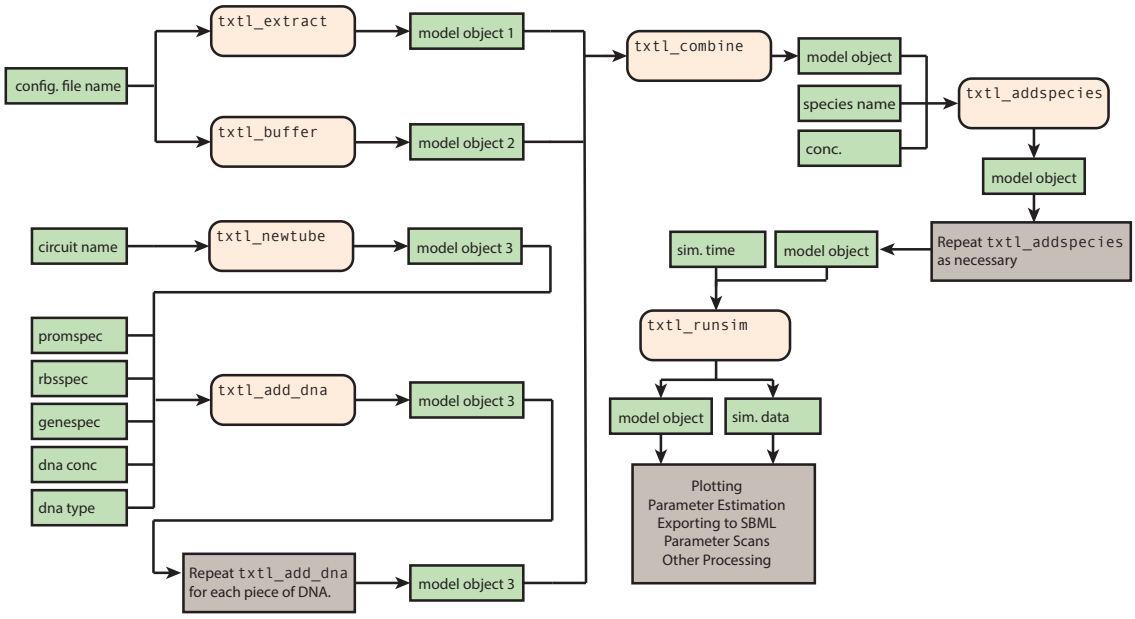

Figure 3.7: Flowchart of the user level code. The `txtl_add_dna` command is the main command that is used to specify the DNA to be added to the model. This allows for all the reactions and species associated with that DNA to be set up in the model. The model is contained in a Simbiology® model class object, and is simulated using the `txtl_runsim` command. See main text for mode details.

rectory are shown in Table 3.3. In particular, we draw attention to the `core`, `config` and `components` directories, which we will be referring to in the code walk-through. The `core` directory contains most of the source code of the network generation part of the toolbox. In contains user end functions like `txtl_add_dna` and hidden functions such as `txtl_mrna_degradation` or `txtl_transcription`. The `config` directory contains (.csv) configuration files containing parameters associated with extracts and buffers. In principle, each extract and buffer has its own configuration file, which is populated using characterization data collected in that extract and buffer. The `components` directory acts as a library of 'parts'. It contains code (.m) and parameter configuration (.csv) files for genetic circuit parts, like promoters, UTRs and CDSs. Promoters in this library can be of an activatable, repressible or combinatorial (e.g.: pLastetO in Figure 3.6) nature. Some promoters, like the arabinose induced pBAD promoter may be repressed by a transcription factor (AraC in this case) when it is not bound to its inducer, and activated by the transcription factor when it is bound to the inducer. There is currently only one type of UTR, the ribosome binding site component, specified as component files in the toolbox. Antisense-

attenuator RNA mediated regulation of transcription, which is also specified via the UTR, is implemented as part of the main source code, and not separate component files. Future releases of the toolbox may separate out this capability into separate component files. Finally, CDSs form the most variable group of component files, and can include reporters, repressors, activators, sigma factors, kinases, phosphatases or proteases, to name a few. All three classes of components can be extended in a straightforward manner to include new components, either as copies of existing files with trivial name changes, of with a small amount of additional work to include capabilities not present in the toolbox.

Table 3.3: Directory structure of the Toolbox

Directory	**Description**
`core`	Core functions of the toolbox, such as `txtl_add_dna` or `txtl_transcription`
`config`	Extract and buffer configuration files (.csv). These contain parameters like transcriptional elongation rate, or the initial concentration of RNA polymerases or nucleotides corresponding to a given extract.
`components`	Component (promoter, UTR and CDS) files. This directory contains both code (.m) and parameter configuration (.csv) files.
`mcmc_simbio`	MCMC toolbox for Simbiology®. This toolbox allows for Bayesian parameter inference to be performed on the parameters of Simbiology® models concurrently over many model-data set pairings. More details can be found in chapter xx.
`examples`	Examples for the modeling toolbox. Includes examples from constitutive gene expression, to the incoherent feedforward loop and the genetic toggle.
`doc`	Contains the User Manual and associated files.

As mentioned in the overview in Section 3.2, the commands `txtl_extract` and `txtl_buffer` are used to initialize the extract specific parameters and species. These functions set up a `txtl_reaction_config` class object that contains methods and properties to manage most of the core (i.e., non part-specific) parameters in the model. The properties of the `txtl_reaction_config` class object are set by a configuration file stored in the `config` directory.

The command `txtl_add_dna` is the workhorse of the network generation phase of the toolbox, and is discussed in some detail here. It takes a model object as its first input, followed by a promoter specification string `promsepc`, a UTR specification string `utrspec`, a CDS specification `cdsspec`, a numerical DNA concentration input, and a DNA type specification string as inputs (Table 3.4). Generically, a call to this function takes the form,

```
txtl_add_dna(model_object, promspec, utrspec,cdsspec, DNAconc, DNAtype),
```

where the promsepc, rbssepc and cdssepc strings are used to access component files of the same names in the component directory. These files contain all the relevant information pertaining to the promoter, RBS or CDS being specified, including the reactions it is involved in and the associated parameters.

Table 3.4: Inputs to the txtl_add_dna command. The parenthetical arguments within the specifications are optional, and if they are not specified, then default values from the component configuration files are used. The DNA concentration can be any nonnegative numerical value, and the DNA type must be either 'linear' or 'plasmid'.

Input	Syntax	Example
model_object	Simbiology® model object	tube3
promspec	string(optional numeric)	'pOR2OR1(50)'
utrspec	string(optional numeric)	'UTR1(40)'
cdsspec	string(optional numeric)	'tetR(650)'
DNAconc	numeric	20
DNAtype	string	'linear'

The txtl_add_dna command is called twice: once when the user first specifies the DNA, and a second time when the command txtl_runsim is called. In the first call, which happens in a 'species setup' mode, most of the species associated with that DNA are specified, and in the second call ('reactions setup' mode), the previously specified set of species is used to set up the reactions within the model. The reason for splitting the set up of the species and the reactions is that the specification of many reactions in the toolbox requires knowledge of exactly which version of the reactants are present. Thus it must be ensured that any command that sets up reactions in the system has access to the exact versions of any species that might appear as reactants in the reactions to be set up. If the txtl_add_dna command were to attempt to set up reactions during its first call, it would not have access to the promoter, UTR and CDS specifications of subsequent lines of txtl_add_dna, and therefore to the versions of the species created due to those specifications. One example where this issue arises is as follows. Consider once again the code snippet for setting up the negative autoregulation circuit shown in Section 3.2. The first call to txtl_add_dna involves the ptet promoter. One of the reactions in this

promoter's component file, `txtl_prom_ptet.m` is the binding of the DNA this promoter is a part of (`DNA ptet--UTR1--deGFP`) to the dimerized tetR protein species. The dimerized tetR species, if present at all, can appear in one of two forms: a form that is not tagged with a protein degradation tag, `protein tetRdimer`, and one that is, `protein tetR-lvadimer`. The version of this species that exists depends on what the string specified by `cdsspec` is: `tetR` or `tetR-lva`. Since the `txtl_add_dna` command specifying this is in a subsequent line, this information is not available to the pTet component file at the time it attempts to set up the reaction in this scenario.

One possibility for the first call to `txtl_add_dna` is that it sets up all the possible versions of the repression reaction, and only the reactions with all reactants with non-zero concentrations have flux through them. While this approach would give the correct system dynamics in principle, it is not scalable as the number of species and reactions would get large quickly, with most of these being unnecessary. A better approach is to only set up the reactions that are actually expected to occur in the system. This approach can be implemented with the two-pass method described above. Specifically, the first set of calls to `txtl_add_dna`, which are the calls explicitly visible in the code snippet, set up all the species that are possible to set up with the information available at this stage. In our example, this means that the protein `protein tetRdimer` is initialized, so that this version of the TetR dimer is used in the specification of the repression reaction, which happens in the second, reaction mode call to `txtl_add_dna` by the `txtl_runsim` command.

One idea hinted at in the above discussion is that when the species are being set up, there might not even be enough information available to set up all the species required. Some species appear as the products of reactions, and are only known once the reactions are specified. Indeed, if species-version dependent reactions lead to product species whose exact version other reactions depend on in turn, then the above two pass method will not suffice. Though we do not implement the solution in this version of the toolbox, one can imagine a multi-pass architecture that alternates between calls to `txtl_add_dna` in species setup and reaction setup modes, with the iterations ending only when the set of reactions and species no longer grows.

In both modes of the call to `txtl_add_dna`, the command performs the following actions: call the component function files for the promoter, the UTR and the CDS, followed by a function to set up mRNA degradation species and reactions, followed by DNA and protein degradation, if present. The promoter file sets up reactions and species (depending on the mode) associated with TF mediated regulation and transcription. Similarly, the UTR function file sets up ribosome binding reactions and other reactions associated with translation.

Returning to the user level code, once all the `txtl_add_dna` commands have been specified, the extract, buffer and DNA model objects (with variable names starting with `tube`) are combined in using the `txtl_combine` command, which simply adds the species and reactions from the three model objects into a single model object, and scales the concentrations of the species to simulate the resulting change in volume. The resulting model object, often named as a variable `Mobj`, can be simulated by `txtl_runsim`. Note that even if simulation is not the immediate goal, one call to `txtl_runsim` should always be performed, since this is where the reactions in the model are set up with the `txtl_add_dna` command. After the call to `txtl_runsim`, we have a fully defined model object, and a `simData` class object containing the results of the simulation. These objects may be used for further simulations, parameter inference, and visualization of the species trajectories, of be exported to other platforms via SBML.

3.5 Tools for Multi-Experiment Concurrent Bayesian Parameter Inference

Bayesian parameter inference via MCMC methods involves designing a reversible Markov chain with stationary distribution matching the posterior parameter distribution (given models and data). This Markov chain can then be simulated and sampled to build an ensemble of points that estimates the desired parameter distribution. One example of this is the Metropolis-Hastings sampler, which was used for parameter inference in [11]. Numerous variations and extensions of MCMC samplers exist, and we use the ensemble

sampler by Goodman and Weare [25], which is particularly well suited to highly anisotropic densities that occur due to parameter non-identifiability in biological models.

In this section we present mcmc_simbio, which performs concurrent Bayesian parameter inference on Simbiology® models. By concurrent parameter inference, we mean the following. Suppose we have a set of different experiments, with a model corresponding to each experiment. Let each experiment-model pair be used to estimate some set of parameters, with the possibility that parameters may be informed by more than one model-experiment pair. Concurrent parameter inference finds the posterior distribution for the parameters given the full set of experiment-model pairs and the specification of the subset of parameters informed by each experiment. This scheme is depicted visually in Figure 3.8. mcmc_simbio builds the concurrent estimation capabilities and Simbiology® specific features around the MATLAB® implementation of the Goodman and Weare ensemble MCMC sampler [20, 25, 28].

One application of this toolbox is during the calibration step of the calibration-correction method introduced in Chapter 2. The calibration step of the method involves sharing the circuit specific parameters (CSPs) between two extracts (i.e., estimating a single set of values for them) while estimating individual sets of values for the extract specific parameters (ESPs). Recall from Section 2.4.2 that we fit the calibration data for each extract in Figure 2.4 to a corresponding model, with each CSP point (comprising the sole parameter coordinate k_{rG}) in the ensemble estimated to fit both models to their respective data sets simultaneously, while each model-data pair fits its own ESPs ($\mathbf{Enz}$ and k_c) independently of the other. This scheme is summarized in Figure 3.9 (A), where each dot represents the set of ESPs or CSPs for one model-experiment pair (determined by the circuit and extract used). A line between two dots indicates that if a parameter appears in both the sets represented by the dots, then it is estimated jointly or concurrently. Figure 3.9 (B) shows a different sharing pattern, where the CSPs are shared between the extracts for both the calibration circuit and the test circuit, and the ESPs are shared between the circuits for each extract. This, and other variations to this pattern, might be useful for comparing with the sets of parameters obtained by the base calibration-correction method.

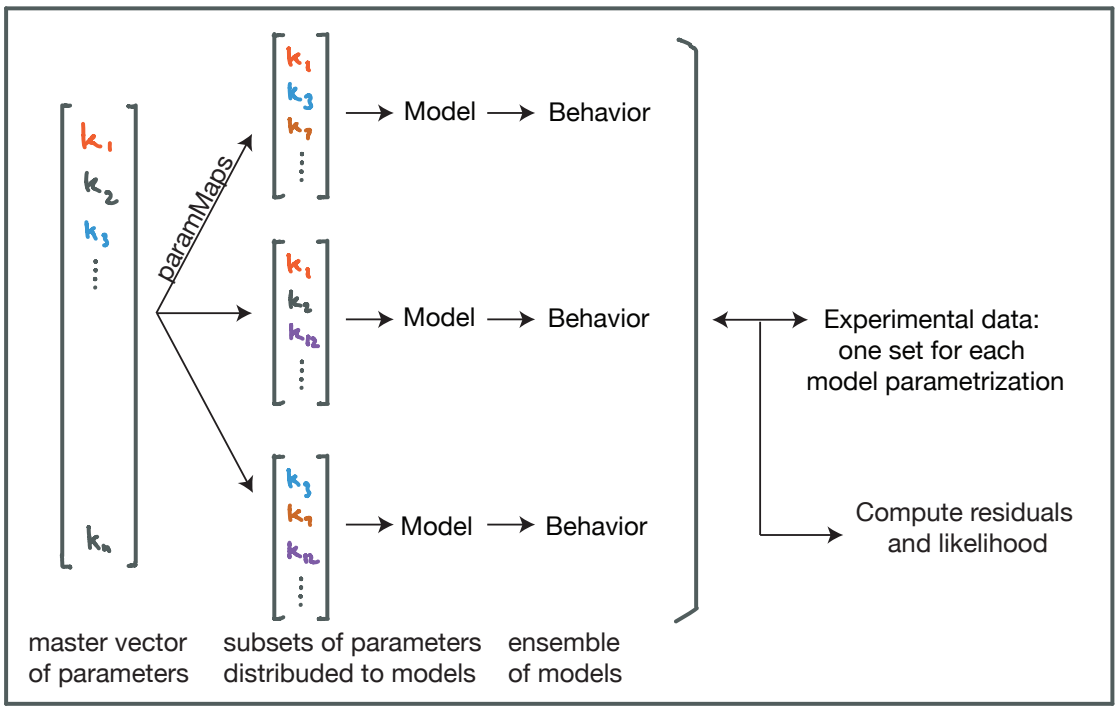

Figure 3.8: Parameter sharing in setting up the concurrent parameter inference problem. Given different sets of experimental data, and corresponding models, which can differ in the structure of the chemical reaction network (network 'topology') or just the parameter values in the models (network 'geometry'), the concurrent parameter inference problem is set up as follows. A master vector is defined, which collects all the parameters in the models into a single vector. Each parameter that is to be shared between models only appears once in the master vector. I.e., parameters that are to be identified with each other between models are treated as an equivalence class of parameters, and their representative is placed in the master vector. Next, **paramMaps** matrices (described in Appendix 3.C) are used to distribute the parameters to models, which are then simulated and their behavior compared against corresponding experimental data to compute the likelihood values for the purposes of the Bayesian Parameter inference.

The ESPs are shared between models of different circuits, corresponding to different biochemical network topologies, while the CSPs are shared between models that differ only in their parameter values. We refer to the first type of sharing as sharing between model *topologies* while the latter as parameter sharing between model *geometries*. In general, each model can be specified by a unique pair of indices, the first of which specifies the model's topology, and the second the model's geometry. Thus, we will often refer to models as topology-geometry pairs.

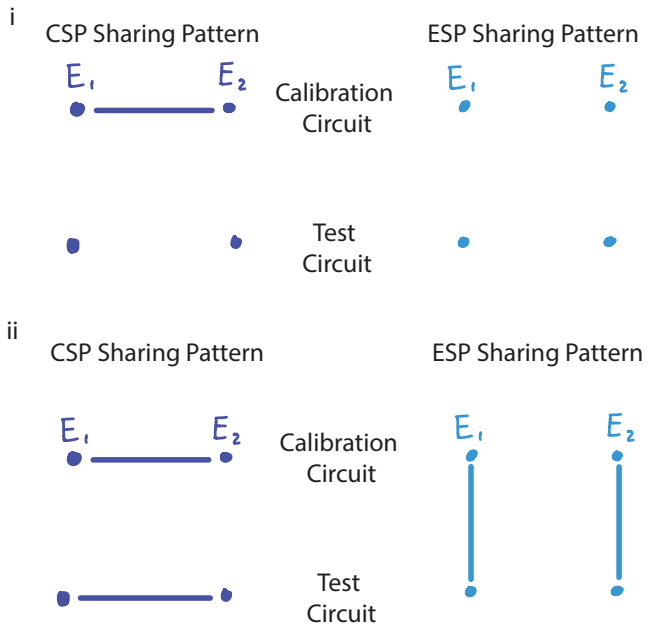

Figure 3.9: Application of the concurrent Bayesian inference capabilities to the calibration-correction problem of Chapter 2. (i) The sharing pattern for the calibration-correction method. At the calibration step, only a single set of values for the circuit specific parameters is estimated. there is no other sharing present. (ii) A sharing pattern where circuit specific parameters are shared within a single model topology (between geometries) and extract specific parameters are shared between circuits within a single extract. We are using different sharing patterns like this to explore, derive and verify the types of mathematical conditions derived in Sections 2.5 and 2.6 of Chapter 2.

An advantage of estimating the entire joint posterior distribution of the parameter, as opposed to using optimization methods for point estimation, is that it can be used to check the identifiability properties of the models. This is useful for understanding which parameters are well constrained by the data, if there is any covariation present between the parameter estimates (Section 2.6), and for designing experiments for reducing param-

eter non-identifiability. Indeed, a particularly useful application of concurrent parameter inference is in experiment design to reduce or remove parameter non-identifiability. One can iterate on the set of models and data, possibly of heterogeneous forms, to find the smallest set that gives identifiable parameters. Indeed, the experiments do not even have to be performed at the design stage, and models may be used to generate artificial data from each model, from which parameter identifiability can be checked.

3.5.1 An Illustrative Example

In this section, we describe the concurrent parameter inference capabilities of mcmc_simbio in some detail using an example similar to the calibration step in the calibration-correction method. Recall from Section 2.4.3 that the calibration step for the example in Chapter 2 involved a model given by

$$\mathrm{D_G} + \mathrm{Enz} \underset{k_{rG}}{\overset{k_{fG}}{\rightleftharpoons}} \mathrm{D_G{:}Enz} \xrightarrow{k_c} \mathrm{D_G} + \mathrm{Enz} + \mathrm{G}, \tag{3.5}$$

where $\mathrm{D_G}$ is the GFP DNA, Enz is the enzyme used to model the transcriptional and translational machinery, $\mathrm{D_G{:}Enz}$ is a complex, and G is the GFP protein. The reaction rate parameters are k_{fG}, k_{rG}, and k_c respectively. The calibration step requires the implementation of this circuit in two extracts, and in the language of mcmc_simbio, we say that there is one topology (circuit, or network topology) and two geometries (implementations of that circuit in different extracts, differing only in their parameter values), leading to two topology-geometry pairs (models). Together, the two models have to be fit to corresponding data sets to estimate an ensemble of parameter values.

The experimental data associated with this parameter inference problem involve the implementation of this circuit in two extracts, at three different initial DNA concentrations ('doses' in the language of mcmc_simbio), with time courses of GFP measured ('measured species'). The parameters expected to be the same across the two extracts are those that pertain to the circuit parts, i.e., the binding-unbinding rates k_{fG} and k_{rG}. In our estimation problem, we set the value of k_{fG} to its true value (the value used to generate the artificial data), and only estimate k_{rG} jointly. The parameters to be estimated individually for each

model are those expected to be different between the two extracts, i.e., the initial enzyme $[\mathrm{Enz}]_0$ concentration and the elongation rate k_c. The resulting parameter space being searched is five dimensional ($\theta = (k_{rG}, [\mathrm{Enz}]_{1,0}, k_{c1}, [\mathrm{Enz}]_{2,0}, k_{c2})$). Figure 3.10 shows the

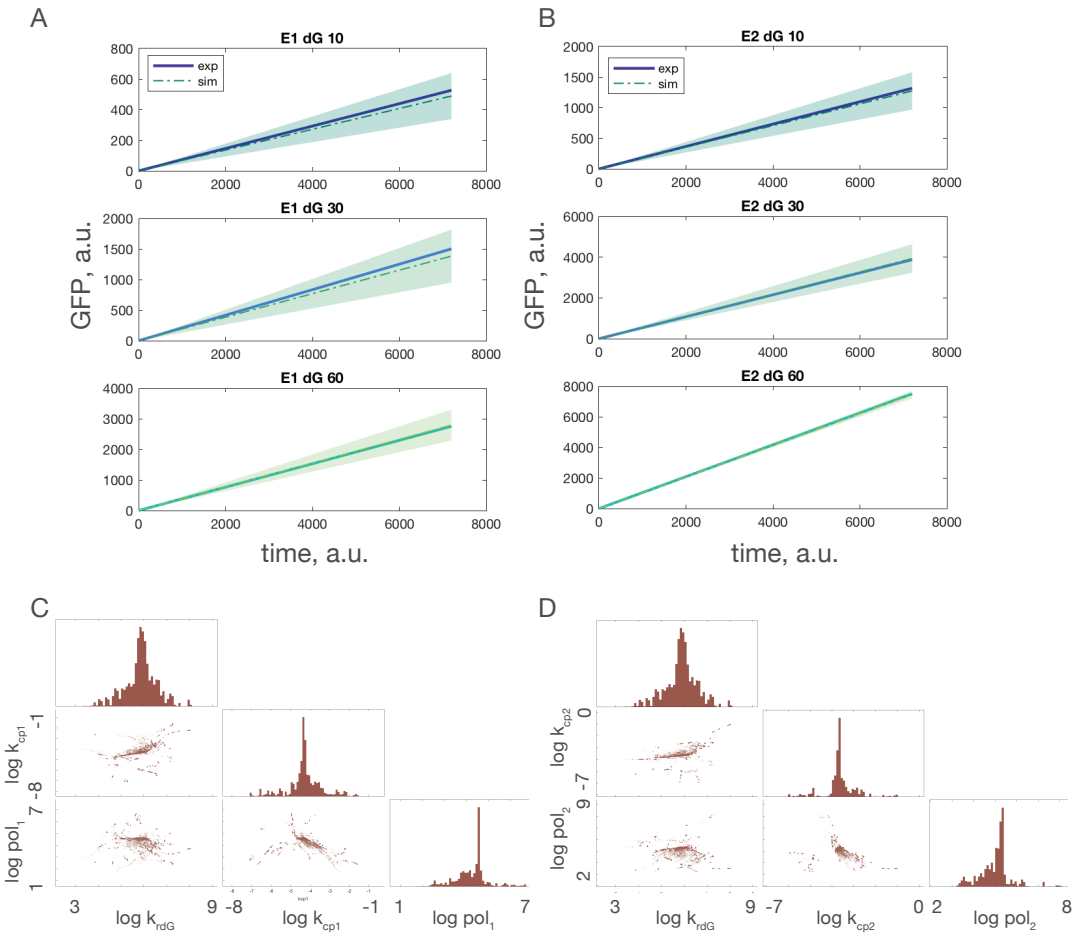

Figure 3.10: mcmc_simbio example. (A, B) Model fits to artificially generated data. Solid line: artificially generated experimental data. Dashed line: mean of 50 simulated trajectories resulting from the ensemble of parameter estimates. Shaded region: standard deviation. (C, D) Pairwise projections of the posterior parameter distributions.

result of estimating the ensemble of parameter points (see Figure 3.10C, D for pairwise projections of the log transformed values of the ensemble) that fit the simulated data to the models. We picked fifty points from the estimated ensemble, and generated model prediction trajectories for each DNA dose (initial condition) and in each of the two extracts (Figure 3.10A, B). Figure 3.11 shows the Markov chains obtained by performing this MCMC run.

The setup of this estimation problem involves setting up a `proj_<projname>.m` project file,

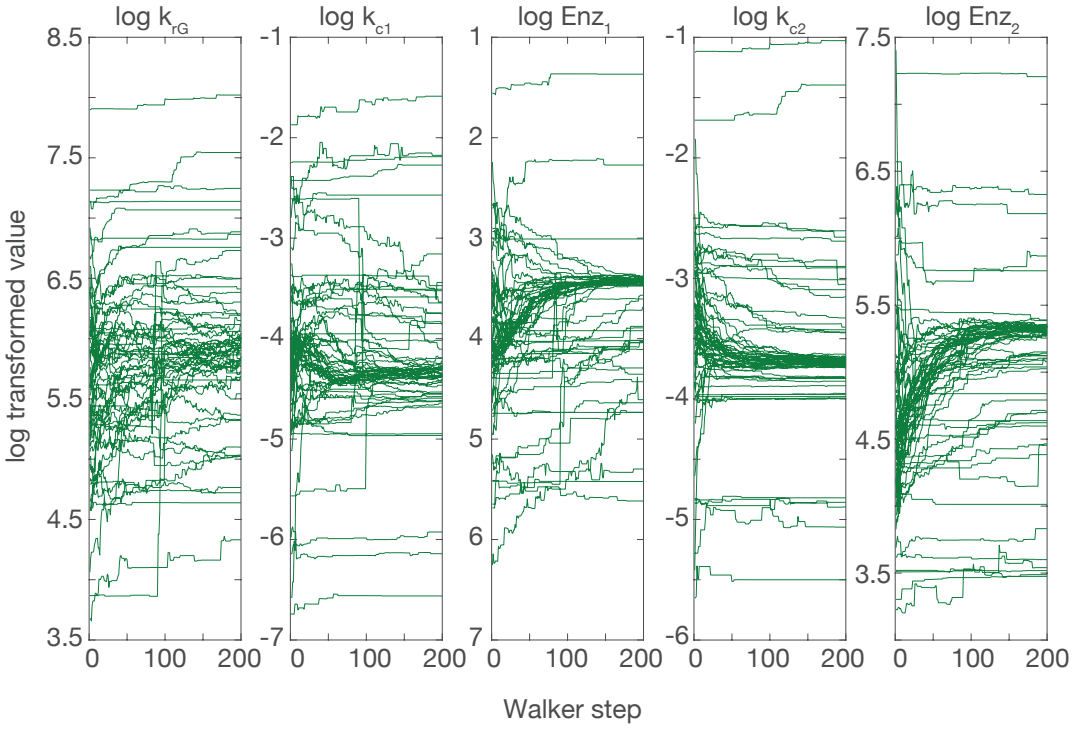

Figure 3.11: Markov chains in the mcmc_simbio example.

which contains information on the experimental data, Simbiology models, specifications of the parameter sharing pattern, the hyperparameters for the MCMC algorithm, and the data visualization specifications. The general layout of the file for this example is shown in the code below.

```
% Initialize the project directory, where the data and plots will be stored.
[tstamp, projdir, st] = project_init;

% Define the simbiology model class object to be used. In this problem there is
    only one topology (circuit): the constitutive gene expression circuit.
    model_protein3 is a file that sets up the appropriate model.
mobj = model_protein3;

% define the mcmc_info struct that specifies the estimation problem structure and
    hyperparameters. See detailed discussion below describing this struct.
mcmc_info = mcmc_info_constgfp3ii(mobj);
```

```
% The model_info field in the mcmc_info struct is a MALTAB struct in itself, and
    contains information about the model topologies, geometries and parameter
    concurrence pattern.
mi = mcmc_info.model_info;

% A list of nominal parameter values to use to generate the data.
rkfG = 5; rkrG = 300; rkc1 = 0.012;
rkc2 = 0.024; cEnz1 = 100; cEnz2 = 200;

% Arrange the parameters in a log transformed 'master' vector.
masterVector = log([rkfG; rkrG; rkc1; rkc2; cEnz1; cEnz2]);

% Generate artificial data for the two extracts using the model object, a vector
    of timepoints, the set of parameters, and information of which species are to
    be dosed and measured.
di = data_artificial_v2({mobj}, {0:180:7200}, {mi.measuredSpecies},...
    {mi.dosedNames}, {mi.dosedVals}, {mi.namesUnord},...
     {exp(masterVector([1:2 3 5])), exp(masterVector([1:2 4 6]))});

% perform the ensemble MCMC parameter estimation.
mi = mcmc_runsim_v2(tstamp, projdir, di, mcmc_info);

% Plotting commands
% get the mcmc chains from saved timestamped data.
marray = mcmc_get_walkers({tstamptouse}, {1:ri.nIter}, projdir);

% plot the parameter distribution corner plots and markov chains
mcmc_plot(marray, mai.estNames, 'tstamp', tstamptouse);

% plot the data trajectories and the simulated data fits.
mvarray = masterVecArray(marray, mai);
marrayOrd = mvarray(mi(1).paramMaps(mi(1).orderingIx, 1),:,:);
fhandle = mcmc_trajectories(mi(1).emo, di(1), mi(1), marrayOrd,...
    titls, lgds, 'projdir', projdir, 'tstamp', tstamptouse, 'extrafignamestring',
        '_extract1');
```

```
% and more plotting commands may be added as needed...
```

The command `mcmc_info_constgfp3ii` is used to set up a MATLAB® 'struct' class object called `mcmc_info`. This struct has three fields, `model_info`, `runsim_info` and `master_info`, which are themselves MATLAB® structs.

The `model_info` struct array, having one entry for each topology, is used to specify information about the models used in the estimation problem. This information includes the full list of parameters in each model topology (`namesUnord`), a specification of how the parameters in the `masterVector` field of the `master_info` struct are to be distributed to each model, and information on dosing (initial conditions) and measurement (output) for each model. This struct is described in detail in Appendix 3.C.

The `master_info` struct is used to specify information about the pool of parameters to be shared across all the topology-geometry pairs. Along with the `masterVector`, it also contains the fields `estNames`, `paramRanges`, and `fixedParams`, which are described below.

Finally, the `runsim_info` struct is used to specify the simulation hyperparameters like the number of points to simulate the chains for, the noise model, the number of MCMC 'walkers' (chains), the step size for the algorithm, whether parallelization is to be used, etc.

Next, we outline how these structs are used to set up the estimation problem. We set up parameter concurrence by first specifying a vector of parameters called the `masterVector`. This vector contains all the parameter values that are to be distributed to all of the topology-geometry pairs. We allow values within the `masterVector` to be either *fixed* or *estimated*. The `masterVector` is initialized to a set of values in the file `mcmc_info_constgfp3ii.m`, and the `master_info.fixedParams` field is used to specify which of these values is to be fixed. The remaining values constitute the vector of parameters to be estimated during the MCMC process, and are named by the cell array `master_info.estNames`. At each iteration of the algorithm, MCMC generates a new proposal of the estimated parameter vector. This proposal is used to populate the relevant entries in the `masterVector`, and the `paramMaps` field of the `model_info` struct is used to distribute the parameters from the `masterVector` to the individual model geometries. Each model is then simulated at each of the dosing

conditions (specified by the `dosedNames` and `dosedVals` fields of the `model_info` struct), and the data for the species to be measured are compared to the experimental data stored in the `data_info` struct array. The `dataToMapTo` and `measuredSpeciesIndex` fields in `model_info` are used to specify which element of the `data_info` struct array a given model's output corresponds to, and the mapping from the model's species to the experimental data trajectories in the data set.

For our example, the code snippets below show the section of `mcmc_info_constgfp3ii.m` that are used to specify this functionality. The comments, shown in green, are used to link the description above to specific functionalities. First, we show the top level constituents of the `mcmc_info` struct.

```
% In this example, model_info is a scalar struct, since there is only one
    topology. In general, each topology gets its own element in this struct.
model_info = struct(...
    'circuitInfo',{circuitInfo},...
    'modelObj', {modelObj},...
    'modelName', {modelObj.name},...
    'namesUnord', {namesUnord}, ...
    'paramMaps', {paramMap}, ...
    'dosedNames', {dosedNames},...
    'dosedVals', {dosedVals},...
    'measuredSpecies', {measuredSpecies}, ...
    'measuredSpeciesIndex', {msIx},...
    'dataToMapTo', dataIndices);

% The master_info struct is a scalar struct and gives the initial masterVector of
    parameter values to be distributed to the topology geometry pairs, which of
    the indices in that vector are to be fixed (fixedParams vector of indices), a
    string of names of parameters (and species initial concentrations) that are
    to be estimated (estParams), and the range of values to seach over for each
    parameter.
master_info = struct(...
    'estNames', {estParams},...
    'masterVector', {masterVector},...
    'paramRanges', {paramRanges},...
```

```
    'fixedParams', {fixedParams});
```

The next code snippet describes how each of the entries of the `model_info` and `master_info` structs is specified. For the single topology in this example, there are two geometries. This is encoded by the fact that the `paramMaps` field of the `model_info` struct is a matrix with two columns, as shown in the code snippet below.

```
% Information describing the circuit. This gets printed in the log file. Here,
    the enzymatic reaction is used to produce the protein G. Since there is only
    one topology, only one string is needed.
circuitInfo = ...
    [' D_G + Enz <-> D_G:Enz (kfG, krG \n'... )
    'D_G:Enz -> G + Enz + protien (kc)\n'...
    'single topology, two geometries.'];

% The masterVector of all the paramters: both fixed and estimated. This vector is
    used during the MCMC algorithm.
% The fixed parameter (kfG) is fixed at a value of 5 (arbitrary units) here, and
    its index in the masterVector is specified by fixedParams.
% At each iteration of the MCMC algorithm, a new 5D parameter point is proposed,
    and used to update
% the relevant entries of the master vector. The values in this vector are
% then distributed to the two geometries.
rkfG = 5; rkrG = 300; rkc1 = 0.012; rkc2 = 0.024; cEnz1 = 100; cEnz2 = 200;
% Note that the values in the masterVector are log transformed.
masterVector = log([rkfG; rkrG; rkc1; rkc2; cEnz1; cEnz2]);
% just the rkfG parameter is fixed, which has index 1 in masterVector
fixedParams = [1];
% The remaining indices are the estimated parameters. The indices are [2:6]
estParamsIx = setdiff((1:length(masterVector))', fixedParams);

% namesUnord is a list of the species and parameters in the model that are set
    from values drawn from the masterVector. These incluce both the fixed and
    estimated values. In each model, we have the parameters 'kfG', 'krG', and
    'kc' whose values get set and the species 'Enz' whose initial value gets set.
namesUnord = {'kfG';'krG';'kc';'Enz'};
```

```
% estParams is a cell array of strings containing the names of the species and
    parameters in the masterVector that are not fixed. There are five values
    here: krG, which is estimated jointly for both geometries, and kc and Enz,
    each of which are estimated separately for each geometry (labeled 1 and 2).
estParams = {'krG';'kc1';'kc2';'Enz1';'Enz2'};

% The paramMaps field is a matrix that maps the elements of the masterVector to
    the individual parameters and species in the topology-geometry pairs. For a
    given topology, we have one matrix, with the number of columns specifying the
    number of geometries associated with that topology, and how the parameters
    from the master vector are to be distributed to each geometry. In this case,
    there are two geometries: the first geometry's parameters and species,
    specified by namesUnord ('kfG', 'krG', 'kc' and 'Enz'), are set to be
    specified (during each MCMC iteration) by indices 1, 2, 3, and 5 of the
    masterVector, i.e., kfG, krG, kc1 and Enz1. Similarly, the second geometry's
    namesUnord species and parameters are set to be specified by
    masterVector(mcmc_info.model_info(1).paramMaps(:,2)), i.e., kfG, krG, kc2,
    and Enz2.
paramMap1 = [1 2 3 5]';
paramMap2 = [1 2 4 6]';
paramMaps = [paramMap1 paramMap2];

% paramRanges: A length(masterVector) by 2 matrix of the ranges of (log
    transformed) values to limit the MCMC sampling to. We limit the search in
    this example to +-3 from the values used to generate the artificial date.
paramRanges = [masterVector(estParamsIx)-3 masterVector(estParamsIx)+3];

% The data_info struct array contains the data sets associated with this
    estimation problem. In this problem, this array is of length two, with the
    first struct entry corresponding to the first geometry, and the second struct
    entry corresponsing to the second geometry.
dataIndices = [1 2];

% next we define the dosing strategy. The species names dG in the Simbiology
    model is to be dosed, and at the values specified.
```

```
dosedNames = {'dG'};
dosedVals = [10 30 60];

% define the species to be measused. Here the species named pG is measured.
measuredSpecies = {{'pG'}};

% The trajectories of the pG species get mapped to the column with index msIx = 1
    in the data_info(dataIndices(i)).dataArray matrix, where i is a geometry
    index.
msIx = 1; %
```

After the `mcmc_info` struct has been defined, the `data_info` struct array is specified. In this example, known models are used to generate artificial data, but in general this struct is defined using real experimental data. In general, `data_info` is a struct array. The (i, j)-th topology-geometry pair uses data specified in

```
data_info(mcmc_info.model_info(i).dataIndices(j)).
```

The struct is used to specify a vector of time points, a list of names of species that are measured, a list of names of species that are dosed, a matrix of dose values, a four dimensional array of data values, and other metadata. This is summarized in Table 3.C.1 in Appendix 3.C.

Once these structs have been defined, they are used as inputs into the `mcmc_runsim` function, which performs the concurrent parameter inference, and saves the results and log files in a time-stamped subdirectory within the toolbox. The toolbox also contains plotting functionalities, functionality for generating data_info structs populated with artificial data, and for converting raw platereader data into data_info structs.

3.6 Discussion

In this chapter, we have described `txtlsim`, a toolbox for simulating batch mode TX-TL reactions using Simbiology®, and mcmc_simbio, a smaller toolbox within `txtlsim` that performs concurrent Bayesian parameter inference on Simbiology® models (not just `txtlsim`

models). The key features of `txtlsim` are that it requires only a few lines of code to generate a model of gene regulatory circuits within TX-TL with enough complexity to model the loading of transcription, translation and RNAse catalytic machinery, and the consumption of resources like nucleotides and amino acids. The requirement for modeling resource consumption while keeping the reaction network size manageable led to the creation of consumption reactions with reaction rates defined to be a function of polymer length and mRNA or protein production rates. These reactions are discussed in greater depth in Chapter 4. The `txtlsim` toolbox also provides support for a wide range of regulatory parts, and is easily extensible by users. Furthermore, the modeling framework of `txtlsim` automatically accounts for retroactivity and loading effects, without needing for these to be explicitly specified in the model equations. We have described the usage of `txtlsim`, and the software architecture needed to automatically generate a complex chemical reaction network from simply specified user inputs. We have validated the model by characterizing core and part parameters using data from the literature, and from experiments performed in the lab, and predicting the behavior of an incoherent feedforward loop circuit.

The mcmc_simbio toolbox enables for different sets of experiments, possibly from heterogeneous sources, to be combined for parameter inference purposes, allowing for more information to be incorporated into the parameter inference problem. Indeed, since the approach returns the joint posterior parameter density, the improvements in parameter identifiability resulting from using multiple experiments to estimate parameters can be checked visually. While we do not show the use of this toolbox for inferring `txtlsim` parameters in this chapter, we do use the toolbox for parameter inference performed in Chapter 2.

There are numerous directions that this work may be extended in. Firstly, capabilities from the MATLAB® based GenSSI toolbox [10] for checking structural identifiability of experiments-model pairs may be added to `txtlsim`. GenSSI uses Lie derivatives of the model output with respect to the parameters to generate approximations to the so called exhaustive summary of model parameters given the initial conditions and outputs of the model. The exhaustive summary contains all the information that can be learned about

the parameters, and if the map from the parameters to the exhaustive summary is injective, the parameters can be shown to be identifiable in the sense of Definition 2. Using GenSSI, along with Bayesian inference on artificial `txtlsim` data, to explore identifiability would form a potent approach for model checking and experiment design.

Another extension of this work would be the incorporation of 'modes' of simulation within `txtlsim`. We might choose to turn on or off reactions to model growth and dilution as part of a 'cell' or 'microfluidics' mode. We may also include modes for more or less detailed models, such as lumping transcription and translation into single reactions, or switching to Hill kinetics from mass action kinetics.

Other extensions include the ability to port models to the bioscrape toolbox [62] and for the models generated by `txtlsim` and other tools to be treated as semantically distinct elements, and be interconnected as subsystems into a larger system.

All in all, we believe that if modeling based approaches are flexible, easy to use and biologically faithful enough for the modeling purpose they are intended for, then they will actually be used by the synthetic biology practitioner, and help accelerate the progress of the field. Our hope is that `txtlsim`, mcmc_simbio, and their extensions help advance this vision.

Appendices

3.A Consumption Reactions as a Means of Tracking Resource Utilization in Reduced Models of Transcription and Translation

In this section, we discuss the use of consumption reactions to maintain the correct stoichiometry of resource utilization during transcription and translation, while still allowing for detailed elongation models to be replaced by single step reactions. An in depth discussion of this subject may be found in Chapter 4 .

Consider the transcription of an mRNA species of length 1kb. Assume that the four types of bases are equally distributed along the mRNA, and so 250 molecules each of ATP, GTP, CTP and UTP are required for the transcription of this mRNA species. In our model, ATP and GTP are modeled together as a species AGTP, where we assume that one unit of the AGTP represents one unit of ATP and one unit of CTP. Similarly, one unit of CUTP represents one unit of CTP and one of UTP. Thus, 250 units each of AGTP and CUTP are needed to transcribe the 1kb mRNA molecule. Looking at the model in Equations (3.1), we see that the mRNA production step reaction consumes one unit each of AGTP and CUTP and produces one mRNA molecule. The consumption reaction also consumes one unit each of AGTP and CUTP, and does not produce an mRNA molecule. Thus, to consume 250 units each of AGTP and CUTP per mRNA produced, we may set the rate of the consumption reaction to be $L_m/4-1=249$ times the rate of the mRNA production step. We now show that with this choice, the correct number of nucleotides gets used per mRNA molecule produced. The rate of mRNA production is given by

$$\frac{\mathrm{d}[\mathrm{mRNA}]}{\mathrm{d}t} = k_{tx} \cdot [\mathrm{CUTP{:}AGTP{:}RNAP{:}DNA}].$$

To compute the rate of nucleotide consumption, we define a variable N_{uninc}, which is the total concentration of nucleotides not incorporated into mRNA. Ie, $N_{\text{uninc}} = 4\cdot[\text{CUTP:AGTP:RNAP:DNA}]+ 2\cdot([\text{AGTP:RNAP:DNA}]+[\text{CUTP:RNAP:DNA}]+[\text{CUTP}]+[\text{AGTP}])$. We would like to show that the rate at which these unincorporated nucleotides are decreasing is $L_m = 1000$ times the rate at which the mRNA is being produced. The rate of consumption of unincorporated nucleotides is calculated as

$$\begin{aligned}\frac{\mathrm{d}N_{\text{uninc}}}{\mathrm{d}t} &= 4\cdot\frac{\mathrm{d}([\text{CUTP:AGTP:RNAP:DNA}])}{\mathrm{d}t},\\ &\quad+2\cdot\left(\frac{\mathrm{d}[\text{AGTP:RNAP:DNA}]}{\mathrm{d}t}+\frac{\mathrm{d}[\text{CUTP:RNAP:DNA}]}{\mathrm{d}t}+\frac{\mathrm{d}[\text{CUTP}]}{\mathrm{d}t}+\frac{\mathrm{d}[\text{AGTP}]}{\mathrm{d}t}\right),\\ &= -4\cdot\left(k_{tx}+\left(\frac{L_m}{4}-1\right)k_{tx}\right),\\ &= -L_m\cdot k_{tx},\end{aligned}$$

where the second equality follows from converting Equations (3.1) into the corresponding mass action ODEs and substituting these into the derivative terms above, and observing that most of the terms in the resulting expression cancel in pairs. We note that the derivation of the consumption reactions for translation is exactly analogous, and the only thing that needs to be stated is that on average, the energetic cost of translation involves four ATP equivalents (two ATP and two GTP) per amino acid incorporation.

3.B MATLAB® Simbiology®

The MATLAB® Simbiology® toolbox follows the SBML standard in its class structure, with classes for models, compartments, species, reactions, parameters, rules, events, kinetic laws and other features. At the top level we have a Simbiology® *model* class object that contains one or more *compartment* class objects. To each compartment, one may associate reaction, species, rule, event, parameter and kinetic law class objects. Individual kinetic law objects, which are associated to a unique parent reaction, are used to specify the reaction properties like the reaction rate law and parameters associated to that reaction. The parameters within a kinetic law refer to parameter class objects, which can be

scoped either at the model level or the kinetic law levels. Parameter objects scoped at the model level can be used by multiple kinetic law objects, while those scoped within a kinetic law object can only be used by that object. Species objects can form either the reactants or products of a reaction, and are scoped at the compartment level. Rules are relationships between parameters, rates and species, and events allow the modeling of discontinuous dynamic changes in the model.

3.C Details of the Data Structures used to Specify the Concurrent Parameter Inference Problem

The `data_info` struct is a MATLAB® struct class array of length nDataSets, where nDataSets is the number of data sets used in the parameter inference problem. Table 3.C.1 gives descriptions of the contents of each field for each element within this struct.

Table 3.C.1: The fields of the `data_info` struct.

Field	Description
`dataInfo`	A human readable description of the data.
`timeVector`	A vector of timepoints of length nTimePoints.
`timeUnits`	A string specifying the time units. Most commonly 'seconds', 'minutes' or 'hours'.
`dataArray`	4-D array of data of size nTimePoints by nMeasuresSpecies by nReplicates by nDoseCombinations.
`measuredNames`	An array of strings representing the names of the measured species. It has length nMeasuredSpecies.
`dataUnits`	An array of strings specifying the units each measured species was measured in. It has length nMeasuresSpecies
`dosedNames`	An array of strings representing the names of the dosed species. It has length nDosedSpecies.
`dosedVals`	A matrix of dose values, of size nDosedSpecies by nDoseCombinations.
`doseUnits`	An array of strings specifying the units of each of the dosed species. It has length nDosedSpecies.

Similarly, the `model_info` struct is of length nTopologies, where nTopologies is the number of different models used in the parameter inference problem. Table 3.C.2 gives de-

scriptions of the contents of each field within this struct for each element within the struct array.

Table 3.C.2: The fields of the `model_info` struct. This struct is of length nTopologies, and specifies the properties of models, and the pattern of parameter sharing across the topologies and geometries for the purposes of setting up the concurrent parameter inference problem.

Field	Description
`circuitInfo`	A human readable description of the model.
`modelObj`	A Simbiology® model class object (in the terminology of the concurrent parameter inference problem, this is a network *topology*).
`namesUnord`	A list of parameters in the model object that are set from values in the master_vector.
`paramMaps`	A matrix of the indices of the `master_vector` that correspond to the parameters specified in the list `namesUnord`. Each column of this matrix specifies one set of elements of the `master_vector` that specify the values of the parameters in `namesUnord` for this model. The number of columns, nCols, of this matrix is the number of different *geometries* of the model, in that the models are different, but only in the values the parameters take, and not in the network topologies.
`dosedNames`	An array of strings representing the names of the dosed species. It has length nDosedSpecies.
`dosedVals`	A matrix of dose values, of size nDosedSpecies by nDoseCombinations.
`measuredNames`	An array of strings representing the names of the measured species. It has length nMeasuresSpecies.
`measuredSpeciesIndex`	An array of indices pointing to the measured species columns of the `dataArray` in the `data_info` struct.
`dataToMapTo`	A numerical vector of length nCols containing the indices of the elements of the `data_info` struct that the model geometries correspond to. These are used when the model predictions are compared to the data in the computation of the log likelihood during MCMC.

The function `mcmc_runsim` generates an inference problem as follows. Suppose there are nTopologies different model topologies specified by `model_info`. Let the topologies be indexed by the letter i. Suppose that for the i-th topology, the corresponding `paramMaps` matrix has nCols_i columns, each corresponding to a geometry. Then, `mcmc_runsim` creates an ensemble of $\text{nCols\_1} + \cdots + \text{nCols\_nTopologies}$ models, and uses the `paramMaps` matrices to distribute the parameter values in `master_vector` into this ensemble of mod-

els. All of these models are then simulated, the residuals generated by comparing the results to the `data_info` elements specified by the `dataToMapTo` field, and the log likelihood computed. The MCMC algorithm uses this to compute the new points in the space of estimated parameters and updates the `master_vector` with the new proposals. The algorithm then repeats until a stopping criterion, such as the number of points to simulate the Markov chains for, is met.

Chapter 4

Model Order Reduction of Transcription and Translation Mass Action Models in the Presence of Resource Consumption

4.1 Introduction

Modeling polymerization reactions like transcription and translation is often used in the study of metabolic pathways [26] in systems biology, and gene regulatory pathways in synthetic biology.

Such modeling can be carried out using either stochastic or deterministic frameworks, each which offers distinct advantages. Stochastic models give us the ability to study the evolution of the probability distributions of species, and work well at low molecular counts, but are computationally expensive. Deterministic models on the other hand, are much less computationally demanding to simulate, but also provide less information than stochastic models.

Models can also exist at various levels of detail. Often, the appropriate level of detail, exemplified by models used in [15], involves the production of RNA and proteins as single steps. In other cases, much more detailed models of transcription incorporating the formation of pre-initiation complexes, release from proximal promoter regions, individual elongation steps, and detailed termination are appropriate [27, 36]. Similarly, models of translation have also been studied at various levels of detail [14, 30].

The choice of which framework to use, stochastic versus deterministic, detailed versus lumped, depends on the specified purpose of the model, and on the computational complexity the user is willing to work with. A detailed stochastic model may be reduced in two different directions: it may be made deterministic under the infinite volume limit [42], or reactions and mechanisms may be lumped into simplified models [51].

The use of models of transcription and translation in cell-free extracts has made it necessary to explicitly account for the consumption and loading of resources that occurs due to gene expression. However, incorporating the consumption of nucleotides and amino acids in the elongation process is done using detailed models, which account for elongation steps individually, as was done in [1] for the case of transcription.

In this chapter, we start with a detailed deterministic ordinary differential equation (ODE) models of transcription similar to the one found in [1], and demonstrate a lumping procedure for reactions that maintains the ability of the model to account for resource consumption. We begin by demonstrating the main idea for the reduction to a single transcription step, and then generalize this to incorporate the possibility of multiple intermediate stages in the transcription process. This general case is required when intermediate nascent transcripts can have some function other than being a precursor to the next elongation step within transcription. An example of this is when non coding RNAs are used as regulatory elements [7, 8]. The model reduction requires the use of the rapid equilibrium assumption [54], which can be rigorously justified using singular perturbation theory [39, 41, 67, 68]. Due to a structural feature of the chemical reaction network describing transcription, when species concentrations are used as state variables in the model, these state variables all possess boundary layer behavior, and converge to a quasi-steady state ([39], Section 1.6). This makes it difficult to bring the differential equations into the standard singular perturbation form, and a change of coordinates, described in Section 4.4.1, is needed before such a form can be achieved.

4.2 Consumption Model

One may divide the stages of transcription and translation into initiation, elongation and termination. Each of these stages involves a complex set of reactions, and may be divided into various smaller stages. For illustrative purposes, we will work with transcription in the rest of this chapter, but a similar reduction procedure may be carried out for translation.

We start with a model similar to the one shown in ([1], Figure 1A), with a few simplifications: we group the different nucleotides into a single species (N), and remove the production of the inorganic pyrophosphate. Defining the notation X:Y to denote the species X and Y bound together into a new species, our resulting model is

$$\begin{aligned}
\mathrm{P} + \mathrm{D} &\underset{k_{Pr}}{\overset{k_{Pf}}{\rightleftharpoons}} \mathrm{P{:}D_1{:}m_0}, && \text{Polymerase binding,}\\
\mathrm{P{:}D_1{:}m_0} + \mathrm{N} &\underset{k_{Nr}}{\overset{k_{Nf}}{\rightleftharpoons}} \mathrm{P{:}D_1{:}m_0{:}N}, && \text{Nucleotide binding,}\\
\mathrm{P{:}D_1{:}m_0{:}N} &\xrightarrow{k_{tx}} \mathrm{P{:}D_2{:}m_1}, && \text{Elongation,}\\
&\vdots &&\\
\mathrm{P{:}D_n{:}m_{n-1}{:}N} &\xrightarrow{k_{tx}} \mathrm{P{:}D_t} + \mathrm{m_n}, && \text{Elongation,}\\
\mathrm{P{:}D_t} &\xrightarrow{k_{\text{term}}} \mathrm{P} + \mathrm{D}, && \text{Termination.}
\end{aligned} \tag{4.1}$$

Here, the entire initiation stage is lumped into a single reaction where an RNA polymerase molecule (P) binds to a DNA molecule (D). Elongation then proceeds iteratively, with each iteration consisting a reversible nucleotide binding reaction and an irreversible elongation step. Finally, termination is modeled as the dissociation of the complex comprising the RNA polymerase bound to the final location on the DNA ($\mathrm{P{:}D_t}$).

We can write this network in terms of ODEs using mass action kinetics

$$
\begin{aligned}
\frac{\mathrm{d}[\mathrm{P}]}{\mathrm{d}t} &= -k_{Pf}[\mathrm{P}][\mathrm{D}] + k_{Pr}[\mathrm{P{:}D_1{:}m_0}] + k_{\mathrm{term}}[\mathrm{P{:}D_t}],\\
\frac{\mathrm{d}[\mathrm{D}]}{\mathrm{d}t} &= -k_{Pf}[\mathrm{P}][\mathrm{D}] + k_{Pr}[\mathrm{P{:}D_1{:}m_0}] + k_{\mathrm{term}}[\mathrm{P{:}D_t}],\\
\frac{\mathrm{d}[\mathrm{P{:}D_1{:}m_0}]}{\mathrm{d}t} &= k_{Pf}[\mathrm{P}][\mathrm{D}] - k_{Pr}[\mathrm{P{:}D_1{:}m_0}] - k_{Nf}[\mathrm{P{:}D_1{:}m_0}][\mathrm{N}] + k_{Nr}[\mathrm{P{:}D_1{:}m_0{:}N}],\\
\frac{\mathrm{d}[\mathrm{P{:}D_1{:}m_0{:}N}]}{\mathrm{d}t} &= k_{Nf}[\mathrm{P{:}D_1{:}m_0}][\mathrm{N}] - k_{Nr}[\mathrm{P{:}D_1{:}m_0{:}N}] - k_{tx}[\mathrm{P{:}D_1{:}m_0{:}N}],\\
\frac{\mathrm{d}[\mathrm{P{:}D_2{:}m_1}]}{\mathrm{d}t} &= k_{tx}[\mathrm{P{:}D_1{:}m_0{:}N}] - k_{Nf}[\mathrm{P{:}D_2{:}m_1}][\mathrm{N}] + k_{Nr}[\mathrm{P{:}D_2{:}m_1{:}N}],\\
&\vdots\\
\frac{\mathrm{d}[\mathrm{P{:}D_n{:}m_{n-1}{:}N}]}{\mathrm{d}t} &= -k_{tx}[\mathrm{P{:}D_n{:}m_{n-1}{:}N}] + k_{Nf}[\mathrm{P{:}D_n{:}m_{n-1}}][\mathrm{N}] - k_{Nr}[\mathrm{P{:}D_n{:}m_{n-1}{:}N}],\\
\frac{\mathrm{d}[\mathrm{P{:}D_t}]}{\mathrm{d}t} &= k_{tx}[\mathrm{P{:}D_n{:}m_{n-1}{:}N}] - k_{\mathrm{term}}[\mathrm{P{:}D_t}],\\
\frac{\mathrm{d}[\mathrm{N}]}{\mathrm{d}t} &= \sum_{k=1}^{n}\left(k_{Nr}[\mathrm{P{:}D_k{:}m_{k-1}{:}N}] - k_{Nf}[\mathrm{P{:}D_k{:}m_{k-1}}][\mathrm{N}]\right),\\
\frac{\mathrm{d}[\mathrm{m_n}]}{\mathrm{d}t} &= k_{tx}[\mathrm{P{:}D_n{:}m_{n-1}{:}N}].
\end{aligned}
\tag{4.2}
$$

We may wish to lump all the elongation steps into a single or a few steps, while maintaining the correct average rates of RNA production and nucleotide consumption. A simple model is given by

$$
\begin{aligned}
\mathrm{P} + \mathrm{D} &\underset{k_{Pr}}{\overset{k_{Pf}}{\rightleftharpoons}} \mathrm{P{:}D}, && \text{Polymerase binding},\\
\mathrm{P{:}D} + \mathrm{nN} &\underset{k_{Nr}}{\overset{k_{Nf}}{\rightleftharpoons}} \mathrm{P{:}D{:}nN}, && \text{Nucleotide binding},\\
\mathrm{P{:}D{:}nN} &\xrightarrow{k_{tx}} \mathrm{P{:}D_t} + \mathrm{m}, && \text{RNA production},\\
\mathrm{P{:}D_t} &\xrightarrow{k_{\mathrm{term}}} \mathrm{P} + \mathrm{D}, && \text{Termination},
\end{aligned}
\tag{4.3}
$$

where n is the number of nucleotides needed to create a single RNA transcript. The names of the relevant rate constants are shown on the arrows in the model. While this simple model preserves the stoichiometry of the consumption of substrate nucleotides and the production of RNA, it models the kinetics of the system incorrectly; it is describing the scenario where n nucleotides simultaneously collide with the P:D complex to form a larger

complex. This is both biologically implausible and computationally intractable, due to the appearance of n as an exponent in some of the terms in the mass action ODEs. Figure 4.1B shows the results of attempting to simulate the resulting model for various transcript lengths.

To circumvent this problem, we propose modeling the consumption of nucleotides separately from the production of RNA, and scaling the RNA production rate by n to get the nucleotide consumption rate. The resulting *consumption model* is given by the equations

$$\begin{aligned}
\mathrm{P} + \mathrm{D} &\underset{k_{Pr}}{\overset{k_{Pf}}{\rightleftharpoons}} \mathrm{P{:}D}, && \text{Polymerase binding,} \\
\mathrm{P{:}D} + \mathrm{N} &\underset{k_{\mathrm{Nr}}}{\overset{k_{\mathrm{Nf}}}{\rightleftharpoons}} \mathrm{P{:}D{:}N}, && \text{Nucleotide binding,} \\
\mathrm{P{:}D{:}N} &\xrightarrow{k_{\mathrm{con}}} \mathrm{P{:}D}, && \text{Consumption,} \\
\mathrm{P{:}D{:}N} &\xrightarrow{k_{\mathrm{reduced}}} \mathrm{P{:}D_t} + \mathrm{R}, && \text{RNA production,} \\
\mathrm{P{:}D_t} &\xrightarrow{k_{\mathrm{term}}} \mathrm{P} + \mathrm{D}, && \text{Termination}
\end{aligned} \tag{4.4}$$

with k_{tx} denoting the transcription rate, and $k_{\mathrm{con}} = (n-1) \times k_{tx}$ the rate of a *consumption reaction*. A pictorial representation of this scheme is shown in Figure 4.1A.

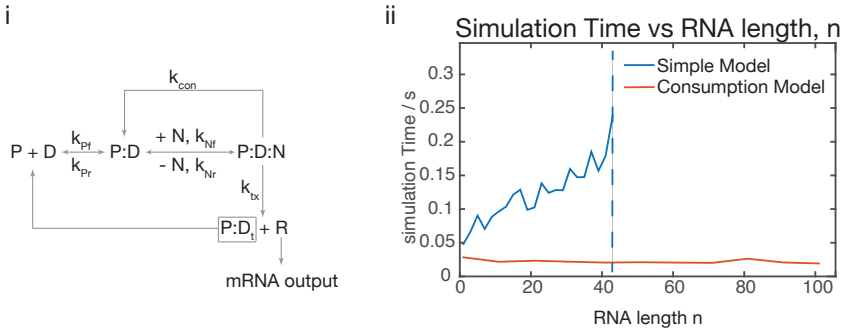

Figure 4.1: (A) Schematic illustrating the consumption reaction. (B) Time required to simulate 10000 seconds of transcription using the simple model (4.3) and the consumption model (4.4). At about n = 42, and MATLAB® is no longer able to complete the simulation.

The simple model (4.3) was able to simulate the production of RNA for up to an n of about 42, after which MATLAB® returned a simulation error. The consumption model (4.4) was able to simulate transcription for all n tested ($n > 2000$). The results for $n < 100$ are shown in Figure 4.1B.

We assume that that at any time, there is only one polymerase molecule bound to the DNA. For this single occupancy model, the features that are preserved between the full and reduced model are that the rate of consumption of nucleotides is independent of the transcript length, while the rate of production of RNA scales inversely with the length. To see this, let k_{tx} be the rate at which the elongation step occurs in the full model (4.1). Then, setting $k_{\text{reduced}} = k_{\text{tx}}/n$ and $k_{\text{con}} = (n-1)k_{\text{reduced}}$ gives us the rate of RNA production as

$$\begin{aligned}\frac{\mathrm{d}[\mathrm{R}]}{\mathrm{d}t} &= k_{\text{reduced}}[\mathrm{P{:}D{:}N}] \\ &= \frac{k_{tx}}{n}[\mathrm{P{:}D{:}N}].\end{aligned}$$

To compute the rate of nucleotide consumption, we define a species concentration $[\mathrm{N}_{\text{uninc}}](t)$, which is the concentration of nucleotides not incorporated into RNA at time t. I.e., $[\mathrm{N}_{\text{uninc}}] = [\mathrm{P{:}D{:}N}] + [\mathrm{N}]$. The rate of consumption of nucleotides, then, does not directly depend on the length n of the RNA, and is n times the rate of RNA production,

$$\begin{aligned}\frac{\mathrm{d}[\mathrm{N}_{\text{uninc}}]}{\mathrm{d}t} &= \frac{\mathrm{d}([\mathrm{P{:}D{:}N}])}{\mathrm{d}t} + \frac{\mathrm{d}([\mathrm{N}])}{\mathrm{d}t} \\ &= -(k_{\text{reduced}} + k_{\text{con}})[\mathrm{P{:}D{:}N}] \\ &= -k_{\text{tx}}[\mathrm{P{:}D{:}N}] \\ &= -n\frac{\mathrm{d}[\mathrm{R}]}{\mathrm{d}t}.\end{aligned}$$

4.3 Mathematical Preliminaries

In this section, we introduce some ideas from chemical reaction network theory (CRNT) and singular perturbation theory, and use them to prove a couple of results which will be useful when we try to carry out our model reduction in Section 4.6. We begin by introducing basic definitions and notions from CRNT.

4.3.1 The Zero Deficiency Theorem and Asymptotic Stability

Let $\{x_1, \ldots, x_s\}$ be a set of s species which participate in r reactions

$$\sum_{j=1}^{s} A_{ij} x_j \xrightarrow{k_i} \sum_{j=1}^{s} B_{ij} x_j, \qquad i \in \{1, \ldots r\}, \tag{4.5}$$

where the $A_{ij} \in \mathbb{R}_{\geq 0}$ are called the stoichiometric coefficients of the system, and the reaction rate of the ith reaction is given by $k_i > 0$. We will call Equation (4.5) a *chemical reaction network* or *reaction network* for short. In the representation above, reversible reactions are modeled as two separate irreversible reactions. The reactants $\sum_{j=1}^{s} A_{ij} x_j$ and products $\sum_{j=1}^{s} B_{ij} x_j$ are called the *complexes* of this reaction network. Let m denote the number of distinct complexes in a reaction network, and label them by $c_1, c_2, \ldots c_m$.

In matrix form we may write this reaction network as

$$Ax \xrightarrow{k} Bx,$$

where the species concentration vector is $x \triangleq [x_1, \ldots, x_s]^T \in \mathbb{R}^s_{\geq 0}$, coefficient matrices are $A \triangleq [A_{ij}] \in \mathbb{R}^{r \times s}_{\geq 0}$, $B \triangleq [B_{ij}] \in \mathbb{R}^{r \times s}_{\geq 0}$, and the reaction rate vector is $k \triangleq (k_1, \ldots, k_r)^T \in \mathbb{R}^r_{>0}$. We define the stoichiometric matrix $S \triangleq (B - A)^T$. Recall that using standard mass action kinetics, we can write the dynamics of the network given by Equation (4.5) as

$$\frac{\mathrm{d}x}{\mathrm{d}t} = S\, v(x, k), \qquad t \geq 0, \quad x(0) = x_0, \tag{4.6}$$

where $v(x, k)$ is a vector function whose ith component gives the velocity of the ith reaction.

Definition 11. The *stoichiometric subspace* associated with the mass action Equation (4.6) is given by $\mathcal{S} \triangleq \mathrm{Im}((B - A)^T)$, and is a subspace of $\mathbb{R}^s$. The *rank of the reaction network* (4.6) is given by $q \triangleq \mathrm{rank}(B - A)^T$, a q dimensional manifold called the *stoichiometric compatibility class* is defined by the affine space $(x_0 + \mathcal{S}) \cap \mathbb{R}^s_{\geq 0}$.

Remark 12. The stoichiometric compatibility class is an important concept when defining properties of trajectories, and in particular those of equilibria. These properties include the existence and multiplicity of equilibria, and whether these equilibria are (asymptotically) stable. Feinberg [18] describes the issues involved in Section 5.2 of his paper. We simply note that trajectories beginning at x_0 stay in the stoichiometric compatibility class $(\mathcal{S}+x_0)\cap\mathbb{R}^s_{\geq 0}$ containing x_0. The standard notion of asymptotic stability will be understood to be with respect to the stoichiometric compatibility class containing the trajectory being considered. More precisely, we will consider an equilibrium x^* to be *asymptotically stable* if any trajectory beginning sufficiently close to x^* and within the stoichiometric compatibility class containing x^* stays close to x^* and approaches x^* in the limit $t \to \infty$. ⋄

Definition 12 ([5], Definition 6.3)**.** Let c_i and c_j be complexes in the reaction network (4.5). We say there is a *direct path* from c_i to c_j if $c_i \to c_j$, an *indirect path* from c_i to c_j if there exists a sequence of complexes $(c_i, c_{i_1}, \ldots, c_{i_p}, c_j)$ such that $c_i \to c_{i_1}$, $c_{i_1} \to c_{i_2}$, ... $c_{i_p} \to c_j$. There esists a *path* from c_i to c_j if there exists a direct or indirect path from c_i to c_j. The complexes c_i and c_j are *linked* if $c_i = c_j$, or if there is a direct or indirect path from one to the other. This definition of linkage can be used to separate a chemical reaction network into equivalence classes known as *linkage classes*. Finally, we call a reaction network (4.5) *weakly reversible* if, for each pair (c_i, c_j), the existence of a path from c_i to c_j implies the existence of a path from c_j to c_i.

Definition 13 ([5], Definition 6.2)**.** The *deficiency* of the network (4.5) is given by $\delta \triangleq m-l-q$, where l is the number of distinct linkage classes and $q = \text{rank}(\nu)$.

Theorem 2. *[[18], Theorem 4.1] Assume that the reaction network* (4.6) *has zero deficiency and is weakly reversible. Then, for arbitrary positive rate constants, the system* (4.6) *has the following properties: Each stoichiometric compatibility class contains precisely one equilibrium, this equilibrium is asymptotically stable (see remark below), and there is no nontrivial periodic orbit in* $\mathbb{R}^s_{\geq 0}$.

Remark 13. As in the remark above, asymptotic stability in Theorem 2 is taken with respect to the stoichiometric compatibility class containing that equilibrium, defined by the initial conditions of the system. A pictorial depiction of this situation is given in [17, Fig. 1, 2] ⋄

4.3.2 Relationship Between Nucleotide Consumption Rate and RNA Production Rate

We state a few results used in carrying out the model reduction in Section 4.6. Ideally, we would like to determine the relationship between the rate of production of RNA and the rate of consumption of nucleotides in the full model (4.1). The approach we will take involves first partitioning the corresponding mass action equations (4.2) into subsets of equations as follows:

$$\frac{\mathrm{d}\xi}{\mathrm{d}t} = F(\xi, [\mathrm{N}]), \qquad \xi \in \mathbb{R}^{2n+3}_{\geq 0}, \tag{4.7a}$$

$$\frac{\mathrm{d}[\mathrm{N}]}{\mathrm{d}t} = G(\xi, [\mathrm{N}]), \tag{4.7b}$$

$$\frac{\mathrm{d}[\mathrm{m_n}]}{\mathrm{d}t} = H(\xi), \tag{4.7c}$$

where ξ is a vector comprising the concentrations of all the species except the completed RNA transcript $\mathrm{m_n}$ and the free nucleotides, N. F, G and H are functions defined using mass action kinetics, which, with their respective arguments, give the rates of change of the vector ξ and the scalars $[\mathrm{N}]$ and $[\mathrm{m_n}]$. This decomposition allows us to consider the rate of production of RNA, a species that does not participate anywhere else in the network, separately from the rate of consumption of the nucleotides, which affect dynamics of many reactions in the network. In particular, we note that in equations (4.7a)–(4.7c), the functions F, G and H do not have $[\mathrm{m_n}]$ as an argument, while both F and G depend on $[\mathrm{N}]$.

We will show that when the concentration of nucleotides, $[\mathrm{N}]$ as an argument of F in Equation (4.7a) is held constant, the trajectories of ξ reach an asymptotically stable equilibrium, ξ_e. At this equilibrium, which can be thought of as an operating point for the local dynamics of $[\mathrm{N}]$ in Equation (4.7b) and of $[\mathrm{m_n}]$ in Equation (4.7c), the rate of consumption of nucleotides is proportional to the rate of production of RNA.

The assumption to hold the concentration of some species constant in order to determine the properties of a network requires some justification. To this end, we note that it has been used in the theory of chemical reactions, for instance by Feinberg ([18], Remark 4.3.1), who notes that when a species is in great excess, then over some 'reason-

able' time-scale, one could expect the concentration of the excess species to not change appreciably, while the remaining species can display non-constant dynamics. One domain where nucleotide concentration is in excess for most of the duration of interest is in cell-free extracts, which were the primary motivation for this study. Another domain of relevance for the constancy of nucleotides is in cells, where nucleotide concentrations are regulated, and one might wish to calculate the consumption rate to obtain a measure for the loading of the cell's metabolic machinery.

We now state a proposition which establishes the relationship between the rate of production of RNA and that of the consumption of nucleotides at this steady state, and furthermore provides steady state relationships among species concentrations, which will turn out to be useful for the model reduction procedure in Section 4.6.

Proposition 3. *Consider the full model given by equations* (4.1) *and* (4.2)*, and its decomposition into subsystems F, G and H given by equations (4.7a)–(4.7c). When the nucleotide concentration is held constant, $[\mathrm{N}] = [\mathrm{N}]_{const}$, in the subsystem F, the trajectories of ξ reach an asymptotically stable equilibrium, ξ_e, in the sense of Remarks 12 and 13. Furthermore, substituting ξ_e and $[\mathrm{N}]_{const}$ into the subsystems G and H gives the relationship*

$$\frac{\mathrm{d}[\mathrm{N}]_{uninc}}{\mathrm{d}t} = -n\frac{\mathrm{d}[\mathrm{m_n}]}{\mathrm{d}t}, \tag{4.8}$$

where n is the length of the RNA, $\mathrm{m_n}$, in nucleotides, and $[\mathrm{N}]_{uninc}$ is the total concentration of nucleotides not incorporated into RNA, i.e., $[\mathrm{N}]_{uninc} \triangleq [\mathrm{N}] + \sum_{k=1}^{n}[\mathrm{P{:}D_k{:}m_{k-1}{:}N}]$.

Proof. We first prove the stability of the equilibrium ξ_e of the subsystem

$$\frac{\mathrm{d}\xi}{\mathrm{d}t} = F(\xi, [\mathrm{N}]_{\mathrm{const}}). \tag{4.9}$$

Using the technique from Feinberg ([18], Section 4.3), we can write out the dynamical Equation (4.9) as a chemical reaction network with certain rate constants modified by the

constant scalar $[\mathrm{N}]_{\mathrm{const}}$

$$
\begin{aligned}
\mathrm{P}+\mathrm{D} &\underset{k_{Pr}}{\overset{k_{Pf}}{\rightleftharpoons}} \mathrm{P{:}D_1{:}m_0} \\
\mathrm{P{:}D_1{:}m_0} &\underset{k_{Nr}}{\overset{k_{Nf}[\mathrm{N}]_{\mathrm{const}}}{\rightleftharpoons}} \mathrm{P{:}D_1{:}m_0{:}N} \\
\mathrm{P{:}D_1{:}m_0{:}N} &\xrightarrow{k_{tx}} \mathrm{P{:}D_2{:}m_1} \\
&\vdots \\
\mathrm{P{:}D_n{:}m_{n-1}} &\underset{k_{Nr}}{\overset{k_{Nf}[\mathrm{N}]_{\mathrm{const}}}{\rightleftharpoons}} \mathrm{P{:}D_n{:}m_{n-1}{:}N} \\
\mathrm{P{:}D_n{:}m_{n-1}{:}N} &\xrightarrow{k_{tx}} \mathrm{P{:}D_t} \\
\mathrm{P{:}D_t} &\xrightarrow{k_{\mathrm{term}}} \mathrm{P}+\mathrm{D}.
\end{aligned}
\tag{4.10}
$$

According to Theorem 2, if we can show that the network given by (4.10) has deficiency zero and is weakly reversible, we would have shown that it possesses an asymptotically stable equilibrium. The set of complexes in the network is $\{\mathrm{P{+}D}, \mathrm{P{:}D_1{:}m_0}, \mathrm{P{:}D_1{:}m_0{:}N}, \ldots, \mathrm{P{:}D_n{:}m_{n-1}}, \mathrm{P{:}D_n{:}m}$ Thus, there are $c \triangleq 2n+2$ complexes in the network. Note that there is a cyclic path through the set of complexes, given by $\mathrm{P{+}D} \to \mathrm{P{:}D_1{:}m_0} \to \mathrm{P{:}D_1{:}m_0{:}N} \to \cdots \to \mathrm{P{:}D_n{:}m_{n-1}} \to \mathrm{P{:}D_n{:}m_{n-1}{:}N} \to \mathrm{P{:}D_t} \to \mathrm{P{+}D}$. Thus, the network is weakly reversible, and has only one linkage class ($l = 1$). Finally, we compute the rank of the stoichiometric matrix as follows. The network can be written in matrix form as

$$
\frac{\mathrm{d}\xi}{\mathrm{d}t} = M\, v(\xi, [\mathrm{N}]_{\mathrm{const}}),
$$

where

$$\frac{\mathrm{d}\xi}{\mathrm{d}t} = \frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix} [\mathrm{P}] \\ [\mathrm{D}] \\ [\mathrm{P{:}D_1{:}m_0}] \\ [\mathrm{P{:}D_1{:}m_0{:}N}] \\ [\mathrm{P{:}D_2{:}m_1}] \\ [\mathrm{P{:}D_2{:}m_1{:}N}] \\ \vdots \\ [\mathrm{P{:}D_n{:}m_{n-1}}] \\ [\mathrm{P{:}D_n{:}m_{n-1}{:}N}] \\ [\mathrm{P{:}D_t}] \end{pmatrix}, \qquad \nu(\xi, [\mathrm{N}]_{\mathrm{const}}) = \begin{pmatrix} k_{Pf}[\mathrm{P}][\mathrm{D}] \\ k_{Pr}[\mathrm{P{:}D_1{:}m_0}] \\ k_{Nf}[\mathrm{N}]_{\mathrm{const}}[\mathrm{P{:}D_1{:}m_0}] \\ k_{Nr}[\mathrm{P{:}D_1{:}m_0{:}N}] \\ k_{tx}[\mathrm{P{:}D_1{:}m_0{:}N}] \\ k_{Nf}[\mathrm{N}]_{\mathrm{const}}[\mathrm{P{:}D_2{:}m_1}] \\ k_{Nr}[\mathrm{P{:}D_2{:}m_1{:}N}] \\ k_{tx}[\mathrm{P{:}D_2{:}m_1{:}N}] \\ \vdots \\ k_{Nf}[\mathrm{N}]_{\mathrm{const}}[\mathrm{P{:}D_n{:}m_{n-1}}] \\ k_{Nr}[\mathrm{P{:}D_n{:}m_{n-1}{:}N}] \\ k_{tx}[\mathrm{P{:}D_n{:}m_{n-1}{:}N}] \\ k_{\mathrm{term}}[\mathrm{P{:}D_t}] \end{pmatrix}, \tag{4.11}$$

and

$$M = \begin{array}{c} \\ r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \\ \vdots \\ r_{2n+1} \\ r_{2n+2} \\ r_{2n+3} \end{array} \begin{array}{c} \begin{array}{cccccccccccc} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 & c_8 & \dots & c_{3n} & c_{3n+1} & c_{3n+2} & c_{3n+3} \end{array} \\ \begin{pmatrix} -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & & 0 & 0 & 0 & 1 \\ -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & & 0 & 0 & 0 & 1 \\ 1 & -1 & -1 & 1 & 0 & 0 & 0 & 0 & & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & -1 & 0 & 0 & 0 & & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 1 & 0 & & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & -1 & & 0 & 0 & 0 & 0 \\ & & & & & & & & \ddots & & & & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & & 1 & -1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & & 0 & 0 & 1 & -1 \end{pmatrix} \end{array},$$

where we denote the rows and columns of the matrix M using r_i and c_i, $i = 1, \dots 2n+2$, respectively. We determine the rank of M as follows. Remove r_1 since it is a duplicate of r_2, and therefore does not affect the rank of the matrix. Also remove columns $\{c_2, c_4, c_7, \dots, c_{3i+1}, \dots, c_{3n+1}\}$, which are all scalar multiples of the columns preceding them. We are then left with the $2n+2 \times 2n+2$ matrix

$$\tilde{M} = \begin{array}{c} \\ \tilde{r}_2 \\ \tilde{r}_3 \\ \tilde{r}_4 \\ \tilde{r}_5 \\ \tilde{r}_6 \\ \vdots \\ \tilde{r}_{2n+1} \\ \tilde{r}_{2n+2} \\ \tilde{r}_{2n+3} \end{array} \begin{array}{c} \begin{array}{ccccccccc} \tilde{c}_1 & \tilde{c}_3 & \tilde{c}_5 & \tilde{c}_6 & \tilde{c}_8 & \dots & \tilde{c}_{3n} & \tilde{c}_{3n+2} & \tilde{c}_{3n+3} \end{array} \\ \begin{pmatrix} -1 & 0 & 0 & 0 & 0 & & 0 & 0 & 1 \\ 1 & -1 & 0 & 0 & 0 & & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & & 0 & 0 & 0 \\ & & & & & \ddots & & & \\ 0 & 0 & 0 & 0 & 0 & & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & & 0 & 1 & -1 \end{pmatrix} \end{array},$$

which has the same rank as M. The sub-matrix $\tilde{M}_1$ obtained by removing $\tilde{c}_{3n+3}$ and $\tilde{r}_{2n+3}$ is lower triangular with nonzero diagonal entries, and thus has a (full) rank of $2n+1$, giving $\mathrm{rank}(M) \geq 2n+1$. We also know that $\mathrm{rank}(M) = \mathrm{rank}(\tilde{M}) \leq 2n+2$. Finally, note that $\tilde{r}_{2n+3}$ can be written as a linear combination of the remaining rows in $\tilde{M}$ as

$$\tilde{r}_{2n+3} = -\sum_{i=2}^{2n+2} \tilde{r}_i.$$

Thus $q \triangleq \mathrm{rank}(M) = 2n+1$. I.e., this network has zero deficiency $\delta = c - l - q = 2n+2-1-(2n+1) = 0$, and is weakly reversible and using Theorem 2, we conclude that there exists a positive equilibrium of the subsystem (4.7a), asymptotically stable relative to its stoichiometric compatibility class.

Next, we obtain the relationship between $\mathrm{d}[\mathrm{N_{uninc}}]/\mathrm{d}t$ and $\mathrm{d}[\mathrm{m_n}]/\mathrm{d}t$. Note that

$$\frac{\mathrm{d}[\mathrm{m_n}]}{\mathrm{d}t} = k_{tx}[\mathrm{P{:}D_n{:}m_{n-1}{:}N}], \tag{4.12}$$

$$\begin{aligned}
\frac{\mathrm{d}[\mathrm{N_{uninc}}]}{\mathrm{d}t} &= \frac{\mathrm{d}\left([\mathrm{N}] + \sum_{i=1}^{n}[\mathrm{P{:}D_i{:}m_{i-1}{:}N}]\right)}{\mathrm{d}t} \\
&= \frac{\mathrm{d}[\mathrm{N}]}{\mathrm{d}t} + \sum_{i=1}^{n}\frac{\mathrm{d}[\mathrm{P{:}D_i{:}m_{i-1}{:}N}]}{\mathrm{d}t} \\
&= \sum_{k=1}^{n}\left(k_{Nr}[\mathrm{P{:}D_k{:}m_{k-1}{:}N}] - k_{Nf}[\mathrm{P{:}D_k{:}m_{k-1}}][\mathrm{N}]\right) \\
&\qquad + \sum_{k=1}^{n}\left(k_{Nf}[\mathrm{P{:}D_k{:}m_{k-1}}][\mathrm{N}] - k_{Nr}[\mathrm{P{:}D_k{:}m_{k-1}{:}N}] - k_{tx}[\mathrm{P{:}D_k{:}m_{k-1}{:}N}]\right) \\
&= -k_{tx}\sum_{k=1}^{n}[\mathrm{P{:}D_k{:}m_{k-1}{:}N}].
\end{aligned} \tag{4.13}$$

For the model (4.7a) to be at steady state, the net flux into and out of every species must be zero, and individual fluxes are constant in time. Consider a set of three consecutive reactions from the subsystem (4.10) at an arbitrary $[\mathrm{N}]_{\mathrm{const}}$

$$\mathrm{P{:}D_{i-1}{:}m_{i-2}{:}N} \xrightarrow{k_{tx}} \mathrm{P{:}D_i{:}m_{i-1}}, \tag{4.14}$$

$$\mathrm{P{:}D_i{:}m_{i-1}} \underset{k_{Nr}}{\overset{k_{Nf}[\mathrm{N}]_{\mathrm{const}}}{\rightleftharpoons}} \mathrm{P{:}D_i{:}m_{i-1}{:}N}, \tag{4.15}$$

$$\mathrm{P{:}D_i{:}m_{i-1}{:}N} \xrightarrow{k_{tx}} \mathrm{P{:}D_{i+1}{:}m_i}. \tag{4.16}$$

Since the instantaneous flux into and out of $\mathrm{P{:}D_i{:}m_{i-1}}$ is zero, the flux in due to (4.14) and the flux out due to the reversible reactions (4.15) must balance, we have

$$k_{tx}[\mathrm{P{:}D_{i-1}{:}m_{i-2}{:}N}] = k_{Nf}[\mathrm{P{:}D_i{:}m_{i-1}}][\mathrm{N}]_{\mathrm{const}} - k_{Nr}[\mathrm{P{:}D_i{:}m_{i-1}{:}N}]. \tag{4.17}$$

Similarly, considering the species $\mathrm{P{:}D_i{:}m_{i-1}{:}N}$ in (4.15) and (4.16), we have

$$k_{Nf}[\mathrm{P{:}D_i{:}m_{i-1}}][\mathrm{N}]_{\mathrm{const}} - k_{Nr}[\mathrm{P{:}D_i{:}m_{i-1}{:}N}] = k_{tx}[\mathrm{P{:}D_i{:}m_{i-1}{:}N}]. \tag{4.18}$$

Thus,

$$[\mathrm{P{:}D_{i-1}{:}m_{i-2}{:}N}] = [\mathrm{P{:}D_{i}{:}m_{i-1}{:}N}] \tag{4.19}$$

$$[\mathrm{P{:}D_{i-1}{:}m_{i-2}}] = [\mathrm{P{:}D_{i}{:}m_{i-1}}], \tag{4.20}$$

and by induction, we have that for all i, j in $\{1, 2, \ldots n\}$,

$$[\mathrm{P{:}D_{i}{:}m_{i-1}}] = [\mathrm{P{:}D_{j}{:}m_{j-1}}], \tag{4.21}$$

$$[\mathrm{P{:}D_{i}{:}m_{i-1}{:}N}] = [\mathrm{P{:}D_{j}{:}m_{j-1}{:}N}]. \tag{4.22}$$

Thus, Equation (4.13) can be reduced to

$$\frac{\mathrm{d}[\mathrm{N_{uninc}}]}{\mathrm{d}t} = -k_{tx} \sum_{k=1}^{n} [\mathrm{P{:}D_{k}{:}m_{k-1}{:}N}] \tag{4.23}$$

$$= -n \cdot k_{tx} [\mathrm{P{:}D_{n}{:}m_{n-1}{:}N}] \tag{4.24}$$

$$= -n \frac{\mathrm{d}[\mathrm{m_{n}}]}{\mathrm{d}t}, \tag{4.25}$$

which completes the proof. □

4.4 Overview of Time-Scale Separation in Chemical Kinetics via Singular Perturbation Theory

4.4.1 Singular Perturbation Theory for Chemical Reaction Networks

Singular perturbation theory has been used widely to decompose models of physical systems containing multiple temporal and spatial scales into subsystems at those scales [19]. This decomposition has been carried out for chemical systems too, where some reactions proceed much more quickly than others, or there exist transient short lived species [41, 67, 68].

To begin, we introduce the notion of decomposing a system into *slow* and *fast subsystems*, operating at two different time-scales. Such a decomposition is only possible if we

can write the model for the system in the *standard singular perturbation form*:

$$\frac{\mathrm{d}x}{\mathrm{d}t} = f(t, x, z, \epsilon), \qquad x(0) = x_0, \qquad x \in \mathbb{R}^n, \tag{4.26}$$

$$\epsilon\frac{\mathrm{d}z}{\mathrm{d}t} = g(t, x, z, \epsilon), \qquad z(0) = z_0, \qquad z \in \mathbb{R}^m, \tag{4.27}$$

where ϵ is a small positive scalar, and f, g are sufficiently many times continuously differentiable in their arguments (t, x, z, ϵ).

The small parameter ϵ in Equation (4.27) is used to capture the effects of large reaction rate constants in chemical kinetics, which lead to fast transient dynamics. These fast dynamics are modeled by the variable z, whose rate of change gets scaled by $1/\epsilon$, and hence becomes very large. Being able to apply tools from singular perturbation theory involves bringing the mass action dynamical equations into the above form as a necessary prerequisite. Singular perturbation theory also requires that there exists at least one asymptotically stable equilibrium (isolated from any others that might exist) to which the trajectories of the variable z, for each allowable x, converge. We defer a discussion of the properties of the equilibria for the moment, and focus on finding a set of state variables that allow us to bring the system into the standard form in the first place. To this end, we will discuss why species concentrations are not appropriate to use as a state variables for the purposes of bringing a system into the standard form, and elaborate on a variable transformation which provides a better system of coordinates.

4.4.1.1 Nonexplicit Time-Scale-Separation

In many applications exhibiting two-time-scale behavior, it is not possible to partition the natural state variables into fast and slow variables to bring the system into the standard singular perturbation form [39]. This is despite the fact that the variables exhibit two-time-scale behavior, where an initial fast transient is followed by slow evolution on an equilibrium manifold [41]. This occurs because these natural state variables are in general a combination of both fast and slow effects, and therefore exhibit *nonexplicit* time-scale-separation. In chemical kinetics, the ODE models are written with species concentrations

as the natural variables, and finding transformations from the natural coordinates to a set of coordinates where the model may be written in the standard form is highly nontrivial. The structural reason for nonexplicit time-scale-separation in chemical kinetics is that each species in a model may participate in both fast and slow reactions.

The problem of finding coordinate changes to allow such models to be written in the standard form has received some attention in the literature. Kokotovic, Khalil and O'Reilly [39] gave a general prescription for constructing such coordinate transformations, while specific ad-hoc transformations for chemical reaction networks were studied in [6,57]. The first systematic procedure for finding a linear coordinate transformation was developed by Van Breusegem and Bastin [67]. These authors first partition the stoichiometric matrix into block matrices corresponding to species participating in fast reactions, both fast and slow reactions, and only slow reactions, and then use these block matrices to construct the desired invertible coordinate transformation. The works of Kumar, Christofidis and Daoutidis [41] and Vora and Daoutidis [68] take an entirely different approach, and develop a very general framework for deriving a family of coordinate transformations that bring nonexplicit two-time-scale models into the standard form. The main idea behind their method involves giving a set of constraints that implicitly define the equilibrium manifold and computing an upper bound on the dimension of this manifold. This allows them to pick an arbitrary subset or transformation of state variables from the original set, and construct an explicit representation of the reduced order model that, after an initial fast transient, evolves on the equilibrium manifold. The generality of this framework arises from the fact that the method is not limited to *isothermal* reaction networks, where the stoichiometric matrix is constant in time, but can instead be used to reduce nonisothermal reaction networks, and even more general systems, like those modeling the dynamics of heat exchange. For isothermal reaction networks, these studies give a method for the explicit construction of the slow variables as a set of linear combinations defined by a basis of the left null space of the subset of the stoichiometric matrix corresponding to the fast reactions.

All of these studies suffer from a set of limitations. We note that the ad-hoc methods

mentioned above require human intuition to find the appropriate coordinate transformation, and such methods do not scale well beyond the simplest models. The remaining methods suffer from the limitation that the transformed state variables do not have a physical interpretation, and are fairly complex. The methods in [41, 68] further suffer from the limitation that even in the case of isothermal reactions, the transformation is nonlinear, and finding the standard form involves inverting this transformation on the equilibrium manifold. Such an inversion is highly nontrivial, and could only be found for the simpler examples in their studies (see, for example, the final step in the esterification example in [68], where no attempt to invert the transformation is made). In this chapter, we provide a general construction for finding a transformation that is simple to construct, allows the transformed variables to have a physical interpretation, and gives a completely explicit representation of the standard form in the transformed coordinates.

4.4.2 Species Concentrations as State Variables

The typical way of reducing an ODE model of a reaction network with two-time-scale behavior into the standard form is to separate the set of mass-action differential equations into those belonging to species participating in slow reactions only, and those participating in fast and (possibly) slow reactions. The enzymatic reaction is a prototypical example of this approach. Consider the reaction

$$\mathrm{Enz} + \mathrm{Su} \underset{\mathrm{d}}{\overset{\mathrm{a}}{\rightleftharpoons}} \mathrm{Cpx} \xrightarrow{k} \mathrm{Pdt} + \mathrm{Enz} \tag{4.28}$$

where Enz is the enzyme, Su the substrate, Cpx the complex, and Pdt the product formed. The binding-unbinding is assumed to be much faster than the catalysis reaction ($a,\ d \gg k$). In differential equations, this is

$$\begin{aligned}
\frac{\mathrm{d}[\mathrm{Cpx}]}{\mathrm{d}t} &= -k[\mathrm{Cpx}] - d[\mathrm{Cpx}] + a[\mathrm{Enz}][\mathrm{Su}] = -\frac{\mathrm{d}[\mathrm{Enz}]}{\mathrm{d}t}, \\
\frac{\mathrm{d}[\mathrm{Su}]}{\mathrm{d}t} &= d[\mathrm{Cpx}] - a[\mathrm{Enz}][\mathrm{Su}], \\
\frac{\mathrm{d}[\mathrm{Pdt}]}{\mathrm{d}t} &= k[\mathrm{Cpx}].
\end{aligned} \tag{4.29}$$

Using $[\mathrm{Enz}] = E_0 - [\mathrm{Cpx}]$, $[\mathrm{Su}] = S_0 - [\mathrm{Cpx}] - [\mathrm{Pdt}]$, $\tau = kt$, $K_d = d/a$, $\epsilon = k/d$ and $K_d x = X$ where $X \in \{[\mathrm{Cpx}],\ [\mathrm{Enz}],\ [\mathrm{Su}],\ [\mathrm{Pdt}],\ S_0,\ E_0\}$ and correspondingly $x \in \{c, e, s, p, s_0, e_0\}$, we arrive at the nondimensionalized model

$$\begin{aligned} \epsilon \frac{\mathrm{d}c}{\mathrm{d}\tau} &= -c - \epsilon c + (e_0 - c)(s_0 - c - p), \\ \frac{\mathrm{d}p}{\mathrm{d}\tau} &= c. \end{aligned} \tag{4.30}$$

Setting $\epsilon = 0$ and using $S_0 - [\mathrm{Cpx}] - [\mathrm{Pdt}] \approx S_0 - [\mathrm{Pdt}]$, allows us to arrive at reduced system

$$\frac{\mathrm{d}[\mathrm{Pdt}]}{\mathrm{d}t} = k[\mathrm{Cpx}] = k\frac{E_0(S_0 - [\mathrm{Pdt}])}{(S_0 - [\mathrm{Pdt}]) + K_d}. \tag{4.31}$$

The reason we are able to write this model in the standard form (4.30) is that the species Pdt only takes part in the slow reaction (rate = k), allowing it to be part of the slow subsystem, while the other species (Enz, Su, Cpx) take part in at least one fast reaction, making them part of the fast subsystem.

The same trend appears when we look at the transcription model given by Equations (4.7a) – (4.7c). For compactness of notation, let us define $\bar{\eta} \equiv [\mathrm{N}]$, $\bar{\gamma} \equiv [\mathrm{P{:}D_t}]$, $\bar{\rho} \equiv [\mathrm{P}]$, $\bar{d} \equiv [\mathrm{D}]$, $\bar{m}_n \equiv [\mathrm{m_n}]$ and for $i = 1, \ldots, n$, denote $\bar{v}_i \equiv [\mathrm{P{:}D_i{:}m_{i-1}}]$ and $\bar{w}_i \equiv [\mathrm{P{:}D_i{:}m_{i-1}{:}N}]$. We may write the model as

$$
\begin{aligned}
\frac{\mathrm{d}\bar{d}}{\mathrm{d}t} &= k_{\text{term}}\bar{\gamma} - k_{Pf}\bar{\rho}\bar{d} + k_{Pr}\bar{\nu}_1, \\
\frac{\mathrm{d}\bar{\rho}}{\mathrm{d}t} &= k_{\text{term}}\bar{\gamma} - k_{Pf}\bar{\rho}\bar{d} + k_{Pr}\bar{\nu}_1, \\
\frac{\mathrm{d}\bar{\nu}_1}{\mathrm{d}t} &= k_{Pf}\bar{\rho}\bar{d} - k_{Pr}\bar{\nu}_1 - k_{Nf}\bar{\nu}_1\bar{\eta} + k_{Nr}\bar{w}_1, \\
\frac{\mathrm{d}\bar{w}_1}{\mathrm{d}t} &= k_{Nf}\bar{\nu}_1\bar{\eta} - k_{Nr}\bar{w}_1 - k_t\bar{w}_1, \\
\frac{\mathrm{d}\bar{\nu}_2}{\mathrm{d}t} &= k_t\bar{w}_1 - k_{Nf}\bar{\nu}_2\bar{\eta} + k_{Nr}\bar{w}_2, \\
\frac{\mathrm{d}\bar{w}_2}{\mathrm{d}t} &= k_{Nf}\bar{\nu}_2\bar{\eta} - k_{Nr}\bar{w}_2 - k_t\bar{w}_2, \\
&\vdots \\
\frac{\mathrm{d}\bar{\nu}_n}{\mathrm{d}t} &= k_t\bar{w}_2 - k_{Nf}\bar{\nu}_3\bar{\eta}k_{Nr}\bar{w}_3, \\
\frac{\mathrm{d}\bar{w}_n}{\mathrm{d}t} &= k_{Nf}\bar{\nu}_3\bar{\eta} - k_{Nr}\bar{w}_3 - k_t\bar{w}_3, \\
\frac{\mathrm{d}\bar{\gamma}}{\mathrm{d}t} &= k_t\bar{w}_3 - k_{\text{term}}\bar{\gamma}, \\
\frac{\mathrm{d}\bar{\eta}}{\mathrm{d}t} &= \sum_{i=1}^{n} k_{Nr}\bar{w}_i - k_{Nf}\bar{\nu}_i\bar{\eta}, \\
\frac{\mathrm{d}\bar{m}_n}{\mathrm{d}t} &= k_t\bar{w}_n,
\end{aligned}
\tag{4.32}
$$

and use the nondimensionalization scheme $\epsilon = k_t/k_{Nr}$, $\tau = tk_t$, $\alpha_{Pf} = k_{Pf}/k_{Nf}$, $\alpha_{Pr} = k_{Pr}/k_{Nr}$, $\alpha_{term} = k_{\text{term}}/k_t$, $\bar{u} = k_{Nr}/k_{Nf}u$ for all $u \in \{\rho, d, \gamma, \eta, m_n, \nu_1, \ldots, \nu_n, w_1, \ldots, w_n\}$ to

obtain the *nondimensionalized model*

$$
\begin{aligned}
\frac{\mathrm{d}\gamma}{\mathrm{d}\tau} &= w_n - \alpha_{term}\gamma, \\
\frac{\mathrm{d}m_n}{\mathrm{d}\tau} &= w_n, \\
\epsilon\frac{\mathrm{d}d}{\mathrm{d}\tau} &= \epsilon\alpha_{term}\gamma - \alpha_{Pf}\rho d + \alpha_{Pr}v_1, \\
\epsilon\frac{\mathrm{d}\rho}{\mathrm{d}\tau} &= \epsilon\alpha_{term}\gamma - \alpha_{Pf}\rho d + \alpha_{Pr}v_1, \\
\epsilon\frac{\mathrm{d}v_1}{\mathrm{d}\tau} &= \alpha_{Pf}\rho d - \alpha_{Pr}v_1 - v_1\eta + w_1, \\
\epsilon\frac{\mathrm{d}w_1}{\mathrm{d}\tau} &= -\epsilon w_1 + v_1\eta - w_1, \\
\epsilon\frac{\mathrm{d}v_2}{\mathrm{d}\tau} &= \epsilon w_1 - v_2\eta + w_2, \\
\epsilon\frac{\mathrm{d}w_2}{\mathrm{d}\tau} &= -\epsilon w_2 + v_2\eta - w_2, \\
&\vdots \\
\epsilon\frac{\mathrm{d}v_n}{\mathrm{d}\tau} &= \epsilon w_{n-1} - v_n\eta + w_n, \\
\epsilon\frac{\mathrm{d}w_n}{\mathrm{d}\tau} &= -\epsilon w_n + v_n\eta - w_n, \\
\epsilon\frac{\mathrm{d}\eta}{\mathrm{d}\tau} &= \sum_{i=1}^{n}(-v_i\eta + w_i).
\end{aligned}
\tag{4.33}
$$

We see here that only γ and m_n can be separated into the slow subsystem, despite every other species participating in slow reactions as well. If we were to consider a network whereby every species took part in at least one fast reaction, then every equation in the corresponding differential equations would have an epsilon in front of it. Setting $\epsilon = 0$ would lead to a loss of all slow dynamics from our model, and we would only be able to make statements about what happens at the equilibrium for a given set of slow variable values. Due to this nonexplicit time-scale-separation, it would not be possible to have a set of reduced differential equations that can be solved or simulated. In the next section, we develop an invertible transformation from the species concentration coordinates to a different set of coordinates, and show how in these coordinates, the fast and slow dynamics can be separated, and the standard form derived.

4.4.3 Reaction Extents as a Natural and Physically Interpretable Coordinate System

4.4.3.1 Preliminary Reduction by Conservation Laws

The coordinate transformation we are interested in will depend on the stoichiometric matrix being invertible. In this section, we will discuss the procedure of removing linear dependencies in the rows and columns of the stoichiometric matrix. Consider again the ODE description of the chemical reaction system given by Equation (4.6), where $S \in \mathbb{R}^{s\times r}$ is the stoichiometric matrix, and $v \in \mathbb{R}^r$ is the reaction rate velocity vector. We assume that whenever a reversible reaction exists in the system, such that there is a corresponding pair of columns of S which are negatives of each other, the column corresponding to the backward reaction has been removed from the matrix. Furthermore, the element of v corresponding to the backward reaction rate is removed from the vector, and subtracted from the element corresponding to the forward rate, such that we have the net forward rate of the reversible reaction. For example, in the enzymatic reaction in Equation (4.28), we would transform the ODE description

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix}[\mathrm{Enz}]\\ [\mathrm{Su}]\\ [\mathrm{Cpx}]\\ [\mathrm{Pdt}]\end{bmatrix} = \begin{bmatrix}-1 & 1 & 1\\ -1 & 1 & 0\\ 1 & -1 & -1\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}a[\mathrm{Enz}][\mathrm{Su}]\\ d[\mathrm{Cpx}]\\ k[\mathrm{Cpx}]\end{bmatrix}, \tag{4.34}$$

where the first two columns of the stoichiometric matrix are negatives of each other, to

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix}[\mathrm{Enz}]\\ [\mathrm{Su}]\\ [\mathrm{Cpx}]\\ [\mathrm{Pdt}]\end{bmatrix} = \begin{bmatrix}-1 & 1\\ -1 & 0\\ 1 & -1\\ 0 & 1\end{bmatrix}\begin{bmatrix}a[\mathrm{Enz}][\mathrm{Su}]-d[\mathrm{Cpx}]\\ k[\mathrm{Cpx}]\end{bmatrix}, \tag{4.35}$$

where the second column has been removed, and the reaction rate vector has been transformed accordingly. We also assume that once all such reversible reaction column pairs

have been converted to single columns, there are no other linear dependencies in the columns of S, and the new number of columns is r'. We expect that a more general case for dependencies between net reactions may be be constructed based on the idea of *reaction pathways*, which are introduced in [58, 59], and refer to reaction cycles arising in the reaction network diagram due to the existence of these linear dependencies.

Once we have removed the linear dependencies in the columns of S, we are ready to do the same for the rows. The argument presented is based on [9] and [56]. Since the columns of S are linearly independent, $\mathrm{rank}(S) = r'$, and there are r' linearly independent rows in S, such that the remaining rows can be written as linear combinations of these r' rows. We may define a full row rank matrix $P \in \mathbb{R}^{r' \times s}$ that picks out the r' linearly independent rows of S. Each row in P is made of all zeros except one element, which is a one. Furthermore, none of the r' rows are equal. By the rank-nullity theorem, we also know that the dimension of the left nullspace of S is $s - r'$, and thus we may find $s - r'$ row vectors which form a basis for this space. Arranging these row vectors into a full row rank matrix $H \in \mathbb{R}^{(s-r') \times n}$, we have $HS = 0$.

We may now use these matrices to define a reduced stoichiometric matrix $S_r \in \mathbb{R}^{r' \times r'}$ such that

$$\begin{bmatrix} S_r \\ 0 \end{bmatrix} \triangleq \begin{bmatrix} P \\ H \end{bmatrix} S, \tag{4.36}$$

where $\begin{bmatrix} P \\ H \end{bmatrix} \in \mathbb{R}^{s \times s}$ is invertible, and 0 is a matrix of zeros of appropriate dimensions.

Similarly, the species concentration vector may be transformed as

$$\begin{bmatrix} x_i \\ x_d \end{bmatrix} \triangleq \begin{bmatrix} P \\ H \end{bmatrix} x, \tag{4.37}$$

where $x_i \in \mathbb{R}^d$ are the dynamic variables in the reduced model, and $x_d \in \mathbb{R}^m$ are combinations of species concentrations which end up being constant, and are determined by the initial concentrations in the system. To see this, apply $\begin{bmatrix} P \\ H \end{bmatrix}$ to Equation (4.6)

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} x_i \\ x_d \end{bmatrix} = \begin{bmatrix} P \\ H \end{bmatrix} \frac{\mathrm{d}x}{\mathrm{d}t} \tag{4.38}$$

$$= \begin{bmatrix} P \\ H \end{bmatrix} S\, v(x) \tag{4.39}$$

$$= \begin{bmatrix} S_r\, v(x) \\ 0 \end{bmatrix} \tag{4.40}$$

$$= \begin{bmatrix} S_r\, v_r(x_i, x_d) \\ 0 \end{bmatrix}, \tag{4.41}$$

$$\begin{bmatrix} x_{i0} \\ x_{d0} \end{bmatrix} \triangleq \begin{bmatrix} x_i \\ x_d \end{bmatrix}(0) \tag{4.42}$$

$$= \begin{bmatrix} P \\ H \end{bmatrix} x_0, \tag{4.43}$$

where

$$v_r(x_i, x_d) \triangleq v(x)\Big|_{x=\left[\begin{smallmatrix} P \\ H \end{smallmatrix}\right]^{-1}\left[\begin{smallmatrix} x_i \\ x_d \end{smallmatrix}\right]}, \tag{4.44}$$

so that we have the reduced system

$$\begin{aligned} \dot{x_i} &= S_r\, v_r(x_i,\, x_d), & x_i(0) &= x_{i0}, \\ \dot{x_d} &= 0, & x_d(t) &= x_{d0} \quad \forall t \geq 0. \end{aligned} \tag{4.45}$$

For example, applying this methodology to the enzymatic reaction (4.28), we may choose $P = \left[\begin{smallmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{smallmatrix}\right]$ and $H = \left[\begin{smallmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{smallmatrix}\right]$, so that $x_i = \left[\begin{smallmatrix} [\mathrm{Cpx}], & [\mathrm{Pdt}] \end{smallmatrix}\right]^T$ and $x_d = \left[\begin{smallmatrix} [\mathrm{Enz}]+[\mathrm{Cpx}], & [\mathrm{Su}]+[\mathrm{Cpx}]+[\mathrm{Pdt}] \end{smallmatrix}\right]^T$ and the reduced system (4.45) can be written:

$$\begin{aligned} &\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} [\mathrm{Cpx}] \\ [\mathrm{Pdt}] \end{bmatrix} = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} a[\mathrm{Enz}][\mathrm{Su}] - d[\mathrm{Cpx}] \\ k[\mathrm{Cpx}] \end{bmatrix}, \quad \begin{bmatrix} [\mathrm{Cpx}] \\ [\mathrm{Pdt}] \end{bmatrix}(0) = \begin{bmatrix} C_0 \\ P_0 \end{bmatrix}, \\ &\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} [\mathrm{Enz}] + [\mathrm{Cpx}] \\ [\mathrm{Su}] + [\mathrm{Cpx}] + [\mathrm{Pdt}] \end{bmatrix} = 0, \qquad \begin{bmatrix} [\mathrm{Enz}] + [\mathrm{Cpx}] \\ [\mathrm{Su}] + [\mathrm{Cpx}] + [\mathrm{Pdt}] \end{bmatrix}(t) = \begin{bmatrix} E_0 \\ S_0 \end{bmatrix}, \ \forall t \geq 0 \end{aligned} \tag{4.46}$$

4.4.3.2 Transforming to Reaction Coordinates

Let us define a coordinate transformation from the $x_i \in \mathbb{R}^{r'}$ variables to a new set of variables $R \in \mathbb{R}^{r'}$,

$$\begin{bmatrix} R \\ x_d \end{bmatrix} \triangleq \left[\begin{array}{c|c} S_r^{-1} & 0 \\ \hline 0 & I_{s-r'} \end{array}\right] \begin{bmatrix} x_i \\ x_d \end{bmatrix}, \tag{4.47}$$

where $I_{s-r'}$ is the identity matrix of dimension $s-r'$. In this new coordinate system, the ODEs are

$$\begin{aligned} \frac{\mathrm{d}R}{\mathrm{d}t} &= S_r^{-1} \frac{\mathrm{d}x_i}{\mathrm{d}t} \\ &= v\left(\begin{bmatrix} P \\ H \end{bmatrix}^{-1} \begin{bmatrix} S_r R \\ x_d \end{bmatrix} \right) \\ &\triangleq v_{rxn}(R, x_d)\,. \end{aligned} \tag{4.48}$$

One of the sources of multiple time-scales in chemical reaction networks is the widely different orders of magnitudes of the reaction rate parameters. Suppose we partition the elements in the reaction velocity vector v into fast and slow rates, as determined by the reaction rate parameters being large or small. In particular, define two matrices $M_s \in \mathbb{R}^{r_s \times r'}$ and $M_f \in \mathbb{R}^{r_f \times r'}$ that pick out the slow and fast reaction velocities respectively. For example, in the enzymatic reaction, recall that the enzyme-substrate binding-unbinding reactions are considered fast, while the product formation reaction is often slow, so that $(a,\ d \gg k)$. Thus, we can partition the elements of v into the set of fast velocities $\{a[\mathrm{Enz}][\mathrm{Su}] - b[\mathrm{Cpx}]\}$ and that of slow velocities $\{k[\mathrm{Cpx}]\}$, with the matrices $M_s = [\,0\ 1\,]$ and $M_f = [\,1\ 0\,]$. Next, partition R using M_s and M_f as

$$\begin{bmatrix} R_s \\ R_f \end{bmatrix} \triangleq \begin{bmatrix} M_s \\ M_f \end{bmatrix} R, \tag{4.49}$$

and note that the above transformation is invertible. With this partitioning scheme in

place, we may write the model as

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} R_s \\ R_f \end{bmatrix} &= \begin{bmatrix} M_s \\ M_f \end{bmatrix}\frac{\mathrm{d}R}{\mathrm{d}t} \\ &= \begin{bmatrix} M_s \\ M_f \end{bmatrix} \nu\left(\begin{bmatrix} P \\ H \end{bmatrix}^{-1}\begin{bmatrix} S_r\begin{bmatrix} M_s \\ M_f \end{bmatrix}^{-1}\begin{bmatrix} R_s \\ R_f \end{bmatrix} \\ x_d \end{bmatrix}\right) \\ &\triangleq \begin{bmatrix} \nu_s \\ \nu_f \end{bmatrix}\left(R_s, R_f\right).\end{aligned} \tag{4.50}$$

Lastly, defining a small parameter ϵ which isolates the effect of the fast reactions, it can be shown that we can bring the above model into the form

$$\begin{aligned}\frac{\mathrm{d}R_s}{\mathrm{d}t} &= \nu_s\left(R_s, R_f\right), \\ \epsilon\frac{\mathrm{d}R_f}{\mathrm{d}t} &= \bar{\nu}_f\left(R_s, R_f\right),\end{aligned} \tag{4.51}$$

where each element of $\bar{\nu}_f = \frac{1}{\epsilon}\nu_f$ has at least one term independent of ϵ. Note that Equations (4.51) can be nondimensionalized if desired. In the context of the enzymatic reaction, Equation (4.50) is

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} R_s \\ R_f \end{bmatrix} &= \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}\nu\left(\begin{bmatrix} E_0 - R_f + R_s \\ S_0 - R_f \\ R_f - R_s \\ R_s \end{bmatrix}\right) \\ &= \begin{bmatrix} k(R_f - R_s) \\ a(E_0 - R_f + R_s)(S_0 - R_f) - d(R_f - R_s) \end{bmatrix},\end{aligned} \tag{4.52}$$

and consequently, Equation (4.51), after nondimensionalization, is

$$\begin{aligned}\frac{\mathrm{d}r_s}{\mathrm{d}\tau} &= r_f - r_s \\ \epsilon\frac{\mathrm{d}r_f}{\mathrm{d}\tau} &= (e_0 - r_f + r_s)(s_0 - r_f) - (r_f - r_s),\end{aligned} \tag{4.53}$$

where $\epsilon = k/d$, $\tau = kt$, and $r = (a/d)R$ for $R \in \{R_f, R_s, E_0, S_0\}$.

4.4.4 Comparison to the Method of Kumar et al. [41]

In this section, we compare the method developed in [41, 68] to the reaction extents method developed in the previous section. Consider the model of the esterificaton of carboxylic acid described in [68]. The state space model with nonexplicit time-scale separation is given by

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \\ \dot{x}_6 \\ \dot{x}_7 \\ \dot{x}_8 \\ \dot{x}_9 \\ \dot{x}_{10} \\ \dot{x}_{11} \end{bmatrix} = \begin{bmatrix} 0 & 0 & -1 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & -1 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ -1 & 1 & 0 & -1 & 1 & 1 & 1 & -1 & 0 & 1 & -1 & -1 \\ 1 & -1 & 1 & 1 & -1 & 0 & -1 & 1 & -1 & -1 & 1 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \\ r_7 \\ r_8 \\ r_9 \\ r_{10} \\ r_{11} \\ r_{12} \end{bmatrix}, \tag{4.54}$$

where the reaction rates are given by the expressions $r_1 = k_1 x_6 x_4$, $r_2 = k_2 x_5 x_2$, $r_3 = k_3 x_1 x_7$, $r_4 = k_4 x_2 x_4$,$r_5 = k_5 x_1 x_5$, $r_6 = k_6 x_{11} x_9$, $r_7 = k_7 x_5 x_{11}$, $r_8 = k_8 x_7 x_4$, $r_9 = k_9 x_5 x_3$, $r_{10} = k_{10} x_8 x_5$, $r_{11} = k_{11} x_4 x_9$ and $r_{12} = k_{12} x_{10} x_4$. We note that the reactions indexed 3, 6, 9, and 12 are slow, and the remaining are fast.

The last six columns of this stoichiometric matrix are a scalar multiple of the first six, and so we pick the first six columns as the linearly independent columns. The reaction velocity vector is transformed accordingly, with the new rates being $r_1 - r_7, \ldots, r_6 - r_{12}$. Furthermore, we can use Gauss-Jordan elimination to design the matrix H, and pick linearly

independent rows by inspection to get the matrix P. Then, we can write

$$P = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix},$$

$$H = \begin{bmatrix} -1/2 & 1/2 & 0 & 0 & 1/2 & 1/2 & 1 & 0 & 0 & 0 & 0 \\ 1/2 & 1/2 & -1 & 0 & -1/2 & -1/2 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

$$S_r = \begin{bmatrix} 0 & -1 & 0 & -1 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

$$v(x) = \begin{bmatrix} r_1(x) - r_7(x) \\ r_2(x) - r_8(x) \\ r_3(x) - r_9(x) \\ r_4(x) - r_10(x) \\ r_5(x) - r_11(x) \\ r_6(x) - r_12(x) \end{bmatrix} = \begin{bmatrix} k_1 x_6 x_4 - k_7 x_5 x_{11} \\ k_2 x_5 x_2 - k_8 x_7 x_4 \\ k_3 x_1 x_7 - k_9 x_5 x_3 \\ k_4 x_2 x_4 - k_{10} x_8 x_5 \\ k_5 x_1 x_5 - k_{11} x_4 x_9 \\ k_6 x_{11} x_9 - k_{12} x_{10} x_4 \end{bmatrix},$$

and the matrices M_s and M_f, which pick out the slow and fast reaction rates from $v(x)$

respectively, are given by

$$M_s = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

$$M_f = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}.$$

With these definitions, we can bring the system into the standard singular perturbation form by following the method in Section 4.4.3.2. Recall that the coordinate transformation to a new set of variables is $R \triangleq S_r^{-1} P x$, and the transformed system is $\dot{R} = \nu(x)$. We can further partition the entries of R as $\begin{bmatrix} R_s \\ R_f \end{bmatrix} \triangleq \begin{bmatrix} M_s \\ M_f \end{bmatrix} R$. Then, x can be written in terms of R_s and R_f as

$$x(R_s, R_f) = \begin{bmatrix} P \\ H \end{bmatrix}^{-1} \begin{bmatrix} S_r \begin{bmatrix} M_s^T & M_f^T \end{bmatrix} \begin{bmatrix} R_s \\ R_f \end{bmatrix} \\ H x_0 \end{bmatrix}$$

$$= \begin{bmatrix} x_{10} - R_{s1} - R_{f4} - x_{2,0} - x_{7,0} - x_{8,0} + x_{9,0} + x_{10,0} \\ -R_{f2} - R_{f3} \\ R_{s1} + x_{2,0} + x_{3,0} + x_{7,0} + x_{8,0} \\ R_{f2} - R_{f1} - R_{f3} + R_{f4} + R_{s2} + x_{2,0} + x_{4,0} - x_{6,0} + 2x_{8,0} - x_{9,0} - 2x_{10,0} \\ R_{f1} - R_{f2} + R_{f3} - R_{f4} + R_{s1} + x_{5,0} + x_{6,0} + x_{7,0} - x_{8,0} + x_{9,0} + x_{10,0} \\ -R_{f1} \\ R_{f2} - R_{s1} \\ R_{f3} \\ R_{f4} - R_{s2} \\ R_{s2} \\ R_{f1} - R_{s2} + x_{6,0} + x_{10,0} + x_{11,0} \end{bmatrix}.$$

Defining a small parameter $\epsilon = 1/k^*$ such that $k_i/k^* \ll \mathcal{O}(1)$, for $i = 3, 6, 9, 12$, and letting

$k' = k_j/k^*$ for $j = 1,2,4,5,7,8,10,11$, we can write

$$\begin{aligned}
\dot{R}_s &= M_s\, v\left(x\left(R_s, R_f\right)\right) \\
&= \nu_s\left(R_s, R_f\right) \\
&= \begin{bmatrix}
-k_3(R_{f2}-R_{s1})(R_{f4}+R_{s1}-x_{1,0}+x_{2,0}+x_{7,0}+x_{8,0}-x_{9,0}-x_{10,0})-k_9(R_{s1} \\
\qquad +x_{2,0}+x_{3,0}+x_{7,0}+x_{8,0})(R_{f1}-R_{f2}+R_{f3}-R_{f4}+R_{s1}+x_{5,0}+x_{6,0} \\
\qquad +x_{7,0}-x_{8,0}+x_{9,0}+x_{10,0}) \\
k_6(R_{f4}-R_{s2})(R_{f1}-R_{s2}+x_{6,0}+x_{10,0}+x_{11,0})-R_{s2}k_{12}(R_{f2}-R_{f1} \\
\qquad -R_{f3}+R_{f4}+R_{s2}+x_{2,0}+x_{4,0}-x_{6,0}+2x_{8,0}-x_{9,0}-2x_{10,0})
\end{bmatrix},
\end{aligned}$$

$$\begin{aligned}
\dot{R}_f &= M_f\, v\left(x\left(R_s, R_f\right)\right) \\
&= \frac{1}{\epsilon}\,\bar{\nu}_f\left(R_s, R_f\right) \\
&= \frac{1}{\epsilon}\begin{bmatrix}
-R_{f1}k_1'(R_{f2}-R_{f1}-R_{f3}+R_{f4}+R_{s2}+x_{2,0}+x_{4,0}-x_{6,0}+2x_{8,0}-x_{9,0} \\
\qquad -2x_{10,0})-k_7'(R_{f1}-R_{s2}+x_{6,0}+x_{10,0}+x_{11,0})(R_{f1}-R_{f2}+R_{f3} \\
\qquad -R_{f4}+R_{s1}+x_{5,0}+x_{6,0}+x_{7,0}-x_{8,0}+x_{9,0}+x_{10,0}) \\
-k_8'(R_{f2}-R_{s1})(R_{f2}-R_{f1}-R_{f3}+R_{f4}+R_{s2}+x_{2,0}+x_{4,0}-x_{6,0}+2x_{8,0} \\
\qquad -x_{9,0}-2x_{10,0})-k_2'(R_{f2}+R_{f3})(R_{f1}-R_{f2}+R_{f3}-R_{f4} \\
\qquad +R_{s1}+x_{5,0}+x_{6,0}+x_{7,0}-x_{8,0}+x_{9,0}+x_{10,0}) \\
-R_{f3}k_{10}'(R_{f1}-R_{f2}+R_{f3}-R_{f4}+R_{s1}+x_{5,0}+x_{6,0}+x_{7,0}-x_{8,0}+x_{9,0} \\
\qquad +x_{10,0})-k_4'(R_{f2}+R_{f3})(R_{f2}-R_{f1}-R_{f3} \\
\qquad +R_{f4}+R_{s2}+x_{2,0}+x_{4,0}-x_{6,0}+2x_{8,0}-x_{9,0}-2x_{10,0}) \\
-k_{11}'(R_{f4}-R_{s2})(R_{f2}-R_{f1}-R_{f3}+R_{f4}+R_{s2}+x_{2,0}+x_{4,0}-x_{6,0}+2x_{8,0} \\
\qquad -x_{9,0}-2x_{10,0})-k_5'(R_{f4}+R_{s1}-x_{1,0}+x_{2,0}+x_{7,0}+x_{8,0}-x_{9,0}-x_{10,0}) \\
\qquad (R_{f1}-R_{f2}+R_{f3}-R_{f4}+R_{s1}+x_{5,0}+x_{6,0}+x_{7,0}-x_{8,0}+x_{9,0}+x_{10,0})
\end{bmatrix},
\end{aligned}$$

which gives us an entirely explicit form of the model in standard form.

Next, we shall briefly summarize the application of the method of Kumar et al. to this example, as was done in [68]. The original system in Equation (4.54) can be written in the

form with nonexplicit time-scale separation as

$$\dot{x} = f(x) + \frac{1}{\epsilon} V_f \bar{r}_f(x). \tag{4.55}$$

This can be reduced to a set of differential algebraic equations of the form

$$\dot{x} = f(x) + V_f z, \tag{4.56}$$

$$\bar{r}_f = 0, \tag{4.57}$$

where $z = \lim_{\epsilon \to 0} \frac{\bar{r}_f}{\epsilon}$. Equation (4.57) gives a set of algebraic constraints that the state trajectories must respect, effectively specifying a lower dimensional manifold near which the system in Equation (4.55) evolves. In some situations, these constraints may be differentiated in time to yield an explicit expression for the variables z. In particular, we have

$$\frac{\mathrm{d}\,\bar{r}_f}{\mathrm{d}\,t} = \mathcal{L}_f \bar{r}_f(x) + \mathcal{L}_{V_f} \bar{r}_f(x) z = 0, \tag{4.58}$$

where the column vector $\mathcal{L}_b a(x)$ is such that $[\mathcal{L}_b a(x)]_i \triangleq \frac{\partial a_i}{\partial x} b(x)$ is the Lie derivative of the scalar field $a_i(x)$ with respect to the vector field $b(x)$, and a Lie derivative of $\bar{r}_f$ with respect to a matrix V_f is interpreted as the matrix formed by concatenating the Lie derivatives with respect to the individual columns of V_f. When $\mathcal{L}_{V_f} \bar{r}_f(x)$ is nonsingular, we have

$$z = -\left[\mathcal{L}_{V_f} \bar{r}_f(x)\right]^{-1} \mathcal{L}_f \bar{r}_f(x). \tag{4.59}$$

The general form of the coordinate transformation given in [68] is

$$\begin{bmatrix} \zeta \\ \eta \end{bmatrix} \triangleq T(x) \triangleq \begin{bmatrix} \phi(x) \\ \bar{r}_f(x) \end{bmatrix},$$

where ζ and η are the slow and fast variables respectively, and $\zeta = \phi(x)$ is a vector of the same dimension as the equilibrium manifold. The function ϕ can be chosen with considerable flexibility, and in general will lead to slow variables that exhibit fast initial

transients. We can derive an explicit expression for these slow variables by differentiating ζ,

$$\dot{\zeta} = \left\{ \mathcal{L}_f \phi(x) - \mathcal{L}_{V_f} \phi(x) \left[\mathcal{L}_{V_f} \bar{r}_f(x) \right]^{-1} \left[\mathcal{L}_f \bar{r}_f(x) \right] \right\} \Big|_{x = T^{-1}(\zeta, 0)}, \tag{4.60}$$

which reduces to

$$\dot{\zeta} = \mathcal{L}_f \phi(x) |_{x = T^{-1}(\zeta, 0)} \tag{4.61}$$

if $\mathcal{L}_{V_f} \phi(x)$ is identically zero. We note that the second term in Equation (4.60) defines the initial fast transients, and we can get 'true' slow variables if the condition $\mathcal{L}_{V_f} \phi(x) \equiv 0$ holds. Kumar et al. show that this is only possible when the distribution spanned by V_f is involutive, and [68] notes that for the constant stoichiometric matrix, this is always the case, and indeed leads to a matrix Φ, with $\phi(x) = \Phi x$, whose rows are in the left null space of the fast reaction stoichiometric matrix V_f. This condition for finding slow variables that are independent of the fast dynamics was also mentioned in [39].

When this method is applied to the model of the esterification of carboxylic acid, as was done in [68], we find that the involutivity condition holds, and the transformation $T(x)$ has the components

$$\phi(x) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & -1 & 0 & 2 & 0 & 0 & 0 \\ -1 & 1 & 0 & -1 & 0 & 1 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} x, \qquad \bar{r}_f(x) = \begin{bmatrix} k_1' x_6 x_4 - k_7' x_5 x_{11} \\ k_2' x_5 x_2 - k_8' x_7 x_4 \\ k_4' x_2 x_4 - k_{10}' x_8 x_5 \\ k_5' x_1 x_5 - k_{11}' x_4 x_9 \end{bmatrix}. \tag{4.62}$$

The resulting final expression for the slow subsystem is given by

$$\dot{\zeta} = \mathcal{L}_f \phi(x)|_{x=T^{-1}(\zeta,0)} \tag{4.63}$$

$$= \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & -1 & 0 & 2 & 0 & 0 & 0 \\ -1 & 1 & 0 & -1 & 0 & 1 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 \\ 0 & 1 & 0 & -1 \\ 0 & -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} k_3 x_1 x_7 \\ k_6 x_{11} x_9 \\ k_9 x_5 x_3 \\ k_{12} x_{10} x_4 \end{bmatrix} \Bigg|_{x=T^{-1}(\zeta,0)} . \tag{4.64}$$

We note that the right hand side of Equation (4.64) requires an evaluation of T^{-1}, which is nontrivial. A merit of this method is that the slow variables can be chosen with great flexibility, even when the stoichiometric matrix is not constant, and when other nonlinearities exist in the system. In this case, Equation (4.60) gives the dynamics of the slow subsystem in terms of these variables, and reduces to a simpler case when the involutivity condition holds, as it does for isothermal reactions. In this sense, this method gives a unified method for bringing systems with nonexplicit time-scale-separation into the standard form. We also note that the fast variables in [41, 68] are the net reaction velocity expressions for the fast reactions, whereas in our method, these quantities are the rates of change of the fast variables. This is a fundamental difference between the two methods, and allows us to give a physical interpretation to our fast variables as the reaction extents of the fast reactions.

4.5 Application of Reaction Extents to the Reduction of Transcription and Translation Reactions

We now apply the transformations provided in Sections 4.4.3.1 and 4.4.3.2 to the transcription model (4.32) to derive the corresponding standard singular perturbation form. We can write the model (4.32) in matrix notation as

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} d \\ \rho \\ v_1 \\ w_1 \\ v_2 \\ w_2 \\ \vdots \\ v_n \\ w_n \\ m_n \\ \eta \\ \gamma \end{bmatrix} = \begin{array}{c} \\ \tilde{r}_{2n+2} \\ \tilde{r}_{2n+1} \\ \tilde{r}_{2n} \\ \tilde{r}_{2n-1} \\ \tilde{r}_{2n-2} \\ \tilde{r}_{2n-3} \\ \vdots \\ \tilde{r}_2 \\ \tilde{r}_1 \\ \tilde{q}_1 \\ \tilde{q}_2 \\ \tilde{q}_3 \end{array} \begin{array}{c} \begin{array}{ccccccccc} \tilde{c}_1 & \tilde{c}_2 & \tilde{c}_3 & \tilde{c}_4 & \tilde{c}_5 & \tilde{c}_6 & \cdots & \tilde{c}_{2n+1} & \tilde{c}_{2n+2} \end{array} \\ \begin{pmatrix} 1 & -1 & 0 & 0 & 0 & 0 & & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 & & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & & 0 & 0 \\ & & & & & & \ddots & & \\ 0 & 0 & 0 & 0 & 0 & 1 & & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & & 0 & 1 \\ 0 & 0 & -1 & 0 & -1 & 0 & & -1 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & & 0 & 1 \end{pmatrix} \end{array} \begin{bmatrix} k_{\mathrm{term}}\gamma \\ k_{Pf}\rho d - k_{Pr}v_1 \\ k_{Nf}v_1\eta - k_{Nr}w_1 \\ k_t w_1 \\ k_{Nf}v_2\eta - k_{Nr}w_2 \\ k_t w_2 \\ \vdots \\ k_{Nf}v_n\eta - k_{Nr}w_n \\ k_t w_n \end{bmatrix}, \tag{4.65}$$

where the $\tilde{q}_i$, $\tilde{r}_i$'s and $\tilde{c}_j$'s are row and column labels we will use for manipulating the matrix $\tilde{S} \in \mathbb{R}^{(2n+5)\times(2n+2)}$. Compactly, we write this as

$$\frac{\mathrm{d}x}{\mathrm{d}t} = S\,v, \qquad x(0) = x_0, \tag{4.66}$$

where we have dropped the bars for notational clarity. Here the species concentration vector is $x \in \mathbb{R}_+^{2n+5}$ and the reaction velocity vector is $v \in \mathbb{R}^{2n+2}$. The stoichiometric matrix

S has linearly dependent rows due to the existence of conservation laws. We can define the matrices P and H as defined in the discussion surrounding Equation (4.36) as follows. The rows $\tilde{r}_{2n+1}$, $\tilde{q}_3$ and $\tilde{q}_1$ are linearly dependent on the remaining rows of the matrix because of the relations $\tilde{r}_{2n+1} = \tilde{r}_{2n+2}$ and $\tilde{q}_3 = -\sum_{i=1}^{2n+1} \tilde{r}_i$, and the argument provided in Lemma 4 in the appendix. The fact that no other linear dependencies exist follows from the fact that the matrix resulting from the removal of these rows is invertible (Lemma 5 in the appendix). Thus, we can define

$$P = \begin{bmatrix} 1 & 0 & 0 & 0 & & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & & 0 & 0 & 0 & 0 \\ & & & & \ddots & & & & \\ 0 & 0 & 0 & 0 & & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & & 0 & 0 & 1 & 0 \end{bmatrix}, \quad H = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & \dots & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & \dots & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 2 & 2 & \dots & n & n & 1 & 0 \end{bmatrix}.$$

With these matrices, we can define S_r, x_i, x_d and R via Equations (4.36), (4.37), (4.44) and (4.47). Let us enumerate the elements of $R \in \mathbb{R}^{2n+2}$ as $R = [\, R_{term}\ R_P\ R_{N_1}\ R_{t_1}\ \dots\ R_{N_n}\ R_{t_n}\,]$. Here, the subscripts are chosen to represent the reactions present in the system, due to

the relationship $\mathrm{d}R/\mathrm{d}t = \nu_{rxn}$ in Equation (4.48) and note that

$$x = \begin{bmatrix} P \\ H \end{bmatrix}^{-1} \begin{bmatrix} S_r R \\ x_d \end{bmatrix} = \begin{bmatrix} R_{term} - R_P \\ R_{term} - R_P - x_{d_{10}} \\ R_P - R_{N_1} \\ R_{N_1} - R_{t_1} \\ R_{t_1} - R_{N_2} \\ \vdots \\ R_{t_{n-1}} - R_{N_n} \\ R_{N_n} - R_{t_n} \\ R_{t_n} - x_{d_{30}}/5 \\ -\sum_{i=1}^{n} R_{N_i} \\ R_{t_n} - R_{term} + x_{d_{10}} + x_{d_{20}} \end{bmatrix}, \tag{4.67}$$

where $x_{d_{m0}}$ for $m \in \{1, 2, 3\}$ are the dependent variables fixed to their intitial conditions (due to Equation (4.45)). Thus, the ODEs in R coordinates for this system (Equation (4.48)) are

$$\frac{\mathrm{d}R}{\mathrm{d}t} = \begin{bmatrix} k_{\mathrm{term}}(R_{t_n} - R_{term} + x_{d_{10}} + x_{d_{20}}) \\ k_{Pf}(R_{term} - R_P - x_{d_{10}})(R_{term} - R_P) - k_{Pr}(R_P - R_{N_1}) \\ k_{Nf}(R_P - R_{N_1})(-\sum_{i=1}^{n} R_{N_i}) - k_{Nr}(R_{N_1} - R_{t_1}) \\ k_t(R_{N_1} - R_{t_1}) \\ \vdots \\ k_{Nf}(R_{t_{j-1}} - R_{N_j})(-\sum_{i=1}^{n} R_{N_i}) - k_{Nr}(R_{N_j} - R_{t_j}) \\ k_t(R_{N_j} - R_{t_j}) \\ \vdots \\ k_{Nf}(R_{t_{n-1}} - R_{N_n})(-\sum_{i=1}^{n} R_{N_i}) - k_{Nr}(R_{N_n} - R_{t_n}) \\ k_t(R_{N_n} - R_{t_n}) \end{bmatrix}, \tag{4.68}$$

and defining M_s, M_f, R_s, R_f to separate out the slow and fast terms, we have,

$$\frac{\mathrm{d}R_s}{\mathrm{d}t} = \begin{bmatrix} k_{\text{term}}(R_{t_n} - R_{term} + x_{d_{10}} + x_{d_{20}}) \\ k_t(R_{N_1} - R_{t_1}) \\ \vdots \\ k_t(R_{N_j} - R_{t_j}) \\ \vdots \\ k_t(R_{N_n} - R_{t_n}) \end{bmatrix}, \tag{4.69}$$

$$\frac{\mathrm{d}R_f}{\mathrm{d}t} = \begin{bmatrix} k_{Pf}(R_{term} - R_P - x_{d_{10}})(R_{term} - R_P) - k_{Pr}(R_P - R_{N_1}) \\ k_{Nf}(R_P - R_{N_1})(-\sum_{i=1}^{n} R_{N_i}) - k_{Nr}(R_{N_1} - R_{t_1}) \\ \vdots \\ k_{Nf}(R_{t_{j-1}} - R_{N_j})(-\sum_{i=1}^{n} R_{N_i}) - k_{Nr}(R_{N_j} - R_{t_j}) \\ \vdots \\ k_{Nf}(R_{t_{n-1}} - R_{N_n})(-\sum_{i=1}^{n} R_{N_i}) - k_{Nr}(R_{N_n} - R_{t_n}) \end{bmatrix}. \tag{4.70}$$

We can further transform this into a nondimensional model, where a small parameter ϵ can be defined to capture the effects of the scale separation in the parameter values.

After nondimensionalization, we have

$$\frac{\mathrm{d}r_s}{\mathrm{d}\tau} = \begin{bmatrix} \alpha_{\text{term}}(r_{t_n} - r_{term} + \bar{x}_{d_{10}} + \bar{x}_{d_{20}}) \\ (r_{N_1} - r_{t_1}) \\ \vdots \\ (r_{N_j} - r_{t_j}) \\ \vdots \\ (r_{N_n} - r_{t_n}) \end{bmatrix}, \tag{4.71}$$

$$\epsilon\frac{\mathrm{d}r_f}{\mathrm{d}\tau} = \begin{bmatrix} \alpha_{Pf}(r_{term} - r_P - \bar{x}_{d_{10}})(r_{term} - r_P) - \alpha_{Pr}(r_P - r_{N_1}) \\ (r_P - r_{N_1})(-\sum_{i=1}^{n} r_{N_i}) - (r_{N_1} - r_{t_1}) \\ \vdots \\ (r_{t_{j-1}} - r_{N_j})(-\sum_{i=1}^{n} r_{N_i}) - (r_{N_j} - r_{t_j}) \\ \vdots \\ (r_{t_{n-1}} - r_{N_n})(-\sum_{i=1}^{n} r_{N_i}) - (r_{N_n} - r_{t_n}) \end{bmatrix}, \tag{4.72}$$

with the scheme

$$\begin{bmatrix} r_{t_i} \\ r_{term} \\ r_{N_i} \\ r_P \\ \bar{x}_{d_{j0}} \\ \epsilon \\ \alpha_{Pf} \\ \alpha_{Pr} \\ \tau \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} R_{t_i} \\ R_{term} \\ R_{N_i} \\ R_P \\ x_{d_{j0}} \end{bmatrix} \\ k_t/k_{Nr} \\ k_{Pf}/k_{Nf} \\ k_{Pr}/k_{Nr} \\ k_t t \end{bmatrix}, \qquad \forall i \in \{1, \ldots, n\},\ j \in \{1, 2, 3\}, \tag{4.73}$$

where the α parameters are of order $O(1)$. Setting $\epsilon = 0$ and transforming back to species

concentration coordinates, we have

$$0 = \begin{bmatrix} \alpha_{Pf}(r_{term} - r_P - \bar{x}_{d_{10}})(r_{term} - r_P) - \alpha_{Pr}(r_P - r_{N_1}) \\ (r_P - r_{N_1})(-\sum_{i=1}^{n} r_{N_i}) - (r_{N_1} - r_{t_1}) \\ \vdots \\ (r_{t_{j-1}} - r_{N_j})(-\sum_{i=1}^{n} r_{N_i}) - (r_{N_j} - r_{t_j}) \\ \vdots \\ (r_{t_{n-1}} - r_{N_n})(-\sum_{i=1}^{n} r_{N_i}) - (r_{N_n} - r_{t_n}) \end{bmatrix} \tag{4.74}$$

$$= \begin{bmatrix} k_{Pf}\rho d - k_{Pr}v_1 \\ k_{Nf}v_1\eta - k_{Nr}w_1 \\ k_{Nf}v_2\eta - k_{Nr}w_2 \\ \vdots \\ k_{Nf}v_n\eta - k_{Nr}w_n \end{bmatrix}. \tag{4.75}$$

4.6 Generalized Consumption Model

In this section, we reduce the full model (4.1) to a generalized version of the consumption model by matching their steady state behaviors. In Section 4.2, we modeled the creation of RNA as a single step transcription reaction, with no intermediate RNA transcripts, and discussed a method for incorporating nucleotide consumption in this model. While this suffices for models where the only regulation of gene expression comes from transcription factor proteins, it becomes inadequate when we wish to model the regulation of transcription via interactions with nascent RNA transcripts, for instance by non-coding RNA like the pT181 transcriptional attenuator [8, 43, 50]. Thus, we wish to have a transcription scheme where the creation of the final RNA transcript occurs in steps, and each intermediate RNA piece is free to interact with other elements of the environment. For an RNA transcript of length n, the detailed model shown in Equation (4.1) has approximately $2n$ chemical reactions and about the same number of differential equations. For typical RNAs of length $500-2000$ bases, these models quickly become unwieldy, especially when modeling large systems composed of multiple genes and regulatory pathways. Thus, we derive a way to

reduce such transcription models while quantifying the error introduced into the dynamics by such reductions.

We begin by re-indexing the detailed model (4.1) as follows. Divide the length RNA (of length n) into K segments of lengths n_i, for $i \in \{1, 2, \ldots, K\}$. The species $\mathrm{P{:}D_{k,j}{:}m_{k,j-1}}$, with $k \in \{1, 2, \ldots, K\}$, refers to the species from the kth segment, with the polymerase attached to the jth DNA base pair site of this segment, where j is in $\{1, \ldots n_k\}$, and an RNA of length $\sum_{i=1}^{k-1} n_i + (j-1)$ attached to the polymerase molecule. Finally, identify the last element of a block with the first element of the next block, i.e., $\mathrm{P{:}D_{k,n_k+1}{:}m_{k,n_k}} = \mathrm{P{:}D_{k+1,1}{:}m_{k+1,0}}$. This new indexing scheme is shown on the left in Figure 4.2. We would like to reduce this model to the one shown in Equation (4.76), where the RNA and DNA have been divided into K blocks. The concentration of each species in the reduced model (4.76) is the sum of the corresponding species in the full model (4.1). For example, $[\mathrm{P{:}D{:}R_k}] = \sum_{h=1}^{n_k} [\mathrm{P{:}D_{(k,h)}{:}m_{(k,h-1)}}]$.

$$
\begin{aligned}
\mathrm{P + D} &\underset{k'_{Pr}}{\overset{k'_{Pf}}{\rightleftharpoons}} \mathrm{P{:}D} && \text{RNA polymerase binding,}\\
\mathrm{P{:}D + N} &\underset{k'_{Nr}}{\overset{k'_{Nf}}{\rightleftharpoons}} \mathrm{P{:}D{:}N} && \text{nucleotide binding,}\\
\mathrm{P{:}D{:}N} &\xrightarrow{k_{\mathrm{con}(1)}} \mathrm{P{:}D} && \text{consumption reaction,}\\
\mathrm{P{:}D{:}N} &\xrightarrow{k_{tx(1)}} \mathrm{P{:}D{:}R_1} && \text{lumped RNA production,}\\
&\vdots &&\\
\mathrm{P{:}D{:}R_{K-1} + N} &\underset{k'_{Nr}}{\overset{k'_{Nf}}{\rightleftharpoons}} \mathrm{P{:}D{:}R_{K-1}{:}N} && \text{nucleotide binding,}\\
\mathrm{P{:}D{:}R_{K-1}{:}N} &\xrightarrow{k_{\mathrm{con}(K)}} \mathrm{P{:}D{:}R_{K-1}} && \text{consumption reaction,}\\
\mathrm{P{:}D{:}R_{K-1}{:}N} &\xrightarrow{k_{tx(K)}} \mathrm{P{:}D_t + R_K} && \text{lumped RNA production,}\\
\mathrm{P{:}D_t} &\xrightarrow{k_{\mathrm{term}}} \mathrm{P + D} && \text{termination.}
\end{aligned}
\tag{4.76}
$$

Let D_T be the total DNA concentration in in either model. Mass conservation gives

$$
D_T = \sum_{i=1}^{K} \sum_{h=1}^{n_i} \Big([\mathrm{P{:}D_{(i,h)}{:}m_{(i,h-1)}}] + [\mathrm{P{:}D_{(i,h)}{:}m_{(i,h-1)}{:}N}] \Big) + [\mathrm{D}] + [\mathrm{P{:}D_t}], \tag{4.77}
$$

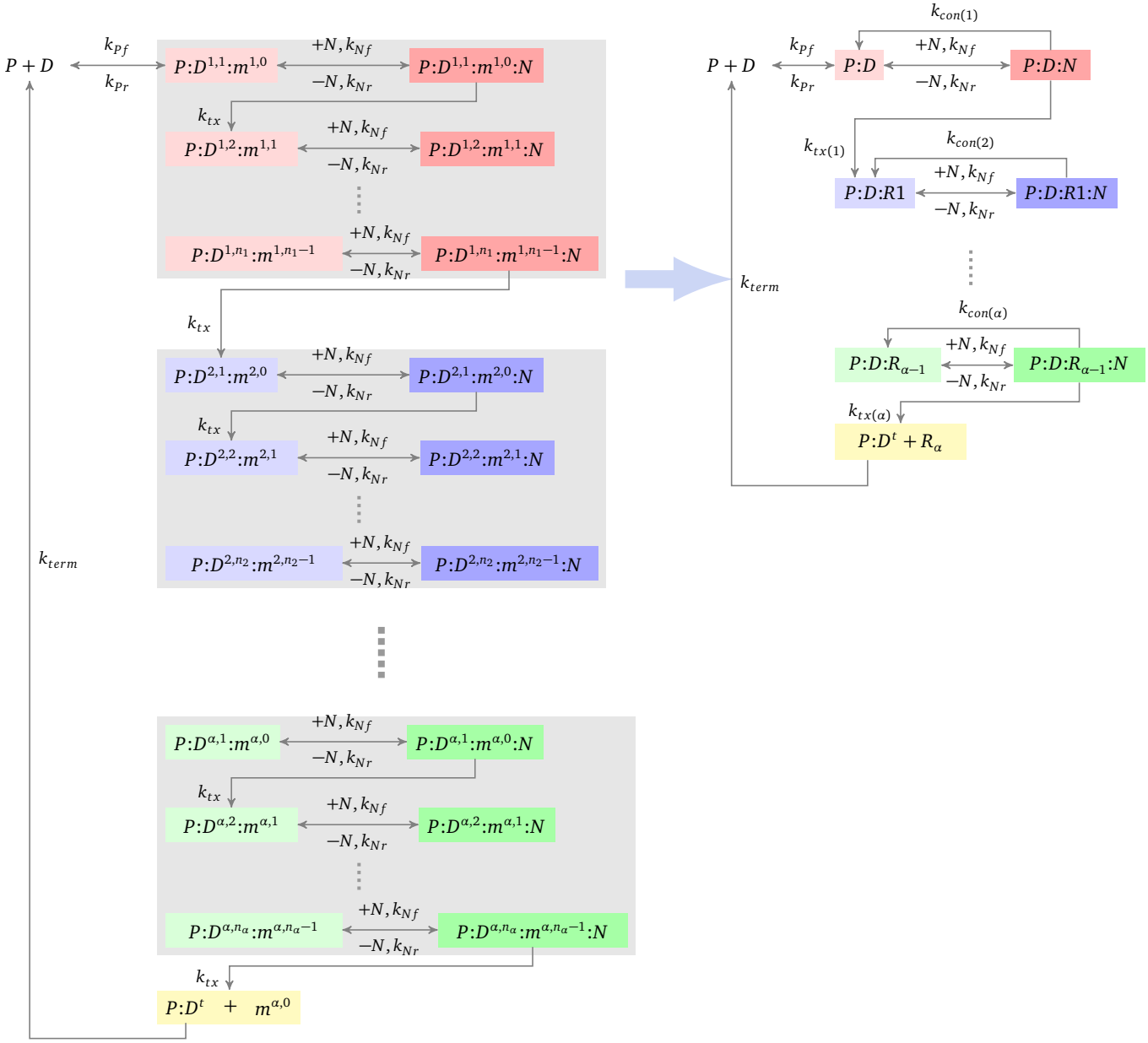

Figure 4.2: The model reduction procedure. Corresponding species are color coded. Each grey 'block' represents the part of the model that transcribes the corresponding segment of the RNA, with the output of the grey block as the intermediate RNA species which will be modeled by the reduced model.

for the full model and

$$D_T = \sum_{i=1}^{K} \left([\mathrm{P{:}D{:}R_{i-1}}] + [\mathrm{P{:}D{:}R_{i-1}{:}N}] \right) + [\mathrm{D}] + [\mathrm{P{:}D_t}], \tag{4.78}$$

with $\mathrm{P{:}D{:}R_0} \triangleq \mathrm{P{:}D}$ and $\mathrm{P{:}D{:}R_0{:}N} \triangleq \mathrm{P{:}D{:}N}$, for the reduced model.

Using the mass balance equations (4.21), (4.22) and rapid equilibrium assumption (4.75), we can express $[\mathrm{P{:}D_{(k,h)}{:}m_{(k,h-1)}}]$ in the full model as:

$$[\mathrm{P{:}D_{(k,h)}{:}m_{(k,h-1)}}] = \frac{1}{n_k} \left(\frac{D_T - [\mathrm{D}] - [\mathrm{P{:}D_t}]}{\left(1 + \frac{k_{Nf}}{k_{Nr}}[\mathrm{N}]\right)} - \sum_{i=1, i\neq k}^{K} n_i [\mathrm{P{:}D_{(i,1)}{:}m_{(i,0)}}] \right). \tag{4.79}$$

The rapid equilibrium assumption for the reduced model can be derived in exactly the same way as that for the full model, and using it gives

$$[\mathrm{P{:}D{:}R_k}] = \left(\frac{D_T - [\mathrm{D}] - [\mathrm{P{:}D_t}]}{\left(1 + \frac{k'_{Nf}}{k'_{Nr}}[\mathrm{N}]\right)} - \sum_{i=0, i\neq k}^{K-1} [\mathrm{P{:}D{:}R_i}] \right). \tag{4.80}$$

We would like the reduced model to be an approximation of the full model at steady state (noting that the steady state is defined in the sense of the discussion preceding Proposition 3, when the nucleotide concentration is sufficiently large to be assumed to be essentially constant in the time-scale of interest). A set of sufficient conditions for this to be true is presented for the remainder of this section.

We first note that the discussion around equations (4.14) to (4.22) in Section 4.3.2 can be applied to the species within a single block to obtain the result that at steady state, corresponding species within a block are in mass balance (i.e., (4.21) and (4.22) hold within each block). This allows us to define and simplify the correspondence $[\mathrm{P{:}D{:}R_k}] = \sum_{h=1}^{n_k} [\mathrm{P{:}D_{(k,h)}{:}m_{(k,h-1)}}]$ between the full and reduced models to

$$[\mathrm{P{:}D{:}R_k}] = n_k [\mathrm{P{:}D_{(k,h)}{:}m_{(k,h-1)}}], \qquad \text{Condition 1.} \tag{4.81}$$

The relation (4.81) is a condition we have imposed upon the model to create a corre-

spondence between the species between corresponding blocks of the full and reduced models. This condition, applied to equations (4.79) and (4.80), gives us further correspondences between the full and reduced models. In Equation (4.79), the term $D_T - [\mathrm{D}] - [\mathrm{P{:}D_t}]$ is the total concentration of species in the grey boxes in Figure 4.2, the $\frac{1}{(1+(k_{Nf}/k_{Nr})[\mathrm{N}])}$ term, which arises out of the analysis of Section 4.4.1, picks out the proportion of $[\mathrm{P{:}D_{(k,h)}{:}m_{(k,h-1)}}]$ from the total $[\mathrm{P{:}D_{(k,h)}{:}m_{(k,h-1)}}]+[\mathrm{P{:}D_{(k,h)}{:}m_{(k,h-1)}{:}N}]$, and so applying it to $D_T - [\mathrm{D}] - [\mathrm{P{:}D_t}]$ gives the total concentration of terms of the form $[\mathrm{P{:}D_{(k,h)}{:}m_{(k,h-1)}}]$. Equation (4.80) has a similar interpretation, with the difference that $[\mathrm{P{:}D{:}R_k}]$ is now the concentration of a block in the reduced model, and thus corresponds to the total concentration of terms of the form $[\mathrm{P{:}D_{(k,h)}{:}m_{(k,h-1)}}]$ in a single block of the full model, which by Equation (4.81) is $n_k[\mathrm{P{:}D_{(k,h)}{:}m_{(k,h-1)}}]$.

The next step in using the reduced model to approximate the full model is to equate the fluxes out of corresponding blocks in the two models. We can then obtain a set of (not necessarily unique) relations between the parameters of the two models so that the steady state behaviors match. Consider the rate of production of the species $\mathrm{P{:}D_{(k+1,1)}{:}m_{(k+1,0)}}$, which corresponds to the output of the kth block, or kth intermediate RNA transcript, in the full model,

$$\begin{aligned}[\mathrm{P{:}D_{(k+1,1)}{:}m_{(k+1,0)}}]_{\text{production}} &= k_{tx}[\mathrm{P{:}D_{(k,n_k)}{:}m_{(k,n_k-1)}{:}N}] \\ &= k_{tx}\frac{k_{Nf}}{k_{Nr}}[\mathrm{P{:}D_{(k,h)}{:}m_{(k,h-1)}}][\mathrm{N}],\end{aligned} \tag{4.82}$$

where we recall that $\mathrm{P{:}D_{(k,n_k+1)}{:}m_{(k,n_k)}}$ is identified with $\mathrm{P{:}D_{(k+1,1)}{:}m_{(k+1,0)}}$.

The corresponding expression for the reduced model is

$$\begin{aligned}[\mathrm{P{:}D{:}R_{k+1}}]_{\text{production}} &= k'_{tx(k)}[\mathrm{P{:}D{:}R_k{:}N}] \\ &= k'_{tx(k)}\frac{k'_{Nf}}{k'_{Nr}}[\mathrm{P{:}D{:}R_k}][\mathrm{N}].\end{aligned} \tag{4.83}$$

Equating the equations (4.82) and (4.83), and comparing terms, we can draw a corre-

spondence between the respective coefficients,

$$k'_{tx(k)} = \frac{k_{tx}}{n_k}, \tag{4.84}$$

$$k'_{Nf} = k_{Nf}, \tag{4.85}$$

$$k'_{Nr} = k_{Nr}, \qquad \text{Conditions 2 - 4.} \tag{4.86}$$

We now state a result that generalizes the result of Proposition 3 to the model lumped into blocks, and states that the rate of consumption of nucleotides within the kth block is n_k times the rate of production of the kth nascent transcript. This, in turn, will allow us to derive a generalization for the consumption reaction for the reduced model (4.76).

Proposition 4. *Consider the full model* (4.1) *under the same assumptions as Proposition 3, now considered with the indexing scheme that divides it into separate transcription blocks, as described at the beginning of Section 4.6. Then, in the kth block, for $k \in \{1, 2, \ldots, K\}$, the rate of nucleotide consumption within the block is proportional to the rate of production of the kth intermediate transcript, $\mathrm{P{:}D_{k,n_k+1}{:}m_{k,n_k}}$ (with the Kth transcript being $\mathrm{m_{K,0}}$).*

Proof. From the proof of Proposition 3, we know that the total rate of nucleotide consumption is

$$\frac{\mathrm{d[N_{uninc}]}}{\mathrm{d}t} = -k_{tx} \sum_{i=1}^{n} \mathrm{[P{:}D_i{:}m_{i-1}{:}N]}. \tag{4.87}$$

This sum can be partitioned into terms using the new indexing scheme into contributions

to the consumption rate by each block as follows:

$$\begin{aligned}
\frac{\mathrm{d}[\mathrm{N}_{\mathrm{uninc}}]}{\mathrm{d}t} &= -k_{tx}\sum_{k=1}^{K}\sum_{h=1}^{n_k}[\mathrm{P{:}D}_{(\mathrm{k,h})}{:}\mathrm{m}_{(\mathrm{k,h-1})}{:}\mathrm{N}] \\
&= -k_{tx}\sum_{h=1}^{n_1}[P{:}D_{(1,h)}{:}m_{(1,h-1)}{:}N] \\
&\quad - k_{tx}\sum_{h=1}^{n_2}[P{:}D_{(2,h)}{:}m_{(2,h-1)}{:}N] \\
&\qquad\qquad \vdots \\
&\quad - k_{tx}\sum_{h=1}^{n_K}[\mathrm{P{:}D}_{(\mathrm{k,h})}{:}\mathrm{m}_{(\mathrm{k,h-1})}{:}\mathrm{N}] \\
&= -n_1k_{tx}[P{:}D_{(1,n_1)}{:}m_{(1,n_1-1)}{:}N] \\
&\quad - n_2k_{tx}[P{:}D_{(2,n_2)}{:}m_{(2,n_2-1)}{:}N] \\
&\qquad\qquad \vdots \\
&\quad - n_Kk_{tx}[\mathrm{P{:}D}_{(\mathrm{K,n_K})}{:}\mathrm{m}_{(\mathrm{K,n_K-1})}{:}\mathrm{N}],
\end{aligned} \tag{4.88}$$

where the last line follows from the fact that the species within a block are in flux balance as given by equations (4.21) and (4.22). In each of the terms of the form $-n_k k_{tx}[\mathrm{P{:}D}_{(\mathrm{k,n_k})}{:}\mathrm{m}_{(\mathrm{k,n_k-1})}{:}\mathrm{N}]$ in the above partition, the term $k_{tx}[\mathrm{P{:}D}_{(\mathrm{k,n_k})}{:}\mathrm{m}_{(\mathrm{k,n_k-1})}{:}\mathrm{N}]$ is the production rate of the kth transcript (bound to the RNA polymerase and DNA) for all $k \in \{1, 2, \ldots, K-1\}$ while it is the production rate of $\mathrm{m}_{\mathrm{K,0}}$ for $k = K$. On the other hand, the full term is the contribution of the kth block to the total consumption rate of the nucleotides not yet incorporated into the any RNA. □

The rate of consumption of nucleotides is the same in the full model (4.1) and the reduced model (4.76) if we set

$$k'_{\mathrm{con}(k)} = k'_{tx(k)}(n_k - 1), \qquad \text{Condition 5.} \tag{4.89}$$

Indeed, from Proposition 4, and equations (4.81) and (4.83), the rate of consumption of

N in the kth block of the full model is

$$k_{tx} n_k [\mathrm{P{:}D_{(k,n_k)}{:}m_{(k,n_k-1)}{:}N}] = k_{tx} \frac{n_k}{n_k} [\mathrm{P{:}D{:}R_k{:}N}] \tag{4.90}$$

$$= \frac{k_{tx}}{n_k} [\mathrm{P{:}D{:}R_k{:}N}] + \frac{k_{tx}}{n_k} (n_k - 1) [\mathrm{P{:}D{:}R_k{:}N}] \tag{4.91}$$

$$= [\mathrm{P{:}D{:}R_{k+1}}]_{\text{production}} + k'_{\text{con}(k)} [\mathrm{P{:}D{:}R_k{:}N}]. \tag{4.92}$$

4.7 Discussion

We have introduced a scalable method for incorporating nucleotide resource consumption in a reduced order model of transcription. This method uses a consumption reaction to emulate the usage of nucleotides, instead of modeling each nucleotide binding event separately or binding all the nucleotides simultaneously (Figure 4.1).

We have also generalized this method to allow for multiple intermediate transcripts. The approach here is to find a set of sufficient conditions, for a given nucleotide concentration, for the steady state dynamics of the full and reduced models to match. In a subsequent work, we plan to also study the deviation in the transient behavior of the two models and quantify the tradeoff between model reduction and fidelity.

In the process of attempting the above generalization, we had to use the deficiency zero theorem from chemical reaction network theory to show that part of the system we were considering has a steady state. We also had to define a coordinate transformation to justify the rapid equilibrium assumption that was made, since in the species concentration coordinate, such an assumption was not supported by singular perturbation theory. A technical point to include in a future work is to bring the two-time-scale model into the true *standard singular perturbation form*. This involves showing that the fast subsystem, with $\epsilon = 0$ in equations (4.27) and (4.72), has an isolated root, and that this root is (at least locally) asymptotically stable.

Other directions this work may be extended in is to allow for multiple occupancy of the DNA transcript by RNA polymerase, adding stochastic effects to the model, and to create an analogous scheme for translation, which should be similar in many respects.

Appendices

4.A Detailed Proofs

Lemma 4. *The row labeled $q_{1,n}$ in matrix P_n, defined as*

$$P_n = \begin{array}{c} r_{2n+1,n} \\ r_{2n,n} \\ r_{2n-1,n} \\ r_{2n-2,n} \\ r_{2n-3,n} \\ \vdots \\ r_{2,n} \\ r_{1,n} \\ q_{1,n} \\ q_{2,n} \end{array} \begin{pmatrix} 1 & -1 & 0 & 0 & 0 & 0 & & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & & 0 & 0 & 0 \\ & & & & & & \ddots & & & \\ 0 & 0 & 0 & 0 & 0 & 0 & & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 & -1 & 0 & & 0 & -1 & 0 \end{pmatrix}, \tag{4.93}$$

is a linear combination of the remaining rows.

Proof. We use induction on the size of the matrix. We will prove that for

$$P_k = \begin{array}{c} \\ r_{2k+1,k} \\ r_{2k,k} \\ r_{2k-1,k} \\ r_{2k-2,k} \\ r_{2k-3,k} \\ \vdots \\ r_{2,k} \\ r_{1,k} \\ q_{1,k} \\ q_{2,k} \end{array} \begin{array}{c} \begin{array}{cccccccccc} 1 & -1 & 0 & 0 & 0 & 0 & & 0 & 0 & 0 \end{array} \\ \begin{array}{cccccccccc} 0 & 1 & -1 & 0 & 0 & 0 & & 0 & 0 & 0 \end{array} \\ \begin{array}{cccccccccc} 0 & 0 & 1 & -1 & 0 & 0 & & 0 & 0 & 0 \end{array} \\ \begin{array}{cccccccccc} 0 & 0 & 0 & 1 & -1 & 0 & & 0 & 0 & 0 \end{array} \\ \begin{array}{cccccccccc} 0 & 0 & 0 & 0 & 1 & -1 & & 0 & 0 & 0 \end{array} \\ \begin{array}{cccccccccc} & & & & & & \ddots & & & \end{array} \\ \begin{array}{cccccccccc} 0 & 0 & 0 & 0 & 0 & 0 & & 1 & -1 & 0 \end{array} \\ \begin{array}{cccccccccc} 0 & 0 & 0 & 0 & 0 & 0 & & 0 & 1 & -1 \end{array} \\ \begin{array}{cccccccccc} 0 & 0 & 0 & 0 & 0 & 0 & & 0 & 0 & 1 \end{array} \\ \begin{array}{cccccccccc} 0 & 0 & -1 & 0 & -1 & 0 & & 0 & -1 & 0 \end{array} \end{array}, \tag{4.94}$$

the row $q_{2,k}$ can be written as the linear combination $q_{2,k} = -(r_{2k-1,k} + r_{2k-2,k}) - 2(r_{2k-3,k} + r_{2k-4,k}) - 3(r_{2k-5,k} + r_{2k-6,k}) - \cdots - k(r_{1,k} + q_{1,k})$, and inverting this relationship gives $q_{1,k}$ as the linear combination

$$q_{1,k} = \frac{-(r_{2k-1,k} + r_{2k-2,k}) - 2(r_{2k-3,k} + r_{2k-4,k}) - \cdots - (k-1)(r_{3,k} + r_{2,k}) - kr_{1,k} - q_{2,k}}{k}. \tag{4.95}$$

Note that the claim holds for P_1 and P_2

$$P_1 = \begin{matrix} r_{3,1} \\ r_{2,1} \\ r_{1,1} \\ q_{1,1} \\ q_{2,1} \end{matrix} \begin{pmatrix} 1 & -1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \end{pmatrix}, \qquad P_2 = \begin{matrix} r_{5,2} \\ r_{4,2} \\ r_{3,2} \\ r_{2,2} \\ r_{1,2} \\ q_{1,2} \\ q_{2,2} \end{matrix} \begin{pmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 & -1 & 0 \end{pmatrix},$$

$$q_{1,1} = -r_{1,1} - q_{2,1}, \qquad q_{1,2} = \frac{-(r_{3,2} + r_{2,2}) - 2r_{1,2} - q_{2,2}}{2}.$$

Assume the claim holds for $k \geq 2$, as in Equation (4.94) and (4.95). For clarity, we remove rows $r_{2k+1,k}$ and $r_{2k,k}$, and the first two columns from P_k (which do not matter) to get

$$\bar{P}_k = \begin{matrix} \bar{r}_{2k-1,k} \\ \bar{r}_{2k-2,k} \\ \bar{r}_{2k-3,k} \\ \vdots \\ \bar{r}_{2,k} \\ \bar{r}_{1,k} \\ \bar{q}_{1,k} \\ \bar{q}_{2,k} \end{matrix} \begin{pmatrix} 1 & -1 & 0 & 0 & & 0 & 0 \\ 0 & 1 & -1 & 0 & & 0 & 0 \\ 0 & 0 & 1 & -1 & & 0 & 0 \\ & & & & \ddots & & \\ 0 & 0 & 0 & 0 & & -1 & 0 \\ 0 & 0 & 0 & 0 & & 1 & -1 \\ 0 & 0 & 0 & 0 & & 0 & 1 \\ -1 & 0 & -1 & 0 & & -1 & 0 \end{pmatrix}, \tag{4.96}$$

with

$$\bar{q}_{1,k} = \frac{-(\bar{r}_{2k-1,k} + \bar{r}_{2k-2,k}) - 2(\bar{r}_{2k-3,k} + \bar{r}_{2k-4,k}) - \cdots - (k-1)(\bar{r}_{3,k} + \bar{r}_{2,k}) - k\bar{r}_{1,k} - \bar{q}_{2,k}}{k}. \tag{4.97}$$

Now consider the $\bar{P}_{k+1}$, obtained by augmenting $\bar{P}_k$ with two rows and two columns:

$$\bar{P}_{k+1} = \left(\begin{array}{cc|ccccccc} 1 & -1 & 0 & 0 & \dots & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & \dots & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & & & & & & & \\ 0 & 0 & & & & & & & \\ \vdots & & & & & & & & \\ 0 & 0 & & & & \bar{P}_k & & & \\ 0 & 0 & & & & & & & \\ 0 & 0 & & & & & & & \\ -1 & 0 & & & & & & & \end{array}\right) \begin{array}{c} \bar{r}_{2(k+1)-1,k+1} \\ \bar{r}_{2(k+1)-2,k+1} \\ \bar{r}_{2(k+1)-3,k+1} \\ \bar{r}_{2(k+1)-4,k+1} \\ \vdots \\ \bar{r}_{2,k+1} \\ \bar{r}_{1,k+1} \\ \bar{q}_{1,k+1} \\ \bar{q}_{2,k+1} \end{array} \tag{4.98}$$

$$= \left(\begin{array}{cc|ccccccc} 1 & -1 & 0 & 0 & \dots & & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & \dots & & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & -1 & 0 & & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & & 0 & 0 & 0 \\ \vdots & & & & & \ddots & & & \\ 0 & 0 & 0 & 0 & 0 & & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & & 0 & 0 & 1 \\ -1 & 0 & -1 & 0 & -1 & & 0 & -1 & 0 \end{array}\right) \begin{array}{c} \bar{r}_{2(k+1)-1,k+1} \\ \bar{r}_{2(k+1)-2,k+1} \\ \bar{r}_{2(k+1)-3,k+1} \\ \bar{r}_{2(k+1)-4,k+1} \\ \vdots \\ \bar{r}_{2,k+1} \\ \bar{r}_{1,k+1} \\ \bar{q}_{1,k+1} \\ \bar{q}_{2,k+1} \end{array}, \tag{4.99}$$

where we identify (augmented) rows of the matrix P_k in Equation (4.94) with the corre-

sponding rows of the matrix $\bar{P}_{k+1}$ in Equation(4.98) as follows:

$$\begin{aligned}
\bar{r}_{2(k+1)-2-i,k+1} &= \begin{pmatrix} 0 & 0 & \bar{r}_{2k-i,k} \end{pmatrix} \qquad \forall i \in \{1,\dots,2k-1\},\\
\bar{q}_{1,k+1} &= \begin{pmatrix} 0 & 0 & \bar{q}_{1,k} \end{pmatrix},\\
\bar{q}_{2,k+1} &= \begin{pmatrix} 0 & 0 & \bar{q}_{2,k} \end{pmatrix}.
\end{aligned}$$

We know from Equation (4.97) that

$$\begin{aligned}
\begin{pmatrix} 0 & 0 & \bar{q}_{2,k} \end{pmatrix} = &-(\bar{r}_{2(k+1)-3,k+1} + \bar{r}_{2(k+1)-4,k}) - 2(\bar{r}_{2(k+1)-5,k} + \bar{r}_{2(k+1)-6,k})\\
&-\cdots- (k-1)(\bar{r}_{3,k+1} + \bar{r}_{2,k+1}) - k(\bar{r}_{1,k+1} - \bar{q}_{1,k+1}).
\end{aligned} \tag{4.100}$$

It is also clear that the rows of $\bar{P}_{k+1}$ in Equation (4.98) satisfy

$$-\sum_{i=1}^{2k+1} \bar{r}_{2(k+1)-i,k+1} - \bar{q}_{1,k+1} = \begin{pmatrix} -1 & 0 & 0 & \dots & 0 \end{pmatrix}, \tag{4.101}$$

and together equations (4.100) and (4.101) give us the expression

$$\begin{aligned}
\bar{q}_{2,k+1} &= \begin{pmatrix} 0 & 0 & \bar{q}_{2,k} \end{pmatrix} + \begin{pmatrix} -1 & 0 & 0 & \dots & 0 \end{pmatrix}\\
&= -\sum_{i=1}^{2k+1} \bar{r}_{2(k+1)-i,k+1} - \bar{q}_{1,k+1} - (\bar{r}_{2(k+1)-3,k+1} + \bar{r}_{2(k+1)-4,k}) - 2(\bar{r}_{2(k+1)-5,k} + \bar{r}_{2(k+1)-6,k})\\
&\quad -\cdots- (k-1)(\bar{r}_{3,k+1} + \bar{r}_{2,k+1}) - k(\bar{r}_{1,k+1} - \bar{q}_{1,k+1}).\\
&= -(\bar{r}_{2(k+1)-1,k+1} + \bar{r}_{2(k+1)-2,k+1}) - 2(\bar{r}_{2(k+1)-3,k+1} + \bar{r}_{2(k+1)-4,k+1})\\
&\quad -\cdots- k(\bar{r}_{3,k+1} + \bar{r}_{2,k+1}) - (k+1)(\bar{r}_{1,k+1} + \bar{q}_{1,k+1}),
\end{aligned} \tag{4.102}$$

which can be rewritten in the desired form of Equation (4.95) as:

$$\bar{q}_{1,k+1} = \frac{\begin{array}{r} -(\bar{r}_{2(k+1)-1,k+1} + \bar{r}_{2(k+1)-2,k+1}) - 2(\bar{r}_{2(k+1)-3,k+1} + \bar{r}_{2(k+1)-4,k+1})\\ -\cdots- k(\bar{r}_{3,k+1} + \bar{r}_{2,k+1}) - (k+1)\bar{r}_{1,k+1} - \bar{q}_{2,k+1} \end{array}}{k+1}. \tag{4.103}$$

Adding to $\bar{P}_{k+1}$ analogues of the rows and columns we removed from P_k to get $\bar{P}_k$, we

can construct P_{k+1},

$$P_{k+1} = \left(\begin{array}{cccc|cccccc} 1 & -1 & 0 & 0 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & \dots & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & -1 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & \dots & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & & & & & & \\ 0 & 0 & 0 & 0 & & & & & & \\ \vdots & & \vdots & & & & & & & \\ 0 & 0 & 0 & 0 & & & \bar{P}_k & & & \\ 0 & 0 & 0 & 0 & & & & & & \\ 0 & 0 & 0 & 0 & & & & & & \\ 0 & 0 & -1 & 0 & & & & & & \end{array}\right) \begin{array}{c} r_{2(k+1)+1,k+1} \\ r_{2(k+1),k+1} \\ r_{2(k+1)-1,k+1} \\ r_{2(k+1)-2,k+1} \\ r_{2(k+1)-3,k+1} \\ r_{2(k+1)-4,k+1} \\ \vdots \\ r_{2,k+1} \\ r_{1,k+1} \\ q_{1,k+1} \\ q_{2,k+1} \end{array}, \tag{4.104}$$

and for this matrix, the linear combination relationship (4.103) now becomes

$$q_{1,k+1} = \frac{\begin{array}{r} -(r_{2(k+1)-1,k+1} + r_{2(k+1)-2,k+1}) - 2(r_{2(k+1)-3,k+1} + r_{2(k+1)-4,k+1}) \\ -\dots - k(r_{3,k+1} + r_{2,k+1}) - (k+1)r_{1,k+1} - q_{2,k+1} \end{array}}{k+1}, \tag{4.105}$$

which completes the proof. □

Lemma 5. *The $2n+2$ by $2n+2$ matrix*

$$S = \begin{array}{c} \\ r_{2n+1} \\ r_{2n} \\ r_{2n-1} \\ r_{2n-2} \\ r_{2n-3} \\ \vdots \\ r_2 \\ r_1 \\ q_2 \end{array} \begin{array}{c} \begin{array}{cccccccccc} 1 & -1 & 0 & 0 & 0 & 0 & & 0 & 0 & 0 \end{array} \\ \begin{array}{cccccccccc} 0 & 1 & -1 & 0 & 0 & 0 & & 0 & 0 & 0 \end{array} \\ \begin{array}{cccccccccc} 0 & 0 & 1 & -1 & 0 & 0 & & 0 & 0 & 0 \end{array} \\ \begin{array}{cccccccccc} 0 & 0 & 0 & 1 & -1 & 0 & & 0 & 0 & 0 \end{array} \\ \begin{array}{cccccccccc} 0 & 0 & 0 & 0 & 1 & -1 & & 0 & 0 & 0 \end{array} \\ \begin{array}{cccccccccc} & & & & & & \ddots & & & \end{array} \\ \begin{array}{cccccccccc} 0 & 0 & 0 & 0 & 0 & 0 & & 1 & -1 & 0 \end{array} \\ \begin{array}{cccccccccc} 0 & 0 & 0 & 0 & 0 & 0 & & 0 & 1 & -1 \end{array} \\ \begin{array}{cccccccccc} 0 & 0 & -1 & 0 & -1 & 0 & & 0 & -1 & 0 \end{array} \end{array} \tag{4.106}$$

is invertible.

Proof. Consider first the upper triangular matrix

$$S_1 = \begin{array}{c} \\ r_{2n+1} \\ r_{2n} \\ r_{2n-1} \\ r_{2n-2} \\ r_{2n-3} \\ \vdots \\ r_2 \\ r_1 \\ q_1 \end{array} \begin{array}{c} \begin{array}{cccccccccc} 1 & -1 & 0 & 0 & 0 & 0 & & 0 & 0 & 0 \end{array} \\ \begin{array}{cccccccccc} 0 & 1 & -1 & 0 & 0 & 0 & & 0 & 0 & 0 \end{array} \\ \begin{array}{cccccccccc} 0 & 0 & 1 & -1 & 0 & 0 & & 0 & 0 & 0 \end{array} \\ \begin{array}{cccccccccc} 0 & 0 & 0 & 1 & -1 & 0 & & 0 & 0 & 0 \end{array} \\ \begin{array}{cccccccccc} 0 & 0 & 0 & 0 & 1 & -1 & & 0 & 0 & 0 \end{array} \\ \begin{array}{cccccccccc} & & & & & & \ddots & & & \end{array} \\ \begin{array}{cccccccccc} 0 & 0 & 0 & 0 & 0 & 0 & & 1 & -1 & 0 \end{array} \\ \begin{array}{cccccccccc} 0 & 0 & 0 & 0 & 0 & 0 & & 0 & 1 & -1 \end{array} \\ \begin{array}{cccccccccc} 0 & 0 & 0 & 0 & 0 & 0 & & 0 & 0 & 1 \end{array} \end{array} . \tag{4.107}$$

This is clearly invertible, and the set of its rows $U_1 = \{q_1,\ r_1,\ \ldots,\ r_{2n+1}\}$ forms a basis

for its rowspace $\text{span}(U_1) = \mathbb{R}^{2n+2}$. From Lemma 4, we have $q_1 \in \text{span}(\{q_2, r_1, \ldots, r_{2n+1}\})$, giving us that $U_2 = \{q_2, r_1, \ldots, r_{2n+1}\}$ is also a basis for $\mathbb{R}^{2n+2}$. Thus the rows of the square matrix S form a basis for its rowspace, implying that S is invertible. □

Chapter 5

Conclusion

Synthetic biology has come a long way since Wacław Szybalski first articulated it as a possibility. Yet, we are still only scratching the surface of the vast potential of this field. We believe that the use of mathematics and modeling can be just as rewarding in our field as it has been in other engineering disciplines. It will not only help us gain deeper understanding of the biological systems we are working with, but also enable more nuanced and carefully designed applications, and ultimately complex architectures such as Szybalski's "new better mouse".

In this thesis, we have developed and demonstrated new applications, tools and mathematical frameworks associated with the use of modeling in synthetic biology.

We began in Chapter 2 with an application of modeling approaches to reduce variability across extracts. We showed that not only could this be achieved, but the classical problem of parameter non-identifiability, first articulated in the control systems literature by Bellman and Astrom in their paper on structural identifiability [4], is not necessarily a death knell for modeling. Indeed, we show in that chapter that for our modeling objective, the sets of values that parameters can take due to non-identifiability may be treated as equivalent with some restrictions. At a fundamental level, this happens because of an exact alignment between two facts. Firstly, the only data we are attempting to transform using the models is the output data. Secondly, the parameters that are non-identifiable are so because their value cannot be discerned from the outputs, precisely because the output is insensitive to variations in these parameters within the sets of output-indistinguishable values. This equivalence of the values the non-identifiable parameters can take allows the

use of *arbitrary* values within these sets, greatly easing the modeling task. Said differently, we showed that with some restrictions, the sets of output indistinguishable parameters can be treated as equivalence classes with respect to the modeling objectives, and therefore any element can be used as a representative of its respective class. Indeed, these results hint at a somewhat deeper fact about the nature of parameter spaces and their use in model-based prediction in a reductionist framework. Specifically, we think that it might be possible to prove and further develop the following statement: Given system component models containing non-identifiable parameters, and the composition of the component models into a whole, the set of parameter values identified using the whole model must be a superset of the Cartesian product of the sets of the corresponding parameters' values identified from the component models. This result is exactly analogous to Condition 2.14 in Chapter 2, and can be thought of as the need for a lack of *emergent* behavior when system behavior has to be predicted from the combination of part models. We also note that when parameter covariation between parameters from different subsystems exists in the whole system's model, a generalization of the CSP fixing method might be applicable.

In Chapter 3, we discussed two tools to help with modeling of genetic circuits. `txtlsim` is a MATLAB® Simbiology® based toolbox bringing computer aided design capabilities to the TX-TL cell-free prototyping platform. This tool should help reduce the reliance on intuition alone for the design of new circuits, and in doing so allow for more rapid and larger scale design iterations. We also introduce mcmc_simbio, which we have packaged as a sub-toolbox within `txtlsim`, that performs Bayesian inference on parameters from multiple models and experimental data sets. The main utility of this toolbox, after its nominal use for estimating parameters for characterization purposes, is in studying the identifiability of parameters informed by different choices of model-experiment sets. This is a consequence of the fact that the parameter inference capabilities are both *Bayesian* and *concurrent*. The use of Bayesian inference allows us to directly visualize parameter identifiability, and indeed explore it a priori using artificial data, as was done in Section 2.7.1 in Chapter 2. The concurrence feature of the parameter inference allows for a common set

of parameters to be identified by an entire ensemble of different experiments, with each experiment informing some appropriate subset of parameters. This allows for different sets of experiment designs to be proposed, and then checked for parameter identifiability. Combined with the type of results in Chapter 2, which show that non-identifiability need not be completely eliminated from models to meet modeling objectives, these tools form a potentially powerful framework for applying computational design to synthetic biology applications.

Finally, in Chapter 4, we developed a mathematical framework for model reduction of transcription and translation reactions while still accounting for resource consumption. Tracking resource consumption is important in finite resource environments like TX-TL, and might even be important for quantifying the metabolic load that synthetic gene circuit expression places on cells. Detailed elongation models are able to capture resource binding and consumption, but are unnecessarily complicated for the purposes of modeling gene circuits. We showed that the incorporation of a *consumption reaction*, which consumes resources at a rate that is a function of the transcript production rate, allows for the reduction of the detailed elongation models into a single step (or a few steps, if nascent polymers need to be modeled), while still accounting for the consumption of resources.

In the process of performing the model reduction, we showed that species concentration coordinates did not allow for a separation of timescales argument to be made using singular perturbation theory. We noted that the timescale separation occurs in the rates of the reactions, and that the timescale separation in the rates of change of species concentrations is simply a consequence of this fact. Thus, while timescale separation can only be achieved for some network topologies when species concentration coordinates are used (specifically, when there is at least one species that only participates in slow reactions), it should be generally achievable when the system is written in our transformed coordinates. Thus, we show that these coordinates are a more natural way to perform timescale separation in biochemical networks.

In conclusion, modeling and computation hold promise of accelerating the develop-

ment of synthetic biology as a field, and will become indispensable as the scope of the systems we try to engineer surpasses what we can intuit.

Bibliography

[1] S. Arnold, M. Siemann, K. Scharnweber, M. Werner, S. Baumann, and M. Reuss, "Kinetic modeling and simulation of in vitro transcription by phage T7 RNA polymerase," *Biotechnology and Bioengineering*, vol. 72, no. 5, pp. 548–561, Mar. 2001. [Online]. Available: http://onlinelibrary.wiley.com/doi/10.1002/1097-0290(20010305)72:5<548::AID-BIT1019>3.0.CO;2-2/abstract

[2] E. August, "Parameter Identifiability and Optimal Experimental Design," in *International Conference on Computational Science and Engineering, 2009. CSE '09*, vol. 1, Aug. 2009, pp. 277–284.

[3] J. B. Barlow, W. H. Rae, and A. Pope, "Low Speed Wind Tunnel Testing," *INCAS Bulletin; Bucharest*, vol. 7, no. 1, pp. 133–135, 2015. [Online]. Available: https://search.proquest.com/docview/1667359793?pq-origsite=gscholar

[4] R. Bellman and K. J. Åström, "On structural identifiability," *Mathematical Biosciences*, vol. 7, no. 3, pp. 329–339, Apr. 1970. [Online]. Available: http://www.sciencedirect.com/science/article/pii/002555647090132X

[5] D. Bernstein and S. Bhat, "Nonnegativity, reducibility, and semistability of mass action kinetics," in *Proceedings of the 38th IEEE Conference on Decision and Control, 1999*, vol. 3, 1999, pp. 2206–2211 vol.3.

[6] J. R. Bowen, A. Acrivos, and A. K. Oppenheim, "Singular perturbation refinement to quasi-steady state approximation in chemical kinetics," *Chemical Engineering Science*, vol. 18, no. 3, pp. 177–188, Mar. 1963. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0009250963850034

[7] S. Brantl and E. G. H. Wagner, "Antisense RNA-mediated transcriptional attenuation: an in vitro study of plasmid pT181," *Molecular Microbiology*, vol. 35, no. 6, pp. 1469–1482, Mar. 2000. [Online]. Available: http://onlinelibrary.wiley.com/doi/10.1046/j.1365-2958.2000.01813.x/abstract

[8] S. Brantl and E. G. H. Wagner, "An Antisense RNA-Mediated Transcriptional Attenuation Mechanism Functions in Escherichia coli," *Journal of Bacteriology*, vol. 184, no. 10, pp. 2740–2747, May 2002. [Online]. Available: http://jb.asm.org/content/184/10/2740

[9] C. Reder, "Metabolic control theory: A Strutural approach," 1988. [Online]. Available: http://www.math.u-bordeaux1.fr/~creder/Recherche/BioSystems/JTB88.pdf

[10] O. Chiş, J. R. Banga, and E. Balsa-Canto, "GenSSI: a software toolbox for structural identifiability analysis of biological models," *Bioinformatics*, vol. 27, no. 18, pp.

2610–2611, Sept. 2011. [Online]. Available: http://bioinformatics.oxfordjournals.org/content/27/18/2610

[11] N. Christensen, R. Meyer, and A. Libson, "A Metropolis–Hastings routine for estimating parameters from compact binary inspiral events with laser interferometric gravitational radiation data," *Classical and Quantum Gravity*, vol. 21, no. 1, p. 317, Jan. 2004. [Online]. Available: http://iopscience.iop.org/0264-9381/21/1/023

[12] Dan Siegal-Gaskins, Z. A. Tuza, J. Kim, V. Noireaux, and R. M. Murray, "Gene Circuit Performance Characterization and Resource Usage in a Cell-Free "Breadboard"," *ACS Synthetic Biology*, vol. 3, no. 6, pp. 416–425, June 2014. [Online]. Available: http://dx.doi.org/10.1021/sb400203p

[13] D. Del Vecchio, A. J. Ninfa, and E. D. Sontag, "Modular cell biology: retroactivity and insulation," *Molecular Systems Biology*, vol. 4, p. 161, Feb. 2008. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2267736/

[14] D. A. Drew, "A mathematical model for prokaryotic protein synthesis," *Bulletin of Mathematical Biology*, vol. 63, no. 2, pp. 329–351, Mar. 2001. [Online]. Available: http://link.springer.com/article/10.1006/bulm.2000.0225

[15] M. B. Elowitz and S. Leibler, "A synthetic oscillatory network of transcriptional regulators," *Nature*, vol. 403, no. 6767, pp. 335–338, Jan. 2000. [Online]. Available: http://www.nature.com/nature/journal/v403/n6767/abs/403335a0.html

[16] D. Endy, "Foundations for engineering biology," *Nature*, vol. 438, no. 7067, pp. 449–453, Nov. 2005. [Online]. Available: http://www.nature.com/nature/journal/v438/n7067/full/nature04342.html

[17] M. Feinberg, "Chemical reaction netowork structure and the stability of complex isothermal reactors - i. the deficiency zero and deficiency one theorems," *Chemical Engineering Science*, vol. 42, no. 10, pp. 2229–2268, 1987. [Online]. Available: http://www.seas.upenn.edu/~jadbabai/ESE680/Fei87a.pdf

[18] M. Feinberg, "The existence and uniqueness of steady states for a class of chemical reaction networks," *Archive for Rational Mechanics and Analysis*, vol. 132, no. 4, pp. 311–370, Dec. 1995. [Online]. Available: http://link.springer.com/article/10.1007/BF00375614

[19] N. Fenichel, "Geometric singular perturbation theory for ordinary differential equations," *Journal of Differential Equations*, vol. 31, no. 1, pp. 53–98, Jan. 1979. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0022039679901529

[20] D. Foreman-Mackey, D. W. Hogg, D. Lang, and J. Goodman, "emcee: The MCMC Hammer," *Publications of the Astronomical Society of the Pacific*, vol. 125, no. 925, pp. 306–312, Mar. 2013, arXiv: 1202.3665. [Online]. Available: http://arxiv.org/abs/1202.3665

[21] K. Fujiwara and N. Doi, "Biochemical Preparation of Cell Extract for Cell-Free Protein Synthesis without Physical Disruption," *PLOS ONE*, vol. 11, no. 4, p. e0154614, Apr. 2016. [Online]. Available: http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0154614

[22] J. Garamella, R. Marshall, M. Rustad, and V. Noireaux, "The All E. coli TX-TL Toolbox 2.0: A Platform for Cell-Free Synthetic Biology," *ACS Synthetic Biology*, vol. 5, no. 4, pp. 344–355, Apr. 2016. [Online]. Available: http://dx.doi.org/10.1021/acssynbio.5b00296

[23] T. S. Gardner, C. R. Cantor, and J. J. Collins, "Construction of a genetic toggle switch in Escherichia coli," *Nature*, vol. 403, no. 6767, pp. 339–342, Jan. 2000. [Online]. Available: http://www.nature.com/nature/journal/v403/n6767/full/403339a0.html

[24] L. Goentoro, O. Shoval, M. Kirschner, and U. Alon, "The incoherent feedforward loop can provide fold-change detection in gene regulation," *Molecular cell*, vol. 36, no. 5, pp. 894–899, Dec. 2009. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2896310/

[25] J. Goodman and J. Weare, "Ensemble samplers with affine invariance," *Communications in Applied Mathematics and Computational Science*, vol. 5, no. 1, pp. 65–80, Jan. 2010. [Online]. Available: http://msp.org/camcos/2010/5-1/p04.xhtml

[26] O. P. C. Gqwaka, "A generic rate equation for catalysed, template-directed polymerisation and its use in computational systems biology," Thesis, Stellenbosch : Stellenbosch University, Dec. 2011. [Online]. Available: http://scholar.sun.ac.za/handle/10019.1/17825

[27] S. J. Greive, J. P. Goodarzi, S. E. Weitzel, and P. H. von Hippel, "Development of a "modular" scheme to describe the kinetics of transcript elongation by RNA polymerase," *Biophysical Journal*, vol. 101, no. 5, pp. 1155–1165, Sept. 2011.

[28] A. Grinsted, "Gwmcmc: An implementation of the goodman and weare mcmc sampler for matlab," GitHub Repository, March 2015.

[29] A. Gyorgy and D. Del Vecchio, "Limitations and trade-offs in gene expression due to competition for shared cellular resources," in *53rd IEEE Conference on Decision and Control*. IEEE, 2014, pp. 5431–5436. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=7040238

[30] A. Heyd and D. A. Drew, "A mathematical model for elongation of a peptide chain," *Bulletin of Mathematical Biology*, vol. 65, no. 6, pp. 1095–1109, Nov. 2003. [Online]. Available: http://link.springer.com/article/10.1016/S0092-8240%2803%2900076-4

[31] S. Hoops, S. Sahle, R. Gauges, C. Lee, J. Pahle, N. Simus, M. Singhal, L. Xu, P. Mendes, and U. Kummer, "COPASI—a COmplex PAthway SImulator," *Bioinformatics*, vol. 22, no. 24, pp. 3067–3074, Dec. 2006. [Online]. Available: http://bioinformatics.oxfordjournals.org/content/22/24/3067

[32] Y. Hori and R. M. Murray, "A state-space realization approach to set identification of biochemical kinetic parameters," in *Control Conference (ECC), 2015 European*. IEEE, 2015, pp. 2280–2285. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/7330878/

[33] C. Y. Hu, J. D. Varner, and J. B. Lucks, "Generating Effective Models and Parameters for RNA Genetic Circuits," *ACS Synthetic Biology*, June 2015. [Online]. Available: http://dx.doi.org/10.1021/acssynbio.5b00077

[34] M. Hucka, A. Finney, H. M. Sauro, H. Bolouri, J. C. Doyle, H. Kitano, a. t. r. o. t. S. Forum, A. P. Arkin, B. J. Bornstein, D. Bray, A. Cornish-Bowden, A. A. Cuellar, S. Dronov, E. D. Gilles, M. Ginkel, V. Gor, I. I. Goryanin, W. J. Hedley, T. C. Hodgman, J.-H. Hofmeyr, P. J. Hunter, N. S. Juty, J. L. Kasberger, A. Kremling, U. Kummer, N. L. Novère, L. M. Loew, D. Lucio, P. Mendes, E. Minch, E. D. Mjolsness, Y. Nakayama, M. R. Nelson, P. F. Nielsen, T. Sakurada, J. C. Schaff, B. E. Shapiro, T. S. Shimizu, H. D. Spence, J. Stelling, K. Takahashi, M. Tomita, J. Wagner, and J. Wang, "The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models," *Bioinformatics*, vol. 19, no. 4, pp. 524–531, Mar. 2003. [Online]. Available: http://bioinformatics.oxfordjournals.org/content/19/4/524

[35] L. Huynh and I. Tagkopoulos, "A Parts Database with Consensus Parameter Estimation for Synthetic Circuit Design," *ACS Synthetic Biology*, vol. 5, no. 12, pp. 1412–1420, Dec. 2016. [Online]. Available: https://doi.org/10.1021/acssynbio.5b00205

[36] I. Jonkers and J. T. Lis, "Getting up to speed with transcription elongation by RNA polymerase II," *Nature Reviews Molecular Cell Biology*, vol. 16, no. 3, pp. 167–177, Mar. 2015. [Online]. Available: http://www.nature.com/nrm/journal/v16/n3/full/nrm3953.html

[37] E. Karzbrun, J. Shin, R. H. Bar-Ziv, and V. Noireaux, "Coarse-Grained Dynamics of Protein Synthesis in a Cell-Free System," *Physical Review Letters*, vol. 106, no. 4, p. 048104, Jan. 2011. [Online]. Available: http://link.aps.org/doi/10.1103/PhysRevLett.106.048104

[38] A. Kohn, *Control of Gene Expression*. Springer Science & Business Media, Dec. 2012, google-Books-ID: GC3vBwAAQBAJ.

[39] P. Kokotovic, H. Khalil, and J. O'Reilly, *Singular Perturbation Methods in Control: Analysis and Design*, ser. Classics in Applied Mathematics. Society for Industrial and Applied Mathematics, Jan. 1999. [Online]. Available: http://epubs.siam.org/doi/book/10.1137/1.9781611971118

[40] S. Kosuri, J. R. Kelly, and D. Endy, "TABASCO: A single molecule, base-pair resolved gene expression simulator," *BMC Bioinformatics*, vol. 8, no. 1, p. 480, Dec. 2007. [Online]. Available: http://www.biomedcentral.com/1471-2105/8/480/abstract

[41] A. Kumar, P. D. Christofides, and P. Daoutidis, "Singular perturbation modeling of nonlinear processes with nonexplicit time-scale multiplicity," *Chemical Engineering Science*, vol. 53, no. 8, pp. 1491–1504, Apr. 1998. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0009250998000062

[42] T. G. Kurtz, "The Relationship between Stochastic and Deterministic Models for Chemical Reactions," *The Journal of Chemical Physics*, vol. 57, no. 7, pp. 2976–2978, Oct. 1972. [Online]. Available: http://scitation.aip.org/content/aip/journal/jcp/57/7/10.1063/1.1678692

[43] J. B. Lucks, L. Qi, V. K. Mutalik, D. Wang, and A. P. Arkin, "Versatile RNA-sensing transcriptional regulators for engineering genetic networks," *Proceedings of the National Academy of Sciences*, vol. 108, no. 21, pp. 8617–8622, May 2011. [Online]. Available: http://www.pnas.org/content/108/21/8617

[44] M. Marchisio and J. Stelling, "Computational design of synthetic gene circuits with composable parts," *Bioinformatics*, vol. 24, no. 17, pp. 1903–1910, Sept. 2008. [Online]. Available: http://bioinformatics.oxfordjournals.org/cgi/doi/10.1093/bioinformatics/btn330

[45] T. S. Moon, C. Lou, A. Tamsir, B. C. Stanton, and C. A. Voigt, "Genetic programs constructed from layered logic gates in single cells," *Nature*, vol. 491, no. 7423, pp. 249–253, Nov. 2012. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3904217/

[46] L. W. Nagel and D. Pederson, "Spice (simulation program with integrated circuit emphasis)," EECS Department, University of California, Berkeley, Tech. Rep. UCB/ERL M382, Apr 1973. [Online]. Available: http://www2.eecs.berkeley.edu/Pubs/TechRpts/1973/22871.html

[47] H. Niederholtmeyer, Z. Z. Sun, Y. Hori, E. Yeung, A. Verpoorte, R. M. Murray, and S. J. Maerkl, "Rapid cell-free forward engineering of novel genetic ring oscillators," *eLife*, vol. 4, p. e09771, Oct. 2015. [Online]. Available: https://elifesciences.org/articles/09771

[48] A. A. K. Nielsen, B. S. Der, J. Shin, P. Vaidyanathan, V. Paralanov, E. A. Strychalski, D. Ross, D. Densmore, and C. A. Voigt, "Genetic circuit design automation," *Science*, vol. 352, no. 6281, pp. aac7341–aac7341, Apr. 2016. [Online]. Available: http://www.sciencemag.org/cgi/doi/10.1126/science.aac7341

[49] V. Noireaux, R. Bar-Ziv, and A. Libchaber, "Principles of cell-free genetic circuit assembly," *Proceedings of the National Academy of Sciences*, vol. 100, no. 22, pp. 12 672–12 677, 2003. [Online]. Available: http://www.pnas.org/content/100/22/12672.short

[50] R. P. Novick, S. Iordanescu, S. J. Projan, J. Kornblum, and I. Edelman, "pT181 plasmid replication is regulated by a countertranscript-driven transcriptional attenuator," *Cell*, vol. 59, no. 2, pp. 395–404, Oct. 1989.

[51] M. S. Okino and M. L. Mavrovouniotis, "Simplification of mathematical models of chemical reaction systems," *Chemical reviews*, vol. 98, no. 2, pp. 391–408, 1998. [Online]. Available: http://pubs.acs.org/doi/pdf/10.1021/cr950223l

[52] R. J. Portugal, "Breadboard for electronic components or the like," US Patent USD228 136S, Aug., 1973. [Online]. Available: https://patents.google.com/patent/USD228136/en

[53] A. Raue, C. Kreutz, T. Maiwald, J. Bachmann, M. Schilling, U. Klingmüller, and J. Timmer, "Structural and practical identifiability analysis of partially observed dynamical models by exploiting the profile likelihood," *Bioinformatics*, vol. 25, no. 15, pp. 1923–1929, Aug. 2009. [Online]. Available: https://academic.oup.com/bioinformatics/article/25/15/1923/213246/Structural-and-practical-identifiability-analysis

[54] J. B. Rawlings, *Chemical reactor analysis and design fundamentals*. Madison, Wis: Nob Hill Publishing, 2002.

[55] V. N. A. P. Ryan Marshall, Jonathan Garamella, "High-throughput microliter-sized cell-free transcription-translation reactions for synthetic biology applications using the echo 550 liquid handler," Application Note, 2018.

[56] H. M. Sauro and B. Ingalls, "Conservation analysis in biochemical networks: computational issues for software writers," *Biophysical Chemistry*, vol. 109, no. 1, pp. 1–15, Apr. 2004. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0301462203002503

[57] M. Schauer and R. Heinrich, "Quasi-steady-state approximation in the mathematical modeling of biochemical reaction networks," *Mathematical Biosciences*, vol. 65, no. 2, pp. 155–170, Aug. 1983. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0025556483900585

[58] C. H. Schilling, D. Letscher, and B. O. Palsson, "Theory for the systemic definition of metabolic pathways and their use in interpreting metabolic function from a pathway-oriented perspective," *Journal of Theoretical Biology*, vol. 203, no. 3, pp. 229–248, Apr. 2000.

[59] S. Schuster, D. A. Fell, and T. Dandekar, "A general definition of metabolic pathways useful for systematic organization and analysis of complex metabolic networks," *Nature Biotechnology*, vol. 18, no. 3, pp. 326–332, Mar. 2000. [Online]. Available: http://www.nature.com/nbt/journal/v18/n3/full/nbt0300_326.html

[60] Z. Z. Sun, C. A. Hayes, J. Shin, F. Caschera, R. M. Murray, and V. Noireaux, "Protocols for Implementing an Escherichia coli Based TX-TL Cell-Free Expression System for Synthetic Biology," *Journal of Visualized Experiments : JoVE*, no. 79, Sept. 2013. [Online]. Available: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3960857/

[61] Z. Z. Sun, E. Yeung, C. A. Hayes, V. Noireaux, and R. M. Murray, "Linear DNA for Rapid Prototyping of Synthetic Biological Circuits in an Escherichia coli Based TX-TL Cell-Free System," *ACS Synthetic Biology*, vol. 3, no. 6, pp. 387–397, June 2014. [Online]. Available: http://pubs.acs.org/doi/abs/10.1021/sb400131a

[62] A. Swaminathan, V. Hsiao, and R. M. Murray, "Quantitative Modeling of Integrase Dynamics Using a Novel Python Toolbox for Parameter Inference in Synthetic Biology," *bioRxiv*, p. 121152, Mar. 2017. [Online]. Available: https://www.biorxiv.org/content/early/2017/03/27/121152

[63] R. S. Swanson and C. L. Gillis, "Wind-Tunnel Calibration and Correction Procedures for Three-Dimensional Models," *NACA Wartime Reports*, vol. L4E31, Oct. 1944. [Online]. Available: http://ntrs.nasa.gov/search.jsp?R=19930090922

[64] J. J. Tabor, H. Salis, Z. B. Simpson, A. A. Chevalier, A. Levskaya, E. M. Marcotte, C. A. Voigt, and A. D. Ellington, "A Synthetic Genetic Edge Detection Program," *Cell*, vol. 137, no. 7, pp. 1272–1281, June 2009. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2775486/

[65] M. K. Takahashi, J. Chappell, C. A. Hayes, Z. Z. Sun, J. Kim, V. Singhal, K. J. Spring, S. Al-Khabouri, C. P. Fall, V. Noireaux, R. M. Murray, and J. B. Lucks, "Rapidly Characterizing the Fast Dynamics of RNA Genetic Circuitry with Cell-Free

Transcription–Translation (TX-TL) Systems," *ACS Synthetic Biology*, Mar. 2014. [Online]. Available: http://dx.doi.org/10.1021/sb400206c

[66] S. Vajda, "Structural equivalence of linear systems and compartmental models," *Mathematical Biosciences*, vol. 55, no. 1, pp. 39–64, July 1981. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0025556481900122

[67] V. Van Breusegem and G. Bastin, "Reduced order dynamical modelling of reaction systems: A singular perturbation approach," in , *Proceedings of the 30th IEEE Conference on Decision and Control, 1991*, Dec. 1991, pp. 1049–1054 vol.2.

[68] N. Vora and P. Daoutidis, "Nonlinear model reduction of chemical reaction systems," *AIChE Journal*, vol. 47, no. 10, pp. 2320–2332, Oct. 2001. [Online]. Available: http://onlinelibrary.wiley.com/doi/10.1002/aic.690471016/abstract

[69] E. Walter and Y. Lecourtier, "Global approaches to identifiability testing for linear and nonlinear state space models," *Mathematics and Computers in Simulation*, vol. 24, no. 6, pp. 472–482, Dec. 1982. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0378475482906450