Parameter Inference for Stochastic Biological Models

Jeremy Revell

School of Computing

University of Newcastle

A thesis submitted for the degree of

Doctor of Philosophy

April 2019

Acknowledgements

Primarily, above all else, I would like to thank my supervisor, Dr. Paolo Zuliani, for his unwavering support and time dedicated throughout the duration of this research project.

Secondly, I would like to thanks all my friends, and colleagues within Newcastle university and the ICOS research group.

I would also like to thank the Newcastle/Liverpool/Durham BBSRC Doctoral Training Partnership for financial support, and for providing numerous professional self-development opportunities.

I would also like to thank Jim Faeder, colleagues, and friends who hosted me during my BBSRC internship at the University of Pittsburgh. (Hail to Pitt!)

Abstract

Parameter inference is the field concerned with estimating reliable model parameters from data. In recent years there has been a trend in the biology community toward single cell technologies such as fluorescent flow cytometry, transcriptomics and mass cytometry: providing a rich array of stochastic time series and temporal distribution data for analysis. Deterministically, there are a wide range of parameter inference and global optimisation techniques available. However, these do not always scale well to non-deterministic (*i.e.*, stochastic) settings — whereby the temporal evolution of the system can be described by a chemical master equation for which the solution is nearly always intractable, and the dynamic behaviour of a system is hard to predict. For systems biology, the inference of stochastic parameters remains a bottleneck for accurate model simulation.

This thesis is concerned with the parameter inference problem for stochastic chemical reaction networks. Stochastic chemical reaction networks are most frequently modelled as a continuous time discrete-state Markov chain using Gillespie's stochastic simulation algorithm.

Firstly, I present a new parameter inference algorithm, SPICE, that combines Gillespie's algorithm with the cross-entropy method. The cross-entropy method is a novel approach for global optimisation inspired from the field of rare-event probability estimation. I then present recent advances in utilising the generalised method of moments for inference, and seek to provide these approaches with a direct stochastic simulation based correction. Subsequently, I present a novel use of a recent multi-level tau-leaping approach for simulating population moments efficiently, and use this to provide a simulation

based correction to the generalised method of moments. I also propose a new method for moment closures based on the use of Padé approximants.

The presented algorithms are evaluated on a number of challenging case studies, including bistable systems — *e.g.*, the Schlögl System and the Genetic Toggle Switch — and real experimental data. Experimental results are presented using each of the given algorithms. We also consider 'realistic' data — *i.e.*, datasets missing model species, multiple datasets originating from experiment repetitions, and datasets containing arbitrary units (*e.g.*, fluorescence values). The developed approaches are found to be viable alternatives to existing state-of-the-art methods, and in certain cases are able to outperform other methods in terms of either speed, or accuracy.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Every living cell can be viewed as a complex network of biochemical pathways. The core functionality of life, *i.e.*, metabolism, signal transduction, and cellular differentiation are robustly controlled by rich networks of genes and molecular reactions. Through experimental evidence, molecular biology and bioinformatics have uncovered large quantities of interacting, multifunctional elements — such as RNA, proteins, and transcription factors — that often behave in a non-linear, but coherent fashion. Forming a complete intuitive description of any complex biological system is near impossible, thus a combination of experimental and computational techniques are required [72].

1.1.1 Computational Systems Biology

Computational systems biology aims to construct mechanistic, dynamic models to achieve a systems level understanding of biological processes [72]. Primarily, the development of accurate, reliable simulations is an invaluable tool for the analysis of cellular systems [128]. The primary goals of modelling are two-fold. Firstly models can be used to validate our understanding of the underlying biochemical mechanisms. Secondly, models can be used make informative predictions about cellular behaviour. Ultimately, *in silico* (computational) experiments are a powerful ally to achieving the goals of synthetic biology, which aims to provide safe

and reliably engineered organisms that satisfy model design requirements [39, 56]. At the highest level of complexity, a golden standard for systems biology is the so called '*whole-cell*' model. A whole-cell model provides a complete account for the molecular dynamics of a cell during its entire life cycle, and is able to predict phenotype from genotype — see for example, the *Mycoplasma genitalium* whole-cell model [68]. In future, it is anticipated that such complex biological models will pave way to a new generation of genetically optimised crops, drug-development, biofuel synthesis, and personalised medicine [22].

1.1.2 The Emergence of Stochastic Modelling

Chemical reaction network models are central to systems biology, forming sets of reaction rules that govern the temporal evolution of a system [128]. Traditionally, these models have been deterministic — using continuous, coupled ordinary differential equations. Only in recent years has the role of stochasticity, both *intrinsic* and *extrinsic*, become better understood in the context of biological processes. Experimental evidence has confirmed that gene expression can occur in transient, stochastic bursts [14, 98], while low molecular counts and rare reaction events can account for profound differences from deterministic dynamics [87]. It is reasonable to conceive that many, if not most biological systems, will consist of autoregulatory network motifs that operate around unstable states — such that they are able to adapt and respond to changes within their environment. Specific examples of cellular phenomena that rely on stochastic noise include chemotaxis and polarization [127], and oscillations within the MinCDE system used by *E.coli* prior to cell division [63].

1.1.3 Parameter Inference

Parameter inference is the field concerned with estimating reliable model parameters from experimental data, and is vital to achieving the aims of computational systems biology [116]. Given a well defined model and its parameters, producing a time-series trajectory via forward simulation is an easy task. On the other hand, inferring a stochastic parameter from time-series snapshot data is not so easy: and remains a hard challenge for computational systems biology [128]. His-

torically, obtaining high resolution time-series data accounting for the dynamics of a molecular species within a single cell has been problematic. Recent years, however, have seen a new generation of single cell technologies including fluorescent flow cytometry, transcriptomics, and mass cytometry — providing a rich array of stochastic temporal data for analysis.

For deterministic systems there exists a vast array of local and global optimisation techniques which have been adapted to perform parameter estimation [90]. Unfortunately, these do not always apply well to non-deterministic systems — whereby the dynamics are described by a probability distribution for which the solution is intractable, and whose temporal evolution is hard to predict. For computational systems biology, the estimation of reliable parameters remains a challenge and bottleneck toward accurate model simulation.

Categorising Parameter Inference Current state-of-the-art methods for parameter inference fall into a wide range of categories. Broadly speaking, the key properties of parameter inference routines for stochastic biological models can be categorised as follows:

- *Maximum-likelihood* vs *likelihood-free* objective optimisation.
- *Exact* vs *inexact* solutions to the chemical master equation.
- *Gradient-based* vs *gradient-free* optimisation.

Maximum Likelihood and Likelihood Free Estimation *Maximum likelihood estimation* (MLE) is one of the most widely used approaches in the parameter inference paradigm. MLE routines aim to compute an estimate of the unknown model parameters, such that the conditional probability of the parameters upon observed data is maximised. Given complete — 'perfect' — data, whereby all the species and their temporal transitions are captured within a system, the parameters that maximise the likelihood can be calculated analytically [128]. In practice, MLE methods must deal with partially observed systems, and use appropriately selected heuristics to compute approximate maximum likelihood estimates. Computing the likelihood of parameters based on simulations and experimental data can be an expensive challenge [128]. Furthermore, for

many systems the exact form of the likelihood function is not easy to compute. This motivates the use of '*likelihood-free*' objective functions (*e.g.*, least squares distances), which when optimised produce approximate parameters that converge to the real maximum likelihood estimates under certain assumptions (*e.g.*, normally distributed data).

Exact versus Inexact Simulation Central to all parameter inference techniques for biological models is the ability to generate samples of the time-course dynamics. 'Exact' samples (to the solution of the chemical master equation) can be generated using stochastic algorithms such as Gillespie's direct method [44]. However, they are computationally intensive and do not scale well to larger systems. Recent advances have seen a host of inexact approximations including tau-leaping [20, 43], system size expansion [100], and moment-closure approximations [34].

Gradient and Gradient Free Descent Gradient-based methods are predominantly used in deterministic, maximum-likehood estimation. The gradient-based methods work by firstly (i) approximating the derivative of the likelihood in the local landscape, and then (ii) using the gradient to take a step in parameter state-space [7]. In stochastic systems, the gradient is often noisy and non-convex, and care must be taken to avoid local minima. On the other hand, gradient-free methods instead navigate the parameter state-space by other heuristics.

1.2 Related Work

An early method for parameter inference within stochastic chemical reaction networks is the simulated maximum likelihood (SML) approach by Tian *et al.* [123]. SML utilizes stochastic differential equations to compute an approximation of the system dynamics, and a genetic optimisation algorithm to maximize a joint transitional likelihood function. More recently, efforts by Horvath *et al.* [62] have been made to integrate the well-known 2-step Expectation-Maximisation (EM) algorithm [30] with direct simulation via Gillespie's stochastic simulation algorithm [44]. One advantage of the EM method over SML is that the EM does not require

prior bounds on the parameters. Another MLE method, the stochastic gradient descent (SGD) [126], derives a general formula for calculating the gradient of the likelihood function and combines Gillespie-type exact stochastic simulations [44] with a reversible jump Markov Chain Monte Carlo sampler. However, much like the SML, the SGD method requires prior bounds on the parameters. More recent work by Daigle *et* al. developed a new Monte Carlo Expectation-Maximisation with Modified Cross-Entropy Method (MCEM^2) [26]. MCEM^2 combines an efficient, ascent-based EM algorithm [19] with the Cross-Entropy method [106] — a recent advance in the field of rare event probability estimation [105, 125].

Another large category of parameter estimation techniques are those that utilise Bayesian inference. Theoretical advantages of Bayesian methods over the frequentist approaches such as the EM and SML are the ability to incorporate notions of experimental error into the posterior distribution, the ability to yield confidence intervals, and the better handling of unobserved model variables [15, 48]. Approximate Bayesian computation (ABC) expands on previous work by becoming 'likelihood-free' [128], and recent advances in sequential Monte Carlo (SMC) samplers have further improved these estimation techniques [118, 124]. In addition, the methods developed by Toni *et* al. [124] and Liepe *et al.* [83] are able to perform model selection, distinguishing between competing model descriptions. A practical application of Bayesian inference can be seen in [36], where it was used alongside model selection and single-cell fluorescent lineage data to study the dynamics of Nanog regulation within mouse embryonic stem cells.

More recent classes of methodologies focus on using approximations to the solution of the chemical master equation — which is otherwise intractable for all but the simplest of systems. One approach taken is to use dynamic truncation of the state space [5] or finite state projection methods [27] — methods which truncate the infinite dimensions of the chemical master equations solution by ignoring the lowest probability states. These methods still suffer from a large computational overhead and can add an integration bias that make the approximations worse the longer the systems are simulated for. Other recent work includes the moment-closure approximations [34, 38, 42, 55, 85, 132], which construct ODEs describing the time evolution for the mean, variance, skewness (and so on) of the underlying probability distribution [50]. These methods have provided promising

results in recent years, however, can fall victim to poor approximations when assumptions made about the model are false, and where the data is insufficient to approximate higher order moments. Finally, another noteworthy approximation of the chemical master equation is system size expansion — using van Kampen's expansion [38, 100] and other solutions of the Fokker-Planck equation such as the linear noise approximation [82]. All of the above simulation approaches can be combined with parameter estimation frameworks, for example, Komorowski *et al.* combine linear noise approximations with Bayesian inference [74].

1.2.1 Related Software and Tools

COPASI COPASI — the COmplex PAthway SImulator — is a stand-alone software application that aids in the simulation and analysis of chemical reaction networks [61]. It includes the ability to construct, simulate, and read file standards such as Systems Biology Markup Language (SBML) models [64]. Of particular interest is COPASI's ability to perform simulations with a wide range of stochastic methods and differential equations — and its range of commonly used parameter estimation routines. The full list of implemented parameter inference techniques are: Evolutionary Programming [9]; Evolution Strategy (Stochastic Ranking) [110]; Genetic Algorithm [9]; Genetic Algorithm (Stochastic Ranking) [110]; Hooke and Jeeves [60]; Levenberg-Marquardt [86]; Nelder-Mead [93]; Particle Swarm [70]; Praxis [16]; Simulated Annealing [71]; Steepest Descent [37]; Truncated Newton [92]. Many of these algorithms, especially those that navigate the parameter space based on computing a gradient, are particularly suited to optimisation in deterministic settings — and do not scale well to stochastic systems. However, they are nonetheless useful in computing parameter estimates for robust models, and are used as the back-end to other modelling environments such as Vcell [112]. Recent work has expanded the accuracy of COPASI by using a multiple shooting technique for stochastic and deterministic simulation [12].

StochKit & StochSS StochKit [111] and StochSS [32] — part of the Stochastic Simulation Service — is an integrated environment that allows for modelling and simulation through a web-based user interface. Like COPASI, StochSS utilises

XML (SBML) formatting, and provides a range of tools including parameter sensitivity analysis, and parameter estimation. Parameter estimation, through the StochOptim module, implements the Monte Carlo Expectation-Maximization with Modified Cross-Entropy Method (MCEM^2) algorithm [26], and yields a multivariate parameter estimate.

CERENA CERENA [69] — the ChEmical REaction Network Analyzer — is a toolbox for simulating and analysing stochastic chemical reaction networks based on using approximations of the chemical master equation. Specifically, CERENA implements standard Gillespie-type simulations [44], alongside finite state projection [91], system size expansion [100], and moment closure approximations [34]. However, CERENA does not implement its own parameter estimation routines, and acts as an intermediary tool.

Other Tools MEANS [1] — for Moment Expansion Approximation, iNference and Simulation — is a python package that implements moment closure approximations and parameter inference through the IPython Jupyter environment. Parameter inference within MEANS is performed through the use of SciPy's fmin function. MOCA [115] — a MOment Closure Analysis tool — allows numerical analysis of various moment closure approximations, while SHAVE [78] — Stochastic Hybrid Analysis of markoV population modEls — aids in the construction of hybrid moment-state based representations of the chemical master equation. Another tool, iNA [122] — the Intrinsic Noise Analyzer — allows for simulation and analysis via van Kampen's system size expansion of the chemical master equation [100]. Similarly to iNA, a recent tool LNA++ [67] provides an efficient implementation of the linear noise approximation, alongside first and second order sensitivity estimates. Like CERENA, iNA and LNA++ act as generalised intermediary tools and do not possess inference capabilities. Within more formal frameworks, Bio-PEPA provides a modelling language enabling the abstraction and analysis of biochemical networks, interfacing with tools such as the model checker PRISM [23]. More recently, ProPPA — the Probabilistic Programming Process Algebra — extends the work of Bio-PEPA and enables machine learning tools to perform inference and uncertainty analysis [40]. Finally, BioNetGen [53]

— the Biological Network Generator — is a rule-based modelling environment for chemical reaction networks. It includes a range of tools for simulation including differential equations and stochastic simulation using Gillespie-type methods. BioNetGen allows for Bayesian based inference [73] using the in-house tool PTempEst — which stands for Parallel Tempering for Estimation. Parallel tempering [33] is a global optimisation method closely related to simulated annealing [71] and associated Markov Chain Monte Carlo sampling techniques.

1.3 Aims and Objectives

The aim of this work is to

investigate novel computational approaches to parameter inference for stochastic biological models.

Specifically, the goals were to:

- investigate a novel approach for parameter inference based on the Cross-Entropy method, and apply it to stochastic biological models.
- to investigate promising developments in scalable moment approximations, for the use in generalised method of moments based inference techniques.
- to compare and evaluate the performance of the developed parameter inference approaches against other state-of-the-art methodologies on numerous challenging case studies.

1.4 Thesis Outline

In this section I introduce the outline and contributions of this thesis.

- **Chapter 2** provides an mathematical framework for modelling and describing stochastic chemical reaction networks.
- **Chapter 3** introduces the Cross-Entropy method for generalised optimisation, and extends the framework to stochastic chemical kinetics. Within

this chapter I present SPICE, an algorithm for stochastic parameter inference using the Cross-Entropy method. Some of the material presented in this chapter was published in [101].

- **Chapter 4** introduces moment approximations, the generalised method of moments, and its application to parameter inference within stochastic chemical reaction network models. I propose a new method for parameter inference, based on combining the generalised method of moments with a novel adapative multi-level tau-leaping technique (MLGMM). Within this chapter I also provide insight into the failure of some moment closure approximations, and propose a novel technique for constructing closures based on using Padé approximants.

- **Chapter 5** implements the new SPICE algorithm for inference on numerous case studies — including a model of the Genetic Toggle Switch for which real experimental data was used.

- **Chapter 6** gives an overview of the conclusions, and final remarks for future work.

1.5 List of Publications

Portions of the work within this thesis have been documented in the following publications:

- **Jeremy Revell** and Paolo Zuliani. "Stochastic rate parameter inference using the cross-entropy method", in *Computational Methods in Systems Biology*, Češka, Milan and Šafránek, David, pages 146–164, 2018. Springer International Publishing.

Chapter 2

Stochastic Chemical Kinetics

In this chapter, I outline useful mathematical concepts and algorithms for the modelling, simulation, and analysis of stochastic biochemical reaction networks.

- In Section 2.1, I present some background theory toward continous time Markov chain processes.
- Within Section 2.2 I outline the framework for describing stochastic chemical reaction networks.
- Section 2.3 presents the chemical master equation for describing the probability distribution of a stochastic process, and the methods used to sample its solutions.
- Section 2.4 presents the complete data likelihood for exact stochastic simulation trajectories, which is used extensively in parameter inference.

2.1 Markov Chains, and Markov Processes

A *Markov process* within a discrete (or continuous) state-space is a stochastic process satisfying the *Markov property* — *i.e.*, the condition that all future states are independent of the past states. A discrete-time Markovian process is called a *Markov chain* and describes the temporal evolution of a sequence of random variables. The transitions within the chain are stochastic — with future transition

probabilities depending on only the current. Mathematically, these transitions are described using a *transition kernel*, and is standard material in probability theory [103].

Definition 2.1 (Transition Kernel). *A* transition kernel *for a Markov process* $\{X_t\}$ *with state-space* $S \subseteq \mathbb{R}^n$ *is a map* $K : S \times \mathcal{B} \to [0, 1]$ *such that*

(i) $\forall x \in S$, $K(x, \cdot)$ *is a probability measure;*

(ii) $\forall A \in \mathcal{B}$, $K(\cdot, A)$ *is measurable,*

where $\mathcal{B}$ *denotes the Borel-*σ*-algebra on* S*.*

For the finite or countable case, K is a transition matrix with elements

$$k_{ij} = P(X_{t+1} = j | X_t = i), \qquad i, j \in S,$$

where each X_t is a random variable. For the continuous case, the kernel also represents the conditional density $K(x, y)$ of the transition $K(x, \cdot)$ such that $P(X_{t+1} \in A | X_t = x) = \int_A K(x, y)dy$.

Definition 2.2 (Markov Chain [103]). *The sequence of random variables* $\{X_1, \ldots, X_t\}$ *generated by a discrete-time Markov process together with a transition kernel,* K*, define a* Markov chain*, providing the Markov property has been satisfied. Specifically for any finite sequence* $x_1, x_2, \ldots, x_t \in S$*,*

$$\begin{aligned} P(X_{t+1} \in A | X_1 = x_1, X_2 = x_2, \ldots, X_t = x_t) &= P(X_{t+1} \in A | X_t = x_t) \\ &= \int_A K(x_t, dx). \end{aligned} \tag{2.1.1}$$

Definition 2.3 (Invariance). *A* σ*-finite measure* π *is* invariant *given the transition kernel* K *if*

$$\pi(A) = \int_S K(x, A)\pi(dx), \qquad \forall A \in \mathcal{B}. \tag{2.1.2}$$

If π is an invariant probability measure with respect to a kernel, π is said to be the *stationary distribution* of the homogeneous Markov chain. The resulting Markov chain is also said to be stationary.

Definition 2.4 (Reversible). *A stationary Markov chain* (X_t) *is said to be* reversible *if the distribution of* X_{t+1} *conditional upon* $X_{t+2} = x$ *is equal to the distribution of* X_{t+1} *conditional on* $X_t = x$*.*

As a consequence, in the case of reversibility, the direction of time is inconsequential upon the dynamics of the chain.

Definition 2.5 (Detailed Balance). *A Markov chain (X_t) satisfies the* detailed balanced condition *if there exists a function π satisfying*

$$K(y,x)\pi(y) = K(x,y)\pi(x) \qquad \forall x, y \in S \tag{2.1.3}$$

Theorem 2.1. *If a Markov chain with a kernel K satisfies the detailed balance condition with a probability density function π then:*

(i) The density π is the invariant (or stationary) density of the chain.

(ii) The chain is reversible.

Proof. Property (i) follows from the detailed balanced condition (2.1.3) for any measurable set A,

$$\begin{aligned}\int_S K(y,A)\pi(y)dy &= \int_S \int_A K(y,x)\pi(y)dxdy \\ &= \int_S \int_A K(x,y)\pi(x)dxdy = \int_A \pi(x)dx,\end{aligned}$$

since $\int K(x,y)dy = 1$. The appearance of the kernel K and an invariant density π make it clear that detailed balance and reversibility are equivalent properties [103]. □

Given the convergence to the stationary distribution π in a state space S, Markov chains also have the interesting property that the empirical average for a Lebesgue integrable function h,

$$\frac{1}{T}\sum_{t=1}^{T} h(X_t),$$

converges to $\mathbb{E}_\pi[h(X)]$ *almost surely*. Thus the empirical averages for $h()$ converge to a probabilistic average of the function over the stationary distribution.

2.2 Stochastic Chemical Kinetics

Traditional models of chemical systems have used continuous descriptions of molecular concentrations via deterministic reaction rate differential equations

(RRE). These form macroscopic approximations of the underlying system, which in reality is composed of discrete-time events that occur probabilistically with a rate dependent on the systems chemical properties [129]. For many systems, the RRE approximation is inadequate, and fails to account for biochemical systems with low molecular numbers [87], rare reactions, or transient burst behaviour events — *i.e.*, stochastic gene expression [14, 98]. A generalised stochastic description of biochemical kinetics is therefore required.

One general approach is to consider each particle in a system undergoing a random walk due to Brownian motion. When two particles approach (or collide), reactions occur stochastically according to a set of rules. However, such a system is infeasible to solve in practice, and the computational burden of tracking all particle positions and velocities can quickly exceed hardware limits [128].

For this thesis I will consider the standard description of stochastic chemical kinetics using *continuous-time Markov chains* (CTMCs). Explicitly, I will assume the system is well-mixed, dilute, contained within a compartment of fixed volume, Ω, and in thermal equilibrium at an absolute temperature [45].

Definition 2.6 (Well-mixed). *A system is deemed 'well-mixed' if the mean free path — the average distance travelled by a moving molecule between successive collisions or reactions — is much less than the length scale of the hosting compartment. Thus, the dynamics of the system depends only on the total number of molecules and can be deemed roughly isotropic,* i.e.*, the physical description of the system is spatially invariant.*

Definition 2.7 (Dilute). *A 'dilute' system is one in which the total volume of the compartment, Ω, is much greater than the combined volume of all the molecules within it. Thus, we can treat all molecules as point particles.*

Given a set of n species, the state of the system at any time within the CTMC is represented by the random vector $\boldsymbol{x}(t) = (x_1(t), \ldots, x_n(t))$, where x_i represents the number of molecules of the i-th species, S_i, at time t, for $i \in \{1, \ldots, n\}$. The probabilistic state transitions that occur within the CTMC are governed by a set of reaction rules, which form a chemical reaction network (CRN) — see Definition 2.8.

Definition 2.8 (Chemical Reaction Network). *A chemical reaction network is a set* $\{\mathcal{R}_j : j \in \{1, \ldots, m\}\}$ *where* $\mathcal{R}_j$ *denotes the* j*-th reaction of type:*

$$\mathcal{R}_j \quad : \quad v_{j,1}^- S_1 + \ldots + v_{j,n}^- S_n \xrightarrow{\theta_j} v_{j,1}^+ S_1 + \ldots + v_{j,n}^+ S_n, \tag{2.2.1}$$

and the vectors $\boldsymbol{v}_j^- = (v_{j,1}^-, \ldots, v_{j,n}^-)$ *and* $\boldsymbol{v}_j^+ = (v_{j,1}^+, \ldots, v_{j,n}^+)$ *represent the stoichiometry of the reactants and products respectively.*

It is useful to let $\boldsymbol{v}_j \in \mathbb{Z}^n$ denote the overall (non-zero) state change vector for $\mathcal{R}_j$, such that $\boldsymbol{v}_j = \boldsymbol{v}_j^+ - \boldsymbol{v}_j^-$, for $j \in \{1, \ldots, m\}$. The vectors $\{\boldsymbol{v}_j : j \in \{1, \ldots, m\}\}$ form the state change matrix. Assuming standard mass action kinetics, the reaction $\mathcal{R}_j$ proceeds at a rate directly proportional to the number of combinations of reactant molecules, leading to a propensity function (also known as the hazard function [128]) of the form:

$$\begin{aligned} h_j(\boldsymbol{x}, \boldsymbol{\theta}) &= \theta_j g_j(\boldsymbol{x}) \\ &= \theta_j \Omega \prod_{i=1}^{n} \frac{x_i!}{(x_i - v_{j,i}^-)!\, \Omega^{v_{j,i}^-}} \\ &= \theta_j \Omega \prod_{i=1}^{n} \binom{x_i}{v_{j,i}^-} \frac{1}{\Omega^{v_{j,i}^-}}, \end{aligned} \tag{2.2.2}$$

where θ_j is the deterministic rate constant, Ω is the total compartment volume, and the time-dependence of $\boldsymbol{x}$ has been neglected [45, 128]. If $\{\exists\, i \mid v_{j,i}^- > x_i\}$, then the j-th propensity, $h_j = 0$, and the reaction is unable to fire. See Example 2.1 for a demonstration of using Equation (2.2.2) for a generic CRN.

Example 2.1 (Mass-Action Propensity Calculations). *Consider the following four generic reactions ranging from zeroth to second order:*

$$\begin{aligned} \mathcal{R}_1 &: \quad \emptyset \xrightarrow{\theta_1} \cdots \quad \textit{(e.g., synthesis)}, \\ \mathcal{R}_2 &: \quad A \xrightarrow{\theta_2} \cdots \quad \textit{(e.g., degradation)}, \\ \mathcal{R}_3 &: \quad A + B \xrightarrow{\theta_3} \cdots \quad \textit{(e.g., binding)}, \\ \mathcal{R}_4 &: \quad A + A \xrightarrow{\theta_4} \cdots \quad \textit{(e.g., dimerisation)}, \end{aligned}$$

where A *and* B *are chemical species,* $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_4)$ *denote the rate constants, and* $\emptyset$ *is used to represent a null species. Using Equation (2.2.2), the correspond-*

ing mass-action propensity functions are:

$$h_1(\boldsymbol{x}, \boldsymbol{\theta}) = \theta_1 \Omega, \qquad h_2(\boldsymbol{x}, \boldsymbol{\theta}) = \theta_2 x_A,$$
$$h_3(\boldsymbol{x}, \boldsymbol{\theta}) = \frac{\theta_3}{\Omega} x_A x_B, \qquad h_4(\boldsymbol{x}, \boldsymbol{\theta}) = \frac{\theta_4}{\Omega} \frac{x_A(x_A - 1)}{2!},$$

where $\boldsymbol{x} = (x_A, x_B)$ *are the numbers of each species, and* Ω *is the volume.*

In general, stochastic chemical kinetics are not restricted to mass action form, and thus the propensity functions $h_j(\boldsymbol{x}, \boldsymbol{\theta})$ may be specified separately, where $\boldsymbol{\theta}$ can be any vector of input parameter values.

2.3 Chemical Master Equation

Let $P(\boldsymbol{x}(t)|\boldsymbol{x}_0(t_0))$ denote the probability distribution for the system to be in state $\boldsymbol{x}$ at time t, given the initial state $\boldsymbol{x}_0$ at time t_0. For an infinitesimal change in t, dt, the distribution $P(\boldsymbol{x}(t + dt)|\boldsymbol{x}_0(t_0))$ is found by subtracting the probability to leave state the $\boldsymbol{x}$ from the probability to transition to state $\boldsymbol{x}$ from all other states $\boldsymbol{x}' \neq \boldsymbol{x}$. The probability for a reaction $\mathcal{R}_j$ to occur within a infinitesimally small time interval, dt, is $h_j(\boldsymbol{x}(t), \boldsymbol{\theta})dt$ — where $h_j(\boldsymbol{x}(t), \boldsymbol{\theta})$ is defined as in Equation (2.2.2). This leads to the following expression for the new distribution:

$$P(\boldsymbol{x}(t + dt)|\boldsymbol{x}_0(t_0)) = P(\boldsymbol{x}(t)|\boldsymbol{x}_0(t_0)) + dt \left(\sum_{j=1}^{m} h_j(\boldsymbol{x}(t) - \boldsymbol{v}_j, \boldsymbol{\theta}) P(\boldsymbol{x}(t) - \boldsymbol{v}_j|\boldsymbol{x}_0(t_0)) - \sum_{j=1}^{m} h_j(\boldsymbol{x}(t), \boldsymbol{\theta}) P(\boldsymbol{x}(t)|\boldsymbol{x}_0(t_0)) \right). \tag{2.3.1}$$

By rearranging Equation (2.3.1), dividing by dt, and taking the limit $dt \to 0$, we can derive the form of the *chemical master equation* (CME) — see [45] for a rigorous approach.

Definition 2.9 (Chemical Master Equation [45]). *The chemical master equation is defined as:*

$$\partial_t P(\boldsymbol{x}(t)) = \sum_{j=1}^{m} h_j(\boldsymbol{x}(t) - \boldsymbol{v}_j, \boldsymbol{\theta}) P(\boldsymbol{x}(t) - \boldsymbol{v}_j) - \sum_{j=1}^{m} h_j(\boldsymbol{x}(t), \boldsymbol{\theta}) P(\boldsymbol{x}(t)) \tag{2.3.2}$$

where the shorthand notation $P(\boldsymbol{x}(t)) = P(\boldsymbol{x}(t)|\boldsymbol{x}_0(t_0))$ *has been used.*

Analytically, Equation (2.3.2) is often intractable. Numerically, the first-order differential equation in Equation (2.3.2) will generally belong to a very large state space that typically exceeds the memory capacity for realistic systems, making the computation infeasible. This can be seen by noting that the state-space is generally unbounded, and thus the number of states $\boldsymbol{x} \in \mathbb{N}^n$ is infinite for all but the simplistic of systems. If the system is relatively simple and the dynamics well understood, this curse of dimensionality can be combated by truncating the state-space in regions of negligible probability — in an approximation known as *finite state projection* (FSP) [91]. However, there exists a plethora of algorithms that are designed to simulate 'exact' samples from the CME — whereby *exact* means that these algorithms provably sample from stochastic chemical kinetics that underpin the CME. These are known as stochastic simulation algorithms (SSAs).

2.3.1 Chemical Master Equation - Exact Simulation

Gillespie's direct method, or SSA, is a method for sampling exact paths of the CME, which is often otherwise computationally intractable as explained above.

Suppose we are given the system state $\boldsymbol{x}(t)$ with the propensities $h_j \equiv h_j(\boldsymbol{x}(t), \boldsymbol{\theta})$ and state-change vectors $\boldsymbol{v}_j$ for $j \in \{1, \ldots, m\}$. The probability of any individual reaction $\mathcal{R}_j$ firing in the next infinitesimal time interval $[t, t+dt)$ is given by $h_j dt$. Thus the total propensity function,

$$\begin{aligned} h_0(\boldsymbol{x}, \boldsymbol{\theta}) &= \sum_{j=1}^{m} h_j(\boldsymbol{x}, \boldsymbol{\theta}) \\ &= \sum_{j=1}^{m} h_j = h_0 \end{aligned} \tag{2.3.3}$$

can be used to sample a time, τ, that remains until any one of the reactions fire. This is done by noting that the time, τ, is an exponentially distributed random variable with mean $1/h_0$ [44, 128]. To select the reaction index j, the SSA samples from the discrete distribution $\{h_1/h_0 \ldots, h_m/h_0\}$. The system time can then be updated $(t = t + \tau)$, and the system species are updated using the corresponding state change matrix for reaction $\mathcal{R}_j$, *i.e.*, $\boldsymbol{x}(t + \tau) = \boldsymbol{x}(t) + \boldsymbol{v}_j$.

This process is repeated until the system time, t, exceeds some fixed simulation endpoint time T, or until no more possible reactions can occur (*i.e.*, $h_0 = 0$). Gillespie's SSA is outlined below in Algorithm 1 [44, 128]. Over recent years,

Algorithm 1 Gillespie Algorithm

procedure FORWARD SIMULATE
  $t \leftarrow t_0$ // set the current system time
  $\boldsymbol{x}(t) \leftarrow \boldsymbol{x}_0$ // set the initial state
  while $t < T$ **do**
    $h_0 \leftarrow \sum_{j=1}^{m} h_j$ // set the total propensity
    if $h_0 = 0$ **then**
      then break
    $\tau \sim \exp(1/h_0)$ // time to next reaction
    if $t + \tau > T$ **then**
      then break
    rxn $\leftarrow$ sample $\{h_1/h_0 \ldots, h_m/h_0\}$ // select reaction firing
    $\boldsymbol{x}(t + \tau) \leftarrow \boldsymbol{x}(t) + \boldsymbol{v}_{\mathsf{rxn}}$ // update state
    $t \leftarrow t + \tau$ // update time

many improvements and variants of the SSA have arisen — some of which are best suited to certain chemical systems over others. These include variations on the direct method such as the Next Reaction Method [41], the Logarithmic Direct Method [81], Optimised Direct Method [21], Sorting Direct Method [88], and Partial Propensity Methods [96]. In addition, inexact methods such as tau-leaping have been proposed to speed up simulation times [43]. In Section 2.3.2, I outline the optimised tau-leaping algorithm [20].

2.3.2 Chemical Master Equation - Inexact Methods

Despite some advancements, Gillespie's direct method often has the drawback of scaling poorly in terms of performance for kinetic systems involving large molecular copy numbers, as well as rapid reaction rates. Intuitively, this is because the SSA works to track every reaction firing event, and is constrained to updating every propensity function after reaction firing event [44].

Tau-Leaping The tau-leaping method, first proposed by Gillespie [44], aims to trade off the accuracy of the stochastic simulation algorithm for computational

speed. The key gain in speed comes from using the Poisson approximation to pick a suitably large enough time step, τ, for which to fire multiple reaction events at once. A leap is deemed acceptable if the time step, τ, satisfies the requirement that the system dynamics during the interval $[t, t+\tau)$ suffer no appreciable change [20, 43, 128]. This is usually termed the *leap condition*.

Suppose τ is a suitably sized leap for a biochemical reaction system in the state $\boldsymbol{x}(t) = \boldsymbol{x}$ with propensity vector $\boldsymbol{h}(\boldsymbol{x}, \boldsymbol{\theta})$. The tau-leaping state update is

$$\boldsymbol{x}(t+\tau) = \boldsymbol{x} + \sum_{j=1}^{m} \boldsymbol{v}_j \boldsymbol{Y}_j, \tag{2.3.4}$$

where $\boldsymbol{Y}_j$ are Poisson random numbers with parameters $h_j(\boldsymbol{x}, \boldsymbol{\theta}) \cdot \tau$. If it happens that $h_j(\boldsymbol{x}, \boldsymbol{\theta}) \cdot \tau \gg 1$, then it can be shown that Equation (2.3.4) is analogous to the Euler approximation method of the chemical Langevin equation [43].

The leap condition can be chosen via numerous procedures, which either return a fixed-sized [43] or adaptively-sized [20] time jump, τ. The original method deemed the leap condition satisfied if the expected change in the propensity functions, $h_j(\boldsymbol{x}, \boldsymbol{\theta})$, were bounded by some $\epsilon h_0(\boldsymbol{x}, \boldsymbol{\theta})$. Here, $\epsilon \in (0, 1)$ is called the error control parameter. Later on, it was shown in [46] that the largest value of τ at each time step to satisfy this requirement may be estimated using the following $m^2 + 2m$ auxiliary variables:

$$f_{jj'}(\boldsymbol{x}) \equiv \sum_{i=1}^{n} \frac{\partial h_j(\boldsymbol{x}, \boldsymbol{\theta})}{\partial x_i} v_{ij'}, \qquad j, j' \in \{1, \ldots, m\}, \tag{2.3.5}$$

$$\mu_j(\boldsymbol{x}) \equiv \sum_{i=1}^{n} f_{jj'}(\boldsymbol{x}) h_{j'}(\boldsymbol{x}, \boldsymbol{\theta}), \qquad j \in \{1, \ldots, m\} \tag{2.3.6}$$

$$\sigma_j^2(\boldsymbol{x}) \equiv \sum_{i=1}^{n} f_{jj'}^2(\boldsymbol{x}) h_{j'}(\boldsymbol{x}, \boldsymbol{\theta}), \qquad j \in \{1, \ldots, m\} \tag{2.3.7}$$

where the final leap time is then calculated by [46]:

$$\tau = \min_{j \in [1,m]} \left\{ \frac{\epsilon h_0(\boldsymbol{x}, \boldsymbol{\theta})}{|\mu_j(\boldsymbol{x})|}, \frac{(\epsilon h_0(\boldsymbol{x}, \boldsymbol{\theta}))^2}{\sigma_j^2(\boldsymbol{x})} \right\}. \tag{2.3.8}$$

The downfall of this method is that it has the potential to reach non-physical, negative states. This is particularly problematic if the system contains rapid rates of reactions, or low molecular copy numbers. In [20], the authors devised

the following extensions for the tau-leaping method.

Non-negative Poisson tau-leaping [20] Primarily, the modification proposed by Cao *et al.* is based on partitioning the reaction channels into two categories: the critical, and the non-critical reactions. Within the proposed framework, any reaction is deemed critical if it is $n_c \approx 10$ firing events away from exhausting any of its constituent reactants [20]. The number of possible firings of a reaction, L_j, is calculated by

$$L_j = \min_{i\in[1,n];v_{ij}<0} \left\lfloor \frac{x_i}{|v_{ij}^-|} \right\rfloor . \tag{2.3.9}$$

If a reaction is critical, it is prevented from firing multiple times during a given leap — and negative populations can subsequently be avoided. As $n_c \to 0$, the method becomes identical to the original method [20, 43]. As n_c increases, the method instead approaches the accuracy (and complexity) of the SSA. The proposed leap, τ', is calculated from Equation (2.3.8) using only the set of non-critical reactions. When there are no critical reactions, $\tau' = \infty$. Should τ' be less than some factor of the total reaction rate $1/h_0(\boldsymbol{x}, \boldsymbol{\theta})$, tau-leaping is postponed n_d steps in favour of the SSA. Otherwise, a second leap candidate τ'' is computed from sampling an exponential distribution with a mean based on the critical reactions only. The leap τ is taken to be $\min\{\tau', \tau''\}$, and the reaction firings are updated according to the following rules [20]:

- if $\tau' < \tau''$, set $\tau = \tau'$. Generate k_j firings of each non-critical reaction $\mathcal{R}_j$ using a Poisson random variable of mean $h_j(\boldsymbol{x}, \boldsymbol{\theta})\tau$.
- if $\tau'' \leq \tau'$, set $\tau = \tau''$. Generate k_j firings of each non-critical reaction $\mathcal{R}_j$ using a Poisson random variable of mean $h_j(\boldsymbol{x}, \boldsymbol{\theta})\tau$. Then, sample one critical reaction that will fire once according to the propensities of the critical reactions only $h_j(\boldsymbol{x}, \boldsymbol{\theta})/h_0^c(\boldsymbol{x}, \boldsymbol{\theta})$.
- if any population is negative after the proposed leap, take $\tau' = \tau'/2$, and repeat the above steps.

Bounding the Propensity Changes As mentioned previously, the tau-leaping approximation is more accurate when the deviation within each propensity func-

tion is bounded by a small fraction of the total propensity, *i.e.*,

$$|\Delta_\tau h_j(\boldsymbol{x}, \boldsymbol{\theta})| \leq \epsilon h_0(\boldsymbol{x}, \boldsymbol{\theta}), \tag{2.3.10}$$

where $\Delta_\tau h_j(\boldsymbol{x}, \boldsymbol{\theta}) = h_j(\boldsymbol{x}(t+\tau), \boldsymbol{\theta}) - h_j(\boldsymbol{x}(t), \boldsymbol{\theta})$. However, this boundary condition is not uniform across all propensity functions, especially for propensity functions $h_j(\boldsymbol{x}, \boldsymbol{\theta}) \ll h_0(\boldsymbol{x}, \boldsymbol{\theta})$. In such cases, the relative change of $h_j(\boldsymbol{x}, \boldsymbol{\theta})$ that is tolerable by the boundary condition can be too large [20, 46]. If instead, the reaction channels were bounded proportionally, *e.g.*:

$$|\Delta_\tau h_j(\boldsymbol{x}, \boldsymbol{\theta})| \leq \epsilon h_j(\boldsymbol{x}, \boldsymbol{\theta}), \tag{2.3.11}$$

then problems occur when the reaction rate of the channel $\mathcal{R}_j$ tends toward zero. [20, 46] note that as reactions are discrete, there is a minimum value by which the propensity can change. Thus, the authors modify the bounds to

$$\Delta_\tau h_j(\boldsymbol{x}, \boldsymbol{\theta}) \leq \max\{\epsilon h_j(\boldsymbol{x}, \boldsymbol{\theta}), \theta_j\}, \tag{2.3.12}$$

where θ_j is the rate constant. The new τ updating formula is given

$$\tau = \min_{j \in [1,m]} \left\{ \frac{\max\{\epsilon h_j(\boldsymbol{x}, \boldsymbol{\theta}), \theta_j\}}{|\mu_j(\boldsymbol{x})|}, \frac{(\max\{\epsilon h_j(\boldsymbol{x}, \boldsymbol{\theta}), \theta_j\})^2}{\sigma_j^2(\boldsymbol{x})} \right\}. \tag{2.3.13}$$

In practice, this gives more accurate results but is also more computationally intensive. As a result, [20] developed the following alternative approximate population-based condition:

$$\Delta_\tau x_i \leq \max\{\epsilon_i x_i, 1\}, \quad \forall i \in I_{\text{RS}}, \tag{2.3.14}$$

where I_{RS} is the set of indices of reactant species used in calculation of the corresponding propensity functions. The values ϵ_i are calculated using the auxiliary variables $g_i(x_i)$:

$$\epsilon_i = \frac{\epsilon}{g_i(x_i)}, \quad \forall i \in I_{\text{RS}}. \tag{2.3.15}$$

Each $g_i(x_i)$ is defined by considering the highest order of reaction (HOR) in which the species S_i occurs.

- if HOR(i)=1, $g_i = 1$,
- if HOR(i)=2, $g_i = 2$, unless any second-order reactions require two molecules of S_i to participate. In the latter case, compute:

$$g_i = \left(2 + \frac{1}{x_i - 1}\right).$$

- if HOR(i)=3, $g_i = 3$, unless any second-order reactions require two or more molecules of S_i to participate. In the case of 2 molecules of S_i, compute:

$$g_i = \frac{3}{2}\left(2 + \frac{1}{x_i - 1}\right).$$

In the case of 3 molecules of S_i, compute:

$$g_i = \left(3 + \frac{1}{x_i - 1} + \frac{2}{x_i - 2}\right).$$

The computation of the auxiliary variables [20, 46], used in computing the leap τ, can now be modified to be taken over the set of reactant species:

$$\hat{\mu}_i(\boldsymbol{x}) = \sum_{j \in J_{\text{NCR}}} v_{ij} h_j(\boldsymbol{x}, \boldsymbol{\theta}), \qquad \forall i \in I_{\text{RS}}, \tag{2.3.16}$$

$$\hat{\sigma}_i^2(\boldsymbol{x}) = \sum_{j \in J_{\text{NCR}}} v_{ij}^2 h_j(\boldsymbol{x}, \boldsymbol{\theta}), \qquad \forall i \in I_{\text{RS}}. \tag{2.3.17}$$

The final leap computation is given by:

$$\tau = \min_{i \in I_{\text{NCR}}} \left\{ \frac{\max\{\epsilon_i x_i / g_i, 1\}}{|\hat{\mu}_i(\boldsymbol{x})|}, \frac{(\max\{\epsilon_i x_i / g_i, 1\})^2}{\hat{\sigma}_i^2(\boldsymbol{x})} \right\}. \tag{2.3.18}$$

The computational complexity grows linearly within the optimised tau-leaping method, with the number of species, whereas the old method grew quadratically with the number of reaction channels [20]. Therefore, large gains in performance can be expected for systems with many species and reaction channels.

2.4 Complete-Data Likelihood for the SSA

To approach the problem of parameter inference for stochastic biochemical reaction networks, it is often useful to obtain a likelihood function for continuous sample paths [128]. With Gillespie's stochastic simulation algorithm in mind, this section aims to derive the complete-data likelihood for trajectories obtained by the SSA. It is useful to consider the case where the given data is 'perfect' — *i.e.*, we know the state of model over a finite interval $[0, T]$ at all times, and the entire sample path for each species is known [128]. The state of the system at any given time may be written as $X(t) = (X_1(t), \ldots, X_n(t))^\intercal$, while the observed sample path is given by $\boldsymbol{x} = \{X(t) : t \in [0, T]\}$. Given the data is complete, then the time, and type, of each reaction event within the system that takes place

is known. That is, the number of reactions r_j of type $\mathcal{R}_j$ that occurred in the path $\boldsymbol{x}$, and the total number of reactions in a given interval $[0, T]$, defined as $r = \sum_{j=1}^{m} r_j$ [128]. The time and type of each reaction event can be represented by (t_l, j), where $l = 1, \ldots, r$, and t_l are assumed to be in increasing order where $t_0 = 0$ and $t_{l+1} = T$. Considering the propensity functions, the joint likelihood of an event can be written:

$$\begin{aligned} h_0(\boldsymbol{x}(t_{l-1}), \boldsymbol{\theta}) \exp\{h_0(\boldsymbol{x}(t_{l-1}), \boldsymbol{\theta})[t_l - t_{l-1}]\} \times \frac{h_j(\boldsymbol{x}(t_{l-1}), \theta_j)}{h_0(\boldsymbol{x}(t_{l-1}), \boldsymbol{\theta})} \\ = \exp\{-h_0(\boldsymbol{x}(t_{l-1}), \boldsymbol{\theta})[t_l - t_{l-1}]\} h_j(\boldsymbol{x}(t_{l-1}), \theta_j), \end{aligned} \tag{2.4.1}$$

where $h_j(\boldsymbol{x}(t_{l-1}), \theta_j)/h_0(\boldsymbol{x}(t_{l-1}), \boldsymbol{\theta})$ is the probability of $\mathcal{R}_j$ firing in $\boldsymbol{x}(t_{l-1})$. The full likelihood is the product of the above terms together with the probability that there is no event in the final interval $(t_r, T]$, which given by [128]

$$\exp\{-h_0(\boldsymbol{x}(t_l), \boldsymbol{\theta})[T - t_l]\} = 1 - \text{Prob(any reaction firing)}. \tag{2.4.2}$$

Thus the full combined likelihood is

$$\begin{aligned} L(\boldsymbol{\theta}; \boldsymbol{x}) &= \left\{ \prod_{l=1}^{r} h_j(\boldsymbol{x}(t_{l-1}), \theta_j) \exp\{-h_0(\boldsymbol{x}(t_{l-1}), \boldsymbol{\theta})[t_l - t_{l-1}]\} \right\} \\ &\qquad \times \exp\{-h_0(\boldsymbol{x}(t_l), \boldsymbol{\theta})[T - t_l]\} \\ &= \left\{ \prod_{l=1}^{r} h_j(\boldsymbol{x}(t_{l-1}), \theta_j) \right\} \left\{ \prod_{l=1}^{r+1} \exp\{-h_0(\boldsymbol{x}(t_{l-1}), \boldsymbol{\theta})[t_l - t_{l-1}]\} \right\} \\ &= \left\{ \prod_{l=1}^{r} h_j(\boldsymbol{x}(t_{l-1}), \theta_j) \right\} \exp\left\{ \sum_{l=1}^{r+1} -h_0(\boldsymbol{x}(t_{l-1}), \boldsymbol{\theta})[t_l - t_{l-1}] \right\} \\ &= \left\{ \prod_{l=1}^{r} h_j(\boldsymbol{x}(t_{l-1}), \theta_j) \right\} \exp\left\{ -\sum_{l=0}^{r} h_0(\boldsymbol{x}(t_l), \boldsymbol{\theta})[t_{l+1} - t_l] \right\}. \end{aligned} \tag{2.4.3}$$

This is known as the complete-data likelihood. The separable nature allows Equation (2.4.3) to be rewritten as Equation (2.4.4).

Proposition 2.1 (Complete-date likelihood for a stochastic kinetic model [128]). *The complete-data likelihood for a stochastic kinetic model over the time interval [0,T] takes the form*

$$L(\boldsymbol{\theta}; \boldsymbol{x}) = \pi(\boldsymbol{x}|\boldsymbol{\theta}) = \left\{ \prod_{l=1}^{r} h_j(\boldsymbol{x}(t_{l-1}), \theta_j) \right\} \exp\left\{ -\int_0^T h_0(\boldsymbol{x}(t), \boldsymbol{\theta}) dt \right\}. \tag{2.4.4}$$

This form is typical of all Markov jump processes where the first (product)

term contains all the information of the transitions, and the second (integral) term holds all the information in the period in which no jumps occur [128]. Equations (2.4.3) and (2.4.4) hold in the most general case for any form of propensity function. For simple models, mass-action kinetic rate laws of the form $h_j(\boldsymbol{x}, \theta_j) = \theta_j g_j(\boldsymbol{x})$ are often assumed. Using substitution within Equation (2.4.4) derives the convenient form

$$\begin{aligned} L(\boldsymbol{\theta}; \boldsymbol{x}) &= \left\{ \prod_{l=1}^{r} \theta_j g_j(\boldsymbol{x}(t_{l-1})) \right\} \exp\left\{ -\int_0^T \sum_{j=1}^{m} \theta_j g_j(\boldsymbol{x}(t)) dt \right\} \\ &\propto \left\{ \prod_{j=1}^{m} \theta_j^{r_j} \right\} \exp\left\{ -\sum_{j=1}^{m} \int_0^T \theta_j g_j(\boldsymbol{x}(t)) dt \right\} \\ &= \prod_{j=1}^{m} L_j(\theta_j; \boldsymbol{x}), \end{aligned} \tag{2.4.5}$$

where the component likelihoods are defined by

$$L_j(\theta_j; \boldsymbol{x}) = \theta_j^{r_j} \exp\left\{ -\theta_j \int_0^T g_j(\boldsymbol{x}(t)) dt \right\}. \tag{2.4.6}$$

Importantly, the factorisation means that each rate constant is independent of the information regarding other rate constants. As a result, separate inference may be carried out for each parameter, and their corresponding likelihoods can be optimised simultaneously or independently. By partially differentiating Equation (2.4.6) with respect to each θ_j and equating to zero [128], the maximum likelihood estimate of θ_j is

$$\hat{\theta}_j = \frac{r_j}{\int_0^T g_j(\boldsymbol{x}(t)) dt}. \tag{2.4.7}$$

Chapter 3

Stochastic Parameter Inference using the Cross-Entropy Method

3.1 Introduction

The Cross-Entropy (CE) method was first proposed by Rubinstein as an approach to address the problems of rare event probability estimation, with applications ranging from queueing models, to the analysis of network reliability, and that of telecommunication systems [105]. Since then, the method has been extended and applied to other domains [107], including the realm of optimisation [106]. Primarily, the CE method can be classed as a variance minimisation approach for finding the parameters of an 'optimal' importance sampling distribution. In the context of stochastic chemical kinetics, only previous work by Daigle *et al.* has combined a stochastic Monte Carlo Expectation-Maximisation (EM) algorithm [62] with a modified CE method (MCEM^2) [26].

In this chapter I propose a novel approach for the inference of stochastic chemical reaction network parameters. Specifically, I will present a new algorithm — Stochastic Parameter Inference using the Cross-Entropy (SPICE) — which combines the stochastic simulation approach (*i.e.*, with Gillespie's SSA or tau-leaping) with the cross-entropy method for optimisation. The proposed method aims to provide precise parameter estimates, while discarding the computational burden that arises from the exactness of the traditional EM algorithm.

3.2 Cross-Entropy Method for Optimisation

One goal of global optimisation is to find the value γ^* for which the performance of a chosen objective function, $J(\boldsymbol{x})$, is maximum over a given set $\boldsymbol{x} \in \S$. Given that (i) the solution exists, and (ii) the solution is unique, the maximum is denoted

$$J(\boldsymbol{x}^*) = \gamma^* = \max_{\boldsymbol{x} \in \S} J(\boldsymbol{x}), \tag{3.2.1}$$

where $\boldsymbol{x}^*$ is the input at which the maximum occurs. To solve Equation (3.2.1), the cross-entropy method formulates and solves an auxiliary, associated stochastic problem (ASP) [29].

Definition 3.1 (Associated Stochastic Problem [29]). *The associated stochastic problem aims to estimate the expectation, ℓ, for varying choices of γ, such that*

$$\ell(\gamma) = \mathbb{P}_{\boldsymbol{u}}(J(\boldsymbol{x}) \geq \gamma) = \mathbb{E}_{\boldsymbol{u}}\left[I_{\{J(\boldsymbol{x}) \geq \gamma\}}\right] = \int I_{\{J(\boldsymbol{x}) \geq \gamma\}} f(\boldsymbol{x}; \boldsymbol{u}) d\boldsymbol{x}, \tag{3.2.2}$$

where $\mathbb{P}_{\boldsymbol{u}}$ and $\mathbb{E}_{\boldsymbol{u}}$ represent the probability measure and expectation operator with respect to the density $f(\boldsymbol{x}; \boldsymbol{u})$ for $\boldsymbol{u} \in \mathcal{V}$, and $I_{\{J(\boldsymbol{x}) \geq \gamma\}}$ denotes the indicator function, i.e.*,*

$$I_{\{J(\boldsymbol{x}) \geq \gamma\}} = \begin{cases} 1, & \text{if } J(\boldsymbol{x}) \geq \gamma, \\ 0, & \text{otherwise.} \end{cases} \tag{3.2.3}$$

Solving the ASP in Definition 3.1 requires (i) the calculation of the threshold levels γ, and (ii) the estimation of the unknown parameter $\boldsymbol{u}$. Supposing both (i) and (ii) are possible, it is reasonable to assume that if $\gamma \approx \gamma^*$ for a parametrisation, $\boldsymbol{v}^* \in \mathcal{V}$, then the associated density, $f(\boldsymbol{x}; \boldsymbol{v}^*)$, will place the majority of its probability mass over $\boldsymbol{x}^*$.

The CE method is a multi-levelled approach for constructing a sequence of distribution parameters, $\{\hat{\gamma}_n, \hat{\boldsymbol{v}}_n : n \in \{1, \ldots, N\}\}$, that aims to converge to the 'optimal' parameters, γ^* and $\boldsymbol{v}^*$. In practice, γ_t is computed by taking the $(\rho-1)$-th quantile of sample performances $J(\boldsymbol{x})$ under $\boldsymbol{v}_{n-1}$. Choosing the reference parameter $\boldsymbol{v}_n$ is equated to finding the parameters that effectively minimise the difference between two distributions based on the Kullback-Leibler divergence (3.2.4) [76].

Definition 3.2 (Kullback-Leibler Divergence [76]). *The Kullback-Leibler Divergence between two distributions g and h is*

$$\begin{aligned}\mathcal{D}(g,h) &= \mathbb{E}_g\left[\ln\frac{g(\boldsymbol{x})}{h(\boldsymbol{x})}\right] \\ &= \int g(\boldsymbol{x})\ln g(\boldsymbol{x})d\boldsymbol{x} - \int g(\boldsymbol{x})\ln h(\boldsymbol{x})d\boldsymbol{x},\end{aligned} \tag{3.2.4}$$

where $\boldsymbol{x}$ is a random variable (or vector).

Remark 3.1 (Distance Measures). *It should be noted that Definition 3.2 is not a distance measure in a formal sense, since in general it is not symmetric, specifically $\mathcal{D}(g,h) \neq \mathcal{D}(h,g)$. However, it is useful to think of Definition 3.2 as a distance since*

$$\mathcal{D}(g,h) \geq 0,$$

with equality if and only if $g = h$ almost everywhere.

More specifically, the CE method aims to minimise the Kullback-Leibler divergence in Equation (3.2.4) between an optimal sampling density, $g^*(\boldsymbol{x})$, and a candidate sampling density $h(\boldsymbol{x})$. For most general purposes, the form of the optimal sampling density is assumed to be derived from importance sampling, *i.e.*, $g^*(\boldsymbol{x}) = I_{\{J(\boldsymbol{x})\geq\gamma\}}f(\boldsymbol{x})/\ell$ [108]. Such a distribution has the property that it places the majority of its probability mass over the mean value of the real sampling distribution. Furthermore, it is convenient to restrict the problem to examine cases where the candidate density h belongs to the parametric family of densities $\{f(\boldsymbol{x};\boldsymbol{v}), \boldsymbol{v}\in\mathcal{V}\}$ that contain the nominal density $f(\boldsymbol{x};\boldsymbol{u})$. This allows the minimisation of Equation (3.2.4) to be restructured into a maximisation problem.

Proposition 3.1 (Minimisation to Maximisation [108]). *Minimising the Kullback-Leibler divergence is equivalent to maximising the expectation*

$$\min_{\boldsymbol{v}} \mathcal{D}(g^*, f(\boldsymbol{x};\boldsymbol{v})) = \max_{\boldsymbol{v}} \mathbb{E}_{\boldsymbol{u}}\left[I_{\{J(\boldsymbol{x})\geq\gamma\}}\ln f(\boldsymbol{x};\boldsymbol{v})\right] \tag{3.2.5}$$

Proof. Following from Definition 3.2, the minimisation problem is given by

$$\min_{\boldsymbol{v}} \mathcal{D}(g^*, f(\boldsymbol{x};\boldsymbol{v})) = \min_{\boldsymbol{v}} \int g^*(\boldsymbol{x})\ln g^*(\boldsymbol{x})d\boldsymbol{x} - \int g^*(\boldsymbol{x})\ln f(\boldsymbol{x};\boldsymbol{v})d\boldsymbol{x}.$$

Substituting for the optimal sampling density, $g^*(\boldsymbol{x}) = I_{\{J(\boldsymbol{x})\geq\gamma\}} f(\boldsymbol{x};\boldsymbol{u})/\ell$, and dropping the constant ℓ yields

$$\min_{\boldsymbol{v}} \int I_{\{J(\boldsymbol{x})\geq\gamma\}} f(\boldsymbol{x};\boldsymbol{u}) \ln I_{\{J(\boldsymbol{x})\geq\gamma\}} f(\boldsymbol{x};\boldsymbol{u}) d\boldsymbol{x} - \int I_{\{J(\boldsymbol{x})\geq\gamma\}} f(\boldsymbol{x};\boldsymbol{u}) \ln f(\boldsymbol{x};\boldsymbol{v}) d\boldsymbol{x}.$$

Noting that the first term drops dependence on $\boldsymbol{v}$, the problem is only minimised when the second term is maximised [108]. Therefore,

$$\begin{aligned}\min_{\boldsymbol{v}} \mathcal{D}(g^*, f(\boldsymbol{x};\boldsymbol{v})) &= \max_{\boldsymbol{v}} \int I_{\{J(\boldsymbol{x})\geq\gamma\}} f(\boldsymbol{x};\boldsymbol{u}) \ln f(\boldsymbol{x};\boldsymbol{v}) d\boldsymbol{x} \\ &= \max_{\boldsymbol{v}} \mathbb{E}_{\boldsymbol{u}} \left[I_{\{J(\boldsymbol{x})\geq\gamma\}} \ln f(\boldsymbol{x};\boldsymbol{v}) \right].\end{aligned}$$

□

In practice, maximising Equation (3.2.5) is solved using Proposition 3.2.

Proposition 3.2 (General Cross-Entropy Solution [108, 109]). *Given the assumption that the form of $\mathcal{D}$ is often convex and differentiable with respect to $\boldsymbol{v}$ [109], the optimal CE reference parameter for $\boldsymbol{v}$, $\boldsymbol{v}^*$, can be found by solving*

$$\mathbb{E}_{\boldsymbol{u}} \left[I_{\{J(\boldsymbol{x})\geq\gamma\}} \nabla_{\boldsymbol{v}} \ln f(\boldsymbol{x};\boldsymbol{v}) \right] = 0 \tag{3.2.6}$$

when the differentiation ($\nabla_{\boldsymbol{v}}$) and expectation ($\mathbb{E}_{\boldsymbol{u}}$) operators can be interchanged.

In practice, solving Proposition 3.2 is infeasible, so it normal to instead consider the stochastic counterpart solution.

Remark 3.2 (General Monte Carlo Solution [108]). *The equivalent Monte Carlo sampling based counterpart of Equation (3.2.6) is given by*

$$\frac{1}{N} \sum_{k=1}^{N} \left[I_{\{J(\boldsymbol{x}_k)\geq\gamma\}} \nabla_{\boldsymbol{v}} \ln f(\boldsymbol{x}_k;\boldsymbol{v}) \right] = 0. \tag{3.2.7}$$

Remark 3.3 (Maximum Likelihood Estimate). *The updating rule for the optimal reference parameter often coincides with the maximum likelihood estimate of the system. This can be seen by taking the limit $\gamma \to \gamma*$ (or $\gamma \to 0$ for minimisation).*

Remark 3.4 (Minimising the Objective Function). *If instead the performance function is to be minimised, the form of the indicator function is changed to $I_{\{J(x_k)\leq\gamma\}}$.*

Altogether, the construction of the ASP (Definition 3.1), joint with the solution from Proposition 3.2, motivates the stochastic Cross-Entropy approach for optimisation [29, 75, 105] — which may be succinctly described by the following iterative 2-step procedure:

1. **Updating of γ_n**: Generate K samples from the distribution parametrised by a fixed $\boldsymbol{v}_{n-1}$. Rank the samples in order of their performances, *i.e.*, $J(\boldsymbol{x}_1) \leq \cdots \leq J(\boldsymbol{x}_K)$. For a small fixed ρ, *e.g.*, $\rho = 10^{-2}$, let $\hat{\gamma}_n$ be defined as the ρth quantile of $J(\boldsymbol{x})$, *i.e.*,

$$\hat{\gamma}_n = J(\boldsymbol{x}_{\lceil \rho K \rceil}). \tag{3.2.8}$$

2. **Updating of $\boldsymbol{v}_n$**: Using the estimated level $\hat{\gamma}_n$, use the same K samples taken from the candidate sampling distribution, $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_K$, to derive $\hat{\boldsymbol{v}}_n$ from the solution of Equation (3.2.7):

$$\frac{1}{K} \sum_{k=1}^{K} \left[I_{\{J(\boldsymbol{x}_k) \geq \gamma\}} \nabla_{\boldsymbol{v}} \ln f(\boldsymbol{x}_k; \boldsymbol{v}) \right] = 0.$$

Figure 3.1 gives an intuitive visualisation toward the Cross-Entropy approach — which aims to construct the sequence of distribution parameters $\{\gamma_n, \boldsymbol{v}_n\}$ that converges to an optimal (approximately point-mass) sampling distribution above the mean $\boldsymbol{x}^*$. Algorithm 2 outlines the simple CE procedure for the case of general optimisation.

Algorithm 2 Cross-Entropy Optimisation

1: Select $\hat{\boldsymbol{v}}_0$.
2: $n \leftarrow 0$ (iteration counter).
3: **repeat**
4: $\quad n \leftarrow n + 1$
5: $\quad$ Generate K samples, $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_K\}$, from $f(\boldsymbol{x}; \hat{\boldsymbol{v}}_{n-1})$.
6: $\quad$ Compute sample performances $J(\boldsymbol{x}_k), \quad \forall k \in \{1, \ldots, K\}$.
7: $\quad$ Compute $\hat{\gamma}_n$ from Equation (3.2.8).
8: $\quad$ Use the same sample, $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_K\}$, to solve Equation (3.2.7) for $\boldsymbol{v}_n$.
9: **until** $\hat{\gamma}_n = \hat{\gamma}_{n-1} = \hat{\gamma}_{n-2} = \ldots$ for a fixed number (*e.g.*, 3) iterations.

One key advantage of the CE method (Algorithm 2) is that a simple analytic solution to Equation (3.2.7) may often be obtained [108]. In particular, should the

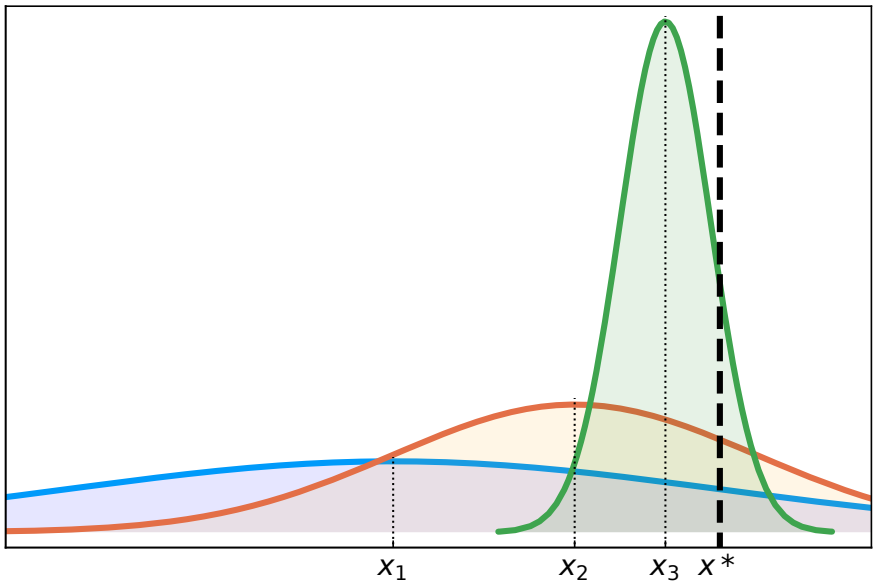


Figure 3.1: Cross-Entropy optimisation: visualising the convergence of candidate sampling distributions toward the 'optimal' sampling distribution. The most optimal sampling distribution, with mean $x*$, is a dirac δ function.

random variable x belong to a one-dimensional exponential family, the solution is always analytic.

Proposition 3.3 (Analytic Solution for the Exponential Family [108]). *The solution for the optimal reference parameter of the one-dimensional exponential family is analytic, and given by:*

$$v^* = \frac{\mathbb{E}_u[I_{\{J(x)\geq\gamma\}}x]}{\mathbb{E}_u[I_{\{J(x)\geq\gamma\}}]}. \tag{3.2.9}$$

Proof. A proof is given in [108]. □

Remark 3.5 (Exponential family Monte Carlo estimator). *In general computing the continuous case in Equation (3.2.9) is infeasible, so it is useful to instead compute the corresponding Monte Carlo sampling based estimator, $\hat{v}$, of v^* is given by*

$$\hat{v} = \frac{\sum_{k=1}^{N} I_{\{J(x_k)\geq\gamma\}}x_k}{\sum_{k=1}^{N} I_{\{J(x_k)\geq\gamma\}}}. \tag{3.2.10}$$

3.3 The Cross-Entropy Method for Exact Stochastic Chemical Kinetics

In this section I apply the Cross-Entropy method for optimisation presented in Section 3.2 to the inference of parameters within stochastic chemical reaction network models.

Recall from Chapter 2, Equation (2.4.6), that the components of the complete-data likelihood for a probabilistic kinetic model, with states $\boldsymbol{x}(t)$, derived from Gillespie's stochastic simulation algorithm can be written in the following form:

$$L_j(\theta_j; \boldsymbol{x}) = \theta_j^{r_j} \exp\left\{-\theta_j \int_0^T g_j(\boldsymbol{x}(t))dt\right\},$$

where θ_j is the stochastic rate constant for the reaction $\mathcal{R}_j$, r_j is the total number firing events of reaction $\mathcal{R}_j$ over the time interval $[0, T]$, and g_j is the propensity function. For this section, I will denote the k-th complete trajectory by $\boldsymbol{z}_k$, such that

$$\boldsymbol{z}_k = (\boldsymbol{x}_0^{(k)}, \ldots, \boldsymbol{x}_r^{(k)})$$

where $r = \sum_{j=1}^{M} r_j$ is the number of states (or reaction events) within that trajectory, and the shorthand $\boldsymbol{x}_i = \boldsymbol{x}(t_i)$ is used to represent the i-th discrete-time state. It is often useful to instead consider discrete-time dynamics — see Remark 3.6.

Remark 3.6 (Discrete-time Component Likelihood). *In the discrete-time case, we can rewrite Equation (2.4.6) as*

$$L_j(\theta_j; \boldsymbol{z}) = \theta_j^{r_j} \exp\left\{-\theta_j \sum_{i=1}^{r+1} g_j(\boldsymbol{x}_{i-1})\tau_i\right\}, \tag{3.3.1}$$

where $\boldsymbol{x}_i = \boldsymbol{x}(t_i)$, τ_i is the time between the ith and $(i-1)$th reaction, and τ_{r+1} is the final time interval at the end of the simulation in which no reaction occurs.

Following the Cross-Entropy method in Section 3.2, the aim of this section is to find the parameters of the optimal sampling distribution of stochastic chemical kinetic models. Explicitly, the aim is to solve the optimisation program via Equation (3.2.6) to derive the optimal kinetic rate parameters, $\hat{\boldsymbol{\theta}} \approx \boldsymbol{\theta}^*$, which are used to generate samples from the stochastic simulation algorithm. Conveniently, it can be seen that the complete-data likelihood (Equation (2.4.4)) of a trajectory

sampled from the chemical master equation using the SSA belongs to the exponential family, and an analytic solution therefore exists — see Proposition 3.3. The exact solution for the updating formula of a kinetic rate constant, $\hat{\theta}_j$ is given below by Proposition 3.4 [26].

Proposition 3.4 (Exact Cross-Entropy Solution for Kinetic Rate Constants). *The exact solution of Equation (3.2.6) for a stochastic kinetic rate constant, θ_j, belonging to a Gillespie-type model is given by the updating formula:*

$$\hat{\theta}_j = \frac{\mathbb{E}_{\boldsymbol{\theta}*}\left[r_j^{(k)} I_{\{J(\boldsymbol{z}_k)\geq\gamma\}}\right]}{\mathbb{E}_{\boldsymbol{\theta}*}\left[I_{\{J(\boldsymbol{z}_k)\geq\gamma\}}\left(\sum_{i=1}^{r^{(k)}+1} g_j(\boldsymbol{x}_{i-1}^{(k)})\tau_i\right)\right]}. \tag{3.3.2}$$

Proof. The proof for Proposition 3.4 was not included in [26], so I have derived, and provided it here. To prove Proposition 3.4, first consider the component likelihoods, $L_j(\theta_j;\boldsymbol{z})$, given by Equation (3.3.1). Substituting into Equation (3.2.6) yields

$$\mathbb{E}_{\boldsymbol{\theta}*}\left[I_{\{J(\boldsymbol{z})\geq\gamma\}}\nabla_{\theta_j}\ln L_j(\theta_j;\boldsymbol{z})\right] = 0.$$

Expanding $L_j(\theta_j;\boldsymbol{z})$ obtains the following formula:

$$\mathbb{E}_{\boldsymbol{\theta}*}\left[I_{\{J(\boldsymbol{z})\geq\gamma\}}\nabla_{\theta_j}\ln\theta_j^{r_j}\exp\left\{-\theta_j\sum_{i=1}^{r+1} g_j(\boldsymbol{x}_{i-1})\tau_i\right\}\right] = 0.$$

Performing the derivative yields

$$\mathbb{E}_{\boldsymbol{\theta}*}\left[I_{\{J(\boldsymbol{z})\geq\gamma\}}\left(\frac{r_j}{\theta_j} - \sum_{i=1}^{r+1} g_j(\boldsymbol{x}_{i-1})\tau_i\right)\right] = 0,$$

which in turn can be rearranged to obtain

$$\mathbb{E}_{\boldsymbol{\theta}*}\left[I_{\{J(\boldsymbol{z})\geq\gamma\}}\frac{r_j}{\theta_j}\right] = \mathbb{E}_{\boldsymbol{\theta}*}\left[\sum_{i=1}^{r+1} g_j(\boldsymbol{x}_{i-1})\tau_i\right].$$

Observing that θ_j is a constant, non-random variable, it can be taken outside of the expectation operator. Rearranging thus yields the solution:

$$\hat{\theta}_j = \frac{\mathbb{E}_{\boldsymbol{\theta}*}\left[r_j^{(k)} I_{\{J(\boldsymbol{z}_k)\geq\gamma\}}\right]}{\mathbb{E}_{\boldsymbol{\theta}*}\left[I_{\{J(\boldsymbol{z}_k)\geq\gamma\}}\left(\sum_{i=1}^{r^{(k)}+1} g_j(\boldsymbol{x}_{i-1}^{(k)})\tau_i\right)\right]}.$$

□

Remark 3.7 (Monte Carlo Solution for Kinetic Rate Parameters). *Com-*

puting the continuous case as in Equation (3.3.2) is infeasible. Instead, $\hat{\theta}_j$ can be obtained from the stochastic counterpart solution [26]:

$$\hat{\theta}_j = \frac{\sum_{k=1}^{N} r_j^{(k)} I_{\{J(\boldsymbol{z}_k)\geq\gamma\}}}{\sum_{k=1}^{N} I_{\{J(\boldsymbol{z}_k)\geq\gamma\}} \left(\sum_{i=1}^{r^{(k)}+1} g_j(\boldsymbol{x}_{i-1}^{(k)})\tau_i\right)}. \tag{3.3.3}$$

Remark 3.8 (Maximum Likelihood Estimator). *The maximum likelihood estimator for a stochastic rate parameter derived from a Gillespie-type model is given by Equation (2.4.7). The Monte Carlo counterpart is as follows*

$$\hat{\theta}_j = \frac{\sum_{k=1}^{N} r_j^{(k)}}{\sum_{k=1}^{N} \left(\sum_{i=1}^{r^{(k)}+1} g_j(\boldsymbol{x}_{i-1}^{(k)})\tau_i\right)}. \tag{3.3.4}$$

It can be seen that Equation (3.3.4) approximates Equation (3.3.3) in the limit $\gamma \to \gamma*$.

3.4 The Cross-Entropy Solution for Inexact Stochastic Chemical Kinetics

The majority of methods seen in parameter estimation and inference rely on optimising an objective cost function, (e.g. mean squared difference between the data and simulation). In this class of methods, nearly *all* either involve simulation from (deterministic) ODE systems, or require stochastic simulations of the chemical master equation using Gillespie's direct method. While ODE systems are very straightforward and fast to simulate, they often fail to capture realistic dynamics of biological systems for which stochasticity plays an important role in the observed behaviour. However, direct sampling from the CME using Gillespie's SSA is computationally intensive. Recent years have seen a new generation of inexact methods, such as tau-leaping [20, 43]. By switching to an inexact method such as tau-leaping, a degree of accuracy is traded off in favour of computational performance. Interestingly, while the use of tau-leaping methods for robust parameter estimation is not novel, they have not been utilised (to my knowledge) in the context of computing maximum likelihood estimates by solving stochastic programs such as Equation (3.2.6) derived by the Cross-Entropy method. Naturally, it is in our interest to explore how viable such an approach could be for

systems with different kinds of dynamics.

In this section, I derive the Cross-Entropy solution for the optimal reference parameters for inexact tau-leaping models, and propose its use for parameter inference.

Suppose as usual we are interested in simulating a system with n species, $\{S_i : i \in \{1, \ldots, n\}\}$, and m reaction channels, $\{\mathcal{R}_j : j \in \{1, \ldots, m\}\}$, using a tau-leaping approximation until a final time T. The complete-data likelihood of a trajectory generated using tau-leaping dynamics is given below by Proposition 3.5.

Proposition 3.5 (Complete-data Likelihood for a Tau-leaping Trajectory [128]). *The complete-data likelihood of a tau-leaping trajectory $\boldsymbol{z}$ is given by*

$$\mathcal{L}(\boldsymbol{\theta}; \boldsymbol{z}) = \prod_{i=0}^{r} \prod_{j=1}^{m} \frac{\exp\{-\lambda_{ji}\}(\lambda_{ji})^{r_{ji}}}{r_{ji}!}, \tag{3.4.1}$$

where $r \in \mathbb{Z}^+$ is the number of discrete-time intervals taken, $\lambda_{ji} = \theta_j g_j(\boldsymbol{x}_i)\tau_i$ is the mean of the Poisson process for reaction $\mathcal{R}_j$ over the interval time $[t_i, t_i + \tau_i)$, and r_{ji} the number of firings of reaction $\mathcal{R}_j$.

Proof. A proof is provided in [128]. □

Remark 3.9. *We can again conveniently factorise the complete-data likelihood in Proposition 3.5 into component likelihoods associated with each reaction channel:*

$$\mathcal{L}(\boldsymbol{\theta}; \boldsymbol{z}) = \prod_{j=1}^{m} \mathcal{L}_j(\theta_j; \boldsymbol{z}), \tag{3.4.2}$$

where each component $\mathcal{L}_j(\theta_j; \boldsymbol{z})$ is given by:

$$\mathcal{L}_j(\theta_j; \boldsymbol{z}) = \prod_{i=0}^{r} \frac{\exp\{-\lambda_{ji}\}(\lambda_{ji})^{r_{ji}}}{r_{ji}!}. \tag{3.4.3}$$

Using Proposition 3.5, it is then possible to compute analytically the optimal kinetic rate constants for a tau-leaping trajectory, obtained from solving the Cross-Entropy optimisation program given by Equation (3.2.6).

Proposition 3.6 (Inexact Cross-Entropy Solution for Kinetic Rate Constants). *The inexact solution of Equation (3.2.6) for a stochastic kinetic rate*

constant, θ_j, belonging to a tau-leaping model is given by the updating formula:

$$\hat{\theta}_j = \frac{\mathbb{E}_{\boldsymbol{\theta}*}\left[r_j^{(k)} I_{\{J(\boldsymbol{z}_k)\geq\gamma\}}\right]}{\mathbb{E}_{\boldsymbol{\theta}*}\left[I_{\{J(\boldsymbol{z}_k)\geq\gamma\}}\left(\sum_{i=1}^{r^{(k)}+1} g_j(\boldsymbol{x}_{i-1}^{(k)})\tau_i\right)\right]}. \tag{3.4.4}$$

Proof. Using Equation (3.2.6) together with Remark 3.9, Equation (3.4.3), obtains the following equation:

$$\mathbb{E}_{\boldsymbol{\theta}*}\left[I_{\{J(\boldsymbol{z})\geq\gamma\}}\nabla_{\theta_j}\ln\prod_{i=0}^{r}\frac{\exp\{-\lambda_{ji}\}(\lambda_{ji})^{r_{ji}}}{r_{ji}!}\right] = 0.$$

Expanding the Poisson process mean, $\lambda_{ji} = \theta_j g_j(\boldsymbol{x}_i)\tau_i$, yields

$$\mathbb{E}_{\boldsymbol{\theta}*}\left[I_{\{J(\boldsymbol{z})\geq\gamma\}}\nabla_{\theta_j}\ln\prod_{i=0}^{r}\frac{\exp\{-\theta_j g_j(\boldsymbol{x}_i)\tau_i\}(\theta_j g_j(\boldsymbol{x}_i)\tau_i)^{r_{ji}}}{r_{ji}!}\right] = 0.$$

By rearranging, the exponential product can be factorised out, thus giving

$$\mathbb{E}_{\boldsymbol{\theta}*}\left[I_{\{J(\boldsymbol{z})\geq\gamma\}}\nabla_{\theta_j}\ln\exp\left\{-\theta_j\sum_{i=0}^{r}g_j(\boldsymbol{x}_i)\tau_i\right\}\prod_{i=0}^{r}\frac{(\theta_j g_j(\boldsymbol{x}_i)\tau_i)^{r_{ji}}}{r_{ji}!}\right] = 0.$$

Noting that $\prod_{i=0}^{r}\theta_j^{r_{ji}} = \theta_j^{r_j}$, where $r_j = \sum_{i=0}^{r} r_{ji}$, gives

$$\mathbb{E}_{\boldsymbol{\theta}*}\left[I_{\{J(\boldsymbol{z})\geq\gamma\}}\nabla_{\theta_j}\ln\theta_j^{r_j}\exp\left\{-\theta_j\sum_{i=0}^{r}g_j(\boldsymbol{x}_i)\tau_i\right\}\prod_{i=0}^{r}\frac{(g_j(\boldsymbol{x}_i)\tau_i)^{r_{ji}}}{r_{ji}!}\right] = 0.$$

Taking the differential with respect to θ_j obtains:

$$\mathbb{E}_{\boldsymbol{\theta}*}\left[I_{\{J(\boldsymbol{z})\geq\gamma\}}\left(\frac{r_j}{\theta_j} - \sum_{i=1}^{r+1}g_j(\boldsymbol{x}_{i-1})\tau_i\right)\right] = 0.$$

Following the proof of Proposition 3.4, the above equation can be separated to obtain

$$\mathbb{E}_{\boldsymbol{\theta}*}\left[I_{\{J(\boldsymbol{z})\geq\gamma\}}\frac{r_j}{\theta_j}\right] = \mathbb{E}_{\boldsymbol{\theta}*}\left[\sum_{i=1}^{r+1}g_j(\boldsymbol{x}_{i-1})\tau_i\right],$$

which can be rearranged to obtain the solution

$$\hat{\theta}_j = \frac{\mathbb{E}_{\boldsymbol{\theta}*}\left[r_j^{(k)} I_{\{J(\boldsymbol{z}_k)\geq\gamma\}}\right]}{\mathbb{E}_{\boldsymbol{\theta}*}\left[I_{\{J(\boldsymbol{z}_k)\geq\gamma\}}\left(\sum_{i=1}^{r^{(k)}+1} g_j(\boldsymbol{x}_{i-1}^{(k)})\tau_i\right)\right]}.$$

□

Remark 3.10 (The Exact and Inexact Solutions). *The only difference between Equation (3.3.2) (the exact solution) and Equation (3.4.4) (the inexact*

solution) is that the propensities g_j and time intervals τ_i are calculated after every reaction firing within the former, while they are calculated after every leap within the latter. Within the exact SSA, a 'reaction event' is defined by the firing of a single reaction, while in tau-leaping, a reaction event consists of firing multiple reactions simultaneously according to the Poisson distribution.

3.5 Variance of the Cross-Entropy Parameter Estimates

In this section, I outline how estimates for the variance of the Cross-Entropy parameter estimates may be obtained. In the books [103, 121], a general formula is provided for deriving the covariance matrix, $\hat{\boldsymbol{\Sigma}}$, of a Monte Carlo maximum-likelihood estimate, $\hat{\boldsymbol{\theta}}$ (*e.g.*, obtained via the Expectation-Maximisation algorithm).

Proposition 3.7 (Covariance Matrix of the Maximum-Likelihood Estimate [103, 121]). *The inverse covariance matrix of a maximum-likelihood estimate θ is given by*

$$\hat{\Sigma}^{-1} = \Bigg[-\frac{1}{K}\sum_{k}^{K}\frac{\partial^2}{\partial\theta^2} - \frac{1}{K}\sum_{k}^{K}\frac{\partial}{\partial\theta}\cdot\frac{\partial}{\partial\theta}^{T} + \frac{1}{K^2}\left(\sum_{k}^{K}\frac{\partial}{\partial\theta}\right)\cdot\left(\sum_{k}^{K}\frac{\partial}{\partial\theta}\right)^{T}\Bigg](\ln f(\theta;x)) \quad (3.5.1)$$

where the operator $\frac{\partial^2}{\partial\theta^2}$ returns a $m\times m$ matrix, $\frac{\partial}{\partial\theta}$ returns an m-dimensional vector ($m\times 1$ matrix), and $\frac{\partial}{\partial\theta}^{T}$ denotes matrix transpose. K represents the number of Monte Carlo samples and $f(\theta;x)$ is the underlying probability distribution.

By substituting the expression for the complete-data likelihood in Equation (2.4.3), $\mathcal{L}(\boldsymbol{\theta};\boldsymbol{z})$, an expression for the covariance matrix of the maximum-likelihood estimates for stochastic kinetic rate parameters may be derived [26]. To ensure that the confidence bounds lie within positive state-space, Daigle *et al.* take a log-transformed parameter ω, such that $\hat{\theta} = \exp(\hat{\omega})$.

Proposition 3.8 (Covariance Matrix of the Kinetic Rate Parameters [26]). *The inverse covariance matrix of the kinetic rate parameter estimates is*

given by

$$
\begin{aligned}
-\hat{\Sigma}^{-1} = & \left\{ \frac{1}{k'} \sum_{k'=1}^{K'} \sum_{i=1}^{r^{(k')}+1} \exp(\hat{\omega}) g_j(\boldsymbol{x}_{i-1}^{(k')}) \tau_i \right\}_j \\
& + \frac{1}{k'} \sum_{k'=1}^{K'} \left(r_j^{(k')} - \sum_{i=1}^{r^{(k')}+1} \exp(\hat{\omega}) g_j(\boldsymbol{x}_{i-1}^{(k')}) \tau_i \right)_j \\
& \times \left(r_j^{(k')} - \sum_{i=1}^{r^{(k')}+1} \exp(\hat{\omega}) g_j(\boldsymbol{x}_{i-1}^{(k')}) \tau_i \right)_j^{\intercal} \\
& - \left(\frac{1}{k'} \sum_{k'=1}^{K'} \left[r_j^{(k')} - \sum_{i=1}^{r^{(k')}+1} \exp(\hat{\omega}) g_j(\boldsymbol{x}_{i-1}^{(k')}) \tau_i \right] \right)_j \\
& \times \left(\frac{1}{k'} \sum_{k'=1}^{K'} \left[r_j^{(k')} - \sum_{i=1}^{r^{(k')}+1} \exp(\hat{\omega}) g_j(\boldsymbol{x}_{i-1}^{(k')}) \tau_i \right] \right)_j^{\intercal}
\end{aligned}
\tag{3.5.2}
$$

where Monte Carlo samples k' (up to K') are taken from only those trajectories that satisfy $J(\boldsymbol{z}_k) \geq \gamma$. $\{\cdot\}_j$ is a diagonal matrix with elements corresponding to each kinetic rate constant, and $(\cdot)_j$ is a column vector. r_j and g_j are as before — representing the number of reaction firing events, and the propensity function, within the time interval τ_i.

In practice, computing Equation (3.5.2) can be computationally heavy, however, it can be seen that the formula makes use of information already calculated for the derivation of the optimal parameters in Equation (3.3.3). Thus, samples can be reused for both calculations, optimising the process [26]. That said, computing Equation (3.5.2) still relies on taking many Monte Carlo samples, and in reality can suffer from numerical issues. Specifically, with too few samples the retrieved covariance matrix is not always positive-definite. A simpler, numerically stable approximation of the variance can instead be derived straight from the Cross-Entropy method [75, 108].

Proposition 3.9 (Variance of the Optimal Cross-Entropy Parameters). *The variance, $\hat{\Sigma}_{jj}$, of the j-th optimal Cross-Entropy parameter, θ_j, can be esti-*

mated via

$$\hat{\Sigma}_{jj} = \frac{1}{K'} \sum_{k'=1}^{K'} \left(\frac{r_j^{(k')}}{\sum_{i=1}^{r^{(k')+1}} g_j(\boldsymbol{x}_{i-1}^{(k')}) \tau_i} - \hat{\theta}_j \right)^2 . \tag{3.5.3}$$

Together, Equation (3.5.2) and Equation (3.5.3) can be used to compute the variance of the estimates for stochastic kinetic rate parameters derived from the Cross-Entropy method. These can be used to approximate a confidence interval or bounds on the estimates. Furthermore, the Cross-Entropy method can be extended not to only update the mean of a candidate sampling distribution, *i.e.*, $\boldsymbol{\theta}$, but to update the variance of the sampling distribution $\boldsymbol{\Sigma}$. By estimating the variance of the candidate sampling density, the Cross-Entropy method can often better explore the parameter state-space. This is because if a parameter is poorly estimated, or not well constrained, the variance of the candidate sampling distribution is allowed to increase, thus exploring a larger parameter state-space. Conversely, if the obtained parameter estimates are precise, the candidate sampling density evolves toward a narrow probability mass over the optimal solution, $\boldsymbol{\theta}^*$, — see Figure 3.1.

3.6 SPICE

In this section, I present the *Stochastic Rate Parameter Inference with Cross-Entropy* (SPICE) algorithm [101]. SPICE is an approach that combines stochastic simulation — either using Gillespie's direct method or tau-leaping — with the cross-entropy method for optimisation. As highlighted in Section 3.1, only previous work by Daigle *et al.* [26] has combined a stochastic Expectation–Maximisation algorithm [62] with a modified cross-entropy method (MCEM^2) for parameter inference. In this work, I instead develop the cross-entropy method in its own right, discarding the costly EM algorithm steps in favour of other heuristics. The material presented in this section contributed to [101].

Algorithm Overview The SPICE algorithm is an iterative process that focuses efforts on generating a sequence of distribution parameters $\{(\gamma_n, \boldsymbol{\theta}^{(n)}, \boldsymbol{\Sigma}^{(n)})\}$ — with the aim of converging to the optimal kinetic rate constants $\hat{\boldsymbol{\theta}}^*$ as follows:

1. Generate K_n sample trajectories, $\{\boldsymbol{z}_k : k \in \{1, \ldots, K_n\}\}$, using the SSA with kinetic rate parameters $\boldsymbol{\theta}_k^{(n-1)}$ sampled from the lognormal distribution. For the initial iteration, SPICE samples the parameter vector $\boldsymbol{\theta}$ within the given bounds $[\boldsymbol{\theta}_{\text{MIN}}^{(0)}, \boldsymbol{\theta}_{\text{MAX}}^{(0)}]$ by using a quasi-random Sobol low-discrepancy sequence [119] to ensure adequate coverage. See below for more information.

2. Using the data, *rank* and *sort* the trajectories $\{\boldsymbol{z}_k\}$ in order of their performances $\{J(\boldsymbol{z}_k)\}$. See Eqs. (3.6.2) and (3.6.1) for the actual definition of the performance, or score, functions adopted.

3. For a fixed number of 'elite' samples, *e.g.*, $K_{\text{elite}} = 10$, let $\rho_n = K_{\text{elite}}/K_n$.

4. Let $\hat{\gamma}_n$ be defined as the ρ_n-th quantile of $\{J(\boldsymbol{z}_k)\}$, *i.e.*, $\hat{\gamma}_n = J(\boldsymbol{z}_{\lceil \rho K_n \rceil})$.

5. Using the estimated level $\hat{\gamma}_n$, use the same K_n sample trajectories $\{\boldsymbol{z}_k\}$ to derive $\hat{\boldsymbol{\theta}}^{(n)}$ and $\hat{\boldsymbol{\Sigma}}^{(n)}$ from the solutions of Eqs. (3.3.3) and (3.5.3). The trajectories that satisfy $J(\boldsymbol{z}) \leq \gamma$ are termed the 'elite' samples.

6. Repeat until termination criterion is reached. A reasonable termination criterion to take would be to stop if $\hat{\gamma}_n \not\leq \hat{\gamma}_{n-1} \not\leq \ldots$ for a fixed number of iterations (*e.g.*, 3). In general, as the parameter estimates approach their optimum values, increasing numbers of samples are required to distinguish between their relative performances.

Remark 3.11 (Parameter State-Space Bounds). *The motivation for placing bounds on the parameter state-space stems from several reasons. The first is that global optimisation is a difficult problem. Exhaustively searching too large a state-space will be (i) computationally intensive, and (ii) more prone to finding local minima. Often it is also the case that researchers modelling a system will have an intuition of what order magnitudes the parameters should be. It should also be noted that SPICE does not require bounds to work, and is able to start from a fixed point — exploring the state-space by adaptively changing the parameters of the candidate sampling distribution. This feature can be useful in local optimisation.*

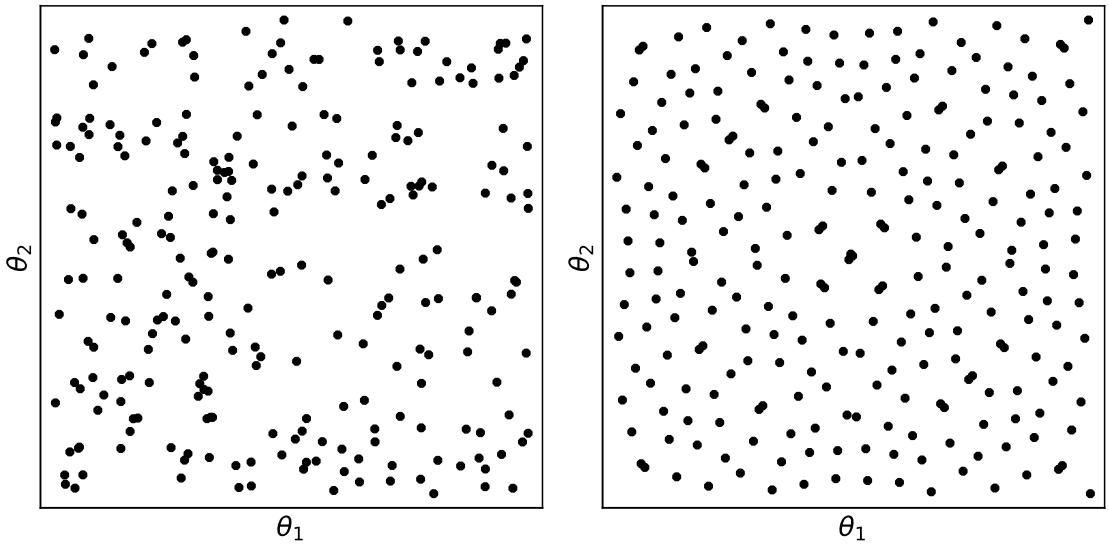
Algorithm 3 presents a simplified pseudocode detailing the above approach of SPICE toward parameter inference.

Algorithm 3 SPICE — Stochastic Parameter Inference using the Cross-Entropy Method (Generic Approach)

Inputs: Dataset represented by $\boldsymbol{y}_i$ at times t_i for $0 \leqslant i \leqslant d$, initial parameter bounds $\left[\boldsymbol{\theta}^{(0)}_{\text{MIN}}, \boldsymbol{\theta}^{(0)}_{\text{MAX}}\right]$, minimum samples K_{min}, number of 'elite' samples K_{elite}.

Output: Estimate of the parameters $\hat{\boldsymbol{\theta}}^{(n)}$, and their variances $\hat{\boldsymbol{\Sigma}}^{(n)}$.

$\gamma_0 \leftarrow \infty$ // cost function value
$n \leftarrow 1$ // iteration
$K_1 \leftarrow K_{\text{min}}$ // initial sample size
$S \leftarrow \left[\hat{\boldsymbol{\theta}}^{(0)}_{\text{MIN}}, \hat{\boldsymbol{\theta}}^{(0)}_{\text{MAX}}\right]$ // generate Sobol sequence hypercube
repeat
  for $k = 1 \rightarrow K_n$ **do**
    if $n = 1$ **then**
      $\boldsymbol{\theta} \sim S(k)$ // sample the Sobol sequence
    else
      $\boldsymbol{\theta} \sim \mathcal{N}\left(\hat{\boldsymbol{\theta}}^{(n-1)}, \hat{\boldsymbol{\Sigma}}^{(n-1)}\right)$ // sample the (log–)normal dist.
    for $i = 1 \rightarrow d$ **do** // time intervals in the dataset
      if $i = 1$ **then**
        $\boldsymbol{z}_k \leftarrow \text{SSA}(\boldsymbol{y}_0, 0, t_1, \boldsymbol{\theta})$ // forward simulate with the SSA
      else
        $\boldsymbol{z}_k \leftarrow \text{SSA}(\boldsymbol{z}_k, t_{i-1}, t_i, \boldsymbol{\theta})$
    $j_k \leftarrow J(\boldsymbol{z}_k)$ // compute the objective function, *e.g.*, Equation (3.6.1)
  Sort $\{\boldsymbol{z}_k\}$ according to performances $\{j_k\}$
  $\rho_n \leftarrow K_{\text{elite}}/K_n$
  $\gamma_n \leftarrow \rho_n$-th quantile of $\{j_k\}$
  if $\gamma_n < \gamma_{n-1}$ **then**
    $\hat{\boldsymbol{\theta}}^{(n)} \leftarrow$ solution to Equation (3.3.3), using $\{\boldsymbol{z}_k\}$, and γ_n
    $\hat{\boldsymbol{\Sigma}}^{(n)} \leftarrow$ solution to Equation (3.5.3), using $\{\boldsymbol{z}_k\}$, and γ_n
    $n \leftarrow n + 1$
  else
    $K_n \leftarrow \min(K_n + K_n/3, K_{\text{max}})$ // adaptive sampling
until convergence of $\{\gamma_n, \gamma_{n-1}, \gamma_{n-2}\}$



(a) Uniform Sampling (b) Sobol Sequence

Figure 3.2: A comparison between 2-dimensional sampling using (a) uniform sampling, and (b) Sobol low discrepancy sequencing. Sobol sequencing provides a better state-space coverage, while uniform sampling leads to under (and over–) sampled regions.

It can be seen that Algorithm 3 is an extension of the generic Cross-Entropy approach, presented in Algorithm 2, with adaptations made for stochastic biochemical kinetics. In addition, SPICE incorporates various additional heuristics to aid in the inference process. Each of these is detailed below.

Sobol Low Discrepancy Sequences In order to make parameter inference routines more robust against local minima, it is essential to efficiently sample the entirety of the desired parameter space, *i.e.*, the bounded region $[\boldsymbol{\theta}^{(0)}_{\text{MIN}}, \boldsymbol{\theta}^{(0)}_{\text{MAX}}]$. A simple naive approach would be to generate uniform samples within the specified interval, *i.e.*, $\boldsymbol{\theta}^{(1)} \sim U[\boldsymbol{\theta}^{(0)}_{\text{MIN}}, \boldsymbol{\theta}^{(0)}_{\text{MAX}}]$, and use these parameter vectors as guesses in the initial iteration. However, for a finite number of samples this leads to an uneven coverage of the parameter state-space, with under (and over–) sampled regions — see Figure 3.2a. The Sobol sequence [119] is a quasi-random low-discrepancy sequence used for generating more favourable samples from a uniform (multi-dimensional) interval — see Figure 3.2b. The Sobol sequence is

closely related to Latin hypercube sampling [97] and can be shown to improve the convergence of Monte Carlo methods.

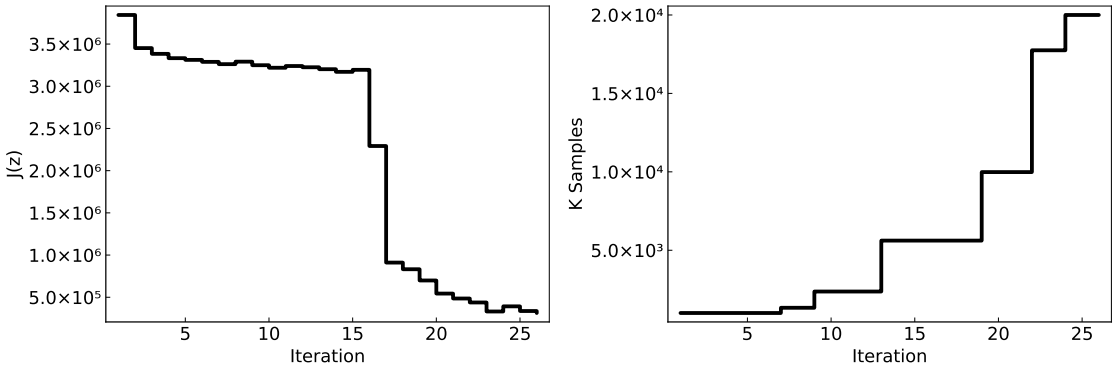
Remark 3.12 (Log-Sobol Sampling). *If logarithmic sampling is being utilised, SPICE first generates the low-discrepancy Sobol sequence within the n-dimensional unit hypercube* $[0, 1]^n$*, and secondly applies the appropriate scaling factors and log-transformations to match the input parameter search space.*

Adaptive Sampling Adaptive sampling aims to update the number of samples K_n taken within each iteration. The motivation is to ensure that parameter estimates improve with statistical significance as the rate of convergence (of the cost function) stagnates. Adaptive sampling has previously been applied to the Monte Carlo expectation-maximisation algorithm — in what was dubbed the 'ascent-based MCEM' [19]. The original ascent-based method was computationally heavy, requiring the calculation of the system likelihood. However, the same concepts can apply in principle to any likelihood-free cost function. The outline of adaptive sampling is as follows:

1. Suppose we wish to update our parameters based on a fixed number of elite samples, K_{elite}, satisfying $J(\boldsymbol{z}_k) \leq \gamma$. The performance of the 'best' elite sample is denoted γ_n^*, and the performance of the 'worst' elite sample is equivalent to the ρ_n-th quantile of $\{J(\boldsymbol{z}_k)\}$, $\hat{\gamma}_n$.

2. The quantile parameter ρ_n is adaptively updated each iteration as $\rho_n = K_{\text{elite}}/K_n$, where K_{elite} is typically taken to be $1 - 10\%$ of the starting number of samples $K_{\min}$.

3. At each iteration, a check is made to see if there is improvement in either the best, or the worst performing elite samples, *i.e.*, if, $\hat{\gamma}_n^* < \hat{\gamma}_{n-1}^*$ or if $\hat{\gamma}_n < \hat{\gamma}_{n-1}$. If there is an improvement then the samples are used to update the parameters according to Equation (3.3.3). Otherwise, the number of samples is increased by a fixed amount, *i.e.*, $K_n/3$, and the iteration is repeated. If the number of samples hits a predetermined maximum, $K_{\max}$, for 3 successive iterations without improvement in the performance function, then the algorithm terminates — suggesting that no further significant improvement can be made given the number of samples.



(a) Cost function (b) Number of samples

Figure 3.3: An illustration of adaptive sampling at work. As the cost function plateaus across iterations to a local minimum, the number of samples taken at each iteration is adaptively updated to improve the accuracy of the estimates — concentrating computation time near the solution.

Figure 3.3 shows adaptive sampling in action for a real run of SPICE. It can be seen that as the cost function in Figure 3.3a converges to a minimum, the number of samples taken within each iteration within Figure 3.3b increases. Thus, a secondary benefit of the adaptive sampling method is that it optimises the placement of computation time across the iterations. Early iterations can find rapid improvement in the parameters and therefore require less samples, while in latter iterations it becomes increasingly hard to distinguish significant improvements of the estimated parameters. SPICE is thus able to make fast evaluations early on, and focus its computational efforts near the optimal solution.

Objective Function The SPICE algorithm has been developed to handle an arbitrary number of datasets. Given N time series datasets (with d time points), SPICE can associate N objective function scores with each simulated trajectory. Each objective function score is typically taken to be the standard sum of L^2 distances across the trajectory with respect to the corresponding dataset:

$$J_n(\boldsymbol{z}) = \sum_{i=1}^{d} (\boldsymbol{y}_{n,i} - \boldsymbol{x}_i)^2 \qquad 1 \leq n \leq N \tag{3.6.1}$$

where $\boldsymbol{x}_i = \boldsymbol{x}(t_i)$ is the simulation, and $\boldsymbol{y}_{n,i}$ is the data point at time t_i in the n-th dataset. To ensure no datasets are ignored, 'elite' samples are taken with

respect to each dataset — *i.e.*, the best performing quantile of trajectories for each individual dataset (with scores $J_n(\boldsymbol{z})$).

In the absence of temporal correlation within the data (*e.g.*, when measurements between time points are independent or individual cells cannot be tracked as in flow cytometry data), it is useful to instead construct an empirical Gaussian mixture model for each time point within the data. Each mixture model at time t_i is comprised of N multivariate normal distributions, each with a vector of mean values $\boldsymbol{y}_{n,i}$ corresponding to the *observed* species in the n-th dataset, and standard deviations (a diagonal covariance matrix $\boldsymbol{\sigma}_n^2$) corresponding to an error estimate, or variance of the measurements on the species. In the absence of prior information a default value of 10% error can be taken on all the species. The objective score function can then be proportional to the negative log-likelihood of the simulated trajectory with respect to the data:

$$J(\boldsymbol{z}) \approx -\sum_{i=1}^{d} \ln\left(\sum_{n=1}^{N} \exp\left[-\frac{1}{2}(\boldsymbol{y}_{n,i} - \boldsymbol{x}_t)^{\intercal}\boldsymbol{\sigma}_n^{-2}(\boldsymbol{y}_{n,i} - \boldsymbol{x}_i)\right]\right). \tag{3.6.2}$$

Smoothed Updates SPICE implements the parameter smoothing update formulae [29, 75]:

$$\hat{\boldsymbol{\theta}}^{(n)} = \lambda\tilde{\boldsymbol{\theta}}^{(n)} + (1-\lambda)\hat{\boldsymbol{\theta}}^{(n-1)}, \tag{3.6.3}$$

$$\hat{\boldsymbol{\Sigma}}^{(n)} = \beta_n\tilde{\boldsymbol{\Sigma}}^{(n)} + (1-\beta_n)\hat{\boldsymbol{\Sigma}}^{(n-1)}, \tag{3.6.4}$$

where $\beta_n = \beta - \beta\left(1-\frac{1}{n}\right)^q$, $\lambda\in(0,1]$, $q\in\mathbb{N}^+$, and $\beta\in(0,1)$ are smoothing constants. $\tilde{\boldsymbol{\theta}}^{(n)}$ and $\tilde{\boldsymbol{\Sigma}}^{(n)}$ are the outputs from the standard solution of the cross-entropy in Equations (3.3.3) and (3.5.3). Parameter smoothing between iterations has three important benefits: (i) the parameter estimate values are smoothed out and therefore converge to a more stable value, (ii) it reduces the probability of a parameter estimate tending toward a value of zero within the first few iterations, and (iii) it prevents the distribution of parameter estimates from converging too quickly to a degenerate point probability mass at a local minima.

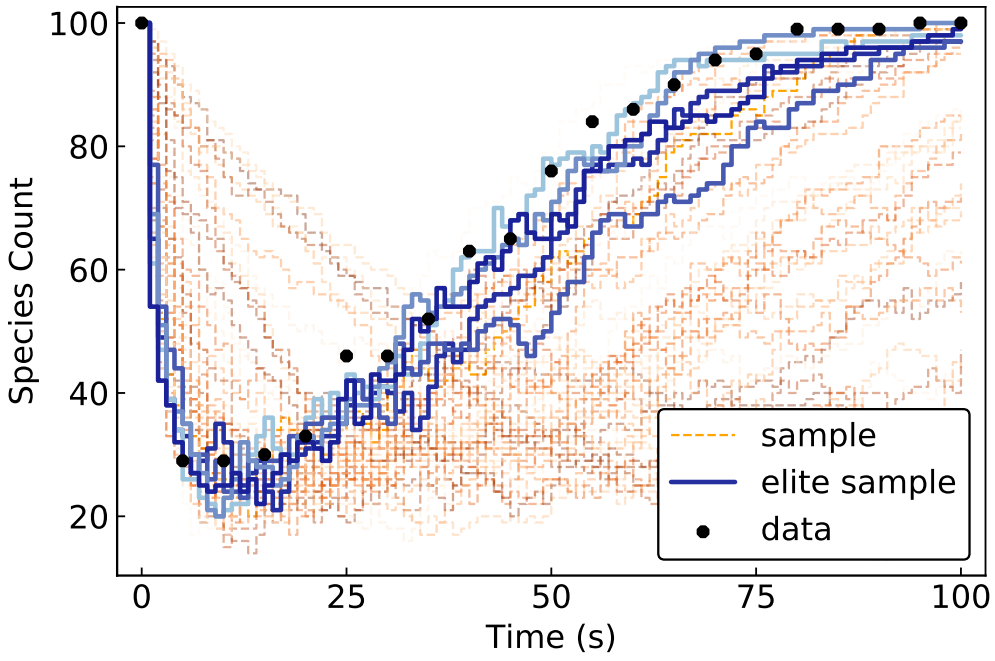
Remark 3.13. *Under certain constraints (*i.e.*, discrete optimisation) there exists a proof that the general Cross-Entropy method for parameter inference may converge to an optimal solution with probability 1, but only in the case of smoothed*

updates [24].

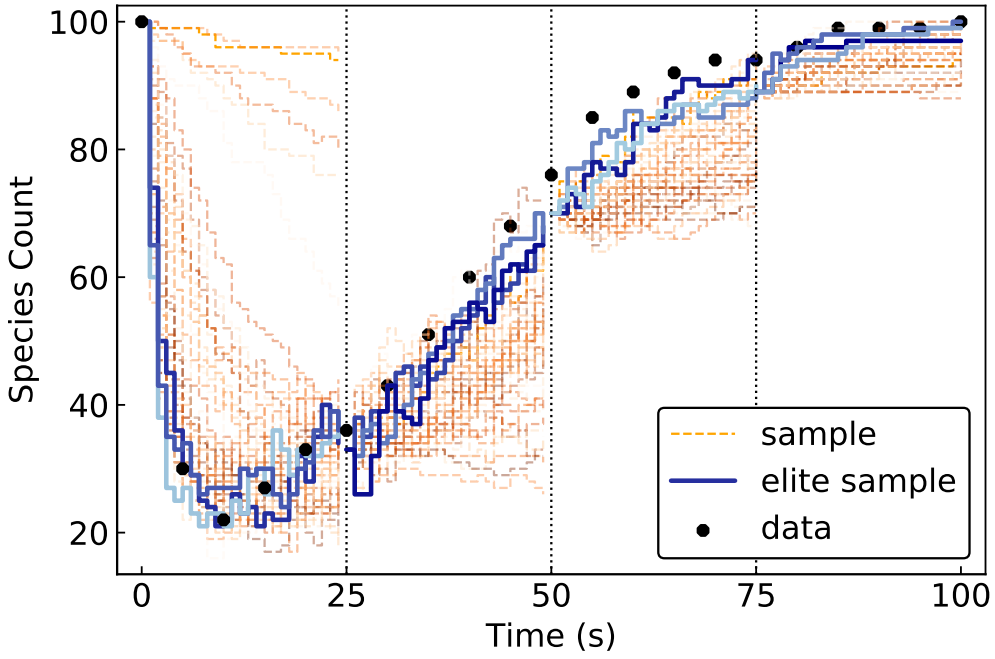
Piecewise Approaches SPICE can optionally utilise two techniques for trajectory simulation between time intervals: a piecewise '*multiple shooting*' approach that treats time-intervals in the data with a degree of independence [133]; and a '*multilevel splitting*' (also termed Particle Splitting) approach as in [26]. Multiple shooting has previously been shown to give an improvement of estimates and better stability when using ODE approximations in place of the SSA [12, 133], and is implemented in the tool COPASI [61] for both ODE and stochastic simulations.

Particle Splitting For particle splitting, I adopt a multilevel splitting approach as in [26], and the objective function in Equation (3.6.1) is instead calculated after the simulation of each piecewise segment from $\boldsymbol{x}_{i-1}$ to $\boldsymbol{x}_i$ — or after any other arbitrary division of the time interval $[0, T]$ (*e.g.*, as in Figure 3.4b). The trajectories $\boldsymbol{z}_k$ satisfying $J(\boldsymbol{z}_k) \leq \hat{\gamma}_n$ are then re-sampled with replacement K_n times before simulation continues (recalling that K_n is the number of samples in the n-th iteration. This process aims at discarding poorly performing trajectories in favour of those 'closest' to the data. This creates an enriched sample, at the cost of introducing an aspect of bias propagation. A comparison between the 'generic' SPICE approach, and the approach of SPICE using multilevel splitting approach is visualised in Figure 3.4. Previous work has also implemented the particle splitting approach in the field of rare-event probability estimation [47].

Remark 3.14 (On the similarities between SPICE and ABC-SMC). *The particle splitting method within SPICE draws many comparisons to the Approximate Bayesian Computation with Sequential Monte Carlo sampling (ABC-SMC) approaches [6, 118, 124]. Both methods draw samples from the candidate sampling distribution in a sequential fashion — and both methods sample with replacement those samples that are performing well according to the cost function, and discard those that are performing poorly. The difference is that while the Bayesian methods use this acceptance-rejection approach to build up an empirical posterior distribution, SPICE instead uses the samples to compute the underlying parameters (*i.e.*, the mean and variance) of a posterior distribution.*



(a) Generic 'elite' sampling approach.



(b) Particle splitting approach.

Figure 3.4: A visualisation of the difference between (a) the generic 'elite' sampling approach, and (b) the particle splitting 'elite' sampling approach. Within the generic approach, 'elite' trajectories are sampled after the complete time interval has been simulated. Within the multilevel splitting approach, 'elite' trajectories are replicated after a chosen number of intervals.

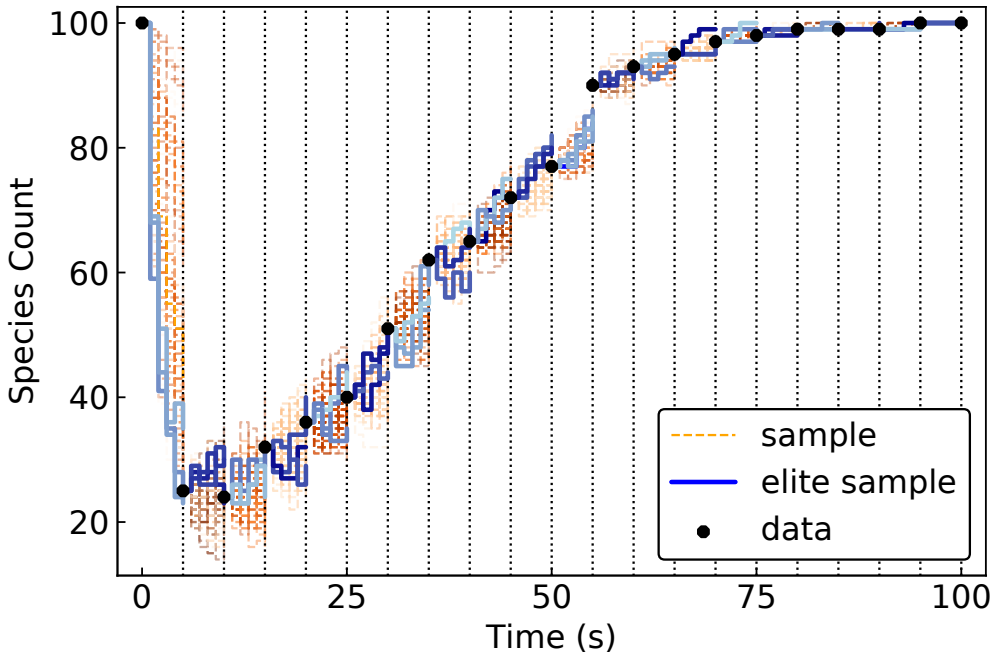


Figure 3.5: A visualisation of the multiple shooting 'elite' sampling approach. Simulation over the complete time interval is divided up according to the time points within the dataset. At the beginning of each interval, the trajectories are 'reset' to the closest data point. The 'elite' samples are those trajectories with the best performance across *all* time intervals.

Multiple Shooting For this implementation of the multiple shooting method, SPICE constructs a sample trajectory comprised of T intervals matching the time stamps within the data $\boldsymbol{y}$. In their original paper [133], the authors simulate each segment from $\boldsymbol{x}_{i-1}$ to $\boldsymbol{x}_i$ using an ODE model with the initial conditions set to the previous time point of the dataset, *i.e.*, $\boldsymbol{x}_{i-1} = \boldsymbol{y}_{i-1}$. In this work, I instead treat the data as being mixture-normally distributed, thus aim to sample initial conditions $\boldsymbol{x}_{i-1} \sim \mathcal{N}(\boldsymbol{y}_{n,i-1}, \sigma^2_{n,i-1})$, where the index of the time series n is first uniformly sampled. Using the SSA, each piecewise section of a trajectory belonging to sample k is then simulated with the same parameter vector $\boldsymbol{\theta}$. An illustration of the multiple shooting approach is presented in Figure 3.5.

Hyperparameters With additional modelling, SPICE allows for the inclusion of hyperparameters $\boldsymbol{\phi}$ — *e.g.*, scaling constants, initial concentrations, volumes,

and non kinetic-rate parameters. The hyperparameters are then sampled simultaneously (from a lognormal distribution) alongside their kinetic rate constant counterparts, $\boldsymbol{\theta}$. Parameter inference can be carried out for these hyperparameters by updating the parameters of their sampling distribution using the standard Cross-Entropy approach for optimisation as in Algorithm 2.

Tau-Leaping With inexact, faster methods such as tau-leaping [43] a degree of accuracy is traded off in favour of computational performance. Thus, for computationally intensive systems, there is interest to be had in replacing the SSA with tau-leaping within the SPICE algorithm. The updating formula for the optimal kinetic rate parameters was derived in Proposition 3.5, and is given by Equation (3.4.4).

3.6.1 Implementation & Usage Details

Overview SPICE is open source, and implemented within the Julia programming language [13]. It is self-contained, implementing all sampling, optimisation, and simulation within the Julia environment (via custom built Gillespie SSA and tau-leaping algorithms). The key inputs given to SPICE are

- the chemical reaction network model specification,
- the data to be used for optimisation,
- initial parameter bounds.

Model Specification The model can be specified to SPICE within the Julia environment by passing the core information about (i) the initial species, (ii) the propensity functions for each reaction and (iii) the system's state-change matrix. For parameter estimation, prior parameter bounds $[\boldsymbol{\theta}^{(0)}_{\text{MIN}}, \boldsymbol{\theta}^{(0)}_{\text{MAX}}]$, and the data (*e.g.*, 'data.csv') are also specified. An example of the input is given in Listing 3.1.

```
#= This is a code sample for defining the
Lotka-Volterra Model in SPICE =#
using SPICE

# initial species & state-change matrix
x0 = [50, 50]
v = [1 0; -1 1; 0 -1]

# bounds on the initial search space for parameters
bounds = [[1e-6, 1e-6, 1e-6], [10.0, 10.0, 10.0]]

# hazard functions (mass-action)
function hazard(p)
        p.a[1] = p.x[1] * p.θ[1]
        p.a[2] = p.x[1] * p.x[2] * p.θ[2]
        p.a[3] = p.x[2] * p.θ[3]
end

model = Model(x0, hazard, v, bounds)
system = System(model, "data.csv")
```

Listing 3.1: This is a code sample for defining the Lotka-Volterra Model in SPICE

The data within 'data.csv' can be comma– or tab– separated, with the independent 'Time' variable in the first column, and each model species appearing in each column thereafter.

Parameter Inference Given a fully specified chemical reaction network model, complete with the data and parameter bounds, parameter estimation can be ran as follows in Listing 3.2.

kwargs	Description	Argument values
nSamples	The initial number of samples to generate.	Int (default=1,000)
mSamples	The maximum number of samples to generate.	Int (default=20,000)
nRepeat	The number of repeat simulations to take.	Int (default=1)
nElite	The fixed number of 'elite' samples to take.	Int (default=10)
maxIter	The iteration limit.	Int (default=250)
ssa	The simulation method to use.	:Direct *or* :Tau
sampling	Type of sampling distribution to use.	:normal *or* :log
smoothing	The smoothing parameter.	Float64 (default=0.7)
shoot	Use the multiple-shooting approach?	true *or* false
splitting	Use the particle-splitting approach?	true *or* false

Table 3.1: A table of variable keyword arguments that can be passed to SPICE to customise the parameter inference routine variables.

```
# run estimation algorithm 'n' times
n = 1
estimate(system, n, "output")
```

Listing 3.2: This is a code sample for running parameter estimation on the model defined in Listing 3.1

The output file is written to the specified '*output*.csv', and contains the values of the final parameter estimates, the estimation of their variance, and the total CPU run time (in seconds). For each n-th run of the algorithm, a separate file '/traces/*output*-n-trace.csv' is also generated for additional analysis — containing the values at each sub-iteration for: the parameter estimates; the objective cost function; and the number of samples taken.

Advanced: Additional Options The SPICE algorithm can be tweaked via a number of variable options, which can be given by the (optional) 'routine = CEM(*kwargs*...)' argument — see Listing 3.3 for an example. The different arguments that can be passed are presented in Section 3.6.1.

```
#= Example Cross-Entropy method (CEM) setup with:
n -> 1000 initial samples
m -> 20000 maximum samples
SSA -> Tau-leaping =#
system = System(model, "data.csv",
    routine=CEM(nSamples=1000, mSamples=20000, ssa=:Tau))
```

Listing 3.3: This is a code example for tweaking the Cross-Entropy method variables.

Chapter 4

Moment Approximations and the Generalised Method of Moments

4.1 Introduction

In this first portion of this chapter I will derive generalised equations to describe the time evolution of moments within a stochastic chemical reaction network. As in Chapter 2, I will make the assumption that the system is both well-mixed and dilute [45]. Furthermore, I will assume all the kinetics are contained within a closed compartment of constant volume Ω. Unless explicitly stated, all given examples will assume a practical, dimensionless volume of $\Omega = 1$. In the midsection of this chapter I discuss some of the present problems regarding the validity of moment closure approximations. Later on, I will outline the application of these moment equations to the paradigm of parameter estimation, and give an overview of the generalised method of moments (GMM) for stochastic rate inference.

Within this chapter I will present two new approaches:

- Firstly, I propose a new method for generating moment equations for stochastic chemical networks based on replacing the Taylor series approximation, with Padé approximants.
- Secondly, I outline a new method that uses a novel adaptive multi-level tau-leaping approach alongside the GMM (MLGMM) to perform population moment-based inference.

4.2 Moment Equations for a Stochastic Chemical Reaction Network

Let the moments of the distribution of the chemical species of a CRN be denoted by $\mathbb{E}[\boldsymbol{x}^{\boldsymbol{k}}]$, with $\boldsymbol{x}^{\boldsymbol{k}} = \prod_{i=1}^{n} x_i^{k_i}$, and let $k = \sum_{i=1}^{n} k_i$ denote the total order of the moments. To derive the moment equations for a stochastic chemical reaction network model, it is useful to first define the *moment generating function* [102].

Definition 4.1 (Moment Generating Function [102]). *The generalised moment generating function of an n-dimensional random vector $\boldsymbol{x} = (x_1, \ldots, x_n)$ is*

$$M_{\boldsymbol{x}}(\boldsymbol{\phi}) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} e^{\boldsymbol{\phi}^{\intercal}\boldsymbol{x}} P(\boldsymbol{x}) dx_1 \cdots dx_n = \mathbb{E}\left[e^{\boldsymbol{\phi}^{\intercal}\boldsymbol{x}}\right], \tag{4.2.1}$$

where the expectation is taken with respect to the probability distribution of the system, $P(\boldsymbol{x})$, and $\boldsymbol{\phi}$ is a fixed vector.

The moments $\mathbb{E}[\boldsymbol{x}^{\boldsymbol{k}}]$, can be found by taking the k-th order derivatives of the moment generating function in Equation (4.2.1) with respect to the auxiliary variable $\boldsymbol{\phi}$, and by setting $\phi_i = 0, \forall i \in \{1, \ldots, n\}$ [2].

Example 4.1 (Raw Moments of a 1D System). *Given a one dimensional system with species x, the moment generating function is*

$$M_x(\phi) = \mathbb{E}\left[e^{\phi x}\right],$$

which can be expanded using the Taylor series to become

$$M_x(\phi) = \mathbb{E}\left[1 + \phi x + \frac{\phi^2 x^2}{2!} + \frac{\phi^3 x^3}{3!} + \cdots\right].$$

Taking the 1st, 2nd, and 3rd order derivatives with respect to ϕ, and setting $\phi = 0$ yields the moments $\mathbb{E}[x], \mathbb{E}[x^2], \mathbb{E}[x^3]$, and so forth.

With a small amount of effort, it can be seen that the moment generating function will readily yield the set of k-th order moments for multivariate systems of any finite dimension.

To derive the time dependence of the moments $\frac{d}{dt}\mathbb{E}[\boldsymbol{x}^{\boldsymbol{k}}]$, we can recall the chemical master equation in (Definition 2.9), and form the time dependent moment generating function [8].

Definition 4.2 (Time Dependent Moment Generating Function [8]). *The time dependent moment generating function is given by*

$$\frac{dM_{\boldsymbol{x}}(\boldsymbol{\phi})}{dt} = \sum_{j=1}^{m} \left[\left(e^{\boldsymbol{\phi}^{\intercal} \boldsymbol{v}_j} - 1 \right) \sum_{\boldsymbol{x}} e^{\boldsymbol{\phi}^{\intercal} \boldsymbol{x}} P(\boldsymbol{x}) h_j(\boldsymbol{x}, \boldsymbol{\theta}) \right]. \tag{4.2.2}$$

where $\boldsymbol{v}_j$ is the state change vector for reaction $\mathcal{R}_j$, with the associated propensity function $h_j(\boldsymbol{x}, \boldsymbol{\theta})$.

From Definition 4.2, it can be seen that the exact form of the moment equations $\frac{d}{dt}\mathbb{E}[\boldsymbol{x}^{\boldsymbol{k}}]$ can be found analogously to the methodology in Example 4.1, *i.e.*,

$$\frac{d}{dt}\mathbb{E}[\boldsymbol{x}^{\boldsymbol{k}}] = \frac{d}{d\boldsymbol{\phi}} \frac{dM_{\boldsymbol{x}}(\boldsymbol{\phi})}{dt} \bigg|_{\boldsymbol{\phi}=0}, \tag{4.2.3}$$

where the derivative has been taken with respect to $\phi_1, \ldots, \phi_n$, and evaluated at zero.

Proposition 4.1 (Time Evolution of the Mean [2]). *The time evolution of the mean of species S_i, $\mathbb{E}[x_i]$, is given by:*

$$\frac{d}{dt}\mathbb{E}[x_i] = \sum_{j=1}^{m} v_{j,i} \mathbb{E}[h_j(\boldsymbol{x}, \boldsymbol{\theta})] \tag{4.2.4}$$

Proof. Using Equation (4.2.3), the evolution of the mean of species S_i, is given by

$$\frac{d}{dt}\mathbb{E}[x_i] = \frac{d}{d\phi_i} \frac{dM_{\boldsymbol{x}}(\boldsymbol{\phi})}{dt} \bigg|_{\boldsymbol{\phi}=0}.$$

From Definition 4.2, the time dependent moment generating function is given by Equation (4.2.2). Taking the derivative with respect to ϕ_i yields

$$\sum_{j=1}^{m} \left[\sum_{\boldsymbol{x}} \left((v_{j,i} + x_i) e^{\boldsymbol{\phi}^{\intercal} \boldsymbol{x} + \boldsymbol{\phi}^{\intercal} \boldsymbol{v}_j} - x_i e^{\boldsymbol{\phi}^{\intercal} \boldsymbol{x}} \right) P(\boldsymbol{x}) h_j(\boldsymbol{x}, \boldsymbol{\theta}) \right] \bigg|_{\boldsymbol{\phi}=0}.$$

Simplifying the expression, and evaluating at $\boldsymbol{\phi} = 0$ gives

$$\sum_{j=1}^{m} v_{j,i} P(\boldsymbol{x}) h_j(\boldsymbol{x}, \boldsymbol{\theta}) \equiv \sum_{j=1}^{m} v_{j,i} \mathbb{E}[h_j(\boldsymbol{x}, \boldsymbol{\theta})].$$

□

Thus, the evolution of the mean becomes a function of the propensity, and the underlying state change matrix. Generalising for a multivariate system with

species vector $\boldsymbol{x}$, the following equations for the exact time evolution of the k-th order moments can be obtained [34]:

$$\frac{d}{dt}\mathbb{E}[\boldsymbol{x}^{\boldsymbol{k}}] = \sum_{j=1}^{m} \mathbb{E}\left[((\boldsymbol{x}+\boldsymbol{v}_j)^{\boldsymbol{k}} - \boldsymbol{x}^{\boldsymbol{k}}) \cdot h_j(\boldsymbol{x},\boldsymbol{\theta})\right], \tag{4.2.5}$$

where

$$(\boldsymbol{x}+\boldsymbol{v}_j)^{\boldsymbol{k}} = \prod_{i=1}^{n}(x_i + v_{j,i})^{k_i}.$$

The expectations in both Equation (4.2.4) and Equation (4.2.5) can be computed by taking a Taylor series expansion around the mean $\boldsymbol{\mu} = (\mathbb{E}[x_1], \ldots, \mathbb{E}[x_n])$ [2, 34, 42], *i.e.*,

$$\frac{d}{dt}\mathbb{E}[x_i] = \sum_{j=1}^{m} v_{j,i} \sum_{k_1=0}^{\infty} \cdots \sum_{k_n=0}^{\infty} \frac{\partial^{\boldsymbol{k}}}{\partial \boldsymbol{x}^{\boldsymbol{k}}} \frac{h_j(\boldsymbol{x},\boldsymbol{\theta})}{\boldsymbol{k}!}\bigg|_{\boldsymbol{x}=\boldsymbol{\mu}} \mathbb{E}[(\boldsymbol{x}-\boldsymbol{\mu})^{\boldsymbol{k}}], \tag{4.2.6}$$

where $\mathbb{E}[(\boldsymbol{x}-\boldsymbol{\mu})^{\boldsymbol{k}}]$ are the central moments of order k, and multi-index notation is used such that

$$\begin{aligned}
\partial^{\boldsymbol{k}} &= \partial^{\sum_{i=1}^{n} k_i}, \\
\partial \boldsymbol{x}^{\boldsymbol{k}} &= \prod_{i=1}^{n} \partial x_i^{k_i}, \\
\boldsymbol{k}! &= \prod_{i=1}^{n} k_i!, \\
\mathbb{E}[(\boldsymbol{x}-\boldsymbol{\mu})^{\boldsymbol{k}}] &= \mathbb{E}\left[\prod_{i=1}^{n}(x_i - \mu_i)^{k_i}\right].
\end{aligned}$$

The relation between the central moments, $\mathbb{E}[(\boldsymbol{x}-\boldsymbol{\mu})^{\boldsymbol{k}}]$, and raw moments, $\mathbb{E}[\boldsymbol{x}^{\boldsymbol{k}}]$, can be found by considering the binomial expansion, *i.e.*,

$$\mathbb{E}[(\boldsymbol{x}-\boldsymbol{\mu})^{\boldsymbol{k}}] = \sum_{k_1'=0}^{k_1} \cdots \sum_{k_n'=0}^{k_n} \binom{\boldsymbol{k}}{\boldsymbol{k}'} (-1)^{(\boldsymbol{k}-\boldsymbol{k}')} \boldsymbol{\mu}^{(\boldsymbol{k}-\boldsymbol{k}')} \mathbb{E}[\boldsymbol{x}^{\boldsymbol{k}}], \tag{4.2.7}$$

with the following defined

$$\begin{aligned}\binom{\boldsymbol{k}}{\boldsymbol{k}'} &= \prod_{i=1}^{n}\binom{k_i}{k_i'},\\ (-1)^{(\boldsymbol{k}-\boldsymbol{k}')} &= \prod_{i=1}^{n}(-1)^{(k_i-k_i')},\\ \boldsymbol{\mu}^{(\boldsymbol{k}-\boldsymbol{k}')} &= \prod_{i=1}^{n}\mu_i^{(k_i-k_i')}.\end{aligned}$$

Conversely, the relationship between the raw moments and central moments are given by the binomial expansion

$$\mathbb{E}[\boldsymbol{x}^{\boldsymbol{k}}] = \sum_{k_1'=0}^{k_1}\cdots\sum_{k_n'=0}^{k_n}\binom{\boldsymbol{k}}{\boldsymbol{k}'}\boldsymbol{\mu}^{(\boldsymbol{k}-\boldsymbol{k}')}\mathbb{E}[(\boldsymbol{x}-\boldsymbol{\mu})^{\boldsymbol{k}}].$$

The time evolution of the central moments are given by,

$$\frac{d}{dt}\mathbb{E}[(\boldsymbol{x}-\boldsymbol{\mu})^{\boldsymbol{k}}] = \sum_{k_1'=0}^{k_1}\cdots\sum_{k_n'=0}^{k_n}\binom{\boldsymbol{k}}{\boldsymbol{k}'}(-1)^{(\boldsymbol{k}-\boldsymbol{k}')}\frac{d}{dt}\left(\boldsymbol{\mu}^{(\boldsymbol{k}-\boldsymbol{k}')}\mathbb{E}[\boldsymbol{x}^{\boldsymbol{k}'}]\right) \tag{4.2.8}$$

where using the chain rule yields

$$\frac{d}{dt}\left(\boldsymbol{\mu}^{(\boldsymbol{k}-\boldsymbol{k}')}\mathbb{E}[\boldsymbol{x}^{\boldsymbol{k}'}]\right) = \frac{d}{dt}\left(\boldsymbol{\mu}^{(\boldsymbol{k}-\boldsymbol{k}')}\right)\mathbb{E}[\boldsymbol{x}^{\boldsymbol{k}'}] + \boldsymbol{\mu}^{(\boldsymbol{k}-\boldsymbol{k}')}\frac{d}{dt}\left(\mathbb{E}[\boldsymbol{x}^{\boldsymbol{k}'}]\right), \tag{4.2.9}$$

with

$$\frac{d}{dt}\left(\boldsymbol{\mu}^{(\boldsymbol{k}-\boldsymbol{k}')}\right) = \boldsymbol{\mu}^{(\boldsymbol{k}-\boldsymbol{k}')}\sum_{i=1}^{n}(k_i-k_i')\mu_i^{-1}\frac{d}{dt}\mathbb{E}[x_i], \tag{4.2.10}$$

and

$$\frac{d}{dt}\mathbb{E}[\boldsymbol{x}^{\boldsymbol{k}'}] = \sum_{j=1}^{m}\sum_{k_1=0}^{\infty}\cdots\sum_{k_n=0}^{\infty}\frac{\partial^{\boldsymbol{k}}}{\partial\boldsymbol{x}^{\boldsymbol{k}}}\frac{((\boldsymbol{x}+\boldsymbol{v}_j)^{\boldsymbol{k}'}-\boldsymbol{x}^{\boldsymbol{k}'})\cdot h_j(\boldsymbol{x},\boldsymbol{\theta})}{\boldsymbol{k}!}\bigg|_{\boldsymbol{x}=\boldsymbol{\mu}}\mathbb{E}[(\boldsymbol{x}-\boldsymbol{\mu})^{\boldsymbol{k}}] \tag{4.2.11}$$

Remark 4.1. *Equation (4.2.11) is different from Eq. (11) presented in [2], which is subtly incorrect as it includes an additional term of $\boldsymbol{x}^{k'}$ when expanded.*

4.2.1 Moment Closure Approximations

In general, the moment equations in Equation (4.2.6) and Equation (4.2.11) form an infinite set of coupled non-linear ODEs, due to the dependence of lower order moments on higher orders. This motivates the use of moment closure approxima-

tions (MAs), whereby the infinite set of equations obtained through Taylor series expansions are truncated above a given order, and the system is closed based on assumptions of the underlying distribution (*e.g.*, multivariate normality [42], lognormality [117], or a gamma distribution [77]).

Zero MA The '*Zero MA*' is the most simple moment closure, in which the Taylor series in Equations (4.2.6) and (4.2.11) are truncated at a given order k. Thus, all central moments above order k are zero.

Normal MA A multivariate normal distribution [102] has the density function

$$f(\boldsymbol{x}) = \frac{1}{\sqrt{(2\pi)^N |\boldsymbol{\Sigma}|}} \exp\left\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\intercal \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right\}. \tag{4.2.12}$$

Given the first and second order moments, all higher order central moments up to a chosen order k can be computed using Isserlis' theorem.

Theorem 4.1 (Isserlis' Theorem [66]). *If $\boldsymbol{x} = (x_1, \ldots, x_{2n})$ is a zero-mean multivariate normal random vector, then*

$$\mathbb{E}[(\boldsymbol{x}-\boldsymbol{\mu})^{\boldsymbol{k}}] = I(\boldsymbol{k}) \sum \prod \mathbb{E}[(x_i - \mu_i)^{k_i}(x_{i'} - \mu_{i'})^{k_{i'}}]$$

where the notation $\sum\prod$ is used to denote summation over all distinct partitions of $\boldsymbol{x}$ into pairs $\{x_i, x_{i'}\}$, and $I(\boldsymbol{k})$ is the indicator function such that

$$I(\boldsymbol{k}) = \begin{cases} 0, & \text{if } \sum_{i=1}^n k_i \text{ is odd,} \\ 1, & \text{if } \sum_{i=1}^n k_i \text{ is even.} \end{cases}$$

Remark 4.2. *By Theorem 4.1, all odd-ordered central moments are set to zero in the normal MA. The number of paired combinations of $\{x_i, x_i'\}$ required to compute the even orders is $(k-1)!!$,* i.e., *the number of terms grows by a double factorial.*

Log-Normal MA A random variable $\boldsymbol{x}$ follows a lognormal distribution if its logarithm is normally distributed, *i.e.*,

$$\boldsymbol{x} = \exp(\boldsymbol{Y}),$$

and $\boldsymbol{Y} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. All higher order moments of $\boldsymbol{x}$ up to a given order k can be expressed in terms of $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ using the following relationship ([77])

$$\mathbb{E}[\boldsymbol{x}^{\boldsymbol{k}}] = \exp(\boldsymbol{k}^{\intercal}\boldsymbol{\mu} + \frac{1}{2}\boldsymbol{k}^{\intercal}\Sigma\boldsymbol{k}). \tag{4.2.13}$$

The variables $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ may be computed via

$$\Sigma_{ij} = \ln(1 + \frac{\mathbb{E}[(x_i - \mathbb{E}[x_i])^2]}{\mathbb{E}[x_i]^2}),$$
$$\mu_i = \ln(\mathbb{E}[x_i]) - \frac{1}{2}\Sigma_{ii}.$$

4.3 On the Validity of Moment Approximations

The moment approximation methods provided within Section 4.2 can be used to provide efficient (and scalable) descriptions of stochastic chemical reaction networks. However, there is no guarantee that the derived moment approximations for a biochemical model are accurate, and no method exists that can predetermine whether a given choice of closure is valid. A simulation is deemed valid if it gives physically meaningful approximations of the CME — such that all species populations remain positive, finite, and reach a global steady state should the underlying CME have a stationary solution as $t \to \infty$. In practice, moment approximations can exhibit non-physical behaviour (*e.g.*, negative populations [115], chaos, and finite-time blowup [31]) and can even fail for simplistic low-dimensional systems — for example, the Lotka-Volterra predator-prey model [28]. In [115], the authors empirically analyse and compare the normal, Poisson [95], log-normal [117], and the zero-MA (termed *central-moment-neglect*) moment closure approximations by applying them to systems with bimodal, ultrasensitivity, and oscillatory behaviours. The authors find that in general, the normal moment approximation is most favourable due to the simulations being more stable, on average, for a larger portion of the state-space [115].

Remark 4.3. *Many research papers stipulate that, theoretically, adding higher order moments to a system will increase the accuracy of the resulting MA — at the cost of additional computational complexity, and numerical stiffness* (e.g., *[28]). However, this is only true if the model, and underlying moment approximations*

are valid. In general, adding additional moments to a poor approximation can in fact, make the resulting simulations much worse.

In the following subsection, Section 4.3.1, I provide an illustrative case study based on the Lotka-Volterra model for which the moment approximations fail.

4.3.1 Example: Collapse of the Lotka-Volterra Predator-Prey Model

Consider the standard Lotka-Volterra CRN model,

$$\begin{aligned} X &\xrightarrow{\theta_1} X + X \\ X + Y &\xrightarrow{\theta_2} Y + Y \\ Y &\xrightarrow{\theta_3} \emptyset \end{aligned}$$

with parameters $(\theta_1, \theta_2, \theta_3) = (0.5, 0.0025, 0.3)$. The system comprises two populations, the prey X, and the predators Y — whose evolutions under the standard (truncated, zero-closure) moment approximation is governed by a finite set of coupled non-linear ODEs. For a Taylor-series expansion up to 2nd order, the Lotka-Volterra model contains 5 moments [28], *i.e.*, one ODE for each of: $\mathbb{E}[X]$, $\mathbb{E}[Y]$, $\mathrm{Var}(X)$, $\mathrm{Cov}(X, Y)$, and $\mathrm{Var}(Y)$. The moment equations are as follows:

$$\begin{aligned} \frac{d\mathbb{E}[X]}{dt} &= \mathbb{E}[X]\theta_1 - \mathbb{E}[X]\mathbb{E}[Y]\theta_2 - \mathrm{Cov}(X,Y)\theta_2, \\ \frac{d\mathbb{E}[Y]}{dt} &= \mathbb{E}[X]\mathbb{E}[Y]\theta_2 + \mathrm{Cov}(X,Y)\theta_2 - \mathbb{E}[Y]\theta_3, \\ \frac{d\mathrm{Var}(X)}{dt} &= \mathbb{E}[X]\theta_1 + 2\mathrm{Var}(X)\theta_1 + \mathrm{Cov}(X,Y)\theta_2 \\ &\quad + \mathbb{E}[X]\mathbb{E}[Y]\theta_2 - 2\mathbb{E}[X]\mathrm{Cov}(X,Y)\theta_2 - 2\mathbb{E}[Y]\mathrm{Var}(X)\theta_2, \\ \frac{d\mathrm{Cov}(X,Y)}{dt} &= \mathrm{Cov}(X,Y)\theta_1 - \mathrm{Cov}(X,Y)\theta_2 - \mathrm{Cov}(X,Y)\theta_3 \\ &\quad - \mathbb{E}[X]\mathbb{E}[Y]\theta_2 + \mathbb{E}[X]\mathrm{Cov}(X,Y)\theta_2 + \mathbb{E}[Y]\mathrm{Var}(X)\theta_2 \\ &\quad - \mathbb{E}[Y]\mathrm{Cov}(X,Y)\theta_2 - \mathbb{E}[X]\mathrm{Var}(Y)\theta_2, \\ \frac{d\mathrm{Var}(Y)}{dt} &= \mathrm{Cov}(X,Y)\theta_2 + \mathbb{E}[Y]\theta_3 - 2\mathrm{Var}(Y)\theta_3 \\ &\quad + \mathbb{E}[X]\mathbb{E}[Y]\theta_2 + 2\mathbb{E}[Y]\mathrm{Cov}(X,Y)\theta_2 + 2\mathbb{E}[X]\mathrm{Var}(Y)\theta_2. \end{aligned}$$

Remark 4.4. *For a 3rd order approximation the total number of computed mo-*

ments is increased to 9, while more terms are added to the (above) lower order ODEs.

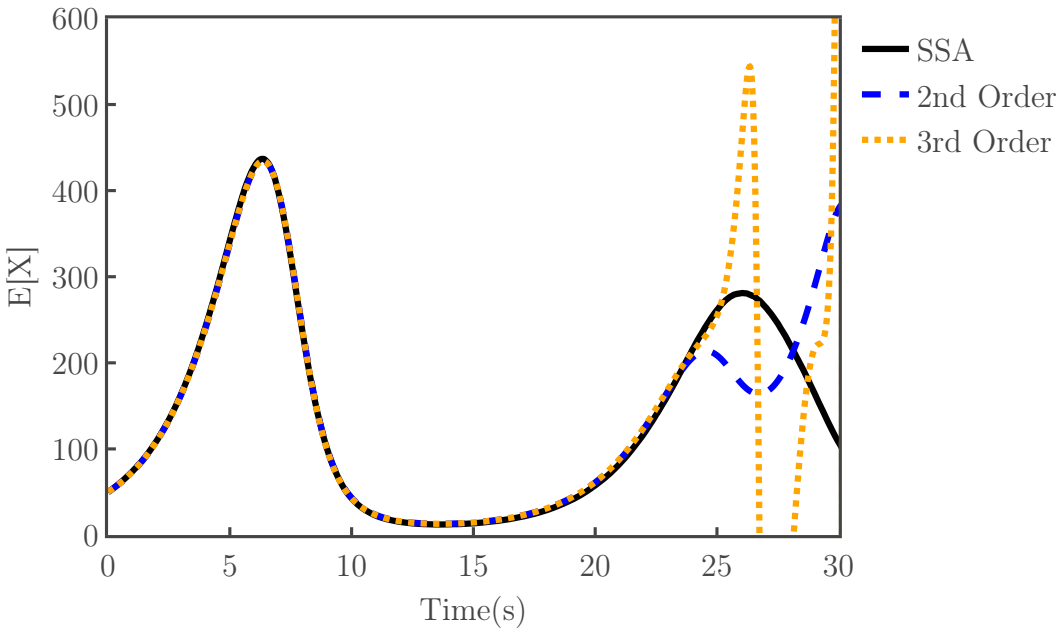
Given the choice of initial populations $(X, Y) = (50, 50)$, the temporal evolution of each species using the 2nd and 3rd order MAs are shown in Figure 4.1 alongside the 'true' solution obtained from averaging 1,000 exact SSA simulations.

It is readily seen that both the 2nd and 3rd order models do a good job of recreating the system dynamics up until time $t \approx 21$, at which point the behaviour diverges. This divergence occurs regardless of the choice of integration library (e.g. SUNDIALS CVODES [58], LSODA [57], FORTRAN RADAU [35]) and their respective absolute (and relative) tolerance settings. Choosing different closures, such as the multivariate-normal moment closure, also results in non-physical behaviour. In addition, Figure 4.1 shows that the problem actually *worsens* when higher orders are taken into account.

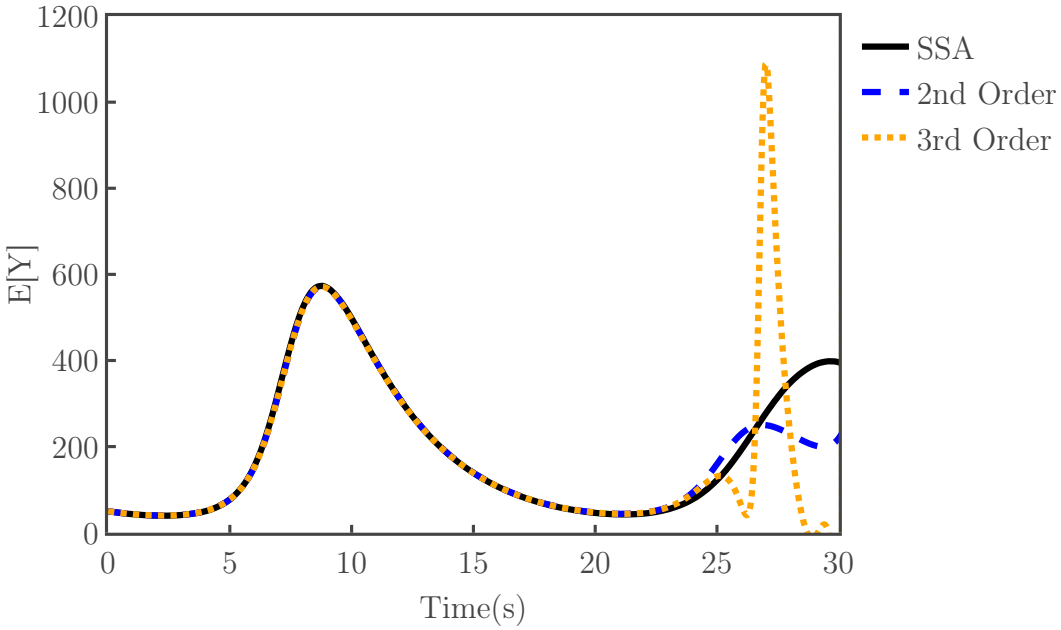
One intuition behind the failure of the moment approximations for the Lotka-Volterra model is that given the initial conditions $(X, Y) = (50, 50)$, the specified model is 'unable to deal' with the level of stochasticity. In fact, the underlying stochastic model possesses a very problematic 'extinction' event — where either the population of the prey or predators can, in theory, reach zero. Should the prey ever become extinct, the population of the predators will also die out. Conversely, if all the predators are removed from the model, the number of prey will increase exponentially without limitation. Thus, while the mean of the prey, $\mathbb{E}[X]$, may be well defined, its variance $\mathrm{Var}(X)$ can diverge rapidly. This suggests the moment approximation technique is not well equipped to handle ill-defined moments. In Section 4.4, I suggest that many poor approximations can stem from the false assumption that the Taylor series expansion is valid, and propose a new technique that could improve moment approximations.

4.4 Padé-type Approximants for Better Moment Closure Expansions

In this section, I propose a new approach to constructing moment approximation equations based on Padé-type approximants. This was motivated, in part, by en-



(a) Evolution of the prey population.



(b) Evolution of the predator population

Figure 4.1: Comparison of the temporal evolution of the Lotka-Volterra model using the SSA, 2nd order MA, and 3rd order MA, for the dynamics of (a) $\mathbb{E}[X]$ — the prey population, and (b) $\mathbb{E}[Y]$ — the predator population. Initial conditions were $(X, Y) = (50, 50)$. Solutions diverge from $t \approx 21$ s.

counters with numerical instabilities, and poor approximations given by existing techniques.

Presently *all* moment closure approximations that have been implemented for stochastic chemical reaction networks use the Taylor series expansion, without discrimination, to truncate infinite sets of ODEs up to a given order. However, it is well known from various problems of mathematics, physics, and engineering that the Taylor series approximation is not always valid, or viable — *e.g.*, in quantum mechanics calculations can often call for the summation of infinite series with unknown terms [10]. It should be recalled that the Taylor series expansion of a function is not valid beyond the radius of convergence [102], which itself can be equal to 0 in many systems — *i.e.*, for some systems it never converges. For stochastic biochemical reaction networks, it can be reasoned that many models — particularly those that exhibit large variance, or a *heavy-tailed* distribution — will produce a non-convergent Taylor series expansion (for motivating examples within the field of econometrics and expected utility, see [59, 84]). This can be seen by noting that the rate at which the sequence of central moments increases, $\{\mathbb{E}[(x-\mu)^2], \ldots, \mathbb{E}[(x-\mu)^k]\}$, is too great for such a heavy-tailed distribution (*i.e.*, by performing a simple ratio test on successive terms of the Taylor series expansion).

Padé approximants have been widely used in practice to provide an effective way of analytically continuing an approximated function beyond its radius of convergence [120]. Mathematically, the Padé approximant of a function is the 'best' *rational* power series approximation up to a given order, and is closely related to continued fractions, orthogonal polynomials, and Gaussian quadrature methods [18]. Interestingly, even with few terms the Padé approximants often provide better approximations than their counterpart Taylor series — for examples, see the works of Bender [10, 11]. Furthermore, Padé approximants can often work in systems where the corresponding Taylor series is divergent. The Padé approximant $R_{[m/n]}(x)$ of a power series function $f(x)$ is defined as follows:

Definition 4.3 (Padé Approximant of Order $[m/n]$ [10]). *For $m \geq 0$, $n \geq 1$, the Padé approximant of a function $f(x)$ up to order $[m/n]$ is the rational function*

$$R_{[m/n]}(x) = \frac{\sum_{k=0}^{m} a_k x^k}{1 + \sum_{k=1}^{n} b_k x^k} = \frac{a_0 + a_1 x + a_2 x^2 + \cdots + a_m x^m}{1 + b_1 x + b_2 x^2 + \cdots + b_n x^n}. \tag{4.4.1}$$

where the Padé has been normalised by choosing $b_0 = 1$, without loss of generality [10].

Remark 4.5 (Agreement of the Padé Approximants). *Given $f(x)$ is a (Taylor) power series, the Padé approximant $R_{[m/n]}(x)$ agrees with $f(x)$ up to the order $m + n$,* e.g.*,*

$$\begin{aligned} f(0) &= R_{[m/n]}(0) \\ f'(0) &= R'_{[m/n]}(0) \\ &\vdots \\ f^{(m+n)}(0) &= R^{(m+n)}_{[m/n]}(0) \end{aligned}$$

Construction of the rational Padé approximants is done by using the corresponding terms of the Taylor series expansion. Suppose that $f(x) = \sum_{k=0}^{\infty} c_k x^k$ is a Taylor power series expansion. Let the coefficient $b_0 = 1$ as in Definition 4.3. The remaining $(m + n + 1)$ coefficients, $\{a_0, \ldots, a_m; b_1, \ldots, b_n\}$, can be derived by solving a set of matrix operations [10]. The denominator coefficients, b_k, are obtained by solving:

$$\begin{bmatrix} c_m & c_{m-1} & \cdots & c_{m-n+1} \\ c_{m+1} & c_m & \cdots & c_{m-n+2} \\ \vdots & \vdots & \ddots & \vdots \\ c_{m+n-1} & \cdots & \cdots & c_m \end{bmatrix} \cdot \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} = - \begin{bmatrix} c_{m+1} \\ c_{m+2} \\ \vdots \\ c_{m+n} \end{bmatrix}. \tag{4.4.2}$$

The numerator coefficients, a_k, can then be found by solving

$$a_k = \sum_{i=0}^{k} c_{k-i} b_i, \qquad 0 \leq i \leq m, \tag{4.4.3}$$

where $b_k = 0$ for $i > n$ [10]. A simple example for the calculation of $R_{[0/1]}(x)$ is given in Example 4.2.

Example 4.2 (Computation of $R_{[0/1]}(x)$ [10]). *Suppose the aim is to compute $R_{[0/1]}(x)$ of a Taylor series, $f(x) = \sum_{k=0}^{\infty} c_k x^k$, such that*

$$R_{[0/1]}(x) = \frac{a_0}{1 + b_1 x}.$$

With respect to Equations (4.4.2) and (4.4.3), this gives two equations: $a_0 = c_0 b_0$, *and* $-c_0 b_1 = c_1$. *With the normalisation* $b_0 = 1$, *solving for the Padé coefficients reveals the approximant*

$$R_{[0/1]}(x) = \frac{c_0}{1 - \frac{c_1}{c_0} x}.$$

Remark 4.6. *It can be seen from Equations (4.4.2) and (4.4.3) that each coefficient in the numerator and denominator of the Padé approximant is a function (comprising sums, fractions and multiplications) of the original Taylor expansion coefficients. Thus, no new information is required to be computed.*

In general, there exists many algorithms for optimally computing Padé-type approximants [17, 65], which are closely related to Shanks tranformations [4]. One particular algorithm, Wynn's ϵ-algorithm [131], is well known to efficiently compute the coefficients in a recursive manner.

To apply the Padé method to stochastic chemical kinetics, the standard Taylor series (which generates the sequence of central moment corrections) is switched out for the rational Padé-type approximant of desired order. For example, computing the temporal evolution of the mean of a one-dimensional system (*e.g.*, Equation (4.2.4)), obtains the following formula comprising a rational function of the central moments, *i.e.*,

$$\frac{d}{dt}\mathbb{E}[x] = \sum_{j=1}^{m} v_j \frac{a_0 + a_1\mathbb{E}[x-\mu] + a_2\mathbb{E}[(x-\mu)^2] + \cdots}{1 + b_1\mathbb{E}[x-\mu] + b_2\mathbb{E}[(x-\mu)^2] + \cdots}, \tag{4.4.4}$$

where v_j is the j-th entry of the state-change matrix, and the coefficients, a_k, b_k, can be analytically solved for prior to simulation. The same computations can be analogously applied for the equations of higher order central moments. Within Equation (4.4.4), the rational coefficients, a_k, b_k, are functions of the Taylor coefficients

$$\frac{\partial^{\boldsymbol{k}}}{\partial \boldsymbol{x}^{\boldsymbol{k}}} \frac{h_j(\boldsymbol{x}, \boldsymbol{\theta})}{\boldsymbol{k}!}\bigg|_{\boldsymbol{x}=\boldsymbol{\mu}},$$

recalling that $h_j(\boldsymbol{x}, \boldsymbol{\theta})$ is the j-th propensity function. For generalised higher order moments, the rational coefficients are functions of the following Taylor coefficients

$$\frac{\partial^{\boldsymbol{k}}}{\partial \boldsymbol{x}^{\boldsymbol{k}}} \frac{((\boldsymbol{x} + \boldsymbol{v}_j)^{\boldsymbol{k}'} - \boldsymbol{x}^{\boldsymbol{k}'}) \cdot h_j(\boldsymbol{x}, \boldsymbol{\theta})}{\boldsymbol{k}!}\bigg|_{\boldsymbol{x}=\boldsymbol{\mu}}.$$

In general, the coefficients of the rational Padé approximants can always be uniquely determined for 1-dimensional systems. However, this is not the case in multivariate systems, and care should be taken. In Section 4.4.1, I apply the above method to the 1-dimensional Schlögl system.

4.4.1 Proof of Concept

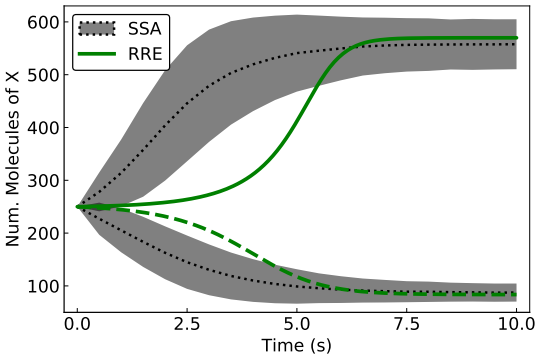
In this subsection, I give a proof of concept for exploring the use of Padé approximants within the construction of moment closure approximation equations. I consider as an example, the 1-dimensional Schlögl system [114].

$$2X + A \xrightarrow{\theta_1} 3X \qquad B \xrightarrow{\theta_3} X$$
$$3X \xrightarrow{\theta_2} 2X + A \qquad X \xrightarrow{\theta_4} B$$

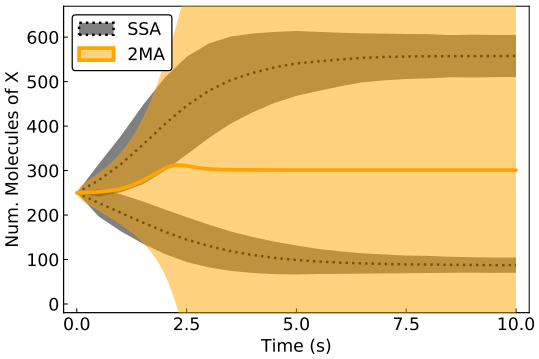
In this thesis, I use the Schlögl model with kinetic rate parameters $\theta_1 = 3\text{e}{-7}$, $\theta_2 = 1\text{e}{-4}$, $\theta_3 = 1\text{e}{-3}$, and $\theta_4 = 3.5$, and the initial population conditions $(X, A, B) = (250, 1\text{e}5, 2\text{e}5)$. Stochastically, the Schlögl system exhibits interesting bimodal behaviour and is therefore frequently studied for its challenging dynamics. Specifically, under stochastic simulation there exists a steady upper state at approximately $x \approx 560$ molecules, and a steady lower state at $x \approx 85$ molecules. Using the above parameters, the system is very sensitive to stochastic fluctuations, or small perturbations (*e.g.*, 1-5 molecules) in the initial molecular counts. Minor changes in parameters can also lead to one steady state dominating the other.

In Figure 4.2, the upper and lower steady states of the stochastic solution are shown, alongside their corresponding standard deviation, obtained from 10,000 stochastic simulations using Gillespie's direct method [44]. For visualisation purposes, the SSA data has been partitioned. Simulations performed using traditional reaction rate equation (RRE) ODE dynamics are presented in Figure 4.2a, showing its (eventual) convergence to the steady states of the SSA solution. The solid green line represents the normal ODE dynamics, whilst the green dashed line has had the initial starting state perturbed (by 1 molecule) to show the lower steady state. Within Figure 4.2b, the solution for the mean and standard deviation of the 2nd order moment approximation (zero-MA) is shown. Under the second order approximation, the system's variance (and standard deviation)

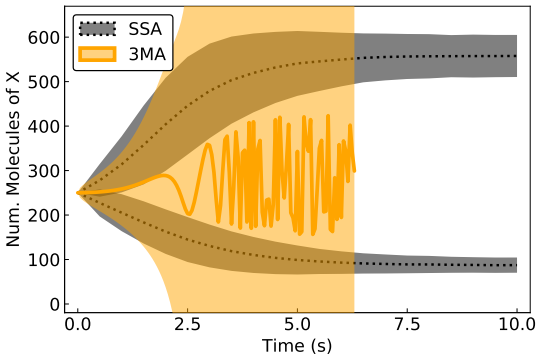
grows until it becomes very large with respect to the mean at $\sim$ 2s. At this point, the approximation becomes poor and non-physical, with the variance exponentially tending toward infinity. Within Figure 4.2c, the corresponding solution to the 3rd order moment approximation is given. It can be seen that the numerical solution is very close to that of the 2nd order approximation until $\sim$ 2s, at which point the numerical stability breaks down dramatically with computation becoming infeasible after $\sim$ 6s. In Figure 4.2d, the results of the simulation using the Padé approximation $R_{[1/1]}(x)$ is depicted. The trajectory of the mean closely follows the same path as the RRE in Figure 4.2a, for both the perturbed and normal initial starting conditions. Using the $R_{[1/1]}(x)$ approximation the standard deviation of the paths are finite, albeit larger than that of the SSA solution. Unlike the standard moment approximation, an improvement is made by taking the higher order $R_{[2/2]}(x)$ Padé approximation. Figure 4.2e shows that the [2/2] order approximation not only does a better job at reaching the exact mean of the upper and lower steady states, but also makes significant improvements on the magnitude of their respective standard deviations. In Table 4.1 I present the relative errors and computation time of the RRE vs Padé approximation methods. Each of the methods provides excellent approximations of the means of the upper and lower steady states. Naturally, the simple RRE approach is the fastest technique on average, followed by the Padé methods which are approximately twice as slow. For completeness, the average relative computation time taken by the 2nd order moment approximation (2MA) is included. It is noted that the CPU time is significantly higher (8$\times$) for the 2MA. This is due to the integration of the equations becoming more stiff as the variance of the system becomes (non-physically) large. In general, it is anticipated that the 2MA should be slightly faster for a well-behaved model.



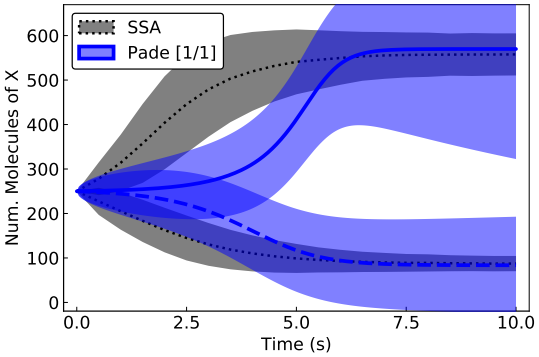
(a) The reaction rate equation (RRE) ODE dynamics of the Schlögl system.



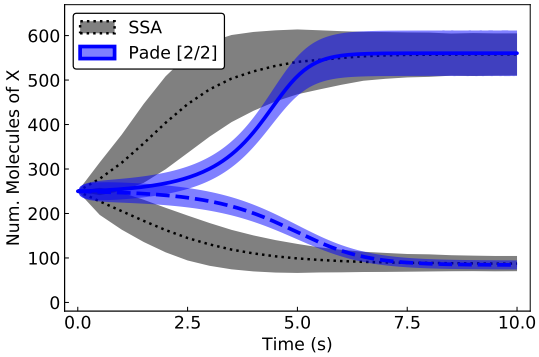
(b) 2nd Order Moment Closure Approximation (2MA).



(c) 3rd Order Moment Closure Approximation (3MA).



(d) Padé Approximation $R_{[1/1]}(x)$



(e) Padé Approximation $R_{[2/2]}(x)$

Figure 4.2: A comparison of the Schlögl system dynamics for various moment approximations, compared to the solution (of 10,000 simulations) of the SSA. For visual aid, the SSA data has been partitioned into 'upward' and 'downward' states. Shown are the means (solid/dashed lines), and the standard deviation (ribbons). To produce the downward states from the RRE (Figure 4.2a), and Padé approximants (Figures 4.2d and 4.2e), the initial starting state was perturbed very slightly. Perturbing the 2MA and 3MA (Figures 4.2b and 4.2c) has no effect. In Figures 4.2b and 4.2c, the standard deviation exponentially increases toward infinity. In Figure 4.2c, numerical instability occurs, and computation is infeasible after $\sim$ 6s.

Method	Upper Mean Rel. Err. (%)	Lower Mean Rel. Err. (%)	Upper Std. Rel. Err. (%)	Lower Std. Rel. Err. (%)	Rel. CPU Time
RRE	2.15	4.05	—	—	1.00
Padé [1/1]	2.15	4.05	420	551	1.80
Padé [2/2]	0.46	3.16	7.19	39.3	2.44
2MA	—	—	—	—	8.08

Table 4.1: The relative error and CPU time of the Padé approximants vs the RRE. The relative errors (Rel. Err) are taken with respect to the final upper- and lower- steady-state molecular means of X and their respective standard deviations (Std.). The relative (Rel.) CPU time taken for each method is scaled according to the average time of the RRE simulation. The RRE method does not include the standard deviation within its simulations. The relative CPU time of the 2MA is included for completeness. The 3MA is not included due to the simulations halting prematurely.

4.5 Generalized Method of Moments for Parameter Inference

The generalized method of moments (GMM) for stochastic biological networks is a fresh approach to the parameter inference problem, inspired by advances in the field of econometrics [50, 51]. Recent developments in the construction of moment closures (see Sections 4.2 and 4.2.1) have led to scalable methods that are able to make *ad-hoc* approximations of the temporal evolution of the underlying probability distribution up to a given order k. Returning a set of ODEs, the attractiveness of these moment approximation methods is their independence to the population size. However, the drawback is that these systems readily become stiff, and their approximations poor for large k. Nevertheless, reconstructions of complex distributions show that even for small orders k (*e.g.*, $k = 2$), the moment approximations can provide sufficient information about the model dynamics [5]. Recent advances include the use of conditional-moment approximations [54], which partition the state-space into two subregions. Within [54], the subregions containing larger molecular populations are solved using standard moment closure approximations, while the low population species are solved using the chemical master equation [89].

The principle behind the GMM for parameter inference is to use the observed sample moments, within the data and simulations, to construct a cost function to be minimised during optimisation. However, special care should be taken as:

- The cost function of low and high order moments need to be weighted carefully — *e.g.*, in the case of a least-squares cost function.
- Higher order moments are generally less accurate than their lower order counterparts.
- There exists a degree of correlation between different order moments — motivating the use of mixed moment cost functions [51, 85].

The general optimisation problem can be posed as a maximum likelihood estimation problem (see *e.g.*, [113]) defined by

$$\boldsymbol{\theta}^*_{\text{MLE}} = \arg\max_{\theta} \mathcal{L}(\boldsymbol{y}, \boldsymbol{\theta}), \tag{4.5.1}$$

where $\mathcal{L}(\boldsymbol{y}, \boldsymbol{\theta})$ is the likelihood of the data, $\boldsymbol{y}$, given the parameters $\boldsymbol{\theta}$. Exact computation of Equation (4.5.1) is often expensive, and computationally infeasible, motivating the use of approximations — such as least-squares distance measures based on the observed sample moments of the data [85], or likelihood approximations based on the observed sample moments [113]. In this thesis, I consider the use of least-squares distance measures due to their efficiency and ease of implementation. Let the k-th order observed sample moment of species S_i be defined as in Definition 4.4.

Definition 4.4 (Sample Moments [85]). *The k-th order sample moment, at time t, for species S_i, can be computed from the data, $\boldsymbol{y}$, by*

$$\hat{Y}_i^{(k)}(t) = \frac{1}{N}\sum_{l=1}^{N} y_i^k(t), \qquad 1 \leq i \leq n, \tag{4.5.2}$$

where N Monte Carlo samples are observed.

Remark 4.7. *Given large N, Equation (4.5.2) tends toward an asymptotically unbiased estimator of the true population moment.*

Similarly, let the k-th order observed *central* moment be defined as in Definition 4.5.

Definition 4.5 (Sample Central Moments [50, 85]). *The k-th order central moment, at time t, for species S_i, can be computed from the data, $\boldsymbol{y}$, by*

$$\hat{F}_i^{(k)}(t) = \frac{1}{N}\sum_{l=1}^{N} \left(y_i(t) - \hat{Y}_i^{(k)}(t)\right)^k, \qquad 1 \leq i \leq n, \tag{4.5.3}$$

where N Monte Carlo samples are observed.

Remark 4.8 (Demean Estimator). *Within [85], the form of the k-th order sample central moment (Equation (4.5.3)) is termed the 'demean' estimator. Using the sample central moments, or demean estimator, often helps to remove numerical inconsistencies obtained by approximating the covariance matrices.*

The least-squares estimator, $\hat{\boldsymbol{\theta}}$, of a moment-based simulation, with respect to the data, is given in Definition 4.6. Under the assumption that data is normally distributed, Equation (4.5.4) coincides with the maximum likelihood estimator of the system.

Definition 4.6 (Least Squares Moments Estimator [50, 85]). *The ordinary least-squares estimator, $\hat{\boldsymbol{\theta}}$, based on the sample moments of the data, $\hat{\boldsymbol{Y}}$, is given by*

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \sum_{k=1}^{r} \left(\hat{Y}^{(k)}(t) - X^{(k)}(\boldsymbol{\theta}, t) \right)^2, \tag{4.5.4}$$

where $X^{(k)}(\boldsymbol{\theta}, t)$ are the moments for species derived from forward simulation with $\hat{\boldsymbol{\theta}}$. r is the maximal occurring order moment.

With analogy to Equation (3.6.1), one sub-element of the cost function (based on species S_i) can be constructed as

$$J_{ik}(\boldsymbol{\theta}, t) = \hat{Y}_i^{(k)}(t) - X_i^{(k)}(\boldsymbol{\theta}, t). \tag{4.5.5}$$

As previously mentioned within this section, appropriate weights must be selected for each moment condition, and steps need to be taken to minimise the correlation between moment conditions. This motivates the use of mixed moment conditions [51], and a weighting matrix, $\boldsymbol{W}$, such that

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \boldsymbol{J}(\boldsymbol{\theta}, t)^\intercal \boldsymbol{W} \boldsymbol{J}(\boldsymbol{\theta}, t), \tag{4.5.6}$$

where $\boldsymbol{J}(\boldsymbol{\theta}, t)$ is a vector of elements $\{J_{ik}(\boldsymbol{\theta}, t) : k \in \{1, \dots, r\}, i \in \{1, \dots, n\}\}$. For the choice that $\boldsymbol{W} = \mathbb{I}$, *i.e.*, the identity matrix, Equation (4.5.6) becomes a straightforward least-squares estimator. Otherwise, if $\boldsymbol{W}$ is a positive semi-definite weighting matrix, we obtain a *weighted* least-squares estimator. The choice of $\boldsymbol{W}$ is often crucial to achieving optimal convergence, and estimating $\boldsymbol{W}$ itself adds an additional level of complexity [50]. It can be shown that the optimal weighting matrix is given by

$$\boldsymbol{W} = \mathbb{E}\left[\boldsymbol{J}_i(\boldsymbol{\theta}^*, t) \boldsymbol{J}_i(\boldsymbol{\theta}^*, t)^\intercal \right]^{-1}, \tag{4.5.7}$$

where $\boldsymbol{\theta}^*$ is the optimal parameter, which is however, unknown.

Remark 4.9. *Computation of Equation (4.5.7) is infeasible, calling for the Monte Carlo sampling counterpart below*

$$\boldsymbol{W} = \left(\sum_{i=1}^{N} \boldsymbol{J}_i(\boldsymbol{\theta}^*, t)^\intercal \boldsymbol{J}_i(\boldsymbol{\theta}^*, t) \right)^{-1}. \tag{4.5.8}$$

In practice, there are several methods that deal with estimating $\boldsymbol{W}$. The most commonly used are (see [50])

- The two-step GMM,
- The iterated GMM,
- The continuously updating GMM.

Two-Step GMM [50] The two-step GMM approach tackles the estimation of $\boldsymbol{\theta}^*$ and $\boldsymbol{W}$ independently in the following iterative manner:

Step 1: Using $\boldsymbol{W} = \mathbb{I}$, solve the optimisation routine (Equation (4.5.6)) to compute a preliminary estimate $\tilde{\boldsymbol{\theta}}$.

Step 2: Using $\tilde{\boldsymbol{\theta}}$, compute the estimate $\hat{\boldsymbol{W}}(\tilde{\boldsymbol{\theta}})$ (from Equation (4.5.7)), and use the estimated weighting matrix to compute the iteration estimate $\hat{\boldsymbol{\theta}}$.

Iterated GMM [50] The iterated GMM approach is near identical to the two-step approach, with the exception that step 2 is repeated over many iterations.

Continuously updating GMM [50, 52] The continuously updating GMM estimates both $\boldsymbol{\theta}^*$ and $\boldsymbol{W}$ simultaneously, such that $\hat{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{W}}$ are obtained using the same parameters.

In general, it can be shown that both the two-step and iterated approach provide asymptotically efficient estimators, while the continuously updating GMM achieves good results in practical applications [52].

4.6 Applying Multi-level Methods to the Generalised Method of Moments

Only recently have state-of-the-art parameter inference techniques utilising the generalised method of moments for stochastic biological models become established. Primarily, the GMM has been popularized as a scalable alternative to intensive (exact SSA) simulation approaches [113, 116]. However, the cost of using ODE moment approximations is the loss of accuracy of the underlying models compared to their exact method counterparts — which may lead to bias, or become detrimental to the quality of the obtained parameter estimates. In recent

work, the authors of [113] perform parameter inference using 'adaptive' moment closures. The core idea behind the adaptive closure method is to use small numbers of exact Gillespie stochastic simulations to automatically assess, and 'choose' the optimal moment closure within each region of the parameter search space.

Within this section I propose a new two-step approach to parameter inference based on combining the traditional generalised method of moments with multi-level approximation techniques (MLGMM). Specifically, I investigate the use of adaptive multi-level tau-leaping (AMLT) [80] — a recent, and novel technique that provides an efficient formulation for stochastically simulating the evolution of population statistics.

- In Section 4.6.1, I outline the adaptive multi-level tau-leaping algorithm.
- In Section 4.6.2, I present a new approach for parameter inference based on the GMM and AMLT algorithm.

4.6.1 Adaptive Multi-Level Tau-Leaping Algorithm

As discussed in Chapter 2, direct stochastic simulation using Gillespie's direct method [44] can be computationally burdensome — motivating the development of approximate tau-leaping methods that trade computational efficiency for accuracy [43]. In Section 2.3.2, I presented the optimised tau-leaping algorithm from [20], which implements adaptive time steps that can be tuned via an error controlling parameter ϵ. As ϵ increases, the time steps taken become (in general) larger, and the computation more efficient — albeit at the cost of bias.

Multi-level Monte Carlo techniques for continuous time Markov chains [130] attempt to efficiently estimate population moment statistics using stochastic simulations. The original multi-level tau-leaping method was proposed by Anderson and Higham [3]. The key idea was to use tau-leaping with large (coarse) time steps to quickly estimate base population statistics, and then to use a handful of trajectories with finer time steps to add a 'correction' term to the coarse path. More recently, the work has been expanded upon by Lester *et al.* within [80] to include adaptive time steps — *i.e.*, the optimised tau-leaping method (with the error control parameter ϵ) discussed previously [20]. Thus, the adaptive multi-level tau-leaping algorithm aims to generate many sample trajectories using a

high tolerance (large ϵ), and then to correct the estimated statistics using fewer follow-up trajectories of low tolerance [80].

Suppose the aim is to estimate the mean of the population, $\boldsymbol{x}_i$, of a species S_i at a time T. The base level estimator is given in Definition 4.7.

Definition 4.7 (Base Level Estimator [80]). *The base level estimator is generated by sampling n_0 tau-leaping paths with an error control parameter ϵ_0,*

$$Q_0 \equiv \mathbb{E}[x(\epsilon_0)] \approx \frac{1}{n_0}\sum_{i=1}^{n_0} x_i(\epsilon_0), \tag{4.6.1}$$

where $x_i(\epsilon_0)$ is generated in the i-th simulated trajectory [80].

The subsequent levels of approximation, $l \in \{1, 2, \ldots\}$, aim to reduce the bias of Q_0 by adding correction terms generated from n_l paths with error tolerances of ϵ_l and ϵ_{l-1}, *e.g.*,

$$Q_1 \equiv \mathbb{E}[x(\epsilon_1) - x(\epsilon_0)] \approx \frac{1}{n_1}\sum_{i=1}^{n_1}[x_i(\epsilon_1) - x_i(\epsilon_0)]. \tag{4.6.2}$$

The general l-th correction term is defined in Definition 4.8

Definition 4.8 (l-th Level Correction Term [80]). *The general l-th level correction term, generated using n_l tau-leaping paths with control parameters ϵ_l and ϵ_{l-1} is given by*

$$Q_l \equiv \mathbb{E}[x(\epsilon_l) - x(\epsilon_{l-1})] \approx \frac{1}{n_l}\sum_{i=1}^{n_l}[x_i(\epsilon_l) - x_i(\epsilon_{l-1})]. \tag{4.6.3}$$

Starting from the base level estimator, Q_0, it can be shown that summing the correction terms Q_l forms a telescoping sum that reduces the bias. This can be seen by noting that

$$Q_0 + Q_1 = \mathbb{E}[x(\epsilon_0)] + \mathbb{E}[x(\epsilon_1) - x(\epsilon_0)] = \mathbb{E}[x(\epsilon_1)].$$

Thus, summing the first two estimators, Q_0 and Q_l has a bias equivalent to that of a tau-leaping method with $\epsilon = \epsilon_1$ [80]. By constructing multiple levels, a final, L-level estimator $\mathbb{Q}$ can be obtained such that

$$\mathbb{Q} = \mathbb{E}[x(\epsilon_L)] = \mathbb{E}[x(\epsilon_0)] + \sum_{l=1}^{L}\mathbb{E}[x(\epsilon_l) - x(\epsilon_{l-1})] = \sum_{l=0}^{L} Q_l. \tag{4.6.4}$$

Remark 4.10. *Calibration is required to optimise the number of samples generated per level, n_l, against the run time, c_l, and estimator variance σ_l^2 from each contributing level, Q_l. This can be formulated as the minimisation of $\sum_{l\geq 0} c_l \cdot n_l$ subject to the constraint $\sum_{l\geq 0} \sigma_l^2/n_l < \alpha$, for some fixed α, which can be achieved in practice using prior simulations [3, 80].*

For constructing the l-th level correction terms within Definition 4.8, the trajectories produced with error control parameters ϵ_l and ϵ_{l-1} are deliberately simulated in a highly coupled, correlated fashion [3, 80]. This is because the correction terms Q_l only depend on the differences in the systems, not on the actual copy numbers — and it is therefore beneficial if the systems are kept as similar to one another as possible during forward simulation. This can be done using the *Poisson process thickening property.*

Proposition 4.2 (Poisson Thickening Property [94]). *Suppose we have the Poisson variates $\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3$ which are of rate parameters a, b, and $a+b$, respectively. The thickening property states:*

$$\mathcal{P}_1(a+b) = \mathcal{P}_2(a) + \mathcal{P}_3(b). \tag{4.6.5}$$

Using Proposition 4.2, it is possible to calculate the Q_l-th correction term for a system that is simulated until a final time T in the following manner:

1. Let $\boldsymbol{z}_c(t)$ and $\boldsymbol{z}_f(t)$ denote the *coarse* and *fine* paths at time t generated with tolerances ϵ_{l-1} and ϵ_l, respectively. Let $h_j^{(c)}$ and $h_j^{(f)}$ denote the propensity vectors for the coarse and fine paths.

2. Let $\tau(\epsilon, \boldsymbol{z})$ be a function that computes the size of the next leap (as in [20]), and thus the next update times, T_c and T_f, for the coarse and fine paths respectively. Define the next coupled update time as $\eta(t) = \min\{T_c, T_f, T\} - t$.

3. At time $\eta(t)$, compute the following auxiliary variables for each reaction channel $\mathcal{R}_j$ [80]:

$$\begin{aligned} b_j^{(1)} &= \min\{h_j^{(c)}, h_j^{(f)}\}, \\ b_j^{(2)} &= h_j^{(c)} - b_j^{(1)}, \\ b_j^{(3)} &= h_j^{(f)} - b_j^{(1)}. \end{aligned}$$

4. Using Proposition 4.2, calculate the rate parameters of the thickened Poisson process, *i.e.*,

$$\mathcal{P}(a_j^{(c)} \cdot \eta) \sim \mathcal{P}(b_j^{(1)} \cdot \eta) + \mathcal{P}(b_j^{(2)} \cdot \eta),$$
$$\mathcal{P}(a_j^{(f)} \cdot \eta) \sim \mathcal{P}(b_j^{(1)} \cdot \eta) + \mathcal{P}(b_j^{(3)} \cdot \eta),$$

5. Update the state of the coarse and fine paths over the interval $\min\{T_c, T_f, T\}$ by sampling from the above thickened Poisson distributions. Update the system time, $t \to t + \eta$. If $t > T_c$ or $t > T_f$, update the propensity and update time of the respective path.

6. Repeat steps 2–5.

The AMLT algorithm for the correction terms is outlined below in Algorithm 4.

4.6.2 Outline of the MLGMM

In this section, I outline the Multi-Level Generalised Method of Moments (MLGMM) — a new approach for parameter inference. At the simplest level of abstraction, the MLGMM can be summarised as an adaptable two-component optimisation method:

Step 1: For a given choice of moment closure (*e.g.*, the normal 2nd order moment approximation), perform a quick, robust *global* optimisation of Equation (4.5.6) using the continuously updating GMM approach. Retrieve the estimate $\tilde{\boldsymbol{\theta}}$.

Step 2: Given the estimated parameters $\tilde{\boldsymbol{\theta}}$ from step 1, use the adaptive multi-level tau-leaping approach to perform a *local* optimisation of Equation (4.5.6) to produce stochastically-corrected, *refined* parameter estimates, $\hat{\boldsymbol{\theta}}$.

Remark 4.11. *Because the outputs of both the moment approximation and adaptive multi-level tau-leaping methods are population moment statistics, switching from one method to the other is relatively straightforward — and requires no need for independent cost functions.*

The above two-component approach is designed to exploit the fast, scalable nature of moment approximations to produce good-enough estimates such that the more

Algorithm 4 Adaptive Multi-level Tau-leaping Algorithm - Correction Terms

procedure GENERATE Q_l-TH CORRECTION TERM
  $t \leftarrow t_0$ // set the current system time
  $\boldsymbol{x}_c(t) \leftarrow \boldsymbol{x}_0(t)$ // set the coarse path at time t
  $\boldsymbol{x}_f(t) \leftarrow \boldsymbol{x}_0(t)$ // set the fine path at time t
  for $1 : j$ **do**
    $h_j^{(c)} \leftarrow h_j(\boldsymbol{x}_c(t), \boldsymbol{\theta})$ // set the propensities
    $h_j^{(f)} \leftarrow h_j(\boldsymbol{x}_f(t), \boldsymbol{\theta})$
  $T_c \leftarrow t + \tau(\epsilon_c, \boldsymbol{x}_f(t))$
  $T_f \leftarrow t + \tau(\epsilon_f, \boldsymbol{x}_c(t))$
  while $t < T$ **do**
    $\eta \leftarrow \min\{T_c, T_f, T\} - t$ // next coupling time
    for $j = 1 : m$ **do**
      $b_j^{(1)} \leftarrow \min\{h_j^{(c)}, h_j^{(f)}\}$ // auxilliary Poisson process variables
      $b_j^{(2)} \leftarrow h_j^{(c)} - b_j^{(1)}$
      $b_j^{(3)} \leftarrow h_j^{(f)} - b_j^{(1)}$
      for $r = 1 : 3$ **do**
        $Y_j^{(r)} \leftarrow \mathcal{P}(b_j^{(r)} \cdot \eta)$ // random Poisson numbers
    $\boldsymbol{x}_c(t + \eta) \leftarrow \boldsymbol{x}_c(t) + \sum_{j=1}^{m}(Y_j^{(1)} + Y_j^{(2)})\boldsymbol{v}_j$ // update states
    $\boldsymbol{x}_f(t + \eta) \leftarrow \boldsymbol{x}_f(t) + \sum_{j=1}^{m}(Y_j^{(1)} + Y_j^{(3)})\boldsymbol{v}_j$
    $t \leftarrow t + \eta$
    if $t = T_c$ **then**
      for $1 : j$ **do**
        $h_j^{(c)} \leftarrow h_j(\boldsymbol{x}_c(t), \boldsymbol{\theta})$ // update propensities
      $T_c \leftarrow t + \tau(\epsilon_c, \boldsymbol{x}_c(t))$ // compute next update time
    else
      if $t = T_f$ **then**
        for $1 : j$ **do**
          $h_j^{(f)} \leftarrow h_j(\boldsymbol{x}_f(t), \boldsymbol{\theta})$ // update propensities
        $T_f \leftarrow t + \tau(\epsilon_f, \boldsymbol{x}_f(t))$ // compute next update time

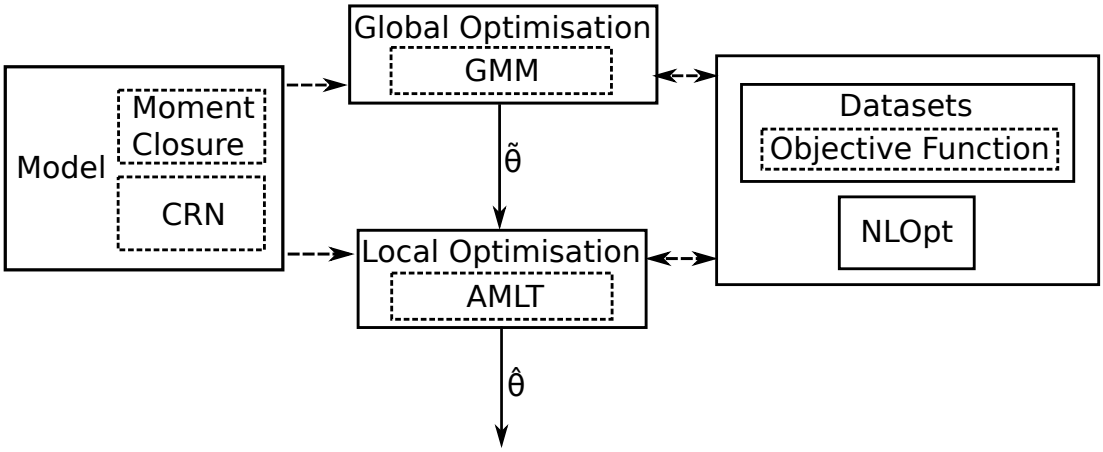


Figure 4.3: An overview of the components to the MLGMM approach to parameter inference. Given a model, the MLGMM approach automatically constructs the moment closure approximations for ODE simulations and the underlying stochastic chemical reaction network for the AMLT simulations. Global optimisation for the GMM is performed using the NLopt libary. The retrieved estimates $\tilde{\boldsymbol{\theta}}$ are then passed to the local optimisation routine which utilises AMLT for simulation. The local optimisation routine then utilises the NLopt library to produce the stochastically-corrected, refined estimates $\hat{\boldsymbol{\theta}}$.

computationally intensive AMLT step is required to do less work. Figure 4.3 provides an overview of the MLGMM approach.

I implement the MLGMM approach within the Julia programming language [13], utilising the *DifferentialEquations* and *NLopt* libraries for simulation and optimisation. In addition, I wrote a custom module, *MeMo* (Method of Moments), to automatically construct moment closure approximations by interfacing with *SymPy*, and implemented a version of the AMLT algorithm.

For the AMLT algorithm, the number of levels taken, and their respective error control parameters can be fine-tuned. As default parameter values, I use the 3 levels ($\epsilon_0 = 0.08, \epsilon_1 = 0.06, \epsilon_2 = 0.03$). The number of simulations at each level, in general, depends on the model, and the solution to the minimisation programme within Remark 4.10. In general, the number of simulations is greatest for the first level, ϵ_0, with fewer simulations at the higher levels ϵ_2. This allows computational effort to be efficiently distributed across the levels.

In Section 4.6.3 I present results for parameter inference using the MLGMM

approach.

4.6.3 Lotka-Volterra Model

To test the MLGMM approach, I first investigate the Lotka-Volterra predator-prey model, using parameters $(\theta_1, \theta_2, \theta_3) = (0.5,\ 0.0025,\ 0.3)$, and the initial population conditions $(X_1, X_2) = (100,\ 100)$.

$$\begin{aligned} X_1 &\xrightarrow{\theta_1} X_1 + X_1 \\ X_1 + X_2 &\xrightarrow{\theta_2} X_2 + X_2 \\ X_2 &\xrightarrow{\theta_3} \emptyset \end{aligned}$$

Remark 4.12 (Initial Conditions). *This model is different to that presented in Section 5.3, due to the higher starting molecular counts. This is because the system becomes more stable under moment approximations, allowing moment-based parameter inference to be efficiently performed. Also, the Lotka-Volterra system becomes more computationally heavy for the standard SSA when the populations are higher — due to the increased number of reactions.*

To construct the moment conditions, I artificially generated 1,000 datasets using Gillespie's direct method, and used them to construct the means and covariances of the species. The motivation behind using a larger number of datasets was to (i) accurately approximate the moments within the data, and (ii) because high-throughput data methods often provide vast quantities of data allowing for such population statistics to be estimated in real practice. Explicitly, the moment conditions used for parameter inference were

$$\begin{aligned} \hat{X}_1(t) - Z_1(\boldsymbol{\theta}, t), \\ \hat{X}_2(t) - Z_2(\boldsymbol{\theta}, t), \\ \hat{X}_1(t)\hat{X}_2(t) - Z_1(\boldsymbol{\theta}, t)Z_2(\boldsymbol{\theta}, t), \end{aligned}$$

For the parameter search state-space, I provided bounds to the *NLopt* library of $\theta_j \in [1\text{e}{-6}, 10]$, for $j = 1, 2, 3$. The *NLopt* method used was the COBYLA algorithm [99]. For the AMLT stage, I used 3 levels, with error controls parameters of $\epsilon = (0.08, 0.06, 0.03)$. The results of parameter estimates are presented in Table 4.2. The mean relative error (MRE) is as defined within Chapter 5. It can be seen that the GMM stage of the algorithm provides reasonable estimates, with

errors less than 30% across all 3 parameters. Moreover, the GMM stage provides the estimates in just 1.65s, eclipsing the times previously reported in Section 5.3. The AMLT however, being based on running multiple tau-leaping simulations for each parameter point within the state search-space, is significantly slower, taking $\sim$ 40 minutes. The retrieved parameters for the AMLT stage, however, have a maximum error of 2%. It can be reasoned that the algorithm requires additional heuristics to optimise the computation time spent within the GMM, and AMLT stages.

Phase	θ_1	θ_2	θ_3	θ_1 MRE(%)	θ_2 MRE(%)	θ_3 MRE(%)	CPU Time (s)
GMM	0.526	0.00322	0.345	5.18	28.6	16.6	1.65
AMLT	0.497	0.00253	0.306	0.569	1.10	2.08	2510

Table 4.2: The estimated parameters and their errors obtained from using the MLGMM for inference. The estimates afer each phase of the algorithm have been shown, alongside the (single-core) CPU time.

Chapter 5

SPICE: Experimental Results

5.1 Introduction

This section demonstrates the application of SPICE to several case studies. Specifically, results are presented for four commonly investigated biochemical reaction network systems:

- The Lotka-Volterra Predator–Prey Model. (Section 5.3)
- A Yeast Polarization Model. (Section 5.4)
- The Bimodal Schlögl System. (Section 5.5)
- The Genetic Toggle Switch. (Section 5.6)

For the first three models, (Lotka-Volterra, Yeast Polarization, Schlögl System), I perform parameter inference using artifically generated *synthetic* data to evaluate the performance of SPICE. Using the same datasets, I also perform parameter inference on the same models using common state-of-the-art algorithms implemented within the COPASI environment [61]. For these models, I also examine the effects of using (i) Gillespie's direct method *vs.* tau-leaping for simulation, (ii) the particle splitting approach, and (iii) the multiple shooting method. For the final model, the Genetic Toggle Switch, I perform parameter inference with SPICE using *real* experimental data obtained from fluorescent flow-cytometry experiments in [79].

5.2 Algorithm Settings

All experiments were conducted on a Intel Xeon 2.9GHz Linux system *without* using multiple cores. Thus, all reported CPU times are single-core.

COPASI Algorithms Within this section, the version of COPASI used to perform parameter inference comparisons against was v4.16. For comparing the efficiency of parameter inference routines, I compare SPICE to four of the best performing inference routines within COPASI [61]. Specifically, I examine the implementations of the following approaches, using the default parameters unless stated otherwise:

- Evolutionary Programming (EP) [9]

  using 250 generations, with a population of 1,000 particles.

- Evolutionary Strategy (ES) [110]

  using 250 generations, with a population of 1,000 particles.

- Genetic Algorithm (GA) [110]

  using 500 generations, with a population of 2,000 particles.

- Particle Swarm (PS) [70]

  using 1000 iterations, with a population of 1,000 particles.

Remark 5.1. *I tested each of the COPASI implementations using greater populations and more iterations (not shown), but found little improvement for the significant increase in computational cost.*

SPICE Algorithm Unless stated otherwise, I used the standard implementation of Gillespie's SSA for all simulations. When tau-leaping is used, I use the optimised algorithm [20] with an error control parameter of $\epsilon = 0.1$. For each run of the algorithm I set the sampling parameters $K_{\text{elite}} = 10$, $K_{\text{min}} = 1{,}000$, $K_{\text{max}} = 20{,}000$, and set an upper limit on the number of iterations to 250. The parameter smoothing constants, (λ, β, q), were set to (0.7, 0.8, 5) respectively — see Equations (3.6.3) and (3.6.4) for details. Logarithmic sampling was used for all estimates.

Mean Relative Error For each comparison between SPICE and the COPASI parameter inference approaches, I define the mean relative error (MRE) as

$$\text{MRE} = \frac{1}{M}\left(\sum_{j=1}^{M} |\hat{\theta}_j - \theta_j^*| \times 100\%\right),$$

where θ_j^* is the true j-th kinetic rate constant used to generate the synthetic data, and $\hat{\theta}_j$ is the j-th parameter estimate.

5.3 Lotka-Volterra Predator Prey Model

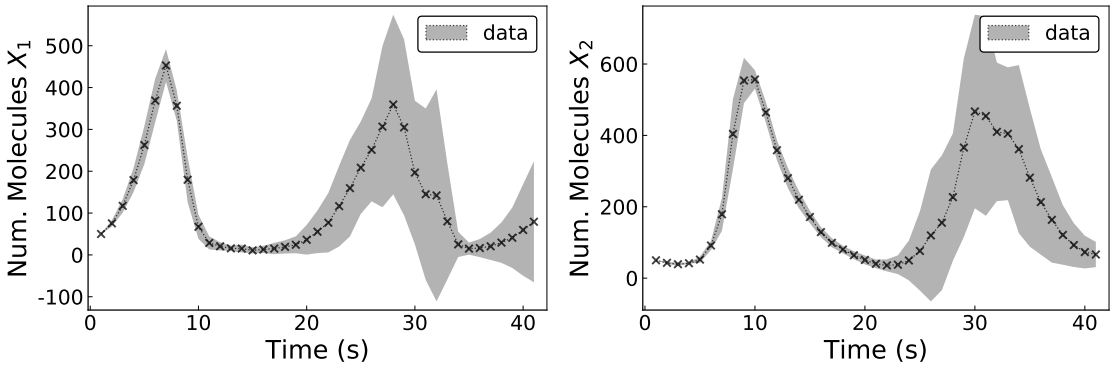
The Lotka-Volterra model is a well known approach to modelling the dynamics of ecological systems that are affected by: predator–prey interactions, resource competition amongst species, and disease. The system comprises two species: the prey, X_1, and the predators, X_2. In addition there are three interactions: (i) the numbers of prey grow, (ii) the predators consume a prey to increase their own population, and (iii) the predators die. I implement the standard Lotka-Volterra model below with parameters $(\theta_1, \theta_2, \theta_3) = (0.5,\ 0.0025,\ 0.3)$, and the initial population conditions $(X_1, X_2) = (50,\ 50)$.

$$\begin{aligned} X_1 &\xrightarrow{\theta_1} X_1 + X_1 \\ X_1 + X_2 &\xrightarrow{\theta_2} X_2 + X_2 \\ X_2 &\xrightarrow{\theta_3} \emptyset \end{aligned}$$

The stoichiometry matrices representing the reactants, $\boldsymbol{v}^-$, products, $\boldsymbol{v}^+$, and state-change matrix, $\boldsymbol{v}$, are given below:

$$\boldsymbol{v}^- = \begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{pmatrix}, \quad \boldsymbol{v}^+ = \begin{pmatrix} 2 & 0 \\ 0 & 2 \\ 0 & 0 \end{pmatrix}, \quad \boldsymbol{v} = \begin{pmatrix} 1 & 0 \\ -1 & 1 \\ 0 & -1 \end{pmatrix}.$$

For the purpose of parameter inference, I used Gillespie's SSA to artificially generate 5 datasets consisting of 40 time-points. The evolution of the species X_1 and X_2 are shown in Figure 5.1. It can be seen that the model exhibits the classic oscillatory behaviour. These same 5 datasets were given to SPICE and each of the parameter inference routines implemented within COPASI. For each method, I specify bounds on the parameter search space of $\theta_j \in [1\text{e}{-6}, 10]$, for $j = 1, 2, 3$.



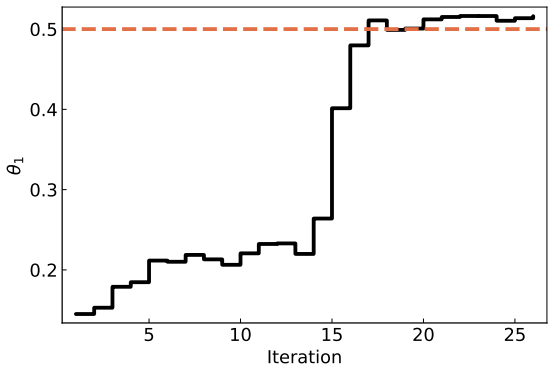
(a) Temporal evolution of species X_1 (b) Temporal evolution of species X_2

Figure 5.1: Lotka-Volterra Model: The temporal evolution of the mean of the 5 datasets (black markers and lines) is plotted across the time interval $[0, 40]$. The respective standard deviations (represented by grey ribbons) of the species, X_1 and X_2, within the dataset are plotted alongside the mean.

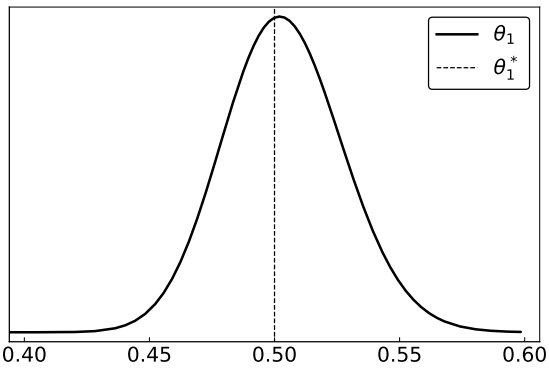
Firstly, in Table 5.1 I present the iteration-by-iteration breakdown (the output of 'trace.csv' — see Section 3.6.1) of one run of the SPICE algorithm. The breakdown contains the values of each parameter estimate, $\theta_1, \theta_2, \theta_3$, the cost function of the worst performing elite sample (γ_n), and the number of samples taken for each iteration. It can be seen that the estimated parameters (0.51610, 0.00263, 0.28618) converge within proximity to the true parameters (0.5, 0.0025, 0.3). In addition, the benefit of adaptive sampling can be seen as the algorithm converges toward the optimal solution. In Figures 5.2a, 5.2c and 5.2e, I present a graphical version of the iterative breakdown of the parameter estimates. In Figures 5.2b, 5.2d and 5.2f, I show the lognormal distribution for each of the parameters $\theta_1, \theta_2, \theta_3$ approximated by SPICE in the final iteration. It can be seen that the variances are small, and the returned means lie over the true parameter values — suggesting that SPICE can return both *accurate* and *precise* parameter estimates. In Figure 5.3, I plot the fitted model trajectories — of species X_1 and X_2, using the estimates returned from one example run of the SPICE algorithm (where $\hat{\boldsymbol{\theta}} = (0.501, 0.00257, 0.302)$). It can be seen that the means (solid lines) and the standard deviations (ribbons) of the fitted model closely match the 5 provided datasets. In Table 5.2, I present the minimum, maximum, and average mean relative errors (MRE) (%) between the estimated parameters and the true

Iteration	$\hat{\theta}_1$	$\hat{\theta}_2$	$\hat{\theta}_3$	**Score** $\hat{\gamma}_n$	**Num. Samples**
1	0.14502	0.00137	0.01279	3844486	1000
2	0.15287	0.00127	0.01440	3452173	1000
3	0.17894	0.00147	0.01719	3383899	1000
4	0.18465	0.00147	0.02005	3332785	1000
5	0.21160	0.00159	0.02421	3312482	1000
6	0.21011	0.00158	0.02789	3289019	1000
7	0.21866	0.00162	0.03184	3261923	1333
8	0.21311	0.00160	0.03695	3291406	1333
9	0.20633	0.00155	0.04107	3249333	2369
10	0.22061	0.00158	0.04963	3219241	2369
11	0.23229	0.00170	0.05555	3239783	2369
12	0.23298	0.00169	0.05899	3225140	2369
13	0.21988	0.00159	0.06399	3202140	5616
14	0.26397	0.00187	0.07648	3170840	5616
15	0.40140	0.00269	0.10459	3194675	5616
16	0.47969	0.00282	0.16154	2292086	5616
17	0.51081	0.00288	0.23915	910918	5616
18	0.49897	0.00276	0.28900	832071	5616
19	0.50066	0.00271	0.29667	698974	9984
20	0.51211	0.00265	0.29423	543723	9984
21	0.51508	0.00259	0.29232	485302	9984
22	0.51631	0.00262	0.28718	438338	17749
23	0.51627	0.00266	0.28978	331472	17749
24	0.51037	0.00261	0.29693	392420	20000
25	0.51359	0.00261	0.29044	337080	20000
26	0.51610	0.00263	0.28618	313594	20000
Real Val.	**0.5**	**0.0025**	**0.3**		

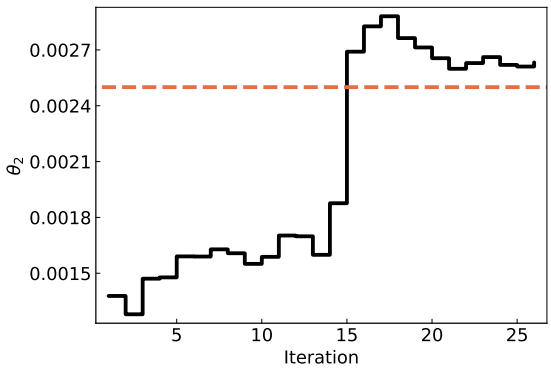
Table 5.1: Example output of a trace (.csv file) given for one run of SPICE on the Lotka-Volterra Predator-Prey model. Displayed are the estimates of the parameters $\theta_1, \theta_2, \theta_3$ for each iteration, alongside the score function (relative to the 5 datasets), and the number of samples taken at each iteration.



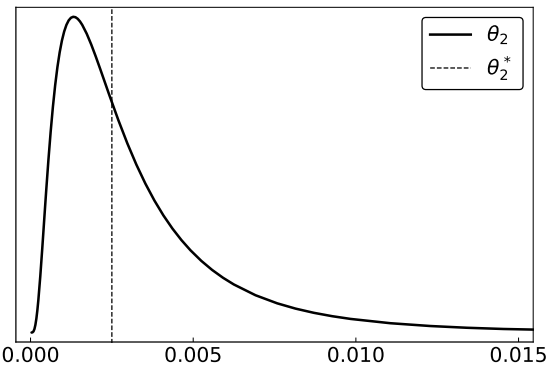
(a) Iteration-by-iteration estimates of θ_1.



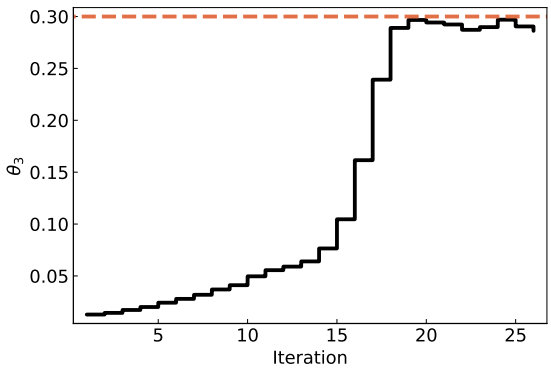
(b) Final estimated distribution of θ_1.



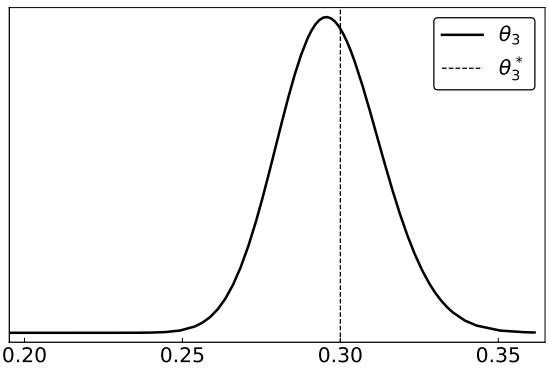
(c) Iteration-by-iteration estimates of θ_2.



(d) Final estimated distribution of θ_2.

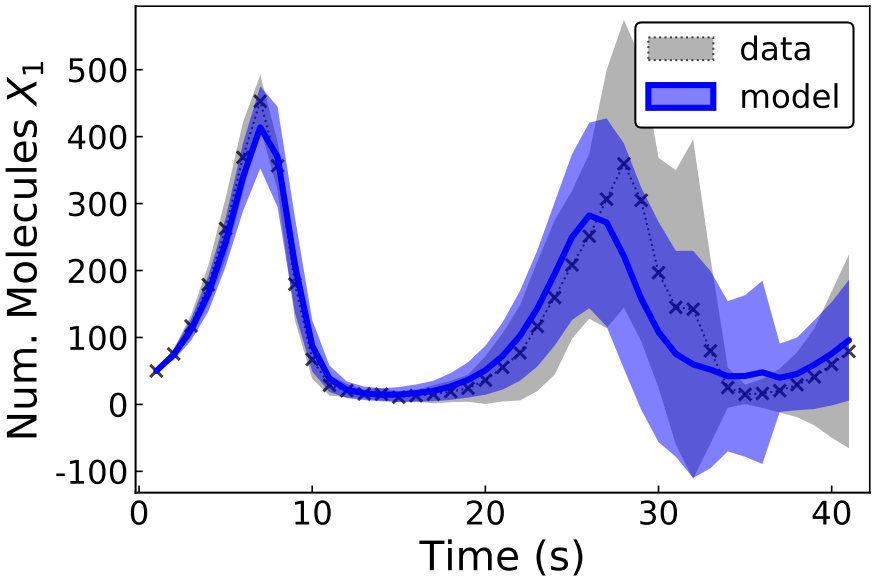


(e) Final estimated distribution of θ_3.

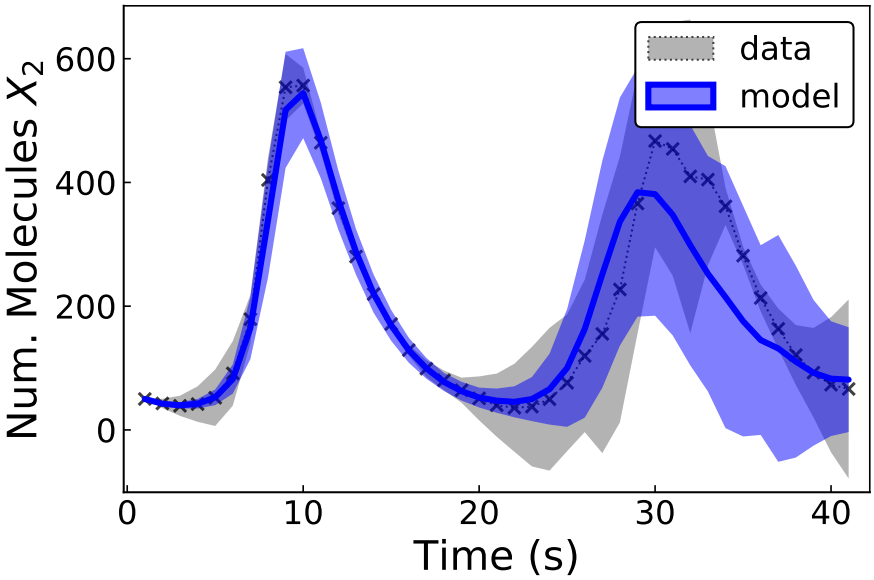


(f) Temporal evolution of species X_2

Figure 5.2: The returned parameter estimates for one run of the SPICE algorithm on the Lotka-Volterra predator-prey model. In Figures 5.2a, 5.2c and 5.2e, I present the iteration-by-iteration breakdown for the estimates of the 3 kinetic rate parameters. The dashed orange line represents the true parameter values. In Figures 5.2b, 5.2d and 5.2f, I show the obtained log-normal parameter distributions for each parameter after the end of the final iteration.



(a) Temporal evolution of species X_1



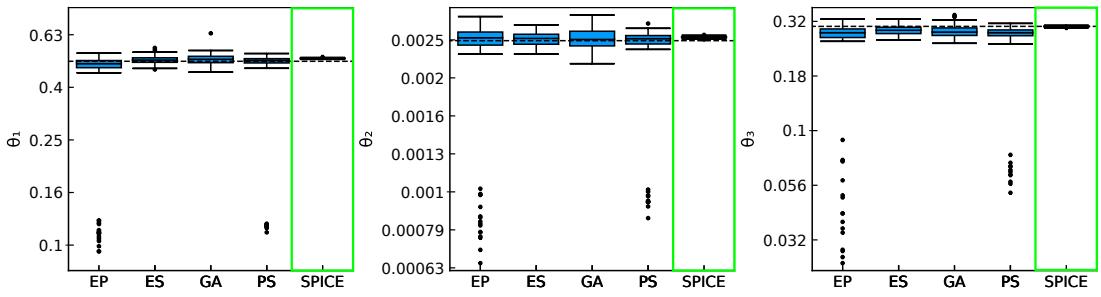
(b) Temporal evolution of species X_2

Figure 5.3: Lotka-Volterra Model: The temporal evolution of the mean of the 5 datasets (black markers and lines) is plotted across the time interval $[0, 40]$. The respective variances (represented by grey ribbons) of the species, X_1 and X_2, within the dataset are plotted alongside the mean. The fitted mean and variance of the fitted model for X_1 and X_2, using 1,000 SSA simulations, are represented by the blue lines and ribbons respectively. In this plot, the model was performed using the best fit parameter estimates from one run of the standard SPICE algorithm, $\hat{\boldsymbol{\theta}} = (0.501, 0.00257, 0.302)$.

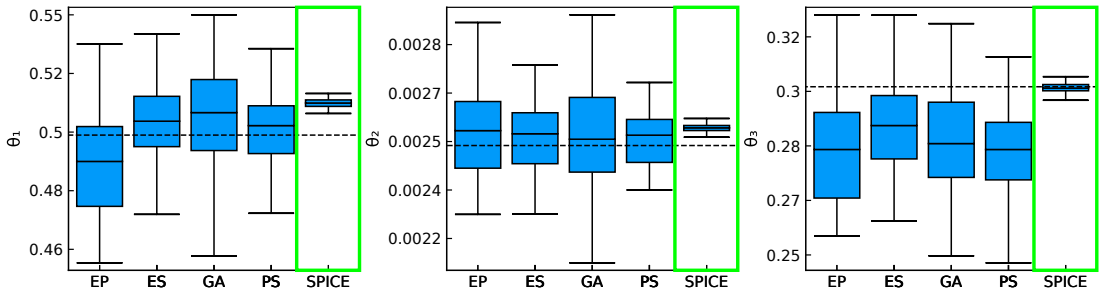
Lotka-Volterra Model							
Alg.	θ_1	θ_2	θ_3	Min. MRE	Av. MRE	Max. MRE	Av. CPU
	($\%_{\text{ERR}}$)			($\%_{\text{ERR}}$)	($\%_{\text{ERR}}$)	($\%_{\text{ERR}}$)	(s)
EP	38.4	3.5	29.6	0.4	23.8	156.5	1200
ES	3.8	0.6	4.4	0.3	3.0	9.0	5763
GA	5.2	0.8	5.7	0.8	3.9	15.2	3640
PS	25.6	2.2	18.6	**0.1**	15.5	126.6	2689
SPICE	**3.6**	**0.4**	**0.4**	1.0	**1.5**	**2.1**	**1025**

Table 5.2: Lotka-Volterra Model: Comparison of the relative errors of parameter estimates obtained using the COPASI implemented parameter inference algorithms *vs* SPICE. Each algorithm was ran 100 times using the exact SSA for simulation. Documented are the minimum, average, and maximum mean relative errors (MRE) across all 3 parameters, alongside the average computation time. The selected algorithms are Evolutionary Programming (EP), Evolution Strategy (ES), Genetic Algorithm (GA), and the Particle Swarm (PS).

parameters across 100 runs of each algorithm (in COPASI and SPICE). Also shown are the averaged CPU run times. It can be seen that the error of the parameter estimates across all algorithms varies, however, SPICE is *always* good — with no relative error larger than 2.1% for SPICE. The next best performing algorithm was the evolution strategy (ES), returning a maximum mean relative error (across all 3 parameters) of 9%. Other algorithms such as the evolutionary programming (EP) approach, produce an unpredictable variety of estimates, with mean relative errors from 0.4–156.5%. For this model, the computation time (single-core) was fastest for SPICE, coming in at under $\sim$ 20 minutes. The CPU time for the ES approach — which produced the next most accurate estimates — was $\sim 6\times$ greater than SPICE. In Figure 5.4 I display box plots summarising the distribution of the obtained parameter estimates across the 100 runs of each parameter inference method. It can be seen that, in general, all the algorithms can precisely estimate all 3 parameters. Both the evolutionary programming (EP) and particle swarm (PS) approaches contain numerous outliers. SPICE consistently has the most precise (least variance) estimates of all the methods.



(a) Box plots of the parameter estimates for $\theta_1, \theta_2, \theta_3$, including outliers.



(b) Box plots of the parameter estimates for $\theta_1, \theta_2, \theta_3$, excluding outliers.

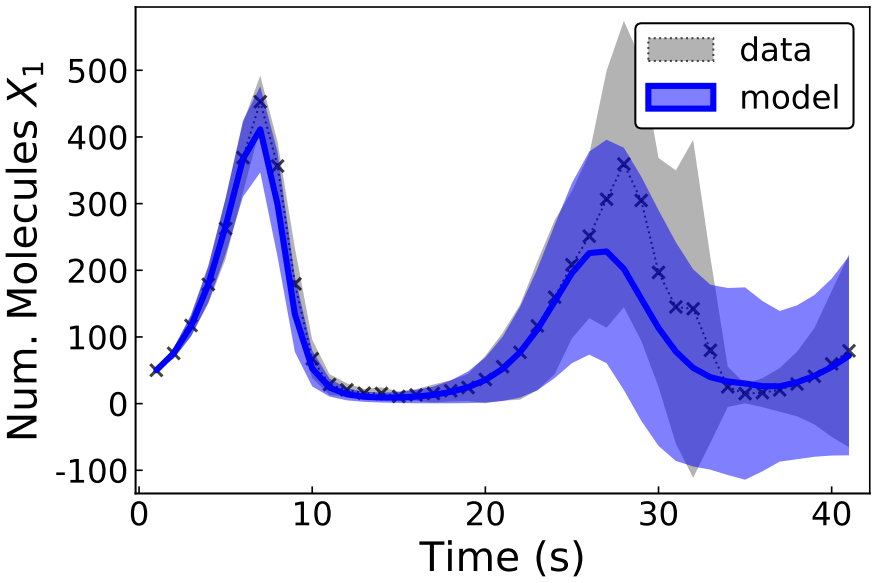
Figure 5.4: Box plots representing the parameter estimates obtained by SPICE and COPASI for the Lotka-Volterra model. Results are presented for the evolutionary programming (EP), evolutionary strategy (ES), genetic algorithm (GA), and particle swarm (PS) methods. Highlighted in green are the results of SPICE. It can seen that SPICE consistently has the least variance.

In Table 5.3, I present results for 100 runs of the alternative implementations of SPICE using (i) tau-leaping ($\epsilon = 0.1$) *vs.* the direct method, (ii) the multiple shooting approach, and (iii) the particle splitting approach. It can be seen that using the direct method without any of the tested heuristic retrieves the best parameter estimates. Furthermore, it can be seen that the heuristics do not improve the obtained parameter estimates, except for the case where tau-leaping is used. Specifically, the second best performing approach was the use of tau-leaping in conjunction with the particle splitting approach, with a maximum mean relative error of 5%. The use of tau-leaping improves the single-core computation time $\sim 3\times$. Lastly, I also examine the Lotka-Volterra predator-prey example for the case where there is an unobserved (latent) species. In this test case, SPICE

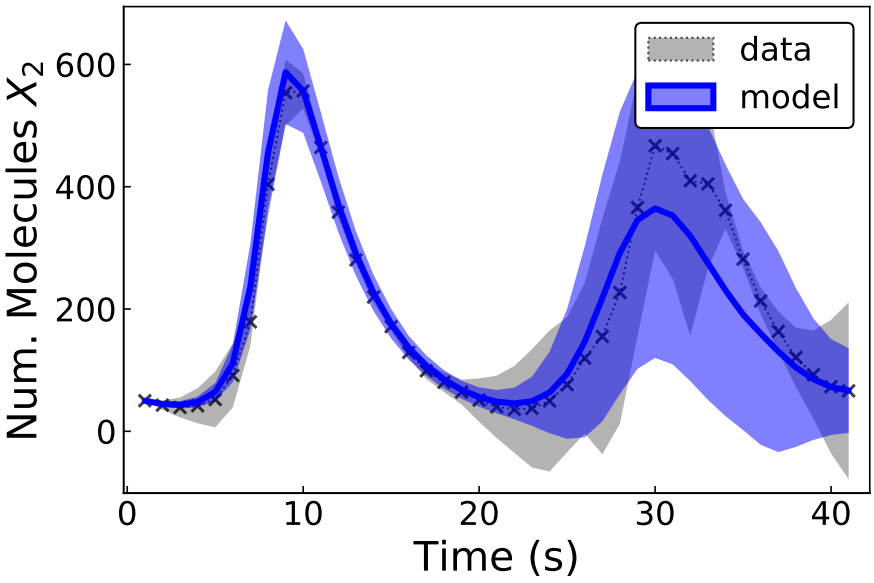
Lotka-Volterra Model				
	Min. MRE $\%_{\text{ERR}}$	Av. MRE $\%_{\text{ERR}}$	Max. MRE $\%_{\text{ERR}}$	Av. CPU (s)
DM	**0.2**	**1.2**	**2.7**	983
DM + MS	1.6	6.2	12.2	1368
DM + SP	**0.2**	2.5	46.6	1171
Tau	0.6	19.7	221.4	329
Tau + MS	0.5	5.7	10.9	432
Tau + SP	0.4	2.3	5.0	**303**

Table 5.3: Lotka-Volterra Model: The averaged mean relative errors (MRE) for all 3 parameter estimates across 100 runs of the SPICE algorithm using the multiple shooting (+MS), and particles splitting (+PS) routines. In addition, the impact of using the direct method (DM) *vs.* tau-leaping (Tau), $\epsilon = 0.1$, is assessed.

was only provided with the snapshot time-series data for species X_1. For fitting the model, X_2 instead had to be inferred from the estimated parameters using only species X_1. The results for 100 runs of SPICE on the unobserved data case is presented in Table 5.4. It can be seen that despite missing 50% of the data, SPICE still produces good parameter estimates, with no error greater than 10% on any one of the 3 parameters. In addition, the averaged mean relative errors even outperform most of the COPASI algorithms, despite the removal of data. In Figure 5.5, I present fitted model trajectories using the obtained parameter estimates for one run of the algorithm on the unobserved data case. The results show that the fitted model is still able to closely match the mean and variance of the data.



(a) Temporal evolution of species X_1



(b) Temporal evolution of species X_2

Figure 5.5: Lotka-Volterra (with X_2 as a latent species): The mean of the data (black markers) and the respective variances of the species X_1 and X_2 are plotted alongside the mean and variance of 1,000 SSA simulations (blue lines and ribbons) performed using the parameter estimates from one run of SPICE ($\boldsymbol{\theta} = (0.536, 0.00257, 0.271)$). The data for X_2 (shown for illustration) was not provided to the SPICE algorithm, and treated as an unobserved latent species.

Lotka-Volterra Model (with X_2 unobserved)					
Alg.	θ_1 θ_2 θ_3 ($\%_{\text{ERR}}$)	Min. MRE ($\%_{\text{ERR}}$)	Av. MRE ($\%_{\text{ERR}}$)	Max. MRE ($\%_{\text{ERR}}$)	Av. CPU (s)
SPICE	9.4 0.41 6.4	4.1	5.4	6.8	1589

Table 5.4: Lotka-Volterra Model (with X_2 unobserved): Results of the parameter estimates obtained using 100 runs of SPICE. Shown are the mean relative errors (MRE) and the average CPU time.

5.4 Yeast Polarization Model

The second model I analyse is a Yeast Polarization model. The model accounts for the temporal evolution of the G-protein cycle in budding yeast, *Saccharomyces cerevisiae*, and has been simplified to include the following species: ligands (L), receptors (R), receptor-ligand complexes (RL), and G-protein subunits (G_a, G_d, G_{bg}). The model species and their interactions are shown graphically in Figure 5.6. The real parameters of the model are $(\theta_1, \ldots, \theta_8) = (0.38,$

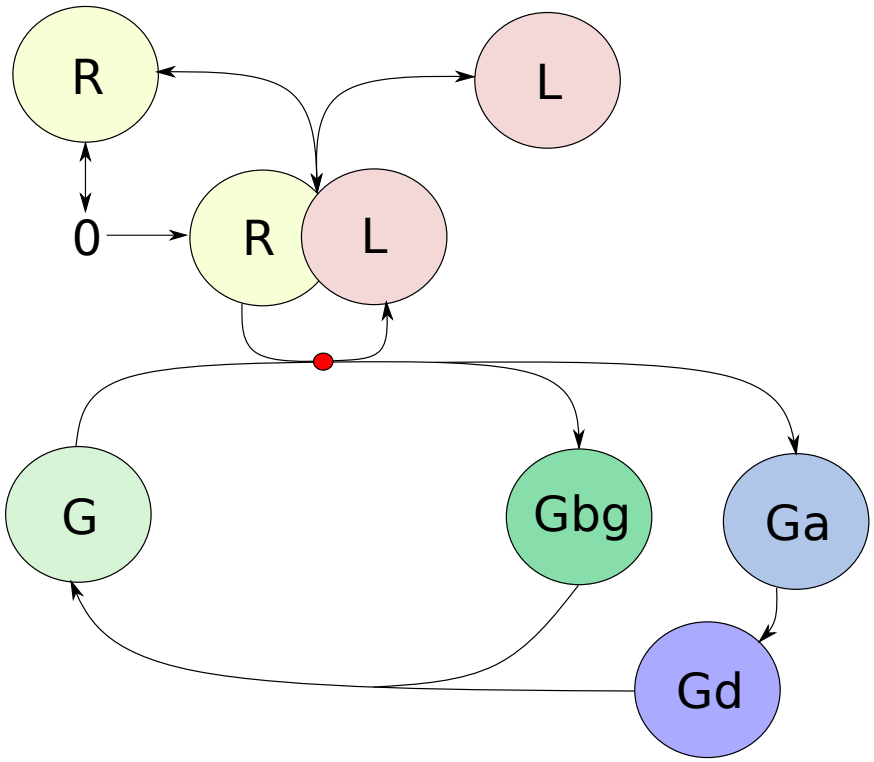


Figure 5.6: A graphical representation of the Yeast Polarization Model. Included are the receptors (R), the ligands (L), their respective complex (RL), and the G protein cycle including the subunits Gbg, Ga, and Gd.

$0.04,\ 0.082,\ 0.12,\ 0.021,\ 0.1,\ 0.005,\ 13.21)$, and initial species population is $(R, L, RL, G, G_a, G_{bg}, G_d) = (500,\ 4,\ 110,\ 300,\ 2,\ 20,\ 90)$. The reaction equa-

tions of the model are [26]:

$$\begin{aligned} \emptyset &\xrightarrow{\theta_1} R & \qquad RL + G &\xrightarrow{\theta_5} G_a + G_{bg} \\ R &\xrightarrow{\theta_2} \emptyset & G_a &\xrightarrow{\theta_6} G_d \\ L + R &\xrightarrow{\theta_3} RL + L & G_d + G_{bg} &\xrightarrow{\theta_7} G \\ RL &\xrightarrow{\theta_4} R & \emptyset &\xrightarrow{\theta_8} RL \end{aligned}$$

For the purpose of parameter inference, I generated 5 datasets consisting of 17 time-points using Gillespie's SSA, and again provided each algorithm (in COPASI and SPICE) the same data. For the initial parameter search space, I place bounds of $\theta_j \in [1e{-}6, 10]$ for $1 \leqslant j \leqslant 7$, and $\theta_8 \in [1e{-}6, 100]$ across all methods.

Firstly, in Table 5.5 I present the iteration-by-iteration breakdown (the output of 'trace.csv' — see Section 3.6.1) of one run of the SPICE algorithm for the yeast polarization model (in this case, using a starting number of 10,000 samples). The breakdown contains the values of each parameter estimate, $\theta_1, \ldots, \theta_8$, the cost function of the worst performing elite sample (γ_n), and the number of samples taken for each iteration. It can be seen that each parameter estimate (0.34696, 0.05728, 0.08981, 0.34202, 0.02039, 0.09714, 0.00461, 14.68639) converge within very close proximity to the true parameters (0.38, 0.04, 0.082, 0.12, 0.021, 0.1, 0.005, 13.21). In Figure 5.7, I plot the fitted model trajectories of the 6 species $R, RL, G, G_a, G_{bg}, G_d$ using the estimates returned from one example run of the SPICE algorithm where $\boldsymbol{\theta}$ = (2.54, 0.0561, 0.0858, 0.198, 0.0210, 0.125, 0.00533, 17.0). The fitted models were produced by 1,000 SSA simulations. It can be seen that the means (solid lines) and the standard deviations (ribbons) of the fitted model closely match the 5 provided datasets. The species L is not plotted, as it is conserved for all reactions.

In Table 5.6, I present the minimum, maximum, and average mean relative errors (MRE) (%) between the estimated parameters and the true parameters across 100 runs of each algorithm (in COPASI and SPICE). Also shown are the averaged CPU run times. As is the case in the Lotka-Volterra predator prey model (Section 5.3), it can be noted that the errors of the parameter estimates across all algorithms varies greatly. Once again, SPICE provides the least-variance in its obtained estimates. For parameters $\theta_1, \theta_2, \theta_3$, and θ_8, the evolutionary strategy (ES) provides the best estimates. For parameters $\theta_4, \theta_5, \theta_6$, and θ_7, SPICE

produces the most accurate estimates, with some errors less than 2% (for each of $\theta_5, \theta_6, \theta_7$). In general, the genetic algorithm (GA), and the evolutionary programming approaches (EP) perform poorly. The CPU times across all parameter inference methods are very comparable at $\sim$ 20-30 minutes. In Figure 5.8 I present box plots for the summary statistics of the first 4 kinetic rate parameter estimates, $\theta_1, \theta_2, \theta_3, \theta_4$, across the 100 runs of each parameter inference method. In Figure 5.9 I present box plots for the summary statistics of the remaining 4 kinetic rate parameter estimates, $\theta_5, \theta_6, \theta_7, \theta_8$. Once again, it can be seen that on the whole, the majority of the parameters are well approximated by the inference methods. As before, SPICE produces estimates with the lowest variance, and greatest *precision*. The other best performing technique is again the evolutionary strategy (ES). It should be noted that the scale of the y-axis is logarithmic, revealing the vast discrepancy between estimates produced from the EP, GA, and PS methods.

Finally, in Table 5.7, I present results for 100 runs of the alternative implementations of SPICE using (i) tau-leaping ($\epsilon = 0.1$) *vs.* the direct method, (ii) the multiple shooting approach, and (iii) the particle splitting approach. Unlike the Lotka-Volterra model, the mean relative errors of the parameter estimates ($\sim 41\%$) for the Yeast Polarization model remain relatively unchanged regardless of the heuristic approach taken. Interestingly, the best estimates were obtained using tau-leaping in conjunction with the particle splitting method, again with a reduction in CPU (single-core) time of $\sim 3\times$. However, it should noted that all the parameter estimates are very close, and this could be down to chance, or bias of the simulations.

The Yeast Polarization model was also studied in [26] — where the authors implemented the MCEM^2 algorithm, which combined a minimal cross-entropy step with the well-known expectation-maximisation algorithm. MCEM^2 is the routine behind parameter inference within StochSS and StochKit [32]. Within [26], the authors reported that the MCEM^2 algorithm retrieved a MRE of 34.7% (based on one run), after a simulation time of $\sim$ 30 days. SPICE, presented within this thesis, achieved an average MRE of 43.3%, and overall minimum MRE of 27.6% for the yeast model. The average computation time was $\sim$ 20 minutes. The difference in computation time, other than implementation-specific reasons, comes

down to the computational burden of the expectation-maximisation algorithm. The EM algorithm requires heavy calculations of the systems likelihood, and also requires more simulations to produce *exact* trajectories. SPICE on the other hand circumvents the need to produce entirely consistent trajectories during its search phase.

Iteration	$\hat{\theta}_1$	$\hat{\theta}_2$	$\hat{\theta}_3$	$\hat{\theta}_4$	$\hat{\theta}_5$	$\hat{\theta}_6$	$\hat{\theta}_7$	$\hat{\theta}_8$	**Score** $\hat{\gamma}_n$	**Samples**
1	0.30980	0.05439	0.09814	0.70634	0.03037	0.05476	0.00342	11.13968	109463	10000
2	0.32035	0.05773	0.09870	0.60884	0.02109	0.06236	0.00372	10.98718	13338	10000
3	0.33282	0.06199	0.09537	0.50139	0.01927	0.06729	0.00378	10.56167	8141	10000
4	0.32444	0.06433	0.09389	0.47487	0.01922	0.07003	0.00383	10.67949	7039	10000
5	0.30990	0.06405	0.09236	0.44298	0.01932	0.07270	0.00387	10.67308	6560	10000
6	0.30119	0.06198	0.09217	0.42092	0.01943	0.07766	0.00402	10.95597	5984	10000
7	0.36016	0.06146	0.09185	0.40835	0.01927	0.08076	0.00408	11.29595	5670	10000
8	0.39162	0.05933	0.09205	0.39847	0.01939	0.08367	0.00416	11.79631	5208	10000
9	0.40393	0.05968	0.09171	0.38676	0.01964	0.08543	0.00419	11.86940	5153	17777
10	0.43008	0.05942	0.09147	0.38730	0.01982	0.08753	0.00429	12.49010	4889	17777
11	0.39261	0.05974	0.09108	0.37672	0.01989	0.08772	0.00434	12.75959	4699	17777
12	0.35909	0.05879	0.09026	0.36477	0.02001	0.08935	0.00439	13.13580	4839	17777
13	0.34793	0.05923	0.09086	0.36803	0.02009	0.09087	0.00444	13.53557	4587	17777
14	0.35902	0.05842	0.09063	0.36589	0.02012	0.09323	0.00446	13.89675	4541	17777
15	0.33620	0.05864	0.09064	0.36477	0.02023	0.09442	0.00455	14.24144	4616	20000
16	0.33728	0.05942	0.08930	0.34900	0.02031	0.09442	0.00457	14.68026	4425	20000
17	0.34696	0.05728	0.08981	0.34202	0.02039	0.09714	0.00461	14.68639	4700	20000
Real Val.	**0.38**	**0.04**	**0.082**	**0.12**	**0.021**	**0.1**	**0.005**	**13.21**		

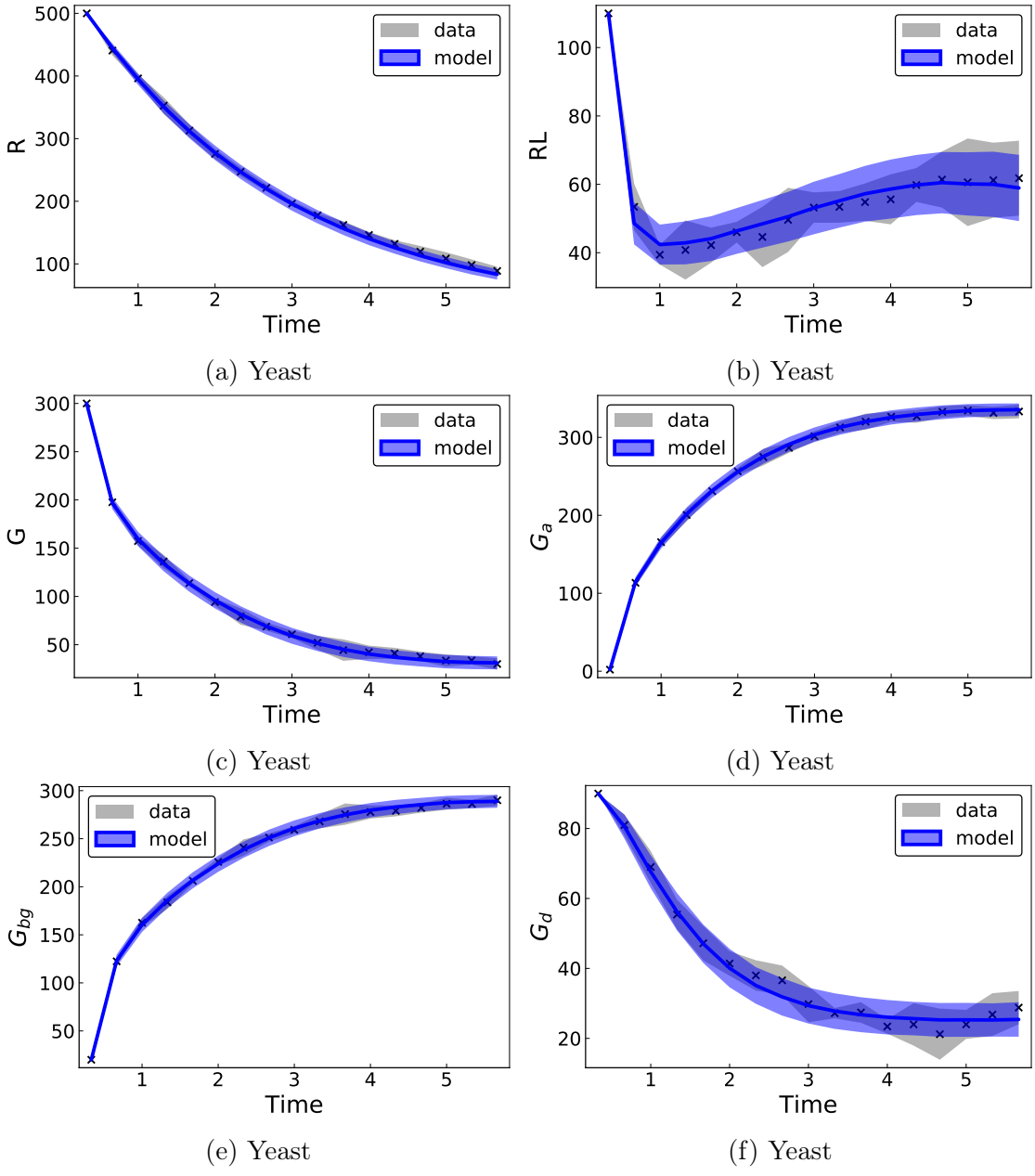
Table 5.5: Example output of a trace (.csv file) given for one run of SPICE on the Yeast Polarization model. Displayed are the estimates of the parameters $\theta_1, \ldots, \theta_8$ for each iteration, alongside the score function (relative to the 5 datasets), and the number of samples taken at each iteration.

Yeast Polarization Model												
Alg.	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8	Min. MRE	Av. MRE	Max. MRE	Av. CPU
	($\%_{\text{ERR}}$)								($\%_{\text{ERR}}$)	($\%_{\text{ERR}}$)	($\%_{\text{ERR}}$)	(s)
EP	662.9	138.4	1.7	235.4	1.7	25.3	3.4	357.0	56.2	178.2	316.9	**405**
ES	**109.8**	**18.5**	**1.2**	35.4	1.3	3.3	1.5	**27.9**	**3.6**	**24.9**	62.8	1650
GA	564.0	120.2	1.3	275.3	1.6	6.5	2.6	312.4	38.8	160.5	299.4	2696
PS	156.4	29.0	1.4	52.6	**0.9**	3.7	1.6	48.6	7.3	36.8	173.6	1755
SPICE	221.2	21.7	2.5	**34.9**	**0.9**	**1.7**	**1.1**	62.7	27.6	43.3	**54.4**	1116

Table 5.6: Yeast Polarization Model: Comparison of the relative errors of parameter estimates obtained using the COPASI implemented parameter inference algorithms *vs.* SPICE. Each algorithm was ran 100 times using Gillespie's SSA for simulation. Documented are the minimum, average, and maximum mean relative errors (MRE) across all 8 parameters, alongside the average computation time. The selected algorithms are Evolutionary Programming (EP), Evolution Strategy (ES), Genetic Algorithm (GA), and the Particle Swarm (PS).

Yeast Polarization Model				
	Min. MRE $\%_{\text{ERR}}$	Av. MRE $\%_{\text{ERR}}$	Max. MRE $\%_{\text{ERR}}$	Av. CPU (s)
DM	27.6	43.3	54.4	1,116
DM + MS	28.1	41.9	51.3	1834
DM + SP	31.9	43.1	57.4	1183
Tau	31.2	41.5	56.3	**303**
Tau + MS	40.5	49.9	60.5	445
Tau + SP	**25.7**	**41.0**	**47.7**	330

Table 5.7: Yeast Polarization Model: The averaged mean relative errors (MRE) for all 8 parameter estimates across 100 runs of the SPICE algorithm using the multiple shooting (+MS), and particles splitting (+PS) routines. In addition, the impact of using the direct method (DM) *vs.* tau-leaping (Tau), $\epsilon = 0.1$, is assessed.

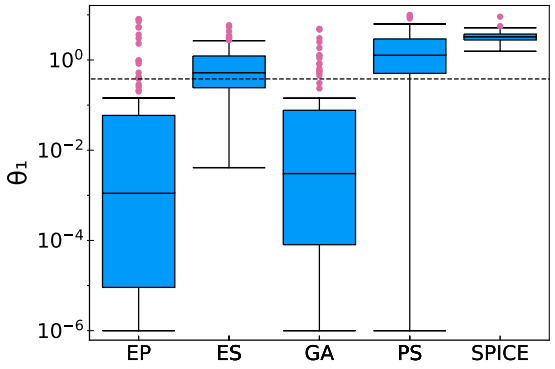


(a) Yeast (b) Yeast

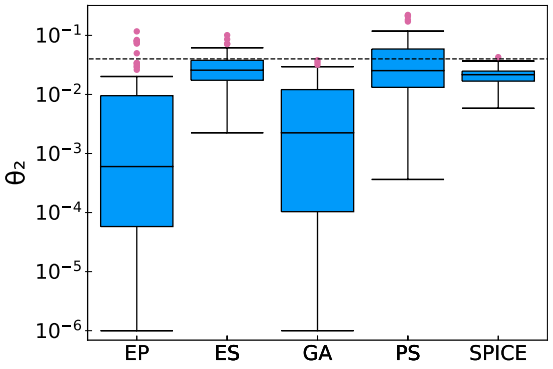
(c) Yeast (d) Yeast

(e) Yeast (f) Yeast

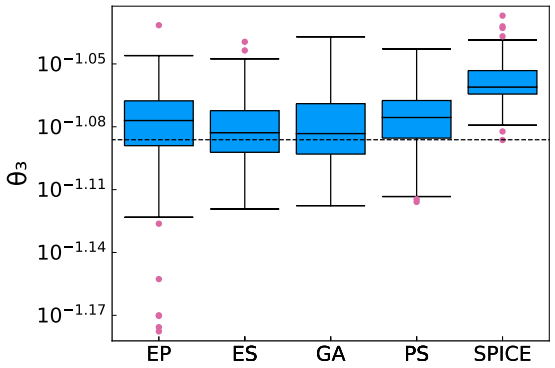
Figure 5.7: Yeast Polarization: The mean of the data (black markers) and the respective standard deviations (ribbons) of the species R, RL, G, G_a, G_{bg} and G_d are plotted alongside the mean and standard error of 1,000 SSA simulations (blue lines and ribbons) performed using the parameter estimates from one run of the standard SPICE algorithm with Gillespie's direct method. The obtained parameters were $\boldsymbol{\theta} = (2.54, 0.0561, 0.0858, 0.198, 0.0210, 0.125, 0.00533, 17.0)$. Species L is always conserved, so it has not been plotted.



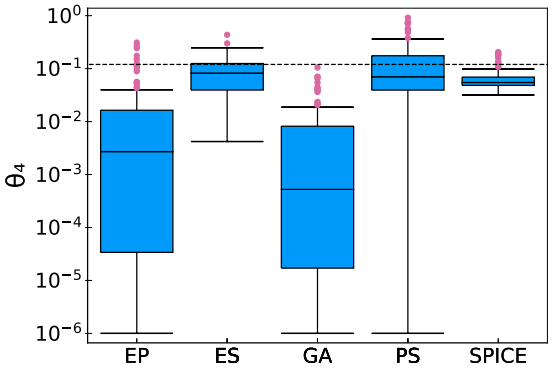
(a) Boxplots of obtained estimates for θ_1.



(b) Boxplots of obtained estimates for θ_2.

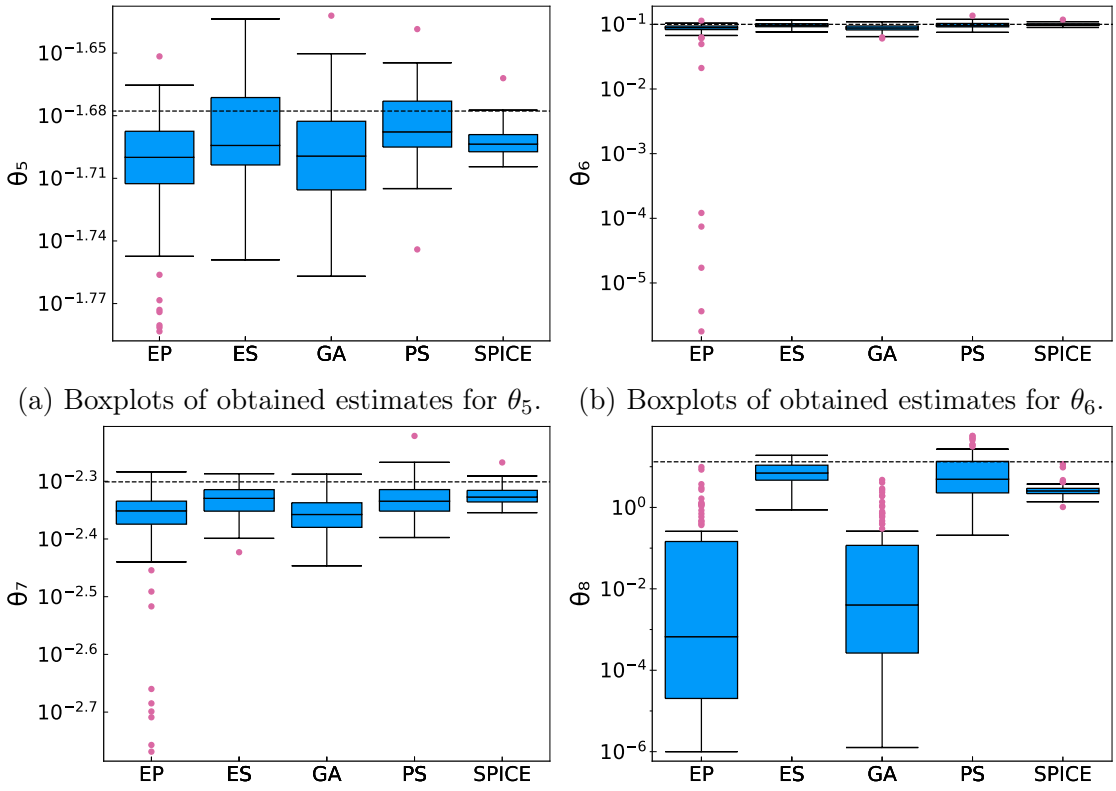


(c) Boxplots of obtained estimates for θ_3.



(d) Boxplots of obtained estimates for θ_4.

Figure 5.8: Boxplots comparing the distribution of parameter estimates for $\theta_1, \theta_2, \theta_3, \theta_4$ obtained by SPICE *vs.* the COPASI parameter inference methods. Each algorithm was ran 100 times using Gillespie's SSA for simulation. The selected algorithms are Evolutionary Programming (EP), Evolution Strategy (ES), Genetic Algorithm (GA), and the Particle Swarm (PS). Outlier results are represented by pink markers.



(a) Boxplots of obtained estimates for θ_5.

(b) Boxplots of obtained estimates for θ_6.

(c) Boxplots of obtained estimates for θ_7.

(d) Boxplots of obtained estimates for θ_8.

Figure 5.9: Boxplots comparing the distribution of parameter estimates for $\theta_5, \theta_6, \theta_7, \theta_8$ obtained by SPICE *vs.* the COPASI parameter inference methods. Each algorithm was ran 100 times using Gillespie's SSA for simulation. The selected algorithms are Evolutionary Programming (EP), Evolution Strategy (ES), Genetic Algorithm (GA), and the Particle Swarm (PS). Outlier results are represented by pink markers.

5.5 Schlögl System

For the third case study, I implement the (reduced) Schlögl model [114] with kinetic rate parameters $(\theta_1, \theta_2, \theta_3, \theta_4) = (3\text{e}{-7}, 1\text{e}{-4}, 1\text{e}{-3}, 3.5)$, and the initial population conditions $(X, A, B) = (250, 1\text{e}5, 2\text{e}5)$. Stochastically, the model is challenging and well studied due to its ability to produce bistable dynamics, for example, as in Figure 5.10. The system is modelled by the following four reactions:

$$2X + A \xrightarrow{\theta_1} 3X \qquad B \xrightarrow{\theta_3} X$$
$$3X \xrightarrow{\theta_2} 2X + A \qquad X \xrightarrow{\theta_4} B$$

To simplify the model, a common practice is to assume that the populations of the species A and B remain fixed, acting as a non-depleting reservoir. For the purpose of parameter inference, I artificially generate 10 synthetic datasets. The motivation behind using more datasets than the previous models (Sections 5.3 and 5.4) is to partially capture a degree of the bistable dynamics. Each datasets consists of 100 time-points, and the same 10 datasets were provided to all the algorithms in COPASI, alongside SPICE. For the initial iteration, I placed bounds on the parameter search space for all the methods of $\theta_1 \in [1\text{e}{-9}, 1\text{e}{-5}]$, $\theta_2 \in [1\text{e}{-6}, 0.01]$, $\theta_3 \in [1\text{e}{-5}, 10]$, $\theta_4 \in [0.01, 100]$. Unlike the previous case studies, I explicitly ran the Schlögl System using tau-leaping for *all* inference algorithms, due to the computation time being largely infeasible under the same conditions — taking 4.5 hours in SPICE, and 48+ hours in COPASI for just one run (with Gillespie's direct method). The reason for this is that the Schlögl system contains higher molecular counts, and a dense number of reactions over the time course — negatively impacting the simulation time with the direct method.

Firstly, in Table 5.8 I present the iteration-by-iteration breakdown (the output of 'trace.csv' — see Section 3.6.1) of one run of the SPICE algorithm for the Schlögl system model using tau-leaping with an error control parameter of $\epsilon = 0.1$ (in this case, using a starting number of 10,000 samples). The breakdown in Table 5.8 contains the values of each parameter estimate, $\theta_1, \theta_2, \theta_3, \theta_4$, the cost function of the worst performing elite sample (γ_n), and the number of samples taken for each iteration. It can be seen that the returned parameter estimates are worse than in Sections 5.3 and 5.4, but of the right order of

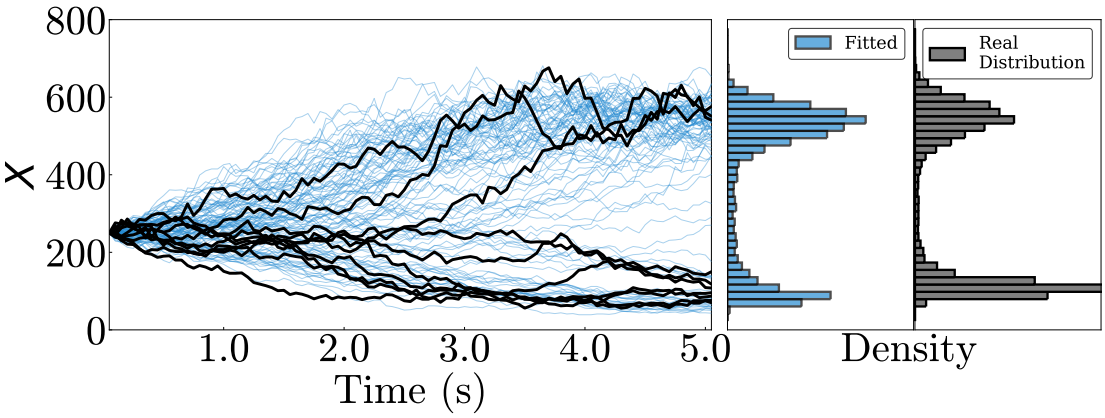


Figure 5.10: Schlögl: From the left: Solid black lines: the 10 datasets generated using the SSA direct method and the real parameters, and used as input for SPICE. Blue lines: 100 model runs with estimated parameters sampled by the final parameter distributions obtained by SPICE using Gillespie's direct method. The parameters means were (2.14e−7, 7.63e−5, 4.54e−4, 2.18); and the variances were (7.81e−16, 2.81e−10, 4.05e−8, 0.13). Fitted: empirical distribution of 1,000 model simulations with sampled parameters from SPICE output. Real distribution: empirical distribution of 1,000 model simulations with the real parameters.

magnitude. In Figure 5.10, I plot the fitted model trajectories of the species X using the estimates returned from one example run of the SPICE algorithm where (2.14e−7, 7.63e−5, 4.54e−4, 2.18). Also plotted are the individual datasets (black lines), and histograms representing the empirical distribution of X at t=10s (generated by producing 1,000 additional datasets). Figure 5.10 shows that SPICE finds estimates which are able to reproduce the bimodal behaviour of the system to a very good degree — considering the limitations of the snapshot time-series data. In Table 5.9, I present the minimum, maximum, and average mean relative errors (MRE) (%) between the estimated parameters and the true parameters across 100 runs of each algorithm (in COPASI and SPICE), alongside the averaged (single-core) CPU run times. The mean relative errors are across all parameter inference methods are fairly good, with SPICE and the evolutionary strategy (ES) once again coming out on top. For $\theta_1, \theta_2, \theta_3$, the estimates of the kinetic rate parameters obtained by SPICE have an error of less than 15% (less than 20% for the ES). The CPU times are comparable, with SPICE taken $\sim$ 20 minutes, the ES taking $\sim$ 30 minutes. The evolutionary programming (EP)

Iteration	$\hat{\theta}_1$	$\hat{\theta}_2$	$\hat{\theta}_3$	$\hat{\theta}_4$	**Score** $\hat{\gamma}_n$	**Samples**
1	0.000000071	0.0000670	0.000493	0.42	64593314	10000
2	0.000000083	0.0000434	0.000511	0.82	52834621	10000
3	0.000000111	0.0000366	0.000388	1.19	31660966	10000
4	0.000000104	0.0000295	0.000333	1.18	28341940	10000
5	0.000000125	0.0000381	0.000258	1.32	15650099	10000
6	0.000000146	0.0000480	0.000242	1.49	13781427	10000
7	0.000000153	0.0000539	0.000213	1.50	9446324	10000
8	0.000000155	0.0000539	0.000220	1.50	7619867	10000
9	0.000000157	0.0000557	0.000239	1.52	7296086	10000
10	0.000000162	0.0000574	0.000211	1.53	6704515	10000
11	0.000000158	0.0000570	0.000194	1.49	6383915	10000
12	0.000000161	0.0000576	0.000182	1.50	6651584	10000
13	0.000000159	0.0000572	0.000171	1.47	6368436	17770
14	0.000000156	0.0000559	0.000154	1.45	6094335	17770
15	0.000000154	0.0000552	0.000163	1.42	5952402	17770
16	0.000000156	0.0000568	0.000138	1.42	5766036	17770
17	0.000000155	0.0000559	0.000129	1.42	5732842	17770
18	0.000000160	0.0000575	0.000145	1.46	5634765	17770
19	0.000000160	0.0000575	0.000145	1.46	5646560	20000
20	0.000000161	0.0000580	0.000146	1.46	5526336	20000
21	0.000000162	0.0000581	0.000171	1.48	5471566	20000
22	0.000000162	0.0000581	0.000180	1.50	5471514	20000
23	0.000000164	0.0000595	0.000179	1.50	5415429	20000
24	0.000000169	0.0000613	0.000183	1.56	5225612	20000
25	0.000000169	0.0000613	0.000183	1.56	5466029	20000
Real Val.	**.0000003**	**0.0001**	**0.001**	**3.5**		

Table 5.8: Example output of a trace (.csv file) given for one run of SPICE on the Schlögl system model. Displayed are the estimates of the parameters $\theta_1, \theta_2, \theta_3, \theta_4$ for each iteration, alongside the score function (relative to the 10 datasets), and the number of samples taken at each iteration.

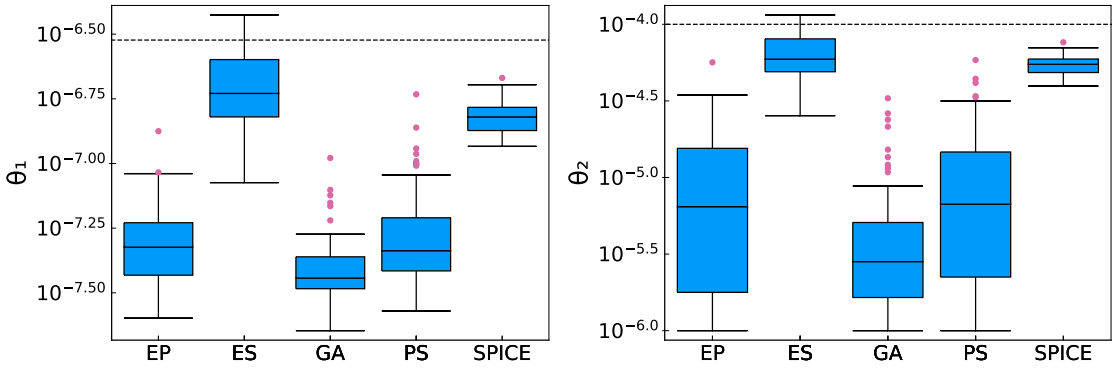
approach has the fastest computations at just under 10 minutes.

In Figure 5.11, I present the box plots summarising the distribution statistics of the parameter estimates obtained over the 100 runs of each of the inference algorithms. Once again, SPICE has the most *precise* estimates with small variances, and only SPICE and the ES approach produce satisfactory estimates on average. The EP, GA, and PS approaches produce estimates which span over several orders of magnitude.

In addition, it is well known that the Schlögl system dynamics are very sensitive to the initial conditions — where even slight perturbations of its parameters can cause the system to fail in producing bimodality. In Figure 5.12, I provide fitted model distributions using the best obtained parameter estimates from each inference algorithm. Interestingly, only SPICE, the ES and PS approaches are able to recover the bimodal dynamics. Furthermore, it is clear that SPICE does the best job at reconstructing the real data distribution.

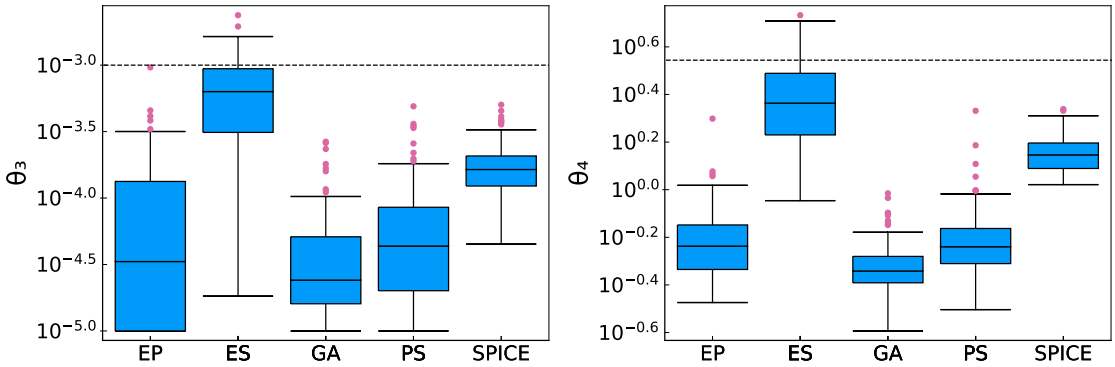
Schlögl System Model								
Alg.	θ_1	θ_2	θ_3	θ_4	Min. MRE	Av. MRE	Max. MRE	Av. CPU
	($\%_{\text{ERR}}$)				($\%_{\text{ERR}}$)	($\%_{\text{ERR}}$)	($\%_{\text{ERR}}$)	(s)
EP	12.2	9.7	15.1	142.9	24.4	45.0	60.5	**307**
ES	**3.3**	15.5	19.0	**40.3**	**11.5**	**19.3**	31.7	1505
GA	13.7	11.0	14.0	159.7	32.2	49.6	66.3	987
PS	12.0	**8.5**	11.4	141.4	18.7	43.3	60.0	1095
SPICE	4.6	14.6	**6.3**	73.0	18.5	24.6	**30.9**	1054

Table 5.9: Schlögl System Model: Comparison of the relative errors of parameter estimates obtained using the COPASI implemented parameter inference algorithms *vs.* SPICE. Each algorithm was ran 100 times using Tau-leaping ($\epsilon = 0.1$) for simulation. Documented are the minimum, average, and maximum mean relative errors (MRE) across all 4 parameters, alongside the average computation time. The selected algorithms are Evolutionary Programming (EP), Evolution Strategy (ES), Genetic Algorithm (GA), and the Particle Swarm (PS).



(a) Boxplots of obtained estimates for θ_1.

(b) Boxplots of obtained estimates for θ_2.



(c) Boxplots of obtained estimates for θ_3.

(d) Boxplots of obtained estimates for θ_4.

Figure 5.11: Schlögl System parameter estimates: box plots comparing the parameter estimates across 100 runs of COPASI's methods and SPICE (all simulated using tau-leaping, $\epsilon = 0.1$). Again, SPICE shows the smallest variance, with mean estimates quite close to the real values of θ_1 and θ_3. For θ_2 and θ_4, all the best mean estimates have variance much larger than SPICE estimates.

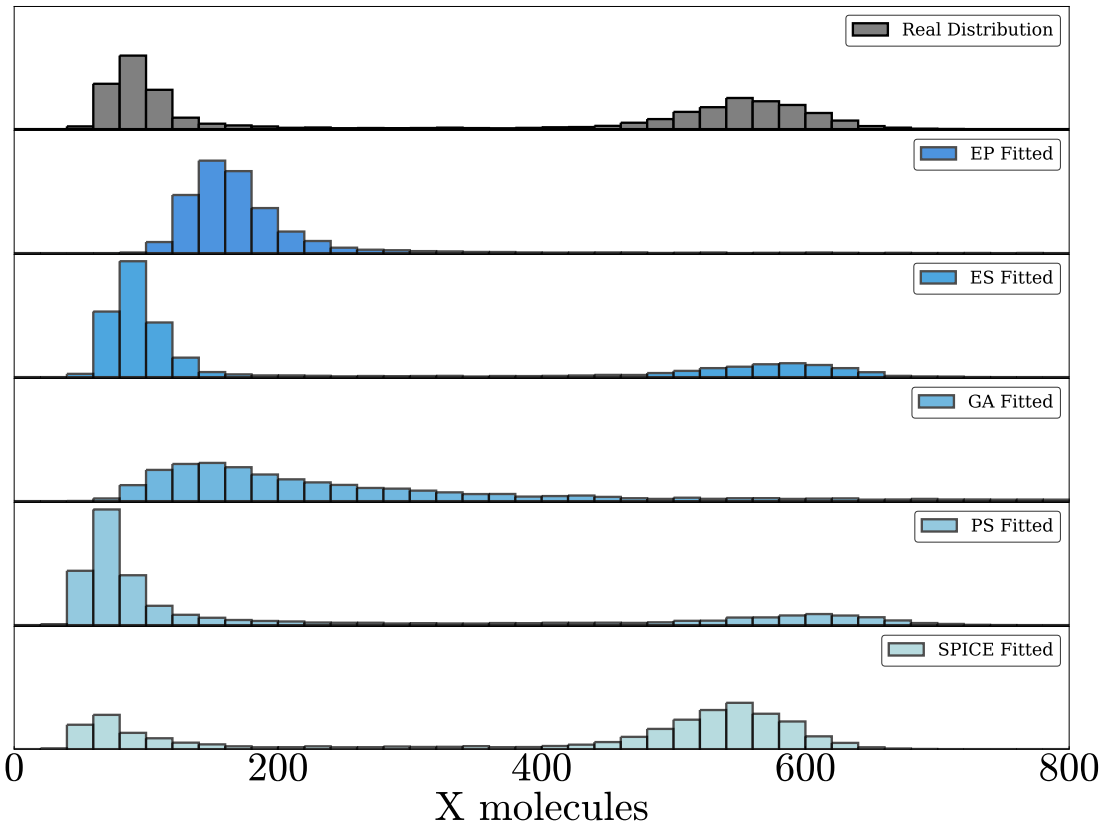


Figure 5.12: Model distribution fits for the Schlögl system. Each of the above distributions are reconstructed by using the best-fit parameters obtained from each parameter inference routine. Simulations for model reconstruction (post-inference) were conducted using Gillespie's SSA. The real data distribution is plotted on top. The selected algorithms are Evolutionary Programming (EP), Evolution Strategy (ES), Genetic Algorithm (GA), and the Particle Swarm (PS). SPICE is the stochastic parameter inference using the Cross-Entropy method presented within this thesis.

5.6 Toggle Switch Model

The final case study is a Genetic Toggle Switch model. The genetic toggle switch is a well studied bistable system, with particular importance toward synthetic biology. The toggle switch is comprised of two repressors, and two promoters, often mediated in practice through IPTG[1] and aTc[2] induction. For this model, I perform parameter inference based on real high-throughput data (see Fig. 5.13). The implemented model was based on an extension of the model proposed in [39]. Explicitly, I define the genetic toggle switch model via the following four propensity functions:

$$h_1 = \theta_1 \cdot \text{GFP} \qquad h_3 = \theta_3 \cdot \text{mCherry}$$

$$h_2 = \frac{\theta_2 \cdot \phi_1}{1 + \phi_1 + \phi_2 \cdot \text{mCherry}^2} \qquad h_4 = \frac{\theta_4 \cdot \phi_3}{1 + \phi_3 + \phi_4 \cdot \text{GFP}^2}$$

where GFP and mCherry are the two model species (fluorescent reporter molecules), and the stochastic kinetic rate parameters are represented by $(\theta_1, \ldots, \theta_4)$. The data used for parameter inference was obtained using fluorescent flow cytometry in [79]. The dataset, comprising the GFP and mCherry reporters, consists of a total of 40,731 measurements across 7 time-points. I specifically investigate the case where the genetic toggle switch starts in the low-GFP (high mCherry) state, and switches to the high-GFP (low-mCherry) state over the time course of 6 hours after aTc induction to the cells.

The inclusion of real, noisy data requires a degree of additional care. Firstly, the data needs to be rescaled from arbitrary fluorescent units (a.u.) to discrete molecular counts. To do this, I assume a linear multiplicative scale, *e.g.*, such that GFP (a.u.) $= n \times$GFP *molecules*. I make the simplifying assumption that each molecule contributes evenly to the total fluorescence values. Furthermore, I no longer assume that all the cells begin at the same initial starting conditions. Specifically, I model the initial states of GFP and mCherry according to the log-normal distribution, with mean and variance parameters to be simultaneously estimated alongside the kinetic rate constants. This introduces extra so-called 'hyperparameters', specifically the GFP molecule count to fluorescent (a.u.) scale

[1] Isopropyl β-D-1-thiogalactopyranoside
[2] anhydrotetracycline

factor ϕ_5, and the respective mCherry scale factor ϕ_6. In addition, the model now contains 4 additional (non-kinetic rate) parameters, $\phi_1, \ldots, \phi_4$, which in turn are required to be estimated. Each hyperparameter is initially sampled as before using the low-discrepancy Sobol sequence, and then from the log-normal distribution using the means and variances of the generated elite samples (as per the generic Cross-Entropy method for optimisation in Algorithm 2).

The placed bounds on the initial kinetic parameter search space were $\theta_{1,3} \in [1, 50]$, and $\theta_{2,4} \in [1\text{e}{-3}, 1]$. The respective bounds on the search space for the hyperparameters were $\phi_{1,2,3,4} \in [1\text{e} - 3, 10]$, and $\phi_{5,6} \in [50, 500]$. To generate the parameter estimates, I used SPICE with the tau-leaping approach ($\epsilon = 0.1$). The total CPU time taken was 4,293s. The resulting fitted model against the data for parameters sampled from SPICE's output distribution can be seen in Figure 5.13. The obtained parameter estimates are given in Table 5.10. Figure 5.13 reveals that SPICE does an excellent job at fitting the initial GFP-mCherry distribution (at time t=0.5hr). Despite the noise within the data, SPICE generally does a good job at reconstructing the overall dynamics of the system — with all simulation points over the time course lying within the data being fitted. The final steady-state distribution is very well-approximated.

For the genetic toggle switch model, I did not make direct comparisons with COPASI. Firstly, the real parameter estimates are unknown. Secondly, COPASI exists within a modelling ecosystem that cannot be easily customised to deal with the additional hyperparameters, such as the fluorescence scaling. Additionlly, the way datasets are passed to COPASI meant the use of real experimental data (without tedious processing) was unviable.

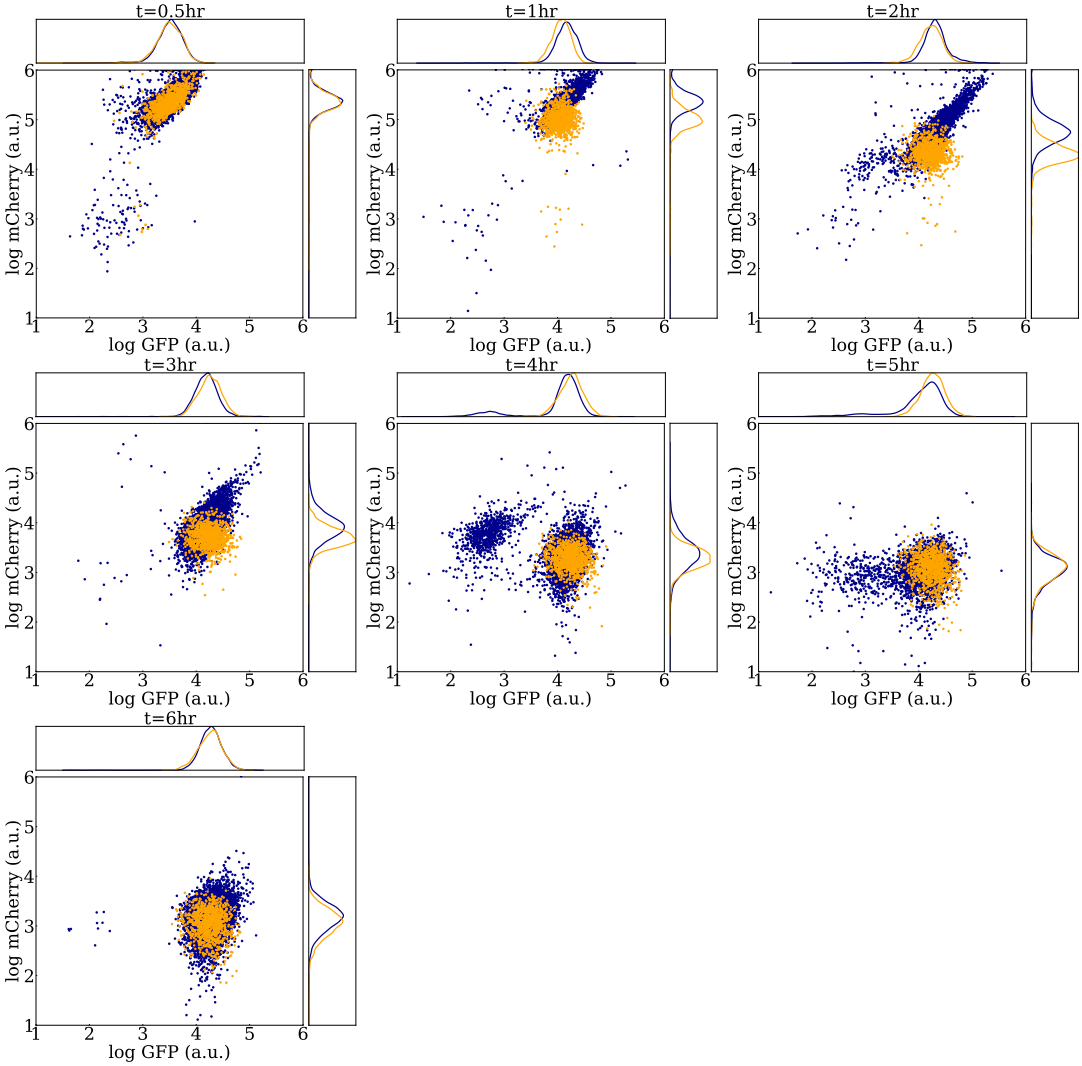


Figure 5.13: Toggle Switch Model: Blue circles: the experimental data with the $\log_{10}$(GFP) fluorescence plotted against the $\log_{10}$(mCherry) fluorescence, across all timepoints up to 6hr. Orange circles: 1,000 model simulations using the direct method, with parameters sampled from the final distribution obtained by SPICE using tau-leaping ($\epsilon = 0.1$).

Est. Rate Parameters	$\hat{\theta}_1$	$\hat{\theta}_2$	$\hat{\theta}_3$	$\hat{\theta}_4$		
mean-log	4.22	0.46	3.16	0.44		
var-log	0.077	0.081	0.19	0.0027		
Est. Hyperparameters	$\hat{\phi}_1$	$\hat{\phi}_2$	$\hat{\phi}_3$	$\hat{\phi}_4$	$\hat{\phi}_5$	$\hat{\phi}_6$
mean-log	16.5	-5.42	7.72	-2.74	6.04	4.48
var-log	2.81	1.60	1.84	0.77	0.042	0.10

Table 5.10: Estimated parameters for the Genetic Toggle Switch model, obtained from real experimental data. Presented are the means and variances of the log-transformed parameters. No error estimates are provided as the true identity of the parameters is unknown.

Chapter 6

Conclusions

6.1 Summary

Computational systems biology is one of the most rapidly developing fields of research over the last two decades [72]. The emergence of DNA sequencing, annotation, and advances in cell technologies such as microscopy, and fluorescent flow cytometry, are providing greater insights into the underlying biochemical mechanisms upon which important biological processes are built. Understanding how these processes operate and fit together is important to achieving the goals of synthetic biology which aims to engineer organisms capable of satisfying their respective model design requirements [39, 56]. Biological processes are often non-linear in behaviour, making computational modelling an efficient and crucial component to the understanding, analysis, and design of the system. The generation of reliable and accurate models is therefore required — making it necessary to use and investigate parameter inference techniques for stochastic biological systems. Parameter inference is the field concerned with using data to find parameters for which a model is valid.

This thesis was focused on the problem of parameter estimation, with the specific research aim to

investigate novel computational approaches to parameter inference for stochastic biological models.

6.2 Evaluation of Research Aims

In this thesis I presented my work on parameter inference for stochastic biological models. I have presented contributions theoretically, and applied them to numerous case studies to show applicability to realistic problems, addressing the needs of systems biology. A summary of key contributions is given below.

Chapter 2 features the mathematical framework for describing stochastic biochemical reaction networks. Within this chapter, I present the chemical master equation, which describes the temporal evolution of the underlying probability distribution of the system. I also present the two commonly used methods to sample solutions from the CME: (i) Gillespie's stochastic simulation algorithm, and (ii) the (optimised) tau-leaping algorithm. I also present useful theory toward parameter inference regarding the complete-data likelihood of SSA trajectories.

Chapter 3 presents the application of the Cross-Entropy method toward global optimisation problems. Investigating the novel use of Cross-Entropy method was one of the main goals of this thesis toward addressing the problem of parameter estimation for stochastic biological models. Within this chapter, I apply the Cross-Entropy method for parameter inference within stochastic chemical reaction networks, deriving unseen solutions for the optimal kinetic rate parameters given the underlying distribution of either Gillespie's stochastic simulation algorithm, or the tau-leaping algorithm. I then present SPICE, the *stochastic parameter inference using the Cross-Entropy* algorithm. Within the SPICE algorithm, I implement various heuristics that can be used for inference, such as low-discrepancy Sobol sequencing, and adaptive sampling which aim to optimise convergence. These heuristic methods introduce additional hyperparameters. For the context of this thesis, I chose nominal hyperparameter values within ranges given by previous literature. However, it is anticipated that future research should optimise these values. I also demonstrate the multiple shooting approach (previously used for ODE systems), and particle splitting approaches that can be used alongside the inference routine. I remark that the use of particle splitting can draw comparisons between SPICE and ABC-SMC approaches. Both SPICE and ABC-SMC are able to improve the speed of convergence by sampling with replacement those simulations (or trajectories) that are performing well based on

the objective function, whilst discarding those performing poorly. The difference is that while Bayesian approaches use acceptance-rejection criterion to formulate an empirical distribution for the parameters, SPICE updates the underlying parameters of the posterior distribution of known form. As a consequence, Bayesian methods may be better suited to estimating parameters whose distribution shape is awkward or multi-modal, while the Cross-Entropy approach can be applied to any well known statistical distribution.

The Cross-Entropy method presented in this thesis relies on stochastic simulations which are known to be computationally expensive. With this in mind, I decided to investigate the use of moment approximations to further address the research question. Moment approximations are ODE systems derived from the chemical master equation that try to recreate the mean, variance, and higher order moments of stochastic models. Because ODEs are quick and efficiently evaluated, the method of moments for parameter inference promises speed gains over other CTMC simulation approaches. Chapter 4 presents the theory of moment approximations for stochastic biological networks, outlining the standard moment closure approximations. In this section, I provide comments and new insight on the validity of the moment approximations. In particular, I address the use of Taylor series within moment approximations, and propose a new method based on constructing Padé approximants. Padé approximants rational approximations of a given function, but can often converge in a radius far beyond their Taylor series counterparts. In this thesis I demonstrate the Padé approximation in one-dimension, laying the grounds for future research into multivariate approximations for stochastic biological models.

To further address the research aims, in Section 4.6 I propose an approach for parameter inference that combines the generalised method of moments with recent advances in multi-level simulation techniques. The use of the multi-level tau-leaping algorithm aims to provide a stochastic correction to parameter estimates retrieved from using robust method of moment approximations. Whist preliminary results show the proposed approach can be very accurate, it was often found that the method of moments was often sufficient enough to produce reasonable parameter estimates. Furthermore, the time spent in the multi-level method phase was significantly greater than that in the moment approximation

phase, raising questions on the overall benefits of the approach compared to direct simulation. This may in part be due to poor optimisation of the introduced hyperparameters within the presented MLGMM algorithm. Specifically, deciding the number of simulations to take at each level of the multi-level approach is not a straightforward problem that can be easily generalised across all models.

In Chapter 5 I provide results and analysis for the implementation of SPICE for various challenging case studies. Specifically, I performed parameter estimation for the Lotka-Voltera model, the Yeast Polarization model, the Schlögl system, and a Genetic Toggle Switch model using real experimental data. Within Chapter 5 I also provide comparisons between SPICE and commonly used parameter inference routines within the modelling environment COPASI. It was shown that SPICE can produce parameter estimates on par, and better to existing state-of-the-art approaches, given the same amount of computation time. Additionally, the presented SPICE algorithm is able to significantly outperform its EM-algorithm cousins, such as MCEM^2 [26], in terms of computation time [101]. However, SPICE's reliance on direct simulation means that newer inference methods based on the method of moments are able to outperform SPICE in terms of computational speed. Finally, it was shown that SPICE is able to handle unobserved data scenarios, alongside noisy experimental data in the Toggle Switch example which required additional hyperparameters to be optimised.

6.3 Limitations and Scalability

Stochastic Simulation In Chapter 3 I presented SPICE — *Stochastic Parameter Inference using the Cross-Entropy* — an algorithm that can be used for estimating parameters for stochastic chemical reaction networks. The method combines Gillespie's stochastic simulation algorithm with the Cross-Entropy method for optimisation. Gillespie's SSA is used to sample exact trajectories of the chemical master equation and provides the highest level of accuracy beyond solving the equation directly, which is often infeasible. Unfortunately, it is computational intensive as it tracks the numbers of each species over the complete time course, performing propensity calculations after every reaction firing. This means that inference can become very slow, especially when models are large and dense. This

could be problematic as research time schedules are often limited. In the field of synthetic biology, it could be anticipated that quick parameter estimation across multiple models may be vital to an efficient design work-flow, or real-time experimental feedback. This motivated the choice to investigate the use of tau-leaping, and to derive its Cross-Entropy method solution. It was shown in Chapter 5 that the use of tau-leaping can achieve significant gains in computation speed, while remaining just as accurate. However, tau-leaping can still be computationally slow. Furthermore, the models used in this thesis generally not very large — either in the numbers of species involved, or the number of reaction types available to the system. These models are, however, the standard models that are frequently analysed within the field — a fact that motivated their choice. The criticism of the models thus applies to the wider field of parameter inference for stochastic chemical reaction networks. For parameter inference, it would be beneficial to introduce larger standard models to test SPICE and other algorithms upon.

One-Dimensional Padé Approximations Within Section 4.4 I motivated and demonstrated the use of Padé approximations as an alternative to the Taylor series for moment expansions. The Padé approximant aims to provide a rational approximation of a function using the original terms of the Taylor series. It was shown (Figure 4.2: the one-dimensional Schlögl system) that the Padé approximation can avoid non-feasible solutions obtained when using normal moment closure approaches, and could improve the accuracy (for both the mean and variance) of the resulting simulations. However, the methods presented within Section 4.4 do not immediately transition well to multivariate settings. Specifically, using the matrix operations given by Equations (4.4.2) and (4.4.3) will yield uniquely determinable Padé coefficients for the univariate case — whilst this is not true for the multivariate scenario. Firstly, for the multivariate case, the number of linear equations is often not sufficient to constrain the solution of the coefficients. Secondly, the multivariate Padé approximations can sometimes have poles near the point of expansion. The presence of poles can make the application of the multivariate Padé approximation to a modelled system difficult, or invalid. Thus, the method presented within this thesis is at present limited to the one-dimensional

case. Unfortunately, the majority of biological systems require more than one variable to be modelled usefully. However, the work in this thesis is important in sowing the seed for future research, and acts as a natural step toward the construction of alternative moment closure techniques. In Section 6.4.2, I outline possible research starting points to overcome these difficulties.

6.4 Future Work

Different future research steps can be taken based on the work presented within this thesis. In the following subsections, I present guidance for future research directions.

6.4.1 Cross-Entropy Method

In Chapter 3 I introduced the Cross-Entropy method for parameter inference within stochastic biological models. As discussed within Section 6.3, one key limitation of the Cross-Entropy approach presented within this thesis is the poor scaling (in terms of computational time) to larger models. This is due to the vast number of calculations required by Gillespie's stochastic simulation algorithm [44]. To partially address this, I provided motivation for the use of the stochastic tau-leaping simulation algorithm [21] in conjunction with the Cross-Entropy method, and derived its solution in Section 3.4. For SPICE, the results shown in Chapter 5 reveal that the use of tau-leaping can yield parameter estimates within error of those derived from using Gillespie's exact algorithm, but in a fraction of the time — with the greatest time saving in the Schlögl system model where the number of reactions is the greatest. However, to address the goals of systems biology — one of which is to construct complete mechanistic 'whole cell' models — larger models will inevitably become available in future. For these large models the time savings gained from using tau-leaping may not be sufficient enough. This motivates the search for alternative stochastic simulation techniques that are not only accurate, but fast, and compatible with the general Cross-Entropy approach. One such recent technique that is not yet well known is that of δ-leaping [104]. δ-leaping is aimed at improving the efficiency of

stochastic simulations of large chemical reaction networks. The technique models the biological system using Petri nets, and applies discrete time leaps in a similar manner to tau-leaping, generating reactions according to a maximum firing rule. Furthermore, evidence suggests that the method scales much better in terms of computational time than both Gillespie's SSA and tau-leaping [104]. Given the dependence of the discrete time steps and reaction firings on the hazard function (or propensity), and convenient CTMC form of the simulation, it could be well suited for parameter inference using the Cross-Entropy method approach, and SPICE.

6.4.2 Multivariate Padé Approximations

Within Section 4.4 I introduced the Padé approximation method for moment expansions. The Padé approximant provides the *best possible* rational approximation of a function, and is easily derived from the numerical terms of the original Taylor series that it replaces. For the case of a one-dimensional, univariate system, the coefficients within the numerator and denominator of the Padé approximant are uniquely determinable. However, as stated in Section 6.3, this is not necessarily true in the multivariate case. More explicitly, there does not exist a formula that can always derive the correct number of linear equations to determine the coefficients. However, there have been many efforts within computational and applied mathematics to provide a generalization of multivariate Padé approximations that are able to yield good convergence properties [49]. In particular, my suggestion would be to use a generalisation of Wynn's ϵ-algorithm [131] based on the work of Cuyt [25]. Within Cuyt's work, the links between the terms of Wynn's ϵ-algorithm are shown to coincide with certain degree orders of the Padé approximants, with extension to multivariate examples.

Importantly, the use of the Padé approximations within moment closure expansions can extend well beyond the scope of biological modelling, as it can be applied to a multitude of domains. For example, econometric models often use moment approximations to analyse trends, risks, and make predictions based on time series data.

References

[1] Angelique Ale, Paul Kirk, and Michael P. H. Stumpf. A general moment expansion method for stochastic kinetic models. *The Journal of Chemical Physics*, **138**[17]:174101, 2013. 7

[2] Angelique Ale, Paul Kirk, and Michael P. H. Stumpf. A general moment expansion method for stochastic kinetic models. *The Journal of Chemical Physics*, **138**[17]:174101, 2013. 54, 55, 56, 57

[3] D. Anderson and D. Higham. Multilevel monte carlo for continuous time markov chains, with applications in biochemical kinetics. *Multiscale Modeling & Simulation*, **10**[1]:146–179, 2012. 74, 76

[4] George E Andrews, Ian P Goulden, and David M Jackson. Shanks' convergence acceleration transform, padé approximants and partitions. *Journal of Combinatorial Theory, Series A*, **43**[1]:70 – 84, 1986. 65

[5] Aleksandr Andreychenko, Linar Mikeev, David Spieler, and Verena Wolf. Approximate maximum likelihood estimation for stochastic chemical kinetics. *EURASIP Journal on Bioinformatics and Systems Biology*, **2012**[1]:9, 2012. 5, 70

[6] Christophe Andrieu, Arnaud Doucet, and Roman Holenstein. Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, **72**[3]:269–342, 2010. 45

[7] Maksat Ashyraliyev, Yves Fomekong-Nanfack, Jaap A. Kaandorp, and Joke G. Blom. Systems biology: parameter estimation for biochemical models. *The FEBS Journal*, **276**[4]:886–902, 2009. 4

[8] P. Azunre. *Mass fluctuation kinetics: analysis and computation of equilibria and local dynamics*. Master Thesis, Massachusetts Intsitute of Technology, 2007. 54, 55

[9] T. Bäck, D.B. Fogel, and Z. Michalewicz. *Handbook of evolutionary computation*. Oxford University Press, 1997. 6, 84

[10] Carl Bender. *Advanced mathematical methods for scientists and engineers I : asymptotic methods and perturbation theory*. Springer, New York, NY, 1999. 63, 64

[11] Carl M Bender and Hugh F Jones. Calculation of low-lying energy levels in quantum mechanics. *Journal of Physics A: Mathematical and Theoretical*, **47**[39]:395303, 2014. 63

[12] Frank T. Bergmann, Sven Sahle, and Christoph Zimmer. Piecewise parameter estimation for stochastic models in COPASI. *Bioinformatics*, **32**[10]:1586–1588, 2016. 6, 45

[13] J. Bezanson, A. Edelman, S. Karpinski, and V. Shah. Julia: A fresh approach to numerical computing. *SIAM Review*, **59**[1]:65–98, 2017. 48, 79

[14] William J Blake, Mads Kaern, Charles R Cantor, and J J Collins. Noise in eukaryotic gene expression. *Nature*, **422**[6932]:633–637, 2003. 2, 14

[15] R Boys, D Wilkinson, and T Kirkwood. Bayesian inference for a discretely observed stochastic kinetic model. *Statistics and Computing*, **18**, 2008. 5

[16] Richard P. Brent. *Algorithms for Minimization Without Derivatives*. Prentice-Hall, Inc., Englewood Cliffs, New Jersery, 1973. Reissued in 2002 by Dover Publications, Mineaola, New York. 6

[17] C. Brezinski. Extrapolation algorithms and padé approximations: a historical survey. *Applied Numerical Mathematics*, **20**[3]:299 – 318, 1996. 65

[18] C. Brezinski. From numerical quadrature to padé approximation. *Applied Numerical Mathematics*, **60**[12]:1209 – 1220, 2010. Approximation and extrapolation of convergent and divergent sequences and series (CIRM, Luminy - France, 2009). 63

[19] Brian S. Caffo, Wolfgang Jank, and Galin L. Jones. Ascent-based Monte Carlo expectation- maximization. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **67**[2]:235–251, 2005. 5, 42

[20] Yang Cao, Daniel T. Gillespie, and Linda R. Petzold. Efficient step size selection for the tau-leaping simulation method. *The Journal of chemical physics*, **124**[4]:044109, 2006. 4, 18, 19, 20, 21, 22, 33, 74, 76, 84

[21] Yang Cao, Hong Li, and Linda Petzold. Efficient formulation of the stochastic simulation algorithm for chemically reacting systems. *Journal of Chemical Physics*, **121**[9]:4059–4067, 2004. 18, 122

[22] Marta Cascante, Laszlo G. Boros, Begoña Comin-Anduix, Pedro de Atauri, Josep J. Centelles, and Paul W. N Lee. Metabolic control analysis in drug discovery and disease. *Nature Biotechnology*, **20**:243 EP –, Mar 2002. 2

[23] Federica Ciocchetta and Jane Hillston. Bio-pepa: A framework for the modelling and analysis of biological systems. *Theoretical Computer Science*, **410**[33]:3065 – 3084, 2009. Concurrent Systems Biology: To Nadia Busi (1968–2007). 7

[24] Andre Costa, Owen Dafydd Jones, and Dirk Kroese. Convergence properties of the cross-entropy method for discrete optimization. *Operations Research Letters*, **35**[5]:573 – 580, 2007. 45

[25] A.A.M. Cuyt. The e-algorithm and multivariate pade-approximants. *Numerische Mathematik*, **40**:39–46, 1982. 123

[26] Bernie J Daigle, Min K Roh, Linda R Petzold, and Jarad Niemi. Accelerated maximum likelihood parameter estimation for stochastic biochemical systems. *BMC Bioinformatics*, **13**[1]:68, 2012. 5, 7, 25, 32, 33, 36, 37, 38, 45, 96, 97, 120

[27] Sandra H. Dandach and Mustafa Khammash. Analysis of stochastic strategies in bacterial competence: A master equation approach. *PLoS Computational Biology*, **6**[11]:1–11, 2010. 5

[28] T. Dayar, L. Mikeev, and V. Wolf. On the numerical analysis of stochastic lotka-volterra models. In *Proceedings of the International Multiconference on Computer Science and Information Technology*, pages 289–296, Oct 2010. 59, 60

[29] Pieter-Tjerk de Boer, Dirk P. Kroese, Shie Mannor, and Reuven Y. Rubinstein. A tutorial on the cross-entropy method. *Annals of Operations Research*, **134**[1]:19–67, Feb 2005. 26, 29, 44

[30] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, **39**[1]:1–38, 1977. 4

[31] L. DeVille, S. Dhople, A. Domínguez-García, and J. Zhang. Moment closure and finite-time blowup for piecewise deterministic markov processes. *SIAM Journal on Applied Dynamical Systems*, **15**[1]:526–556, 2016. 59

[32] Brian Drawert, Andreas Hellander, Ben Bales, Debjani Banerjee, Giovanni Bellesia, Bernie J. Daigle, Jr., Geoffrey Douglas, Mengyuan Gu, Anand Gupta, Stefan Hellander, Chris Horuk, Dibyendu Nath, Aviral Takkar, Sheng Wu, Per Lötstedt, Chandra Krintz, and Linda R. Petzold. Stochastic simulation service: Bridging the gap between the computational expert and the biologist. *PLOS Computational Biology*, **12**[12]:1–15, 12 2016. 6, 97

[33] David J. Earl and Michael W. Deem. Parallel tempering: Theory, applications, and new perspectives. *Phys. Chem. Chem. Phys.*, **7**:3910–3916, 2005. 8

[34] Stefan Engblom. Computing the moments of high dimensional solutions of the master equation. *Applied Mathematics and Computation*, **180**[2]:498 – 515, 2006. 4, 5, 7, 56

[35] Edgar Everhart. An efficient integrator that uses gauss-radau spacings. In Andrea Carusi and Giovanni B. Valsecchi, editors, *Dynamics of Comets: Their Origin and Evolution*, pages 185–202, Dordrecht, 1985. Springer Netherlands. 61

[36] Justin Feigelman, Stefan Ganscha, Simon Hastreiter, Michael Schwarzfischer, Adam Filipczyk, Timm Schroeder, Fabian J. Theis, Carsten Marr, and Manfred Claassen. Analysis of cell lineage trees by exact bayesian inference identifies negative autoregulation of nanog in mouse embryonic stem cells. *Cell Systems*, **3**[5]:480–490.e13, Nov 2016. 5

[37] D. B. Fogel, L. J. Fogel, and J. W. Atmar. Meta-evolutionary programming. In *[1991] Conference Record of the Twenty-Fifth Asilomar Conference on Signals, Systems Computers*, pages 540–545 vol.1, Nov 1991. 6

[38] Fabian Fröhlich, Philipp Thomas, Atefeh Kazeroonian, Fabian J. Theis, Ramon Grima, and Jan Hasenauer. Inference for stochastic chemical kinetics using moment equations and system size expansion. *PLoS Comp. Bio.*, **12**[7]:1–28, 2016. 5, 6

[39] Timothy S. Gardner, Charles R. Cantor, and James J. Collins. Construction of a genetic toggle switch in Escherichia coli. *Nature*, **403**[6767]:339–342, 2000. 2, 112, 117

[40] Anastasis Georgoulas, Jane Hillston, Dimitrios Milios, and Guido Sanguinetti. Probabilistic programming process algebra. In

Gethin Norman and William Sanders, editors, *Quantitative Evaluation of Systems*, pages 249–264, Cham, 2014. Springer International Publishing. 7

[41] Michael A. Gibson and Jehoshua Bruck. Efficient exact stochastic simulation of chemical systems with many species and many channels. *The Journal of Physical Chemistry A*, **104**[9]:1876–1889, 2000. 18

[42] C. S. Gillespie. Moment-closure approximations for mass-action models. *IET Systems Biology*, **2**[2]:64–72, 2008. 5, 56, 58

[43] D. T. Gillespie. Approximate accelerated stochastic simulation of chemically reacting systems. *Journal of Chemical Physics*, **115**[4]:1716–1733, 2001. 4, 18, 19, 20, 33, 48, 74

[44] Daniel T Gillespie. Exact stochastic simulation of coupled chemical reactions. *The Journal of Physical Chemistry*, **81**[25]:2340–2361, 1977. 4, 5, 7, 17, 18, 66, 74, 122

[45] Daniel T. Gillespie. A rigorous derivation of the chemical master equation. *Physica A: Statistical Mechanics and its Applications*, **188**[1]:404 – 425, 1992. 14, 15, 16, 53

[46] Daniel T. Gillespie and Linda R. Petzold. Improved leap-size selection for accelerated stochastic simulation. *The Journal of Chemical Physics*, **119**[16]:8229–8234, 2003. 19, 21, 22

[47] Paul Glasserman, Philip Heidelberger, Perwez Shahabuddin, and Tim Zajic. Multilevel splitting for estimating rare event probabilities. *Oper. Res.*, **47**[4]:585–600, 1999. 45

[48] Andrew Golightly and Darren J Wilkinson. Bayesian parameter inference for stochastic biochemical network models using particle Markov chain Monte Carlo. *Interface Focus*, **1**[6]:807–820, 2011. 5

[49] Philippe Guillaume and Alain Huard. Multivariate padé approximation. *Journal of Computational and Applied Mathematics*, **121**[1]:197 – 219, 2000. 123

[50] Alastair R. Hall. *Generalized Method of Moments*. Oxford University Press, 2005. 5, 70, 71, 72, 73

[51] Lars Peter Hansen. Large sample properties of generalized method of moments estimators. *Econometrica*, **50**[4]:1029–1054, 1982. 70, 72

[52] Lars Peter Hansen, John Heaton, and Amir Yaron. Finite-sample properties of some alternative gmm estimators. *Journal of Business & Economic Statistics*, **14**[3]:262–280, 1996. 73

[53] Leonard A. Harris, Justin S. Hogg, José-Juan Tapia, John A. P. Sekar, Sanjana Gupta, Ilya Korsunsky, Arshi Arora, Dipak Barua, Robert P. Sheehan, and James R. Faeder. Bionetgen 2.2: advances in rule-based modeling. *Bioinformatics*, **32**[21]:3366–3368, 2016. 7

[54] J Hasenauer, V Wolf, A Kazeroonian, and F J Theis. Method of conditional moments (MCM) for the chemical master equation. *J Math Biol*, **69**, 2013. 70

[55] J Hasenauer, V Wolf, A Kazeroonian, and F J Theis. Method of conditional moments (MCM) for the Chemical Master Equation. *Journal of Mathematical Biology*, **69**[3]:687–735, 2014. 5

[56] Jeff Hasty, David McMillen, and J. J. Collins. Engineered gene circuits. *Nature*, **420**:224 EP –, Nov 2002. 2, 117

[57] A. C. Hindmarsh and L. R. Petzold. Lsoda, ordinary differential equation solver for stiff or non-stiff system, Sep 2005. 61

[58] Alan C Hindmarsh, Peter N Brown, Keith E Grant, Steven L Lee, Radu Serban, Dan E Shumaker, and Carol S Woodward. SUNDIALS: Suite of nonlinear and differential/algebraic equation solvers. *ACM Transactions on Mathematical Software (TOMS)*, **31**[3]:363–396, 2005. 61

[59] Walter Hlawitschka. The empirical nature of taylor-series approximations to expected utility. *The American Economic Review*, **84**[3]:713–719, 1994. 63

[60] Robert Hooke and T. A. Jeeves. " direct search" solution of numerical and statistical problems. *J. ACM*, **8**[2]:212–229, April 1961. 6

[61] Stefan Hoops, Sven Sahle, Ralph Gauges, Christine Lee, Jürgen Pahle, Natalia Simus, Mudita Singhal, Liang Xu, Pedro Mendes, and Ursula Kummer. Copasi - a complex pathway simulator. *Bioinformatics*, **22**[24]:3067–3074, 2006. 6, 45, 83, 84

[62] András Horváth and Daniele Martini. Parameter estimation of kinetic rates in stochastic reaction networks by the EM method. In *BMEI*, **1**, pages 713–717. IEEE, 2008. 4, 25, 38

[63] Martin Howard and Andrew D. Rutenberg. Pattern formation inside bacteria: Fluctuations due to the low copy number of proteins. *Phys. Rev. Lett.*, **90**:128102, Mar 2003. 2

[64] M. Hucka, A. Finney, H. M. Sauro, H. Bolouri, J. C. Doyle, H. Kitano, 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. Le 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*, **19**[4]:524–531, 2003. 6

[65] O.L. Ibryaeva and V.M. Adukov. An algorithm for computing a padé approximant with minimal degree denominator. *Journal of Computational and Applied Mathematics*, **237**[1]:529 – 541, 2013. 65

[66] L. ISSERLIS. On certain probable errors and correlation coefficients of multiple frequency distributions with skew regression. *Biometrika*, **11**[3]:185–190, 1916. 58

[67] Feigelman Justin, Weindl Daniel, Theis Fabian J., Marr Carsten, and Hasenauer Jan. Lna++: Linear noise approximation with first and second order sensitivities. In Milan Češka and David Šafránek, editors, *Computational Methods in Systems Biology*, Cham, 2018. Springer International Publishing. 7

[68] Jonathan R. Karr, Jayodita C. Sanghvi, Derek N. MacKlin, Miriam V. Gutschow, Jared M. Jacobs, Benjamin Bolival, Nacyra Assad-Garcia, John I. Glass, and Markus W. Covert. A whole-cell computational model predicts phenotype from genotype. *Cell*, **150**[2]:389–401, 2012. 2

[69] Atefeh Kazeroonian, Fabian Fröhlich, Andreas Raue, Fabian J. Theis, and Jan Hasenauer. Cerena: Chemical reaction network analyzer–a toolbox for the simulation and analysis of stochastic chemical kinetics. *PLOS ONE*, **11**[1]:1–15, 01 2016. 7

[70] James Kennedy and Russell C. Eberhart. Particle swarm optimization. In *Proceedings of the 1995 IEEE International Conference on Neural Networks*, **4**, pages 1942–1948, Perth, Australia, IEEE Service Center, Piscataway, NJ, 1995. 6, 84

[71] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, **220**[4598]:671–680, 1983. 6, 8

[72] Hiroaki Kitano. Computational systems biology. *Nature*, **420**:206 EP –, Nov 2002. 1, 117

[73] David J. Klinke. In silico model-based inference: A contemporary approach for hypothesis testing in network biology. *Biotechnology Progress*, **30**[6]:1247–1261, 2014. 8

[74] M Komorowski, B Finkenstädt, C V Harper, and D A Rand. Bayesian inference of biochemical kinetic parameters using the linear noise approximation. *BMC Bioinformatics*, **10**[1]:343, 2009. 6

[75] Dirk P. Kroese. Cross-entropy method. In *Wiley Encyclopedia of Operations Research and Management Science*, pages 1–12. Wiley, 2010. 29, 37, 44

[76] S. Kullback and R. A. Leibler. On information and sufficiency. *Ann. Math. Statist.*, **22**[1]:79–86, 1951. 26, 27

[77] Eszter Lakatos, Angelique Ale, Paul D. W. Kirk, and Michael P. H. Stumpf. Multivariate moment closure techniques for stochastic kinetic models. *The Journal of Chemical Physics*, **143**[9]:094107, 2015. 58, 59

[78] Maksim Lapin, Linar Mikeev, and Verena Wolf. Shave: stochastic hybrid analysis of markov population models. In *Proceedings of the 14th ACM International Conference on Hybrid Systems: Computation and Control, HSCC 2011, Chicago, IL, USA, April 12-14, 2011*, pages 311–312. ACM, January 2011. pub_id: 705 Bibtex: LaMiWo_11:Shave URL date: None. 7

[79] Miriam Leon. *Computational design and characterisation of synthetic genetic switches*. PhD thesis, Department of Cell and Developmental Biology, University College London, 2017. Available at http://discovery.ucl.ac.uk/1546318/1/Leon_Miriam_thesis_final.pdf. 83, 112

[80] C. Lester, C. A. Yates, M. B. Giles, and R. E. Baker. An adaptive multi-level simulation algorithm for stochastic biological systems. *The Journal of Chemical Physics*, **142**[2]:024113, 2015. 74, 75, 76

[81] Hong Li and Linda Petzold. Logarithmic direct method for discrete stochastic simulation of chemically reacting systems. Technical report, University of California Santa Barbara, 2006. 18

[82] Shuohao Liao, Tomáš Vejchodský, and Radek Erban. Tensor methods for parameter estimation and bifurcation analysis of stochastic reaction networks. *Interface, Journal of the Royal Society*, **12**[108]:20150233, 2015. 6

[83] Juliane Liepe, Paul Kirk, Sarah Filippi, Tina Toni, Chris P. Barnes, and Michael P. H. Stumpf. A framework for parameter estimation and model selection from experimental data in systems biology using approximate bayesian computation. *Nature Protocols*, **9**:439 EP –, Jan 2014. 5

[84] Otto Loistl. The erroneous approximation of expected utility by means of a taylor's series expansion: Analytic and computational results. *The American Economic Review*, **66**[5]:904–910, 1976. 63

[85] Alexander Lück and Verena Wolf. Generalized method of moments for estimating parameters of stochastic reaction networks. *BMC Systems Biology*, **10**[1]:98, 2016. 5, 70, 71, 72

[86] D. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial and Applied Mathematics*, **11**[2]:431–441, 1963. 6

[87] Harley H McAdams and Adam Arkin. Stochastic mechanisms in gene expression. *PNAS*, **94**[3]:814–819, 1997. 2, 14

[88] James M. McCollum, Gregory D. Peterson, Chris D. Cox, Michael L. Simpson, and Nagiza F. Samatova. The sorting direct method for stochastic simulation of biochemical systems with varying reaction execution behavior. *Computational Biology and Chemistry*, **30**[1]:39 – 49, 2006. 18

[89] Donald A. McQuarrie. Stochastic approach to chemical kinetics. *Journal of Applied Probability*, **4**[3]:413–478, 1967. 70

[90] Carmen G Moles, Pedro Mendes, and Julio R Banga. Parameter Estimation in Biochemical Pathways: A Comparison of Global Optimization Methods. *Genome Research*, pages 2467–2474, 2003. 3

[91] Brian Munsky and Mustafa Khammash. The finite state projection algorithm for the solution of the chemical master equation. *The Journal of Chemical Physics*, **124**[4]:044104, 2006. 7, 17

[92] Stephen G. Nash. Newton-type minimization via the lanczos method. *SIAM Journal on Numerical Analysis*, **21**[4]:770–788, 1984. 6

[93] J. A. Nelder and R. Mead. A simplex method for function minimization. *The Computer Journal*, **7**[4]:308–313, 1965. 6

[94] J. R. Norris. *Markov chains*. Cambridge University Press, Cambridge, UK; New York, 1998. 76

[95] Ingemar NÃĕsell. An extension of the moment closure method. *Theoretical Population Biology*, **64**[2]:233 – 239, 2003. 59

[96] Oleksandr Ostrenko, Pietro Incardona, Rajesh Ramaswamy, Lutz Brusch, and Ivo F. Sbalzarini. pssalib: The partial-propensity stochastic chemical network simulator. *PLoS Comput Biol*, **13**[12]:e1005865, Dec 2017. 29206229[pmid]. 18

[97] Art B. Owen. A central limit theorem for latin hypercube sampling. *Journal of the Royal Statistical Society. Series B (Methodological)*, **54**[2]:541–551, 1992. 42

[98] Jason R. Pirone and Timothy C. Elston. Fluctuations in transcription factor binding can explain the graded and binary responses observed in inducible gene expression. *Journal of Theoretical Biology*, **226**[1]:111–112, 2004. 2, 14

[99] M. J. D. Powell. *A Direct Search Optimization Method That Models the Objective and Constraint Functions by Linear Interpolation*, pages 51–67. Springer Netherlands, Dordrecht, 1994. 80

[100] M. Quack. N. g. van kampen: Stochastic processes in physics and chemistry. north holland publishing company, amsterdam 1981. 419 seiten, preis: $ 76.50. *Berichte der Bunsengesellschaft für physikalische Chemie*, **87**[4]:370–370, 2007. 4, 6, 7

[101] Jeremy Revell and Paolo Zuliani. Stochastic rate parameter inference using the cross-entropy method. In Milan Češka and David Šafránek, editors, *Computational Methods in Systems Biology*, pages 146–164, Cham, 2018. Springer International Publishing. 9, 38, 120

[102] K. F. Riley, M. P. Hobson, and S. J. Bence. *Mathematical Methods for Physics and Engineering: A Comprehensive Guide*. Cambridge University Press, 2 edition, 2002. 54, 58, 63

[103] Christian P. Robert and George Casella. *Monte Carlo Statistical Methods*. Springer-Verlag, 2 edition, 2004. 12, 13, 36

[104] Christian Rohr. Discrete-time leap method for stochastic simulation. In *PNSE @ Petri Nets*, 2016. 122, 123

[105] Reuven Y. Rubinstein. Optimization of computer simulation models with rare events. *European Journal of Operational Research*, **99**[1]:89–112, 1997. 5, 25, 29

[106] Reuven Y Rubinstein. The cross-entropy method for combinatorial and continuous optimization. *Methodology and Computing in Applied Probability*, **1**[2]:127–190, 1999. 5, 25

[107] Reuven Y. Rubinstein and Dirk P. Kroese. *The Cross-Entropy Method*. Springer, 2004. 25

[108] Reuven Y. Rubinstein and Dirk P. Kroese. *Simulation and the Monte Carlo Method*. Wiley, 2008. 27, 28, 29, 30, 37

[109] Reuven Y. Rubinstein and Alexander Shapiro. *Discrete event systems: sensitivity analysis and stochastic optimization by the score function method*. John Wiley & Sons, 1993. 28

[110] T. P. Runarsson and Xin Yao. Stochastic ranking for constrained evolutionary optimization. *IEEE Transactions on Evolutionary Computation*, **4**[3]:284–294, Sept 2000. 6, 84

[111] Kevin R. Sanft, Sheng Wu, Min Roh, Jin Fu, Rone Kwei Lim, and Linda R. Petzold. Stochkit2: software for discrete stochastic simulation of biochemical systems with events. *Bioinformatics*, **27**[17]:2457–2458, 2011. 6

[112] J. Schaff, C.C. Fink, B. Slepchenko, J.H. Carson, and L.M. Loew. A general computational framework for modeling cellular structure and function. *Biophysical Journal*, **73**[3]:1135 – 1146, 1997. 6

[113] Christian Schilling, Sergiy Bogomolov, Thomas A. Henzinger, Andreas Podelski, and Jakob Ruess. Adaptive moment closure for parameter inference of biochemical reaction networks. *Biosystems*, **149**:15 – 25, 2016. Selected papers from the Computational Methods in Systems Biology 2015 conference. 70, 71, 73, 74

[114] Friedrich Schlögl. Chemical reaction models for non-equilibrium phase transitions. *Zeitschrift für physik*, **253**[2]:147–161, 1972. 66, 105

[115] David Schnoerr, Guido Sanguinetti, and Ramon Grima. Comparison of different moment-closure approximations for stochastic chemical kinetics. *The Journal of Chemical Physics*, **143**[18]:185101, 2015. 7, 59

[116] David Schnoerr, Guido Sanguinetti, and Ramon Grima. Approximation and inference methods for stochastic biochemical kinetics–a tutorial review. *Journal of Physics A: Mathematical and Theoretical*, **50**[9]:093001, 2017. 2, 73

[117] A. Singh and J. P. Hespanha. Lognormal moment closures for biochemical reactions. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 2063–2068, Dec 2006. 58, 59

[118] S A Sisson, Y Fan, and Mark M Tanaka. Sequential Monte Carlo without likelihoods. *PNAS*, **104**[6]:1760–5, 2007. 5, 45

[119] Ilya M. Sobol'. On the distribution of points in a cube and the approximate evaluation of integrals. *USSR Computational Mathematics and Mathematical Physics*, **7**[4]:86 – 112, 1967. 39, 41

[120] S P Suetin. Padé approximants and efficient analytic continuation of a power series. *Russian Mathematical Surveys*, **57**[1]:43, 2002. 63

[121] Martin A. Tanner. *Tools for Statistical Inference: Methods for the Exploration of Posterior Distributions and Likelihood Functions (Springer Series in Statistics)*. Springer, 2011. 36

[122] Philipp Thomas, Hannes Matuschek, and Ramon Grima. Intrinsic noise analyzer: A software package for the exploration of stochastic biochemical kinetics using the system size expansion. *PLOS ONE*, **7**[6]:1–20, 06 2012. 7

[123] Tianhai Tian, Songlin Xu, Junbin Gao, and Kevin Burrage. Simulated maximum likelihood method for estimating kinetic rates in gene expression. *Bioinformatics*, **23**[1]:84–91, 2007. 4

[124] Tina Toni, David Welch, Natalja Strelkowa, Andreas Ipsen, and Michael P. H. Stumpf. Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. *Interface, Journal of the Royal Society*, **6**[31]:187–202, 2009. 5, 45

[125] Imbi Traat. Simulation and the Monte Carlo Method, 2nd Edition by Reuven Y. Rubinstein, Dirk P. Kroese. *International Statistical Review*, **77**:153–154, 2009. 5

[126] Yuanfeng Wang, Scott Christley, Eric Mjolsness, and Xiaohui Xie. Parameter inference for discretely observed stochastic kinetic models using stochastic gradient descent. *BMC Systems Biology*, **4**[1]:99, 2010. 5

[127] Roland Wedlich-Soldner and Rong Li. Spontaneous cell polarization: undermining determinism. *Nature Cell Biology*, **5**:267 EP –, Apr 2003. 2

[128] Darren J. Wilkinson. *Stochastic Modelling for Systems Biology*. CRC Press, 2006. 1, 2, 3, 5, 14, 15, 17, 18, 19, 22, 23, 24, 34

[129] Darren J Wilkinson. Stochastic modelling for quantitative description of heterogeneous biological systems. *Nature Reviews Genetics*, **10**[2]:122–133, 2009. 14

[130] D. Wilson and R. E. Baker. Multi-level methods and approximating distribution functions. *AIP Advances*, **6**[7]:075020, 2016. 74

[131] P. Wynn. On a device for computing the $e_m(s_n)$ transformation. *Mathematical Tables and Other Aids to Computation*, **10**[54]:91–96, 1956. 65, 123

[132] C Zechner, J Ruess, P Krenn, S Pelet, M Peter, J Lygeros, and H Koeppl. Moment-based inference predicts bimodality in transient gene expression. *PNAS*, **109**, 2012. 5

[133] Christoph Zimmer and Sven Sahle. Parameter estimation for stochastic models of biochemical reactions. *Journal of Computer Science & Systems Biology*, **6**[1]:11–21, 2012. 45, 47