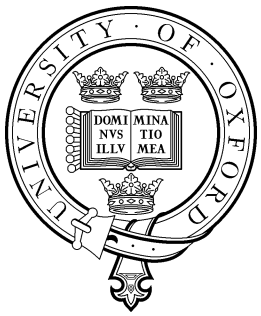
Bayesian Verification of Stochastic Models in Systems Biology



Gareth Wyn Molyneux

Wadham College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

Michaelmas 2020

Abstract

The accurate construction and verification of mathematical models from data in biology are paramount to the success of the fields of systems and synthetic biology. For systems biology, mathematical models are required to accurately model and provide further understanding of biological phenomena of interest, while for synthetic biology, such models are required for the efficient design and development of novel biological devices. In this thesis we integrate techniques from Bayesian methods, formal verification and machine learning to provide a framework that allows one to formally verify a property of interest on an underlying stochastic system with limited observed data. First we introduce this framework, titled Bayesian verification, which integrates likelihood-free inference techniques and probabilistic model checking to construct and verify mathematical models with accurate results, using sparse and noisy experimental data. Secondly, we then extend the generality and thus applicability of this framework by utilising statistical model checking and supervised machine learning, resulting in the Approximate Bayesian Computation - sequential Monte Carlo with Statistical model Checking (ABC-(SMC)^2) framework. With this extended Bayesian verification framework, with noisy and sparsely observed data, we can simultaneously learn and verify a model of interest, the only requirement being that we are able to generate simulations from a proposed model. Although this extension greatly extends the generality of the Bayesian verification framework, the number of simulations required to perform Bayesian verification becomes restrictive. Thirdly, because of the aforementioned, we present a multifidelity Bayesian verification framework. This multifidelity Bayesian verification framework reduces the computational cost of the simulations required to perform data-driven verification, and here we also seek to further reduce the number of simulations required by exploring when and how to perform statistical model checking. We conclude the thesis with a summary of our results and discuss directions for future work.

Acknowledgements

Even the most skilled of wordsmiths would not be able to capture how grateful and appreciative I am to all those that have supported me endlessly during my DPhil studies.

The first of many thanks goes to my supervisor, Professor Alessandro Abate, for granting me the academic freedom to pursue any ideas of interest and for providing support and guidance, frequently in the form of sage and oft needed advice. I have thoroughly enjoyed our thought provoking discussions over the past few years, ranging from academic pursuits and ideas to the exploits of the Mighty Clarets and everything in between. I wouldn't be in Oxford at all if not for the help of the directors of the Synthetic Biology CDT, Professor Antonis Papachristodoulou and Professor Robert Carlisle, the CDT being an excellent programme that enabled me to learn about topics which were previously unfamiliar. On this programme I also met lifelong friends in Jesús Siller Farfán, Lottie Hoskin and Thomas Meiller-Legrand, whos support and countless fun evenings have kept me sane during the toughest part of my studies.

During my time in the Department of Computer Science, I was a proud member of the Oxford Control and Verification (OxCAV) group, spearheaded by Professor Abate. I cannot thank the past and present members of the group enough for providing the much welcomed respite from my studies. I would like to extend my gratitude and thanks in particular to Viraj Brian Wijesuriya, Nathalie Cauchi and Andrea Peruffo for their entertaining conversations, wise words and all round high-jinks.

On the social side of my time during my DPhil studies, I would like to thank the many members of the Wadham College graduate community who have provided numerous fun evenings in the MCR. Regrettably the list of people who have made my time at Wadham College that much more enjoyable is far too long and have been necessarily omitted. I would like to extend particular thanks and acknowledge that I truly appreciate the

friendships I have formed with the people who have had the misfortune of spending a lot of time with me during my time at Wadham. To prove that names were not picked in a particular order, the following list of people I would like to thank have been generated randomly; I would like to thank Rory Gregson, Jonny Warby, George Webster, Joel Butler, Kathi Lawall, Sarah-Beth Amos, Jacob Bakermans, Leonie Maurer, Elise Hamerslag, Luana De Borst, Panarat Anamwathana, Richard East and Marianne van der Vaart.

Evidenced by the length of this acknowledgements section, I have been blessed enough to encounter and befriend many brilliant people during my time at Oxford, especially during my time at Wadham College. I have been endlessly supported by brilliant people throughout my life and with lifelong friends from the days of Ysgol Uwchradd Bodedern, I would like to thank Dafydd Jones, Kieran Desmond, Aled Hughes and David Jones.

I would like to thank Brooke Prakash for her unbridled enthusiasm for the small things in life and for being the bright Texan sunshine that a gloomy and grey clouded Welshman like myself often needs.

Finally, I would like to thank my siblings, Rhiannon, Lois, Gethin, Alun and Catrin, especially my parents, Philip and Delyth, who have supported me unconditionally for my entire life. Without their support and love I wouldn't have achieved half of what I have managed to do.

Diolch am bopeth.

Contents

1 Introduction **1**

1.1 Overview of the Thesis . . . 2
1.2 Contributions . . . 3
1.2.1 Publications . . . 4

2 Background **5**

2.1 Technical Preliminaries . . . 6
2.1.1 Conventions . . . 6
2.1.2 Notation . . . 6
2.1.3 Probability Theory . . . 6
2.1.4 Mathematical Modelling in Biology . . . 8
2.1.5 Deterministic versus Stochastic Modelling . . . 8
2.2 Stochastic Models . . . 10
2.2.1 Markov Processes . . . 10
2.2.2 Stochastic Differential Equations . . . 14
2.3 Chemical Reaction Networks . . . 16
2.3.1 Chemical Master Equation . . . 18
2.4 Stochastic chemical kinetics . . . 21
2.4.1 Stochastic Simulation Algorithm . . . 21
2.4.2 Chemical Langevin Equation . . . 24
2.5 Bayesian Inference . . . 26
2.6 Formal Verification . . . 29
2.6.1 Defining Properties . . . 30
2.6.2 Probabilistic Model Checking . . . 32
2.6.3 Statistical Model Checking . . . 33
2.7 Related work . . . 35

3 Bayesian Verification of Chemical Reaction Networks **41**
3.1 Introduction . . . 41
3.2 Bayesian Verification Calculation . . . 42
3.2.1 Parameter Synthesis . . . 43
3.2.2 Likelihood-free Bayesian Inference . . . 44
3.2.3 Credibility Computation . . . 53
3.3 Bayesian Verification Case Study . . . 53
3.3.1 Case Study: Finite-state SIR Model . . . 54
3.4 Initial challenges of Bayesian verification . . . 58

4 Simultaneous Inference and Verification **62**
4.1 Extending Bayesian Verification . . . 63
4.1.1 Statistical Model Checking with the Massart Algorithm . . . 64
4.2 Integrating formal verification and inference . . . 65
4.2.1 ABC-$(SMC)^2$ Algorithm . . . 66
4.2.2 Approximate Parameter Synthesis via Statistical MC . . . 67
4.3 Application of the ABC-$(SMC)^2$ Algorithm . . . 72
4.4 Discussion of ABC-$(SMC)^2$ . . . 75

5 Multifidelity Bayesian Verification **80**
5.1 Bayesian Statistical Model Checking . . . 81
5.2 Multifidelity Methods . . . 88
5.2.1 Approximate Abstractions of Stochastic Systems . . . 96
5.3 Multifidelity Bayesian Verification . . . 99
5.3.1 MF-ABC-$(SMC)^2$ Algorithm . . . 100
5.3.2 Case studies . . . 103
5.4 Discussion of Multifidelity Bayesian Verification . . . 120

6 Conclusion and Future Work **123**
6.1 Future Work . . . 125

A ABC-$(SMC)^2$ Results and Massart Algorithm **127**
A.1 Absolute-Error Massart Algorithm . . . 127
A.2 ABC-$(SMC)^2$ SIR Case Study Results . . . 127

B Optimal Continuation Probabilities **130**
B.1 Multifidelity Approximate Bayesian Computation - Rejection Sampling 130
B.2 Multifidelity Approximate Bayesian Computation - Sequential Monte Carlo 132

Bibliography **135**

List of Figures

2.1 Example of a DTMC 11
2.2 Example of a CTMC 14
2.3 Modelling a CRN as a CTMC 23

3.1 Bayesian Verification Framework 43
3.2 Example model observation data for SIR case study 55
3.3 Parameter Synthesis for SIR case study using PRISM 56
3.4 Bayesian verification results for SIR case study 58
3.5 Traces generated from true and inferred model for SIR case study . . 61

4.1 ABC-$(\text{SMC})^2$ Framework 65
4.2 Parameter synthesis generated via ABC-$(\text{SMC})^2$ for SIR case study . 74
4.3 Bayesian verification results via ABC-$(\text{SMC})^2$ for SIR Case study . . 76
4.4 Estimated satisfaction probabilities for SIR case study 77

5.1 Frequentist versus Bayesian Statistical Model Checking 87
5.2 Inference results (via MF-ABC-$(\text{SMC})^2$) for production-degradation case study 107
5.3 Estimated satisfaction probabilities (via MF-ABC-$(\text{SMC})^2$) for production-degradation case study 108
5.4 Approximate abstraction assessment of production-degradation case study 109
5.5 Inference results (via MF-ABC-$(\text{SMC})^2$) for production-degradation case study when negative weights are present 110
5.6 Estimated satisfaction probabilities (via MF-ABC-$(\text{SMC})^2$) for production-degradation case study with negative weights 111
5.7 Inference results (via MF-ABC-$(\text{SMC})^2$) for SIR case study 113
5.8 Simulations generated from true $\mathcal{M}_{\theta_\star}$ and inferred (via MF-ABC-$(\text{SMC})^2$) model $\mathcal{M}_{\hat{\theta}}$ for SIR case study 114

5.9 Estimated satisfaction probabilities for HF model and LF model for SIR case study 115
5.10 Approximate parameter synthesis generated from posterior for SIR case study 116
5.11 Approximate parameter synthesis generated from prior for SIR case study 117
5.12 Neural network predicted satisfaction probabilities and parameter synthesis for SIR case study 118
5.13 Approximate abstraction metric results for SIR case study 119

List of Tables

3.1 Table of inference results for initial Bayesian verification framework . 60

4.1 Parameter synthesis runtimes 75
4.2 Estimated satisfaction probabilities and number of simulations required 77

5.1 Runtimes for MF-ABC-$(SMC)^2$ and MF-BSMC 104

A.1 Inferred posterior and Bayesian Verification Results for ABC-$(SMC)^2$ framework 128
A.2 Runtimes for ABC-$(SMC)^2$ framework 129

Abbreviations

PDF ..... Probability Distribution Function

CDF ..... Cumulative Distribution Function

r.v. ..... Random Variable

iid ..... independent and identically distributed

s.t. ..... Such that

w.r.t ..... With respect to

DTMC ..... Discrete-Time Markov chain

(p)CTMC ..... (parametrised) Continuous-Time Markov Chain

(p)SDE ..... (parametrised) Stochastic Differential Equation

CRN ..... Chemical Reaction Network

CLE ..... Chemical Langevin Equation

PMC ..... Probabilistic Model Checking

SMC ..... Statistical Model Checking

CSL ..... Continuous Stochastic Logic

ABC ..... Approximate Bayesian Computation

ABC-SMC ..... Approximate Bayesian Computation - Sequential Monte Carlo

ABC-(SMC)^2 ..... Approximate Bayesian Computation - sequential Monte Carlo with Statistical Model Checking

MF-ABC ..... Multifidelity Approximate Bayesian Computation

MF-ABC-SMC ... Multifidelity Approximate Bayesian Computation - sequential Monte Carlo

MF-ABC-(SMC)^2 Multifidelity Approximate Bayesian Computation - sequential Monte Carlo with Statistical Model Checking

List of Symbols

$\mathcal{M}$: Stochastic model / High-fidelity model / CTMC, a DTMC is denoted by $\mathcal{M}^D$

$\mathcal{X}$: Set of states

x_0 : Initial state of stochastic model $\mathcal{M}$, x_i is the ith state of the model

$x \in \mathcal{X}$: States of CTMC

t_i : Time spent in state x_i before jumping to the next state

$\mathcal{T}(x, x')$: Transition rate matrix from state x to x'

L : Labelling function

$\mathbf{T}$: Transition probability matrix for a DTMC

ω_∞ : Infinite path of a CTMC

ω : Finite path of a CTMC

$Path^{\mathcal{M}}(x)$: Set of finite paths of CTMC starting from x

$\theta \in \Theta \subset \mathbb{R}^k$: Parameters in parameter space Θ with dimensionality k

$\mathcal{M}_\theta$ or $p(y \mid \theta)$: Parametrised stochastic model $\mathcal{M}$ or parametrised CTMC

$(\Omega, \mathcal{F}, P)$: Probability space, Ω sample space, $A \in \mathcal{F}$ and $\omega \in \Omega$ event or outcome

$W = (W_t)_{t \geq 0}$: Brownian motion

$\mathcal{N}(\mu, \sigma^2)$: Normal distribution with mean μ and variance σ^2

$\tilde{\mathcal{M}}$: Low-fidelity stochastic model or SDE model

$\tilde{\mathcal{M}}_\theta$ or $\tilde{p}(\tilde{y} \mid \theta)$: Parametric low-fidelity stochastic model or pSDE model

$\tilde{\mathcal{X}}$: State space of SDE

$(\tilde{x}_t)_{t\geq 0}$: Stochastic process on continuous space $\tilde{x}_t \in \tilde{\mathcal{X}} \subset \mathbb{R}^d$

$\varsigma(\theta, x)$: Functions of a given SDE

T : Simulation time

τ_k : Sampling times $k \in \mathbb{N}$

Δt : Timestep for numerically solving SDE

$\mathcal{CRN}$: Chemical Reaction Network

N_S : Number of chemical species in a given CRN

N_R : Number of reactions in a given CRN

$\mathcal{R}$: $= \{R_1, R_2, \ldots, R_{N_R}\}$ set of the N_R chemical reactions

X : State vector $X = [X_1, \ldots, X_{N_S}]^\top$ of CRN , X_i denotes number of molecules of species i present in system

$\Bbbk$: Vector of kinetic parameters in CRN, $\Bbbk \subset \mathbb{R}^{N_R}$

$\upsilon_{i,j}^-$ $(\upsilon_{i,j}^+)$: Stoichiometric coefficients of the reactants (products)

$\upsilon_{i,j}$: $= \upsilon_{i,j}^+ - \upsilon_{i,j}^-$, stoichiometric matrix

$\boldsymbol{\upsilon}_j$: $= [\upsilon_{1,j}, \upsilon_{2,j}, \ldots, \upsilon_{N_S,j}]^\top$ jth column of stoichiometric matrix $\upsilon_{i,j}$

$a_j(X)$: Propensity function for reaction j

$X(t)$: CRN configuration of X at time t

$\mathcal{K}_j(X(t), \mathrm{d}t)$: number of R_j reactions that occur over $[t, t + \mathrm{d}t]$ given $X(t)$

$\mathcal{P}_j(a_j, \mathrm{d}t)$: Poisson random variable

n^R : number of reactions that occur within a time interval $[t, t + \mathrm{d}t]$

$\tilde{x}(t)$: State vector of a CLE, where $\tilde{x} \in \mathbb{R}^d$

$W^j(t)$: independent scalar Brownian motions

ϕ : Property or CSL formula

φ : Formula over paths

ζ : Probability threshold within ϕ property

$\phi_1 U^{[t,t']} \phi_2$: Until operator

$\Lambda_\phi(\theta)$: Satisfaction probability

y_{obs} : Observations (data) from CRN

y : Simulation from high-fidelity model, $\mathcal{M}_\theta$ or CTMC

$\tilde{y}$: Simulation from low-fidelity model, $\tilde{\mathcal{M}}_\theta$ or SDE

$\pi(\theta)$: Prior probability distribution

$p(y \mid \theta)$: Likelihood function

$\pi(\theta \mid y_{obs})$: Posterior probability distribution

$p(y_{obs})$: Evidence

$\pi_{ABC}(\theta \mid y_{obs})$: ABC approximation to the posterior probability distribution

$y \sim p(y \mid \theta)$: Simulation Traces generated from a parametrised stochastic model $\mathcal{M}_\theta$

$y_{obs} = O(X(t))$: Observations from system, where O is arbitrary observation function on $X(t)$

$d(y, y_{obs})$: Distance metric between observations, used for inference

$s = S(y)$: Summary statistics, similarly for s_{obs}

$h \in \mathbb{R}_{\geq 0}$: Distance threshold for ABC

$h_m \in \mathbb{R}_{\geq 0}$: Distance for threshold m for ABC-SMC algorithm

$\delta_x(A)$: Dirac delta measure, $\delta_x(A) = 1$ if $x \in A$, 0 otherwise

$\mathbb{1}(\cdot)$: Indicator function

Ω_h : $\Omega_h(y_{obs}) = \Omega_h = \{y \in \mathcal{Y} \mid d(y, y_{obs}) \leq h\}$, distance neighbourhood where y is at most a distance h from y_{obs}

$\mathcal{Y}$: Output space of the simulations $y \sim p(y, \mid \theta)$

$K_h(d(y, y_{obs}))$: $= \mathbb{1}_{\Omega_h}(y)$, standard smoothing kernel function for ABC

$f_m(\theta)$: $= \pi_{ABC}^{h_m}(\theta, s \mid s_{obs}) \propto \mathbb{1}_{\Omega_{h_m}}(s)p(y \mid \theta)\pi(\theta)$ intermediary distributions for ABC-SMC scheme

N : Number of Monte Carlo samples to take from ABC/ABC-SMC/MF-ABC-SMC

M : Number of thresholds in ABC-SMC

λ : Rate at which thresholds h_m decrease

$\hat{q}(\theta) = q(\theta)/Z_q$: $\hat{q}$ is the importance distribution proportional to $q(\theta)$, Z_q is the normalisation constant

B_m : Number of simulations to generate for each sampled parameter in ABC-SMC scheme

$F_m(\theta^* \mid \theta)$: Perturbation kernel in ABC-SMC

$s_b = S(y_b)$: Summary statistics for simulation y_b

b_m : Summation of $K_h(u)$

h_m : Distance for threshold m

$w_m^{(i)}$: Weight of ith sampled particle, $\theta_m^{(i)}$ in mth threshold

$\tilde{y}$: Low-fidelity simulation

$\tilde{\cdot}$: in general is the low-fidelity equivalent of many things defined previously

$\hat{\cdot}$: in general are estimates of the true function or variable $(\cdot)$

$c(\theta)$ $(\tilde{c}(\theta))$: Computational cost of high-fidelity (low-fidelity) model

$\tilde{w}$: Low-fidelity weight

w_{mf} : Multifidelity weight

$U_j \sim \mathcal{U}(a, b)$: Sample from Uniform distribution defined on $[a, b]$

$\eta(\tilde{y})$: Continuation probability

$g(\theta)$: Arbitrary function on the parameters

$\{\theta^{(i)}, w_{MF}^{(i)}\}_{i=1}^{N}$: Weights from MF-ABC

$r_m(\theta)$: Importance distribution with non-negative weights for MF-ABC-SMC

$\varpi(\theta)$: Function that the MF-ABC-SMC framework targets

$d_T(\cdot, \cdot)$: Distance metric chosen for approximate abstraction metric, γ_ε

ε : Error for approximate abstraction

γ_ε : Approximate abstraction metric

z : Number of realisations to be discarded for approximate abstraction metric

j_i : Indices of discarded simulations

β_γ : Confidence parameter for approximate abstraction metric

ϵ : Error parameter in SMC

$\hat{\Lambda}_\phi(\theta)$: Satisfaction probability estimate

δ : $(1-\delta)$-confidence interval for SMC

$n_{\mathcal{O}}$: Number of simulations required for Okamoto bound

$C(\Lambda_\phi(\theta), I)$: Coverage of $\Lambda_\phi(\theta)$ by confidence interval I

$h_a(\Lambda_\phi(\theta), \epsilon)$: Massart bound

δ_c : Coverage parameter in Massart algorithm

V_i : Bernoulli random variable

α, β : Beta prior parameters

$\mathcal{B}(\alpha, \beta)$: Beta function

$F_{(\alpha,\beta)}(u)$: CDF for Beta distribution

c : Bayesian SMC coverage parameter

(l_0, l_1) : Upper and lower limits for integral associated with c, the coverage parameter for the Bayesian SMC algorithm

ℓ : Half-interval width for Bayesian SMC

$\mathcal{C}$: Credibility calculation for Bayesian verification

$\bar{N}$: Number of total samples taken in ABC-(SMC)^2

n : Number of simulations required for SMC

$\mathcal{C}_{h_m,N,n}$: Credibility calculation obtained from ABC-(SMC)^2 and MF-ABC-(SMC)^2

ϵ_{MC} : Monte Carlo error in ABC-(SMC)^2 scheme

ϵ_{SMC} : SMC error in ABC-(SMC)^2 scheme

ϵ_{ABC} : ABC error in ABC-(SMC)^2 scheme

$\hat{\theta}_m$: Posterior mean

$\tilde{n}$: number of LF simulations required for SMC

$N_{\mathcal{O}}$: number of observations taken for y_{obs}

$\bar{t}_k$: The times at which observations y_{obs} are taken

$\theta_\star$: $\theta_\star = [k_1^\star, k_2^\star, \ldots, k_{N_R}^\star]^\top$, true parameters we wish to infer

c_p (estimate, $\hat{c}_p$)) : Computational cost of positive cases

c_n (estimate, $\hat{c}_n$)) : Computational cost of negative cases

p_{tp} : True positive probability

p_{fp} : False positive probability

p_{fn} : False negative probability

(η_1^*, η_2^*) : Optimal continuation probabilities

Chapter 1

Introduction

The motivation for mathematical modelling of biological systems is to both improve quantitative understanding of biological systems (systems biology) or to design novel biological devices to obtain new and complex cellular behaviours in a predictive, robust and reliable fashion through the design and construction of synthetic biological parts (synthetic biology). Synthetic biology in particular aims to streamline the design and construction of novel biological devices by adhering to the principles found in many different areas of engineering. Within engineering, the formalism of mathematics is essential to achieve pure precision when designing novel devices. The problem with applying the current principles of engineering to biology is that most biological systems behave more obstreperously than typical engineered systems, and thus require a further detailed analysis. Many systems that undermine biological processes are large complex biochemical networks that are inherently stochastic, making the construction of quantitative models that accurately predict the dynamics of a given biological system a challenge in itself. In particular, given the inherent stochasticity within biochemical systems, any mathematical analysis or design of novel biological systems must consider these dynamics. Validating that a novel biological device works as designed is a challenge in itself, as biological experiments are not only expensive and time consuming, but the data observed is noisy and is often times only a partial observation of the underlying stochastic dynamics.

The aim of this thesis is to tackle some of the mathematical modelling challenges that face synthetic and systems biology with the introduction of a data-driven formal verification framework. Whilst synthetic biology has been motivated by principles found in engineering, we tackle these challenges using rigorous mathematical techniques, as is done in systems biology, and methods developed in computer science, specifically formal verification. We integrate data-driven methods, such as Bayesian inference and machine learning, with formal verification techniques to construct and

verify stochastic biochemical models that describe the underlying true biological system. This leads to very challenging and interesting research questions regarding the fields of systems and synthetic biology:

- With limited and partially observed data, how can we construct an accurate stochastic model that represents an underlying stochastic biological system?
- Given limited data produced from a constructed novel biological device, can we formally verify that the novel biological device achieves what is it designed to do?
- With a given property in mind, can we synthesize a novel biological system that satisfies this property?

In alignment with these questions, the work in this thesis composes of:

- The introduction of a framework that integrates Bayesian inference and formal verification, applied to chemical reaction networks, that allows the data-driven formal verification of certain stochastic biological models (**Chapter 3**).
- The extension of the above framework to incorporate supervised machine learning methods to improve the scalability and generality of the framework and a novel parameter synthesis technique, allowing the framework to be applied to any stochastic data-generating system (**Chapter 4**).
- The deployment of a multifidelity data-driven formal verification technique (**Chapter 5**).

Whilst the framework was designed with the challenges of systems and synthetic biology in mind, the framework allows for the formal verification of any data-generating system, and is not limited to the fields of synthetic and systems biology.

1.1 Overview of the Thesis

This thesis presents a data-driven verification framework that can be applied to a wide variety of parametrised stochastic models, with the only prerequisite being the ability to generate simulations of the model. The introduction of this framework aims to address the research questions introduced in the previous section and we present a brief overview of the structure of the thesis:

- We begin by covering the necessary background material in **Chapter 2**, where we introduce stochastic models, chemical reaction networks, Bayesian inference and the formal verification techniques. We conclude this section with a discussion of related works.

- In **Chapter 3** we present and explore a likelihood-free inference technique that allows us to infer models with partially and scarce data-sets, integrating this technique with a numeric based parameter synthesis technique to introduce the Bayesian verification framework.

- We address the limitations of the numeric based parameter synthesis technique in **Chapter 4** by introducing an alternative statistical based parameter synthesis technique that utilises supervised machine learning. Doing so improves both the generality and scalability of the Bayesian verification technique, allowing it to be applied on a wider variety of stochastic models and systems.

- Given that the framework introduced in Chapter 4 relies on a high-number of simulations, in **Chapter 5** we deploy multifidelity methods within the inference and statistical model checking techniques to decrease the total number of simulations required and the computational burden of generating stochastic simulations. We also introduce a multifidelity statistical model checking algorithm that allows one to formalise a relationship between the high-fidelity model and the low-fidelity model.

- We conclude the thesis in **Chapter 6**, summarising our results and presenting further avenues of research related to our Bayesian verification technique.

1.2 Contributions

In line with the major research questions and the work undertaken in this thesis, we summarise the main contributions of this thesis:

- We introduce a data-driven formal verification technique that allows one to simultaneously infer a parametrised stochastic model from noisy or partially observed data and formally verify a property of interest. This allows one to formally verify properties of interest on a variety of data-generating stochastic systems, not just biological models, by utilising this data to calculate the probability that the system satisfies a property given data.

- By addressing the limitations of parameter synthesis based on probabilistic model checking techniques, we introduce an alternative but scalable parameter synthesis technique using supervised machine learning methods, specifically support vector machines. With the introduction of this technique, we are able to perform a parameter synthesis on any given parametrised stochastic model and a property of interest, as long as we are able to generate simulations from this parametrised stochastic model.

- We deploy multifidelity techniques into our Bayesian verification framework and we introduce a multifidelity statistical model checking algorithm - with this algorithm we are able to further formalise the relationship between a chosen high-fidelity model and its low-fidelity equivalent. We also explore how multifidelity techniques can be used within the realms of formal verification.

1.2.1 Publications

The contributions of this thesis have been published in conference proceedings and have been peer-reviewed. Specifically, the following publications have been covered in chapters of this thesis:

- (**Chapter 3**) Molyneux G.W., Wijesuriya V.B., Abate A. (2020) Bayesian Verification of Chemical Reaction Networks. In: Sekerinski E. et al. (eds) Formal Methods. FM 2019 International Workshops. FM 2019. Lecture Notes in Computer Science, vol 12233. Springer, Cham. https://doi.org/10.1007/978-3-030-54997-8_29

- (**Chapter 4**) Molyneux G.W., Abate A. (2020) ABC(SMC)^2: Simultaneous Inference and Model Checking of Chemical Reaction Networks. In: Abate A., Petrov T., Wolf V. (eds) Computational Methods in Systems Biology. CMSB 2020. Lecture Notes in Computer Science, vol 12314. Springer, Cham. https://doi.org/10.1007/978-3-030-60327-4_14

We also aim to submit the final piece of work of this thesis (**Chapter 5**) to a peer-reviewed publication. I was the leading author of all the work presented in this thesis - in the paper that is covered by Chapter 3, I collaborated with a colleague (and good friend of mine, Brian) on the statistical model checking via hypothesis testing and the numerical evaluation of the parameter synthesis.

Chapter 2

Background

In this section we will discuss all the necessary background material required for the remaining chapters of the thesis. We first discuss the motivation of working with stochastic models in the realms of systems and synthetic biology and the advantages of doing so. We explain how biological models can be described using biochemical networks and subsequently turn our focus on deriving stochastic models for biochemical networks of interest. To derive these stochastic models we first discuss basic ideas within probability that are essential in the construction of stochastic processes. A brief tutorial as to how to simulate a stochastic model of interest is then given before turning our focus to Markov processes, a subset of stochastic models that are of interest to the biological community. We explain how stochastic chemical kinetics can be derived for these Markov processes and discuss how these Markov processes can be extended to continuous-state stochastic differential equations, namely the Chemical Langevin equation.

There is great interest in the data-driven construction of these stochastic models within biology; in particular, Bayesian methods such as parameter inference are vastly popular. In this section we first introduce the notion of Bayesian inference, while later in the thesis we focus on Bayesian methods that are more appropriate for the type of data and models we work with. For these learned models, we rely upon the techniques of formal verification to rigorously model check and perform further detailed analysis of these stochastic models. To formally verify properties of interest, we rely on two popular methods of model checking, probabilistic model checking and statistical model checking. We introduce these two forms of model checking and then conclude the background with a review of the related literature regarding Bayesian methods, formal verification and the applications of these two fields within the realms of biology. We consider the motivation behind constructing mathematical models in biology and delve deeper as to why we are more focused on stochastic as opposed to

deterministic models of interest. But first, we highlight the technical preliminaries that form the basis for this thesis.

2.1 Technical Preliminaries

2.1.1 Conventions

$\mathbb{Z}$ - set of integers

$\mathbb{N}$ - set of natural numbers (0 inclusive)

$\mathbb{Z}^d$ - set of d-dimensional vectors having integer entries

$\mathbb{R}$ - set of real numbers

$\mathbb{R}^d$ - set of d-dimensional vectors having real entries

$[x]$ - the integer part of the real number x

$\mathbb{E}(X)$ - expected value of a random variable X

$\mathrm{Var}(X)$ - variance of a random variable X

$\mathrm{Cov}(X, Y)$ - covariance between random variable X and random variable Y

2.1.2 Notation

We denote the set of real numbers by $\mathbb{R}$, the set of integers $\mathbb{Z}$ and the set of natural numbers $\mathbb{N}$ (inclusive of 0) and denote the empty set as $\emptyset$. If $x \in \mathbb{R}$ is a real number on the closed interval $\{x \in \mathbb{R} | a \leq x \leq b\}$, we denote this as $x \in [a, b]$. Similarly, if $x \in \mathbb{R}$ is a real number on the open interval $\{x \in \mathbb{R} | a < x < b\}$ we denote this as $x \in (a, b)$. We also define $[a, b) = \{x \in \mathbb{R} | a \leq x < b\}$ as a left-closed, right-open interval while $(a, b] = \{x | a < x \leq b\}$ is a left-open, right-closed interval. Given a set of numbers, we use subscripts to denote the restriction to the non-negative fragments of a set, for example $\mathbb{Z}_{\geq 0} = \mathbb{N} := \{0, 1, \dots\}$ or $\mathbb{R}_{\geq 0} = [0, \infty)$ or $\mathbb{R}_{> 0} = (0, \infty)$.

2.1.3 Probability Theory

We now define some essential concepts from probability theory needed for the material in this thesis

Definition 2.1.1 (Probability Measure and Space)**.** Let Ω be a non-empty sample space and let $\mathcal{F}$ be a σ-algebra on Ω, where $A \in \mathcal{F}$ is an event. Then $(\Omega, \mathcal{F})$ is a measurable space and a map $P : \mathbb{F} \to [0, 1]$ is called a *probability measure* on $(\Omega, \mathcal{F})$ if

- $P(\Omega) = 1, P(\emptyset) = 0$,

- $\forall A_i \in \mathcal{F}$ with $A_i \cap A_j = \emptyset \ \forall j \neq i \implies P\left(\bigcup_{i=1}^{\infty} A_i\right) = \sum_{i=1}^{\infty} P(A_i)$.

Then we call $(\Omega, \mathcal{F}, P)$ a probability space

If two events A, B are independent, then $P(A \cap B) = P(A)P(B)$.

Definition 2.1.2 (Theorem of Total Probability)**.** Given a partition $\{B_1, \ldots, B_n\}$ of a sample space Ω and an event A, the theorem of total probability states

$$P(A) = \sum_{i=1}^{n} P(A|B_i)P(B_i). \tag{2.1}$$

We define a random variable (r.v.) as

Definition 2.1.3 (Random Variable)**.** Let $X : \Omega \to \mathbb{R}^d$ be measurable, i.e., for every Borel measurable set B of $\mathbb{R}^d$,

$$X^{-1}(B) := \{\omega : X(\omega) \in B\} \in \mathcal{F}.$$

Then X is called a *random variable* (r.v.) or a *random vector* if $d > 1$.

For a random variable X, its *mean* (or *expectation*) is given by

$$\mathbb{E}[X] := \int_{\Omega} X(\omega) dP(\omega).$$

Similarly, if $X, Y \in \mathcal{L}^2(\Omega, \mathcal{F}, P)$, then we define the covariance between r.v.s X, Y as

$$\mathrm{Cov}(X, Y) := \mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])],$$

where the variance of X is defined as $\mathrm{Var}(X) = \mathrm{Cov}(X, X)$. We note that if X and Y are independent random variables, then $\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y]$. For discrete random variables taking values in $\mathbb{N}$, the mean is instead evaluated using a sum, $\mathbb{E}[X] = \sum_{k \in \mathbb{N}} kP(X = k)$.

Definition 2.1.4 (Cumulative Distribution Function)**.** Let $X \equiv (X_1, X_2, \ldots, X_d)^\top$ be a r.v. The cumulative distribution function (CDF) is defined as

$$F(x) \equiv F(x_1, x_2, \ldots, x_d) := P(\{\omega : X_i(\omega) \leq x_i, i = 1, 2, \ldots, d\}).$$

Definition 2.1.5 (Probability Density Function)**.** If there exists a function $f \in \mathcal{L}^1(\mathbb{R}^d)$ s.t.

$$F(x) = \int_{-\infty}^{x_d} \cdots \int_{-\infty}^{x_1} f(y_1, \ldots, y_d) dy_1 \ldots dy_d,$$

then the function f is called the probability density function (PDF) of F.

2.1.4 Mathematical Modelling in Biology

The aim of mathematical modelling is to formulate a mathematical description of real world processes which is amenable to further theoretical and numerical analysis, providing a further insight and a deeper understanding of the systems that we are interested in. A mathematical model should be sufficiently detailed and precise in order to correctly capture the important properties and dynamics of a system of interest - doing so would allow one to investigate further properties through model simulation for example. The first issue that commonly arises when modelling systems is to decide exactly what level of detail the model will capture. This issue is exemplified within the realms of biology where biologists are used to thinking about biological processes on different scales - a model that describes the dynamics of an individual cell accurately is unlikely to be appropriate to accurately describe an entire organism for example. A good mathematical model is a model that captures the essential features of the system without superfluous detail, providing one with a quantitative and accurate understanding of an underlying system.

Model validation is done by comparing the mathematical model that has been derived with observed experimental data and assessing whether the mathematical model captures the properties of the underlying experimental data with a high degree of accuracy. Once we are content with the model, these models are used predictively to ascertain further properties of interest or uncover new discoveries through rigorous analysis and simulation. In the biological world, and indeed many areas of science, these simulations are a viable alternative to pursuing further experiments, either because these experiments are difficult to perform or conducting the experiment is too costly, either with regards to time or finances. Whilst the goal of systems biology is to construct an accurate biological model to allow for further analysis, in synthetic biology mathematical models are integral to the foundational engineering principles for the accurate design and construction of novel biological devices - different fields of course make use of different models to satisfy the specific requirements of that field. Whilst there are a variety of considerations that need to be taken into account when developing a model, one of the most fundamental decisions is whether the model behaves deterministically or stochastically.

2.1.5 Deterministic versus Stochastic Modelling

An important aspect that needs to be considered when developing a mathematical model is whether the mathematical model is a deterministic model or a stochastic

model. In deterministic models, the output of the model is fully determined by the parameter values and initial conditions, whereas stochastic models, by their nature, are inherently random - no two simulations of a stochastic model should be the same.

Stochastic Model: A stochastic model is a mathematical model that has some inherent randomness - the same set of parameter values and initial conditions will yield different outcomes.

Deterministic Model: A deterministic model is a mathematical model that produces the same outcomes when the same set of parameter values and initial conditions are provided - the dynamics of the system only changes when either of these are altered.

The decision as to whether a deterministic or stochastic model is appropriate depends on the system that we are interested in. Within biology, deterministic models are typically favoured and utilised in many aspects of the field but it is well established that at the intra-cellular level the biological processes are stochastic by nature [116]. Whilst many deterministic models are based on mass action principles, when we focus on systems involving a small number of molecules then the stochastic nature of fluctuations must be taken into consideration. At the cellular level, when low number of molecules are present, stochastic fluctuations play a major role in the underlying dynamics; for example gene expression [2, 7, 37, 79, 95] has been well established to behave obstreperously, where both internal and external factors cause these stochastic behaviours [103]. Stochastic models allow for a further level of detail that can capture properties that many deterministic systems cannot, such as stochastic bistability [105, 106]. Therefore, key cellular functions cannot be properly understood without building this inherent stochasticity into these mathematical models.

In order to understand such stochastic processes and their behaviour, we typically require many realisations of the process to understand not only their average behaviour but overall variability. The nature of these stochastic realisations also resonate with the nature of experimental data that is often encountered in biology - both stochastic simulations and experiments are repeated in order to identify a trend of the underlying dynamics. Allowing for these stochastic effects in our models presents us with the ability to measure the overall volatility of the system and perhaps utilise this additional information to explain dynamics that are otherwise unaccounted for. For the remainder of this thesis, we restrict our attention to working with stochastic models, focusing on examples of biochemical network models that are known to be inherently stochastic at the cellular level. To describe biological and biochemical

systems, we resort to chemical reaction networks (CRNs), but first, we introduce the stochastic models that are used to describe these stochastic biochemical systems.

2.2 Stochastic Models

A stochastic process is a random variable which evolves through time. The random variable may be on a discrete or a continuous space and can evolve through time in a discrete or continuous way. We will consider three different stochastic processes that evolve through time and space in different ways. We first consider processes that are discrete in both time and space, specifically discrete-time Markov chains. Following this, we consider processes that evolve through time in a continuous manner, but evolve in a discrete-space manner, namely continuous-time Markov chains. Finally we discuss stochastic processes that evolves through time and space in a continuous manner, namely stochastic differential equations.

2.2.1 Markov Processes

We consider first two stochastic models that evolve through time in a discrete or continuous way but remain on a discrete state space. Specifically, we concern ourselves with discrete-time and continuous-time Markov processes. A Markov process is a stochastic process that has the defining feature that the future behaviour of the stochastic process depends only on the current state of the stochastic system - this property is known as the *memoryless property*. Whilst Markov processes are particularly amenable to theoretical and computational analyses, we do not provide a detailed discussion here and refer the interested reader to analysis and insight into Markov processes from a formal verification perspective of Markov Processes [10] and similar insights from a systems biology perspective [116]. We first define a discrete-time, discrete-state stochastic process, discrete-time Markov chains.

Discrete-time Markov Chains

Definition 2.2.1 (Discrete-time Markov Chain)**.** A discrete-time Markov chain (DTMC) is a tuple $\mathcal{M}^D = (\mathcal{X}, \mathbf{T}, AP, L, x_0)$, where;

- $\mathcal{X}$ is a finite, non-empty set of states,
- x_0 is the initial state of the DTMC,

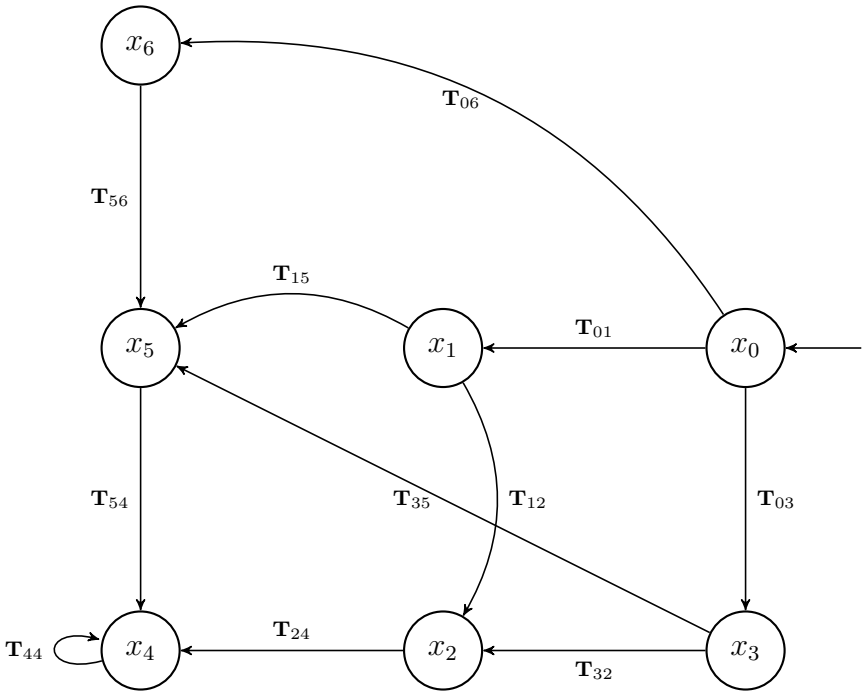


Figure 2.1. Example of a DTMC where transition probabilities are denoted as $\mathbf{T}(X_i, X_j) = \mathbf{T}_{ij}$ and x_i are states of the discrete-time Markov chain.

- $\mathbf{T} : \mathcal{X} \times \mathcal{X} \to [0, 1]$ is a transition probability matrix such that $\sum_{x' \in \mathcal{X}} \mathbf{T}(x, x') = 1$, $\forall x \in \mathcal{X}$, that is, $\mathbf{T}(x, x')$ is the probability of transitioning from state x to state x',
- $L : \mathcal{X} \to 2^{AP}$ is a labelling function mapping each state, $x \in \mathcal{X}$, to the set $L(x) \subseteq AP$ of atomic propositions AP, that hold true in x.

For the models in this thesis, we assume x_0 is unique and deterministically given. We define the behaviour of DTMCs with the use of its execution paths, defined below

Definition 2.2.2 (Path of a DTMC)**.** Let $\mathcal{M}^D = (\mathcal{X}, \mathbf{T}, AP, L)$ be a DTMC. An infinite path of a DTMC $\mathcal{M}^D$ is a non-empty sequence $x_0 x_1 x_2 \ldots$ where $\mathbf{T}(x_i, x_{i+1}) > 0$ for all $i \geq 0$. A finite path is a sequence $x_0 x_1 \ldots x_{K-1} x_K$ such that x_K is absorbing, that is, $\mathbf{T}(x_K, x') = 0$ for all $x' \in \mathcal{X}$ where $x' \neq x_K$. We denote by $Path^{\mathcal{M}^D}(x)$ the set of all (infinite or finite) paths of the DTMC $\mathcal{M}^D$ starting in state x. A trace of a DTMC is the mapping of a path through the labelling function L.

In Figure 2.1 we present an example of a DTMC with seven states and transition probabilities denoted as $\mathbf{T}_{ij} = \mathbf{T}(x_i, x_j)$. The Markov process described by a

DTMC is appropriate when transitions in the state space occur at predetermined discrete time steps. For many applications it is preferable to use a continuous model of time, where delays between a process transitioning from one state to another is done at arbitrary times. Continuous-time Markov chains extend the above discrete-time Markov chain framework with real-time by modelling these delays according to an exponential distribution. Formally we define continuous-time Markov chains as follows.

Continuous-time Markov Chains

Definition 2.2.3 (Continuous-time Markov Chain)**.** A continuous-time Markov chain (CTMC) $\mathcal{M}$ is a tuple $(\mathcal{X}, R, AP, L, x_0)$, where;

- $\mathcal{X}$ is a finite, non-empty set of states,
- x_0 is the initial state of the CTMC,
- $\mathcal{T} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$ is the transition rate matrix, where $\mathcal{T}(x, x')$ is the rate of transition from state x to state x',
- $L : \mathcal{X} \to 2^{AP}$ is a labelling function mapping each state, $x \in \mathcal{X}$, to the set $L(x) \subseteq AP$ of atomic propositions AP, that hold true in x.

For the models in this paper, we assume x_0 is unique and deterministically given. The transition rate matrix $\mathcal{T}$ governs the dynamics of the overall model. A transition can only occur between states x and x' if $\mathcal{T}(x, x') > 0$ and the probability of this transition being triggered within t time-units is $1 - e^{-\mathcal{T}(x,x')t}$. If there is more than one possible state x' for which $\mathcal{T}(x, x') > 0$, the first transition to be triggered determines the next state of the CTMC.

The time that the CTMC remains in a state $x \in \mathcal{X}$ is governed by an exponential distribution with a rate parameter $E(x)$, which we denote as the exit rate, defined as $E(x) = \sum_{x' \neq x} \mathcal{T}(x, x')$. Thus, the probability of exiting state x in t time units is given by $\int_0^t E(x) \cdot e^{-E(x)t'} \, \mathrm{d}t'$, and the probability to transition from state x to another state x' in t time units is given by $\mathbf{T}(x, x') \cdot \int_0^t E(x) \cdot e^{-E(x)t'} \, \mathrm{d}t'$, where $\mathbf{T}(x, x')$ is the probability of transitioning from state x to state x'. The transition probability $\mathbf{T}(x, x')$ is defined similarly to that defined for DTMCs and for a CTMC we define $\mathbf{T} = \mathcal{T}(x, x')/E(x)$ if $E(x) \neq 0$, 1 if $E(x) = 0$ and $x = x'$, and 0 otherwise. Intuitively, a CTMC path can be generated as follows. In each state of the CTMC, x, the chain remains in this state for a time t, where t is drawn from the exponential distribution

parametrised by $E(x)$. At this time, the chain transitions to a state x' with probability $\mathbf{T}(x, x')$. Below we formally define paths of a CTMC.

Definition 2.2.4 (Path of a CTMC)**.** Let $\mathcal{M} = (\mathcal{X}, \mathcal{T}, AP, L)$ be a CTMC. An infinite path of a CTMC $\mathcal{M}$ is a non-empty sequence $\omega_\infty = x_0 t_0 x_1 t_1 \ldots$ where $R(x_i, x_{i+1}) > 0$ and $t_i \in \mathbb{R}_{>0}$ for all $i \geq 0$. A finite path is a sequence $\omega = x_0 t_0 x_1 t_1 \ldots x_{k-1} t_{k-1} x_k$ such that x_k is absorbing. The value t_i represents the amount of time spent in the state x_i before jumping to the next state in the chain, namely state x_{i+1}. We denote by $Path^{\mathcal{M}}(x)$ the set of all (infinite or finite) paths of the CTMC $\mathcal{M}$ starting in state x. A trace of a CTMC is the mapping of a path through the labelling function L.

Parametric CTMCs extend the notion of CTMCs by allowing transition rates to depend on a vector of model parameters, $\theta \in \Theta \subseteq \mathbb{R}^d$. The domain of each parameter θ_i is given by a closed bounded real interval describing the range of possible values, $[\theta_i^\perp, \theta_i^\top]$. The parameter space Θ is defined as the Cartesian product of the individual intervals, $\Theta = \bigtimes_{i \in \{1,\ldots,d\}} [\theta_i^\perp, \theta_i^\top]$, so that Θ is a hyperrectangle.

Definition 2.2.5 (Parametric CTMC)**.** Let Θ be a parameter space. A parametric Continuous-time Markov Chain (pCTMC) over θ is a tuple $(\mathcal{X}, \mathcal{T}_\theta, AP, L, x_0)$:

- $\mathcal{X}, x_0, AP$ and L are as in Definition 2.2.3, and
- $\theta = (\theta_1, \ldots, \theta_d)$ is the vector of parameters, taking values in a compact hyperrectangle $\Theta \subset \mathbb{R}^d_{\geq 0}$,
- $\mathcal{T}_\theta : \mathcal{X} \times \mathcal{X} \to \mathbb{R}[\theta]$ is the parametric rate matrix, where $\mathbb{R}[\theta]$ denotes a set of polynomials over $\mathbb{R}_{\geq 0}$ with variables $\theta \in \Theta$.

Given a pCTMC and a parameter space Θ, we denote with $\mathcal{M}_\Theta$ the set $\{\mathcal{M}_\theta, \theta \in \Theta\}$ where $\mathcal{M}_\theta = (\mathcal{X}, \mathcal{T}_\theta, AP, L)$ is the instantiated CTMC obtained by replacing the parameters in R with their valuation in θ. So a standard CTMC is induced by selecting a specific parameter $\theta \in \Theta$: the sampled paths of an instantiated pCTMC $\mathcal{M}_\theta$ are defined similarly to ω. We present a finite-state CTMC in Figure 2.2 noting that the DTMC in Figure 2.1 can be constructed from this finite-state CTMC.

Having presented Markov processes that represent a stochastic process that evolves in discrete time and discrete space (DTMCs) and a process that evolves in continuous time and discrete space (CTMCs, with parametrised equivalent pCTMC), we now present stochastic differential equations (SDEs), which are stochastic processes that evolve continuously in both time and space.

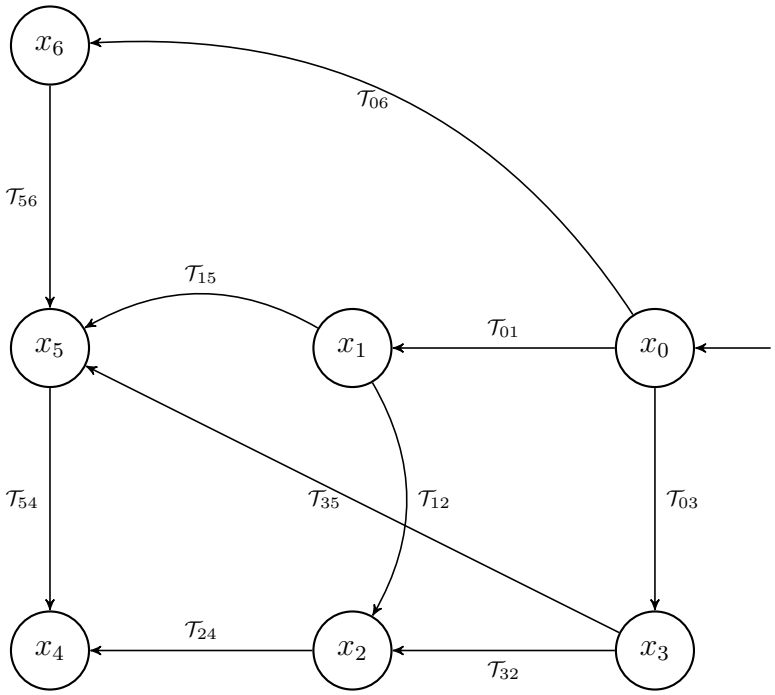


Figure 2.2. Example of a CTMC where edges $\mathcal{T}_{ij} = \mathcal{T}(x_i, x_j)$ are the *rates* of transition from state x_i to x_j in the CTMC, not the transition probabilities.

2.2.2 Stochastic Differential Equations

We now formally define SDEs, but in order to do so, we first define Wiener processes, often called Brownian motion.

Definition 2.2.6 (Wiener Process/Brownian Motion)**.** Let $(\Omega, \mathcal{F}, P)$ be a probability space where Ω is a sample space, $A \in \mathcal{F}$ an event in the sample space $\mathcal{F}$, and $\omega \in \Omega$ is a sample or outcome. A process $W = (W_t)_{t \geq 0}$ on $(\Omega, \mathcal{F}, P)$ taking values in $\mathbb{R}^d$ is called Brownian motion (BM) if

1. (Independent Increments) For any $0 \leq t_0 < t_1 < \cdots < t_n$, the random variables $W_{t_0}, W_{t_1} - W_{t_0}, \ldots, W_{t_n} - W_{t_{n-1}}$ are mutually independent,
2. (Normality) For any $t > s \geq 0$, the random variable $W_t - W_s \sim \mathcal{N}(0, (t-s)I)$,
3. (Continuity) $(W_t)_{t \geq 0}$ is continuous almost surely.

If $P(W_0 = 0) = 1$, then $(W_t)_{t \geq 0}$ is called a standard Brownian motion.

Wiener processes are the continuous-time stochastic processes that drive the stochastic nature of SDEs, which we can now define.

Definition 2.2.7 (Stochastic Differential Equation)**.** We define a stochastic process on the probability space $(\Omega, \mathcal{F}, P)$ as $(\tilde{x}_t)_{t\geq 0} : \mathbb{R}_{\geq 0} \times \Omega \to \mathbb{R}^d$ and we describe the evolution of this continuous stochastic process with the SDE defined by

$$\mathrm{d}\tilde{x}_t^i = \varsigma_0^i(\tilde{x}_t)\ \mathrm{d}t + \sum_{j=1}^{J} \varsigma_j^i(\tilde{x}_t)\ \mathrm{d}W_t^j,\ i = 1, 2, \ldots, d,$$

where $\tilde{x}_0 \in \mathbb{R}^d$ is the initial condition, $W_t = [W_t^1, \ldots, W_t^J]_{t\geq 0}^\top$ is a standard Brownian Motion in $\mathbb{R}^J$ on $(\Omega, \mathcal{F}, P)$ and

$$\varsigma_j^i : \mathbb{R}^d \to \mathbb{R},\ i = 1, 2, \ldots, d,\ j = 0, 1, 2, \ldots, J,$$

are Borel measurable functions under proper structural conditions, which ensure the existence and uniqueness of solutions to the SDEs [86].

We can similarly introduce a labelling map over the state space, as done for the case of CTMCs. We introduce the semantics of SDE over a discrete time-step: SDEs can be integrated numerically using various techniques, for instance via the known Euler-Maruyama scheme [68];

Definition 2.2.8 (Euler-Maruyama Method)**.** Given an SDE of the form

$$\mathrm{d}\tilde{x}_t = \varsigma_0(\tilde{x}_t)\ \mathrm{d}t + \sum_{j=1}^{J} \varsigma_j(\tilde{x}_t)\ \mathrm{d}W_t, \tag{2.2}$$

it can be numerically simulated over an increasing sequence of sampling times $\{\tau_k\}_{k\in\mathbb{N}}$ (finite or infinite) via the Euler-Maruyama scheme [58,68], which generates a stochastic *difference* equation of the following form:

$$\tilde{x}_{\tau_{k+1}} = \tilde{x}_{\tau_k} + \varsigma_0(\tilde{x}_{\tau_k})(\tau_{k+1} - \tau_k) + \sum_{j=1}^{J} \varsigma_j(\tilde{x}_{\tau_k})(W_{\tau_{k+1}} - W_{\tau_k}), \tag{2.3}$$

where W_t is the Brownian motion introduced in Definition 2.2.6 and evaluated at time $t \in \mathbb{R}_{\geq 0}$.

In this work we shall exclusively focus on finite time horizons, $t \in [0, T]$: let $\Delta t = T/K$ for some $K \in \mathbb{N}$, then $\tau_k = k\Delta t$, where $k = 1, 2, \ldots, K$, is a uniform sequence of sampling times and we will use W_t to denote Brownian motion. With the Euler-Maruyama discretisation scheme, the definition of a path of the SDE follows intuitively as a sequence $\tilde{\omega} = \tilde{x}_0 t_0 \tilde{x}_1 t_1 \ldots t_{K-1} \tilde{x}_K$ comprising sampled times $t_k = \tau_k$ and corresponding states $\tilde{x}_k \in \mathbb{R}^d$ for $k = 1, 2, \ldots, K$. We denote by $Path^{\tilde{\mathcal{M}}}(\tilde{x})$ the set of all finite (and, by extension, infinite) paths of the SDE $\tilde{\mathcal{M}}$ starting in state $\tilde{x}$. We can also work with parametrised SDEs, which is defined similarly to SDEs above and shall later be denoted as $\tilde{\mathcal{M}}_\theta$.

Definition 2.2.9 (Parametric SDEs)**.** Let $\theta = (\theta_1, \ldots, \theta_k)$ be a vector of parameters, taking values in a compact hyperrectangle $\Theta \subset \mathbb{R}^k_{\geq 0}$. A parametric stochastic differential equation (pSDE) over θ is defined similarly to the SDE introduced in Definition 2.2.7, except the functions ς^i_j are now a function of the parameters, $\varsigma^i_j = \varsigma^i_j(\theta, \tilde{x}_t)$:

$$\varsigma^i_j : \mathbb{R}^k \times \mathbb{R}^J \to \mathbb{R},\ i = 1, 2, \ldots, d,\ j = 0, 1, 2, \ldots, J.$$

Stochastic differential equations allow us to model stochastic processes that evolve through continuous time and space which will prove useful for modelling biological processes. Now that we have defined the stochastic models that we will use to capture the obstreperous nature of biological systems, we turn our attention as to how we describe biological systems of interest. To do so, we will use CRNs.

2.3 Chemical Reaction Networks

Biological and biochemical systems governed by cellular mechanisms can be modelled at the level of chemical interactions. For the remainder of this thesis, we utilise CRNs to model biological processes of interest and in this section, we cover the basics of chemical reaction networks and how they can be utilised to model biological processes. A CRN is a set of chemical reactions R_j, $j = 1, 2, \ldots, N_R$, that govern the dynamics of an underlying chemical system that consists of a total of N_S different types of molecules or chemical species, denoted by X_i, $i = 1, 2, \ldots, N_S$. These N_S chemical species may take part in one of the various N_R types of chemical reactions within a given system, where each individual reaction can be denoted as

$$R_j : \sum_{i=1}^{N_S} \upsilon^-_{i,j} X_i \xrightarrow{k_j} \sum_{i=1}^{N_S} \upsilon^+_{i,j} X_i,$$

where the $\upsilon^-_{i,j}$ ($\upsilon^+_{i,j}$) are nonnegative integers that represent the number of molecules that are consumed (produced) when reaction R_j occurs. As a result, the compounds on the left-hand (right-hand) side are referred to as the *reactants* (*products*). The nonnegative integers $\upsilon^+_{i,j}$ and $\upsilon^-_{i,j}$ are referred to as *stoichiometric coefficients*. These stoichiometric coefficients can be arranged into a $N_S \times N_R$ stoichiometric matrix υ, where each element is defined as $\upsilon_{i,j} = \upsilon^+_{i,j} - \upsilon^-_{i,j}$, where $\upsilon_{i,j}$ is the stoichiometry of species i for reaction j.

If the converse transformation or reaction also occurs, then the reaction is said to be *reversible* and a separate reaction R_l where $l \neq j$ representing the reverse reaction needs to be included in the network. If a reaction occurs spontaneously (without

reactants) or due to an inflow of reactants from a system that is not of interest, we may denote this event by replacing the left hand side of the above formula with $\emptyset$, e.g. $\emptyset \xrightarrow{k_j} \sum_{i=1}^{N_S} \upsilon_{i,j}^{+} X_i$. Alternatively, if reactants are degraded or a reaction occurs between reactants of interest and the product is not of particular importance or interest, we denote this type of reaction by replacing the right hand side with $\emptyset$, e.g. $\sum_{i=1}^{N_S} \upsilon_{i,j}^{-} X_i \xrightarrow{k_j} \emptyset$. Biological systems are inherently complex and are interconnected with many processes that may not be of immediate interest, therefore the introduction of the empty set $\emptyset$ is necessary to distil the processes of focus from a larger biological system. Here we present a formal definition of CRNs.

Definition 2.3.1 (Chemical Reaction Network)**.** A Chemical Reaction Network (CRN) $\mathcal{CRN}$ is a tuple $(X, \mathcal{R}, \Bbbk, X_0)$, where

- $X = (X_1, ..., X_{N_S})$ is a vector where each X_i represents the number of molecules of each species $i = 1, \dots N_S$. $X \subseteq \mathbb{N}^{N_S}$, the state space,
- $\mathcal{R} = \{R_1, \dots, R_{N_R}\}$ is the set of N_R chemical reactions, each of the form $R_j = (\boldsymbol{\upsilon}_j, a_j)$, with $\boldsymbol{\upsilon}_j$ the stoichiometry vector of size N_S for reaction j and $a_j(X) = a_j(X, k_j)$ is the propensity or rate function,
- $\Bbbk = (k_1, \dots, k_{N_R})$ is the vector of (kinetic) parameters, taking values in a compact hyperrectangle $\Bbbk \subset \mathbb{R}^{N_R}$.
- $X_0 = (X_1(0), ..., X_{N_S}(0))$ is the initial (at time $t = 0$) number of molecules within the CRN

In many fields of research, the exact position and velocity of the molecules considered is of great interest, particularly if one is interested in the detailed molecular dynamics of the biological system. However, this extra level of detail comes at a great computational cost and we focus our efforts on the dynamics of the overall number of chemical species present in the biological system. We will assume for all chemical reaction networks considered in this thesis that the chemical mixture considered is spatially uniform (or *well-stirred*), that the system exists in a thermal equilibrium and that the volume of the spatial domain remains at a constant, V. Therefore, the focus is on the overall number of molecules of each chemical species type and how the numbers for each chemical species evolve over time. We define a *state vector* of size $N_S \times 1$ that represents the number of molecules of each chemical species, or

alternatively the copy number, at any given time t as

$$X(t) = \begin{bmatrix} X_1(t) \\ X_2(t) \\ \vdots \\ X_{N_S}(t) \end{bmatrix}. \tag{2.4}$$

Given that we are not concerned with the position or velocity of individual molecules, the biological system of interest can be described in terms of the given state vector $X(t)$. The state vector $X(t)$ evolves when reactions occur in the chemical system. Specifically, recalling the stoichiometric matrix defined earlier, $\upsilon_{i,j}$, we note that when reaction j occurs, the copy number of species X_i changes by $\upsilon_{i,j}$. Therefore, when reaction j occurs, the state vector $X(t)$ is updated by the jth column of the stoichiometric matrix, $\boldsymbol{\upsilon}_j = [\upsilon_{1,j}, \upsilon_{2,j}, \ldots, \upsilon_{N_S,j}]^\top$, updating the state vector to $X(t) = X(t^-) + \boldsymbol{\upsilon}_j$, where time t^- is the time immediately before reaction j occurs. Given that we are not concerned with the spatial information of the underlying molecules, we have to consider that each reaction has a given probability of taking place, given the current state of the system. This leads to a natural description where we consider the probability of the system being in a given state at a given time t. In order to calculate the probability of a given chemical system being in a particular state at time t, the *chemical master equation* (CME) was introduced to describe the evolution of these probabilities.

2.3.1 Chemical Master Equation

The CME is a set of linear, autonomous ordinary differential equations (ODEs) where each possible state of the system has a corresponding ODE. The solution of the kth equation at time t gives the probability of a system being in that particular state at time t. That is, the time evolution of states with given probabilistic transitions is given by the set of ODEs provided by the CME. The number of states of a given system depends upon the total number of molecules that are present in the system and the form of the chemical reactions that take place. The reactions we consider are of the following general forms:

1. Zero-order: $\emptyset \xrightarrow{k_j}$ reaction products,
2. First-order: $X_m \xrightarrow{k_j}$ reaction products,
3. Second-order: e.g. $X_m + X_n \xrightarrow{k_j}$ reaction products, where $n \neq m$

A reaction, say R_j, can only occur if there is a sufficient number of molecules in the system present for a reaction to occur. Specifically a reaction R_j is viable if $\forall i$ s.t. $\upsilon_{i,j} > 0$, and $X_i(t) = X_i(t^-) + \upsilon_{i,j} \geq 0$, where $X_i(t)$ is the copy number of species i at time t after reaction j has occurred, $X_i(t^-)$ is the number of molecules of species i at a time t^- just before the reaction R_j occurs. What this means is that there needs to be a sufficient number of molecules in a system in order for R_j to occur, so that the total number of molecules in the system remains positive. With the above formulation, for a system with N_R reactions, the stoichiometric matrix with elements $\upsilon_{i,j}$ and copy number $X(t)$ help us determine which reactions are viable. However, more information is needed before we decide which reactions are the most and least likely to occur. Determining *which* reaction in a system is most likely to fire is encapsulated in the *propensity function*. For a reaction R_j, the associated propensity function $a_j(X(t))$ encapsulates the information needed to calculate the probability that the jth reaction is to occur. Specifically, for an infinitesimal time interval $[t, t + \mathrm{d}t)$, the probability that reaction j occurs is given by $a_j(X(t))\,\mathrm{d}t$. For the reactions we consider above, the given propensity functions are given by

1. Zero-order propensity function: ($\emptyset \xrightarrow{k_j}$ products) $a_j(X(t)) = k_j$.
2. First-order propensity function: ($X_m \xrightarrow{k_j}$ products) $a_j(X(t)) = k_j X_m(t)$.
3. Second-order propensity function: (e.g. $X_m + X_n \xrightarrow{k_j}$ products $n \neq m$) $a_j(X(t)) = k_j X_m(t) X_n(t)$.

A special type of bimolecular reaction we must take care when calculating propensity functions is that of homo-dimer formation: e.g. $X_m + X_m \xrightarrow{k_j}$ reaction products. With this type of reaction, the propensity function is given by $a_j(X(t)) = k_j \frac{1}{2} X_m(t)(X_m(t) - 1)$. The factor in the final product of the homo-dimer formation reaction represents the number of ways of choosing an (unordered) pair of objects from a total of $X_m(t)$ and can often be absorbed into the kinetic rate k_j. Given that we are working with stochastic models, we cannot model the evolution of a CRN exactly and must think in terms of probabilities. In particular, we are interested in calculating the probability that the state $X(t)$ is at a particular value at a given time t, say $X(t) = x$ where $X(0) = x_0$. We denote the probability of $X(t) = x$ at time t, starting from $X(t = 0) = x_0$ as $P(x, t \mid x_0)$. The time evolution of $P(x, t \mid x_0)$ is given by the CME, which we will derive via recurrence here. We first assume that $\mathrm{d}t$ is small enough that only one reaction can occur in the $[t, t + \mathrm{d}t)$. For the system to be in a state $X(t + \mathrm{d}t) = x$ at time $t + \mathrm{d}t$, either the system was in $X(t) = x$ at the

previous time t and no reaction has occurred, or for some reaction $0 \leq j \leq N_R$ the system was in a state $X(t) = x - \boldsymbol{\upsilon}_j$ at time t and then reaction j occurs over $[t, t+\mathrm{d}t)$ to bring the system into the state $X(t+\mathrm{d}t) = x$. To obtain the CME we must apply the law of total probability (Equation (2.1)). Let $\mathcal{H}_0$ be the event that the system is in state x at time $t+\mathrm{d}t$, and $\mathcal{H}_j$ where $1 \leq j \leq N_R$ be the event that the system is in state $x - \boldsymbol{\upsilon}_j$ at time t, and let $\mathcal{H}_{N_R+1}$ be the event that the system is in any other state at time t. By the law of total probability and omitting the dependence on x_0,

$$P(x, t+\mathrm{d}t) = \sum_{j=1}^{N_R+1} P(x, t+\mathrm{d}t \mid \mathcal{H}_j) P(\mathcal{H}_j). \tag{2.5}$$

From the propensity functions, for $1 \leq j \leq N_R + 1$, $P(x, t+\mathrm{d}t \mid \mathcal{H}_j) = a_j(x - \boldsymbol{\upsilon}_j)\,\mathrm{d}t$ as this is simply the probability that reaction j occurs to transition the state from $X(t) = x - \boldsymbol{\upsilon}_j$ to $X(t+\mathrm{d}t) = x$. When $j = 0$, $P(x, t+\mathrm{d}t \mid \mathcal{H}_0) = 1 - \sum_{j=1}^{N_R+1} a_j(x)\,\mathrm{d}t$ as this is the probability that no reactions occur as $\mathcal{H}_0$ is the event that $X(t) = x$ at time t therefore $X(t+\mathrm{d}t) = x$ if no reaction fire. For $\mathcal{H}_{N_R+1}$, $P(x, t+\mathrm{d}t \mid \mathcal{H}_0) = 0$ as the event $\mathcal{H}_{N_R+1}$ contains all the states that are at least 1 reaction away from the state x. By noting that $P(\mathcal{H}_0) = P(x, t \mid x_0)$ and $P(\mathcal{H}_j) = P(x - \boldsymbol{\upsilon}_j, t \mid x_0)$, we obtain

$$P(x, t+\mathrm{d}t \mid x_0) = \left(1 - \sum_{j=1}^{N_R} a_j(x)\,\mathrm{d}t\right) P(x, t \mid x_0) + \sum_{j=1}^{N_R} a_j(x - \boldsymbol{\upsilon}_j)\,\mathrm{d}t P(x - \boldsymbol{\upsilon}_j, t \mid x_0). \tag{2.6}$$

Rearranging we obtain

$$\frac{P(x, t+\mathrm{d}t \mid x_0) - P(x, t)}{\mathrm{d}t} = \sum_{j=1}^{N_R} \left(a_j(x - \boldsymbol{\upsilon}_j) P(x - \boldsymbol{\upsilon}_j, t \mid x_0) - a_j(x) P(x, t \mid x_0)\right), \tag{2.7}$$

and by taking the limit $\mathrm{d}t \to 0$, this gives us the CME,

$$\frac{\mathrm{d}P(x, t \mid x_0)}{\mathrm{d}t} = \sum_{j=1}^{N_R} \left(a_j(x - \boldsymbol{\upsilon}_j) P(x - \boldsymbol{\upsilon}_j, t \mid x_0) - a_j(x) P(x, t \mid x_0)\right). \tag{2.8}$$

The CME defined above is a linear ODE system which has one ODE for each possible state. As the number of states increase, the CME becomes too high-dimensional to allow for analytic solutions apart from the simplest of cases. In particular, if we consider zero-order reactions as above, $\emptyset \xrightarrow{k_j}$ reaction products, there is no upper limit as to how many chemical reactions can occur. This leads to the set of all possible states in a CRN to be infinite and indeed, many CRNs are infinite dimensional.

Whilst we are unable in many cases to obtain an analytic solution to the CME for most cases, the Gillespie algorithm [45] allows one to generate sample traces which are sampled from the solution of the CME. In this next section, we will present how stochastic models introduced in Section 2.2 can be used to analyse the biochemical systems described by CRNs.

2.4 Stochastic chemical kinetics

In this section we present how the stochastic models introduced previously can be used to analyse biochemical systems described by CRNs. We recall that the CME in Equation (2.8) is a set of ODEs where the kth ODE gives the probability of being in the kth state at time t. Analytic solutions for the CME can be derived only for the simplest of case studies and in this section, we explore how to analyse stochastic models that are derived from the CME. We first discuss the Gillespie algorithm or the stochastic simulation algorithm (SSA) [45], which generates sample traces or simulations that are sampled from the solution of the CME, $P(x, t \mid x_0)$.

2.4.1 Stochastic Simulation Algorithm

Let us consider a CRN with R_j reactions where $1 \leq j \leq N_R$. The number of molecules present in the system at a time t, given by $X(t)$, can be modelled by a CTMC if we consider each state of the CTMC to be a unique combination of the number of chemical species (or copy number). That is, the state of the CTMC at a time t is simply given by the state vector, $X(t) = [X_1(t), X_2(t), \ldots, X_{N_S}(t)]^\top$, which represents the number of molecules of each chemical species at time t. Parametrising the reaction rates by $\theta \in \Theta \subseteq \mathbb{R}^{N_R}$ within a CRN results in a parametric CRN, where $\theta = [\theta_1, \theta_2, \ldots, \theta_{N_R}]^\top$ are the N_R kinetic rates present in the CRN of interest. The pCTMC $\mathcal{M}_\theta$, induced by the kinetic parameters $\theta \in \Theta$, leads to a pCTMC that represents a parametrised CRN, where θ are the kinetic rates of the CRN.

Example 2.4.1. Let us consider a simple CRN with the following reactions

$$R_1 : A \xrightarrow{k_1} \emptyset,\ R_2 : B \xrightarrow{k_2} \emptyset,\ R_3 : A + A \xrightarrow{k_3} B,$$

where the state vector is given by $X(t) = [A(t), B(t)]^\top$, initially containing two molecules of the chemical species A and one molecule of the chemical species B, $X(0) = [2, 1]^\top$. The kinetic parameters $\theta = [k_1, k_2, k_3]^\top$ dictate the dynamics of the above CRN, where k_1 is the rate at which chemical species A degrades (reaction R_1),

k_2 is the rate at which B degrades (reaction R_2) and k_3 is the rate at which two molecules of A combine to create one molecule of B (R_3). The stoichiometric vectors and propensities are given by

$$\begin{aligned}
R_1 &: \boldsymbol{\upsilon}_1 = [-1, 0]^\top \text{ with propensity } a_1(X(t)) = k_1 A(t) \\
R_2 &: \boldsymbol{\upsilon}_2 = [0, -1]^\top \text{ with propensity } a_2(X(t)) = k_2 B(t) \\
R_3 &: \boldsymbol{\upsilon}_3 = [-2, 1]^\top \text{ with propensity } a_3(X(t)) = k_3 \frac{1}{2} A(t)(A(t) - 1).
\end{aligned}$$

We can model the above CRN as a pCTMC $\mathcal{M}_\theta$ where each state of the pCTMC x_i is a unique combination of the molecule counts in the system. We present an example pCTMC of the above system in Figure 2.3. In this example we illustrate the use of atomic propositions to label states of the system that are of interest to us, for example, states which satisfy a particular property. Here we consider the set of atomic propositions AP given by $\{\text{Maj}_A, \text{Maj}_B\}$, where a state is labelled Maj_A if $A(t) > B(t)$ and a state is labelled Maj_B if $B(t) > A(t)$, that is, we are interested in the states of the system where the chemical species A has a strict majority (Maj_A) and the states where B has a strict majority (Maj_B). We note that the transition rates $\mathcal{T}(x, x')$ are given by the parameters $\theta = [k_1, k_2, k_3]^\top$ which is *not* equal to the transition probabilities $\mathbf{T}(x, x')$, however we can define $\mathbf{T}(x, x') = \mathcal{T}(x, x')/E(x)$, where $E(x) = \sum_{x' \neq x} \mathcal{T}(x, x')$, where the probability of the chain jumping from state x to a state x' where $x' \neq x$ within t time units is given by $\mathbf{T}(x, x') \int_0^t E(x) \cdot e^{-E(x)t'} \, \mathrm{d}t'$.

Simulations of a CRN, modelled by a pCTMC, can be generated using the SSA (or Gillespie) Algorithm presented in Algorithm 1, where the full derivation is omitted here but can be found in [45]. This algorithm allows us to simulate independently a reaction index and a reaction time for each reaction that occurs, which gives us a detailed simulation for an induced CRN, $\mathcal{M}_\theta$, as to which reaction occurs and when. Of course, this high level of detail comes at a high computational cost. At each iteration of the SSA, both a reaction time and index must be drawn to calculate the associated propensity function which also updates the state vector at every stage. Additionally, when $a_{\text{sum}}(X(t)) = \sum_{j=1}^{N_R} a_j(X(t))$ is large - either due to a high number of molecule counts or a number of fast reactions present in the system - the reaction times τ become increasingly small, which results in a high number of random number generations and updates [59], which can greatly slow down the simulation times.

To circumvent this problem, instead of considering each reaction time independently we can consider simulating the number of reactions that can occur in a fixed

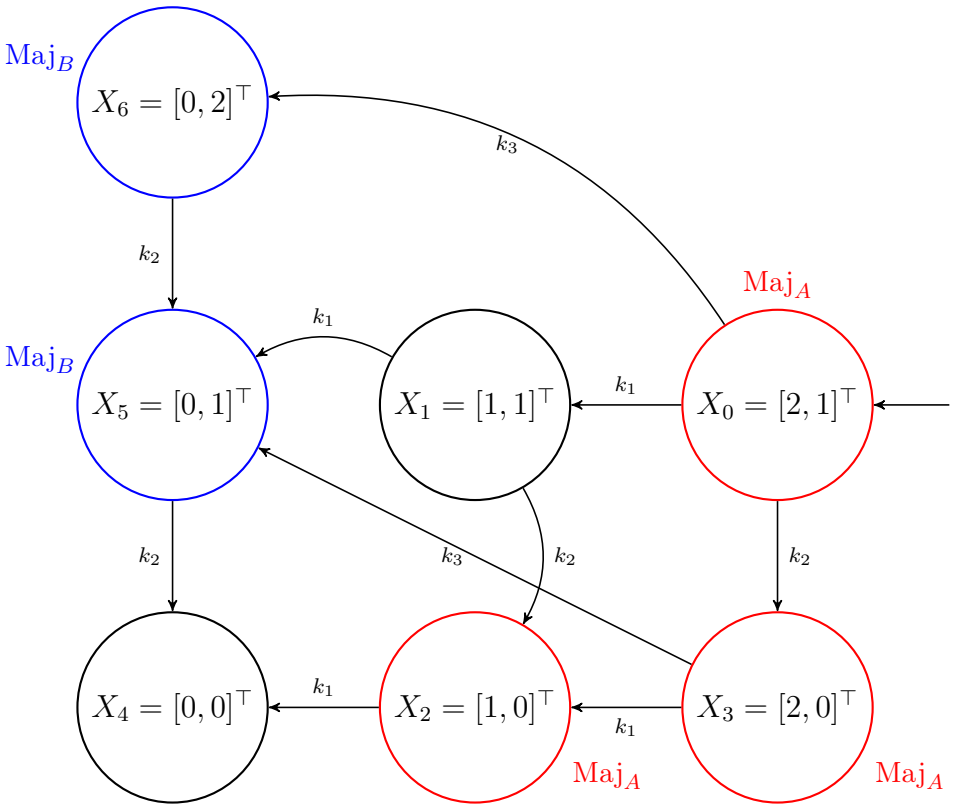


Figure 2.3. Modelling a CRN as a pCTMC where $X(t) = [A(t), B(t)]^\top$ with initial state $X(0) = x_0 = [2,1]^\top$ and the transition rates of the pCTMC are parametrised by $\theta = [k_1, k_2, k_3]^\top$ and the atomic propositions are given by $AP = \{\text{Maj}_A, \text{Maj}_B\}$.

Algorithm 1 Stochastic Simulation Algorithm (SSA)

Input:

- Initial time $t_0 \geq 0$ and state $X(t) = x_0$
- Simulation time $T > t_0 \geq 0$.
- Parametrised CRN $\mathcal{M}_\theta$ with induced parameters $\theta = (\theta_1, \ldots, \theta_{N_R})$, associated propensity functions $a_j(X) = a_j(X, k_j)$ and stoichiometric vectors $\boldsymbol{\upsilon}_j$ for $j = 1, \ldots N_R$ reactions.

Output:

- A path or trace $\omega = x_0 t_0 \ldots x_{k-1} t_{k-1} x_k$ generated from a pCTMC, $\omega \sim \mathcal{M}_\theta$, where we note $\sum_{i=0}^{k-1} t_i \leq T$

$k = 1$
while $t \leq T$ **do**
  Calculate $\{a_j(X(t))\}_{j=1}^{N_R}$ and $a_{\text{sum}}(X(t)) = \sum_{j=1}^{N_R} a_j(X(t))$.
  Draw $U_1 \sim \mathcal{U}(0,1)$, and $U_2 \sim \mathcal{U}(0,1)$.
  Set j to be the smallest integer satisfying $\sum_{i=1}^{j} a_i(X(t), \theta_i) > U_1 a_{\text{sum}}(X(t))$
  Set $\tau = \ln(1/U_2)/a_{\text{sum}}(X(t))$
  Set $x_k = X(t)$ and $t_k = \tau$.
  Set $X(t+\tau) = X(t) + \boldsymbol{\upsilon}_j$.
  Update $t \leftarrow t + \tau$.
  Update $k \leftarrow k + 1$.
end while
return Path generated from pCTMC, $\omega \sim \mathcal{M}_\theta$, $\omega = x_0 t_1 \ldots x_{k-1} t_{k-1} x_k$.

---

time interval of length τ. Over this interval of time, all reactions that are likely to happen are simulated simultaneously with the state vector being updated using fixed propensity functions $\{a_j(X)\}_{j=1}^{N_R}$. This method leads to the τ-leaping [59] and the Chemical Langevin Equation [47, 59] approximations, which are approximations to the SSA algorithm that allow a speed-up of simulation time but introduce errors that will be small as long as the updates are relatively small. In this thesis, we do not work with τ-leaping techniques and direct the interested reader to the work of [59] for further detail. In the work pursued in this thesis, we are interested in the continuous space and time stochastic process that models a CRN of interest, the chemical Langevin equation.

2.4.2 Chemical Langevin Equation

The chemical Langevin equation (CLE) simultaneously fires multiple reactions in a short time interval and simulates the CRN using N_S SDEs, where each stochastic differential equation (SDE) corresponds to the evolution of a chemical species over a time interval, $t \in [0, T]$.

Here we present the CLE introduced and derived in [47]. We begin the presentation of the CLE by formalising the idea introduced at the end of the previous section. For cases when many reactions occur or when reaction times converge to zero over a small time frame, either due to fast reactions occurring or a high number of molecules being present in the system, it is useful to consider generating a number of reactions simultaneously to occur over a fixed period of time, given by $\tau > 0$. Specifically, we let the random variable $\mathcal{K}_j(X(t), \tau)$ denote the number of R_j reactions that occur in a given time interval $[t, t+\tau]$ and for brevity we omit the explicit dependencies of the propensity function, $a_j = a_j(X(t), \theta_j)$. The number of molecules of chemical species i at time $t + \tau$ is then given as

$$X_i(t + \tau) = X_i(t) + \sum_{j=1}^{N_R} \mathcal{K}_j(X(t), \tau)\upsilon_{i,j}. \tag{2.9}$$

Calculating $\mathcal{K}_j(X(t), \tau)$ directly can be difficult, but they can be well approximated by imposing two conditions [47]. The first condition is to choose τ small enough such that

$$a_j(X(t')) \cong a_j(X(t)),\ \forall t' \in [t, t + \tau],\ \forall j \in \{1, \ldots, N_R\}.$$

Since the reactions that occur in this time interval $[t, t+\tau]$ do not alter the propensity functions drastically, all of the reaction events occurring in the time interval can be

considered to be independent of one another. Given this condition, each $\mathcal{K}_j(X(t),\tau)$ will be a statistically independent Poisson random variable, $\mathcal{P}_j(a_j,\tau)$, which allows for the approximation

$$X_i(t+\tau) = X_i(t) + \sum_{j=1}^{N_R} \mathcal{P}_j(a_j,\tau)\upsilon_{i,j}, \quad \text{for } i = 1,\ldots,N_S. \tag{2.10}$$

Conversely, the second condition we require is that τ is to be large enough such that the expected number of times that each reaction R_j is fired in $[t, t+\tau]$ is much larger than 1,

$$\mathbb{E}\left[\mathcal{P}_j(a_j,\tau)\right] = a_j\tau \gg 1, \ \forall j \in \{1,\ldots,N_R\}. \tag{2.11}$$

It is clear that there will indeed be situations where both of these conditions cannot be met, which would lead this formulation of the CLE to fail and this indeed is a recognised problem. Alternative formulations of the CLE equation, such as [99], have been proposed to circumvent issues raised in this formulation. However, there are many scenarios where both conditions will be met, particularly when we have a large molecular population in a CRN of interest. When these two conditions are met, the Poisson random variable $\mathcal{P}_j(a_j,\tau)$ can be approximated by a Normal or Gaussian random variable with the same mean and variance, $\mathcal{P}_j(a_j,\tau) \approx \mathcal{N}(a_j\tau, a_j\tau)$. To see this, first define the probability density function of $\mathcal{P}_j(a_j,\tau)$,

$$P(\mathcal{P}_j(a_j,\tau) = n^R) = \frac{e^{-a_j\tau}(a_j\tau)^{n^R}}{n^R!}, \ n^R \in \mathbb{N}, \tag{2.12}$$

which denotes the probability that n^R reactions R_j happen within a time interval τ. Whenever $a_j\tau \gg 1$, we can use Stirling's approximation to write

$$\frac{e^{a_j\tau}(a_j\tau)^{n^R}}{n^R!} \approx (2\pi a_j\tau)^{-1/2} \exp\left(-\frac{(n^R - a_j\tau)^2}{2a_j\tau}\right), \tag{2.13}$$

whereby discrete Poisson random variables have been approximated by continuous Normal variables. Accordingly, the evolution of the state vector $X_i(t)$ is now described by the continuous approximation $\tilde{x}_i(t)$,

$$\tilde{x}_i(t+\tau) = \tilde{x}_i(t) + \sum_{j=1}^{N_R} \upsilon_{i,j}\mathcal{N}_j(a_j\tau, a_j\tau), \ i = 1,\ldots,N_S, \tag{2.14}$$

where $\mathcal{N}(\mu,\sigma^2)$ denotes the Normal random variable with mean μ and variance σ^2 and $a_j = a_j(\tilde{x}(t))$. Expressing $\mathcal{N}(\mu,\sigma^2) = \mu + \sigma\mathcal{Z}_j$, where $\mathcal{Z}_j \sim \mathcal{N}(0,1)$ this equation now becomes

$$\tilde{x}_i(t+\tau) = \tilde{x}_i(t) + \sum_{j=1}^{N_R} \upsilon_{i,j}a_j(\tilde{x}(t))\tau + \sum_{j=1}^{N_R} \upsilon_{i,j}\sqrt{a_j(\tilde{x}(t))\tau}\mathcal{Z}_j. \tag{2.15}$$

The stochastic difference equation in (2.15) is in fact an Euler-Maruyama discretisation [77] with uniform time samples of duration τ of an SDE of the form

$$\mathrm{d}\tilde{x}(t) = \sum_{j=1}^{N_R} \boldsymbol{\upsilon}_j a_j(\tilde{x}(t))\ \mathrm{d}t + \sum_{j=1}^{N_R} \boldsymbol{\upsilon}_j \sqrt{a_j(\tilde{x}(t))}\ \mathrm{d}W_j(t), \tag{2.16}$$

where the $W^j(t)$ are independent scalar Brownian motion also known as Wiener processes. We note that the dimensionality of the Brownian motion $d = N_S$ and that the dimensionality of the stochastic process $J = N_R$ from Definition 2.2.7

Whilst the CLE leads to computational savings and faster simulating times, these benefits come at a price. We highlight that the CLE is an approximation to the underlying dynamics when there is a high molecule count and it is important to note that the processes driving the CLE can lead to non-physical results, such as negative molecule concentrations as the stoichiometric vector $\boldsymbol{\upsilon}_j$ has negative elements which can drive $\tilde{x}(t)$ to have negative elements. These results arise due to the nature of the CLE approximation and not via the SDE discretisation. Modifications [99, 100] have been introduced to circumvent these issues. In this work, to mitigate this effect, we consider the propensity functions of the form $a_j(\tilde{x}(t)) = a_j(\tilde{x}(t))$ if $\tilde{x}_i(t) > \upsilon_{i,j}^-$ for all $i = 1, \ldots, N$, where $\upsilon_{i,j}^-$ is the stoichiometric coefficient for the reactants, and $a_j(\tilde{x}(t)) = 0$ otherwise.

In this section we have introduced numerous stochastic models to describe the forward evolution of a stochastic biochemical reaction network. Recall the aim of this thesis is to perform data-driven verification; we wish to construct accurate stochastic models, $\mathcal{M}_\theta$, from data that are amenable to rigorous formal verification methods to ensure that the underlying biological system, $\mathcal{S}$, satisfies a property of interest. In order to learn these parametrised stochastic models, we will work with Bayesian methods.

2.5 Bayesian Inference

Throughout this thesis, we focus on the Bayesian interpretation of probability and work largely with Bayesian statistics. Bayes' theorem underpins all of Bayesian statistics as it allows one to compute and update probabilities after obtaining new and unseen data. Bayesian methods have been used extensively in the life sciences for parameter estimation, model selection and even the design of experiments [40, 75, 98, 108, 112]. The important result that forms the basis of Bayesian inference is given by Bayes' theorem, defined as the following.

Definition 2.5.1 (Bayes' Theorem)**.** For all given events A and B, such that $P(B) > 0$, we have

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}. \tag{2.17}$$

Suppose $\mathcal{H}$ is a hypothesis and $\mathcal{D}$ is data which may give evidence in favour or against the hypothesis $\mathcal{H}$. Bayesian inference is defined as reinterpreting the events in Bayes' formula, given by

$$\pi(\mathcal{H} \mid \mathcal{D}) = \frac{\mathcal{L}(\mathcal{D} \mid \mathcal{H})\pi(\mathcal{H})}{p(\mathcal{D})}, \tag{2.18}$$

where $\pi(\mathcal{H} \mid \mathcal{D})$ is the posterior probability distribution, which is the probability that the hypothesis $\mathcal{H}$ is true after the data is considered, $\pi(\mathcal{H})$ is the prior probability distribution, which is the probability that $\mathcal{H}$ is true before any data is considered whereas $\mathcal{L}(\mathcal{D} \mid \mathcal{H})$ is the likelihood, which is the evidence about $\mathcal{H}$ provided by the data $\mathcal{D}$. Finally, $p(\mathcal{D})$ is the total probability of the data taking into account all possible hypotheses. In Bayesian inference, knowledge of the hypothesis $\mathcal{H}$ is contained in this posterior distribution, $\pi(\mathcal{H} \mid \mathcal{D})$, which contains all of the necessary information required for the analysis of a hypothesis with respect to the data collected thus far.

It is of particular interest to the biological community to constrain any uncertainty within the model parameters (or indeed the model itself) by using the observed data of biological systems. Moreover, when one is working with obstreperous stochastic models, noisy observations may add another layer of uncertainty. A plethora of literature is focused on the problem of Bayesian inference in stochastic biochemical models [100,114,116], let alone stochastic models [28]. In this thesis, we will be working with a parametrised stochastic model $\mathcal{M}_\theta$ that represents an underlying stochastic system $\mathcal{S}$, where the stochastic system generates data $y_{obs} \in \mathcal{Y}$. Rather than a particular hypothesis, we will use Bayesian inference to infer a vector of model parameters $\theta \in \Theta$ given data, y_{obs}, that we have collected via experiments. Prior beliefs about the model parameters are thus encapsulated in the prior probability distribution $\pi(\theta)$, where the posterior probability of the parameters given data, $\pi(\theta \mid y_{obs})$, is updated through the likelihood function, $p(y_{obs} \mid \theta)$. Considering this interpretation of the Bayesian inference problem, and the fact that $p(y_{obs}) = \int_\Theta p(y_{obs} \mid \theta)\pi(\theta)\,\mathrm{d}\theta$, the posterior distribution is defined as

$$\pi(\theta \mid y_{obs}) = \frac{p(y_{obs} \mid \theta)\pi(\theta)}{\int_\Theta p(y_{obs} \mid \theta)\pi(\theta)\,\mathrm{d}\theta}, \tag{2.19}$$

and the Bayesian aspect of our framework will be concerned with inferring this posterior distribution. However, for many models, obtaining closed form solutions of $\pi(\theta \mid y_{obs})$ is often not possible and numerical methods are typically needed to allow for inference.

As one can imagine, calculating the evidence, $p(y_{obs})$, is often difficult and in many models, obtaining an exact or tractable likelihood, $p(y_{obs} \mid \theta)$, is difficult or nigh on impossible. It is particular useful within Bayesian inference to take the expectation of an arbitrary function $g(\theta)$ function over the posterior,

$$\mathbb{E}_{\pi}[g(\theta)] = \int_{\Theta} g(\theta)\pi(\theta \mid y_{obs}) \, \mathrm{d}\theta, \tag{2.20}$$

which is difficult to evaluate directly as $p(y_{obs})$ is impossible to calculate in all but the simplest of cases. In addition to this, we may have problems obtaining analytical or tractable likelihoods and in this case, one can resort to alternative Bayesian techniques such as variational Bayesian methods and likelihood-free methods [83]. It is common to use Monte Carlo techniques to estimate integrals such as those introduced above by using N samples generated from the posterior, $\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(N)} \sim \pi(\theta \mid y_{obs})$. Specifically, with a finite number of samples from the posterior, we can obtain a finite sample approximation to the posterior given by

$$\pi(\theta \mid y_{obs}) \approx \frac{1}{N} \sum_{i=1}^{N} \delta_{\theta^{(i)}}(\theta), \tag{2.21}$$

where $\delta_Z(z)$ denotes the Dirac measure, defined as $\delta_Z(z) = 1$ if $z \in Z$ and $\delta_Z(z) = 0$ otherwise. Using samples generated from the posterior, we can approximate the expectation of a function $g(\theta)$ as

$$\mathbb{E}_{\pi}[g(\theta)] \approx \int_{\theta} a(\theta) \frac{1}{N} \sum_{i=1}^{N} \delta_{\theta^{(i)}}(\theta) \, \mathrm{d}\theta = \frac{1}{N} \sum_{i=1}^{N} g(\theta^{(i)}). \tag{2.22}$$

where $\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(N)} \sim \pi(\theta \mid y_{obs})$ are N independent samples from the posterior distribution, $\pi(\theta|y_{obs})$. For many problems, obtaining a numerical evaluation of the likelihood function $p(y_{obs} \mid \theta)$ may not be possible and thus evaluating the posterior probability distribution to draw samples from makes the implementation of Bayesian methods difficult or impractical. As mentioned before, we could use variational inference, the main idea being to perform inference on an alternative distribution, of which we know how to calculate the posterior distribution, and use this to approximate the true distribution of interest. However, this model may not be an appropriate model and may not capture the true dynamics of the underlying system of interest. A

more desirable alternative is to to consider an approximation of the true or preferred model, forfeiting accuracy in order to retain the model that best describes the underlying system. In this thesis, we consider this latter approach, working with posterior approximation methods, namely approximate Bayesian computation [101] (ABC). We will introduce the notion of ABC later in this thesis to see how it fits in with the framework we wish to work with. Whilst this covers to an extent the Bayesian aspect of our Bayesian verification framework, we turn our attention to the formal verification aspect of our work, specifically methods to formally verify properties on stochastic models of interest.

2.6 Formal Verification

In this thesis we introduce a framework that aids in the design and construction of mathematical models in systems and synthetic biology. Through Bayesian methods, we aim to learn models that accurately describe the underlying stochastic biological systems of interest to a high-degree of accuracy. The motivation for learning mathematical models that accurately describe underlying systems is obvious - accurate modelling leads to discoveries, such as dynamics previously unknown, or to prove or disprove previously held hypotheses on the underlying system. More often than not, the aim of constructing a mathematical model is to ascertain particular dynamics or properties of an underlying system. Within the realms of synthetic biology, when constructing a novel biological device to perform a specific task, it is of utmost importance to verify that the constructed device performs exactly as it was designed to. However, informally defining properties of interest that we wish to test the system on leads to the discovery of inconsistencies or perhaps incompleteness of the system specifications. Formal verification techniques provide a way to specify properties in a formal manner that allows one to establish system correctness with mathematically rigorous techniques [10]. Specifically, we are interested in model-based verification techniques, which are based on models describing the system behaviour in a mathematically precise and unambiguous manner. Verification techniques can range from an exhaustive exploration of the system's behaviour (through model checking) or to performing experiments under a restrictive set of scenarios in the model (via simulation) or in reality (testing). It goes without saying that the success of formal verification relies on the accuracy of the model itself; any formal verification using model-based techniques is only as good as the model itself. We consider applying

these formal verification techniques for applications within biology, as testing properties via experiments is infeasible; conducting various experiments to explore multiple properties of interest can be difficult, costly, and time-consuming. We propose that by integrating Bayesian methods (to learn models from data) and formal verification techniques (to rigorously verify properties on these learned models), that we would be able to avoid performing exhaustive experiments to explore further properties of interest that are not immediately evident from observing the data obtained in any initial experiments. In fact, formal verification has been applied successfully within the realms of biology: in [72] the authors present an application of probabilistic or numeric model checking to biological systems modelled as continuous-time Markov chains, in [119] the author presents a survey of statistical model checking techniques for biological applications meanwhile specific applications of model checking within biology include model checking properties on biological pathways [57] and genetic circuits [76]. To first model check properties of interest over these inferred or learned models, we must consider how to formally define properties.

2.6.1 Defining Properties

To reason about our stochastic models and verify properties of interest, we need a means of formally defining these properties. In our work and the remainder of our thesis, we work with a formal temporal logic to define properties of interest. Given that our parametrised stochastic models $\mathcal{M}_\theta$ of interest evolve continuously over time, we employ a time-bounded fragment of *continuous stochastic logic* (CSL) [8, 69].

Definition 2.6.1 (Continuous Stochastic Logic (CSL))**.** Let ϕ be a CSL formula interpreted over states $x \in \mathcal{X}$ of a stochastic model, and φ be a formula over its paths. Its syntax is

$$\phi := \text{true} \mid a \mid \neg\phi \mid \phi \wedge \phi \mid \phi \vee \phi \mid P_{\sim\zeta}[\varphi] \ ,$$

$$\varphi := \ \phi_1 U^{[t,t']} \phi_2 \ ,$$

where $a \in AP$, $\sim \in \{<, \leq, \geq, >\}$, $\zeta \in [0, 1]$, and $t, t' \in \mathbb{R}_{\geq 0}, t \leq t'$.

Path formulae are defined by combining state formulas through the temporal operator $\phi_1 U^I \phi_2$, which is true if ϕ_2 is satisfied at some $\tau \in I$ and ϕ_1 holds at all preceding time instants [69]. Notice that, with very limited loss of expressivity, we have excluded the *next* operator, which meaningfully applies to CTMC models, but not to SDE models. We consider a subset of CSL formulae over models, and associate

atomic propositions over their state spaces via labelling functions. In particular we focus on non-nested CSL formulae which exclude the next and steady-state operator - see [104] for a full treatment of model checking CSL properties for related models. Let $\mathcal{M}_\theta$ denote a parametrised stochastic model of choice, with associated initial state, x_0, path ω and set of paths $Path^{\mathcal{M}_\theta}(x_0)$. $P_{\sim\zeta}[\varphi]$ holds if the probability of the path formula φ being satisfied from a given state meets the inequality $\sim \zeta$. The measurability of the concerned events for a CTMC has been argued in [10] and this can be similarly done for SDE models, assuming labelling functions have non-trivial co-domains, namely labels are associated to sets of non-zero measure. The syntax of CSL is split between properties defined on states, ϕ, and those defined on paths φ. We write $x \models \phi$ to indicate that a CSL formula ϕ is satisfied in a state x and we denote by $Sat(\phi)$ the set $\{x \in \mathcal{X} \mid x \models \phi\}$. Similarly, for a path formula φ satisfied by a path $\omega \in Path^{\mathcal{M}_\theta}(x)$, we write $\omega \models \varphi$. For any state $x \in \mathcal{X}$, the relation $x \models \phi$ is defined inductively by

$$\begin{aligned}
x \models true &\quad \forall x \in \mathcal{X}, \\
x \models a &\quad \Leftrightarrow \quad a \in L(x), \\
x \models \neg\phi &\quad \Leftrightarrow \quad x \not\models \phi, \\
x \models \phi_1 \wedge \phi_2 &\quad \Leftrightarrow \quad x \models \phi_1 \wedge x \models \phi_2, \\
x \models P_{\sim\zeta}[\varphi] &\quad \Leftrightarrow \quad P(\omega \in Path^{\mathcal{M}_\theta}(x) \mid \omega \models \varphi) \sim \zeta,
\end{aligned}$$

and for any path $\omega \in Path^{\mathcal{M}_\theta}(x)$,

$$\omega \models \phi_1 U^{[t,t']} \phi_2 \quad \Leftrightarrow \quad \exists \tau \in I \ s.t. \ (\omega[\tau] \models \phi_2 \wedge (\forall \tau' \in [0,\tau), \omega[\tau'] \models \phi_1)),$$

where $\omega[\tau]$ denotes the state of the path at time τ. In our work we are interested only in these properties although additional properties can be specified such as rewards [69]. Given that we are solely concerned with parametrised stochastic models of interest, $\mathcal{M}_\theta$, we define a *satisfaction function* to capture how the satisfaction probability of a given property over model paths relates to its parameters and initial state.

Definition 2.6.2 (Satisfaction Function)**.** Let ϕ be a CSL formula, $\mathcal{M}_\theta$ be a stochastic parametrised model over a space Θ, x_0 the initial state, and $Path^{\mathcal{M}_\theta}(x_0)$ the set of all paths generated by a parametrised stochastic model $\mathcal{M}_\theta$ with initial state x_0. Denote by $\Lambda_\phi : \theta \to [0,1]$ the satisfaction function such that

$$\Lambda_\phi(\theta) = P\left(\{\omega \models \varphi, \omega \in Path^{\mathcal{M}_\theta}(x_0)\} \mid \omega(0) = x_0\right), \tag{2.23}$$

where a path $\omega \models \varphi$ if its associated trace satisfies the path formula φ corresponding to the CSL formula ϕ. That is, $\Lambda_\phi(\theta)$ is the probability that the set of paths from a given parametrised stochastic model $\mathcal{M}_\theta$ satisfies a property φ. If $\Lambda_\phi(\theta) \sim \zeta$, then we say that $\mathcal{M}_\theta \models \phi$.

We note that the notation involved with the satisfaction function above is aligned with pCTMCs, however it would be straightforward to define an equivalent satisfaction function for pSDEs. Having defined the satisfaction function, we consider two methods of formal verification utilised to calculate or estimate this satisfaction probability - probabilistic model checking and statistical model checking. Probabilistic model checking calculates the satisfaction probability numerically while statistical model checking techniques estimate this satisfaction probability. We first consider probabilistic model checking [10,72], restricting our focus to parametrised continuous-time Markov chains.

2.6.2 Probabilistic Model Checking

Probabilistic model checking is an automated verification technique for the analysis of models that evolve stochastically. It involves the exhaustive analysis of a probabilistic model against a range of quantitative properties to calculate the probability that desired or undesired events are likely to occur. Probabilistic model checking has been applied successfully to many fields including biology [72]. In this section we focus on how probabilistic model checking can be performed on Markov chains, both discrete-time and continuous-time, that are used to model a variety of biological phenomena. Under certain conditions, probabilistic model checking techniques allow for the exact computation of probabilities of events for both discrete-time and continuous-time Markov chains.

Given that we are interested in the continuous-time modelling of biological systems, we will consider probabilistic model checking of CTMCs. Probabilistic model checking of CTMCs for the probabilistic operator reduces to transient probability calculation, which is defined as the probability of being in a given state at a time instant t. Transient probability calculation for CTMCs typically proceeds through discretisation via uniformisation. Specifically, given a CTMC $\mathcal{M} = (\mathcal{X}, \mathcal{T}, AP, L)$, with the transition rates matrix $\mathcal{T}$, we can define an infinitesimal generator matrix $\mathcal{Q} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ by $\mathcal{Q}(x, x') = \mathcal{T}(x, x')$ if $x \neq x'$ and $\mathcal{Q}(x, x) = -\sum_{x' \neq x} \mathcal{T}(x, x') = E(x)$ otherwise. A uniformised DTMC $unif(\mathcal{M}) : \mathcal{T} \times \mathcal{T} \to [0, 1]$ is then constructed by setting

$unif(\mathcal{M}) = I + \mathcal{Q}/q$ where I is the identity matrix and $q \geq \max\{E(x) \mid x \in \mathcal{X}\}$. Informally, the execution of this CTMC can be considered in terms of discrete jumps of the uniformised DTMC given by $unif(\mathcal{M})$, where each step corresponds to one epoch of value q. The transient probability at a given time t can then be approximated via an infinite summation of Poisson-distributed jumps corresponding to q [72]. Model checking for CSL proceeds by bottom-up traversal of the parse tree of the formula ϕ. For the properties we concern ourselves with in this paper, evaluating the probabilistic operator $P(\phi_1 U^{[t,t']} \phi_2)$ reduces to the computation of a transient probability which can be computed numerically on the uniformised DTMC $unif(\mathcal{M})$. For further detail on alternative properties and how to model check properties on DTMCs, we refer the interested reader to the informative tutorial paper [69].

Whilst model checking via these numerical techniques can provide exact probabilities to a high accuracy, due to the computational and memory requirements, numeric techniques do not scale particularly well. Regarding probabilistic model checking of CTMCs that describe CRNs, uniformisation based approaches such as the above break down for many sizeable CRNs of interest as state-spaces grow exponentially with respect to the number of species and molecule counts present within a system. For stochastic chemical kinetics however, ongoing work such as [20] have been developed within probabilistic model checking to allow for more scalable solutions to numeric based model checking. In this work for example, the authors employ a continuous-space approximation of CTMCs in terms of Gaussian processes (GPs) based on the Linear Noise Approximation [111], where model checking algorithms on the resulting GPs are developed. Whilst work such as [20] has been developed to address the scalability issue of probabilistic model checking, statistical approaches to formal verification tend to be favoured due to the general applicability of statistical approaches

2.6.3 Statistical Model Checking

Statistical model checking (SMC) techniques can be used to estimate the validity of quantitative properties of probabilistic models by simulating their traces [74]. Unlike precise (up to numerics) probabilistic model checking, SMC results are typically attained with statistical precision and can come, in particular, with confidence bounds [32, 80]. These statistical approaches can be applied to a variety of systems of interest, including those with complicated dynamics, as long as the underlying system is stochastic. This approach is a particularly powerful approach to model checking

as these statistical approaches are applicable even to systems that are not fully understood or defined. So long as one is able to generate executions from a system of interest, one can perform model checking - approaches such as SMC can even be applied on infinite-state stochastic systems where measures on the probability space are not completely known [3]. To perform a statistical approach to model checking, SMC proposes a simulation-based approach to reason and verify precise properties, say ϕ, defined in a temporal logic over a stochastic model of interest, $\mathcal{M}_\theta$. We recall that a property ϕ has an associated path formula φ and we are interested in properties of the form $\phi = P_{\sim\zeta}(\varphi)$, similar to those defined previously.

The statistical model checking problem can be quantified by estimating a Bernoulli random variable with unknown mean u. Given a trace, y, (or similarly, a path ω) generated from a model $y \sim \mathcal{M}_\theta$, the satisfaction of this trace $y \models \varphi$ is quantified by a Bernoulli random variable V which denotes the outcome of $y \models \varphi$ with mean parameter u. The conditional probability density function associated with V is given by

$$p(v \mid u) = u^v(1-u)^{1-v}, \tag{2.24}$$

where $v = 1$ if $y_i \models \varphi$, otherwise $v = 0$. Thus, SMC techniques may be used for either hypothesis testing or for estimating the probability that a given property is satisfied. In this thesis, we are concerned solely with estimation based techniques, that is, we are interested in estimating the probability that a given property is satisfied, and we refer those interested in hypothesis testing using SMC to [3]. Let us assume that $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ is a given set of observations of v_i, a random Bernoulli variable with unknown mean u, $V \sim \text{Ber}(u)$. We can estimate the unknown mean u by $\hat{u}$ via Monte Carlo approaches, where

$$\hat{u} = \frac{1}{n}\sum_{i=1}^{n} v_i \approx u.$$

For a property of interest $\phi = P_{\sim\zeta}(\varphi)$, the estimated $\hat{u}$ may be compared with ζ to verify whether $\hat{u} \sim \zeta$ or $\hat{u} \not\sim \zeta$. We note that u is in fact $u = \Lambda_\phi(\theta)$, and thus, we estimate the satisfaction probability using these Monte Carlo approaches, where

$$\hat{\Lambda}_\phi(\theta) = \frac{1}{n}\sum_{i=1}^{n} v_i \approx \mathbb{E}[V] = \Lambda_\phi(\theta), \tag{2.25}$$

where once again the number of simulations n is chosen so that the estimate is accurate up to statistical guarantees.

The success of estimation techniques is based on the number of traces n needed to generate to obtain an accurate enough approximation to $u = \Lambda_\phi(\theta)$. The number of simulations n required can be obtained via standard concentration inequalities, such as the Chernoff [32] or the Okamoto [85] bounds. For example, the application of a bound proved by Hoeffding yields

$$P(|\hat{\Lambda}_\phi(\theta) - \Lambda_\phi(\theta)| \geq \epsilon) \geq 2e^{-2n\epsilon^2}.$$

By setting $n = \lceil \ln(2\delta)/(2\epsilon^2) \rceil$, we obtain $P(|\hat{\Lambda}_\phi(\theta) - \Lambda_\phi(\theta)| \leq \epsilon) \geq 1 - \delta$. Given a desirable absolute error tolerance ϵ and a confidence δ, we are able to obtain the number of samples or simulations required n to estimate $\hat{\Lambda}_\phi(\theta)$ such that $\Lambda_\phi(\theta) \in [\hat{\Lambda}_\phi(\theta) - \epsilon, \hat{\Lambda}_\phi(\theta) + \epsilon]$ where $[\hat{\Lambda}_\phi(\theta) - \epsilon, \hat{\Lambda}_\phi(\theta) + \epsilon]$ is a $(1 - \delta)$-confidence interval.

Although we have only considered a very basic SMC method here, there is extensive research to develop further methods to reduce the number of simulations, n, required but to maintain this high level of accuracy for the estimated parameter $\hat{\Lambda}_\phi(\theta)$. In fact, SMC has proved to be a valuable tool within systems biology where extensive examples of SMC within the field can be found in [119]. In this thesis we discuss further SMC techniques and how these techniques complement the Bayesian verification framework.

We have covered the fundamental ideas that form the core idea behind Bayesian verification. In the remainder of this thesis, we illustrate how to bring these ideas together in order to calculate the probability that an underlying system $\mathcal{S}$, satisfies a property ϕ, given data y_{obs}. Our technique brings together stochastic models in biology, Bayesian methods and formal verification techniques. However, many permutations of these three fields have been integrated in the literature and we discuss select publications that directly relate to our work considered in this thesis.

2.7 Related work

Chemical reaction networks (CRNs) have been utilised to model biological systems both deterministically as a set of autonomous ODEs [6, 60] and stochastically [116] via the CME [46]. Deterministic descriptions of CRNs based on ODEs are described in terms of concentrations of the reactants and the rate of a chemical reaction is directly proportional to the the rate of the reactions and the product of the concentrations of the reacting chemical species. However, when biological systems of interest have low concentrations or low molecule counts in the system, stochastic models are more appropriate as dynamics at this low level can lead to phenomena that cannot

be modelled by deterministic models, such as stochastic bistability [105, 106]. The relationship between the stochastic and deterministic methods of modelling CRN systems have been discussed in previous works such as [38, 59], where informally, a deterministic description of a CRN is more appropriate when there are large numbers of molecules of each species.

Obtaining analytic solutions for the CME [46] is typically infeasible and for the analysis of stochastic nature of CRNs, we rely on simulations. In the work of [45], Gillespie derives an algorithm, the SSA, that generates exact traces from the CME of interest which allows for the analysis of the underlying CRN via simulations. These simulations of molecule counts are generated on discrete space but in continuous time, typically modelled as continuous-time Markov chains [66]. Although these simulations are detailed, they are computationally costly and as one can imagine, a high number of simulations are required to fully understand a system of interest. Approximations to this scheme have been introduced to allow a more scalable way of analysing the stochastic system of interest; a continuous space and continuous time approximation to the CME is given by the CLE [47] which was derived in its entirety earlier in this thesis. The CLE is a set of SDEs that describe the evolution of the CRN and can be solved numerically via the Euler-Maruyama scheme [77]. Whilst in our work we rely on these two models to generate stochastic simulations, a review of alternative simulation techniques such as the tau-leaping method [48], the linear noise approximation [110] and other methods that sacrifice accuracy for speed can be found in [59, 100, 114].

In this thesis, we are interested in the data-driven construction of models that represent the underlying biological system. To do this, we work with Bayesian methods, such as inference. Bayesian inference [25, 28] techniques have long been applied to biological systems. In particular, our focus is inferring the kinetic parameters of the CRNs which is a topic that has been extensively studied [26, 100, 100, 115, 116]. Exact inference is in general difficult due to the intractability of the likelihood function, either due to the nature of the model or the data itself. Sampling techniques such as particle Markov chain Monte Carlo [49, 50] and likelihood-free methods [83, 97] such as (ABC) [101, 114] have been utilised to circumvent intractable likelihoods. Given that the ABC method relies heavily on generating simulations, extensions such as sequential Monte Carlo (ABC) [108] have been developed to efficiently search regions of the parameter space where samples are more likely to be accepted. For simulation based inference techniques such as ABC, multifidelity methods [87] have also been developed, orthogonal to approaches such as sequential Monte Carlo, to increase the

scalability and applicability of such schemes. Multifidelity methods integrate information from multiple models of the same system to accelerate tasks such as inference and uncertainty quantification. Typical multifidelity problems work with a true or high-fidelity model, which describes a system of interest with high accuracy, and a low-fidelity model, which is cheaper to generate simulations from at the cost of accuracy. By deploying the low-fidelity model, we are able to accelerate tasks such as inference as opposed to working with the high-fidelity model alone. Multifidelity tasks such as Multilevel Monte Carlo [43, 44] have been incredibly successful, where in the Multilevel Monte Carlo framework, continuous-time SDEs are simulated using progressively finer discretisations. By using variance reduction techniques where estimates have been built from simulations at different discretisations, equivalent to generating simulations from different fidelities, the statistical error of Monte Carlo estimates can be reduced. In fact this Multilevel Monte Carlo result has been exploited to perform likelihood-free inference using ABC [113]. Later in this thesis, we utilise multifidelity methods to accelerate the parameter inference aspect of our framework, specifically working with Multifidelity ABC [93]. Multifidelity ABC utilises simulations generated from both a low-fidelity model and a high-fidelity model to infer the posterior probability distribution of the parameters at a lower computational cost than using high-fidelity methods alone. Specifically, the low-fidelity model can be used to determine whether to reject the sampled parameter without generating a costly high-fidelity simulation, similar to an approach named lazy ABC [91], or to use the low-fidelity model to compute a weight for the sampled parameter without needing to simulate the high-fidelity model. This allows one to construct samples generated from an inferred posterior probability distribution at a lower computational cost than doing so using high-fidelity simulations alone. However, similar to ABC, Multifidelity ABC also naively samples the whole parameter space, including regions which would lead to low acceptance rates of the sampled parameters. To circumvent this, an extension that explores the parameter space more efficiently using sequential Monte Carlo techniques was proposed in the work of [94]. Further model construction and selection via Bayesian design for the construction of novel biological systems subject to specifications is presented in [11, 12, 117].

Once we have a model that accurately represents the underlying biological system, we would like to assert whether certain properties hold on said models. Both probabilistic model checking approaches [9, 69] and statistical model checking approaches [3] have been applied in many areas within biology [71, 72, 119] with tools

such as PRISM [70], providing crucial support to perform procedures for continuous-time Markov chains such as parameter synthesis, which partitions the parameter space to combinations of parameters that satisfy and don't satisfy the property of interest [24, 27, 30, 56]. In [42] for example, they synthesis viable spaces of parameters using constraint solvers to compute the robustness of evolving gene regulatory networks. Probabilistic model checking techniques allows one to numerically calculate the probability of certain events occurring to verify properties of interest, but such approaches tend to suffer from scalability. Methods to improve the scalability of probabilistic model checking techniques have been considered. For example in the work of [20], the authors develop a novel framework that allows one to construct a GP that approximates a CTMC of interest that allows one to perform probabilistic model checking. Whilst extensions to address scalability issues such as [20] is encouraging, statistical based model checking techniques [3] are popular alternatives due to the ease that they can be implemented, the caveat being that the speed and success of the statistical model checking techniques hinges solely on the ability to generate simulations from the stochastic model of interest. A Bayesian approach to statistical model checking is considered in [65] and presented later in this thesis, where the probability of satisfying a property is treated as a random variable, which allows one to define a prior with the associated probability. Parametric verification has been considered from a statistical approach underpinned by GPs: smoothed model checking [21] provides an estimate of the satisfaction probability with uncertainty estimates, and has been used for parameter estimation from Boolean observations [23] and for parameter synthesis [24].

The integration of Bayesian methods and formal verification or data-driven verification is a theme central to the work undertaken this thesis and many different interpretations of integrating these fields have been considered. In [16], they propose a methodology that, given a reachability specification, computes a related probability distribution on the parameter space, and an automaton-based adaptation of the ABC method is introduced to estimate it. Haesaert et al. [54] computes the probability that an underlying stochastic system satisfies a given property using data produced by the system and leveraging system's models. Along this line of work, the integration of verification of parametrised discrete-time Markov chains and Bayesian inference is considered in [88], with an extension to Markov decision processes in [89]. Both [88, 89] work with small finite-state models with fully observable traces, which allows the posterior probability distribution to be calculated analytically and parameters to be synthesised symbolically. On the contrary, in this thesis we work with

partially observed data and stochastic models with intractable likelihoods, and must rely on likelihood-free methods and statistical parameter synthesis procedures. Directly related work is presented in [21]; a Bayesian statistical algorithm was developed that defines a GP [96] over the parameter space based on a few observations of true evaluations of the satisfaction function. The authors build upon the idea presented in [23] and define the satisfaction function as a smooth function of the uncertain parameters of a given CTMC, where this smooth function can be approximated by a GP. This GP allows one to predict the value of the satisfaction probability at every value of the uncertain parameters from individual model simulations at a finite number of distinct parameter values. However, this method faces the same pitfalls as many GPs; the further the data points from the training data, the less accurate the predictions will inherently be. This model checking approach is incorporated into the parameter synthesis problem considered in [24] which builds upon the parameter synthesis problem defined in [31], but differs with the incorporation of the model checking approach presented in [21] and an active learning step being introduced to adaptively refine the synthesis. For DTMCs, a data-driven parameter synthesis of DTMCs modelling biological phenomena is presented in [55]. In [15] the authors introduce a Bayesian verification approach similar to the work introduced in [54, 88] from a statistical perspective. Inspired by the introduction of satisfaction functions in [21] they define a *subjective* satisfaction function that incorporates pre-defined uncertainty, encapsulated in model parameters, which allows them to perform Bayesian statistical model checking [120] on this model with uncertainty. However, being able to accurately incorporate this uncertainty into any model of choice is no trivial task, which motivates the work in this thesis to align with the data-driven construction of models to perform Bayesian verification akin to [54, 88]. Other work that is directly related to our approach is an approach that utilises Markov chain Monte Carlo [53]. Here they infer parameters and formally verify properties using a Markov chain Monte Carlo statistical model checking for deterministic models. In their work they work with data that is perturbed with a known noise function that allows them to work with analytic likelihoods to easily evaluate a posterior probability distribution, and then sample from said distribution to assert whether the sampled parameters, and thus the instantiated model, satisfies a property or not. Whilst interesting, unfortunately in many real-life experiments being able to exactly model the noise and have a full observation of the whole trace of the underlying system is exceedingly rare, and in our work we look to emulate data that is typically gathered from experiments and look to verify properties on stochastic models as opposed to deterministic models.

There has been great interest in integrating data-driven learning of models and verification and in our work we look to introduce a general and easily applicable framework that integrates Bayesian methods and formal verification techniques to verify properties on stochastic models that represent underlying data-generating systems.

Chapter 3

Bayesian Verification of Chemical Reaction Networks

3.1 Introduction

Constructing mathematical models of biological systems with a high degree of accuracy is a prevalent problem in systems and synthetic biology. Attaining full knowledge of many existing biological systems is impossible, making their analysis, prediction, and the designing of novel biological devices an encumbrance. In this chapter we introduce the notion of Bayesian verification which aims to tackle some of the problems prevalent within the realms of synthetic and systems biology. The Bayesian verification methodology allows for the construction of an accurate model from the data to verify that the underlying data-generating system satisfies a given formal property. Thus, by verifying the properties of the model, we can assert quantitatively whether the underlying data generating system satisfies a given property of interest, assuming that the model of choice is an accurate model for the system of interest. We leverage model analysis by means of formal verification, namely quantitative model checking [10], and to describe biological systems of interest, we use CRNs. The end result is the computation of a probability, based on the collected data, that the underlying system satisfies a given formal specification. If the obtained probability is closer to either one or zero, we can confidently draw an assertion on the satisfaction of the property over the underlying biological system. On the other hand, with a moderate probability value, a decision on the experimental setup or on the models needs to be made: we can either collect more data from the experiments, or propose alternative models and start the procedure once more. Our work targets partially known systems that produce noisy observations at discrete points in time, which are commonplace

in biology: these systems are captured by a parametric model class with imperfect knowledge of rates within a known stoichiometry.

The Bayesian verification framework comprises of three phases. First, we propose a parametric model of a given, partially known biological system, and perform parameter synthesis [30] to determine a set of parameters over the parametric models that relates to models verifying the given property. This is performed via PRISM [30, 31]. The second phase, executed in parallel with the first, uses Bayesian inference to infer posterior distributions over the likely values of the parameters, based on data collected from the underlying partially known and discretely observed system. In the third phase, we combine the outputs from the two phases to compute the probability that the model satisfies the desired property, which results in an assertion on the satisfaction of the property over the underlying biological system. We now discuss the details of the credibility calculation, which is central to the Bayesian verification framework.

3.2 Bayesian Verification Calculation

We will apply the Bayesian verification framework on CRNs modelled by pCTMCs and thus we aim to verify properties over pCTMCs. To achieve this, we employ the time-bounded fragment of CSL [8, 69] to define properties over pCTMCs. We recall the satisfaction function defined previously, $\Lambda_\phi(\theta)$, which is the probability that the set of paths from a parametrised stochastic model (pCTMC in this instance) satisfies a path property φ corresponding to the CSL formula ϕ, and we now formally define the Bayesian verification framework.

Consider a real-life, data generating biological system $\mathcal{S}$, where we denote the data generated by the system as y_{obs} and we are interested in verifying a property of interest, say ϕ. Can we use this obtained data and the existing knowledge of the model to formally verify a given property over this system, $\mathcal{S}$?

We will assume that we have sufficient knowledge to propose a parametric stochastic model for the underlying system, denoted by $\mathcal{M}_\theta$ - although the framework can be trivially extended to consider multiple models. For the remainder of this thesis, while we use $\mathcal{M}_\theta$ as a parametrised stochastic model of interest, in this chapter and the following chapters, we will also let $\mathcal{M}_\theta$ represent a pCTMC, and we may use these definitions interchangeably. We define the property of interest, ϕ, in CSL and we also assume that we are able to obtain data, y_{obs}, from the underlying system. There are three aspects to the Bayesian Verification framework: parameter synthesis [30, 31],

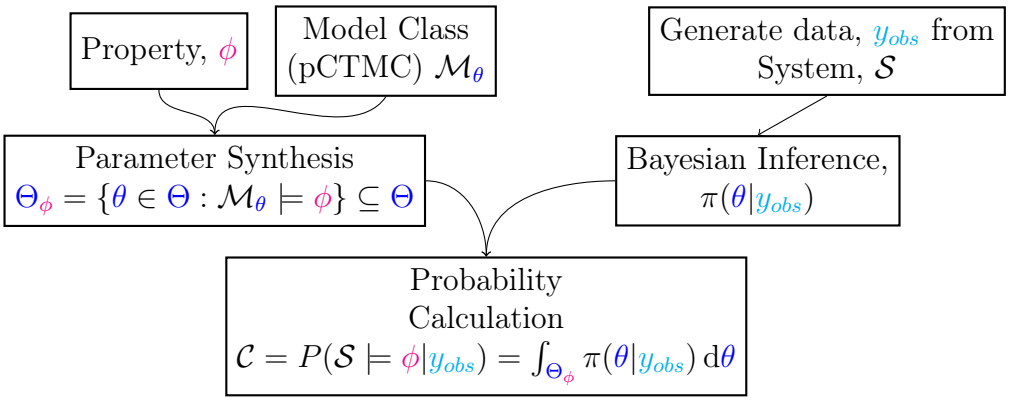


Figure 3.1. Bayesian Verification Framework.

Bayesian inference [25,28,83] and a probability or credibility interval calculation [25]. We shall discuss the data we work with and these methods in detail later but the basic idea is described herein and further illustrated in Figure 3.1. Given a model class $\mathcal{M}_\theta$ and a property of interest, ϕ, we first synthesise a set of parameter valuations $\Theta_\phi \subseteq \Theta$. If we were to choose a vector of parameters θ' such that $\theta' \in \Theta_\phi$, then the paths or traces generated from the induced pCTMC, $\mathcal{M}_{\theta'}$, would satisfy the property of interest with some probability, which we denote as $\Lambda_\phi(\theta')$. We learn the parameters of interest by inferring them from the data via Bayesian inference, to provide us with a posterior distribution, $\pi(\theta \mid y_{obs})$. Once we have this posterior distribution and a synthesised set of parameter regions, Θ_ϕ, we integrate the posterior probability distribution over these regions to obtain a probability on whether the underlying data generating system satisfies the property or not.

3.2.1 Parameter Synthesis

Given a parametric model class $\mathcal{M}_\theta$ and a property ϕ defined in CSL, we synthesise parameter regions that satisfy ϕ using the approach introduced in [31]. We will focus on the threshold synthesis problem. Note that solutions to the threshold synthesis problem may sometimes lead to parameter points that are left undecided, that is, parameter points that either do or do not satisfy the property with a given probability bound, $\sim \zeta$. Let us define this problem formally.

Definition 3.2.1 (Threshold Synthesis)**.** Let $\mathcal{M}_\theta$ be a parametrised stochastic model over a parameter space Θ, ϕ a formula defined in a modal logic, $\sim \zeta$ a threshold where $\zeta \in [0,1]$, $\sim \in \{\leq, <, >, \geq\}$ and $\mathcal{E} > 0$ be a volume tolerance. The threshold synthesis problem is finding a partition $\{\Theta_\phi, \Theta_{\neg\phi}, \Theta_{\mathcal{U}}\}$ of Θ such that:

1. $\forall\theta \in \Theta_\phi.\ \Lambda_\phi(\theta) \sim \zeta$; and

2. $\forall\theta \in \Theta_{\neg\phi}.\ \Lambda_\phi(\theta) \nsim \zeta$; and

3. $vol(\Theta_{\mathcal{U}})/vol(\Theta) \leq \mathcal{E}$

where $\theta \in \Theta_{\mathcal{U}}$ if $\theta \notin \Theta_\phi$ and $\theta \notin \Theta_{\neg\phi}$ and $vol(A)$ is the volume of A .

The goal of parameter synthesis is to synthesise the set of all possible valuations for which the model class $\mathcal{M}_\theta$ satisfies the property ϕ:

$$\Theta_\phi = \{\theta \in \Theta \ :\ \mathcal{M}_\theta \models \phi\}. \tag{3.1}$$

We define the region Θ_ϕ as the feasible set of parameters. Parametric model checking capabilities of the tool introduced in [31] is leveraged in this chapter to perform parameter synthesis over the CTMC constructed from a given pCRN. In the background of this thesis we briefly discussed Bayesian inference and in this next section we discuss in detail how we are able to perform Bayesian inference when working with partially observed and noisy data, which makes it difficult to work with with analytically tractable likelihood functions.

3.2.2 Likelihood-free Bayesian Inference

We briefly recall the Bayesian inference problem introduced earlier. Given a set of observations or data, y_{obs}, and a model (stochastic or deterministic) governed by the parameters θ, say $\mathcal{M}_\theta$, the task of Bayesian inference is to learn the true parameter values given the data and some existing prior knowledge. This is expressed through Bayes' theorem:

$$\pi(\theta \mid y_{obs}) = \frac{p(y_{obs} \mid \theta)\pi(\theta)}{p(y_{obs})}. \tag{3.2}$$

Here $\pi(\theta \mid y_{obs})$ represents the *posterior* distribution, which is the probability density function for the parameter vector, θ, given the data, y_{obs}; $\pi(\theta)$ is the *prior* probability distribution which is the probability density of the parameter vector before considering the data; $p(y_{obs} \mid \theta)$ is the *likelihood* of observing the data given a parameter combination; and $p(y_{obs})$ is the *evidence*, that is, the probability of observing the data over all possible parameter valuations. Assumptions about the parameters are encoded in the prior meanwhile assumptions about the model itself are encoded into the likelihood. The evidence acts as a normalisation constant and ensures the posterior distribution is a proper probability distribution. To estimate the posterior probability distribution, we will utilize Monte Carlo techniques, see Section 2.5. These

Monte Carlo techniques can be utilised to estimate further informative statistics, such as the posterior mean, $\hat{\theta}$, which is used as a Bayesian estimate to θ, the variance of the underlying posterior distribution or the maximum a posteriori probability estimate. In this application of the framework we work with parametrised CRNs, that are modelled by pCTMCs and are concerned with the Bayesian parameter inference problem for pCTMCs. Inferring parameters from pCTMCs is a widely studied problem in the realms of biology [26, 41, 50, 51, 100, 108, 114–116]. The focus of our work in this thesis is to work with data of the form that is typically obtained within biological experiments, which is not necessarily convenient for parameter inference. Specifically, we will look to infer parameters of a stochastic model over time series data that has been observed a finite number of times at discrete points in time - replicating the data that would typically be obtained in an experimental environment.

Partially observed data

Let us consider the case where the data y_{obs} consists of $N_{\mathcal{O}}$ observations of the chemical reaction network state vector at discrete points in time, $\bar{t}_1, \bar{t}_2, \ldots, \bar{t}_{N_{\mathcal{O}}}$. Let $y_{obs} = [y_{obs}(\bar{t}_1), y_{obs}(\bar{t}_2), \ldots, y_{obs}(\bar{t}_{N_{\mathcal{O}}})]^\top$, where $y_{obs}(\bar{t}_i) \in \mathcal{Y} = \mathbb{R}^{N_S}$ represents an observation of the true molecule count sample $y(\bar{t}_i)$, which has a corresponding state x_i in the pCTMC. It is common to incorporate uncertainty in these observations with the use of additive noise [100, 116],

$$y_{obs}(\bar{t}_i) = \mathbf{O}y(\bar{t}_i) + \xi_i, \tag{3.3}$$

where $\mathbf{O}$ is a $O \times N_S$ matrix, $\mathbb{R}^{O \times N_S}$, and ξ is a $O \times 1$ vector of independent Gaussian random variables. The observation vectors $y_{obs}(\bar{t}_i)$, are $O \times 1$ vectors where $O \leq N_S$, which reflects the fact that only a sub-set of chemical species of $y(t_i)$ may be observed. For this work, $\mathbf{O} = I$, where I is an $N_S \times N_S$ identity matrix, recalling that N_S is the number of different chemical species. A more general observation function can be applied, such as $y_{obs}(t) = O(y(t))$, although for this chapter, we will consider observation noise as considered in Equation (3.3).

Due to both the nature of data we are working with and the intractability of the CME [46] that determines the likelihood, we consider likelihood-free methods [83,114]. For motivation as to why we utilise likelihood-free methods, we consider performing inference on the CME.

Intractable Likelihoods for CRNs

We discuss next why we resort to likelihood-free methods for inferring parameters of CRN networks from noisy data observed at discrete points in time. We assume we are

working with a CRN where the reaction rates are parametrised by θ. A biochemical reaction network model can be described by the CME [46],

$$\frac{\mathrm{d}P(x,t \mid x_0)}{\mathrm{d}t} = \sum_{j=1}^{N_R} a_j(x,\theta_j)P(x-\boldsymbol{\upsilon}_j,t \mid x_0) - P(x,t \mid x_0)\sum_{j=1}^{N_R} a_j(x,\theta_j), \tag{3.4}$$

where $P(x,t \mid x_0)$ is the probability that the state of the CRN at time t is x, whilst starting from x_0 at time $t=0$, and where $\upsilon_{i,j} = \upsilon_{i,j}^{+} - \upsilon_{i,j}^{-}$, with $\boldsymbol{\upsilon}_j$ being the jth column of the stoichiometric matrix $\upsilon_{i,j}$. The solution of the CME characterises the exact probability that the model is in any state at any time. Unfortunately, analytic solutions to the CME are only known for very special (and restrictive) CRNs. Furthermore, often the state of the CRN $X(t)$ cannot be observed directly. Instead, an observation of the state vector sample path is observed, $y_{obs}(t) = O(X(t))$, where O is an arbitrary observation function on $X(t)$ that takes values in $\mathcal{Y}$, the space of observations. As a consequence of the analytical intractability of the CME, the likelihood given by $p(X \mid \theta)$ is also seldom expressible analytically, and neither is the Bayesian posterior: hence we resort to likelihood-free methods, such as ABC.

Two popular classes of likelihood-free inference methods available are ABC [14, 101] and pseudo-marginal Markov chain Monte Carlo [5]. In our work, we utilise ABC to infer parameters of our model. Not only do ABC methods allow highly complicated models with intractable likelihoods to be investigated, but also ABC methods are very intuitive and easy to implement - they have proven to be an invaluable tool in the life sciences [13, 14, 73, 108].

Approximate Bayesian Computation

ABC methods [101] produce an approximation to the posterior probability distribution when the likelihood $p(y \mid \theta)$ is intractable. The likelihood is approximated by matching simulated data $y \sim p(y \mid \theta)$ with the observed data y_{obs}, according to some function of the distance metric, $d(y, y_{obs})$, or correspondingly over summary statistics of the simulated and observed data, $d(s, s_{obs})$.

Ideally, the observations y_{obs} are directly mapped to the variables of the model, which is endowed with sufficient statistics y. However, in many real world settings, model variables cannot be fully observed, and moreover outputs y can be perturbed by noise due to measurement error. Since it is in general hard to identify a finite-dimensional set of sufficient statistics, it is common and computationally advantageous to use (insufficient) summary statistics $s = S(y)$, where the function S performs

a simplification of the signals y (e.g., averaging, smoothing, or sampling), and which ideally is so that $\pi(\theta \mid y_{obs}) \approx \pi(\theta \mid s_{obs})$ [90].

The procedure for ABC rejection sampling is as follows: first samples are generated by $\theta^* \sim \pi(\theta)$, each of which is used to generate simulated data $y \sim p(y \mid \theta^*)$, where the proposed sample θ^* is accepted if $d(y, y_{obs}) \leq h$ for some $h \geq 0$, $h \in \mathbb{R}_{\geq 0}$, and rejected if $d(y, y_{obs}) > h$. This procedure is equivalent to drawing a sample (θ, y) from the joint distribution

$$\pi_{ABC}(\theta, y \mid y_{obs}) \propto K_h(d(y, y_{obs}))p(y \mid \theta)\pi(\theta), \tag{3.5}$$

where $K_h(u)$ is a standard smoothing kernel function [101], which depends on a predetermined distance h and on $u = d(y, y_{obs})$.

Introducing smoothing kernels allows one to discriminate between samples for which the generating dataset y are either very close to the observed data y_{obs} ($d(y, y_{obs}) \to 0$) and samples for which the associated simulations are further away from the observed data, but still within an acceptable distance ($d(y, y_{obs}) \to h$). More details on the motivation and selection of these smoothing kernel functions and distance metrics can be found in [101]. In the remainder of our work, we work with a simple kernel function, namely the indicator function. A standard choice we use for the smoothing kernel function is the indicator function, $K_h(d(y, y_{obs}) = \mathbb{I}(d(y, y_{obs}) \leq h)$ and the distance metric we use in this work will simply be the Euclidean distance, $d(y, y_{obs}) = \|y - y_{obs}\|$. Following the arguments and notation provided in [93] we let $\Omega_h(d, y_{obs}) = \{y \in \mathcal{Y} \mid d(y, y_{obs}) \leq h\}$ denote the neighbourhood of simulations y (taking values in the output space $\mathcal{Y}$) generated from θ that is at most a distance of h away from the observed data y_{obs}. Given that the observations y_{obs} and choice of distance metric are fixed and decided beforehand, we can drop this notation from the neighbourhood definition introduced above ($\Omega_h(d, y_{obs}) = \Omega_h$) and our smoothing kernel simplifies to a function of the simulated data, y and distance threshold h:

$$K_h(d(y, y_{obs})) = \mathbb{1}_{\Omega_h}(y) = \mathbb{I}(y \in \Omega_h) = \begin{cases} 1, & \text{if } y \in \Omega_h, \\ 0, & \text{if } y \notin \Omega_h. \end{cases} \tag{3.6}$$

The ABC rejection sampling algorithm is then equivalent to drawing samples (θ, y) from the joint distribution

$$\pi_{ABC}(\theta, y \mid y_{obs}) \propto \mathbb{1}_{\Omega_h}(y)p(y \mid \theta)\pi(\theta), \tag{3.7}$$

Accordingly, the ABC approximation to the true posterior distribution is

$$\pi_{ABC}(\theta \mid y_{obs}) = \int \pi_{ABC}(\theta, y \mid y_{obs}) \, \mathrm{d}y. \tag{3.8}$$

As $h \to 0$ samples from the true posterior distribution are obtained [101] as:

$$\lim_{h \to 0} \pi_{ABC}(\theta \mid y_{obs}) \propto \int \delta_{y_{obs}}(y) p(y \mid \theta) \pi(\theta) \, \mathrm{d}y = p(y_{obs} \mid \theta) \pi(\theta),$$

where $\delta_{y_{obs}}(y)$ is the Dirac delta measure, where $\delta_x(A) = 1$ if $x \in A$ and $\delta_x(A) = 0$ otherwise. In practice, it is highly unlikely that $y \approx y_{obs}$ can be generated from $p(y \mid \theta)$, thus a non-trivial distance threshold h is needed. Furthermore, the full datasets y_{obs} and y are often replaced by summary statistics s_{obs} and s, respectively, leading to sampling from the posterior distribution $\pi_{ABC}(\theta \mid s_{obs})$. The ABC approximation to $\pi(\theta \mid s_{obs})$ is given by

$$\pi_{ABC}(\theta \mid s_{obs}) \propto \int \mathbb{1}_{\Omega_h}(s) p(y \mid \theta) \pi(\theta) \, \mathrm{d}y, \tag{3.9}$$

where, by slight abuse of notation, $\mathbb{1}_{\Omega_h}(s)$ is defined as for y, y_{obs}.

Algorithm 2 ABC Rejection Sampler (ABC-RS)

Input:

- Observed data y_{obs}
- Prior distribution $\pi(\theta)$ and data generating function $p(y \mid \theta)$
- A neighbourhood $\Omega_h = \{y \in \mathcal{Y} \mid d(y, y_{obs}) \leq h\}$ with corresponding kernel $\mathbb{1}_{\Omega_h}(y)$
- $N > 0$, Monte Carlo sample size

Output:

- $\{\theta^{(i)}, w^{(i)}\}_{i=1}^N$, accepted parameters with corresponding weights.

1: **for** $i = 0, \ldots, N$: **do**
2: Generate $\theta^{(i)} \sim \pi(\theta)$
3: Simulate $y \sim p(y \mid \theta^*)$
4: Calculate $w^{(i)} = \mathbb{1}_{\Omega_h}(y)$
5: **end for**
6: **return** $\{\theta^{(i)}, w^{(i)}\}_{i=1}^N$

Approximate Bayesian Computation - Sequential Monte Carlo

The major issue with standard ABC is that if the prior $\pi(\theta)$ differs significantly from the posterior distribution, $p(\theta \mid y_{obs})$, then the acceptance rates, namely the rates at which sampled parameters are accepted, will be low, thus resulting in more proposed parameters and associated simulations, which leads to an increase in computational burden. Approximate Bayesian Computation - Sequential Monte Carlo (ABC-SMC) [108] techniques are developed to mitigate this issue. ABC-SMC algorithms [107, 108] are designed to overcome this burden by constructing a sequence of slowly-changing intermediate distributions, $f_m(\theta)$, $m = 0, \ldots, M$, where $f_0(\theta) = \pi(\theta)$ is the initial sampling distribution and $f_M(\theta) = f(\theta)$ is the target distribution of interest, namely

the approximated posterior, $\pi_{ABC}(\theta \mid s_{obs})$. A population of particles or samples from generation m, $\theta_m^{(i)}$, where $i = 1, \ldots, N$ is the number of particles, are propagated between these distributions sequentially, so that these intermediary distributions act as an importance sampling scheme [101], which is a technique used to sample from a distribution that over-weights specific regions of interest.

The success of importance sampling hinges on the choice of the importance probability distribution which generates sample proposals. Consider an importance probability distribution of the form $\hat{q}(\theta) = q(\theta)/Z_q$, where $q(\theta) > 0$ for all θ in the support of $\pi(\theta)$ and Z_q is a normalization constant. An example of importance sampling for ABC [101] with importance distribution $\hat{q}$ is presented in Algorithm 3. The ABC

Algorithm 3 ABC Importance Sampler (ABC-IS)

Input:

- Observed data y_{obs}
- Prior distribution $\pi(\theta)$ and data generating function $p(y \mid \theta)$
- Importance distribution $\hat{q}$ proportional to q
- A neighbourhood $\Omega_h = \{y \in \mathcal{Y} \mid d(y, y_{obs}) \leq h\}$ with corresponding kernel $\mathbb{1}_{\Omega_h}(y)$
- $N > 0$, Monte Carlo sample size
- Summary statistic function $s = S(y)$
- $B_m > 0$ number of simulations

Output:

- $\{\theta^{(i)}, w^{(i)}\}_{i=1}^N$, accepted parameters with corresponding weights.

for $i = 0, \ldots, N$: **do**
  Generate $\theta^{(i)} \sim \hat{q}(\theta)$
  for $b = 1, \ldots, B_m$: **do**
    Simulate $y_i \sim p(y \mid \theta^{(i)})$
  end for
  Calculate $w^{(i)} = \left[\pi(\theta^{(i)})/q(\theta^{(i)})\right] \cdot \sum_{b=1}^{B_m} \mathbb{1}_{\Omega_h}(y_b)$
end for
return $\{\theta^{(i)}, w^{(i)}\}_{i=1}^N$

importance sampling distribution first samples a parameter $\theta^* \sim \hat{q}(\theta)$ and simulates a corresponding trace $y \sim p(y \mid \theta)$ which is accepted or rejected with a corresponding weight

$$w(\theta^*, y) = \left[\pi(\theta^*)/q(\theta)\right] \cdot \mathbb{1}_{\Omega_h}(y) \geq 0, \tag{3.10}$$

to produce a weighted sample $\{\theta^{(i)}, w^{(i)}\}$. The ABC Rejection Sampling (ABC-RS) algorithm in Algorithm 2 is equivalent to ABC-IS in Algorithm 3 when the importance distribution is chosen to be equal to the prior $\hat{q} = \pi$. If we were to choose an importance distribution that closely matches the posterior probability distribution, we would accept more parameter proposals and less simulations would be required

to approximate the posterior distribution as opposed to naively sampling from the prior distribution. The difficulty of course is choosing such an importance distribution when we are attempting to learn the posterior probability distribution. The ABC-SMC technique attempts to bridge the gap between the prior $\pi(\theta)$ and the (unknown) posterior $\pi(\theta \mid s_{obs})$ using importance sampling via a series of intermediary distributions. In the ABC-SMC framework, a natural choice for the sequence of intermediary distributions is

$$f_m(\theta) = \pi_{ABC}^{h_m}(\theta, s \mid s_{obs}) \propto K_{h_m}(s)p(y \mid \theta)\pi(\theta), \tag{3.11}$$

where $m = 0, \ldots, M$ and h_m is a monotonically decreasing sequence, namely such that $h_m > h_{m+1} \geq 0$. The neighbourhood where simulations (and in turn parameter proposals are accepted) now depends on the distance h_m. We expect that $\lim_{h_m \to 0} \pi_{ABC}^{h_m}(\theta \mid s_{obs}) = \pi(\theta \mid s_{obs})$ [101], and that the more samples N are generated, the more accurate the approximated quantity will become.

A key part of the ABC-SMC scheme is the generation of samples θ^* and the setting of weights (which is typical for other importance sampling schemes). Sample θ^* is initially ($m = 0$) taken from the prior and subsequently ($m > 0$) sampled from the intermediary distributions $f_{m-1}(\theta)$ through its corresponding weights (see below), as parameter $\theta_{m-1}^{(j)}$. Afterwards, θ^* is perturbed into θ^{**} by a kernel, $F_m(\theta^{**} \mid \theta^*)$. This procedure of sampling parameters using previous weights and perturbing the parameters using the kernel $F_m(\theta^{**}|\theta^*)$ is equivalent to sampling from an importance distribution $\hat{q}_m$ that is constructed from the previous weights and samples generated at generation $m-1$, which is defined as being proportional to $q(\theta)$;

$$q_m(\theta_m^{(i)}) = \begin{cases} \sum_{i=1}^N w_{m-1}^{(i)} F_{m-1}(\theta_m^{(i)}|\theta_{m-1}^{(i)}) / \sum_{j=1}^N w_{m-1}^{(i)}, & \pi(\theta_m^{(i)}) > 0 \\ 0, & \text{otherwise.} \end{cases} \tag{3.12}$$

For the perturbed parameter, θ^{**}, a number of B_m simulations y_b, and in turn s^b, are generated from the sampled model, $p(y \mid \theta^{**})$, and the quantity $b_m(\theta^{**}) = \sum_{b=1}^{B_m} \mathbb{1}_{\Omega_{h_m}}(s^b)$ is calculated. If $b_m(\theta^{**}) = 0$, then θ^{**} is discarded and we resample θ^* again. Otherwise, the accepted θ^{**} results in the pair $\{\theta_m^{(i)}, w_m^{(i)}\}$, where the corresponding weights $w_m^{(i)}$ are set to

$$w_m^{(i)} = \begin{cases} b_m\left(\theta_m^{(i)}\right), & \text{if } m = 0 \\ \dfrac{\pi\left(\theta_m^{(i)}\right) b_m\left(\theta_m^{(i)}\right)}{\sum_{j=1}^N w_{m-1}^{(j)} F_m\left(\theta_m^{(i)} \mid \theta_{m-1}^{(j)}\right)}, & \text{if } m > 0 \end{cases} \tag{3.13}$$

and later normalised, after re-sampling each ith particle, $i = 1, \ldots, N$. If B_m is large, the estimate of $\pi_{ABC}(\theta \mid s_{obs})$ is accurate, which implies the acceptance probability is accurate, but it might come at the cost of many Monte Carlo draws. Conversely, if B_m is small, the acceptance probability is cheaper to evaluate [19] but can become highly variable.

The algorithm controls the transitioning between the intermediary distributions $f_{m-1}(\theta)$ and $f_m(\theta)$, by setting a user-inputted rate λ, at which the thresholds h_m reduce until the algorithm stops. Stopping rules for ABC-SMC schemes vary: here, we have opted for terminating the algorithm after a predetermined number M of steps (a.k.a. epochs). The algorithm returns weighted samples,

$$\left\{\theta_M^{(i)}, w_M^{(i)}\right\} \sim \pi_{ABC}^{h_M}(\theta \mid s_{obs}) \propto \int \mathbb{1}_{\Omega_{h_M}}(s) p(y \mid \theta)\pi(\theta)\,\mathrm{d}y.$$

We note the ABC-SMC algorithm is presented in Algorithm 4, and we note this algorithm is equivalent to setting $\hat{q} = f_m(\theta)$ and performing Algorithm 3 at each threshold $m = 0 \ldots, M$ whilst reducing the sequence of distances $h_{m+1} < h_m$.

Algorithm 4 ABC-SMC

Input:

- Prior $\pi(\theta)$ and data-generating likelihood function $p(y_{obs} \mid \theta)$
- A neighbourhood that evolves with distance thresholds $\Omega_{h_m} = \{y \in \mathcal{Y} \mid d(y, y_{obs}) \leq h_m\}$ with corresponding kernel $K_{h_m}(y)$ $(= \mathbb{1}_{\Omega_{h_m}}(y)$ in this paper)
- $N > 0$, number of particles used to estimate posterior distributions
- Sequence of perturbation kernels $F_m(\theta \mid \theta^*)$, $m = 0, 1, \ldots, M$
- A quantile $\lambda \in [0, 1]$ to control the rate of decrease of h_m
- Summary statistic function $s = S(y)$
- $B_m > 0$, number of simulations per sampled particle. For stochastic systems $B_m > 1$

Output:

- Set of weighted parameter vectors $\left\{\theta_M^{(i)}, w_M^{(i)}\right\}_{i=1}^N$ drawn from $\pi_{ABC}(\theta \mid s_{obs}) \propto \int K_{h_M}(s) p(y \mid \theta) \pi(\theta)\, \mathrm{d}s$

1: **for** $m = 0, \ldots, M$: **do**
2: **for** $i = 0, \ldots, N$: **do**
3: **if** $m = 0$ **then**
4: Generate $\theta^{**} \sim \pi(\theta)$
5: **else**
6: Generate θ^* from the previous population $\{\theta_{m-1}^{(i)}\}$ with weights $\{w_{m-1}^{(i)}\}$ and perturb the particle to obtain $\theta^{**} \sim F_m(\theta \mid \theta^*)$
7: **end if**
8: **if** $\pi(\theta^{**}) = 0$ **then**
9: **goto** line 4
10: **end if**
11: **for** $b = 1, \ldots, B_m$: **do**
12: Generate $y_b \sim p(y \mid \theta^{**})$
13: Calculate $s^b = S(y_b)$
14: **end for**
15: Calculate $b_m(\theta^{**}) = \sum_{b=1}^{B_m} K_{h_m}(s^b)$
16: **if** $b_m(\theta^{**}) = 0$ **then**
17: **goto** line 4
18: **end if**
19: Set $\theta_m^{(i)} = \theta^{**}$, $\bar{d}_m^{(i)} = \frac{1}{B_m} \sum_{b=1}^{B_m} \|s^b - s_{obs}\|$ and calculate

20:
$$w_m^{(i)} = \begin{cases} b_m\left(\theta_m^{(i)}\right), & \text{if } t = 0, \\ \dfrac{\pi\left(\theta_m^{(i)}\right) b_m\left(\theta_m^{(i)}\right)}{\sum_{j=1}^N w_{m-1}^{(j)} F_m\left(\theta_m^{(i)} \mid \theta_{m-1}^{(j)}\right)}, & \text{if } t > 0, \end{cases}$$

21: **end for**
22: Generate $\{\theta_m^{(i)}, w_m^{(i)}\}_{i=1}^N$ using Algorithm 3 (ABC-IS) with importance distribution $f_m(\theta)$ and neighbourhood Ω_{h_m}
23: Normalise weights: $w_m^{(i)} \leftarrow w_m^{(i)} / \left(\sum_{i=1}^N w_m^{(i)}\right)$
24: Set $h_{m+1} = (\lambda/N) \sum_{i=1}^N \bar{d}_m^{(i)}$
25: **end for**
26: **return** $\left\{\left(\theta_M^{(i)}, w_M^{(i)}\right)\right\}_{i=1}^N$

3.2.3 Credibility Computation

In the final phase of our approach, a probability estimate is computed corresponding to the satisfaction of a CSL specification formula ϕ by a system of interest such that $\mathcal{S} \models \phi$. To calculate the probability that the system satisfies the specified property, we require two inputs - the posterior distribution over the whole set of kinetic parameters, θ, discussed in the previous section, Section 3.2.2, and the feasible set of parameters that have been calculated in Section 3.2.1:

Definition 3.2.2. Given a CSL specification ϕ and observed data y_{obs} from the system $\mathcal{S}$, the probability that $\mathcal{S} \models \phi$ is given by

$$\mathcal{C} = P(\mathcal{S} \models \phi \mid y_{obs}) = \int_{\Theta_\phi} \pi(\theta \mid y_{obs}) \, \mathrm{d}\theta, \tag{3.14}$$

where Θ_ϕ denotes the feasible set of parameters.

Given that we sample a finite number of parameter samples from the ABC posterior, $\theta \sim \pi_{ABC}(\theta \mid y_{obs})$, we estimate this integral with the use of Monte Carlo methods. We consider now a case study where we implement this Bayesian verification framework.

3.3 Bayesian Verification Case Study

Experimental Setup

All experiments have been run on an Intel(R) Xeon(R) CPU E5-1660 v3 @ 3.00GHz, 16 cores with 16GB memory. We work with partially observed data of the type discussed in Section 3.2.2. Data is of the form $y_{obs}(\bar{t}_i) = \bar{y}(\bar{t}_i) + \xi$, where $\bar{y}(t)$ is a *single* true trace used to generate the observation data and in the case of noisy observations, the additive noise for each observation j will be given by, $\xi_j \sim \mathcal{N}(0, \sigma^2)$ and $\sigma = 2$. The data generating system, $\mathcal{S}$, will in fact be a pCTMC with a chosen combination of parameters, of which we consider two. The first combination, $\theta_\phi \in \Theta_\phi$, have been chosen such that $\mathcal{M}_{\theta_\phi} \models \phi$, that is, the pCTMC model $\mathcal{M}_{\theta_\phi}$, governed by θ_ϕ, satisfies the property of interest. The second combination we choose are the parameters given by $\theta_{\neg\phi} \in \Theta \setminus \Theta_\phi$, such that $\mathcal{M}_{\theta_{\neg\phi}} \not\models \phi$. We will consider the scenario where we have both noisy and noiseless observations. To summarise, we have instances where we observe either 10 or 20 data points per species, which can be either noisy or noiseless and working with data that has been produced by either $\mathcal{M}_{\theta_\phi}$ or $\mathcal{M}_{\theta_{\neg\phi}}$. To ensure the inference does not depend on the initialisation of

the ABC-SMC technique, we ran 10 independent batches with $N = 1000$ particles each and calculated the corresponding weighted means and variance of the batches to derive the inferred mean and credibility intervals, where we set $B_m = 1$ and the number of thresholds to $M = 10$. The ABC-SMC method produces sampled particles from the posterior probability distribution, which we use to calculate the posterior mean, $\hat{\theta}$, and the covariance, Σ, of the kinetic parameters. We assume the parameters are independent of each other, thus the non-diagonal elements of the covariance matrix are equal to 0. The inferred parameters $\pi_{ABC}(\theta \mid y_{obs})$ can be described by a multivariate Normal distribution $\mathcal{N}(\hat{\theta}, \Sigma)$, and that is what we pursue in this work. We will compare this probabilistic model checking approach versus a Bayesian hypothesis testing model checking approach [65]. Statistical model checking approaches collect sample trajectories from the system, and then determine whether the trajectories satisfy a given property and apply statistical techniques, such as calculation of credibility intervals and hypothesis testing, to decide whether the system satisfies the property or not with a degree of probability.

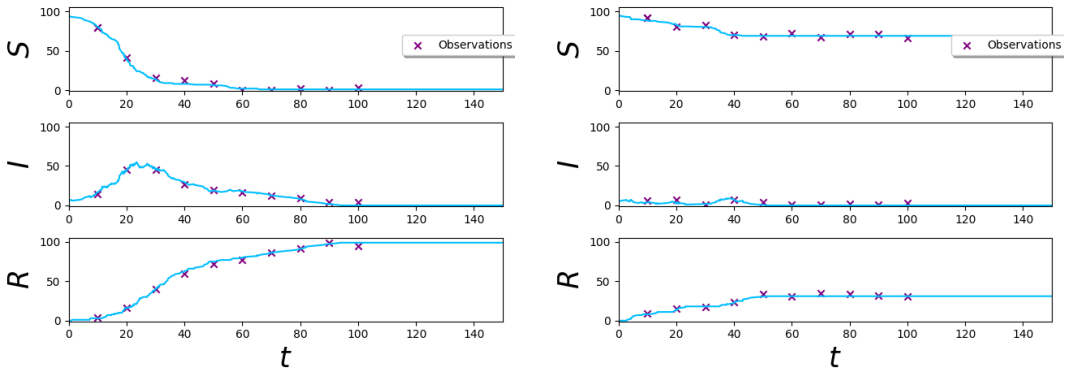
3.3.1 Case Study: Finite-state SIR Model

We take into account the stochastic epidemic model [67], known alternatively as the SIR model. Epidemiological models of this type can be modelled by CRNs [31]. We treat individuals of a group, whether susceptible, infected or recovered, as individual molecules of a given chemical species. In a similar fashion as to how infected individuals can come into contact and infect other individuals, molecules can collide with other molecules to undertake a chemical reaction. The model describes the epidemic dynamics of three types, the susceptible group (S), the infected group (I), and recovered group of individuals (R) - here we consider S, I and R to represent three different chemical species. The epidemic dynamics can be described with mass action kinetics:

$$S + I \xrightarrow{k_i} 2I,\ I \xrightarrow{k_r} R. \tag{3.15}$$

Whenever a susceptible individual S encounters an infected individual I, the susceptible individual becomes infected with rate k_i and infected individuals recover at rate k_r. The CRN is governed by the parameters $\theta = (k_i, k_r)$, where each state of the CTMC describes the combination of the number of molecules (or individuals) for each species. Specifically, the CTMC models the evolution of the state vector $X(t) = [S(t), I(t), R(t)]^\top$, which gives the number of individuals of each species at time t. The problem we consider is as follows. We assume that initially there are

95 individuals of species S, 5 individuals of species I and 0 individuals of species R, thus, the initial state is $X(0) = x_0 = (S_0, I_0, R_0) = (95, 5, 0)$. We wish to verify the following property, $\phi = P_{>0.1}[(I > 0)U^{[100,150]}(I = 0)]$, i.e. whether, with a probability greater than $\zeta = 0.1$, the infection dies out strictly within the interval of $t = 100$ and $t = 150$ seconds. The data is produced by both $\mathcal{M}_{\theta_\phi}$ and $\mathcal{M}_{\theta_{\neg\phi}}$, where $\theta_\phi = (0.002, 0.05)$ and $\theta_{\neg\phi} = (0.002, 0.18)$. The type of data we work with through the duration of this thesis is presented in Figure 3.2, where we present the y_{obs} generated from $\mathcal{M}_{\theta_\phi}$ and $\mathcal{M}_{\theta_{\neg\phi}}$.



(a) $y_{obs} \sim \mathcal{M}_{\theta_\phi}$. **(b)** $y_{obs} \sim \mathcal{M}_{\theta_{\neg\phi}}$.

Figure 3.2. Data generated by (3.2a) $\mathcal{M}_{\theta_\phi}$ and (3.2b) $\mathcal{M}_{\theta_{\neg\phi}}$, taking 10 observations, represented by the purple crosses, of each species taken between time points $\bar{t}_i \in [0, 100]$ with additive noise $\sigma = 2$ such that $y(\bar{t}_i) = y(\bar{t}_i) + \xi_i$, where $\xi_i \sim \mathcal{N}(0, \sigma)$. In this case, $\bar{t}_i \in \{0, 10, 20, \ldots, 100\}$.

In the first phase of our method, we synthesise the feasible set of parameters, Θ_ϕ. For the parameter synthesis technique used in our work [30, 31], we define parameter bounds and we confine our parameters to the set $\Theta = [k_i^\perp, k_i^\top] \times [k_r^\perp, k_r^\top] = [5 \times 10^{-5}, 0.003] \times [0.005, 0.2]$. Focusing on the threshold synthesis problem, the work in [**?**, 30] The results of the parameter synthesis is shown in Figure 3.3 and took a total of 3096 seconds (51.6 minutes) to compute. The second phase of our approach involves learning the kinetic parameters from data, y_{obs}, via the ABC-SMC method introduced in Section 3.2.2. To showcase the accuracy of our method, we consider different data scenarios. We take into account observed data where the observations are either distorted or not by additive noise, and the aforementioned two cases but with additional observed data points. We reiterate that y_{obs} is generated from a single trace from a true stochastic system. A full list of different data scenarios and corresponding inferred parameters can be seen in Table 3.1. As expected, if we

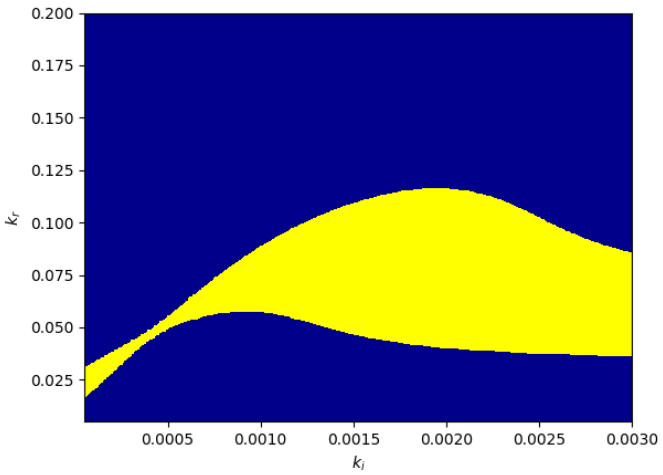


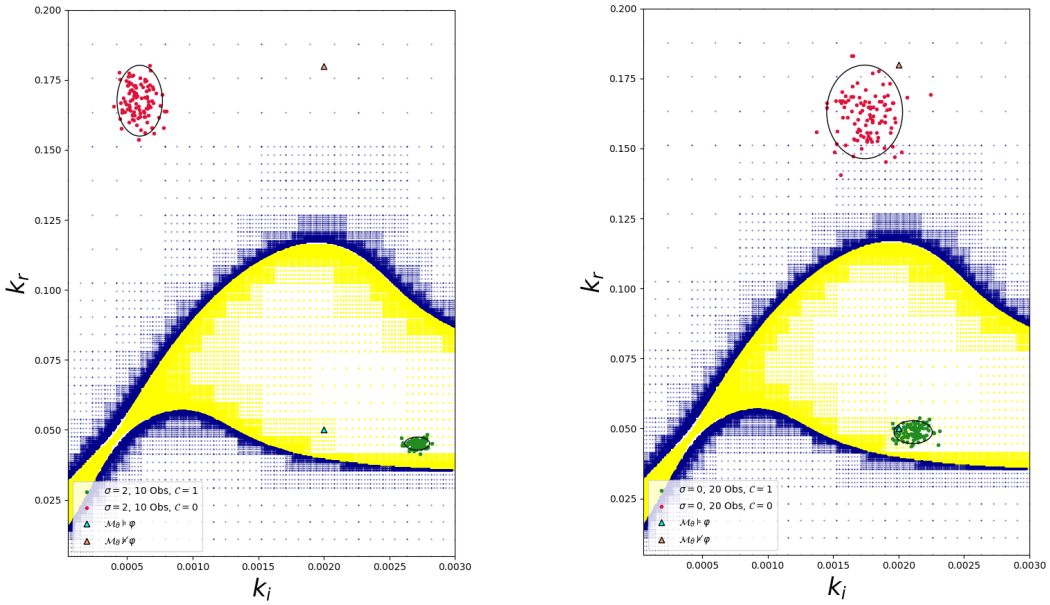
Figure 3.3. Synthesised Parameter regions are shown here. The feasible set of parameters, Θ_ϕ, is shown in yellow (lighter colour), meanwhile the infeasible set of parameters, $\Theta_{\neg\phi}$, is shown in blue (darker colour) $\Theta_{\neg\phi}$, with the undecided areas (if any) shown in white, the set $\mathcal{U}$.

observe more, noiseless data points then our inferred parameters converge to the true parameters, θ_ϕ or $\theta_{\neg\phi}$. The accuracy decreases drastically for data produced by the model $\mathcal{M}_{\theta_{\neg\phi}}$. This is due to the largely uninformative observations as the samples reach steady state, see Figure 3.2b. For any parametrisation, the model reaches the same steady state - this means that in order to increase accuracy, more observations should be taken during the transient period of the model. With a smarter choice of summary statistic, we could further improve the accuracy of these inference results, for example, we could consider a summary statistic that uses more data in the transient phase of the underlying system as we know that the underlying system reaches the same steady state, regardless of parametrisation. Examples of a simulated trace using inferred parameters are given in Figure 3.5.

An alternative way to formally verify properties given data y_{obs} would be to perform SMC on the inferred posterior probability distribution, and we consider comparing our Bayesian verification technique to this approach. Specifically, we consider utilising SMC by Bayesian hypothesis testing [65]. The two hypotheses we wish to test are $\mathcal{H}_0 : \Lambda_\phi(\theta) \geq \zeta$ and $\mathcal{H}_1 : \Lambda_\phi(\theta) < \zeta$ for models $\mathcal{M}_\theta$ instantiated by parameters $\theta \in \Theta$, where we recall that ζ is the probability threshold in property ϕ. With an instantiated model $\mathcal{M}_\theta$, we generate traces $y \sim p(y \mid \theta)$ and check whether $y \models \varphi$, where φ is the associated path formula of the property ϕ. After we verify whether each path satisfies the property, we calculate a Bayes factor, $\mathcal{B}_a$, which is a likelihood ratio of the marginal likelihood of the two competing hypotheses, and

accept $\mathcal{H}_0$ if $\mathcal{B}_a > K_{\mathcal{B}}$, where $K_{\mathcal{B}}$ is a user chosen threshold. Likewise, we accept $\mathcal{H}_1$ if $\mathcal{B}_a < \frac{1}{K_{\mathcal{B}_a}}$. The Bayes factor is defined as $\mathcal{B}_a = p(\mathcal{V} \mid \mathcal{H}_0)/p(\mathcal{V} \mid \mathcal{H}_1)$ and can be used as a measure relative of confidence in $\mathcal{H}_0$ against the alternative hypothesis $\mathcal{H}_1$, where $\mathcal{V}$ is data gathered from a particular experiment - in this case the number of simulations that satisfy the path formula φ. We set the threshold $K_{\mathcal{B}_a} = 1000$ and we accept $\mathcal{H}_0$ when the likelihood of observing $\mathcal{V}$ assuming $\mathcal{H}_0$ is true is 1000 times more likely compared to $\mathcal{H}_1$. Further details about computing the Bayes factor can be found in [65]. Due to this simulation based approach to verification, this SMC technique requires a high number of simulated trajectories over a given model $\mathcal{M}_\theta$ to determine whether $\mathcal{M}_\theta \models \phi$. The issue with SMC based approaches however is that we would need to perform SMC for each sampled parameter $\theta \sim \pi_{ABC}(\theta \mid y_{obs})$ and we would require a high number of samples to adequately cover the posterior probability distribution, especially if the distribution were to have a high variance. However, by using the parameter synthesis approach in our Bayesian verification framework, we would only need to integrate the posterior distribution $\pi(\theta|y_{obs})$ over the feasible parameter set Θ_ϕ to obtain a probability whether this property is satisfied or not. The alternative approach of working with SMC methods is illustrated in Figure 3.4, where we generate sampled parameters from the posterior, $\theta^* \sim \pi(\theta \mid y_{obs})$, and then perform SMC via Bayesian hypothesis testing to estimate $\Lambda_\phi(\theta)$.

For both the SMC approach and our method, we first had to infer the parameters to obtain a posterior probability distribution, which in this case, is a bivariate normal distribution. For SMC, we sampled 100 independent evaluations of the parameters and performed SMC on these instantiated models to estimate the probability of satisfying the property. The sampled parameters are represented by the points represented by points in Figure 3.4, meanwhile the 95% credibility interval for the inferred parameters are represented by the black ellipses. The computation time for the SMC approach was 756 seconds (12.6 minutes). With our approach, we simply need to integrate the bivariate normal distribution over the feasible parameter regions over the parameter regions to obtain the values in Table 3.1, and we do this numerically via slice sampling [84]. Both our technique and the SMC approach are in agreement, but for the SMC approach we would require a larger number of sampled parameters to verify whether or not the entirety of the posterior probability distribution lies in these feasible regions. Despite the fact that the parameter synthesis for the whole region takes longer to compute, the exhaustive parameter synthesis technique provides us a picture of the whole parameter space which is useful for further experiments and can be done entirely offline. Given the simulation based nature of both the inference



(a) $\pi(\theta \mid y_{obs})$ where $N_{\mathcal{O}} = 10$, $\sigma = 2$. **(b)** $\pi(\theta \mid y_{obs})$ where $N_{\mathcal{O}} = 20$, $\sigma = 0$.

Figure 3.4. SMC is performed over $\theta \sim \pi(\theta \mid y_{obs})$ where (3.4a) $N_{\mathcal{O}} = 10$ and $\sigma = 2$ and (3.4b) $N_{\mathcal{O}} = 20$ where $\sigma = 0$. Green points represent $\mathcal{M}_\theta \models \phi$ meanwhile red points represent $\mathcal{M}_\theta \not\models \phi$. True data is generated from $\mathcal{M}_{\theta_\phi}$ where θ_ϕ (cyan $\triangle$) and $\mathcal{M}_{\theta_{\neg\phi}}$ where $\theta_{\neg\phi}$ (orange $\triangle$). The black elliptical lines represent the 95% credibility intervals of the inferred posterior distribution. The yellow (or blue) points represent the parameter valuation regions that satisfy (Θ_ϕ) (or don't satisfy, $\Theta_{\neg\phi}$) the property.

and SMC methods, these methods break down if simulating traces for CRNs is costly. Fortunately, we can work with approximations to the exact Gillespie/SSA algorithm as mentioned previously, specifically working with either the CLE approximation or the tau-leaping method.

3.4 Initial challenges of Bayesian verification

In this chapter we introduced the idea of Bayesian verification, which is a data-driven approach for the verification of data-generating stochastic systems, applying this framework to CRNs modelled as pCTMCs. The framework integrates Bayesian inference and formal verification techniques, specifically numerical parameter synthesis. We demonstrated how to infer parameters using noisy and discretely observed data using ABC and with the inferred posterior probability distribution of the parameters at hand, we calculate the probability that the underlying data generating

system satisfies a property by integrating over the synthesised feasible parameter regions. Thus, given data from an underlying system, we can quantitatively assert whether properties of the underlying system are satisfied or not. With this approach we calculate a single probability with respect to the entire posterior probability distribution, meanwhile if we were to resort to SMC approaches, we would be required to sample a high number of parameters to cover the entirety of the posterior distribution, thereon generate traces to determine whether a property is satisfied or not. However, the Bayesian verification technique in its current form has some limitations. The parameter synthesis technique utilised for this work hinges on a numeric or analytic technique that provides accurate results, but is generally infeasible for many systems of interest. In particular, due to their computational and memory requirements, numerical techniques such as those deployed by probabilistic model checking do not scale particularly well, and this is where statistical approaches such as SMC become preferable. Generally, numerical techniques are suitable for real-world systems that have small finite state spaces and an explicit system description. The parameter synthesis technique [31] that we deploy depends on CTMCs that are amenable to uniformisation, that allows one to construct a DTMC that can be used to calculate the probability of satisfying a property numerically. Thus for large state spaces or for CRNs of a certain structure, parameter synthesis techniques such as [31] are no longer viable due to the high or infinite dimensionality of the state space, making the uniformisation of CTMCs particularly difficult. Statistical approaches for verification become attractive as we simply only need to be able to generate traces from a stochastic system of interest, where we can verify whether individual traces satisfy a property of interest. Although the Bayesian verification technique introduced in this chapter works well, the success of the method hinges largely on the ability to partition the parameter space via a parameter synthesis. The parameter synthesis technique we considered hinges on finite-state CTMCs and doesn't scale particularly well to models of interest, and in this next chapter, we consider how we can mitigate this limitation and improve upon the scalability of this work by integrating learning and verification further.

Inferred Parameters					
Data	**True par.**	**Mean**	**Std. Dev.**	**Prob.**	**Comp. Time (s)**
10 Obs. with Noise	θ_ϕ	$\mu_{k_i} = 0.0027$ $\mu_{k_r} = 0.0451$	$\sigma_{k_i} = 4.7 \times 10^{-5}$ $\sigma_{k_r} = 0.0012$	1	11791
	$\theta_{\neg\phi}$	$\mu_{k_i} = 0.0006$ $\mu_{k_r} = 0.1676$	$\sigma_{k_i} = 8.7 \times 10^{-5}$ $\sigma_{k_r} = 0.0063$	0	75
20 Obs. with Noise	θ_ϕ	$\mu_{k_i} = 0.0022$ $\mu_{k_r} = 0.0468$	$\sigma_{k_i} = 0.0001$ $\sigma_{k_r} = 0.0036$	0.9901	10840
	$\theta_{\neg\phi}$	$\mu_{k_i} = 0.0015$ $\mu_{k_r} = 0.1620$	$\sigma_{k_i} = 0.0002$ $\sigma_{k_r} = 0.0104$	0	3256
10 Obs without Noise	θ_ϕ	$\mu_{k_i} = 0.0019$ $\mu_{k_r} = 0.0549$	$\sigma_{k_i} = 8.8 \times 10^{-5}$ $\sigma_{k_r} = 0.00514$	0.9969	7585
	$\theta_{\neg\phi}$	$\mu_{k_i} = 0.0015$ $\mu_{k_r} = 0.1565$	$\sigma_{k_i} = 0.0001$ $\sigma_{k_r} = 0.0074$	0	3802
20 Obs. without Noise	θ_ϕ	$\mu_{k_i} = 0.0021$ $\mu_{k_r} = 0.0487$	$\sigma_{k_i} = 7.3 \times 10^{-5}$ $\sigma_{k_r} = 0.0020$	1	15587
	$\theta_{\neg\phi}$	$\mu_{k_i} = 0.0017$ $\mu_{k_r} = 0.1630$	$\sigma_{k_i} = 0.0001$ $\sigma_{k_r} = 0.0084$	0	5194

Table 3.1. List of different data scenarios and inference results. Data is produced given a combination of parameters that satisfy the property of interest, $\theta_\phi = (0.002, 0.05)$ do not satisfy the property of interest $\theta_{\neg\phi} = (0.002, 0.18)$, (with or without additive noise $\sigma = 2$). We integrate the corresponding posterior distribution to give us the probability in column 5 .

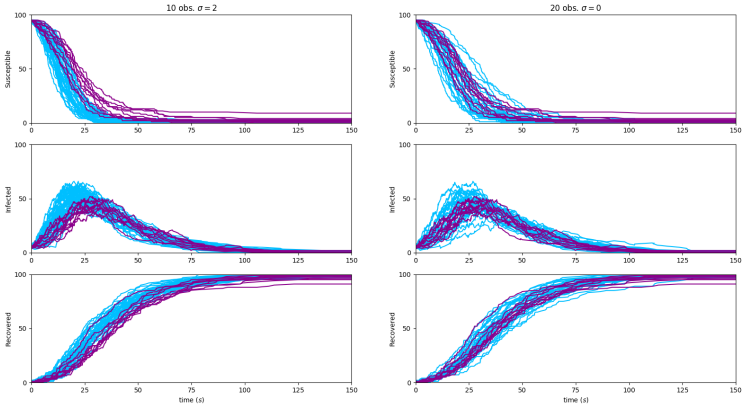


Figure 3.5. Purple traces represent 10 simulated traces from the true parameters θ_ϕ and 30 blue simulated traces are simulated from the mean of the inferred parameters $\tilde{\theta}$ for the two different data scenarios: left column represent traces simulated from parameters inferred with 10 noisy observations, $\sigma = 2$ and the right hand column is the case corresponding to 20 noiseless ($\sigma = 0$) observations.

Chapter 4

Simultaneous Inference and Verification

In the previous chapter we introduced the notion of Bayesian verification and applied the framework to a simple CRN case study. We worked with likelihood-free Bayesian inference techniques, namely approximate Bayesian computation, to infer a posterior distribution of the parameters given the data, then we integrated this posterior over the parameter space Θ_ϕ which is generated from a numeric-based parameter synthesis. Whilst this allowed us to formally verify properties on an underlying stochastic system, the major limitations of the Bayesian verification framework is the ability to synthesise parameter regions, Θ_ϕ, as the numerical method we utilised in the previous chapter is highly accurate but only feasible for small, finite state systems of interest. In this chapter we extend the Bayesian Verification framework introduced in the previous chapter to allow for a more scalable and widely applicable framework. The parameter synthesis technique used in the previous chapter cannot be applied in many practical applications, as this means of parameter synthesis typically doesn't scale to realistic models. To address this limitation, in this chapter we resort to statistical approaches (via SMC) for parameter synthesis, similar to [24], aiming to tackle the limitations of the Bayesian verification framework introduced in the previous chapter.

Specifically, we formally integrate an SMC technique into the algorithm that performs Bayesian inference. More precisely, we utilise the simulations needed in the ABC-SMC algorithm (Algorithm 4) to perform SMC, which yields the estimation of the probability of satisfying the property of interest, $\hat{\Lambda}_\phi(\theta)$. Whilst the ABC-SMC algorithm rejects and discards a number of proposed parameters, we propose to retain these samples, and their corresponding simulations, to later provide a classification (via support vector machines) of the parameter space. With these statistically-estimated parameter regions, we complete the Bayesian verification framework, as

per (3.14). The new framework (detailed in the next section and presented in Fig. 4.1), which employs models to extract information from the observation data y_{obs} or a summary statistic thereof s_{obs}, is likelihood-free and entirely based on simulations, which makes it applicable to stochastic models of different size and structure. We discuss now the specifics of extending the Bayesian verification framework, discussing in more detail the SMC algorithm of choice and how exactly we can utilise these SMC results to allow us to calculate (3.14).

4.1 Extending Bayesian Verification

We briefly summarise the problem of interest and how we improve on the initial Bayesian verification framework introduced in the previous chapter. Given a property of interest, ϕ, a class of parametrised stochastic models $\mathcal{M}_\theta$, and data from the underlying system, y_{obs}, we want to calculate the probability that the underlying data-generating system , denoted as $\mathcal{S}$, satisfies a given property of interest, given by the credibility calculation; $\mathcal{C} = P(\mathcal{S} \models \phi \mid y_{obs})$. Instead of using numerical techniques to calculate the parameter synthesis regions needed for this calculation, we resort to statistical methods. We propose to simultaneously infer parameters and perform model-based SMC, using the simulations required to infer the model parameters for both the inference and the verification aspect of the framework. With these SMC results, we use a supervised machine learning method to determine regions of the parameter space that relate to models verifying the given property of interest. With these approximate parameter synthesis regions, we are now able to integrate the generated posterior over these synthesised parameter regions, to quantify a probability (or credibility calculation) on whether or not the system satisfies the given property.

Whilst this Bayesian verification framework theoretically can be applied to any stochastic model of interest, our interest lies in working within the realms of biology and we apply this framework to CRNs [59, 116], where the CRN represents the data-generating biological system, $\mathcal{S}$, which can be modelled by parametrised continuous-time Markov Chains [66], denoted as $\mathcal{M}_\theta$, and we define properties, ϕ, of interest in CSL. As is the case in the previous chapter, we argue that the alternative use of CRN data for black-box SMC would be infeasible. We now present the SMC algorithm that is performed simultaneously with the Bayesian inference aspect of the algorithm, the Massart algorithm.

4.1.1 Statistical Model Checking with the Massart Algorithm

We recall that SMC techniques are used to estimate the validity of quantitative properties of probabilistic systems by simulating traces from an executable model of the system [74] - here we utilise Monte Carlo simulations to estimate the probability of properties of interest with a user-defined degree of accuracy (denoted below as ϵ). As mentioned earlier in the thesis, these can be obtained via standard concentration inequalities, such as the Chernoff [32] or the Okamoto [85] bounds. Recalling the SMC problem briefly, we wish to estimate a probability $\hat{\Lambda}_\phi(\theta)$ that approximates the unknown $\Lambda_\phi(\theta)$ within an absolute error ϵ and with a $(1-\delta)$ confidence lower bound, namely

$$P(|\hat{\Lambda}_\phi(\theta) - \Lambda_\phi(\theta)| > \epsilon) \leq \delta. \tag{4.1}$$

In this work, we leverage the sharper Massart bounds [78]: we use the Sequential Massart algorithm [63, 64] (described below), which progressively defines confidence intervals of the estimated probability and then applies the Massart bounds [78]. Massart bounds depend on the unknown probability $\Lambda_\phi(\theta)$ that we are estimating, which forces one to numerically evaluate with certainty an interval in which $\Lambda_\phi(\theta)$ evolves. Let us denote by $C(\Lambda_\phi(\theta), I)$ the coverage of $\Lambda_\phi(\theta)$ by a confidence interval I, i.e., the probability that $\Lambda_\phi(\theta) \in I$.

Theorem 4.1.1 (Absolute-Error Massart Bound with Coverage [64])**.** *Let $\hat{\Lambda}_\phi(\theta)$ be the probability estimated from n Monte Carlo simulations, ϵ be a given error, $\hat{\Lambda}^L_\phi(\theta)$ and $\hat{\Lambda}^U_\phi(\theta)$ be the lower and upper bounds of a confidence interval $I = [\hat{\Lambda}^L_\phi(\theta), \hat{\Lambda}^U_\phi(\theta)]$ and I^c be its complement within $[0,1]$. The Massart bound is defined as*

$$P(|\hat{\Lambda}_\phi(\theta) - \Lambda_\phi(\theta)| > \epsilon) \leq 2\exp\left(-n\epsilon^2 h_a(\Lambda_\phi(\theta), \epsilon)\right) + C(\Lambda_\phi(\theta), I^c), \tag{4.2}$$

$$\textit{where } h_a(\Lambda_\phi(\theta), \epsilon) = \begin{cases} \frac{9}{2}\frac{1}{\left(3\Lambda_\phi(\theta)+\epsilon\right)\left(3(1-\Lambda_\phi(\theta))-\epsilon\right)}, & \textit{if } 0 < \Lambda_\phi(\theta) < 1/2 \\ \frac{9}{2}\frac{1}{\left(3\left(1-\Lambda_\phi(\theta)\right)+\epsilon\right)\left(3\Lambda_\phi(\theta)+\epsilon\right)}, & \textit{if } 1/2 \leq \Lambda_\phi(\theta) < 1. \end{cases}$$

Notice that the above theorem requires the true satisfaction probability $\Lambda_\phi(\theta)$, which is not known. We can replace it with its estimate $\hat{\Lambda}_\phi(\theta)$, which can be conservatively set to $\hat{\Lambda}_\phi(\theta) = \hat{\Lambda}^U_\phi(\theta)$ if $\hat{\Lambda}^U_\phi(\theta) < 1/2$, $\hat{\Lambda}_\phi(\theta) = \hat{\Lambda}^L_\phi(\theta)$ if $\hat{\Lambda}^L_\phi(\theta) > 1/2$, and $\hat{\Lambda}_\phi(\theta) = 1/2$ if $1/2 \in I$. The following sample-size result follows:

Theorem 4.1.2 ([64])**.** *Let δ_c be a coverage parameter chosen such that $\delta_c < \delta$ and $C(\Lambda_\phi(\theta), I^c) < \delta_c$. Under the conditions of Theorem 4.1.1, a Monte Carlo algorithm that outputs an estimate $\hat{\Lambda}_\phi(\theta)$ fulfils the condition in* (4.1) *if $n > \lceil \frac{1}{h_a(\Lambda_\phi(\theta),\epsilon)\epsilon^2} \log \frac{2}{\delta-\delta_c} \rceil$.*

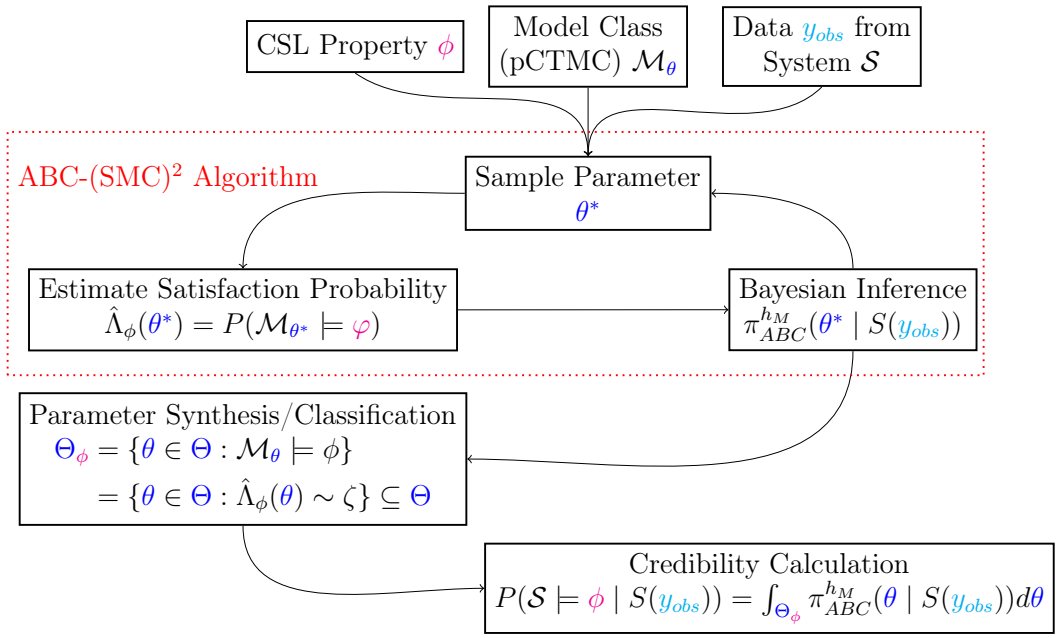


Figure 4.1. Bayesian verification via ABC-(SMC)^2.

The Sequential Massart Algorithm requires three inputs: an error parameter ϵ and two confidence parameters δ and δ_c. Initially, $\hat{\Lambda}^L_\phi(\theta) = 0$, $\hat{\Lambda}^U_\phi(\theta) = 1$, $C(\Lambda_\phi(\theta), [0,1]^c) = 0$, and $\hat{\Lambda}_\phi(\theta) = 1/2$, which results in the Okamoto-like bound with $h_a(1/2, \epsilon) \approx 2$ when $\epsilon \to 0$: the quantity $n_{\mathcal{O}} = \lceil \frac{1}{2\epsilon^2} \log \frac{2}{\delta} \rceil$ thus represents an upper-bound on the number of simulations required for the statistical guarantees. After each sampled trace, we update both a Monte Carlo estimator and a $(1-\delta_c)$-confidence interval for $\Lambda_\phi(\theta)$. The updated confidence interval is then used in the Massart function to compute an updated required sample size n satisfying Theorem 4.1.2. This process is repeated until the calculated sample size is lower than or equal to the current number of simulated traces.

4.2 Integrating formal verification and inference

We will now show how we address the scalability limitations of the work presented in the previous chapter, and specifically the parameter synthesis part: in the previous chapter the synthesis was calculated symbolically, which practically limited the applicability to CTMCs with small state spaces and a few parameters. We incorporate here SMC within the Bayesian inference framework and instead *estimate* parameter regions. We name the modified algorithm Approximate Bayesian Computation - Sequential Monte Carlo with Statistical Model Checking: ABC-(SMC)^2 and present the

idea behind the algorithm in Fig. 4.1.

In the ABC-SMC scheme (Algorithm 4), a total of B_m simulations are performed for each sampled parameter θ^{**}, whether the sample is retained or not towards the approximate posterior $\pi_{ABC}^{h_M}(\theta \mid s_{obs})$: this leads to a considerable amount of wasted computational effort. We propose instead to statistically model check each of the sampled parametrised models by means of the generated simulations, whilst parameter inference on the model is run (ABC-SMC); we shall use the outcome of the SMC algorithm for the Bayesian verification framework, by classifying the parameter synthesis regions using statistical approaches.

At any of the M iterations, for each sampled point $\theta^{**} \in \Theta$, we estimate the probability $\hat{\Lambda}_\phi(\theta^{**}) \approx \Lambda_\phi(\theta^{**})$, with statistical guarantees, that an instantiated model $\mathcal{M}_{\theta^{**}}$ satisfies a given property of interest ϕ, namely $P(\mathcal{M}_{\theta^{**}} \models \varphi) = \hat{\Lambda}_\phi(\theta^{**})$. We then proceed with the ABC-SMC algorithm as normal, calculating whether the sampled parameter θ^{**} contributes to the approximate posterior (acceptance) or not (rejection). The greatest advantage of integrating the inference and model checking schemes is that the framework provides both samples $\{\theta_{h_M}^{(i)}, w_{h_M}^{(i)}\}$, which allows one to construct an approximation of the posterior distribution $\pi_{ABC}^{h_M}(\theta \mid s_{obs})$, and satisfaction probability estimates, $\left\{\theta^{**}, \hat{\Lambda}_\phi\left(\theta^{**}\right), \hat{\Lambda}_\phi^L\left(\theta^{**}\right), \hat{\Lambda}_\phi^U\left(\theta^{**}\right)\right\}$ for all the sampled parameters θ^{**} (whether accepted or not). These values are later used to train a classifier to generate the parameter synthesis regions. We shall then integrate the approximate posterior over the classified parameter regions, to obtain a credibility calculation.

4.2.1 ABC-(SMC)^2 Algorithm

Recall that the output of the ABC-SMC algorithm is a set of samples $\theta_M^{(i)}$ with their corresponding weights $w_M^{(i)}$, which satisfy the following:

$$\{\theta_M^{(i)}, w_M^{(i)}\} \sim \pi_{ABC}^{h_M}(\theta \mid s_{obs}) \propto \int K_{h_M}\left(\|s - s_{obs}\|\right) p(y \mid \theta)\pi(\theta)dy, \tag{4.3}$$

where $i = 1, \ldots, N$ is the number of particles used to approximate the posterior. For each parameter θ^{**}, simulation data is generated from the model $y_b \sim p(y \mid \theta^{**})$ to calculate $s^b = S(y_b)$, for a total of B_m times, and this data is used to estimate $\hat{\Lambda}_\phi(\theta^{**}) \approx \Lambda_\phi(\theta^{**})$.

We utilise the sequential Massart algorithm [64] presented in the previous section for this SMC procedure. We replace the number of simulations for each sampled parameter, B_m, with the calculated minimum number of samples estimated in the sequential Massart algorithm [64], $B_m = n \leq n_{\mathcal{O}}$, to calculate an estimated probability $\hat{\Lambda}_\phi(\theta^{**})$ with accuracy and confidence. We sample θ^{**} a total of $\bar{N}$

times, whether or not these samples are accepted as samples from the posterior at any generation m. For these sampled parameters, $\theta^{(r)}$, $r = 1, \ldots, \bar{N}$, we estimate the corresponding mean estimated probabilities $\hat{\Lambda}_\phi\left(\theta^{(r)}\right)$ and $(1-\delta)$ uncertainty bounds: $\left\{\theta^{(r)}, \hat{\Lambda}_\phi\left(\theta^{(r)}\right), \hat{\Lambda}^L_\phi\left(\theta^{(r)}\right), \hat{\Lambda}^U_\phi\left(\theta^{(r)}\right)\right\}$. Here $\bar{N}$ depends on the acceptance rate of the sampled parameters $\theta^{(r)}$, where $\bar{N} \geq N \times M$, where N is the number of particles to sample and M is the total number of generations of the ABC-SMC scheme. From this new algorithm, we obtain a set of weighted parameter vectors from the final generation M, $\{\theta^{(i)}_M, w^{(i)}_M\} \sim \pi^{h_M}_{ABC}(\theta \mid s_{obs})$ where $i = 1, \ldots, N$, as well as $\bar{N}$ sampled parameters and their corresponding estimated probabilities $\left\{\theta^{(r)}, \hat{\Lambda}_\phi\left(\theta^{(r)}\right), \hat{\Lambda}^L_\phi\left(\theta^{(r)}\right), \hat{\Lambda}^U_\phi\left(\theta^{(r)}\right)\right\}_{r=1}^{\bar{N}}$.

We recall that by increasing the number of Monte Carlo samples in the ABC-SMC scheme, we increase the accuracy of the posterior distribution estimate, and that is indeed the case in this framework too. However, as we increase the number of parameter samples to accept, we must also remember that we increase the number of times we perform SMC for each sampled parameter. While this leads to a more accurate posterior distribution estimate and allows for a rigorous model verification across the parameter space, this leads to a higher computational cost.

When it comes to implementing the ABC-$(\text{SMC})^2$ framework on more complex models with a higher number of parameters, the algorithm has the same limitations as the ABC-SMC algorithm. While the ABC-SMC aspect of the framework provides one with a global parameter sensitivity analysis [108], as we increase the model complexity - and thus the number of parameters - the probability of accepting the perturbed or sampled parameter θ^{**} decreases and we thus need to explore alternative and more efficient sampling schemes.

We present the ABC-$(\text{SMC})^2$ scheme in Algorithm 5, with the $MASSART$ function corresponding to the Absolute-Error Massart Algorithm presented in Appendix A.1. The ABC-$(\text{SMC})^2$ algorithm takes as inputs a property of interest, ϕ, a prior probability distribution $\pi(\theta)$ an absolute-error tolerance ϵ as well as a coverage parameter δ_c and confidence value δ. The estimated probabilities $\left\{\theta^{(r)}, \hat{\Lambda}_\phi(\theta)^{(r)}, \hat{\Lambda}^L_\phi(\theta)^{(r)}, \hat{\Lambda}^U_\phi(\theta)^{(r)}\right\}$, will be utilised for approximate parameter synthesis, which is discussed in the next section.

4.2.2 Approximate Parameter Synthesis via Statistical MC

The aim of parameter synthesis is to partition the parameter space Θ according to the satisfaction of the CSL property ϕ. Unlike the PMC-based synthesis in [82]

Algorithm 5 ABC-(SMC)^2

Input:

- CSL specification ϕ
- Prior distribution $\pi(\theta)$ and data generating function $p(y \mid \theta)$
- A kernel function $K_h(u)$ and scale parameter $h > 0$ where $u = \|y - y_{obs}\|$ (in this work, $K_h(d(y, y_{obs}) = \mathbb{1}_{\Omega_h}(y) = \mathbb{I}(y \in \Omega_h)$.
- $N > 0$, number of particles used to estimate posterior distributions
- Sequence of perturbation kernels $F_m(\theta \mid \theta^*)$, $m = 1, \ldots, M$
- A quantile $\lambda \in [0, 1]$ to control the rate of decrease of thresholds h_m
- Summary statistic function $s = S(y)$
- Parameters for statistical MC: absolute-error value ϵ, confidence δ, coverage δ_c

Output:

- Set of weighted parameter vectors $\left\{\theta_M^{(i)}, w_M^{(i)}\right\}_{i=1}^N$ drawn from $\pi_{ABC}(\theta \mid s_{obs}) \propto \int K_{h_M}(\|s - s_{obs}\|)p(s \mid \theta)\pi(\theta)ds$
- Set of parameters with corresponding estimated mean, $\hat{\Lambda}_\phi\left(\theta^{(r)}\right)$ and $(1-\delta)$ confidence interval $\left[\hat{\Lambda}_\phi^L\left(\theta^{(r)}\right), \hat{\Lambda}_\phi^U\left(\theta^{(r)}\right)\right]$ of estimated probability to satisfy ϕ, $P\left(\mathcal{M}_{\theta^{(r)}} \models \varphi\right) = \hat{\Lambda}_\phi\left(\theta^{(r)}\right)$: $\left\{\theta^{(r)}, \hat{\Lambda}_\phi\left(\theta^{(r)}\right), \hat{\Lambda}_\phi^L\left(\theta^{(r)}\right), \hat{\Lambda}_\phi^U\left(\theta^{(r)}\right)\right\}$

1: Set $r = 0$
2: **for** $m = 0, \ldots, M$: **do**
3: **for** $i = 0, \ldots, N$: **do**
4: **if** $m = 0$ **then**
5: Sample $\theta^{**} \sim \pi(\theta)$
6: **else**
7: Sample θ^* from the previous population $\{\theta_{m-1}^{(i)}\}$ with weights $\{w_{m-1}^{(i)}\}$ and perturb the particle to obtain $\theta^{**} \sim F_m(\theta \mid \theta^*)$
8: **end if**.
9: **if** $\pi(\theta^{**}) = 0$ **then**
10: **goto** line 4
11: **end if**
12: Calculate $\left(\left\{\hat{\Lambda}_\phi\left(\theta^{**}\right), \left[\hat{\Lambda}_\phi^L\left(\theta^{**}\right), \hat{\Lambda}_\phi^U\left(\theta^{**}\right)\right\}, B_m, \sum_{b=1}^{B_m} K_{h_m}(\|s^b - s_{obs}\|), \bar{d}\right)$ from the modified Massart Algorithm: $MASSART\left(\epsilon, \delta, \delta_c, h_m, \theta^{**}, s_{obs}\right)$
13: Calculate $b_m(\theta^{**}) = \frac{1}{B_m}\sum_{b=1}^{B_m} K_{h_m}(\|s^b - s_{obs}\|)$
14: Set $\left(\theta^{(r)}, \hat{\Lambda}_\phi\left(\theta^{(r)}\right), \hat{\Lambda}_\phi^L\left(\theta^{(r)}\right), \hat{\Lambda}_\phi^U\left(\theta^{(r)}\right)\right) = \left(\theta^{**}, \hat{\Lambda}_\phi(\theta^{**}), \hat{\Lambda}_\phi^L(\theta^{**}), \hat{\Lambda}_\phi^U(\theta^{**})\right)$
15: $r \leftarrow r + 1$
16: **if** $b_m(\theta^{**}) = 0$ **then**
17: **goto** line 4
18: **end if**
19: Set $\theta_m^{(i)} = \theta^{**}$, $\bar{d}_m^{(i)} = \bar{d} = \frac{1}{B_m}\sum_{b=1}^{B_m}\|s^b - s_{obs}\|$ and calculate

20:
$$w_m^{(i)} = \begin{cases} b_m\left(\theta_m^{(i)}\right), & \text{if } m = 0 \\ \dfrac{\pi\left(\theta_m^{(i)}\right) b_m\left(\theta_m^{(i)}\right)}{\sum_{j=1}^N w_{m-1}^{(j)} F_m\left(\theta_m^{(i)} \mid \theta_{m-1}^{(j)}\right)}, & \text{if } m > 0 \end{cases}$$

21: **end for**
22: Normalise weights: $w_m^{(i)} \leftarrow w_m^{(i)} / \left(\sum_{i=1}^N w_m^{(i)}\right)$
23: Set $h_{m+1} = (\lambda/N)\sum_{i=1}^N \bar{d}_m^{(i)}$
24: **end for**
25: **return** $\left\{\left(\theta_M^{(i)}, w_M^{(i)}\right)\right\}_{i=1}^N, \left\{\theta^{(r)}, \hat{\Lambda}_\phi\left(\theta^{(r)}\right), \hat{\Lambda}_\phi^L\left(\theta^{(r)}\right), \hat{\Lambda}_\phi^U\left(\theta^{(r)}\right)\right\}_{r=1}^{\bar{N}}$

(recalled in Sec. 3.2), we utilise a statistical approach to classify the parameter space, akin to [24]. So instead of employing the true satisfaction probability $\Lambda_\phi(\theta) \sim \zeta$ (where ζ is the probability bound contained in formula ϕ) to determine Θ_ϕ (and its complement), we use $\hat{\Lambda}_\phi(\theta^{(r)})$, a statistical approximation computed at each sampled parameter point $\theta^{(r)}$. Evidently, recalling the confidence parameter δ, we should compute $\hat{\Lambda}_\phi(\theta^{(r)}) \sim \zeta \pm \epsilon$ (where the sign $\pm$ depends on the direction of the inequality $\sim$).

In practice, we use the estimated lower $\hat{\Lambda}^L_\phi(\theta^{(r)})$ and upper bounds $\hat{\Lambda}^U_\phi(\theta^{(r)})$, such that $\Lambda_\phi(\theta^{(r)}) \in \left[\hat{\Lambda}^L_\phi(\theta^{(r)}), \hat{\Lambda}^U_\phi(\theta^{(r)})\right]$, to partition the parameter space as:

- $\Theta_\phi = \{\theta \in \Theta : \hat{\Lambda}^L_\phi(\theta) > \zeta\}$,
- $\Theta_{\neg\phi} = \{\theta \in \Theta : \hat{\Lambda}^U_\phi(\theta) < \zeta\}$,
- $\Theta_{\mathcal{U}} = \Theta \backslash (\Theta_\phi \cup \Theta_{\neg\phi})$.

Notice that these formulas are a function of $\theta \in \Theta$. Since in the ABC-(SMC)^2 procedure we generate a finite number of parameter samples $\theta^{(r)}$, which are biased towards the sought posterior distribution, there might be areas of the parameter space Θ that are insufficiently covered. We thus resort to supervised learning techniques to classify parameter synthesis regions: we utilise support vector machines (SVMs) [34, 102] as a classification technique. Support vector machines are a simple yet effective supervised learning technique as they are effective in high-dimensional spaces and only require a subset of training points in the decision function. Given a set of training examples, each classified by the user, the SVM model assigns unseen data to one category or the other, making it a non-probabilistic binary linear classifier. The SVM constructs a hyper-plane or set of hyper-planes in a higher dimensional space that is trained to maximise the largest distance between the two (or more) classes - the larger this margin is, the lower the generalization error of the classifier. Given the relatively low number of data points required to train SVMs and the high number of simulations required to perform SMC that generate the data points we wish to classify, we make use of SVMs as our classification technique - alternative classifiers such as Stochastic Gradient Descent [83] make for better alternatives if we work with much larger datasets. We train the SVM classifier on the data produced from the ABC-(SMC)^2 algorithm, namely on the set $\{\theta^{(r)}, \hat{\Lambda}_\phi(\theta^{(r)}), \hat{\Lambda}^L_\phi(\theta^{(r)}), \hat{\Lambda}^U_\phi(\theta^{(r)})\}$ where $r = 1, \ldots, \bar{N}$. The SVM which is trained on this data then provides a non-linear classifying function, $\xi_\phi(\theta)$, where $\xi_\phi(\theta) = 1$ if $\theta \in \Theta_\phi$, $\xi_\phi(\theta) = -1$ if $\theta \in \Theta_{\neg\phi}$ and $\xi_\phi(\theta) = 0$ if $\theta \in \Theta_{\mathcal{U}}$. With this classifying function, we can then generate parameter

synthesis regions by sampling parameters across the whole parameter space $\theta \in \Theta$, and then classify the parameters according to this function.

Theoretical results on ABC-(SMC)^2

We present a rationale for integrating the SMC algorithm and Bayesian inference, and argue that the ABC-(SMC)^2 algorithm converges asymptotically to the quantity $\mathcal{C} = P(\mathcal{S} \models \phi \mid s_{obs})$. Recall that N denotes the total number of accepted Monte Carlo samples, n is the number of simulations required for SMC estimates, with absolute error ϵ and confidence parameter δ leading to $(1-\delta)$-confidence intervals and h_m is the distance threshold for an ABC posterior $\pi^{h_m}_{ABC}(\theta \mid s_{obs})$. We let the credibility calculation that we compute from the ABC-(SMC)^2 algorithm be denoted as $\mathcal{C}_{m,N,n}$.

Theorem 4.2.1 (ABC-(SMC)^2 Convergence)**.** *$\mathcal{C}_{m,N,n}$ converges in probability to $\mathcal{C}$ as the quantities $h_m \to 0$ $(m \to \infty)$ $N \to \infty$ and $n \to \infty$, namely:*

$$P(|C_{m,N,n} - \mathcal{C}| > \epsilon_{tot}) \to 0, \ \forall \epsilon_{tot} > 0. \tag{4.4}$$

Proof. To prove convergence, we work from the credibility calculation in Equation (3.14), applying the ABC approximation that converges to the true posterior [101]; the approximation scheme via Monte Carlo integration that converges in probability to the true integral [36]; and finally the SMC approximation to the satisfaction probability, given by $\hat{\Lambda}_\phi(\theta)$, which converges in probability to the true value $\Lambda_\phi(\theta)$. Starting from the credibility calculation, we note that

$$\mathcal{C} = \int_{\Theta_\phi} \pi(\theta \mid s_{obs}) \, \mathrm{d}\theta = \int_{\Theta} \mathbb{1}_{\Theta_\phi}(\theta) \pi(\theta \mid s_{obs}) \, \mathrm{d}\theta = \mathbb{E}_\pi[\mathbb{1}_{\Theta_\phi}(\theta)],$$

where $\mathbb{1}_{\Theta_\phi}(\theta) = 1$ if $\theta \in \Theta_\phi$, and 0 otherwise. Given that we work with likelihood-free methods, namely ABC, we see

$$\begin{aligned} \mathcal{C} &= \int_{\Theta} \mathbb{1}_{\Theta_\phi}(\theta) \pi(\theta \mid s_{obs}) \, \mathrm{d}\theta \\ &= \lim_{m \to \infty} \int_{\Theta} \mathbb{1}_{\Theta_\phi}(\theta) \pi^{h_m}_{ABC}(\theta \mid s_{obs}) \, \mathrm{d}\theta \quad \text{(ABC approximation)}, \end{aligned}$$

where we have used the ABC approximation to the true posterior, specifically $\pi(\theta \mid s_{obs}) = \lim_{m\to\infty} \pi^{h_m}_{ABC}(\theta \mid s_{obs}) = \lim_{h_m \to 0^+} \pi^{h_m}_{ABC}(\theta \mid s_{obs})$ [101]. We then apply the Monte Carlo approximation, taking the iid samples to be samples from the ABC

posterior, $\pi_{ABC}(\theta \mid s_{obs})$,

$$\begin{aligned}\mathcal{C} &= \lim_{m\to\infty} \int_{\Theta} \mathbb{1}_{\Theta_\phi}(\theta) \lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} \delta_{\theta^{(i)}}(\theta) \text{ (Monte Carlo approximation)} \\ &= \lim_{m\to\infty} \left[\lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}_{\Theta_\phi}(\theta^{(i)}) \right],\end{aligned}$$

where $\theta^{(i)}$ are iid samples from the ABC posterior, $\theta^{(i)} \sim \pi_{ABC}^{h_m}(\theta \mid s_{obs})$, $\delta_{\theta^{(i)}}(\theta)$ is the Dirac delta measure, where $\delta_x(A) = \mathbb{1}_A(x) = 1$ if $x \in A$, and $\delta_x(A) = \mathbb{1}_A(x) = 0$ otherwise, and we denote the Monte Carlo approximation to the integral $\mathcal{I} = \int_\Theta f(\theta)$ as $\mathcal{I}_n = (1/n)\sum_{i=1}^{N} f(\theta^{(i)})$, where it can be shown [36] that $\mathcal{I}_n \to \mathcal{I}$ in probability, $P(|I_n - I| > \epsilon_{MC}) \to 0$ $\forall \epsilon_{MC} > 0$ as $N \to \infty$. The difficulty now is determining whether the sampled parameters $\theta^{(i)}$ lie within the set of parameters that satisfy the property, $\theta^{(i)} \in \Theta_\phi$. We note that Θ_ϕ is defined as $\Theta_\phi = \{\theta \in \Theta : \Lambda_\phi(\theta) \sim \zeta\}$, where ζ is the probability bound contained in a formula ϕ which allows us to present the function $\mathbb{1}_{\Theta_\phi}(\theta)$ as

$$\mathbb{1}_{\Theta_\phi}(\theta) = \mathbb{1}_{\Lambda_\phi(\theta)\sim\zeta}(\theta) = \begin{cases} 1, & \Lambda_\phi(\theta) \sim \zeta, \\ 0, & \text{otherwise}, \end{cases} \tag{4.5}$$

which results in

$$\mathcal{C} = \lim_{m\to\infty} \left[\lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}_{\Lambda_\phi(\theta)\sim\zeta}(\theta^{(i)}) \right].$$

However, we statistically estimate the satisfaction probability $\Lambda_\phi(\theta)$. Let V be a Bernoulli random variable with corresponding mean parameter $\Lambda_\phi(\theta^{(i)})$, that is, $v \sim \text{Bern}(\Lambda_\phi(\theta^{(i)}))$. Then $\Lambda_\phi(\theta)$ can be estimated using n sample draws from this distribution, $\hat{\Lambda}_\phi(\theta^{(i)}) \approx (1/n)\sum_{i=1}^{n} v_i$. This estimate has been shown [3, 64, 120] to converge in probability to the true mean $\Lambda_\phi(\theta)$, that is, $P(|\hat{\Lambda}_\phi(\theta) - \Lambda_\phi(\theta)| > \epsilon_{SMC}) \to 0$, $\forall \epsilon_{SMC} > 0$ as $n \to \infty$. Thus, taking a further limit $n \to \infty$, this leads us to

$$\mathcal{C} = \lim_{m\to\infty} \left[\lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} \left[\lim_{n\to\infty} \mathbb{1}_{\hat{\Lambda}_\phi(\theta)\sim\zeta}(\theta^{(i)}) \right] \right] \approx \mathcal{C}_{m,N,n}, \tag{4.6}$$

where $\mathcal{C}_{m,N,n} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}_{\hat{\Lambda}_\phi(\theta)\sim\zeta}(\theta^{(i)})$ and $\theta^{(i)}$ are iid draws from the ABC posterior, $\theta^{(i)} \sim \pi_{ABC}^{h_m}(\theta \mid s_{obs})$. Thus by taking the limits $\lim_{m,N,n\to\infty} \mathcal{C}_{m,N,n} = \mathcal{C}$.

□

We have shown that we can get an estimate $\mathcal{C}_{m,N,n}$ to the true credibility calculation $\mathcal{C}$ by generating simulations alone, where the simulations are used to infer

parameters θ and perform statistical model checking. We now turn our efforts to a case study of ABC-(SMC)^2, where we deploy the Massart algorithm [64] for our SMC technique.

4.3 Application of the ABC-(SMC)^2 Algorithm

Experimental Setup

All experiments have been run on an Intel(R) Xeon(R) CPU E5-1660 v3 @ 3.00GHz, 16 cores with 16GB memory. ABC-(SMC)^2 is coded in C++, while Python is used for the SVM classifier.

SIR System and Parametrised Stochastic Model

For a direct comparison to the parameter synthesis technique and our work in the previous chapter, once again we consider the SIR case study, which we recap here. The SIR model describes the dynamics of three epidemic types, a susceptible group (S), an infected group (I), and a recovered group of individuals (R) - here we let S, I and R evolve via the rules

$$S + I \xrightarrow{k_i} 2I, \quad I \xrightarrow{k_r} R.$$

This CRN is governed by the rate parameters $\theta = (k_i, k_r)$, and each state of the pCTMC describes the combination of the number of each type (S, I, R) (equating to molecule/species counts in CRNs), where the state vector is given by $X(t) = (S(t), I(t), R(t))$, which denotes the number of molecules of each species at a given time t. The initial state of the pCTMC is once again $s_0 = X(0) = (S_0, I_0, R_0) = (95, 5, 0)$. We will also verify the same property, $\phi = P_{>0.1}((I > 0)U^{[100,150]}(I = 0))$, i.e. whether, with a probability greater than 0.1, the infection dies out within a time interval between $t = 100$ and $t = 150$ seconds. We confine our parameters to the set $\Theta = [k_i^\perp, k_i^\top] \times [k_r^\perp, k_r^\top] = [5 \times 10^{-5}, 0.003] \times [0.005, 0.2]$. We generate observation data from the SIR model with three different parameter choices, corresponding to the CTMCs $\mathcal{M}_{\theta_\phi}$, $\mathcal{M}_{\theta_{\neg\phi}}$ and $\mathcal{M}_{\theta_\mathcal{U}}$, where $\theta_\phi = (0.002, 0.075)$, $\theta_{\neg\phi} = (0.001, 0.15)$ and $\theta_\mathcal{U} = (0.002, 0.125)$. From Figure 3.3 in the previous chapter, we see that $\theta_\phi \in \Theta_\phi$, $\theta_{\neg\phi} \in \Theta_{\neg\phi}$, and finally $\theta_\mathcal{U}$ is near the borderline. These models will correspond to three "true" underlying stochastic systems $\mathcal{S}$, with associated observation data. For each instance, we work with observed data y_{obs} that is sampled at a finite number of time steps. The observed data consists of only 5 simulated traces, observed at 10 time points. The summary statistics $S(y_{obs}) = s_{obs}$ is the average of the five traces.

It is worth emphasising that with so few observation traces, black-box SMC (directly based on observation traces, not on model-generated simulations) would be hopeless.

Our algorithm outputs samples from the approximated posterior and their corresponding weights, $\left\{\theta_M^{(i)}, w_M^{(i)}\right\} \sim \pi_{ABC}^{h_M}(\theta \mid s_{obs})$ where $i = 1, \ldots, N$. By the strong law of large numbers, letting $\hat{\theta}_M = \sum_{i=1}^{N} \theta_M^{(i)} w_M^{(i)}$, $P\left(\lim_{N\to\infty} \sum_{i=1}^{N} w_M^{(i)} \theta_M^{(i)} - \mathbb{E}[\hat{\theta}_M] = 0\right) = 1$. Thus we assume that the approximated posterior can be modelled by a multivariate Normal distribution, $\pi_{ABC}^{h_M}(\theta \mid s_{obs}) \approx \mathcal{N}(\hat{\theta}_M, \Sigma_M)$, where the mean is given by $\hat{\theta}_M$ and the elements of the empirical covariance matrix are defined as

$$\Sigma^M{}_{jk} = \frac{1}{1 - \sum_{i=1}^{N} (w_M^{(i)})^2} \sum_{i=1}^{N} w_M^{(i)} \left(\theta_M^{(i)} - \hat{\theta}_M\right)_j \left(\theta_M^{(i)} - \hat{\theta}_M\right)_k.$$

We choose the number of samples to be $N = 500$; the number of sequential steps to be $M = 20$; the kernel function $K_h(d(y, y_{obs})) = \mathbb{1}_{\Omega_h}(y)$ to be a simple indicator function, i.e. $K_h(d(y, y_{obs})) = 1$ if $d(y, y_{obs}) = \|y - y_{obs}\| < h$, $K_h(d(y, y_{obs})) = 0$ otherwise; the rate at which the thresholds h_m decrease to be $\lambda = 0.5$; and the summary statistic $s = S(y)$ is chosen to be the sample mean of the simulations and of the observations. We choose $\pi(\theta)$ to be a uniform prior over Θ, which we define as

$$\pi(\theta) = \begin{cases} \left(\Pi_{i=1}^{N_R} (\theta_i^{\top} - \theta_i^{\perp})\right)^{-1}, & \text{if } \theta \in \Theta = [\theta_1^{\perp}, \theta_2^{\top}] \times \cdots \times [\theta_{N_R}^{\perp}, \theta_{N_R}^{\top}] \\ 0, & \text{Otherwise.} \end{cases} \tag{4.7}$$

The perturbation kernel $F_m(\theta^{**} \mid \theta^*)$ is chosen to be a multivariate Normal distribution, so that $\theta^{**} \sim \mathcal{N}(\theta^*, 2\Sigma_{m-1})$, where the covariance is twice the second moment computed over the accepted weights and particles at step $m-1$, namely $\left\{\theta_{m-1}^{(i)}, w_{m-1}^{(i)}\right\}$, where $i = 1, \ldots, N$. For further details on alternative choices for threshold sequences, summary statistics and perturbation kernels, see [18, 35, 39, 92, 101].

Parameter synthesis: A Motivating Comparison

For the SMC component of the algorithm, we select the parameters $(\epsilon, \delta, \delta_c) = (0.01, 0.05, 0.001)$, which results in a maximum number of necessary simulations that equals $B_m \leq n_{\mathcal{O}} = \lceil \frac{1}{2\epsilon^2} \log \frac{2}{\delta} \rceil = 18445$. At the conclusion of the ABC-(SMC)^2 algorithm, we train the classifier over half of the sampled parameters (denoted by $\theta^{(r)}$, whether eventually accepted or rejected), with the corresponding estimated probabilities and test it on the other half, which results in the SVM classifier accuracy presented in Table A.1, which can be seen in the Appendix. We now briefly present

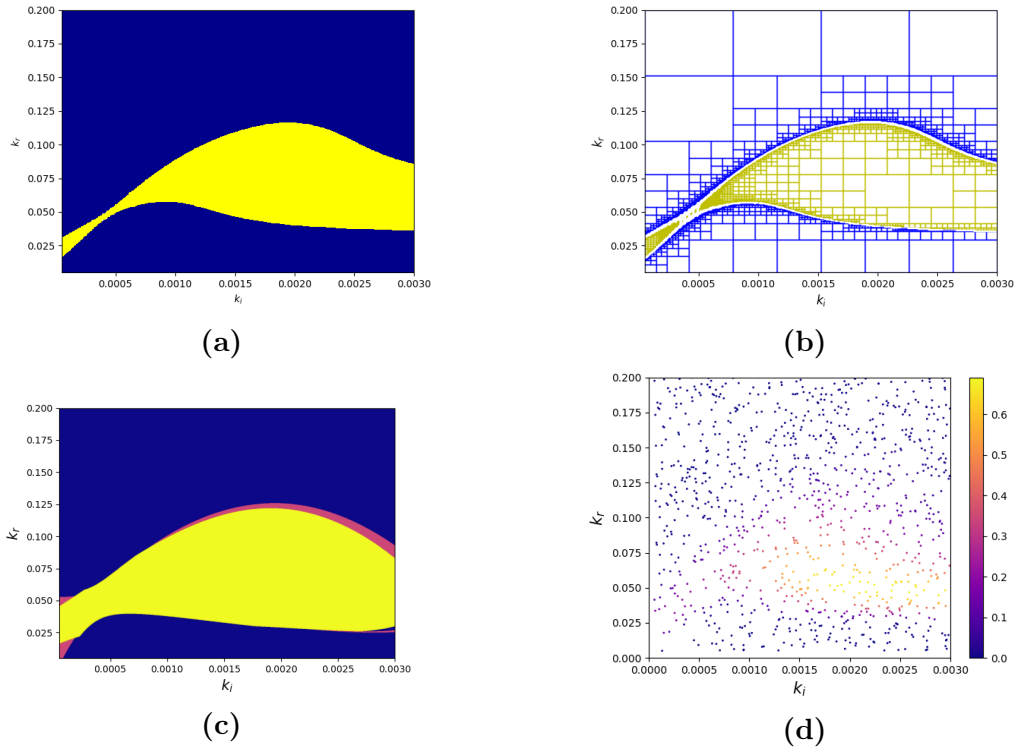


Figure 4.2. The set Θ_ϕ, is shown in yellow (lighter colour), meanwhile $\Theta_{\neg\phi}$, is shown in blue (darker colour) $\Theta_{\neg\phi}$. The undecided areas, $\Theta_{\mathcal{U}}$ (if any) are shown in magenta.
(4.2a) Parameter regions synthesised by GPU-Accelerated PRISM [31]. (4.2b) Gridding scheme.
(4.2c) Parameter regions from SVM classification using 1000 samples from a uniform distribution. (4.2d) Estimated probabilities $\Lambda_\phi(\theta^*)$.

the approximate parameter synthesis generated from the ABC-$(\text{SMC})^2$ approach and compare that to the numerical approach of [30, 31], which we present in Figure 4.2.

The PRISM-based parameter synthesis technique dissects the parameter space into 14413 grid regions (Figure 4.2b), which results in calculating the satisfaction probability at 57652 points. Instead, to illustrate how we can generate an approximate parameter synthesis using SMC, we consider sampling 1000 points from the prior $\pi(\theta)$, which is a Uniform distribution across the parameter space Θ. We run the Massart algorithm at each point to obtain an estimated probability with corresponding $(1-\delta)$ confidence bounds, where $\delta = 0.05$. With these samples and probabilities, we classify parameter regions with an SVM, which results in Figure (4.2c), with corresponding estimated probabilities in Figure 4.2d. The runtimes presented in Table 4.1 suggest that we obtain a good approximation of the parameter synthesis region in half the

Table 4.1. Parameter synthesis runtimes.

Parameter synth	Times [seconds]
PRISM-GPU	3096
SVM & SMC	1653.8

time of the GPU-accelerated PRISM tool, which could be further improved if we parallelised the computation [64]. These considerations have led us to embed the statistical parameter synthesis in the parameter inference algorithm.

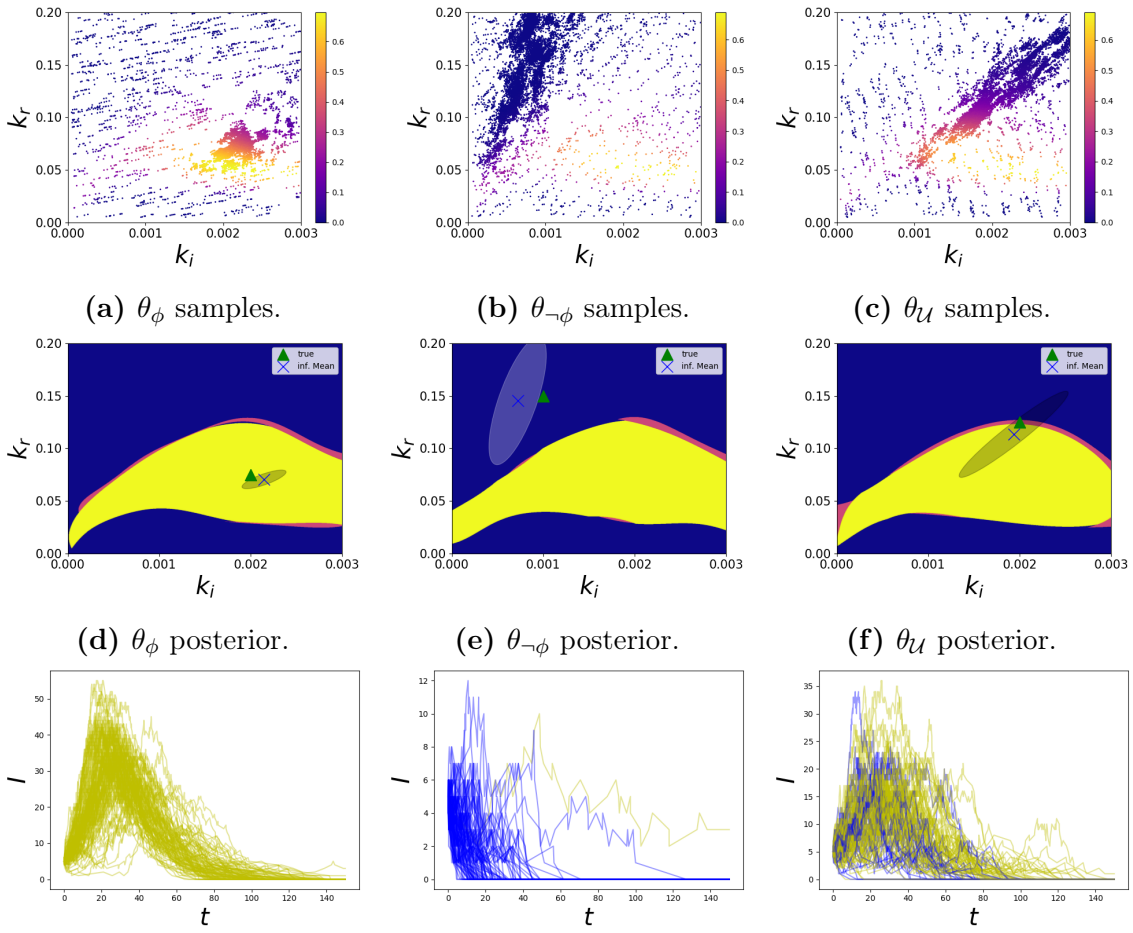
Outcomes of ABC-(SMC)^2 Algorithm

For the three case studies, the inferred posterior mean $\hat{\theta}_M$, covariance Σ_M, total number of sampled parameters ($\theta^{(r)}$, $r = 1, \ldots, R$) and resulting credibility calculation are given in Table A.1, with corresponding runtimes in Table A.2 (Appendix A.2). Figures 4.3d, 4.3e and 4.3f plot the inferred posterior, showing the mean (denoted by $\times$) and two standard deviations from the mean (corresponding ellipse around the mean), as well as the true parameter value ($\triangle$). In Case θ_ϕ, we can assert, with a parameter synthesis based off a confidence of $(1 - \delta) = 0.95$ and absolute-error $\epsilon = 0.01$, that the underlying stochastic system $\mathcal{S}$ does indeed satisfy the property of interest, as the credibility calculation gives $P(\mathcal{S} \models \phi \mid s_{obs}) = 1$. Case $\theta_{\neg\phi}$ has a low probability of satisfying the property of interest ($P(\mathcal{S} \models \phi \mid s_{obs}) = 0.0054$), whereas for Case $\theta_{\mathcal{U}}$ the inferred mean converges to the true mean that we would expect the estimated probability of satisfying the property to converge to, which is 0.5.

Table 1, and Figure 4.4 suggest that simulation times are largely dependent on the estimated probabilities, $\hat{\Lambda}_\phi(\theta)$: the closer the estimated probabilities are to 0.5, the larger the number of simulations required. To improve the runtime of Case $\theta_{\mathcal{U}}$, we would need to reduce variance and improve the accuracy of the inferred parameters, for instance by increasing the number of observed data points y_{obs} or with an alternative choice of either the summary statistics chosen or of the perturbation kernels [39].

4.4 Discussion of ABC-(SMC)^2

The ABC-(SMC)^2 framework allows the Bayesian verification framework of [82] to be applied to a wide variety of models. In ABC-(SMC)^2 we have newly utilised the simulations needed for the likelihood-free inference (also present in [82]) for statistical model checking of properties of interest and used the outputs of this procedure to allow



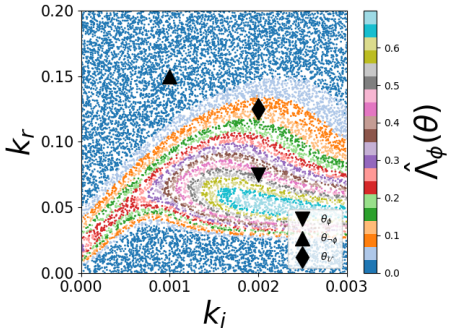
(a) θ_ϕ samples. **(b)** $\theta_{\neg\phi}$ samples. **(c)** $\theta_{\mathcal{U}}$ samples.

(d) θ_ϕ posterior. **(e)** $\theta_{\neg\phi}$ posterior. **(f)** $\theta_{\mathcal{U}}$ posterior.

(g) Samples traces from θ_ϕ posterior. **(h)** Samples traces from $\theta_{\neg\phi}$ posterior. **(i)** Samples traces from $\theta_{\mathcal{U}}$ posterior.

Figure 4.3. Bayesian verification results from ABC-(SMC)^2 for Case θ_ϕ (4.3a and 4.3d), Case $\theta_{\neg\phi}$(4.3b and 4.3e), and Case $\theta_{\mathcal{U}}$ (4.3c and 4.3f). Sampled points θ with estimated probabilities $\hat{\Lambda}_\phi(\theta)$ (4.3a, 4.3b and 4.3c). Inferred posterior $\pi^{h_M}_{ABC}(\theta \mid s_{obs})$ and parameter regions (4.3d, 4.3e and 4.3f). Traces of I molecules simulated from $\mathcal{M}_\theta$, with θ sampled from the θ_ϕ posterior (4.3g), the $\theta_{\neg\phi}$ posterior (4.3h) and the $\theta_{\mathcal{U}}$ posterior (4.3i). The set Θ_ϕ, is shown in yellow, whereas $\Theta_{\neg\phi}$ is shown in blue. The undecided areas $\Theta_{\mathcal{U}}$ is shown in magenta. For Fig (4.3g-4.3i) the traces colour represent which set the sampled parameters are members of.

for approximate parameter synthesis. The ABC-(SMC)^2 framework presented here hinges largely on the ABC-SMC scheme of [108] and is thus bound to its limitations: theoretically, there is nothing stopping the framework to be considered for models with a higher number of latent variables, but we would expect a higher runtime due to more proposals $F_m(\theta^{**} \mid \theta^*)$ being rejected due to the higher dimensionality



Case	$\hat{\Lambda}_\phi(\theta)$	Total simulations
θ_ϕ	0.47254	18445
$\theta_{\neg\phi}$	0.00408719	2202
$\theta_{\mathcal{U}}$	0.100433	14775

Figure 4.4 & Table 4.2. True parameter values with corresponding estimated probabilities using SMC (15000 uniform samples), and number of SMC simulations used in ABC-(SMC)^2 .

and would thus need to employ alternative proposal distributions to those considered here. Another limitation of the framework is the dependence on a possibly large number of simulations for parameter synthesis (the SMC part), which however it is a strong alternative to the parameter synthesis technique of [31] that leverages GPU acceleration. In its current state, the overall ABC-(SMC)^2 scheme can easily be parallelised over its components to drastically speed up the framework. Each aspect of the scheme - the CRN simulations [118], the inference scheme [62] and the SMC scheme [64] - naturally allow for parallelisation that can drastically speed up results. Although parallelisation allows for a large workload to be split up and run concurrently, our interests lie in reducing the overall workload that would need to be split up and parallelised. No matter how the simulations are performed, either sequentially or in parallel, the ABC-(SMC)^2 framework leads to a large number of simulations and thus a long running time and huge computational burden.

Whilst running this number of simulations allows for a high accuracy in both inference and verification, the number of simulations required to attain this high accuracy is restrictive, and we argue that we are able to perform Bayesian verification with a fewer number of simulations. In the next chapter, we seek to reduce the workload with the introduction of techniques that aim to:

1. reduce the total number of simulations required for both inference and statistical model checking;

2. utilise a technique to reduce the to infer parameters and formally verify parametrised stochastic models.

To reduce the total number of simulations required for Bayesian verification and address the current limitations of ABC-(SMC)^2, we consider an alternative Bayesian SMC technique and investigate further how and when statistical model checking should be considered and applied. In the current framework, if B_m, the number of simulations required for each sampled point, is large, then the estimate of $\pi_{ABC}(\theta \mid s_{obs})$ will be more accurate and would naturally allow for statistical model checking. However, by setting B_m to a small value or even $B_m = 1$, the acceptance probability is cheaper to evaluate [19] but can become highly variable, allowing for more time to explore the parameter space rather than obtaining more detailed information for a single sampled parameter that may be rejected. The trade-off here of course, is whilst the ABC-(SMC)^2 framework in its current state allows for a rigorous Bayesian verification technique across the whole parameter space, for the credibility calculation, we are only really interested in the posterior distribution, $\pi^{h_M}_{ABC}(\theta \mid s_{obs})$, and the satisfaction probabilities associated with this posterior probability distribution. In addition to this, by setting B_m to the minimum number of simulations required for SMC, in a worst case scenario this would lead to a total of $\bar{N} \times n_{\mathcal{O}}$ simulations where $\bar{N} \geq N \times M$ is the total number of sampled parameters, N is the number of accepted particles and M is the total number of generations or epochs in the sequential Monte Carlo scheme. Since we are no longer performing SMC naively on every sampled parameter, we explore different stages of the framework to perform SMC and highlight the benefits and drawbacks of performing an extensive statistical model checking procedure at a given point of the framework. As a result, instead of working with a summary statistic of the simulations, $s = S(y)$, we will be working with the simulations themselves, y.

Whilst these proposals drastically reduce the total number of simulations required, we look at further mitigating the computational burden of requiring a high number of simulations by deploying multifidelity Monte Carlo methods [87], specifically Multifidelity ABC [93,94]. By deploying multifidelity Bayesian methods, the total number of simulations required doesn't decrease but simulations can be generated more cheaply to evaluate weights that allows us to construct the ABC posterior $\pi^{h_M}_{ABC}(\theta \mid s_{obs})$ with a lower computational burden. The success of multifidelity methods is hinged on whether the low-fidelity model is an adequate choice of model for the high-fidelity model. In the work of [94] (and indeed their preliminary work [93]), they work with multifidelity models that are inherently coupled which speeds up the multifidelity framework and allows for a direct comparison between the low and high-fidelity model. As a result, the distance metric, output spaces and distance thresholds used

for the high-fidelity model and the low-fidelity model are equal. Here we explore the case when the high-fidelity ($\mathcal{M}_\theta$) and low-fidelity ($\tilde{\mathcal{M}}_\theta$) models have different output spaces, where the low-fidelity model is an approximation of the high-fidelity model with both models sharing the same parametrisation, $\theta \in \Theta$. When the low-fidelity and high-fidelity models aren't coupled or naturally aligned, there is a need to evaluate the adequacy of the low-fidelity model in capturing the dynamics of the high-fidelity model.

To assess the adequacy or quality of the low-fidelity model compared to the high-fidelity model, we employ the methods introduced in [1], which designs an approximate abstraction of a stochastic system and assesses its quality. Rather than assessing the abstraction $\mathcal{M}$ of a stochastic system $\mathcal{S}$, we use the metric introduced in [1] to quantify how well the low-fidelity model $\tilde{\mathcal{M}}_\theta$ acts as an abstraction of the high-fidelity model $\mathcal{M}_\theta$, that is, whether the low-fidelity model captures the dynamics of the high-fidelity model to a high-degree of accuracy.

In this next chapter, we present the integration of these three techniques (MF-ABC-SMC [94], Bayesian SMC [120] and the approximate abstraction technique [1]) to improve the scalability of this Bayesian verification framework. Upon presenting the integration of these techniques to improve the scalability of the framework, we also apply and analyse this improved framework on CRN case-studies.

Chapter 5

Multifidelity Bayesian Verification

In the previous chapter we extended the scalability and generality of the Bayesian verification framework by deploying statistical based model checking techniques and supervised machine learning. This was done by running statistical model checking on every instantiated model where parameter samples were either accepted or rejected and subsequently classified with supervised machine learning techniques, specifically support vector machines. Whilst this extension allows one to apply the Bayesian verification framework on a wider variety of parametrised stochastic models, the number of simulations required to verify properties given data becomes restrictive.

In this chapter we introduce techniques that reduce the total number of simulations required, as well as the computational cost of generating these simulations, allowing for a more general and scalable Bayesian verification technique. We reduce the computational cost of generating the simulations by deploying multifidelity methods [87, 93, 94]; multifidelity methods utilise both high- and LF models, where high-fidelity (HF) models act as the "true" model and low-fidelity (LF) models serve as an approximation to the HF model that is faster or computationally cheaper to evaluate. We make use of an approximate abstraction metric [1] to measure how adequate of an approximation the LF is to the HF model.

Whilst the use of multifidelity methods reduces the computational cost of the simulations, we also aim to reduce the total number of simulations by exploring an alternative statistical model checking technique and consider *when* to perform this alternative SMC method. The alternative SMC technique we explore makes use of the Bayesian interpretation of statistics, specifically working with the Bayesian statistical model checking [120] technique as opposed to the frequentist model checking approach presented previously, the Massart algorithm [64]. In the framework introduced in the previous chapter, the SMC was performed for each sampled parameter. Whilst this leads to a thorough exploration of the satisfaction probabilities $\Lambda_\phi(\theta)$ across the

parameter space Θ, having to run extensive SMC on every sampled parameter quickly becomes computationally restrictive. Instead of performing extensive SMC on every sampled parameter we will explore when to perform SMC and what insights we gain from doing so at particular stages of the algorithm.

In addition to reducing the computational cost of inference, we also deploy multifidelity methods within the SMC aspect of our scheme to perform SMC on both the high- and LF methods. By doing so we calculate the aforementioned approximate abstraction metric and we are given further information to explore the relationship between the multifidelity models and the property of interest. By integrating these multifidelity techniques into the Bayesian verification technique, we conclude the chapter with the introduction and application of a multifidelity Bayesian verification framework that addresses the computational overheads of the simulation-based Bayesian verification technique introduced in the previous chapter.

We begin this chapter with an overview of Bayesian SMC and compare this with the Massart algorithm introduced previously. Multifidelity inference techniques are then presented and we illustrate how the approximate abstraction metric can be used to assess the quality of the LF model with respect to the HF model. In this chapter the HF model are pCTMCs which are used to represent an underlying chemical reaction network. The LF model we work with are pSDEs, making use of the chemical Langevin equation approximation to the pCTMC. The multifidelity inference technique, Bayesian SMC and approximate abstraction metric are integrated to culminate with the presentation and application of a multifidelity Bayesian verification framework, applying this framework on CRNs with limited and noisy data.

5.1 Bayesian Statistical Model Checking

The Massart algorithm for statistical model checking introduced in the previous chapter relies on frequentist statistical results to quantify the confidence attached with the estimate of the satisfaction probability of a given property. An alternative to this approach is that of Bayesian statistical model checking [65,120]. In the work of [65,120], the Bayesian framework allows us to use Bayes' theorem to incorporate prior information about the model being verified, meanwhile the sequential sampling method (similar to that used in the sequential Massart algorithm presented fully in [63] and above) determines the sample size at run-time of the algorithm. Depending on the evidence gathered by the samples observed so far, this can lead to a significantly

smaller number of sampled traces required to estimate satisfaction probabilities. Zuliani et al. [65,120] presents two forms of Bayesian statistical model checking; in [65], the authors treats the SMC as a Bayesian hypothesis testing problem, considered in Chapter 3, while in [120] SMC is performed via Bayesian interval estimation. In the Bayesian hypothesis testing method, we verify whether a model, say $\mathcal{M}_\theta$, satisfies a given property $\phi = P_{\geq\zeta}[\varphi]$, that is whether $\mathcal{M}_\theta \models P_{\geq\zeta}[\varphi]$, or alternatively whether $\Lambda_\phi(\theta) \geq \zeta$, where φ is the path formula corresponding to the CSL formula ϕ. The resulting hypotheses to be tested are the null hypothesis, given by $\mathcal{H}_0 : \mathcal{M}_\theta \models P_{\geq\zeta}[\varphi]$, versus the alternative hypothesis $\mathcal{H}_1 : \mathcal{M}_\theta \models P_{<\zeta}[\varphi]$. Given that there is less emphasis on estimating the satisfaction probabilities to a high degree of accuracy and more emphasis on ensuring in concluding whether $\mathcal{H}_0$ or $\mathcal{H}_1$ is true, the Bayesian hypothesis testing SMC method can converge much faster than the Bayesian interval estimation SMC [120]. Whilst the Bayesian hypothesis technique verifies whether a model satisfies property with a probability at least ζ, we are interested in estimating precise satisfaction probability values with statistical guarantees, $\hat{\Lambda}_\phi(\theta)$, to allow for a further detailed analysis. We now present the Bayesian interval estimation SMC method presented in [120] and refer the interested reader to [65, 120] for further details on the Bayesian hypothesis SMC method. Recall that the goal of SMC is to calculate an estimate of the true satisfaction probability $\Lambda_\phi(\theta)$, which we denote as $\hat{\Lambda}_\phi(\theta)$. Identically to the work of [64], the estimate will be given in the form of a confidence interval.

The statistical model checking problem can be quantified by estimating a Bernoulli random variable with unknown mean $\Lambda_\phi(\theta)$. Given a trace $y \sim p(y \mid \theta)$ from a model $\mathcal{M}_\theta$ ($y \sim \mathcal{M}_\theta$), the satisfaction of this trace $y \models \varphi$ is quantified by a Bernoulli random variable V with mean parameter u. The conditional probability density function associated with V is given by $p(v \mid u) = u^v(1-u)^{1-v}$, where $v = 1$ if $y \models \varphi$, otherwise $v = 0$. It is then clear to see that $u = \Lambda_\phi(\theta)$, the satisfaction probability that we wish to estimate. In a frequentist approach, $\Lambda_\phi(\theta)$ is estimated using Monte Carlo approaches, whereas in a Bayesian approach $\Lambda_\phi(\theta)$ is treated as a random variable. As such, the satisfaction probability requires a prior probability density, $\pi(\cdot)$, where the prior can either be based on prior information on the system of interest, or we can use a non-informative prior. We know however that the satisfaction probability $\Lambda_\phi(\theta) \in [0, 1]$ and as such, the prior density needs to be defined over $[0, 1]$. For the satisfaction probability, a natural choice for the prior are Beta priors, $\pi(\cdot) =$

$\pi(u, \alpha, \beta)$, as the Beta distribution is a conjugate prior to the Bernoulli distribution.[1] If dealing with non-conjugate priors, we would often need to resort to numerical integration techniques to approximate the posterior or sampling techniques which rely on Monte Carlo approximations. The Beta probability distribution, with real parameters $\alpha, \beta > 0$, is defined as

$$\pi(u, \alpha, \beta) = \frac{1}{B(\alpha, \beta)} u^{\alpha-1}(1-u)^{\beta-1},$$

where the Beta function $B(\alpha, \beta)$ is defined as $B(\alpha, \beta) = \int_0^1 t^{\alpha-1}(1-t)^{\beta-1}\,\mathrm{d}t$. Varying the shape parameters α and β allows one to approximate unimodal densities on the interval $(0, 1)$, incorporating prior information into the model. If we are lacking prior information, a non-informative uniform prior can be chosen by setting $\alpha = \beta = 1$. The cumulative density function $F_{(\alpha,\beta)}(u)$ is defined by

$$F_{(\alpha,\beta)}(u) = \int_0^u \pi(t, \alpha, \beta)\,\mathrm{d}t = \frac{1}{B(\alpha, \beta)} \int_0^u t^{\alpha-1}(1-t)^{\beta-1}. \tag{5.1}$$

We observe the data $\mathcal{V} = \{v_1, \ldots, v_n\}$ which is sampled from a Bernoulli distribution $f(\cdot|u)$ with unknown u, where parameter u has a prior density $g(\cdot)$, recalling Bayes' theorem we define

$$f(u \mid \mathcal{V}) = \frac{f(\mathcal{V} \mid u)g(u)}{\int_0^1 f(\mathcal{V} \mid l)g(l)\,\mathrm{d}l}, \tag{5.2}$$

alternatively, keeping consistent with our Bayesian notation in our work,

$$\pi(u \mid \mathcal{V}) = \frac{\mathcal{L}(\mathcal{V} \mid u)\pi(u)}{\int_0^1 p(\mathcal{V} \mid l)\pi(l)\,\mathrm{d}l}. \tag{5.3}$$

Here, given that v_i (from the Bernoulli distribution defined earlier) are independent and identically distributed (iid), the likelihood factorises as $\mathcal{L}(\mathcal{V} \mid u) = p(v_1, \ldots, v_n \mid u) = \Pi_{i=1}^n p(v_i \mid u)$. Thus, we can estimate $\Lambda_\phi(\theta)$ by the posterior mean. In fact, the posterior mean is a posterior Bayes estimate of $\Lambda_\phi(\theta)$, i.e., it minimizes the risk over the whole parameter space of $\Lambda_\phi(\theta)$. If we define a coverage goal $c \in (\frac{1}{2}, 1)$, any interval (l_0, l_1) such that

$$\int_{l_0}^{l_1} \pi(u \mid v_1, \ldots, v_n)\,\mathrm{d}u = c, \tag{5.4}$$

is defined as a $100c$ percent Bayesian interval estimate or Bayesian credible interval of $u = \Lambda_\phi(\theta)$. An optimal interval that minimises $l_1 - l_0$ can be determined for some

[1] A distribution $\pi(\theta)$ is said to be a conjugate prior for a likelihood function, $\pi(\mathcal{D} \mid \theta)$ if the posterior $\pi(\theta \mid \mathcal{D})$ is in the same family of distributions as the prior $\pi(\theta)$.

cases [120] but in general it is difficult to determine these optimal interval estimates and this approach is not considered here. Rather, we fix a desired half-interval width ℓ and sample until the posterior probability of an interval of width 2ℓ which contains the posterior mean exceeds the coverage parameter c. When sampling from a Bernoulli distribution with conjugate Beta prior distribution parameters α and β, the estimated mean of the posterior with unknown parameter $u = \Lambda_\phi(\theta)$ is given by

$$\hat{u} = \hat{\Lambda}_\phi(\theta) = \frac{v + \alpha}{n + \alpha + \beta}, \tag{5.5}$$

where $v = \sum_{i=1}^{n} v_i$ is the number of successes and $\mathcal{V}$ can be constructed from the sampled simulated data $\mathcal{D} = \{y_1, \ldots, y_n\}$. The integral in Equation (5.4) can be calculated analytically in terms of the Beta distribution,

$$\int_{l_0}^{l_1} \pi(u \mid v_1, \ldots, v_n)\, \mathrm{d}u = F_{(v+\alpha, n-v+\beta)}(l_1) - F_{(v+\alpha, n-v+\beta)}(l_0), \tag{5.6}$$

where $F_{(\alpha,\beta)}(\cdot)$ is the Beta cumulative density function, defined in (5.1). We now have the necessary materials to present the Bayesian statistical model checking algorithm (BSMC) (via Bayesian interval/credibility estimation).

Let us recall the problem at hand. We wish to compute an interval estimate of $\Lambda_\phi(\theta) = P(\mathcal{M}_\theta \models \phi)$, where $\mathcal{M}_\theta$ is a stochastic model instantiated by parameters θ and ϕ is a property of interest defined in a formal logic. We also define a half-size interval $\ell \in (0, \frac{1}{2})$ of the derived interval estimate for $\Lambda_\phi(\theta)$, a coverage parameter $c \in (\frac{1}{2}, 1)$ and coefficients α, β to be decided for the Beta prior, $\pi(\cdot)$. The BSMC algorithm proceeds as follows. We generate traces $y_i \sim p(y|\theta)$ from a model $\mathcal{M}_\theta$ and verify whether the trace satisfies a property of interest $y_i \models \varphi$, where φ is the path formula associated with ϕ. At each iteration of the algorithm, we compute the posterior mean $\hat{\Lambda}_\phi(\theta)$ analytically, which is the Bayes estimator for $\Lambda_\phi(\theta)$, and then compute the posterior probability of the interval $(l_0, l_1) = (\hat{\Lambda}_\phi(\theta) - \ell, \hat{\Lambda}_\phi(\theta) + \ell)$, given by

$$\Upsilon = \int_{l_0}^{l_1} \pi(u \mid v_1, \ldots, v_n)\, \mathrm{d}u. \tag{5.7}$$

If $\Upsilon \geq c$, then the algorithm terminates and returns $\hat{\Lambda}_\phi(\theta)$ with corresponding upper and lower limits, $(l_0, l_1) = (\hat{\Lambda}_\phi(\theta) - \ell, \hat{\Lambda}_\phi(\theta) + \ell) = (\hat{\Lambda}^L_\phi(\theta), \hat{\Lambda}^U_\phi(\theta))$. The algorithm always returns an interval with a fixed width of 2ℓ, which cannot be guaranteed for the sequential Massart algorithm [64] presented in the previous chapter. With this fixed interval width, if the posterior mean estimate, $\Lambda_\phi(\theta)$, converges to the extreme points of the interval $(0, 1)$, the algorithm returns $(l_0, l_1) = (0, 2\ell)$ if $l_0 < 0$ or $(l_0, l_1) = (1 - 2\ell, 1)$ if $l_1 > 1$. The BSMC algorithm is presented in Algorithm 6.

Algorithm 6 Bayesian Statistical Model Checking (BSMC)

Input:

- Parametrised stochastic model $\mathcal{M}_\theta$
- Property ϕ defined in a formal logic, with path formula φ
- Half-interval width size $\ell \in (0, \frac{1}{2})$
- Interval coverage parameter $c \in (\frac{1}{2}, 1)$
- Prior Beta distribution with parameters α, β for (unknown) probability $\Lambda_\phi(\theta) = P(\mathcal{M}_\theta \models \phi)$

Output:

- Posterior mean estimate $\hat{\Lambda}_\phi(\theta)$ for true probability $\Lambda_\phi(\theta)$
- Interval of width 2ℓ, $(l_0, l_1) = (\hat{\Lambda}^L_\phi(\theta), \hat{\Lambda}^U_\phi(\theta))$ with posterior probability at least c

Initialise $n = 0$, total number of simulations.
Initialise $v = 0$, number of traces satisfying, $y \models \varphi$
while $\Upsilon < c$ **do**
    Generate trace $y \sim p(y \mid \theta)$ from $\mathcal{M}_\theta$
    $n \leftarrow n + 1$
    if $y \models \varphi$ **then**
        $v \leftarrow v + 1$
    end if
    Calculate $\hat{\Lambda}_\phi(\theta) = (v + \alpha)/(n + \alpha + \beta)$
    Set $(l_0, l_1) = (\hat{\Lambda}_\phi(\theta) - \ell, \hat{\Lambda}_\phi(\theta) + \ell)$
    if $l_1 > 1$ **then**
        $(l_0, l_1) = (1 - 2\ell, 1)$
    else if $l_0 < 0$ **then**
        $(l_0, l_1) = (0, 2\ell)$
    end if
    Compute posterior probability $\Upsilon = F_{(v+\alpha, n-v+\beta)}(l_1) - F_{(v+\alpha, n-v+\beta)}(l_0)$
end while
return Posterior mean estimate, $\hat{\Lambda}_\phi(\theta)$ and interval $(l_0, l_1) = \left(\hat{\Lambda}^L_\phi(\theta), \hat{\Lambda}^U_\phi(\theta)\right) = \left(\hat{\Lambda}_\phi(\theta) - \ell, \hat{\Lambda}_\phi(\theta) + \ell\right)$.

Note that the posterior mean $\hat{\Lambda}_\phi(\theta)$ estimated in BSMC is a biased estimator of the true probability $\Lambda_\phi(\theta)$. However, this isn't too problematic as this posterior mean is a consistent estimator, that is, it converges in probability to the true probability as the sample size tends to infinity. Unlike the Massart Algorithm, an upper limit of the number of simulations required in BSMC isn't formally defined, although the BSMC algorithm terminates with a probability of one [120]. In the work of [120], they show that the approach can be up to two orders of magnitude faster than SMC techniques based on the Chernoff-Hoeffding [32] bounds, which provide a minimum number of samples required for statistical guarantees. The authors also derive Bayesian Type I (reject null hypothesis $\mathcal{H}_0$ when $\mathcal{H}_0$ is true) and Type II (don't reject null hypothesis $\mathcal{H}_0$ when $\mathcal{H}_0$ is false) error probability bounds for both the hypothesis testing variant and the interval estimation variant of Bayesian SMC. Here we present the error bounds for the interval estimation algorithm [120].

Theorem 5.1.1 (Error bound for estimation)**.** *For any discrete random variable and prior, the Type I and II errors for the output interval* (l_0, l_1) *of the Bayesian SMC Algorithm 6 are bounded above by* $(1-c)\pi_0/c(1-\pi_0)$*, where* c *is the coverage coefficient given as input and* π_0 *is the prior probability of the hypothesis* $\mathcal{H}_0 : \Lambda_\phi(\theta) \in (l_0, l_1)$*.*

Both SMC based on Chernoff-Hoeffding or Bayesian methods offer bounded confidence intervals, where for SMC methods utilising Chernoff-Hoeffding bounds, these bounds hold for any random variable with a bounded variance, meanwhile the Bayesian SMC approach explicitly constructs the posterior distribution based on the Bernoulli distribution. Through some empirical results, the authors show that the performance of the BSMC algorithm depend more on the size of the half-size ℓ interval than that of the coverage of the interval, although further theoretical work is needed to confirm this. However, we note that the Bayesian SMC result indeed requires a fewer number of simulations in comparison with the frequentist Massart Algorithm introduced above. Whilst not formally proved here, similar to the comparison undertaken in [120], we compare the frequentist SMC approach with the Bayesian approach by comparing the average number of simulations required to obtain results from BSMC (Algorithm 6) and from the Massart Algorithm [63]. Results are shown in Figure 5.1.

Relationship between confidence intervals and Bayesian credibility intervals

In some circles, confidence intervals and credibility intervals are used interchangeably but when comparing methods that are derived from frequentist and Bayesian perspectives it is very important to distinguish the two. The coverage probability c of a

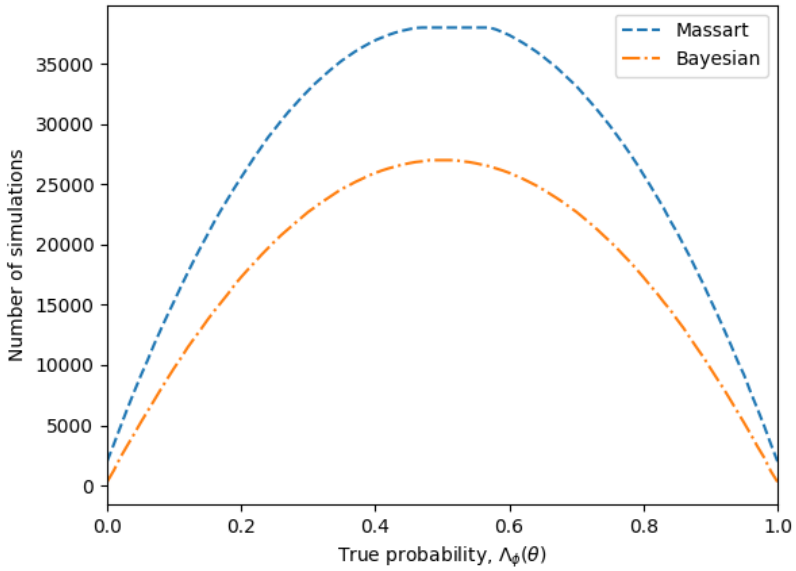


Figure 5.1. Comparison of average number of simulations required to estimate the true probability of satisfying a given property, $\Lambda_\phi(\theta)$ using Bayesian SMC or frequentist SMC (Massart Algorithm [64]) with coverage parameter $c = 0.999$ ($(1-\delta)$-confidence interval) and width parameter $\ell = 0.01$. ($\epsilon = 0.01$). The number of samples obtained is the mean of 100 repetitions of each SMC technique.

Bayesian interval is a probability statement about the unknown parameter belonging to a specified interval, given the data collected thus far and prior information. However, the coverage probability carried by a confidence interval of $(1-\delta)$ introduced previously is defined differently. With a confidence interval of $(1-\delta)$, this tells us that $(1-\delta)\%$ of the time, the procedure will output an interval that contains the unknown parameter.

Returning to the Bayesian perspective, under certain conditions the Bernstein-von Mises theorem [109] states that the posterior distribution is said to converge to a multivariate normal distribution. This means, that with a large sample size, the posterior distribution converges to a normal distribution of mean $\Lambda_\phi(\theta)$ and a variance which is proportional to $1/n$, where n is the sample size. Following the asymptotic normality of the posterior distribution, the coverage probability of Bayesian credible intervals that are centred on the posterior mean will be equal to that of confidence intervals that are computed via techniques based on the normal distribution. However, in the more practical case when we have a finite sample size, Bayesian estimates will only enjoy frequentist properties when non-informative or vague priors are deployed.

5.2 Multifidelity Methods

For the Bayesian verification framework introduced in the previous chapter, in addition to the simulations needed to verify properties of interest through SMC, we also rely on simulation-based likelihood-free inference techniques, namely ABC [101], and make use of the same simulations to simultaneously infer parameters and verify properties of interest. Whilst ABC methods are popular due to the ease with which they can be utilised to provide likelihood-free inference, ABC methods (in addition to SMC) rely on a large number of simulations and can lead to high computational costs, especially if they require a high number of parameter proposals and subsequent simulations. In this section we introduce the Multifidelity Approximate Bayesian Computation (MF-ABC) [93] and Multifidelity Approximate Bayesian Computation Sequential Monte Carlo [94] algorithms, which make use of multifidelity methods [87] to reduce the computational cost of ABC sampling. In the multifidelity framework, we work directly with the simulations y and not with a summary statistic thereof (previously denoted as $s = S(y)$). We focus first on the extension of ABC methods to work with multifidelity methods, then discuss the sequential Monte Carlo extension to MF-ABC.

Multifidelity Approximate Bayesian Computation

Whilst ABC methods are popular due to the ease with which they can be utilised to provide likelihood-free inference, they rely on a large number of simulations and can lead to high computational costs, especially if they require a high number of parameter proposals and subsequent simulations. In this section we introduce MF-ABC [93] algorithms, which make use of multifidelity methods [87] to reduce the computational cost of ABC sampling. Notice that in the multifidelity framework, we work directly with the simulations y and not with summary statistic thereof (previously denoted as $s = S(y)$). In multifidelity methods, we consider both a HF model of interest and a corresponding LF model, which is a model that approximates the HF model but is significantly cheaper to generate simulations from.

The HF model $p(y \mid \theta)$ or $\mathcal{M}_\theta$, is a detailed model that is computationally expensive to generate simulations from: generating $y \sim p(y \mid \theta)$, where $y \in \mathcal{Y}$ is the output space, comes at a high computational cost, say $c(\theta)$. The LF model, denoted by $\tilde{p}(\tilde{y} \mid \theta)$ or $\tilde{\mathcal{M}}_\theta$, maps parameter samples $\theta \in \Theta$ from a common parameter space to a distribution on an output space $\tilde{\mathcal{Y}}$. Whilst in general this output space may differ from that of the HF model, we assume the signals y and $\tilde{y}$ can be compared

amongst each other and against the observations y_{obs}. (In our setup, the HF model is the pCTMC $\mathcal{M}_\theta$, whereas the LF model is the pSDE $\tilde{\mathcal{M}}_\theta$.) For the remainder of our work, we assume that both the HF and LF models share a parameter space Θ. In the case that the LF model and HF model have different parameter spaces, we could extend the framework in a similar fashion as to how ABC-SMC [108] can be extended to consider alternative models with alternative parameter spaces. In their work, they first sample a competing parametrised model which comes paired with its own parameter space (that may be of different dimensionality). Once the model has been sampled, parameter sampling of the parameter space then proceeds as normal and the parameters may be accepted or rejected subject to a common criterion. Within the framework of MF-ABC, if desired, we assume that we can define a LF equivalent of y_{obs}, denoted by $\tilde{y}_{obs}$, which we obtain by mapping y_{obs} to the output space $\tilde{\mathcal{Y}}$. In our work however, we simply work with the case where $\tilde{y}_{obs} = y_{obs}$. Similarly to how a distance function $d(y, y_{obs})$ and a neighbourhood Ω_h is defined for ABC, one can define a LF distance function $\tilde{d}(\tilde{y}, \tilde{y}_{obs})$ and a neighbourhood $\tilde{\Omega}_{\tilde{h}} = \{\tilde{y} \in \tilde{\mathcal{Y}} \mid \tilde{d}(\tilde{y}, \tilde{y}_{obs})\}$ with corresponding kernel $\mathbb{1}_{\tilde{\Omega}_{\tilde{h}}}(\tilde{y})$ associated with this alternative output space and LF model.

For the work introduced in this chapter, we set $\tilde{y}_{obs} = y_{obs}$ and we use simple indicator functions for both the high- and LF kernels, where $\mathbb{1}_{\Omega_h}(y)$ is the kernel for the HF model and $\mathbb{1}_{\tilde{\Omega}_{\tilde{h}}}(\tilde{y})$ is the LF kernel, where the distance metric is chosen to be the same, that is, $\tilde{d}(\cdot, \cdot) = d(\cdot, \cdot)$. These LF models come with an associated cost $\tilde{c}(\theta)$ and it is assumed that it is cheaper to generate simulations from the LF model as opposed to the HF model, $\mathbb{E}[\tilde{c}(\theta)] \ll \mathbb{E}[c(\theta)]$. With this lower cost, parameters and their associated weights, $\tilde{w}(\theta) = \mathbb{1}_{\tilde{\Omega}_{\tilde{h}}}(\tilde{y})$, $\{\theta^{(i)}, \tilde{w}^{(i)}\}_{i=1}^N$ can be sampled quickly compared to the HF equivalent $\{\theta^{(i)}, w^{(i)}\}_{i=1}^N$. However, the ABC approximations are not identical, $\mathbb{1}_{\Omega_h}(y)p(y \mid \theta)\pi(\theta) \neq \mathbb{1}_{\tilde{\Omega}_{\tilde{h}}}(\tilde{y})\tilde{p}(\tilde{y} \mid \theta)\pi(\theta)$ and using the LF model incurs a further bias when implementing the ABC scheme. Prescott and Baker [93] develop an MF-ABC algorithm that utilises both the low- and HF models to estimate $\pi_{ABC}(\theta \mid y_{obs})$. Doing so reduces the reliance of the ABC algorithm on generating simulations on the HF model which reduces the computational burden of the algorithm. The key idea to the MF-ABC algorithm is the introduction of a multifidelity weight, namely

$$w_{\text{mf}}(\theta) = \mathbb{1}_{\tilde{\Omega}_{\tilde{h}}}(\tilde{y}) + \frac{\mathbb{I}(U < \eta(\tilde{y}))}{\eta(\tilde{y})} \left[\mathbb{1}_{\Omega_h}(y) - \mathbb{1}_{\tilde{\Omega}_{\tilde{h}}}(\tilde{y})\right], \tag{5.8}$$

where $U \sim \mathcal{U}(0, 1)$ and $\eta(\tilde{y}) \in (0, 1)$ is a continuation probability, defined by

$$\eta(\tilde{y}) = \eta_1 \mathbb{1}_{\tilde{\Omega}_{\tilde{h}}}(\tilde{y}) + \eta_2(1 - \mathbb{1}_{\tilde{\Omega}_{\tilde{h}}}(\tilde{y})). \tag{5.9}$$

This choice of $\eta(\tilde{y})$ allows both early acceptance and early rejection with constant probabilities $1-\eta_1$ and $1-\eta_2$ respectively. Here, the values of (η_1, η_2) are inputs to be determined at the discretion of the user where $\eta_1 \in [0,1]$ and $\eta_2 \in [0,1]$. An extensive analysis of the performance of the algorithm as well as optimal estimated values for (η_1, η_2) are provided in the work of [93], where the optimal values of (η_1, η_2) are estimated with a burn-in period of the MF-ABC algorithm, see Appendix B.1. The continuation probabilities explored and used in [93, 94] may not be the optimal choice for what we aim to achieve in our work. The continuation probabilities were chosen such that they optimally balance the benefit of reducing the simulation time against the cost of reducing the effective sample size (ESS), maximising the limiting efficiency of their algorithm. For our work we could consider a continuation probability that takes into account the satisfaction of a particular property for example, but for simplicity we consider the above continuation probability.

The multifidelity weight defined in Equation (5.8) combines ideas in alternative ABC techniques that make use of LF simulations to speed up the ABC algorithm. Specifically, the MF-ABC algorithm combines ideas from early rejection ABC [33, 91] and early decision ABC. In early rejection ABC, the low fidelity simulation $\tilde{y}$ is used to determine whether or not to proceed and simulate $y \sim p(y \mid \theta)$ from the HF model, meanwhile in early decision ABC, the LF simulations are used to determine whether or not the proposed parameter is to be accepted without simulating the HF model. Combining these ideas for early rejection and early decision ABC into the multifidelity weight $w_{\text{mf}}(\theta)$ gives rise to four possible values:

$$w_{\text{mf}}(\theta) = \begin{cases} 1 & \tilde{y} \in \tilde{\Omega}_{\tilde{h}} \cap U \geq \eta_1 \text{ (early accept)} \\ 0 & \tilde{y} \notin \tilde{\Omega}_{\tilde{h}} \cap U \geq \eta_2 \text{ (early reject)} \\ 1 & \tilde{y} \in \tilde{\Omega}_{\tilde{h}} \cap y \in \Omega_h \cap U < \eta_1 \text{ (checked true positives)} \\ 0 & \tilde{y} \notin \tilde{\Omega}_{\tilde{h}} \cap y \notin \Omega_h \cap U < \eta_2 \text{ (checked true negative)} \\ 1 - 1/\eta_1 & \tilde{y} \in \tilde{\Omega}_{\tilde{h}} \cap y \notin \Omega_h \cap U < \eta_1 \text{ (checked false positive)} \\ 0 + 1/\eta_2 & \tilde{y} \notin \tilde{\Omega}_{\tilde{h}} \cap y \in \Omega_h \cap U < \eta_2 \text{ (checked false negative).} \end{cases} \tag{5.10}$$

Unfortunately, due to the possible negative values, the pairs $\{\theta^{(i)}, w_{\text{mf}}^{(i)}\}_{i=1}^N$ cannot be treated as direct weighted samples from the ABC posterior $\pi_{ABC}(\theta \mid y_{obs})$. Despite this shortcoming, we still obtain an estimator for the ABC approximation of the posterior expectation of an arbitrary function $g(\theta)$ as

$$\mathbb{E}_{ABC}(g(\theta) \mid y_{obs}) = \int g(\theta) \pi_{ABC}(\theta \mid y_{obs}) \, \mathrm{d}\theta \approx \frac{\sum_{i=1}^N w^{(i)} g(\theta^{(i)})/N}{\sum_{j=1}^N w^{(i)}/N}. \tag{5.11}$$

Algorithm 7 summarises the Multifidelity Approximate Bayesian Computation Rejection Sampler (MF-ABC-RS).

Algorithm 7 Multifidelity ABC Rejection Sampler (MF-ABC-RS)

Input:

- Observed data y_{obs} $(= \tilde{y}_{obs})$
- Prior distribution $\pi(\theta)$, HF model $p(y \mid \theta)$ and LF model $\tilde{p}(\tilde{y} \mid \theta)$
- High fidelity neighbourhood $\Omega_h = \{y \in \mathcal{Y} \mid d(y, y_{obs}) \leq h\}$ and LF neighbourhood $\tilde{\Omega}_{\tilde{h}} = \{\tilde{y} \in \tilde{\mathcal{Y}} \mid \tilde{d}(\tilde{y}, \tilde{y}_{obs}) \leq \tilde{h}\}$ with respective kernels $\mathbb{1}_{\Omega_h}(y)$ and $\mathbb{1}_{\tilde{\Omega}_{\tilde{h}}}(\tilde{y})$
- Continuation probabilities (η_1, η_2)
- Monte Carlo sample size $N > 0$

Output:

- $\{\theta^{(i)}, w^{(i)}\}_{i=1}^N$, accepted parameters with corresponding weights

for $i = 0, \ldots, N$ **do**
  Generate $\theta^{(i)} \sim \pi(\theta)$ and $U \sim \mathcal{U}(0, 1)$
  Simulate $\tilde{y} \sim \tilde{p}(\tilde{y} \mid \theta^{(i)})$ from LF model
  Calculate $\tilde{w} = \mathbb{1}_{\tilde{\Omega}_{\tilde{h}}}(\tilde{y})$
  Set $w^{(i)} = \tilde{w}$ and $\eta = \eta_1 \tilde{w} + \eta_2(1 - \tilde{w})$
  if $U < \eta$ **then**
    Simulate $y \sim p(y \mid \theta^{(i)})$ from the HF model
    Calculate $w = \mathbb{1}_{\Omega_h}(y)$
    Update $w^{(i)} \leftarrow w^{(i)} + (w - w^{(i)})/\eta$
  end if
end for
(Optional) Calculate the approximation to $\mathbb{E}_{ABC}(g(\theta) \mid y_{obs})$, $\hat{\mathbb{E}}_{ABC}(g(\theta) \mid y_{obs}) = \sum_{i=1}^N w^{(i)} g(\theta^{(i)}) / \sum_{i=1}^N w^{(i)}$
return $\{\theta^{(i)}, w^{(i)}\}_{i=1}^N$

Multifidelity Approximate Bayesian Computation - Sequential Monte Carlo

The multifidelity approach of [93] presented in the previous section improves the performance of the ABC algorithm (Algorithm 2) by allowing sampled parameters to be weighted and evaluated with a combination of low- and HF models. This method specifically allows one to take advantage of LF models to evaluate weights instead of relying solely on HF models, which means that a fixed number of Monte Carlo samples N from the ABC posterior $\pi_{ABC}(\theta \mid y_{obs})$ can be obtained for a lower computational cost, and thus faster, compared to HF models. Whilst this extension doesn't decrease the total number of simulations required, it does increase the speed at which weights can be evaluated. SMC approaches [107, 108] on the other hand were introduced to reduce the total number of simulations needed to approximate the likelihood by exploring the parameter space better than basic ABC. In [94] the authors introduce a framework that brings these two orthogonal approaches of improving the ABC framework together, as a combined multifidelity sequential Monte Carlo ABC algorithm (MF-ABC-SMC). The MF-ABC algorithm speeds up the evaluation of the weights given sampled parameters, whereas the SMC approach reduces the total number of sampled parameters (and in turn, simulations) required by exploring the parameter space in a more efficient manner. The two approaches are integrated by extending the ABC importance sampling approach presented in Algorithm 3 to the multifidelity framework, resulting in a Multifidelity ABC Importance sampler scheme (MF-ABC-IS). Akin to how the ABC-RS algorithm can be treated similarly to the ABC-IS algorithm when the importance distribution is equal to the prior distribution $\hat{q}(\theta) = \pi(\theta)$, the MF-ABC-IS scheme can be similarly defined with respect to MF-ABC-RS where we generate parameter samples from a proposal distribution $\hat{q}(\theta)$ instead of the prior $\pi(\theta)$. The continuation probabilities presented in the MF-ABC-IS algorithm, $\eta(\tilde{y})$, now an explicit function on both the LF simulation, $\tilde{y}$, and the parameter samples θ; $\eta = \eta(\theta, \tilde{y}) \in (0, 1]$. The MF-ABC-IS algorithm is presented in Algorithm 8.

With the MF-ABC-IS algorithm, if we were to define both a sequence of decreasing thresholds $h_1 > \cdots > h_M$ that induce the neighbourhoods $\Omega_{h_1} \supseteq \cdots \supset \Omega_{h_M}$ for the HF model and a corresponding sequence of decreasing thresholds for the LF model, $\tilde{h}_1 > \cdots > \tilde{h}_M$ that induce the neighbourhoods $\tilde{\Omega}_{\tilde{h}_1} \supseteq \cdots \supseteq \tilde{\Omega}_{\tilde{h}_m}$, using an appropriate perturbation kernel $F_m(\theta^* \mid \theta)$, then the sequential Monte Carlo algorithm could proceed similarly to the ABC-SMC introduced previously. The difficulty with extending the ABC-SMC scheme to incorporate multifidelity models is that the newly generated weights might be negative. These weights arise when the low fidelity

Algorithm 8 Multifidelity ABC Importance Sampler (MF-ABC-IS)

Input:

- Observed data y_{obs} $(= \tilde{y}_{obs})$
- Prior distribution $\pi(\theta)$, HF model $p(y \mid \theta)$ and LF model $\tilde{p}(\tilde{y} \mid \theta)$
- Importance distribution $\hat{q}(\theta) \propto q(\theta)$
- High fidelity neighbourhood $\Omega_h = \{y \in \mathcal{Y} \mid d(y, y_{obs}) \leq h\}$ and LF neighbourhood $\tilde{\Omega}_{\tilde{h}} = \{\tilde{y} \in \tilde{\mathcal{Y}} \mid \tilde{d}(\tilde{y}, \tilde{y}_{obs}) \leq \tilde{h}\}$ with respective kernels $\mathbb{1}_{\Omega_h}(y)$ and $\mathbb{1}_{\tilde{\Omega}_{\tilde{h}}}(\tilde{y})$
- Continuation probability function $\eta = \eta(\theta, \tilde{y})$
- Monte Carlo sample size $N > 0$

Output:

- $\{\theta^{(i)}, w^{(i)}, \}_{i=1}^{N}$, accepted parameters with corresponding weights

for $i = 0, \ldots, N$ **do**
  Generate $\theta^{(i)} \sim \hat{q}(\theta)$ and $U \sim \mathcal{U}(0, 1)$
  Simulate $\tilde{y} \sim \tilde{p}(\tilde{y} \mid \theta^{(i)})$ from LF model
  Calculate $\tilde{w} = \mathbb{1}_{\tilde{\Omega}_{\tilde{h}}}(\tilde{y})$
  Set $w^{(i)} = \tilde{w}$
  if $U < \eta(\theta, \tilde{y})$ **then**
    Simulate $y \sim p(y \mid \theta^{(i)})$ from the HF model
    Calculate $w = \mathbb{1}_{\Omega_h}(y)$
    Update $w^{(i)} \leftarrow w^{(i)} + (w - w^{(i)})/\eta$
  end if
  $w^{(i)} \leftarrow \left[\pi(\theta^{(i)})/q(\theta^{(i)})\right] w^{(i)}$
end for
return $\{\theta^{(i)}, w^{(i)}\}_{i=1}^{N}$

model satisfies the distance threshold, $\tilde{y} \in \tilde{\Omega}_{\tilde{h}}$, but the corresponding HF model does not satisfy the distance threshold, $y \notin \Omega_h$. The presence of negative weights makes the proposal of an importance distribution $\hat{q}_m(\theta) \propto q_m(\theta)$ that can be sampled from more difficult. To circumvent this problem, the authors in [94] propose an importance distribution that produces non-negative weights $\hat{w}_m^{(i)}$ in parallel with the true weights $w_m^{(i)}$, where $\hat{w}_m^{(i)} = |w_m^{(i)}|$. At each generation m, two weights for the same sampled parameters are produced at once, $\{\theta_m^{(i)}, \hat{w}_m^{(i)}\}$ and $\{\theta_m^{(i)}, w_m^{(i)}\}$. The importance distribution $\hat{q}_m(\theta)$ in Equation (3.12) is replaced with the distribution $\hat{r}_m(\theta)$ that works solely with the non-negative weights $\hat{w}_m^{(i)}$. This importance distribution, $\hat{r}_m(\theta)$, is proportional to

$$r_m(\theta) = \begin{cases} \sum_{i=1}^{N} \hat{w}_{m-1}^{(i)} F_m(\theta | \theta_{m-1}^{(i)}) / \sum_{j=1}^{N} \hat{w}_{m-1}^{(j)} & \text{if } \pi(\theta) > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{5.12}$$

With this alternative importance distribution, the weighted samples $\{\hat{w}_m^{(i)}, \theta_m^{(i)}\}$ are drawn from an alternative target distribution,

$$\varpi_m(\theta) \propto \pi_{ABC}^{h_m}(\theta \mid y_{obs}) + \delta_m(\theta),$$

where the difference between this new target distribution $\varpi_m(\theta)$ and the ABC posterior we wish to sample from $\pi_{ABC}^{h_m}(\theta \mid y_{obs})$ is given by

$$\delta_m(\theta) = 2\pi(\theta) \int_{\mathcal{Y}^2} (1 - \eta(\tilde{y})) \mathbb{1}_{\tilde{\Omega}_{\tilde{h}_m}}(\tilde{y})(1 - \mathbb{1}_{\Omega_{h_m}}(y)) \tilde{p}(\tilde{y} \mid \theta) p(y \mid \theta) \, \mathrm{d}\tilde{y} \, \mathrm{d}y.$$

This new target density contains additional density in regions of parameter space where $w_m^{(i)} < 0$ is more likely. Although this new importance distribution now targets $\varpi_m(\theta)$ at each generation m instead of the desired ABC posterior $\pi_{ABC}^{h_M}(\theta \mid y_{obs})$, the weighted samples $\{\theta_m^{(i)}, w_m^{(i)}\}$ based on the multifidelity weights $w_m^{(i)}$ from each generation's weighting function,

$$w_m(\theta, \tilde{y}, y, u) = \frac{\pi(\theta)}{r_m(\theta)} \left(\mathbb{1}_{\tilde{\Omega}_{\tilde{h}_m}}(\tilde{y}) + \frac{\mathbb{I}(u < \eta(\tilde{y}))}{\eta(y)} \left[\mathbb{1}_{\Omega_{h_m}}(y) - \mathbb{1}_{\tilde{\Omega}_{\tilde{h}_m}}(\tilde{y}) \right] \right), \tag{5.13}$$

remain from the desired ABC posterior $\pi_{ABC}^{h_m}(\theta \mid y_{obs})$, hence with the MF-ABC-SMC introduced in [94], we are able to produce an estimate of a function $g : \Theta \to \mathbb{R}$, such that

$$\mathbb{E}_{ABC}^{h_m}(g(\theta) \mid y_{obs}) \approx \frac{\sum_{i=1}^{N} w_m^{(i)} g(\theta_m^{(i)})}{\sum_{j=1}^{N} w_m^{(j)}}, \tag{5.14}$$

is a consistent Monte Carlo estimate of g under the ABC posterior at the m-th epoch.

The Multifidelity Approximate Bayesian Computation - Sequential Monte Carlo (MF-ABC-SMC) algorithm is presented in Algorithm 9. In the ABC-SMC algorithm we presented in Section 3.2.2 (Algorithm 4), in lines 3-9 we state explicitly how to generate samples from the importance distribution $\hat{q}(\theta)$ and for brevity, we have omitted this level of detail in Algorithm 9. We note that the continuation probabilities for generation $m+1$, $\eta_{m+1}(\theta, \tilde{y})$, are constructed at generation m after obtaining the Monte Carlo outputs of Algorithm 8 in each generation. The full derivations required (in addition to the rationale) to estimate the optimal continuation probabilities in Algorithm 9 are given in [94] where we have presented the result of this derivation in Appendix B.2.

In the framework introduced in [93,94] the multifidelity model allows for a coupling between the LF ($\tilde{p}(\tilde{y} \mid \theta)$) and HF ($p(y \mid \theta)$) model, which allows one to produce a HF simulation that is constructed conditionally on a previously-simulated LF simulation. Coupled models can range from a LF model that is a good approximation to the HF model, to stochastic models simulated for different time horizons, the LF model being simulated over time $t \in [0, \tau]$ and the HF one being simulated for a time horizon of $t \in [0, T]$ where $\tau < T$. For instance, the HF model might generate simulations of a CTMC model via the Gillespie Algorithm (the known "SSA" 1), whereas the LF

Algorithm 9 Multifidelity ABC-SMC (MF-ABC-SMC)

Input:

- Observed data y_{obs} $(= \tilde{y}_{obs})$
- Prior distribution $\pi(\theta)$, HF model $p(y \mid \theta)$ and LF model $\tilde{p}(\tilde{y} \mid \theta)$
- Initial importance distribution $\hat{r}_0(\theta)$ (commonly set to $\hat{r}_0(\theta) = \pi(\theta)$)
- Sequence of high fidelity neighbourhoods $\Omega_{h_m} = \{y \in \mathcal{Y} \mid d(y, y_{obs}) \leq h_m\}$ and LF neighbourhoods $\tilde{\Omega}_{\tilde{h}_m} = \{\tilde{y} \in \tilde{\mathcal{Y}} \mid \tilde{d}(\tilde{y}, \tilde{y}_{obs}) \leq \tilde{h}_m\}$ with respective kernels $\mathbb{1}_{\Omega_h}(y)$ and $\mathbb{1}_{\tilde{\Omega}_{\tilde{h}}}(\tilde{y})$ and $m = 1, 2, \ldots, M$
- Initial continuation probability function $\eta_0 = \eta_0(\theta, \tilde{y}) = 1$
- Lower bounds on continuation probabilities, $\eta_1^L, \eta_2^L \in (0, 1)$
- Sequence of perturbation kernels $F_m(\theta|\theta^*)$
- Monte Carlo sample size $N > 0$

Output:

- $\{\theta^{(i)}, w^{(i)}, \}_{i=1}^N$, accepted parameters with corresponding weights

1: **for** $m = 0, \ldots, M$ **do**
2: Produce $\{\theta_m^{(i)}, w_m^{(i)}\}_{i=1}^N$ and store information required to estimate optimal continuation probabilities (Appendix B.2) from MF-ABC-IS algorithm (Algorithm 8) with inputs $\left(y_{obs}, \tilde{y}_{obs}, \Omega_{h_m}, \tilde{\Omega}_{\tilde{h}_m}, \pi(\theta), p(y \mid \theta), \tilde{p}(\tilde{y} \mid \theta), \eta_m(\theta, \tilde{y})\right)$ setting $\hat{q} = \hat{r}_m(\theta)$
3: Set $\hat{w}_m^{(i)} = |w_m^{(i)}|$ for $i = 1, \ldots, N$
4: Define $\hat{r}_{m+1}$ proportional to r_{m+1} defined in Equation (5.12)
5: Calculate (η_1^*, η_2^*) with lower bounds (η_1^L, η_2^L) using importance distribution $r_{t+1}(\theta)$, ABC thresholds h_{m+1} and $\tilde{h}_{m+1}$ with values stored from MF-ABC-IS algorithm at step 2. (Appendix (B.2)
6: Set $\eta_{m+1}(\theta, \tilde{y}) = \eta_1^* \mathbb{1}_{\tilde{\Omega}_{\tilde{h}_m}}(\tilde{y}) + \eta_2^*(1 - \mathbb{1}_{\tilde{\Omega}_{\tilde{h}_m}}(\tilde{y}))$
7: **end for**
8: **return** $\{\theta_M^{(i)}, w_M^{(i)}\}_{i=1}^N$

model resorts to τ-leaping simulations [59]. Given this choice of multi-fidelity models, in [93, 94] the parameters $\tilde{h}_m = h_m$ and functions $\tilde{d}(\cdot,\cdot) = d(\cdot,\cdot)$ and $\tilde{\Omega}_{\tilde{h}_m} = \Omega_{h_m}$ are the same. Although multifidelity models that are naturally "close" are convenient to work with, there is a need to formalise the relationship between a LF and HF model.

In the next section we introduce work [1] that discusses a randomised method for the design of an approximate abstraction of a stochastic system and provides an assessment of its quality, which would aim to address this need.

5.2.1 Approximate Abstractions of Stochastic Systems

The work introduced in [1] is concerned with the design of an approximate abstraction of a given stochastic model and with the assessment of its quality. The authors in [1] argue that an abstraction of a system has to resemble the behaviour of the underlying system of interest, whilst also being structurally simpler, much like how LF models should resemble the behaviour of the underlying HF model. Imperative to the success of designing an adequate abstraction of a system is the need for a quantitative way to measure the distance between the abstracted model and the true system. For deterministic models, abstractions can be defined through the notion of bisimulation while for stochastic models, defining and studying abstractions introduces further challenges. The authors propose a randomised approach to determine an approximate abstraction $\mathcal{M}$ of a stochastic system of interest, $\mathcal{S}$, where the approach itself reformulates the problem as a semi-infinite chance-constrained optimisation program that is solved via randomisation.

Instead of proposing an abstracted model $\mathcal{M}$ of a stochastic system of interest $\mathcal{S}$, we will reframe the framework in terms of assessing the quality of a LF model $\tilde{\mathcal{M}}_\theta$ as an abstraction or approximation of a concrete model $\mathcal{M}_\theta$, which in our framework, we assume is an accurate enough model of the true underlying biological system of interest, $\mathcal{S}$. The approach is applicable to any stochastic system $\mathcal{M}_\theta$ with a candidate abstract model $\tilde{\mathcal{M}}_\theta$, with the only pre-requisite being that one is able to generate simulations from both models, that is, $\tilde{y} \sim \tilde{p}(\tilde{y} \mid \theta)$ for the LF model $\tilde{\mathcal{M}}_\theta$, and $y \sim p(y \mid \theta)$ for the HF model $\mathcal{M}_\theta$. The quality of the approximate abstraction is considered by calculating a distance between the two executions of $\tilde{\mathcal{M}}_\theta$ and $\mathcal{M}_\theta$, and given that this metric is determined via randomisation, it comes with a certain degree of confidence (a design parameter). We first define the notion of an approximate abstraction.

Definition 5.2.1 (Approximate Abstraction)**.** A model, $\tilde{\mathcal{M}}_\theta$, is a γ-approximate abstraction of $\mathcal{M}_\theta$ up to a level $1-\varepsilon$ if the following condition is satisfied:

$$P(d_T(y,\tilde{y}) \leq \gamma) \geq 1-\varepsilon, \tag{5.15}$$

where y and $\tilde{y}$ are simulations generated from $\mathcal{M}_\theta$ and $\tilde{\mathcal{M}}_\theta$ respectively, and $d_T(\cdot,\cdot)$ is a distance metric over a time-horizon with terminal time T chosen at the discretion of the user. Here we choose the metric

$$d_T(y,\tilde{y}) = \sup_{t\in[t_0,T]} \|y(t)-\tilde{y}(t)\|, \tag{5.16}$$

where t_0 is the initial time, T is the time horizon and $\tilde{y}(t)$ $(y(t))$ is the state of the LF model (HF model) at time t.

We note that the distance metric here, $d_T(\cdot,\cdot)$, is independent of the distance metric employed for the ABC schemes previously, $d(\cdot,\cdot)$, however both of these metrics can be chosen such that they either align, or alternative metrics that are more appropriate for either the inference aspect of the scheme or formal verification scheme can be chosen, for example, the Wasserstein metric is chosen as the distance metric for ABC in the work of [17] and would be a good choice for $d(\cdot,\cdot)$ but perhaps not $d_T(\cdot,\cdot)$. For the work presented in this thesis, however, here we use the metric presented in [1] for $d_T(\cdot,\cdot)$, using this metric to explore regions of the parameter space Θ where $\tilde{\mathcal{M}}_\theta$ is a good approximation of $\mathcal{M}_\theta$, given $\theta\in\Theta$.

The quality γ of a LF model $\tilde{\mathcal{M}}_\theta$ as an approximate abstraction of $\mathcal{M}_\theta$ up to a level of $1-\varepsilon$ can be assessed by solving the following optimisation problem,

$$\min_{\gamma\in\mathbb{R}} \gamma, \text{ subject to } P(d_T(y,\tilde{y}) \leq \gamma) \geq 1-\varepsilon. \tag{5.17}$$

Let γ_ε be obtained from Equation (5.17), then $\tilde{\mathcal{M}}_\theta$ is a γ_ε-approximate abstraction of $\mathcal{M}_\theta$ up to level $1-\varepsilon$. Computing γ_ε can be done using a randomised approach, presented in Algorithm 10, which provides an estimate of $\gamma_\varepsilon \approx \hat{\gamma}_\varepsilon$ with approximation guarantees.

By discarding the fraction $\frac{z}{N} < \varepsilon$ that corresponds to the largest discrepancy between $\tilde{\mathcal{M}}_\theta$ and $\mathcal{M}_\theta$, one can improve the quality bound γ while guaranteeing that the violation set has size smaller than or equal to the chosen value ε [1]. The estimate $\hat{\gamma}_\varepsilon$ generated from Algorithm 10 can be shown to be an approximation of γ_ε up to a confidence: Specifically, let $\beta_\gamma \in (0,1)$ be a confidence parameter and $\nu_\gamma \in (0,\varepsilon)$ be a degradation parameter; if N and z are chosen such that

$$\sum_{i=0}^{z} \binom{N}{i} \varepsilon^i (1-\varepsilon)^{(N-1)} + \sum_{i=z+1}^{N} \binom{N}{i} (\varepsilon-\nu_\gamma)^i (1-\varepsilon+\nu_\gamma)^{(N-1)} \leq \beta_\gamma,$$

Algorithm 10 Randomised quality assessment of approximate abstraction

Input:

- Instance of parameters $\theta \in \Theta$
- HF model $\mathcal{M}_\theta$ and LF model $\tilde{\mathcal{M}}_\theta$
- $N > 0$, number of realisations
- z, number of realisations to discard where $0 < z < N$

Output:

- $\hat{\gamma}_\varepsilon$, estimate of γ_ε

1: **for** $i = 1, \ldots, N$ **do**
2: Generate $\tilde{y}^{(i)} \sim \tilde{p}(\tilde{y} \mid \theta)$ from $\tilde{\mathcal{M}}_\theta$ and $y^{(i)} \sim p(y \mid \theta)$ from $\mathcal{M}_\theta$ with parametrisation θ
3: Calculate $\hat{\gamma}^{(i)} = d_T(y^{(i)}, \tilde{y}^{(i)})$
4: **end for**
5: Discard the $z(< N)$ largest values of $\hat{\gamma}^{(i)}$, $i = 1, \ldots, N$
6: Set

$$\hat{\gamma}_\varepsilon = \max_{i \in \{1,2,\ldots,N\} \setminus \{l_1, l_2, \ldots, l_k\}} \hat{\gamma}^{(i)}$$

where $\{l_1, l_2, \ldots, l_k\} \subset \{1, 2, \ldots, N\}$ denote the indices of the discarded values of the $\hat{\gamma}^{(i)}$
7: **return** $\hat{\gamma}_\varepsilon$

then, with a probability at least $1 - \beta_\gamma$, the solution $\hat{\gamma}_\varepsilon$ calculated in Algorithm 10 satisfies

1. $P(d_T(y, \tilde{y}) \leq \hat{\gamma}_\varepsilon) \geq 1 - \varepsilon$,
2. $\gamma_\varepsilon \leq \hat{\gamma}_\varepsilon \leq \gamma_{\varepsilon - \nu_\gamma}$,

where $\gamma_{\varepsilon-\nu_\gamma}$ denotes the optimal chance-constrained solution when the size of the violation set is $\varepsilon - \nu_\gamma$. The number z, which denotes the number of values $\hat{\gamma}^{(i)}$ that need to be discarded, is a function of N and of the confidence parameters β_γ and ε and ν_γ. Further details as to how to how the choice of parameter values can affect the quality assessment of the approximate abstraction can be found in [1]. With Algorithm 10, we are now able to have a quantitative assessment of the quality of an abstracted model $\tilde{\mathcal{M}}_\theta$ used to approximate $\mathcal{M}_\theta$. The framework introduced in [1] and presented here complements the multifidelity work of [93, 94], as it allows us to define a more formal relationship between the LF and HF models through the choice of distance metric $d_T(\cdot, \cdot)$.

So far in this chapter we have introduced the multifidelity counterparts to ABC and ABC-SMC in MF-ABC and MF-ABC-SMC and we have also introduce a way to formalise the relationship between HF and LF models with the introduction of an approximate abstraction method. By considering the Bayesian SMC framework introduced earlier in this chapter, we can integrate these multifidelity techniques and alternative SMC technique to introduce Multifidelity Bayesian verification.

5.3 Multifidelity Bayesian Verification

In this section, we integrate multifidelity methods and SMC methods to reduce the workload of the current ABC-(SMC)^2 [81] work by simultaneously reducing the total number of simulations required in the framework and also lessen the computational burden of those simulations. Specifically, we will work with integrating MF-ABC-SMC [94] (Algorithm 9) with Bayesian SMC [120] (Algorithm 6). As mentioned before, the multifidelity framework is reliant on the LF model being an accurate approximation of the HF model. Whilst these multifidelity models can be somewhat aligned, we aim to formalise this relationship with the use of the approximate abstraction technique [1] introduced in Algorithm 10. With Algorithm 10, we are now able to have a quantitative assessment of the quality of an abstracted model $\tilde{\mathcal{M}}_\theta$ used to approximate $\mathcal{M}_\theta$. Using MF-ABC and the work of [1] as inspiration, we will explore utilising multifidelity methods in SMC (specifically, BSMC) and see what insights we can gain from performing this multifidelity SMC at different stages of the sequential Monte Carlo algorithm. We will compare the credibility calculation $\mathcal{C}_{\text{mf}}$ and approximate parameter synthesis results by performing this multifidelity SMC approach at different stages of the algorithm and compare these results with those obtained by performing BSMC on the HF model, with parameters sampled from the posterior. We will first discuss this multifidelity SMC approach.

Multifidelity Bayesian Statistical Model Checking

Inspired by the multifidelity approach, in Algorithm 11 we introduce a multifidelity Bayesian statistical model checking (MF-BSMC) algorithm. Whilst we can perform BSMC separately for the HF and LF models, if we perform BSMC simultaneously for the multifidelity model, we are able to seamlessly quantify the assessment via [1] of the LF model as an approximate abstraction of the HF model. Let us recall the MF-ABC-SMC technique in Algorithm 9 where the HF model is denoted as $\mathcal{M}_\theta$ with respective simulations y generated via $y \sim p(y \mid \theta)$ and we denote the LF model as $\tilde{\mathcal{M}}_\theta$ with respective simulations generated by $\tilde{y} \sim \tilde{p}(\tilde{y} \mid \theta)$. Both models are parametrised by the same set of parameters θ, but both models may lead to different satisfaction probabilities, $\Lambda_\phi(\theta)$. Let us denote the property we wish to verify on the HF model as ϕ (with respective threshold probability ζ), which we assume is adequate to define a property on the LF model. If not, we assume that we can define a property $\tilde{\phi}$ that is the LF equivalent to the property ϕ, however, in the work considered in this thesis, we set $\tilde{\phi} = \phi$. We denote the HF satisfaction probability estimate as $\hat{\Lambda}_\phi(\theta)$ with

the corresponding LF satisfaction probability estimate as $\tilde{\Lambda}_\phi(\theta)$, where we omit the $\tilde{\cdot}$ notation for brevity in the LF model.

The MF-BSMC method follows closely to the BSMC (Algorithm 6) algorithm, where we assume in this case that we are interested in the same coverage (c), half-interval width size (ℓ) and prior Beta distribution parameters (α, β) for both models. By storing the information generated from both the HF and LF models, we can compute a randomised quality assessment up to a level $1 - \varepsilon$ of the LF model compared to the HF model (Algorithm 10). This results in an algorithm that performs BSMC on both the LF and HF models, returning $\{\hat{\Lambda}_\phi(\theta), \hat{\Lambda}^L_\phi(\theta), \hat{\Lambda}^U_\phi(\theta)\}$ for the HF model, $\{\tilde{\Lambda}_\phi(\theta), \tilde{\Lambda}^L_\phi(\theta), \tilde{\Lambda}_\phi(\theta)^U\}$ for the LF model and $\hat{\gamma}_\varepsilon$, an assessment of the quality of abstraction the LF model $\tilde{p}(\tilde{y} \mid \theta)$ is of the high fidelity model, $p(y \mid \theta)$. We note here that the number of simulations required to estimate the HF satisfaction probability estimates, n, and the number of simulations required for the LF satisfaction probability estimates, $\tilde{n}$, may differ as these are not determined beforehand. As a result, we set $n_\gamma = \min\{n, \tilde{n}\}$ for the approximate abstraction $\hat{\gamma}_\varepsilon$ and cease to store the simulations within the algorithm if $n \neq \tilde{n}$ at any point - that is, if n and $\tilde{n}$ diverge at any point during the algorithm, we cease to calculate $\hat{\gamma}^{(i)} = d_T(\tilde{y}, y)$ and set n_γ to be the minimum of the two simulations.

5.3.1 MF-ABC-(SMC)^2 Algorithm

In this alternative Bayesian verification technique, rather than naively performing rigorous SMC techniques for every parameter sampled, we explore when to perform SMC and what are the ramifications of performing SMC at different stages of the algorithm. The output of Algorithm 9 are multifidelity weighted samples $\{\theta^{(i)}_M, w^{(i)}_M\}^N_{i=1}$. Given that the multifidelity weights $w^{(i)}_M$ have a non-negative probability of being negative, these weighted samples cannot be interpreted as a weighted sample from the ABC posterior, $\pi^M_{ABC}(\theta \mid y_{obs})$. However, we are able to produce an estimate of a $\pi^M_{ABC}(\theta \mid y_{obs})$-integrable function $g : \Theta \to \mathbb{R}$, such that

$$\mathbb{E}^{h_M}_{ABC}(g(\theta) \mid y_{obs}) \approx \frac{\sum^N_{i=1} w^{(i)}_M g(\theta^{(i)}_M)}{\sum^N_{j=1} w^{(j)}_M}. \tag{5.18}$$

Recalling the definition of the credibility calculation, $\mathcal{C} = P(\mathcal{S} \models \phi \mid y_{obs}) = \mathbb{E}_\pi\left[\mathbb{1}_{\Theta_\phi}(\theta)\right]$ and noting the fact that the set Θ_ϕ is defined as $\Theta_\phi = \{\theta \in \Theta : \Lambda_\phi(\theta) \sim \zeta\}$, we can set $g(\theta) = \mathbb{1}_{\Theta_\phi}(\theta) = \mathbb{1}_{\Lambda_\phi(\theta)\sim\zeta}(\theta)$, and using the multifidelity weights we

Algorithm 11 Multifidelity Bayesian Statistical Model Checking (MF-BSMC)

Input:

- Parametrised HF (HF) model $p(y \mid \theta)$ and LF (LF) model $\tilde{p}(\tilde{y} \mid \theta)$
- Property ϕ defined over HF and LF model, encoded as temporal logic formulae
- Half-interval width size $\ell \in (0, \frac{1}{2})$
- Interval coverage parameter $c \in (\frac{1}{2}, 1)$
- Prior Beta distribution with parameters α, β
- Parameter to obtain a $(1-\varepsilon)$-level approximate abstraction, ε

Output:

- Posterior mean estimates for HF satisfaction probability $\hat{\Lambda}_\phi(\theta)$ and LF sat prob $\tilde{\Lambda}_\phi(\theta)$
- Interval of width 2ℓ for HF model $(l_0, l_1) = (\hat{\Lambda}^L_\phi(\theta), \hat{\Lambda}^U_\phi(\theta))$, and for LF model $(\tilde{l}_0, \tilde{l}_1) = (\tilde{\Lambda}^L_\phi(\theta), \tilde{\Lambda}^U_\phi(\theta))$ with posterior probability at least c
- Measure of quality of the LF model compared to HF, $\hat{\gamma}_\varepsilon$

Initialise $n = 0$, $\tilde{n} = 0$, total number of simulations
Initialise $v = 0$ and $\tilde{v} = 0$, number of traces satisfying, $y \models \phi$ and $\tilde{y} \models \phi$ respectively
Initialise $n_\gamma = 0$
while $\Upsilon < c$ or $\tilde{\Upsilon} < c$ **do**
  if $\Upsilon < c$ **then**
    Generate trace $y \sim p(y \mid \theta)$ from $\mathcal{M}_\theta$
    $n \leftarrow n + 1$
    if $y \models \varphi$ **then**
      $v \leftarrow v + 1$
    end if
    Calculate $\hat{\Lambda}_\phi(\theta) = (v + \alpha)/(n + \alpha + \beta)$
    Set $(l_0, l_1) = (\hat{\Lambda}_\phi(\theta) - \ell, \hat{\Lambda}_\phi(\theta) + \ell)$
    if $l_1 > 1$ **then**
      $(l_0, l_1) = (1 - 2\ell, 1)$
    else if $l_0 < 0$ **then**
      $(l_0, l_1) = (0, 2\ell)$
    end if
    Compute posterior probability $\Upsilon = F_{(v+\alpha, n-v+\beta)}(l_1) - F_{(v+\alpha, n-v+\beta)}(l_0)$
  end if
  if $\tilde{\Upsilon} < c$ **then**
    Generate trace $\tilde{y} \sim \tilde{p}(\tilde{y} \mid \theta)$ from $\tilde{\mathcal{M}}_\theta$
    $\tilde{n} \leftarrow \tilde{n} + 1$
    if $\tilde{y} \models \varphi$ **then**
      $\tilde{v} \leftarrow \tilde{v} + 1$
    end if
    Calculate $\tilde{\Lambda}_\phi(\theta) = (\tilde{v} + \alpha)/(\tilde{n} + \alpha + \beta)$
    Set $(\tilde{l}_0, \tilde{l}_1) = (\tilde{\Lambda}_\phi(\theta) - \ell, \tilde{\Lambda}_\phi(\theta) + \ell)$
    if $\tilde{l}_1 > 1$ **then**
      $(\tilde{l}_0, \tilde{l}_1) = (1 - 2\ell, 1)$
    else if $\tilde{l}_0 < 0$ **then**
      $(\tilde{l}_0, \tilde{l}_1) = (0, 2\ell)$
    end if
    Compute posterior probability $\tilde{\Upsilon} = F_{(\tilde{v}+\alpha, \tilde{n}-\tilde{v}+\beta)}(\tilde{l}_1) - F_{(\tilde{v}+\alpha, \tilde{n}-\tilde{v}+\beta)}(\tilde{l}_0)$
  end if
  while $n = \tilde{n}$ **do**
    $i \leftarrow i + 1$
    Calculate $\hat{\gamma}^{(i)} = d_T(\tilde{y}, y)$ and store
    $n_\gamma \leftarrow n_\gamma + 1$
  end while
end while
Calculate number of z values of $\hat{\gamma}^{(i)}$, $i = 1, \ldots, n_\gamma$ to discard via [1] with parameter ε
Set

$$\hat{\gamma}_\varepsilon = \max_{i \in \{1,2,\ldots,n_\gamma\} \setminus \{j_1, j_2, \ldots, j_z\}} \hat{\gamma}^{(i)},$$

where $\{j_1, j_2, \ldots, j_z\} \subset \{1, 2, \ldots, n_\gamma\}$ denote the indices of the discarded values of $\hat{\gamma}^{(i)}$
return $\Lambda_\phi(\theta)$ estimates for HF and LF model and $\hat{\gamma}_\varepsilon$

can get a credibility calculation of the form

$$\mathcal{C}_{\text{mf}} = \mathbb{E}^{h_M}_{ABC}(\mathbb{1}_{\Lambda_\phi(\theta)\sim\zeta}(\theta) \mid y_{obs}) \approx \frac{\sum_{i=1}^{N} w_M^{(i)} \mathbb{1}_{\Lambda_\phi(\theta)\sim\zeta}(\theta_M^{(i)})}{\sum_{j=1}^{N} w_M^{(j)}}. \tag{5.19}$$

The satisfaction probabilities associated with $\mathbb{1}_{\Theta_\phi}(\theta)$ are statistically estimated with guarantees via BSMC, which provides a mean $\hat{\Lambda}_\phi(\theta)$ with lower and upper bounds such that $\hat{\Lambda}_\phi(\theta) \in \left[\hat{\Lambda}^L_\phi(\theta), \hat{\Lambda}^U_\phi(\theta)\right]$. Similar to how we performed approximate parameter synthesis in the previous chapter, these lower and upper limits will be chosen to make a decision on whether $\theta \in \Theta_\phi$, depending on the sign of the threshold decision in $\Lambda_\phi(\theta) \sim \zeta$. To improve the scalability of the Bayesian verification framework, we set the number of simulations associated with each sampled parameter to $B_m = 1$ according to [19], which allows us to evaluate the acceptance probability in ABC quickly at the cost of higher variability. We note however by setting B_m to the number of simulations determined by BSMC we can achieve an algorithm similar to ABC-(SMC)^2. With this change, we no longer perform SMC for each sampled point and must make a decision as to when to perform SMC.

In the previous section we introduced the multifidelity SMC technique, MF-BSMC. Given that we are no longer naively performing SMC for each sampled parameter, we need to decide where to input the MF-BSMC algorithm into the MF-ABC-SMC algorithm introduced in Algorithm 9. As mentioned previously, given that the number of simulations has been set to $B_m = 1$, rather than perform extensive SMC on each sampled parameter, we choose when to perform BSMC. The optimal choice would be to perform BSMC on the HF model when we have inferred the posterior probability distribution after M thresholds, $\pi^{h_M}_{ABC}(\theta \mid y_{obs})$, as we are interested in the credibility calculation, $\mathcal{C}_{\text{mf}} = P(\mathcal{S} \models \phi \mid y_{obs})$.

However, we will explore alternative options of when to perform SMC to see if we obtain any further information about the parametrised models $\mathcal{M}_\theta$ and $\tilde{\mathcal{M}}_\theta$, as well as their relationships with the respective property of interest, ϕ. Specifically, we will explore results obtained when we perform SMC on parameters sampled from the posterior distribution, $\pi^{h_M}_{ABC}(\theta \mid y_{obs})$, parameters from the prior $\pi(\theta)$ and when we perform SMC on both the prior $\pi(\theta)$ and posterior $\pi^{h_M}_{ABC}(\theta \mid y_{obs})$. The MF-ABC-(SMC)^2 method that results from integrating MF-ABC-SMC with MF-BSMC is presented in Algorithm 12. To assess the performance of MF-ABC-(SMC)^2 we apply the Bayesian verification scheme on a few case studies.

5.3.2 Case studies

In this section we will explore the multifidelity Bayesian verification scheme, MF-ABC-(SMC)^2, applied on different case studies which have parametrised HF models $\mathcal{M}_\theta$ and LF (LF) models $\tilde{\mathcal{M}}_\theta$ with a varying number of parameters, $\theta \in \mathbb{R}^{N_R}$, where N_R varies over different case studies. We will use pCTMCs as HF models $\mathcal{M}_\theta$, and pSDEs, namely the chemical Langevin equations (CLE), as LF models $\tilde{\mathcal{M}}_\theta$. The CLE is known to be a good approximation for a CTMC when the CRN has a large number of molecules [47] (see Section 2.4.2), as in our case studies. With this large number of molecule counts, pCTMCs suffer from an increased computational burden, thus utilising pSDEs is advantageous: the states of a CLE are real variables $\tilde{x}(t) \in \mathbb{R}^{N_S}$, as opposed to discrete molecule counts in a pCTMC $x(t) \in \mathbb{N}^{N_S}$.

Drawing simulations $y \sim p(y \mid \theta)$ from $\mathcal{M}_\theta$ is equivalent to drawing a sample from the SSA algorithm (Algorithm 1) up to a time T, whereas drawing $\tilde{y} \sim \tilde{p}(\tilde{p} \mid \theta)$ from $\mathcal{M}_\theta$ is equivalent to drawing samples traces from the CLE given in Equation (2.15) up to a time T, where we utilise the Euler-Maruyama scheme to integrate the SDEs, with a time step Δt chosen subject to the case study. Recall that Δt is a time-discretisation of the simulation time that needs to be chosen carefully to balance the trade-off between the accuracy of the CLE as an approximation to the CTMC, and the computational cost of generating the simulations. Choosing a small value of Δt would lead to a very detailed but computationally costly simulation of the underlying CRN, while choosing large values of Δt would lead to faster simulations but may lead to less accurate simulations. Work has been pursued to develop methods to choose these step sizes to optimise this trade-off, such as the work considered in [61]. We denote the observed data as y_{obs}, generating the data for y_{obs} using true parametrisations $\theta_\star$ and set $\tilde{y}_{obs} = y_{obs}$. The observed data is an average of a finite number of traces generated from the true model $\mathcal{M}_{\theta_\star}$, sampled at a total number of $N_{\mathcal{O}}$ discrete points in time, $\bar{t}_i$, $i = 1, 2, \ldots, N_{\mathcal{O}}$.

We deploy multifidelity methods in SMC, using the approximate abstraction presented above and in [1], to measure how well the LF model acts as an approximation for the HF model across the parameter space. This choice of approximate abstraction metric requires no a-priori assumptions on the underlying LF or HF models and only requires the ability to generate stochastic simulations of the LF and HF model, making it generally applicable to a variety of stochastic systems. By incorporating it into the simulation-based SMC algorithm, we are able to output a quantity that measures how well the LF model performs as an approximation for the HF model as we estimate the probability of satisfying a property on both the LF and HF models

Table 5.1. Runtimes for MF-ABC-(SMC)^2 and MF-BSMC.

	Times [seconds]		
	MF-BSMC		MF-ABC-(SMC)^2
Case	$\pi(\theta)$	$\pi_{ABC}^{h_M}(\theta \mid y_{obs})$	
Production-degradation	1916	7172	9122
SIR	2913	6107	9058

over the parameter space Θ. Alternative metrics do not benefit from the simulation-based metric of [1] because they often have restrictive assumptions on the dynamics of the underlying abstraction and system of interest, see [1] for details.

That is, y_{obs} consists of $N_{\mathcal{O}}$ observations of the state vector at discrete points in time $\bar{t}_1, \bar{t}_2, \ldots, \bar{t}_{N_{\mathcal{O}}}$ such that $y_{obs} = \tilde{y}_{obs} = [y_{obs}(\bar{t}_1), y_{obs}(\bar{t}_2), \ldots, y_{obs}(\bar{t}_{N_{\mathcal{O}}})]^\top$. We note that Δt is a time discretisation used to numerically solve the SDE, whereas $\bar{t}_i$ are observation times for y_{obs} - the discretisation Δt is chosen such that we are able to observe the state of the pSDEs at times $\bar{t}_i$, that is, $\bar{t}_i = j\Delta t$ for some $j = 1, 2, \ldots, T/\Delta t$ where $i \neq j$. Whilst the model used to generate the data y_{obs} is intrinsically noisy or stochastic, to emulate observation error in some case studies we perturb the observations further $y_{obs}(\bar{t}_k)$ with noise, given by σ^2, $y_{obs}(\bar{t}_k) \sim \mathcal{N}(y_{obs}(\bar{t}_k), \sigma^2)$. Note for generated simulations, $y \sim p(y \mid \theta)$ or $\tilde{y} \sim \tilde{p}(\tilde{y} \mid \theta)$, to calculate $d(y, y_{obs})$ or $\tilde{d}(\tilde{y}, \tilde{y}_{obs})$ we take the values of y and $\tilde{y}$ at the time points at which we take the observations, given by $\bar{t}_i$ We verify the same property, ϕ, on both the HF and LF model, where the property is defined in CSL. We emphasise yet again that with so few observation traces, SMC based directly on observation traces would not be possible. In these case studies, we work with a fixed number of M thresholds for the algorithm to iterate through, that is, $m = 0, 1, \ldots, M$, and we set the initial thresholds $h_0 = \infty$ and $\tilde{h}_0 = \infty$ such that we accept every sample generated from the prior. Rather than inputting a predetermined sequence of neighbourhoods Ω_{h_m} and $\tilde{\Omega}_{\tilde{h}_m}$, we set h_{m+1} and $\tilde{h}_{m+1}$ to be equal to the average distances of the accepted sampled parameters in threshold h_m, such that $h_{m+1} < h_m$ and we will terminate the algorithm if this is not the case. The runtimes for the MF-ABC-(SMC)^2 case studies are given in Table 5.1, which include the time taken to perform MF-BSMC for samples generated from the prior and the posterior. We now turn our attention to the first case study, a simple production-degradation model.

Algorithm 12 Multifidelity ABC-(SMC)^2 (MF-ABC-(SMC)^2)

Input:

- Observed data $y_{obs}(= \tilde{y}_{obs})$
- Prior distribution $\pi(\theta)$, HF model $p(y \mid \theta)$ and LF model $\tilde{p}(\tilde{y} \mid \theta)$
- Initial importance distribution $\hat{r}_0(\theta)$ (commonly set to $\hat{r}_0(\theta) = \pi(\theta)$)
- Sequence of high fidelity neighbourhoods $\Omega_{h_m} = \{y \in \mathcal{Y} \mid d(y, y_{obs}) \leq h\}$ and LF neighbourhoods $\tilde{\Omega}_{\tilde{h}_m} = \{\tilde{y} \in \tilde{\mathcal{Y}} \mid \tilde{d}(\tilde{y}, \tilde{y}_{obs}) \leq \tilde{h}\}$ with respective kernels $\mathbb{1}_{\Omega_h}(y)$ and $\mathbb{1}_{\tilde{\Omega}_{\tilde{h}}}(\tilde{y})$
- Initial continuation probability function $\eta_0 = \eta_0(\theta, \tilde{y}) = 1$
- Lower bounds on continuation probabilities, $\eta_1^L, \eta_2^L \in (0, 1)$
- Sequence of perturbation kernels $F_m(\theta|\theta^*)$
- Monte Carlo sample size $N > 0$
- Property defined on both HF and LF models, ϕ
- Half-interval width size $\ell \in (0, \frac{1}{2})$
- Interval coverage parameter $c \in (\frac{1}{2}, 1)$
- Prior Beta distribution with parameters α, β
- Parameter ε to obtain $(1 - \varepsilon)$-level approximate abstraction

Output:

- $\{\theta^{(i)}, w^{(i)}, \}_{i=1}^N$, accepted parameters with corresponding weights

for $m = 0, \ldots, M$: **do**
  Produce $\{\theta_m^{(i)}, w_m^{(i)}\}_{i=1}^N$ and store information required to estimate optimal continuation probabilities (Appendix B.2) from MF-ABC-IS algorithm (Algorithm 8) with inputs $\left(y_{obs}, \tilde{y}_{obs}, \Omega_{h_m}, \tilde{\Omega}_{\tilde{h}_m}, \pi(\theta), p(y \mid \theta), \tilde{p}(\tilde{y} \mid \theta), \eta_m(\theta, \tilde{y})\right)$ setting $\hat{q} = \hat{r}_m(\theta)$
  Set $\hat{w}_m^{(i)} = |w_m^{(i)}|$ for $i = 1, \ldots, N$
  Define $\hat{r}_{m+1}$ proportional to r_{m+1} defined in Equation (5.12)
  Calculate (η_1^*, η_2^*) with lower bounds (η_1^L, η_2^L) using importance distribution $r_{t+1}(\theta)$, ABC thresholds h_{m+1} and $\tilde{h}_{m+1}$ with values stored from MF-ABC-IS algorithm at step 2 (Appendix B.2)
  Set $\eta_{m+1}(\theta, \tilde{y}) = \eta_1^* \mathbb{1}_{\tilde{\Omega}_{\tilde{h}_m}}(\tilde{y}) + \eta_2^*(1 - \mathbb{1}_{\tilde{\Omega}_{\tilde{h}_m}}(\tilde{y}))$
  if $(m = 0) \cap$ {Perform MF-BSMC for $\pi(\theta)$ } **then**
    Calculate $\{\hat{\Lambda}_\phi(\theta_0^{(i)}), \tilde{\Lambda}_\phi(\theta_0^{(i)}), \hat{\gamma}_\varepsilon^{(i)}\}_{i=1}^N$ from MF-BSMC algorithm (Algorithm 11)
  end if
end for
if {Perform MF-BSMC for $\pi_{ABC}^{h_M}(\theta \mid y_{obs})$ } **then**
  Calculate $\{\hat{\Lambda}_\phi(\theta_M^{(i)}), \tilde{\Lambda}_\phi(\theta_M^{(i)}), \hat{\gamma}_\varepsilon^{(i)}\}_{i=1}^N$ from MF-BSMC algorithm (Algorithm 11)
end if
return $\{\theta_M^{(i)}, w_M^{(i)}\}_{i=1}^N$

Production-degradation model.

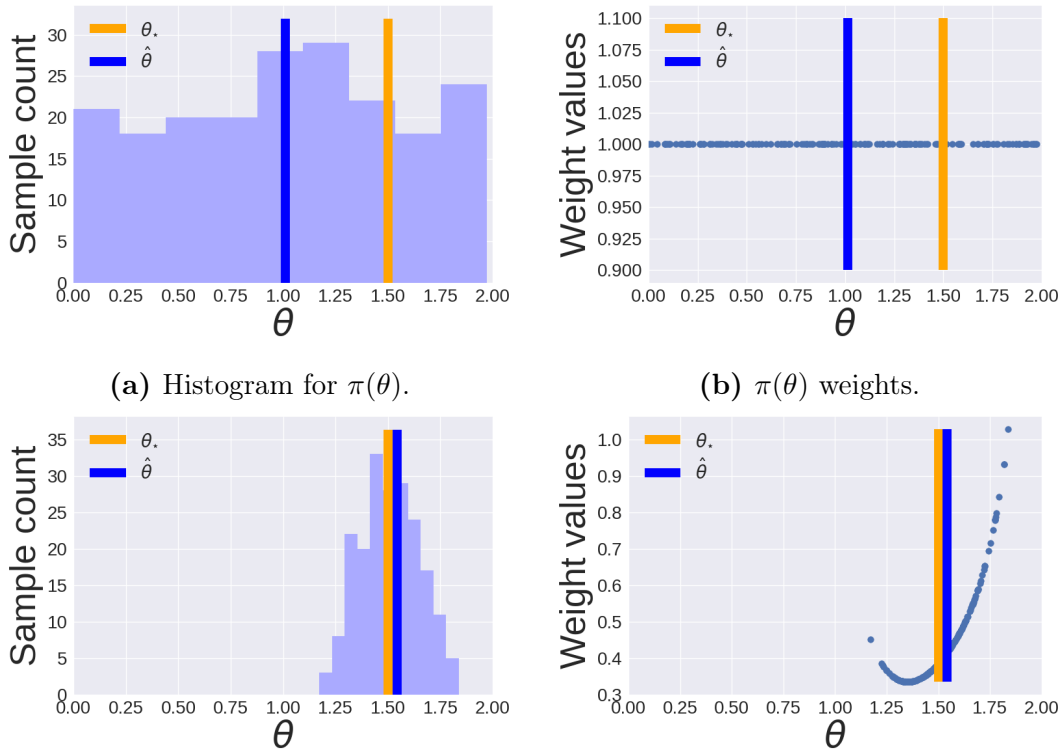
Here we consider a very simple case study to investigate how the MF-ABC-(SMC)^2 performs on a simple model with one parameter, specifically the production-degradation model. We work with a biochemical system that consists of a single chemical species X that involves production and degradation reactions of the form

$$\emptyset \xrightarrow{k_1} X \text{ and } X \xrightarrow{k_2} \emptyset, \tag{5.20}$$

that is, X is produced at a rate $k_1 \in \mathbb{R}_{\geq 0}$, the parameter we wish to infer $\theta = k_1$, and X is degraded at a fixed rate of $k_2 = 0.01$. The propensity functions are given by $a_1(X(t)) = k_1$ and $a_2(X(t)) = k_2 X(t)$ where $X(t)$ is the number of X molecules at time t. The associated stoichiometric vectors are given by $\upsilon_1 = 1$ and $\upsilon_2 = -1$. For the production-degradation model we obtain the following CLE,

$$d\tilde{x}(t) = (k_1 - k_2\tilde{x}(t))dt + \sqrt{k_1}\,\mathrm{d}W^{(1)}(t) - \sqrt{k_2\tilde{x}(t)}\,\mathrm{d}W^{(2)}(t), \tag{5.21}$$

where $W^{(i)}(t)$, $i = 1, 2$ are independent scalar Wiener processes, and we chose a time discretisation of $\Delta t = 0.5$ to solve this SDE using the Euler-Maruyama scheme. We set the time-horizon $T = 500$ such that $t \in [0, 500]$, the initial state is given as $X(0) = A_0 = 50$, and we generate our observation data by averaging five simulations to generate $y_{obs} = \tilde{y}_{obs}$ using a true parameter instantiation $\theta_\star$ and we take observations every $\bar{t}_i = 20{\cdot}i$ time units. In this case study we do not consider observation error and set $\sigma = 0$. The property we wish to verify is $\bar{\phi} = P_{\geq 0.7}(\Box^{[200,500]}\,(120 \leq X \leq 180))$, that is, whether with a probability of at least 0.7, the number of X molecules remain between $120 \leq X \leq 180$ for $t \in [200, 500]$. This is equivalent to checking $\phi = P_{<0.3}(true\ U^{[200,500]}\,(X < 120 \cup X > 180)$ and this is the property that we check for, thus, $\zeta = 0.3$. The true model has parametrisation $\theta_\star = k_1^\star = 1.5$ and by running MF-BSMC with coverage $c = 0.95$ and width parameter $\ell = 0.01$ on this true parametrisation, we obtain $\hat{\Lambda}_\phi(\theta_\star) = 0.238791 \in [0.228791, 0.248791]$ which required a total number of $n = 6979$ simulations from the HF model while for the LF model $\tilde{\Lambda}_\phi(\theta_\star) = 0.258822 \in [0.248822, 0.268822]$, which required $\tilde{n} = 7366$ simulations of the LF model. We present the satisfaction probabilities for both the HF and LF models to highlight that there is a discrepancy and in general $\hat{\Lambda}_\phi(\theta) \neq \tilde{\Lambda}_\phi(\theta)$. The number of Monte Carlo samples to be drawn is set to $N = 200$ and we decide on iterating the algorithm for $M = 5$ thresholds. We let the prior $\pi(\theta)$ be a non-informative uniform, $\theta \sim \mathcal{U}(0.0, 2.0)$, over the parameter space $\Theta = [0.0, 2.0]$. We first present the results of the inference technique, given in Figure 5.2.



(a) Histogram for $\pi(\theta)$.

(b) $\pi(\theta)$ weights.

(c) Histogram for $\pi^{h_M}_{ABC}(\theta \mid y_{obs})$.

(d) $\pi^{h_M}_{ABC}(\theta \mid y_{obs})$ weights.

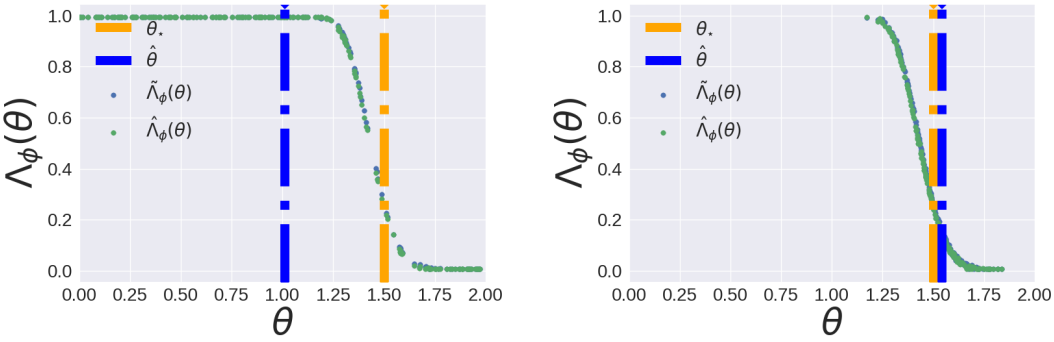
Figure 5.2. Inference results (5.2a and 5.2c) and weights (5.2b and 5.2d) for sampled parameters, obtained from the inference part of the multifidelity Bayesian verification approach for the production-degradation case study for the set prior $\pi(\theta)$ and the obtained posterior $\pi^{h_M}_{ABC}(\theta \mid y_{obs})$. We note that $\theta = k_1$, the inferred posterior mean is denoted by $\hat{\theta}$ while the true parameter is denoted by $\theta_\star$.

From Figure 5.2 we obtain a posterior mean of $\hat{\theta} = 1.54$ and we note that the weights are strictly positive, which implies that for most cases the LF model comes to the same conclusion as the HF model. We estimate the satisfaction probabilities for the HF and LF models using MF-BSMC (Algorithm 11) where the parameters are the multifidelity weighted parameters from the MF-ABC-SMC part of the algorithm, $\{\theta^{(i)}_M, w^{(i)}_M\}^N_{i=1}$, and these estimated satisfaction probabilities are presented in Figure 5.3. We compute a credibility calculation of $\mathcal{C}_{\text{mf}} = 0.63100$ using $\hat{\Lambda}_\phi(\theta)$ for the HF model and weights generated from the posterior. As we increase the accuracy of the inference technique, the posterior should converge to the true parameter value $\theta_\star$ and we expect $\mathcal{C}_{\text{mf}}$ to converge to 1.

From the satisfaction probabilities given in Figure 5.3, we can see that the success of the Bayesian verification technique not only hinges on the accuracy and variance

of the inference part, but on the location of the posterior distribution with respect to the satisfaction probability $\Lambda_\phi(\theta)$. To see this, consider Figure 5.3a and the estimated mean values of the multifidelity satisfaction probabilities. In a region around $\theta_\star = 1.5$ the gradient of the satisfaction function is at its steepest, thus a small change in the value of θ will lead to a large change in the values of $\hat{\Lambda}_\phi(\theta)$ and $\tilde{\Lambda}_\phi(\theta)$: hence, inferring parameters with a very high degree of accuracy is of utmost importance. If we were to consider a parameter value of $\theta = 0.25$ say, then from the shape of the satisfaction functions, this would give us a lot more flexibility with respect to the accuracy of the inference technique - specifically, having a larger variance on the inferred posterior probability distribution would have less of an affect on the credibility calculation (as opposed to when $\theta = \theta_\star$). Although the mean satisfaction probabilities in Figure 5.3 look to be in agreement, from Figure 5.4, we can see that there are regions of the parameter space when the LF model is a more accurate abstraction for the HF model, recalling that the approximate abstraction metric is given by

$$\hat{\gamma}_\varepsilon = \max_{i \in \{1,2,\ldots,N\} \setminus \{l_1, l_2, \ldots, l_k\}} \hat{\gamma}^{(i)} \text{ where } \hat{\gamma}^{(i)} = d_T(y^{(i)}, \tilde{y}^{(i)}) = \sup_{t \in [t_0, T]} \|y^{(i)}(t) - \tilde{y}^{(i)}(t)\|,$$



(a) $\pi(\theta)$.

(b) $\pi^{h_M}_{ABC}(\theta \mid y_{obs})$.

Figure 5.3. Estimated satisfaction probabilities obtained for both HF ($\hat{\Lambda}_\phi(\theta)$, denoted by blue points) and LF models ($\tilde{\Lambda}_\phi(\theta)$, denoted by green points), obtained from parameters $\theta = k_1$ sampled from set prior and obtained posterior distributions; once again the true parameter ($\theta_\star = 1.5$) and the posterior mean ($\hat{\theta} = 1.54$) are highlighted.

The effect of negative weights

In the MF-ABC framework that we are now working in, there is a non-negative probability of obtaining negative weights. We now consider the case when negative

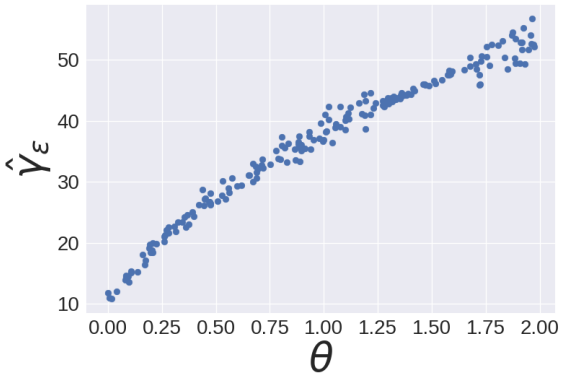


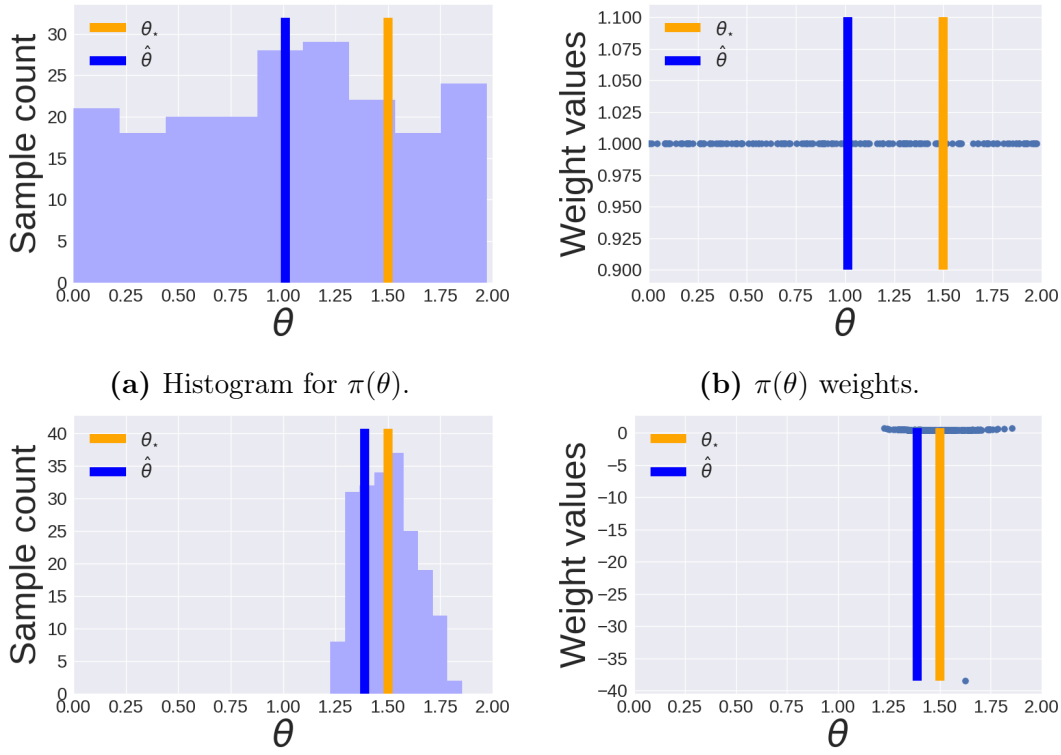
Figure 5.4. Production-degradation case study: assessment of the quality of the LF model $\tilde{p}(\theta \mid y_{obs})$ abstraction versus the HF model $p(y \mid \theta)$ where $\theta = k_1$ and $\hat{\gamma}_\epsilon$ is the estimated approximate abstraction metric.

weights are present in the sample and what effect they have on the results obtained from the MF-ABC-(SMC)^2 algorithm.

From Figure 5.5 we see that the algorithm infers a posterior mean of $\hat{\theta} = 1.3992$ and we see that the weights are mainly positive, meaning that the LF model comes to the same acceptance/rejection decision as the HF model. However, we note the magnitude of the singular negative weight - recalling the fact that negative weights arise when $\tilde{y} \in \tilde{\Omega}_{\tilde{h}}$ but $y \notin \Omega_h$. The magnitude of this negative weights influences the credibility calculation $\mathcal{C}$ as it influences the posterior mean parameter $\hat{\theta}$ and in turn $\mathcal{C}_{\text{mf}}$. As we can see from Figure 5.6, the negative weight skews the posterior mean and the resulting credibility calculation, which is given by $\mathcal{C}_{\text{mf}} = 0.1935$. If we were to only consider the positive weights (that is, disregarding the only negative weight, not taking the absolute values), we obtain a posterior mean of $\hat{\theta}^+ = 1.5188$, which is much closer to the true value $\theta_\star$ and the resulting credibility calculation increases to $\mathcal{C}^+_{\text{mf}} = 0.5726$ when considering only the positive weights. From Figure 5.5, a credibility calculation of around 0.5 would be expected as the sampled parameters look to be equally distributed around either side of the probability threshold, $\zeta = 0.3$. When working with multifidelity methods, care must be taken when negative weights arise as the negative weights can influence the inferred parameters and the resulting credibility calculation across the posterior distribution $\pi^{h_M}_{ABC}(\theta \mid y_{obs})$. Whilst these negative weights could be thought of as penalising a region of the parameter space where $\tilde{y} \in \tilde{\Omega}_{\tilde{h}_m}$ but $y \notin \Omega_{h_m}$, in the MF-ABC-(SMC)^2 scheme the importance sampling distribution $\hat{r}_m(\theta)$ works with an absolute value of these negative weights leading to additional density in regions where $w^{(i)}_m < 0$ is more likely. However, if the magnitude of these weights are large, the sampling scheme can lead to oversampling regions in

the parameter space when the weight is negative which could further influence $\hat{\theta}$ and $\mathcal{C}_{\text{mf}}$.

We plot the estimated satisfaction probabilities for the weighted sample that contains negative weights in Figure 5.6, where once again the success of the Bayesian verification technique hinges on both the accuracy of the inference technique, and the location of the parameter samples $\theta_M^{(i)}$ with respect to the shape of the satisfaction probability, $\Lambda_\phi(\theta)$. We note the location of $\hat{\theta}$ and compare this with $\hat{\theta}^+$, where in Figure 5.6b the location of $\hat{\theta}^+$ is indistinguishable from the true parameter $\theta_\star$.



(a) Histogram for $\pi(\theta)$.

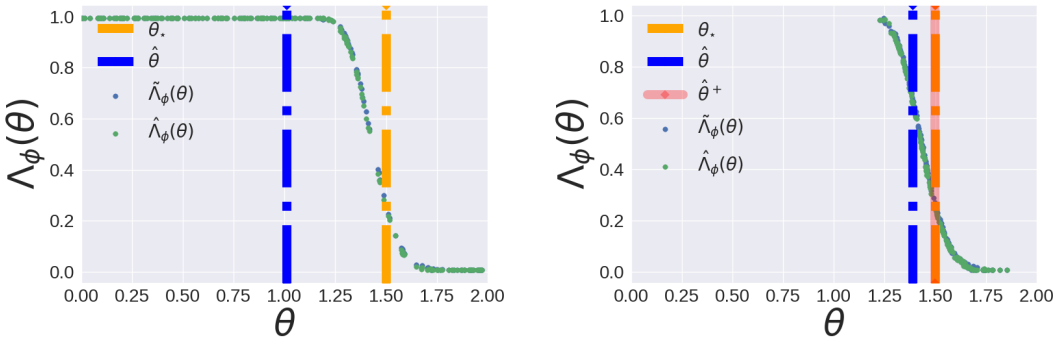
(b) $\pi(\theta)$ weights.

(c) Histogram for $\pi_{ABC}^{h_M}(\theta \mid y_{obs})$.

(d) $\pi_{ABC}^{h_M}(\theta \mid y_{obs})$ weights.

Figure 5.5. Inference results (including negative weights) (5.5a and 5.5a) and weights (5.5b and 5.5b) for sampled parameters, obtained from the Multifidelity Bayesian Verification approach for the production-degradation case study for the prior ($\pi(\theta)$) and posterior $\pi_{ABC}^{h_M}(\theta \mid y_{obs})$ We note that $\theta = k_1$, and highlight the posterior mean ($\hat{\theta}$) and the true parameter ($\theta_\star$).

From this basic case study, we gain some valuable insight as to how the Bayesian verification technique behaves with the introduction of multifidelity methods and how the framework depends heavily on the shape of the satisfaction function, $\Lambda_\phi(\theta)$, and the location of the true parameter $\theta_\star$.



(a) $\pi(\theta)$. **(b)** $\pi^{h_M}_{ABC}(\theta \mid y_{obs})$.

Figure 5.6. Estimated satisfaction probabilities obtained for both HF ($\hat{\Lambda}_\phi(\theta)$, denoted by blue points) and LF models ($\tilde{\Lambda}_\phi(\theta)$, denoted by green points) with samples generated from the prior and posterior. We note that $\theta = k_1$ and highlight the true parameter ($\theta_\star = 1.5$), the posterior mean ($\hat{\theta} = 1.3992$) and the posterior mean considering only positive weights ($\hat{\theta}^+ = 1.5188$).

SIR model revisited.

In the previous case study we highlighted how the success of the Bayesian verification technique hinges on the shape of the satisfaction function for the LF and HF models. Whilst the multifidelity weight is central to the MF-ABC-(SMC)^2 scheme, we also showed what effect possible negative weights can have on both the inference aspect of the Bayesian verification technique, and the credibility calculation involved. For the above case study we performed MF-BSMC for samples generated by the prior, $\pi(\theta)$, and the posterior $\pi^{h_M}_{ABC}(\theta \mid y_{obs})$.

We now we revisit the SIR case study presented in the previous chapter and introduced for the ABC-(SMC)^2 scheme, but increase the total number of molecules that are present in the system and alter the property of interest to allow for the CLE approximation to be an appropriate LF model. In this case study however, more emphasis is placed on how and when to perform the MF-BSMC technique and what use or insight we can gain from doing so.

This case study is given by

$$S + I \xrightarrow{k_1} 2I, \qquad I \xrightarrow{k_2} R, \tag{5.22}$$

thus the parameters of interest are $\theta = (\theta_1, \theta_2) = (k_1, k_2)$. For the CLE approximation to be accurate, we increase the number of molecules in the system by setting the initial state to $X(0) = x_0 = (S_0, I_0, R_0) = (500, 125, 0)$. The above CRN has

propensity functions $a_1(X(t)) = k_1 S(t) I(t)$ and $a_2(X(t)) = k_2 I(t)$ which have respective stoichiometric vectors $\boldsymbol{\upsilon}_1 = [-1, 1, 0]^\top$ and $\boldsymbol{\upsilon}_2 = [0, -1, 1]^\top$, which leads us to the CLE approximation

$$\begin{aligned}
\mathrm{d}\tilde{x}_1(t) &= -k_1 \tilde{x}_1(t)\tilde{x}_2(t)\,\mathrm{d}t - \sqrt{k_1 \tilde{x}_1(t)\tilde{x}_2(t)}\,\mathrm{d}W^{(1)}(t), \\
\mathrm{d}\tilde{x}_2(t) &= \tilde{x}_2(t)\left(k_1\tilde{x}_1(t) - k_2\right)\mathrm{d}t + \sqrt{\tilde{x}_2(t)}\left(\sqrt{k_1\tilde{x}_1(t)}\,\mathrm{d}W^{(1)}(t) - \sqrt{k_2}\,\mathrm{d}W^{(2)}(t)\right), \\
\mathrm{d}\tilde{x}_3(t) &= \sqrt{k_2\tilde{x}_2(t)}\left(\sqrt{k_2\tilde{x}_2(t)}\,\mathrm{d}t + \mathrm{d}W^{(2)}(t)\right),
\end{aligned} \tag{5.23}$$

where $\tilde{x}(t) = [\tilde{x}_1(t), \tilde{x}_2(t), \tilde{x}_3(t)]^\top = [\tilde{S}(t), \tilde{I}(t), \tilde{R}(t)]^\top$ are the continuous approximations to the state vector $X(t) = [S(t), I(t), R(t)]^\top$. Once again the $W^{(j)}(t)$, $j = 1, 2$ are scalar independent Wiener processes and we solve this SDE numerically using a timestep of $\Delta t = 0.1$.

Given the number of molecules in this current system, we consider an alternative property to verify, specifically $\phi = P_{\geq 0.5}((I > I_0)U^{[7.5,10]}(I > R))$. Informally, the property we wish to verify is as follows. With probability greater than $\zeta = 0.5$, we want the number of I molecules to be greater than the number of R molecules within a time frame $t' \in [7.5, 10]$, and we run the simulation for a total time of $T = 25$, such that the simulation time is $t \in [0, 25]$.

Once again, we set the priors to be independent Uniform priors on $\left[k_i^\perp, k_i^\top\right] = [5 \times 10^{-5}, 0.003]$ and $\left[k_r^\perp, k_r^\top\right] = [0.005, 0.2]$ for θ_1 and θ_2 respectively, where the whole parameter space Θ is defined as $\Theta = \left[k_i^\perp, k_i^\top\right] \times \left[k_r^\perp, k_r^\top\right]$. We run the MF-BSMC algorithm with coverage $c = 0.95$ and width parameter $\ell = 0.01$. The true model we wish to infer is parametrised by $\theta_\star = [k_1^\star, k_2^\star] = [0.002, 0.125]^\top$, and we generate y_{obs} by taking the average of five traces generated from the HF and LF model with this parametrisation, taking observations every $\bar{t}_i = 2.5 \cdot i$, $i = 1, 2, \ldots, 10$ time units. With this parametrisation, the estimated satisfaction probability for the HF model is given by $\hat{\Lambda}_\phi(\theta_\star) = 0.8688 \in [0.8588, 0.8788]$ which required $n = 4370$ simulations to compute while the estimated satisfaction probability for the LF model is given by $\tilde{\Lambda}_\phi(\theta_\star) = 0.993333 \in [0.98, 1]$ which required $\tilde{n} = 148$ simulations. For our multifidelity Bayesian verification scheme, we set the number of Monte Carlo samples to $N = 200$ and the number of thresholds to $M = 5$. As mentioned previously, in this case study we explore how the credibility calculation changes when we perform the SMC at different stages of the MF-ABC-$(\text{SMC})^2$ algorithm. As noted in the previous case study, the success of Bayesian verification hinges on the ability to accurately infer the parameters and we present our inference results in Figure 5.7.

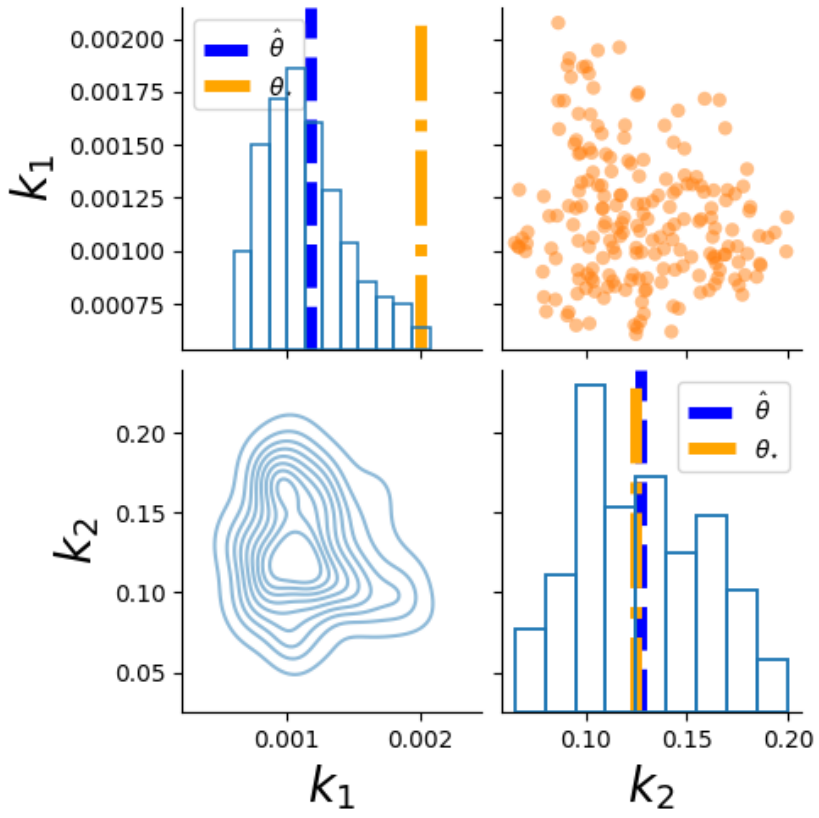
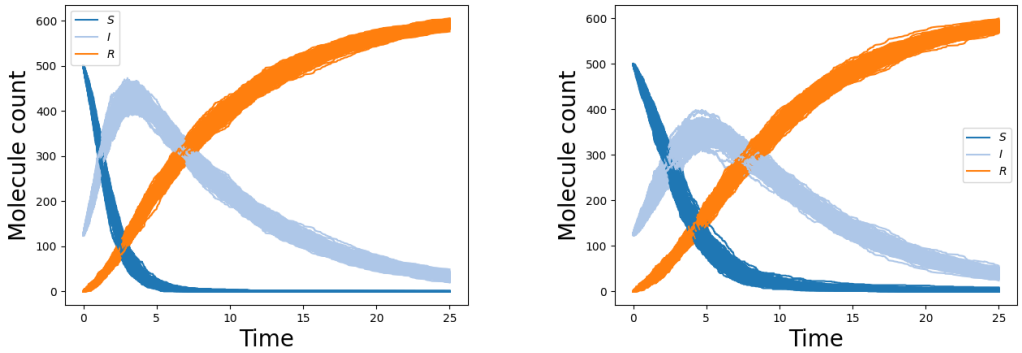


Figure 5.7. Inference results for SIR case study. Diagonal plots show histograms where we highlight the true parameter ($\theta_\star$) and posterior mean ($\hat{\theta}$) of the parameters while the lower left diagonal shows a kernel density estimate (KDE) plot and the upper right plot shows the location of the sampled parameters.

The posterior mean is given by $\hat{\theta} = [0.00118, 0.12713]^\top$ where we can see that the MF-ABC-(SMC)^2 algorithm does a better job at inferring the parameter k_2 compared to inferring k_1. Here, the associated weights are all positive. Simulation-based methods such as ABC suffer from identifiability issues that arise in other inference techniques; matching y or $\tilde{y}$ to observed data y_{obs} works best when unique parametrisations give rise to unique dynamics or simulations. When two or more parametrisations are observationally equivalent, the system is said to be unidentifiable. With the currently observed data points, the ABC algorithm cannot discern the subtleties between the observed data y_{obs} and the generated simulations y, which leads to inaccurate parameter inference. To improve the inference results, one could increase the number of observations over the time horizon, increase the number of observations during the transient phase of the dynamical system, consider alternative distance metrics, $d(\cdot, \cdot)$ or consider an alternative set of summary statistics.



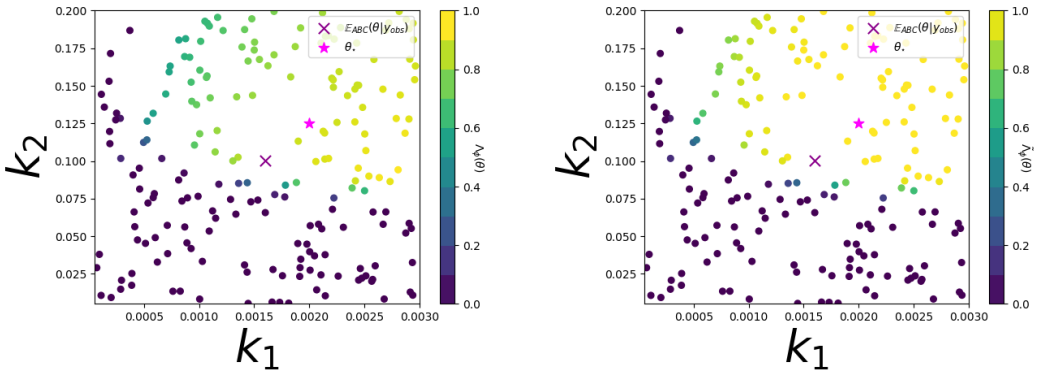
(a) $y \sim p(y \mid \theta_\star)$. **(b)** $y \sim p(y \mid \hat{\theta})$.

Figure 5.8. 200 simulations generated from the true model $\mathcal{M}_{\theta_\star}$ in (5.8a) with parametrisation $\theta_\star = [k_1^\star, k_2^\star]^\top = [0.002, 0.125]^\top$, posterior mean $\mathcal{M}_{\hat{\theta}}$ in (5.8b) with parametrisation $\hat{\theta} = [0.00118, 0.12713]$.

Optimal placement of Statistical Model Checking

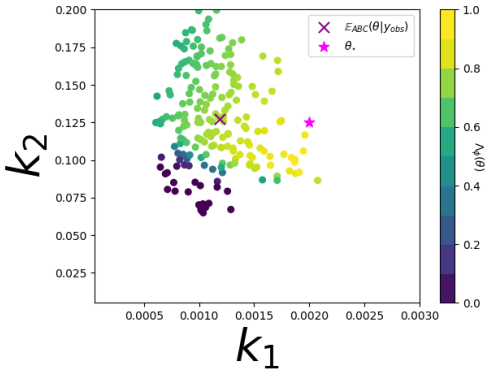
We now consider when to perform SMC and how doing so will affect the outcome of the credibility calculation. Of course, performing SMC on the HF model upon termination of the inference aspect of the algorithm (alternatively, performing SMC for the posterior probability distribution of parameters) would lead to the most accurate credibility calculation, and we will compare our other techniques to this credibility calculation. First of all, we present the MF-BSMC results obtained from running the MF-ABC-(SMC)^2 algorithm, in Figure 5.9.

By performing MF-BSMC both for the prior and posterior distributions, extensively explore the satisfaction probabilities for both the LF and HF models across the parameter space. However, this would require a high number of simulations to be generated for both models before and after the inference aspect of the algorithm. We will now explore what information we gain when SMC is performed is either performed using samples from the prior probability distribution or the posterior probability distribution - that is, at different stages of the algorithm. We note that for the true parametrisation, $\theta_\star$, that the instantiated HF model $\mathcal{M}_{\theta_\star}$ does indeed satisfy the property, as $\hat{\Lambda}_\phi^L(\theta_\star) > \zeta = 0.5$ and thus, $\mathcal{M}_{\theta_\star} \models \phi$. As mentioned before, the most accurate technique would be to estimate the satisfaction probabilities for the HF model with parameter instantiations generated from the posterior probability distribution. For the inference and satisfaction probabilities presented in Figure 5.9, we obtain a credibility calculation of $\mathcal{C}_{\text{mf}} = P(\mathcal{M}_\theta \models \phi \mid y_{obs}) = 0.8085$ for the HF model using samples from the posterior, while the LF model satisfies the

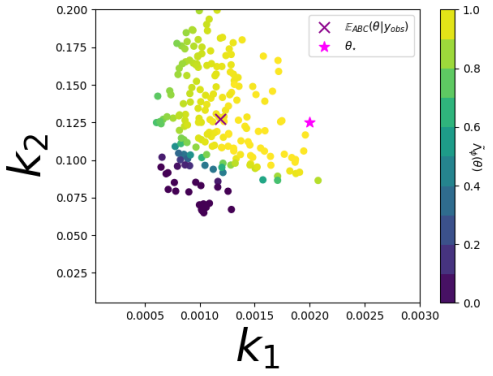


(a) $\hat{\Lambda}_\phi(\theta)$ where $\theta \sim \pi(\theta)$.

(b) $\tilde{\Lambda}_\phi(\theta)$ where $\theta \sim \pi(\theta)$.



(c) $\hat{\Lambda}_\phi(\theta)$ where $\theta \sim \pi^{h_M}_{ABC}(\theta \mid y_{obs})$.



(d) $\tilde{\Lambda}_\phi(\theta)$ where $\theta \sim \pi^{h_M}_{ABC}(\theta \mid y_{obs})$.

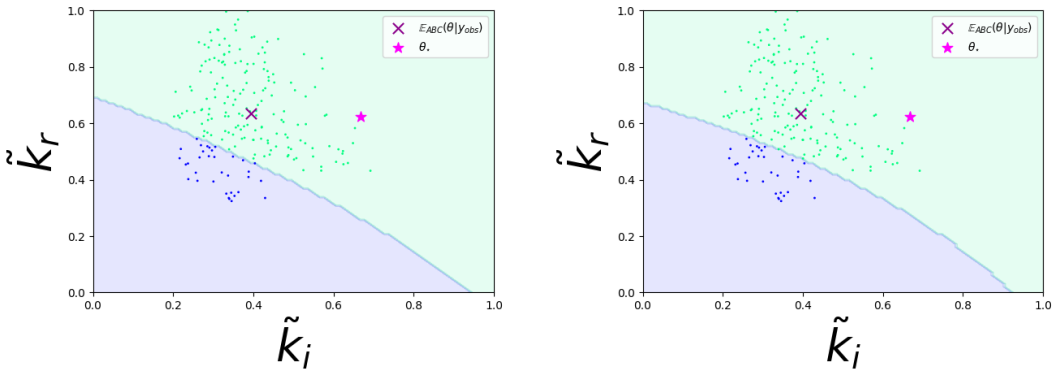
Figure 5.9. Mean estimated satisfaction probabilities for the HF model $\mathcal{M}_\theta$ (5.9a and 5.9c) and the LF model $\tilde{\mathcal{M}}_\theta$ (5.9b and 5.9d) when the parameters are sampled from the prior $\pi(\theta)$ (5.9a and 5.9b), and from the posterior $\pi^{h_M}_{ABC}(\theta \mid y_{obs})$ (5.9c and 5.9d).

property with $\mathcal{C}_{\text{mf}} = P(\tilde{\mathcal{M}}_\theta \models \phi \mid y_{obs}) = 0.8341$. That is, these are the credibility calculations we obtain by performing MF-BSMC after the MF-ABC aspect of the algorithm terminates.

Prior and Posterior Parameter Synthesis

Akin to our work in the ABC-(SMC)^2 framework, presented in the previous chapter, we can use these estimated satisfaction probabilities to perform an approximate parameter synthesis to partition the parameter space to combinations of parameters that satisfy the property Θ_ϕ and those that do not, $\Theta_{\neg\phi}$. We first consider performing this approximate parameter synthesis with samples generated from the posterior probability, $\pi^{h_M}_{ABC}(\theta \mid y_{obs})$, presented in Figure 5.10. If we were to compute the credibility calculations using these approximate parameter synthesis regions, instead of

the estimated satisfaction probabilities $\hat{\Lambda}_\phi(\theta)$ and $\tilde{\Lambda}_\phi(\theta)$, that is, asserting whether samples parameters $\theta \sim \pi^{h_M}_{ABC}(\theta \mid y_{obs})$ are either $\theta \in \Theta_\phi$ or $\theta \in \Theta_{\neg\phi}$, we obtain credibility calculations of $\mathcal{C}_{\text{mf}} = 0.8516$ for the HF model and $\mathcal{C}_{\text{mf}} = 0.8464$ for the LF model.



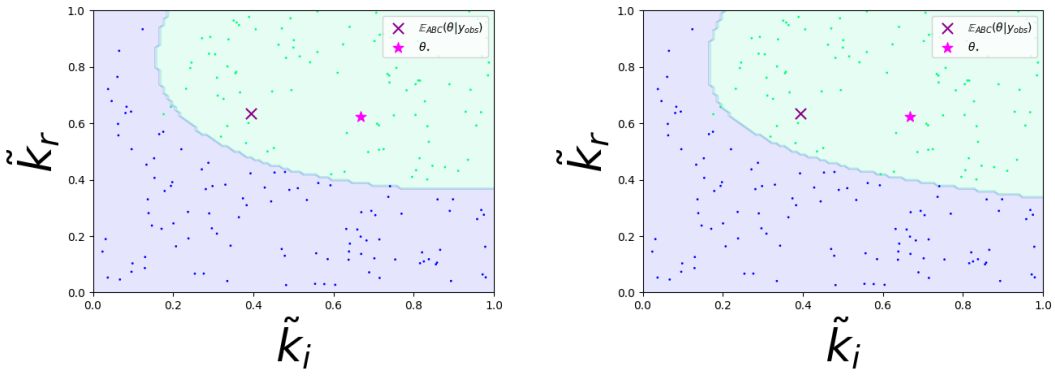
(a) Param. synthesis for $\mathcal{M}_\theta$.

(b) Param. synthesis for $\tilde{\mathcal{M}}_\theta$.

Figure 5.10. Approximate parameter synthesis regions generated via SVMs, using estimated satisfaction probabilities from the posterior, $\pi^{h_M}_{ABC}(\theta \mid y_{obs})$ for (5.10a) the HF model, $\mathcal{M}_\theta$ and (5.10b) the LF model, $\tilde{\mathcal{M}}_\theta$.

When we perform an approximate parameter synthesis on samples generated from the posterior distribution, the parameter space is partitioned well within the region of parameter space covered by the posterior but tends to be incorrect for the remainder of the parameter space. When we are concerned with only the credibility calculation and not the parameter synthesis, performing this parameter synthesis seems superfluous and in fact leads to overestimating the credibility calculation - thus for accuracy since we are performing SMC on posterior samples, it would be better to use these values directly as opposed to generating these parameter regions. When we are interested in the parameter synthesis regions, if given the choice to only perform SMC once, we should consider performing the SVM classification on samples generated from the prior probability distribution. In Figure 5.11 we present the approximate parameter synthesis regions generated running SMC using samples from the prior probability distribution $\theta \sim \pi(\theta)$.

By performing the parameter synthesis on the prior, we obtain a clearer idea of how the parametrised model satisfies the property across the whole parameter space and the credibility calculation can be calculated using this approximate parameter synthesis. Particularly, using the parameter synthesis regions generated from the



(a) Param. synthesis for $\mathcal{M}_\theta$.

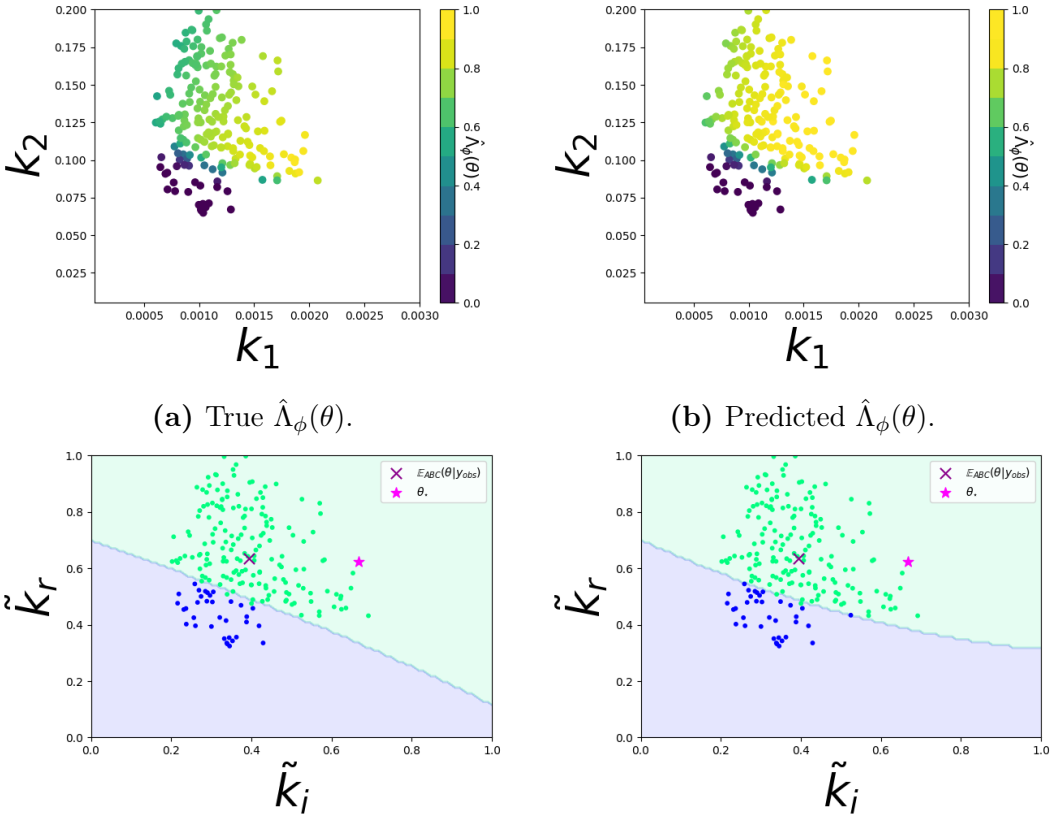
(b) Param. synthesis for $\tilde{\mathcal{M}}_\theta$.

Figure 5.11. Approximate parameter synthesis regions generated via SVMs, using estimated satisfaction probabilities from the prior, $\pi(\theta)$ for (5.11a) the HF model, $\mathcal{M}_\theta$ and (5.11b) the LF model, $\tilde{\mathcal{M}}_\theta$.

prior, the credibility calculation can be calculated using the weighted samples generated from the posterior probability distribution without the need to perform SMC for these sampled parameters. Using the SVMs generated for the HF and LF models from the prior, the credibility calculation for the posterior is given by $\mathcal{C}_{\text{mf}} = P(\mathcal{M}_\theta \models \phi \mid y_{obs}) = 0.7944$ for the HF model and $\mathcal{C}_{\text{mf}} = P(\tilde{\mathcal{M}}_\theta \models \phi \mid y_{obs}) = 0.7702$. We note in this case that the credibility calculation for the HF model that utilises information gathered from prior samples is in fact more accurate (error of 0.0141) compared to using the parameter synthesis regions generated from the posterior (error of 0.0431). Alternatively, we could use the information gathered from the prior to understand the relationship between the satisfaction probabilities of the LF model and how they map to the HF model across the parameter space. If we were able to formalise the relationship or create a mapping between the estimated satisfaction probabilities of the LF model, $\left[\tilde{\Lambda}^L_\phi(\theta), \tilde{\Lambda}^U_\phi(\theta)\right]$ to the satisfaction probabilities of the HF model, $\left[\hat{\Lambda}^L_\phi(\theta), \hat{\Lambda}^U_\phi(\theta)\right]$, then we would be able to run SMC on the LF model and assert whether the corresponding HF model would satisfy the property of interest.

What we propose as one way to formalise this relationship is to train a neural network [52] to perform this mapping from the LF model to the HF model, $\mathcal{G} : \left[\tilde{\Lambda}^L_\phi(\theta), \tilde{\Lambda}^U_\phi(\theta)\right] \rightarrow \left[\hat{\Lambda}^L_\phi(\theta), \hat{\Lambda}^U_\phi(\theta)\right]$. We train the neural network using information gathered from the SMC scheme gathered from the prior, and then compute the credibility calculation by performing SMC on the LF model with the posterior samples, which is computationally cheaper to evaluate compared to the HF model, and input these estimations to predict the satisfaction probabilities of the HF model

for the posterior probability distribution. From the results in Figure 5.12 we can see that the neural network predicts the satisfaction probabilities for the HF model reasonably well by using estimates obtained for the LF model. By using the neural network to predict the HF satisfaction probabilities, we obtain a credibility calculation of $\mathcal{C}_{\text{mf}} = P(\mathcal{M}_\theta \models \phi \mid y_{obs}) = 0.8230$, which slightly overestimates the "true" credibility calculation value $\mathcal{C}_{\text{mf}} = P(\mathcal{M}_\theta \models \phi \mid y_{obs}) = 0.8085$.



(a) True $\hat{\Lambda}_\phi(\theta)$.

(b) Predicted $\hat{\Lambda}_\phi(\theta)$.

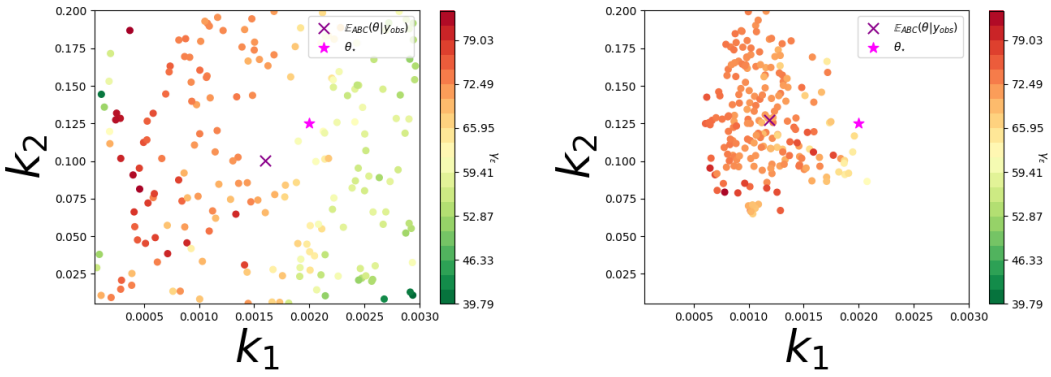
(c) True param. synthesis.

(d) Predicted param. synthesis.

Figure 5.12. True and predicted satisfaction probabilities for the HF model, where the predictions are from a neural network that takes $\left[\tilde{\Lambda}^L_\phi(\theta), \tilde{\Lambda}^U_\phi(\theta)\right]$ as input and output $\left[\hat{\Lambda}^L_\phi(\theta), \hat{\Lambda}^U_\phi(\theta)\right]$. The samples have been classified whether or not they satisfy the property ϕ, where $\theta \in \Theta_\phi$ is represented by yellow in (5.12a) and (5.12b) and green in (5.12c) and (5.12d).

From Figure 5.9, we can see that there is a significant difference in the estimated satisfaction probabilities for the HF and LF model - clearly there are regions in the parameter space where the LF model is a better approximation to the HF model. Using the approximate abstraction metric introduced earlier, represented by $\hat{\gamma}_\varepsilon$, in Figure 5.13 we quantify this difference and present the regions of the parameter space

where the LF model is a better approximate abstraction to the HF model. The results presented in Figure 5.13 are obtained by running MF-BSMC on parameter samples generated from the prior, $\theta \sim \pi(\theta)$ and samples generated from the posterior, $\theta \sim \pi^{h_M}_{ABC}(\theta \mid y_{obs})$. Future work will involve further incorporating this information to the continuation probabilities estimated in the MF-ABC [93] and MF-ABC-SMC [94] work and develop a framework where we can perform formal verification on the LF model and use this relationship to determine whether the same properties are satisfied or not on the HF model, similar to what is initially explored when we introduce the mapping $\mathcal{G} : \left[\tilde{\Lambda}^L_\phi(\theta), \tilde{\Lambda}^U_\phi(\theta)\right] \rightarrow \left[\hat{\Lambda}^L_\phi(\theta), \hat{\Lambda}^U_\phi(\theta)\right]$.



(a) $\pi(\theta)$.

(b) $\pi^{h_M}_{ABC}(\theta \mid y_{obs})$.

Figure 5.13. Estimated γ_ε values for the LF and HF model, where θ is sampled from the prior $\pi(\theta)$ (5.13a), and from the posterior $\pi^{h_M}_{ABC}(\theta \mid y_{obs})$ (5.13b).

We conclude from this case study that we achieve the most accurate credibility calculation when we perform SMC for the HF model for samples generated from the posterior, $\pi^{h_M}_{ABC}(\theta \mid y_{obs})$. However, when we want more information as to how the model performs over the whole parameter space, we can perform the SMC technique using samples generated from the prior which can then be used to synthesise the parameter regions Θ_ϕ and $\Theta_{\neg\phi}$. In addition to this approximate parameter synthesis, we can use the information gathered from the SMC ran from the prior to obtain a credibility calculation for the HF model of the posterior probability distribution by training a neural network to predict the satisfaction probabilities of the HF model given the satisfaction probabilities of the LF model - although this would require us to run MF-BSMC first for the HF and LF model, then we would run BSMC for the LF model once we terminate the inference aspect of our algorithm. This latter option may be a viable option for when the posterior converges to regions of the parameter

space that lead to stiff systems that lead to particularly costly simulations from the HF model.

With the integration of Bayesian SMC and MF-ABC-SMC, we have succeeded in reducing the total number of simulations and the computational costs of the inference technique. With the multifidelity ABC framework, calculating weights $w^{(i)}$ for sampled parameters $\theta^{(i)}$ may be calculated without incurring the computational cost of simulating y, whereas in the previous ABC-(SMC)^2 we are required to simulate $y \sim p(y \mid \theta)$ for every weight evaluation, where the computational cost of doing so is $\mathbb{E}[c(\theta)] \gg \mathbb{E}[\tilde{c}(\theta)]$, where $\mathbb{E}[\tilde{c}(\theta)]$ is the cost of generating LF simulations $\tilde{y} \sim p(\tilde{y} \mid \theta)$. With regards to the number of simulations, recall that in the ABC-(SMC)^2 for a given model the worst case scenario for the total number of simulations would be $\bar{N} \times n_{\mathcal{O}}$, where $n_{\mathcal{O}}$ is the maximum number of simulations required for the SMC scheme in ABC-(SMC)^2 and $\bar{N} \geq N \times M$. Given that we do not perform SMC for every sampled parameter, the total number of simulations that we run SMC for the MF-ABC-(SMC)^2 scheme is now $\bar{N} = N$ if we run SMC only on samples generated from the prior or posterior, or $\bar{N} = 2N$ if we run SMC for both sets of samples. Additionally, we see from Figure 5.1 that a fewer number of simulations are required to verify properties using Bayesian SMC compared to the Massart algorithm. By denoting the maximum number of simulations required for Bayesian SMC as $n_{\mathcal{B}}$, we note that $n_{\mathcal{B}} < n_{\mathcal{O}}$ and the total number of simulations now is $N \times n_{\mathcal{B}}$ (or $2N \times n_{\mathcal{B}}$ if we use samples from both $\pi(\theta)$ and $\pi_{ABC}^{h_M}(\theta \mid y_{obs})$), where we note $N \times n_{\mathcal{B}} < \bar{N} \times n_{\mathcal{O}}$ where $\bar{N}$ here is defined for the ABC-(SMC)^2 scheme.

5.4 Discussion of Multifidelity Bayesian Verification

In this chapter we introduced a multifidelity extension to our Bayesian verification framework. Whilst the ABC-(SMC)^2 extension introduced in the previous chapter greatly improves the applicability and scalability technique of the Bayesian Verification framework, in this form the framework may require a restrictive number of simulations. While the work can be easily parallelised, in this chapter we looked to reduce the overall workload required from the ABC-(SMC)^2 technique. The first thing we considered was reducing the total number of simulations required. In ABC-(SMC)^2, we perform SMC for each sampled parameter, which is possible for classic ABC-SMC algorithms such as [108] as we generate a number of simulations for each sampled parameter. At the cost of higher variability, we can reduce the number of

simulations of each sampled parameter to allow for a faster exploration of the parameter space [19]. Instead of naively performing SMC for each sampled point, we reduce the number of simulations to $B_m = 1$ and instead explore when to perform SMC. We also explore using an alternative form of SMC, Bayesian SMC that requires a fewer number of simulations compared to the frequentist Massart SMC technique. To further improve the scalability of the work, we propose a further extension of the work that utilises multifidelity methods, which allows one to use LF models, which are cheaper and faster to simulate, in combination with HF models to perform Bayesian inference. We also extend the use of multifidelity methods to also perform statistical model checking, which we introduce in MF-BSMC. By performing simultaneous SMC for the HF and LF model, we can quantitatively assess how well the LF model approximates the HF model, utilising techniques introduced in [1]. We define the extension of ABC-(SMC)^2 to multifidelity methods as MF-ABC-(SMC)^2. Whilst this extension improves the scalability of the Bayesian Verification framework further, with the introduction of multifidelity weights in the MF-ABC-SMC aspect of the work, we introduce a non-negative probability of obtaining negative weights. Due to the presence of non-negative weights, we are not able to treat these weighted samples as direct samples from the inferred posterior probability distribution. However, for an arbitrary function $g(\theta)$, we can estimate an expected value of $g(\theta)$ under the inferred posterior $\pi^{h_M}_{ABC}(\theta \mid y_{obs})$ using the multifidelity weights. What we proposed is setting $g(\theta) = \mathbb{1}_{\Theta_\phi}(\theta)$ which we estimate statistically as in the ABC-(SMC)^2 framework to obtain a multifidelity credibility calculation $\mathcal{C}_{\text{mf}}$. Care must be taken when dealing with these negative weights as the magnitude of the negative weights can have substantial affects on the credibility calculation as well as the accuracy of the inference aspect of the algorithm, which we explored in our case study.

In our case studies, we also explored when to perform the multifidelity SMC technique, MF-BSMC. If we are interested in the credibility calculation alone, the most accurate yet one of the most expensive ways to do so would be to perform SMC using parameters that have been instantiated with the parameter samples generated from the posterior probability distribution. However, if one is interested in generating an approximate parameter synthesis technique over the whole parameter space, then we should consider performing MF-BSMC on models generated from the prior. By doing so, we obtain an approximate parameter synthesis technique which can also be used to perform a credibility calculation, using SVMs similar to that proposed in ABC-(SMC)^2. We can also use the estimated satisfaction probabilities generated by models sampled from the prior distribution to formalise a relationship between the

LF and HF satisfaction probabilities across the parameter space Θ. In our case study, we explored formalising this relationship by training a neural network which would allow us to predict the satisfaction probability of the HF model using satisfaction probabilities estimated using the LF model, which in this work is assumed to be much cheaper to simulate and evaluate.

Although the ABC-(SMC)^2 framework can be applied on a wide variety of models, we have extended the framework to allow for more scalable implementations with the use of multifidelity weights to introduce the MF-ABC-(SMC)^2 algorithm. Future work would involve further investigation as to how formally verify properties defined on the HF model by using information gathered on the LF model alone. Further extensions would be to consider how to avoid generating negative weights in the multifidelity framework, as by doing so would allow us to treat the weighted parameter samples from $\pi^{h_M}_{ABC}(\theta \mid y_{obs})$ as true samples from the posterior, and explore alternative likelihood-free Bayesian inference techniques that would improve the accuracy of the inference part.

Chapter 6

Conclusion and Future Work

In this thesis, we presented a framework that integrates stochastic modelling, Bayesian methods, formal verification and machine learning techniques to present a data-driven formal verification framework. This framework allows for simultaneous inference and verification which addresses some of the challenges present when constructing mathematical models of stochastic biological systems given data. With this framework, given limited and partially observed data, we can construct an accurate stochastic model that represents an underlying biological system using Bayesian inference. Given this model, we formally verify properties to either further our understanding about the underlying biological system of interest, or allow for the accurate design of novel biological devices. For our Bayesian verification framework, we need only the ability to generate simulations from a parametrised stochastic model of choice, making the framework intuitive and simple to use. In the earlier chapters of the thesis, we introduce the motivation for integrating these fields for the data-driven verification of stochastic systems and also introduce the necessary background material. In Chapter 3, we tackle the problem of learning model parameters given sparse and partially observed data, and we use this inferred model to demonstrate Bayesian verification. We utilise likelihood-free techniques (ABC) to infer parameters given partially observed and sparse data and perform parameter synthesis given in [31] to calculate the probability that an underlying data-generating system satisfies a property of interest. This chapter introduces the notion of Bayesian verification, explores the effects of varying the data and highlights the limitations of the framework when it is reliant on numeric based parameter synthesis techniques such as [31]. Whilst established parameter synthesis techniques such as [30, 31] allow for a thorough exploration of the parameter space, these numeric based techniques do not scale particularly well and are only applicable to restrictive stochastic models. In Chapter 4 we integrate the likelihood-free inference technique introduced in Chapter 3 with statistical model checking and

supervised machine learning to tackle the limitations that are introduced from using numeric based parameter synthesis techniques. By integrating these techniques further, for *any* given generic parametrised stochastic model we can simultaneously infer parameters and perform statistical model checking to perform parameter synthesis, which allows for the Bayesian verification framework initially introduced in Chapter 3 to be applied to a wider variety of stochastic models of interest. While the work in Chapter 4 greatly increases the scalability and generality of the Bayesian verification framework, the framework is now entirely simulation-based and in particular for statistical model checking, requires a high number of simulations to estimate the probability of a model satisfying a property of interest. In our final work in Chapter 5, we employ multifidelity methods, specifically [93, 94], to reduce the computational burden of this entirely simulation based Bayesian verification framework. To further reduce the number of simulations required, we explore when is the optimal time to deploy statistical model checking and introduce multifidelity statistical model checking, which we use to formalise the relationship between the low-fidelity model and the high-fidelity model. This multifidelity statistical model checking algorithm estimates the probability of satisfying a given property on both the low-fidelity and high-fidelity model and quantifies how adequately the low-fidelity model acts as an approximation of the high-fidelity model, using the metric introduced in [1]. We also explore when to perform the approximate parameter synthesis and train a neural network to map the estimated satisfaction probabilities of the low-fidelity model to the high-fidelity model equivalent. Doing so would allow us to estimate satisfaction probabilities on the high-fidelity model given the estimates obtained by running statistical model checking on a low-fidelity model, which by definition is cheaper and faster to evaluate. The development of the multifidelity Bayesian verification framework and multifidelity statistical model checking in Chapter 5 concludes the work presented in this thesis.

The Bayesian verification framework presented in this thesis can be applied to a variety of stochastic systems of interest to learn and subsequently formally verify properties of interest on underlying data-generating stochastic systems. The framework presents a viable and simple alternative to established techniques in the realm of formal verification, for example, the approximate parameter synthesis technique we've presented is faster and circumvents the restrictions of thorough procedures such as [30] and its GPU accelerated equivalent [31]. The flexibility of the Bayesian verification does come at a cost of requiring a high number of simulations to allow for the data-driven verification of stochastic systems and in Chapter 5 we deploy

multifidelity methods and alternative SMC techniques to both reduce the total number of simulations required and reduce the computational burden of the necessary simulations. The introduction of multifidelity techniques addresses some of the computational overheads of the framework, however, the use of multifidelity weights in the inference scheme gives rise to negative weights, which can influence the inference results drastically. The Bayesian verification framework is flexible and easy to use and the work presented in this thesis, in particular with the deployment of multifidelity models in formal verification, gives rise to many interesting future research questions.

6.1 Future Work

Whilst we have addressed some of the limitations in the field of data-driven formal verification with our Bayesian verification framework, there are many areas in which the Bayesian verification framework could be developed further.

Experimental Evaluation

The first and most obvious future work is to evaluate the Bayesian verification framework experimentally on living cells either to learn further about the biological system, or to aid in the design of a novel biological device as is common in synthetic biology. Future collaboration with experimental biologists would involve working with alternative stochastic models of interest, formally defining properties of interest and considering further the type of data that is worked with and how this would affect certain metrics in the Bayesian verification framework. For example, determining an appropriate choice of summary statistic $s = S(y)$ in the inference aspect of the work.

Generating alternative models to explore

Throughout the thesis, we assume that there is access to a parametrised model $\mathcal{M}_\theta$ which adequately describes the underlying stochastic system of interest. Obviously, the success of the Bayesian verification framework hinges on this assumption and future work consists of proposing or synthesising the parametrised stochastic model $\mathcal{M}_\theta$, similar to [29], or extending the framework to consider multiple models $\mathcal{M}_\theta^i$ where $i = 1, 2, \ldots, N_{mod}$ where we denote the total number of possible models as N_{mod}.

Generating low-fidelity models and alternative metric

With regards to the work considered in Chapter 5, we work with high-fidelity models $\mathcal{M}_\theta$ (pCTMC) with a well defined low-fidelity model $\tilde{\mathcal{M}}_\theta$ (pSDE) that approximates the high-fidelity model. Instead of working with a well-defined low-fidelity model, we can consider working with abstract models of high-fidelity models, as is done in the work of [22], to deploy in our multifidelity framework. Further, we could consider exploring alternative metrics to [1] in the multifidelity statistical model checking algorithm we introduced.

Addressing negative weights introduced in multifidelity work

A particular limitation of the multifidelity framework is the introduction of negative weights. These negative weights prevent from treating the sampled weights as direct samples from the inferred posterior $\pi^{h_M}_{ABC}(\theta \mid y_{obs})$. In particular, due to the magnitude of the negative weights, these negative weights can drastically influence the mean posterior $\hat{\theta}$ and thus the inferred posterior $\pi^{h_M}_{ABC}(\theta \mid y_{obs})$. Future research would consider alternative ways to work with the multifidelity extension that avoids the introduction of negative weights - other fields of research, such as [4] have proposed ways to circumvent negative weights arising in experiments.

Parallelisation

As highlighted in the thesis, the extensions of Chapter 5 are introduced to reduce the *total* workload of the Bayesian verification framework. Many aspects of the framework introduced in Chapter 4 can be parallelised to speed up the process but don't necessarily reduce the overall workload - the simulations (namely CRN simulations) [118], the inference scheme [62] and the SMC scheme [64] all naturally allow for parallelisation which would drastically speed up results. In addition to this, the multifidelity framework introduced in the final chapter could also be parallelised.

Extend formal properties

Currently we restrict ourselves to a time-bounded fragment of continuous stochastic logic [69]. Future work would involve working with more complex properties or perhaps properties defined in alternative logics.

Appendix A

ABC-(SMC)^2 Results and Massart Algorithm

Here we present the Massart algorithm introduced for the ABC-(SMC)^2 framework and the inference results.

A.1 Absolute-Error Massart Algorithm

Here we present the slightly modified Sequential Massart Algorithm with Absolute Error (Algorithm 13). The outputs of Algorithm 13 are $\hat{\Lambda}_\phi(\theta)$, the total number of simulation undertaken $B_m = n$, the sum of the kernel smoothing functions $\sum_{b=1}^{B_m} K_{h_m}(\|s^b - s_{obs}\|)$ and the mean summary statistic produced from n simulations, $\bar{d}$. The algorithm is slightly modified to consider the distance function that is crucial for the ABC-SMC aspect of the algorithm.

A.2 ABC-(SMC)^2 SIR Case Study Results

In this section we present the inferred posterior from the SIR case study in Chapter 4 in Table A.1 with the corresponding runtimes in Table A.2.

Algorithm 13 Modified Absolute-Error Sequential Massart Algorithm

Input:

- Absolute-error value ϵ, a confidence parameter δ and coverage parameter δ_c.
- Current distance threshold h_m.
- Sampled parameter θ^{**}.
- True data s_{obs}
- CSL specification ϕ

Output:

- Estimated probability $\hat{\Lambda}_\phi(\theta^{**})$ with corresponding bounds $[\Lambda^L_\phi(\theta^{**}), \Lambda^U_\phi(\theta^{**})]$.
- Sum of kernel smoothing functions $\sum_{b=1}^{B_m} K_{h_m}(\|s^b - s_{obs}\|)$.
- Mean summary statistic from B_m simulations $\bar{d}$.

Set initial number of successes, $l = 0$, and initial iteration $k = 0$.
Set $B_m = n_{\mathcal{O}}$, where $n_{\mathcal{O}} = \lceil \frac{1}{2\epsilon^2} \log \frac{2}{\delta} \rceil$ is the Okamoto bound and the initial confidence interval $I_0 = [a_0, b_0] = [0, 1]$ in which $\Lambda_\phi(\theta^{**})$ belongs to.
while $k < B_m$ **do**
  $k \leftarrow k + 1$
  Generate trace $y^{(k)} \sim p(y \mid \theta^{**})$ and calculate $s^k = S(y^{(k)})$.
  Calculate $K_{h_m}(\|s^k - s_{obs}\|)$
  $z(y^{(k)}) = \mathbb{1}(y^{(k)} \models \phi)$
  $l \leftarrow l + z(y^{(k)})$
  $I_k = [a_k, b_k] \leftarrow \text{CONFINT}(l, k, \delta_c)$
  if $1/2 \in I_k$ **then**
    $B_m = n_{\mathcal{O}}$
  else if $b_k < 1/2$ **then**
    $B_m = \lceil \frac{2}{h_a(b_k,\epsilon)\epsilon^2} \log \frac{2}{\delta - \delta_c} \rceil$
  else
    $B_m = \lceil \frac{2}{h_a(a_k,\epsilon)\epsilon^2} \log \frac{2}{\delta - \delta_c} \rceil$
  end if
  $B_m \leftarrow \min(B_m, n_{\mathcal{O}})$
end while
Calculate $\bar{d} = (1/B_m) \sum_{b=1}^{B_m} s^b$.
Calculate $\sum_{b=1}^{B_m} K_{h_m}(\|s^b - s_{obs}\|)$.
Set $a_k = \hat{\Lambda}^L_\phi(\theta^{**})$, $b_k = \hat{\Lambda}^U_\phi(\theta^{**})$.
return $\hat{\Lambda}_\phi(\theta^{**}) = l/B_m$, $\sum_{b=1}^{B_m} K_{h_m}(\|s^b - s_{obs}\|)$, $\bar{d}$, $[\hat{\Lambda}^L_\phi(\theta^{**}), \hat{\Lambda}^U_\phi(\theta^{**})]$.

Table A.1. Inferred posterior and Bayesian Verification Results for ABC-(SMC)^2 framework.

Case	$\hat{\theta}_M$	Σ_M	Sampled Pars θ^{**}	SVM Accuracy Accuracy	Credibility Calculation
θ_ϕ	$\begin{bmatrix} 0.00215 \\ 0.07050 \end{bmatrix}$	$\begin{bmatrix} 1.46 \cdot 10^{-8} & 4.24 \cdot 10^{-7} \\ 4.24 \cdot 10^{-7} & 1.97 \cdot 10^{-5} \end{bmatrix}$	10952	99.6%	1
$\theta_{\neg\phi}$	$\begin{bmatrix} 0.00072 \\ 0.14519 \end{bmatrix}$	$\begin{bmatrix} 2.47 \cdot 10^{-8} & 3.41 \cdot 10^{-6} \\ 3.41 \cdot 10^{-6} & 9.22 \cdot 10^{-4} \end{bmatrix}$	10069	99.8%	0.0054
$\theta_{\mathcal{U}}$	$\begin{bmatrix} 0.00193 \\ 0.11337 \end{bmatrix}$	$\begin{bmatrix} 8.89 \cdot 10^{-8} & 5.86 \cdot 10^{-6} \\ 5.86 \cdot 10^{-6} & 4.21 \cdot 10^{-4} \end{bmatrix}$	10807	98.7%	0.6784

Table A.2. Runtimes for ABC-(SMC)^2 framework.

	Times [seconds]		
Case	ABC-(SMC)^2	SVM Optimisation	SVM Classification
θ_{ϕ}	64790	168	3.98
$\theta_{\neg\phi}$	8014	82	4.25
$\theta_{\mathcal{U}}$	35833	2166	5.12

Appendix B

Optimal Continuation Probabilities

For both the multifidelity schemes, we are required to input optimal continuation probabilities (η_1, η_2). Whilst the choice of these continuation parameters are user determined, the authors of [93,94] derive optimal continuation probabilities, with the aim deriving continuation probabilities that aim to maximise the ratio of the effective sample size to the expected simulation time required. Here we present the results of these derivation for the interested user, however alternative optimal continuation probabilities could be derived.

B.1 Multifidelity Approximate Bayesian Computation - Rejection Sampling

In Algorithm 7 the continuation probabilities given by (η_1, η_2) are user inputs but optimal values of the continuation probabilities were derived in [93] and are presented here for the interested reader. The aim here is not to reduplicate the results presented in [93] but to provide the interested reader with a choice of optimal continuation probabilities, which are optimal in the sense that the MF-ABC-RS algorithm balances the benefit of reducing the simulation time against the cost of reducing the effective sample size (ESS),

$$ESS = \frac{\left(\sum_{i=1}^{N} w^{(i)}\right)^2}{\sum_{i=1}^{N} (w^{(i)})^2}.$$

The estimates for the optimal continuation probabilities can be calculated for running the MF-ABC algorithm for a number of $m < N$ Monte Carlo samples where the continuation probabilities are set initially to $(\eta_1, \eta_2) = (1, 1)$ to ensure that the low-fidelity model and high-fidelity model are both simulated for each sampled parameter. For this run-in period of length m, assume that both $\tilde{y}_i$ and y_i have been

generated for k of the m sampled parameter values thus far. The remaining $m-k$ values of θ have only generated $\tilde{y}$ and have been accepted or rejected early. Denote the index set $I_m = \{1, \ldots, m\}$ and $I_k = \{i \in I_m \text{ s.t. both } \tilde{y}_i \text{ and } y_i \text{ are generated }\}$. To estimate the optimal continuations, we first need to estimate the computation times $c_p = \mathbb{E}[c \mid \tilde{y} \in \tilde{\Omega}_{\tilde{h}}]P(\tilde{y} \in \tilde{\Omega}_{\tilde{h}})$ and $c_n = \mathbb{E}[c \mid \tilde{y} \notin \tilde{\Omega}_{\tilde{h}}]P(\tilde{y} \notin \tilde{\Omega}_{\tilde{h}})$,

$$\mathbb{E}[\tilde{c}] \approx \frac{1}{m} \sum_{i \in I_m} \tilde{c}(\theta^{(i)}), \tag{B.1a}$$

$$\hat{c}_p = \frac{\rho_m}{\rho_k} \cdot \frac{1}{k} \sum_{i \in I_k} c(\theta^{(i)}) \mathbb{1}_{\tilde{\Omega}_{\tilde{h}}}(\tilde{y}_i), \tag{B.1b}$$

$$\hat{c}_n = \frac{1-\rho_m}{1-\rho_k} \cdot \frac{1}{k} \sum_{i \in I_k} c(\theta^{(i)}) \left(1 - \mathbb{1}_{\tilde{\Omega}_{\tilde{h}}}(\tilde{y}_i)\right). \tag{B.1c}$$

We estimate the true positive probability $\left(p_{tp} = P\left(\left\{\tilde{y} \in \tilde{\Omega}_{\tilde{h}}\right\} \cap \{y \in \Omega_h\}\right)\right)$, the false positive probability $\left(p_{fp} = P\left(\left\{\tilde{y} \in \tilde{\Omega}_{\tilde{h}}\right\} \cap \{y \notin \Omega_h\}\right)\right)$ and the false negative probability $\left(p_{fn} = P\left(\left\{\tilde{y} \notin \tilde{\Omega}_{\tilde{h}}\right\} \cap \{y \in \Omega_h\}\right)\right)$ using the following estimates,

$$\begin{aligned}
\hat{p}_{tp} &= \frac{\rho_m}{\rho_k} \cdot \frac{1}{k} \sum_{i \in I_k} \mathbb{1}_{\Omega_h}(y_i) \mathbb{1}_{\tilde{\Omega}_{\tilde{h}}}(\tilde{y}_i), \\
\hat{p}_{fp} &= \frac{\rho_m}{\rho_k} \cdot \frac{1}{k} \sum_{i \in I_k} \left(1 - \mathbb{1}_{\Omega_h}(y_i)\right) \mathbb{1}_{\tilde{\Omega}_{\tilde{h}}}(\tilde{y}_i), \\
\hat{p}_{fn} &= \frac{1-\rho_m}{1-\rho_k} \cdot \frac{1}{k} \sum_{i \in I_k} \mathbb{1}_{\Omega_h}(y) \left(1 - \mathbb{1}_{\tilde{\Omega}_{\tilde{h}}}(\tilde{y}_i)\right),
\end{aligned}$$

where the g-function dependent equivalent are estimated by

$$\begin{aligned}
\hat{p}_{tp} &= \frac{\rho_m}{\rho_k} \cdot \frac{1}{k} \sum_{i \in I_k} (g(\theta^{(i)}) - \hat{\mu}) \mathbb{1}_{\Omega_h}(y) \mathbb{1}_{\tilde{\Omega}_{\tilde{h}}}(\tilde{y}_i), \\
\hat{p}_{fp} &= \frac{\rho_m}{\rho_k} \cdot \frac{1}{k} \sum_{i \in I_k} (g(\theta^{(i)}) - \hat{\mu}) \left(1 - \mathbb{1}_{\Omega_h}(y_i)\right) \mathbb{1}_{\tilde{\Omega}_{\tilde{h}}}(\tilde{y}_i), \\
\hat{p}_{fn} &= \frac{1-\rho_m}{1-\rho_k} \cdot \frac{1}{k} \sum_{i \in I_k} (g(\theta^{(i)}) - \hat{\mu}) \mathbb{1}_{\Omega_h}(y) \left(1 - \mathbb{1}_{\tilde{\Omega}_{\tilde{h}}}(\tilde{y}_i)\right),
\end{aligned}$$

where $\hat{\mu} = \sum_{i \in I_m} g(\theta^{(i)}) w^{(i)} / \sum_{j \in I_m} w_j$. With these given estimates, we have what we need to estimate optimal continuation probabilities $\hat{\eta}_1$ and $\hat{\eta}_2$. First, the values (η_1^*, η_2^*) are determined,

$$(\eta_1^*, \eta_2^*) = \left(\sqrt{\frac{Q_p}{Q_0}}, \sqrt{\frac{Q_n}{Q_0}}\right),$$

where

$$Q_p = \frac{\hat{p}_{fp}}{\hat{c}_p/\mathbb{E}[\tilde{c}]},\ Q_n = \frac{\hat{p}_{fn}}{\hat{c}_n/\mathbb{E}[\tilde{c}]},\ Q_0 = \hat{p}_{tp} - \hat{p}_{fp}.$$

If $\max\{Q_p, Q_n\} \leq Q_0$, then the optimal continuation probabilities are $(\hat{\eta}_1, \hat{\eta}_2) = (\eta_1^*, \eta_2^*)$. Conversely, if $\max\{Q_p, Q_n\} > Q_0$ then the optimal continuation probabilities are given by

$$(\hat{\eta}_1, \hat{\eta}_2) = \begin{cases} (1, \bar{\eta}_2) & \text{if } \iota(1, \bar{\eta}_2) \leq \iota(\bar{\eta}_1, 1) \\ (\bar{\eta}_1, 1) & \text{otherwise.} \end{cases}$$

where

$$\iota(\eta_1, \eta_2) = \left((\hat{p}_{tp} - \hat{p}_{fp}) + \frac{1}{\eta_1} p_{fp} + \frac{1}{\eta_2} p_{fn}) \right) \left(\mathbb{E}[\tilde{c}] + \eta_1 c_p + \eta_2 c_n \right),$$

$$\bar{\eta}_1 = \min\left\{ 1, \eta_1^* / \sqrt{\frac{1 + \hat{p}_{fn}/Q_0}{1 + \hat{c}_n/\mathbb{E}[\tilde{c}]}} \right\},$$

$$\bar{\eta}_2 = \min\left\{ 1, \eta_2^* / \sqrt{\frac{1 + \hat{p}_{fp}/Q_0}{1 + \hat{c}_p/\mathbb{E}[\tilde{c}]}} \right\}.$$

The continuation bounds are set to these values, subject to a lower bound for both η_1 and η_2, given by η_1^L and η_2^L, respectively,

$$\eta_1 = \max\left(\hat{\eta}_1, \eta_1^L\right),\ \eta_2 = \max\left(\hat{\eta}_2, \eta_2^L\right).$$

For further details as to how these optimal continuation probabilities are derived see [93].

B.2 Multifidelity Approximate Bayesian Computation - Sequential Monte Carlo

Optimal continuation probabilities and their estimates for MF-ABC-SMC were derived fully in the work of [94]. Here we do not provide the full derivation, just the Monte Carlo estimates that are required to estimate the optimal continuation probabilities, $\eta^*(\theta, \tilde{y})$. To estimate optimal continuation probabilities $\eta^*(\theta, \tilde{y})$, we work with the weighted Monte Carlo samples obtained by running MF-ABC-IS, (Algorithm 3), $\{\theta^{(i)}, w^{(i)}\}_{i=1}^N$. In addition to these weighted samples, we store the simulation times for the low-fidelity, $\tilde{t}_i^{sim}$, and high-fidelity model, t_i^{sim}; distances from data $\tilde{d}_i = d(\tilde{y}_i, \tilde{y}_{obs})$, $d_i = d(y_i, y_{obs})$; importance densities $q_i = q(\theta^{(i)})$; and the continuation probabilities $\eta_i = \eta(\theta^{(i)}, \tilde{y}_i)$. We denote the set of low-fidelity simulations by $I_N = \{\tilde{y}_i\}_{i=1}^N$ and $I_k = \{i \in I_N$ where both $\tilde{y}$ and y were generated$\}$ We now define the Monte Carlo

estimates needed find the optimal continuation probability, η^*, which requires the new importance distribution $q^*(\theta)$ and the new ABC distance thresholds, $\tilde{h}$ and h;

$$\hat{Z} = \frac{1}{N}\left[\sum_{n\in I_N}\frac{\pi(\theta^{(n)})}{q_n}\mathbb{I}(\tilde{d}_n \leq \tilde{h}) + \sum_{k\in I_k}\frac{\pi(\theta^{(k)})}{q_k\eta_k}\left(\mathbb{I}(d_k \leq h) - \mathbb{I}(\tilde{d}_k \leq \tilde{h})\right)\right], \quad \text{(B.5a)}$$

$$\hat{W} = \frac{1}{N}\left[\sum_{n\in I_n}\frac{\pi(\theta^{(n)})^2}{q^*(\theta^{(n)})q_n}\mathbb{I}(\tilde{d}_n \leq \tilde{h}) + \sum_{k\in I_k}\frac{\pi(\theta^{(k)})^2}{q^*(\theta^{(k)})q_k\eta_k}\left(\mathbb{I}(d_k \leq h) - \mathbb{I}(\tilde{d}_k \leq \tilde{h})\right)\right], \quad \text{(B.5b)}$$

$$\hat{W}_{fp} = \frac{1}{N}\sum_{k\in I_k}\frac{\pi(\theta^{(k)})^2}{q^*(\theta^{(k)})q_k\eta_k}\mathbb{I}(\tilde{d}_k \leq \tilde{h})\left(1 - \mathbb{I}(d_k \leq h)\right), \quad \text{(B.5c)}$$

$$\hat{W}_{fn} = \frac{1}{N}\sum_{k\in I_k}\frac{\pi(\theta^{(k)})^2}{q^*(\theta^{(k)})q_k\eta_k}(1 - \mathbb{I}(\tilde{d}_k > \tilde{h}))\mathbb{I}(d_k \leq h), \quad \text{(B.5d)}$$

$$\hat{T}_{lo} = \frac{1}{N}\sum_{n\in I_n}\frac{q^*(\theta^{(n)})}{q_n}\tilde{t}_n^{sim}, \quad \text{(B.5e)}$$

$$\hat{T}_{hi,p} = \frac{1}{N}\sum_{k\in I_k}\frac{q^*(\theta^{(k)})}{q_k\eta_k}\mathbb{I}(\tilde{d}_k \leq \tilde{h})t_k^{sim}, \quad \text{(B.5f)}$$

$$\hat{T}_{hi,n} = \frac{1}{N}\sum_{k\in I_k}\frac{q^*(\theta^{(k)})}{q_k\eta_k}(1 - \mathbb{I}(\tilde{d}_k \leq \tilde{h})t_k^{sim}. \quad \text{(B.5g)}$$

The optimal continuation probabilities (η_1^*, η_2^*) must lie on the bounded domain $\mathcal{W} \subseteq (0,1]^2$, $(\eta_1^*, \eta_2^*) \in \mathcal{W}$. We impose minimal values for the choice of continuation probabilities, (η_1^L, η_2^L) such that the choice of continuation probabilities must lie in the domain $\mathcal{W} = [\eta_1^L, 1] \times [\eta_2^L, 1]$. The theoretical efficiency varies with the choice of continuation probabilities and is defined as

$$\psi(\eta_1, \eta_2) = \frac{\mathbb{E}[w]^2}{\mathbb{E}[w^2]\mathbb{E}[T]} = \frac{Z^2}{\Xi(\eta_1, \eta_2)}, \quad \text{(B.6)}$$

where the denominator is estimated by

$$\hat{\Xi}(\eta_1, \eta_2) = \left(\hat{W} + \left(\frac{1}{\eta_1} - 1\right)\hat{W}_{fp} + \left(\frac{1}{\eta_2} - 1\right)\hat{W}_{fn}\right)\left(\hat{T}_{lo} + \eta_1\hat{T}_{hi,p} + \eta_2\hat{T}_{hi,n}\right). \quad \text{(B.7)}$$

Clearly this efficiency is maximised when $\hat{\Xi}$ is minimised. To determine the optimal continuation probabilities (η_1^*, η_2^*) we first calculate $(\bar{\eta}_1, \bar{\eta}_2)$ and $\bar{\Xi}$, given by

$$\bar{\Xi} = \left(\sqrt{(\hat{W} - \hat{W}_{fp} - \hat{W}_{fp})\hat{T}_{lo}} + \sqrt{\hat{W}_{fp}\hat{T}_{hi,p}} + \sqrt{\hat{W}_{fn}\hat{T}_{hi,n}}\right)^2, \quad \text{(B.8a)}$$

$$(\bar{\eta}_1, \bar{\eta}_2) = \left(\sqrt{\frac{\hat{T}_{lo}}{\hat{W} - \hat{W}_{fp} - \hat{W}_{fn}} \cdot \frac{\hat{W}_{fp}}{\hat{T}_{hi,p}}}, \sqrt{\frac{\hat{T}_{lo}}{\hat{W} - \hat{W}_{fp} - \hat{W}_{fn}} \cdot \frac{\hat{W}_{fn}}{\hat{T}_{hi,n}}}\right), \quad \text{(B.8b)}$$

and define the two functions

$$\eta_1(x) = \max\left[\eta_1^L, \min\left[1, \sqrt{\frac{\hat{T}_{lo} - \hat{T}_{hi,n}x}{\hat{W} - \hat{W}_{fp} - (1-x^{-1})\hat{W}_{fn}} \cdot \frac{\hat{W}_{fp}}{\hat{T}_{hi,p}}}\right]\right], \tag{B.9a}$$

$$\eta_2(x) = \max\left[\eta_2^L, \min\left[1, \sqrt{\frac{\hat{T}_{lo} - \hat{T}_{hi,p}x}{\hat{W} - (1-x^{-1})\hat{W}_{fp} - \hat{W}_{fn}} \cdot \frac{\hat{W}_{fn}}{\hat{T}_{hi,n}}}\right]\right]. \tag{B.9b}$$

The optimal continuation probabilities are chosen as follows. If $(\bar{\eta}_1, \bar{\eta}_2) \in \mathcal{W}$, then $(\eta_1^*, \eta_2^*) = (\bar{\eta}_1, \bar{\eta}_2)$. Otherwise, let

$$(\eta_1^*, \eta_2^*) = \arg\min\left\{\hat{\Xi}(1, \eta_2(1)), \hat{\Xi}(\eta_1(1), 1), \hat{\Xi}(\eta_1^L, \eta_2(\eta_1^L)), \hat{\Xi}(\eta_1(\eta_2^L), \eta_2^L)\right\}, \tag{B.10}$$

That is, we first find the minimum value of $\hat{\Xi}(1, \eta_2(1))$, $\hat{\Xi}(\eta_1(1), 1)$, $\hat{\Xi}(\eta_1^L, \eta_2(\eta_1^L))$, or $\Xi(\eta_1(\eta_2^L), \eta_2^L)$ and assign (η_1^*, η_2^*) to its associated arguments.

Bibliography

[1] Alessandro Abate and Maria Prandini. Approximate abstractions of stochastic systems: A randomized method. In *2011 50th IEEE Conference on Decision and Control and European Control Conference*, pages 4861–4866. IEEE, 2011.

[2] Janis L. Abkowitz, Sandra N. Catlin, and Peter Guttorp. Evidence that hematopoiesis may be a stochastic process in vivo. *Nature Medicine*, 2(2):190–197, 1996.

[3] Gul Agha and Karl Palmskog. A survey of statistical model checking. *ACM Transactions on Modeling and Computer Simulation*, 28(1):6:1–6:39, 2018.

[4] Jeppe R Andersen, Christian Gütschow, Andreas Maier, and Stefan Prestel. A positive resampler for monte carlo events with negative weights. *The European Physical Journal C*, 80(11):1–10, 2020.

[5] Christophe Andrieu and Gareth O. Roberts. The pseudo-marginal approach for efficient Monte Carlo computations. *The Annals of Statistics*, 37(2):697–725, 2009.

[6] David Angeli. A tutorial on chemical reaction network dynamics. *European Journal of Control*, 15(3):398–406, 2009.

[7] Adam Arkin, John Ross, and Harley H. McAdams. Stochastic kinetic analysis of developmental pathway bifurcation in phage λ-infected escherichia coli cells. *Genetics*, 149(4):1633–1648, 1998.

[8] Adnan Aziz, Kumud Sanwal, Vigyan Singhal, and Robert Brayton. Verifying continuous time markov chains. In *International Conference on Computer Aided Verification*, pages 269–276. Springer, 1996.

[9] Christel Baier, Boudewijn R. Haverkort, Holger Hermanns, and Joost-Pieter Katoen. Model-checking algorithms for continuous-time Markov chains. *IEEE Transactions on Software Engineering*, 29(6):524–541, 2003.

[10] Christel Baier and Joost-Pieter Katoen. *Principles of Model Checking*. MIT Press, 2008.

[11] Chris P. Barnes, Daniel Silk, Xia Sheng, and Michael P.H. Stumpf. Bayesian design of synthetic biological systems. *Proceedings of the National Academy of Sciences*, 108(37):15190–15195, 2011.

[12] Chris P. Barnes, Daniel Silk, and Michael P.H. Stumpf. Bayesian design strategies for synthetic biology. *Interface Focus*, 1(6):895–908, 2011.

[13] Mark A. Beaumont. Approximate Bayesian computation in evolution and ecology. *Annual Review of Ecology, Evolution, and Systematics*, 41:379–406, 2010.

[14] Mark A. Beaumont, Wenyang Zhang, and David J. Balding. Approximate Bayesian computation in population genetics. *Genetics*, 162(4):2025–2035, 2002.

[15] Lenz Belzner and Thomas Gabor. Bayesian verification under model uncertainty. In *2017 IEEE/ACM 3rd International Workshop on Software Engineering for Smart Cyber-Physical Systems*, pages 10–13. IEEE, 2017.

[16] Mahmoud Bentriou, Paolo Ballarini, and Paul-Henry Cournède. Reachability design through approximate Bayesian computation. In *International Conference on Computational Methods in Systems Biology*, pages 207–223. Springer, 2019.

[17] Espen Bernton, Pierre E. Jacob, Mathieu Gerber, and Christian P. Robert. Approximate Bayesian computation with the Wasserstein distance. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 81(2):235–269, 2019.

[18] Fernando V. Bonassi and Mike West. Sequential Monte Carlo with adaptive weights for approximate Bayesian computation. *Bayesian Analysis*, 10(1):171–187, 2015.

[19] Luke Bornn, Natesh S. Pillai, Aaron Smith, and Dawn Woodard. The use of a single pseudo-sample in approximate Bayesian computation. *Statistics and Computing*, 27(3):583–590, 2017.

[20] Luca Bortolussi, Luca Cardelli, Marta Kwiatkowska, and Luca Laurenti. Central limit model checking. *ACM Transactions on Computational Logic (TOCL)*, 20(4):1–35, 2019.

[21] Luca Bortolussi, Dimitrios Milios, and Guido Sanguinetti. Smoothed model checking for uncertain continuous-time Markov chains. *Information and Computation*, 247(C):235–253, 2016.

[22] Luca Bortolussi and Luca Palmieri. Deep abstractions of chemical reaction networks. In *International Conference on Computational Methods in Systems Biology*, pages 21–38. Springer, 2018.

[23] Luca Bortolussi and Guido Sanguinetti. Learning and designing stochastic processes from logical constraints. In *International Conference on Quantitative Evaluation of Systems*, pages 89–105. Springer, 2013.

[24] Luca Bortolussi and Simone Silvetti. Bayesian statistical parameter synthesis for linear temporal properties of stochastic models. In Dirk Beyer and Marieke Huisman, editors, *Tools and Algorithms for the Construction and Analysis of Systems*, pages 396–413, Cham, 2018. Springer International Publishing.

[25] George E.P. Box and George C. Tiao. *Bayesian Inference in Statistical Analysis*. Wiley Classics Library. Wiley, 1973.

[26] Richard J. Boys, Darren J. Wilkinson, and Thomas B.L. Kirkwood. Bayesian inference for a discretely observed stochastic kinetic model. *Statistics and Computing*, 18(2):125–135, 2008.

[27] Lubos Brim, Milan Ceska, Sven Drazan, and David Safranek. Exploring parameter space of stochastic biochemical systems using quantitative model checking. In Natasha Sharygina and Helmut Veith, editors, *Computer Aided Verification*, pages 107–123, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[28] Lyle D. Broemeling. *Bayesian Inference for Stochastic Processes*. CRC Press, 2017.

[29] Luca Cardelli, Milan Ceska, Martin Fränzle, Marta Kwiatkowska, Luca Laurenti, Nicola Paoletti, and Max Whitby. Syntax-guided optimal synthesis for chemical reaction networks. In *International Conference on Computer Aided Verification*, pages 375–395. Springer, 2017.

[30] Milan Ceska, Frits Dannenberg, Nicola Paoletti, Marta Kwiatkowska, and Lubos Brim. Precise parameter synthesis for stochastic biochemical systems. *Acta Informatica*, 54(6):589–623, 2014.

[31] Milan Ceska, Petr Pilar, Nicola Paoletti, Lubos Brim, and Marta Kwiatkowska. PRISM-PSY: precise GPU-accelerated parameter synthesis for stochastic systems. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 367–384. Springer, 2016.

[32] Herman Chernoff. A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *The Annals of Mathematical Statistics*, pages 493–507, 1952.

[33] J. Andrés Christen and Colin Fox. Markov chain Monte Carlo using an approximation. *Journal of Computational and Graphical Statistics*, 14(4):795–810, 2005.

[34] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[35] Pierre Del Moral, Arnaud Doucet, and Ajay Jasra. An adaptive sequential Monte Carlo method for approximate Bayesian computation. *Statistics and Computing*, 22(5):1009–1020, 2012.

[36] Rick Durrett. *Probability: Theory and Examples*, volume 49. Cambridge University Press, 2019.

[37] Michael B. Elowitz, Arnold J. Levine, Eric D. Siggia, and Peter S. Swain. Stochastic gene expression in a single cell. *Science*, 297(5584):1183–1186, 2002.

[38] Stewart N. Ethier and Thomas G. Kurtz. *Markov Processes: Characterization and Convergence*, volume 282. John Wiley & Sons, 2009.

[39] Sarah Filippi, Chris P. Barnes, Julien Cornebise, and Michael P.H. Stumpf. On optimality of kernels for approximate Bayesian computation using sequential Monte Carlo. *Statistical Applications in Genetics and Molecular Biology*, 12(1):87–107, 2013.

[40] Nikhil Galagali and Youssef M Marzouk. Bayesian inference of chemical kinetic models from proposed reactions. *Chemical Engineering Science*, 123:170–190, 2015.

[41] Anastasis Georgoulas, Jane Hillston, and Guido Sanguinetti. Unbiased Bayesian inference for population Markov jump processes via random truncations. *Statistics and Computing*, 27(4):991–1002, 2017.

[42] Mirco Giacobbe, Călin C. Guet, Ashutosh Gupta, Thomas A. Henzinger, Tiago Paixão, and Tatjana Petrov. Model checking gene regulatory networks. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 469–483. Springer, 2015.

[43] Michael B. Giles. Multilevel Monte Carlo path simulation. *Operations Research*, 56(3):607–617, 2008.

[44] Michael B. Giles. Multilevel Monte Carlo methods. *Acta Numerica*, 24:259–328, 2015.

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

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

[47] Daniel T. Gillespie. The chemical Langevin equation. *The Journal of Chemical Physics*, 113(1):297–306, 2000.

[48] Daniel T. Gillespie. Approximate accelerated stochastic simulation of chemically reacting systems. *The Journal of Chemical Physics*, 115(4):1716–1733, 2001.

[49] Andrew Golightly and Darren J. Wilkinson. Bayesian sequential inference for stochastic kinetic biochemical network models. *Journal of Computational Biology*, 13(3):838–851, 2006.

[50] 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.

[51] Andrew Golightly and Darren J. Wilkinson. Bayesian inference for Markov jump processes with informative observations. *Statistical applications in genetics and molecular biology*, 14(2):169–188, 2015.

[52] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[53] Benjamin M Gyori, Daniel Paulin, and Sucheendra K Palaniappan. Probabilistic verification of partially observable dynamical systems. *arXiv preprint arXiv:1411.0976*, 2014.

[54] Sofie Haesaert, Alessandro Abate, and Paul M.J. Van den Hof. Data-driven and model-based verification: A Bayesian identification approach. In *2015 54th IEEE Conference on Decision and Control (CDC)*, pages 6830–6835. IEEE, 2015.

[55] Matej Hajnal, Morgane Nouvian, David Safranek, and Tatjana Petrov. Data-informed parameter synthesis for population Markov chains. In *International Workshop on Hybrid Systems Biology*, pages 147–164. Springer, 2019.

[56] Tingting Han, Joost-Pieter Katoen, and Alexandru Mereacre. Approximate parameter synthesis for probabilistic time-bounded reachability. *2008 Real-Time Systems Symposium*, pages 173–182, 2008.

[57] John Heath, Marta Kwiatkowska, Gethin Norman, David Parker, and Oksana Tymchyshyn. Probabilistic model checking of complex biological pathways. *Theoretical Computer Science*, 391(3):239–257, 2008.

[58] Desmond J. Higham. An algorithmic introduction to numerical simulation of stochastic differential equations. *SIAM Review*, 43(3):525–546, 2001.

[59] Desmond J. Higham. Modeling and simulating chemical reactions. *SIAM Review*, 50(2):347–368, 2008.

[60] Fritz Horn and Roy Jackson. General mass action kinetics. *Archive for Rational Mechanics and Analysis*, 47(2):81–116, 1972.

[61] Silvana Ilie and Alexandra Teslya. An adaptive stepsize method for the chemical langevin equation. *The Journal of Chemical Physics*, 136(18):184101, 2012.

[62] Nick Jagiella, Dennis Rickert, Fabian J. Theis, and Jan Hasenauer. Parallelization and high-performance computing enables automated statistical inference of multi-scale models. *Cell Systems*, 4(2):194–206, 2017.

[63] Cyrille Jegourel, Jun Sun, and Jin Song Dong. Sequential schemes for frequentist estimation of properties in statistical model checking. In *International Conference on Quantitative Evaluation of Systems*, pages 333–350. Springer, 2017.

[64] Cyrille Jegourel, Jun Sun, and Jin Song Dong. On the sequential Massart algorithm for statistical model checking. In *International Symposium on Leveraging Applications of Formal Methods*, pages 287–304. Springer, 2018.

[65] Sumit K. Jha, Edmund M. Clarke, Christopher J. Langmead, Axel Legay, André Platzer, and Paolo Zuliani. A Bayesian approach to model checking biological systems. In *International Conference on Computational Methods in Systems Biology*, pages 218–234. Springer, 2009.

[66] Samuel Karlin and Howard M. Taylor. *A First Course on Stochastic Processes*. 1966.

[67] William O. Kermack and Anderson G. McKendrick. A contribution to the mathematical theory of epidemics. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, 115(772):700–721, 1927.

[68] Peter E. Kloeden and Eckhard Platen. *Stochastic Differential Equations*. Springer, 1992.

[69] Marta Kwiatkowska, Gethin Norman, and David Parker. Stochastic model checking. In *International School on Formal Methods for the Design of Computer, Communication and Software Systems*, pages 220–270. Springer, 2007.

[70] Marta Kwiatkowska, Gethin Norman, and David Parker. PRISM 4.0: Verification of probabilistic real-time systems. In *International Conference on Computer Aided Verification*, pages 585–591. Springer, 2011.

[71] Marta Kwiatkowska, Gethin Norman, and David Parker. Probabilistic model checking: Advances and applications. In *Formal System Verification*, pages 73–121. Springer, 2018.

[72] Marta Kwiatkowska and Chris Thachuk. Probabilistic model checking for biology. In *Software Systems Safety*, pages 165–189. IOS Press, 2014.

[73] Theodore Kypraios, Peter Neal, and Dennis Prangle. A tutorial introduction to Bayesian inference for stochastic epidemic models using approximate Bayesian computation. *Mathematical Biosciences*, 287:42 – 53, 2017.

[74] Axel Legay, Benoît Delahaye, and Saddek Bensalem. Statistical model checking: An overview. In *International Conference on Runtime Verification*, pages 122–135. Springer, 2010.

[75] Juliane Liepe, Sarah Filippi, Michał Komorowski, and Michael P.H. Stumpf. Maximizing the information content of experiments in systems biology. *PLOS Computational Biology*, 9(1):1–13, 2013.

[76] Curtis Madsen, Chris J. Myers, Nicholas Roehner, Chris Winstead, and Zhen Zhang. Utilizing stochastic model checking to analyze genetic circuits. In *2012 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, pages 379–386. IEEE, 2012.

[77] Gisiro Maruyama. Continuous Markov processes and stochastic equations. *Rendiconti del Circolo Matematico di Palermo*, 4(1):48–90, 1955.

[78] Pascal Massart. The tight constant in the Dvoretzky-Kiefer-Wolfowitz inequality. *The Annals of Probability*, pages 1269–1283, 1990.

[79] Harley H. McAdams and Adam Arkin. Stochastic mechanisms in gene expression. *Proceedings of the National Academy of Sciences*, 94(3):814–819, 1997.

[80] Nicholas Metropolis and Stanislaw Ulam. The Monte Carlo method. *Journal of the American Statistical Association*, 44(247):335–341, 1949.

[81] Gareth W. Molyneux and Alessandro Abate. ABC(SMC)^2: Simultaneous inference and model checking of chemical reaction networks. In Alessandro Abate, Tatjana Petrov, and Verena Wolf, editors, *Computational Methods in Systems Biology*, pages 255–279. Springer International Publishing, 2020.

[82] Gareth W. Molyneux, Viraj B. Wijesuriya, and Alessandro Abate. Bayesian verification of chemical reaction networks. In *International Symposium on Formal Methods*, pages 461–479. Springer, 2019.

[83] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. MIT press, 2012.

[84] Radford M. Neal. Slice sampling. *The Annals of Statistics*, 31(3):705–767, 2003.

[85] Masashi Okamoto. Some inequalities relating to the partial sum of binomial probabilities. *Annals of the Institute of Statistical Mathematics*, 10(1):29–35, 1959.

[86] Bernt Øksendal. *Stochastic Differential Equations: An Introduction with Applications*. Hochschultext / Universitext. Springer, 2003.

[87] Benjamin Peherstorfer, Karen Willcox, and Max Gunzburger. Survey of multifidelity methods in uncertainty propagation, inference, and optimization. *SIAM Review*, 60(3):550–591, 2018.

[88] Elizabeth Polgreen, Viraj B. Wijesuriya, Sofie Haesaert, and Alessandro Abate. Data-efficient Bayesian verification of parametric Markov chains. In *Quantitative Evaluation of Systems - 13th International Conference*, pages 35–51, 2016.

[89] Elizabeth Polgreen, Viraj B. Wijesuriya, Sofie Haesaert, and Alessandro Abate. Automated experiment design for data-efficient verification of parametric Markov decision processes. In *Quantitative Evaluation of Systems - 14th International Conference, QEST*, pages 259–274, 2017.

[90] Dennis Prangle. Summary statistics in approximate Bayesian computation. *arXiv preprint arXiv:1512.05633*, 2015.

[91] Dennis Prangle. Lazy ABC. *Statistics and Computing*, 26(1-2):171–185, 2016.

[92] Dennis Prangle. Adapting the ABC distance function. *Bayesian Analysis*, 12(1):289–309, 2017.

[93] Thomas P. Prescott and Ruth E. Baker. Multifidelity approximate Bayesian computation. *SIAM/ASA Journal on Uncertainty Quantification*, 8(1):114–138, 2020.

[94] Thomas P. Prescott and Ruth E. Baker. Multifidelity approximate Bayesian computation with sequential Monte Carlo parameter sampling. *SIAM/ASA Journal on Uncertainty Quantification*, 9(2):788–817, 2021.

[95] Arjun Raj and Alexander Van Oudenaarden. Nature, nurture, or chance: stochastic gene expression and its consequences. *Cell*, 135(2):216–226, 2008.

[96] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer School on Machine Learning*, pages 63–71. Springer, 2003.

[97] 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.

[98] Guido Sanguinetti, Neil D. Lawrence, and Magnus Rattray. Probabilistic inference of transcription factor concentrations and gene-specific regulatory activities. *Bioinformatics*, 22(22):2775–2781, 2006.

[99] David Schnoerr, Guido Sanguinetti, and Ramon Grima. The complex chemical Langevin equation. *The Journal of Chemical Physics*, 141(2):024103, 2014.

[100] 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.

[101] Scott A. Sisson, Yanan Fan, and Mark Beaumont. *Handbook of Approximate Bayesian Computation*. Chapman and Hall/CRC, 2018.

[102] Alex J. Smola and Bernhard Schölkopf. A tutorial on support vector regression. *Statistics and Computing*, 14(3):199–222, 2004.

[103] Mohammad Soltani, Cesar A. Vargas-Garcia, Duarte Antunes, and Abhyudai Singh. Intercellular variability in protein levels from stochastic expression and noisy cell cycle processes. *PLOS Computational Biology*, 12(8):1–23, 2016.

[104] David Spieler, Ernst Moritz Hahn, and Lijun Zhang. Model checking CSL for Markov population models. *arXiv preprint arXiv:1111.4385*, 2011.

[105] Mukund Thattai and Alexander Van Oudenaarden. Intrinsic noise in gene regulatory networks. *Proceedings of the National Academy of Sciences*, 98(15):8614–8619, 2001.

[106] Tianhai Tian and Kevin Burrage. Stochastic models for regulatory networks of the genetic toggle switch. *Proceedings of the National Academy of Sciences*, 103(22):8372–8377, 2006.

[107] Tina Toni and Michael P.H. Stumpf. Simulation-based model selection for dynamical systems in systems and population biology. *Bioinformatics*, 26(1):104–110, 2010.

[108] 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. *Journal of the Royal Society Interface*, 6(31):187–202, 2008.

[109] Aad W. Van der Vaart. *Asymptotic Statistics*, volume 3. Cambridge University Press, 2000.

[110] Nicolaas G. Van Kampen. The expansion of the master equation. In *Advances in Chemical Physics*, pages 245–309. Wiley-Blackwell, 1976.

[111] Nicolaas G. Van Kampen. *Stochastic Processes in Physics and Chemistry*, volume 1. Elsevier, 1992.

[112] Joep Vanlier, Christian A. Tiemann, Peter A.J. Hilbers, and Natal A.W. van Riel. Optimal experiment design for model selection in biochemical networks. *BMC Systems Biology*, 8(1):20, 2014.

[113] David J. Warne, Ruth E. Baker, and Matthew J. Simpson. Multilevel rejection sampling for approximate Bayesian computation. *Computational Statistics & Data Analysis*, 124:71–86, 2018.

[114] David J. Warne, Ruth E. Baker, and Matthew J. Simpson. Simulation and inference algorithms for stochastic biochemical reaction networks: from basic concepts to state-of-the-art. *Journal of the Royal Society Interface*, 16(151):20180943, 2019.

[115] Darren J. Wilkinson. Parameter inference for stochastic kinetic models of bacterial gene regulation: a Bayesian approach to systems biology. In *Proceedings of 9th Valencia International Meeting on Bayesian Statistics*, pages 679–705, 2010.

[116] Darren J. Wilkinson. *Stochastic Modelling for Systems Biology*. CRC press, 2018.

[117] Mae L. Woods, Miriam Leon, Ruben Perez-Carrasco, and Chris P. Barnes. A statistical approach reveals designs for the most robust stochastic gene oscillators. *ACS Synthetic Biology*, 5(6):459–470, 2016.

[118] Yanxiang Zhou, Juliane Liepe, Xia Sheng, Michael P.H. Stumpf, and Chris P. Barnes. GPU accelerated biochemical network simulation. *Bioinformatics*, 27(6):874–876, 2011.

[119] Paolo Zuliani. Statistical model checking for biological applications. *International Journal on Software Tools for Technology Transfer*, 17(4):527–536, 2015.

[120] Paolo Zuliani, André Platzer, and Edmund M. Clarke. Bayesian statistical model checking with application to simulink/stateflow verification. In *Proceedings of the 13th ACM International Conference on Hybrid Systems: Computation and Control*, pages 243–252, 2010.