Applications of Gaussian Processes in Modelling Biological Collectives

Master Thesis

presented by

Julia Klein

at the

Modelling of Complex, Self-Organising Systems
Department of Computer and Information Science

1. Evaluated by	Jun.-Prof. Dr. Tatjana Petrov
2. Evaluated by	Dr. rer. nat. Matthias Rupp

Konstanz, 2022

Klein, Julia
Applications of Gaussian Processes in Modelling Biological Collectives
Master Thesis Computer and Information Science
University of Konstanz
February 2022

Abstract

Complex mechanisms in biological systems are typically modelled and analysed by simpler stochastic computer models. Population Markov Chains are a popular choice to describe interactions between agents of a biological collective to examine the underlying behaviour. Uncertainties, especially due to the lack of data or only partially-defined models, complicate the execution of formal verification methods to the chains. Therefore, new methods are desired that overcome these shortcomings. We identified Gaussian Processes within the field of Machine Learning as a promising technique to improve the existing approaches in the analysis of biological collectives. Combining classical formal verification with Gaussian Processes gives rise to creating scalable, flexible, and data-efficient frameworks. We have implemented three powerful workflows that address different research questions of analysing biological collectives. Moreover, we show the wide applicability of our model-agnostic method on three different experimental contexts.

Table of Contents

List of Figures

List of Abbreviations

ARD	automatic relevance determination	17
BSCC	bottom strongly connected component	6
CDF	cumulative distribution function	22
CRN	Chemical Reaction Network	9
CSL	Continuous Stochastic Logic	8
CTMC	Continuous-Time Markov Chain	6
DTMC	Discrete-Time Markov Chain	5
EP	Expectation Propagation	24
GP	Gaussian Process	15
GPC	Gaussian Process Classification	21
GPR	Gaussian Process Regression	17
LOOCV	Leave-One-Out Cross-Validation	32
MSE	mean squared error	32
ODE	ordinary differential equations	9
PCTL	Probabilistic Computation Tree Logic	7
PMC	Probabilistic Model Checking	27
SCC	strongly connected component	6
SMMC	Smoothed Model Checking	29

CHAPTER 1

Introduction

1.1 Background

From cells to social animals and humans, there exist diverse, complex mechanisms in the interactions of biological systems that are not yet understood. Studying the collective behaviour of such systems is based on interdisciplinary research between the fields of biology, psychology, sociology, physics and computer science.

Usually, domain experts observe a phenomenon and aim to find evidence and insights about the underlying process. The more complex this process is, the harder it becomes to study and understand it. Typically, complex biological systems are analysed by simpler, mathematical computer models. Stochastic behaviours that are inherent in collective biological systems, such as animal populations, need to be captured in the underlying computer models. Therefore, stochastic models that include randomness, such as Markov Chains and Chemical Reaction Networks, are a popular choice for modelling biological systems [6]. Since Chemical Reaction Networks can be analysed using Continuous-Time Markov Chains [10], we will often omit their explicit mentioning for the sake of brevity.

Population Markov Chains are widely used to describe individual decisions as well as interactions of discrete agents of a biological population through simple transition rules in the chain [9, 18]. Oftentimes, we can only observe some discrete, final state of the agents that can be mapped to the steady-state of the chain, but we have no information about the intermediate, individual behaviours. Formal verification methods are applied to these chains to analyse specific properties and explore the underlying mathematical model to eventually explain individual, as well as collective behaviours. Most commonly, methods from Probabilistic Model Checking are applied to verify biological systems [11, 30].

Uncertainties in form of lacking data, parametric models, or insufficient expert knowledge complicate the execution of formal methods. Such uncertainties are, however, often part of biological systems and need to be handled carefully to achieve higher-level analyses of collective phenomena. In this thesis, we look at problems in understanding collective behaviour, for which weaknesses of formal methods become evident and their limits are quickly reached.

Within the field of computer science, we identify Machine Learning as a promising area for finding methods that overcome the shortcomings of formal methods and improve the existing procedures in the analysis of biological collectives. More specifically, we aim for a technique that is *flexible*, to allow for a widespread use, *scalable*, to account even for large populations, and *data-efficient*, to produce accurate results for scarce data sets.

Here, we focus on the application of *Gaussian Processes* that are an innovative Bayesian approach in Machine Learning to learn unknown functions. Fulfilling all previously mentioned requirements, Gaussian Processes are considered a "desired meta-model in various applications" [16]. Besides the power of Gaussian Process models to produce good results from little data, another advantage lies in the handling of uncertainty. In contrast to other, mostly black-box Machine Learning models, Gaussian Processes deal not only with uncertainty in the training data, but also provide guarantees of the predictions in form of confidence intervals.

In this thesis, we will look at research questions in modelling and analysing biological collectives, where formal verification methods are not satisfactory. The goal is to combine existing approaches with novel techniques from Machine Learning, i.e. Gaussian Processes, to exploit the strengths of both areas. More precisely, the focus is on three research questions, for which three workflows are implemented: predict the collective response of a population, derive the fitness function a collective tries to optimise, and infer unknown parameters of an underlying stochastic model. The implementations utilise Gaussian Process Regression, as well as Smoothed Model Checking [13], a novel technique based on Gaussian Process Classification to analyse stochastic models.

All three developed methods are tested and evaluated on a case study, where we discuss a social feedback mechanism in a population of honeybees [26]. Within this case study, three different experimental contexts are considered trying to explain the observed phenomenon, which shows the wide applicability of the developed

framework.

The structure of the thesis is as follows: Chapter 2 presents the necessary theoretical background on Markov Chains, Chemical Reaction Networks, and statistical methods used to work with the given experimental data. It concludes with a running example, the SIR model, on which the methods will be shown and explained. We define the concise problem we want to solve in Chapter 3 and state the explicit research questions. In Chapter 4, the methodology of existing approaches we utilise is explained. First, Gaussian Processes are introduced along with their applications in regression and classification. Then, the concepts of Probabilistic Model Checking are described, followed by the novel approach of Smoothed Model Checking. A section about the implemented model selection technique concludes the chapter. The main contribution of this thesis can be found in Chapter 5, where the practical framework is explained and three workflows are described. Chapter 6 presents the case study about honeybees and explains how the workflows are applied depending on three different experimental contexts. Finally, we discuss advantages and limitations of the proposed method in Chapter 7, and further show possible directions of future works.

1.2 Related Work

There exists a substantial amount of work an model-checking parametric stochastic models [f.ex., 25, 29, 46]. Combining the standard methods with Gaussian Processes is a new approach, though some work can already be found leveraging the advantages of Gaussian Processes to verify uncertain, complex stochastic models [f.ex., 1, 7, 11, 28].
The first part of the developed framework is based on Gaussian Process Regression and implemented using the mathematical derivations explained by Rasmussen and Williams [50].
In the other two workflows, we adapt a novel technique called Smoothed Model Checking developed by Bortolussi et al. [13] that is based on Gaussian Process Classification. This model checking method is part of the tool *U-Check* [12] and infers the satisfaction probability of a property as a smooth function over uncertain parameters of a Continuous-Time Markov Chain. The focus of this paper is to find the most robust parameter of an uncertain model to satisfy a specified property with high robustness, but it was not designed to analyse experimental data

without the underlying stochastic model, or general collective phenomena. An active learning approach to improve the scalability of Smoothed Model Checking is introduced by Piho and Hillston [44].
Bartocci et al. [6] investigate how to design a stochastic model such that the robustness of a property is maximised. In particular, the goal is that a specific behaviour of a biological process is maintained despite uncertainties from noise or uncertain model parameters.
The case study we test our framework on is introduced in a work by Hajnal et al. [26], where the goal is to find parameters of an uncertain population Markov Chain from data measurements that can be mapped to the chain's steady-state distribution.

CHAPTER 2

Preliminaries

In this section, we briefly introduce the formal objects and their basic properties used throughout this thesis. Subsequently, we present fundamental statistical methods and a running example on which we will illustrate the developed methods.

2.1 Markov Chain

A popular choice when modelling stochastic random processes are Markov Chains. Markov Chains model sequences of probabilistic events. According to the Markov property, the value of the next state depends only on the value of the current state, and is not affected by any state of the past [3].

2.1.1 Discrete-Time Markov Chain

A Discrete-Time Markov Chain (DTMC) [3] is a tuple $\mathcal{M} = (S,\ \mathbf{P},\ \iota_{init},\ AP,\ L)$ where

- S is a countable, nonempty set of states,
- $\mathbf{P} : S \times S \to [0, 1]$ is the transition probability function specifying for each state s the probability of moving to another state s' in one step, such that for all states s:
$$\sum_{s' \in S} \mathbf{P}(s, s') = 1,$$
- $\iota_{init} : S \to [0, 1]$ is the initial distribution such that $\sum_{s \in S} \iota_{init}(s) = 1$,
- AP is a set of atomic propositions, and
- $L : S \to 2^{AP}$ a labeling function.

2.1.2 Continuous-Time Markov Chain

A Continuous-Time Markov Chain (CTMC) [32] is similar to a DTMC, but allows the modelling of continuous time instead of only discrete time-steps. Formally, a CTMC is a tuple $\mathcal{M} = (S,\ \mathbf{R},\ \iota_{init},\ AP,\ L)$ where

- S is a countable, nonempty set of states,
- $\mathbf{R} : S \times S \to \mathbb{R}$ is the transition rate matrix that gives the rate $\mathbf{R}(s, s')$ at which transitions occur between each pair of states s, s'; the probability of moving from s to s' within t time units is specified by a negative exponential distribution $1 - e^{-\mathbf{R}(s,s')t}$,
- $\iota_{init} : S \to [0, 1]$ is the initial distribution such that $\sum_{s \in S} \iota_{init}(s) = 1$,
- AP is a set of atomic propositions, and
- $L : S \to 2^{AP}$ a labeling function.

In both Markov models, a path consists of a sequence of states that is connected by non-zero transitions and corresponds to a single execution of the system.

2.1.3 Steady-State Behaviour of Markov Chains

The steady state distribution $\pi : S \to [0, 1]$ of a Markov Chain $\mathcal{M}$ is a vector that captures the probabilities of $\mathcal{M}$ to eventually reach the respective states in the long run [3].
A subset of states, T of S, is called strongly connected if for each pair (s, t) of states in T there exists a path fragment $s_0 s_1 ... s_n$ such that $s_i \in T$ for $0 \leq i \leq n$, $s_0 = s$, and $s_n = t$. A strongly connected component (SCC) of $\mathcal{M}$ denotes a strongly connected set of states such that no proper superset of T is strongly connected. A bottom strongly connected component (BSCC) of $\mathcal{M}$ is an SCC T from which no state outside T is reachable, i.e. for each state $t \in T$ it holds that $\mathbf{P}(t, T) = 1$.
Since almost surely any finite Markov Chain eventually reaches a BSCC and visits all its states infinitely often, the steady state distribution depends on the present BSCCs. Therefore, it holds that $\sum_{T \in BSCC(M)} \sum_{s \in T} \pi(s) = 1$.

2.1.4 Parametric Markov Chain

In a parametric Markov Chain, the transition probabilities in $\mathbf{P}$ (or rates in $\mathbf{R}$) involve m unknown parameters $\boldsymbol{\theta} = [\theta_1, ..., \theta_m] \in \mathbb{R}^m_{\geq 0}$. Transitions can either be simple assignments of parameters θ, or also rational or polynomial functions of θ. For a specific assignment of θ we get a concrete Markov Chain denoted by $\mathcal{M}_\theta$. Usually, qualitatively different dynamics arise at different values of θ, which is why the dependency on θ is often crucial [39].

2.1.5 Population Markov Chain

We consider a parametric Markov Chain as a population model with n identical agents to model the stochastic behaviour of biological populations. By specifying simple rules for the decisions and interactions of agents, emerging behaviours can be analysed on an individual- as well as population-level [9, 18]. We study population models of interacting agents, where reaction rules describe how the system state can change. These behaviours of the agents are represented by unknown transitions between the states of the Markov Chain. Furthermore, observations of the final states of agents can be mapped to the chain's BSCCs [7].

2.2 Temporal Logic

For verifying Markov Chains, we need to describe properties as behaviours of the chain and check if the model satisfies the property or not. Properties are written in a property specification language to ensure a precise description. Here, we introduce two languages of temporal logic, where the operators refer to behaviours over time [3].

2.2.1 Probabilistic Computation Tree Logic

As a language for properties of DTMCs, we introduce Probabilistic Computation Tree Logic (PCTL) [27]. There are two types of formulae, state formulae that are evaluated over states, and path formulae that are evaluated over paths.

The syntax of PCTL formulae is defined as:

- state formula: $\Phi ::= \ tt \mid a \mid \neg\Phi \mid \Phi \wedge \Phi \mid P_{\bowtie p}(\varphi)$
- path formula: $\varphi ::= \ X\Phi \mid \Phi \ \mathcal{U}^{\leq t}\Phi$

for $a \in AP$, $\bowtie \in \{\leq, <, \geq, >\}$, $p \in [0,1]$ and $t \in \mathbb{N} \cup \{\infty\}$. State formula P defines the probability measure of paths satisfying φ in the interval $\bowtie \ p$. Operators $\neg$ ("not"), $\wedge$ ("and"), and $\mathcal{U}$ ("until") are standard operators in temporal logic.
The truth of PCTL formulae is defined by the satisfaction relation $s \models \Phi$ for a state s and a formula Φ, and $w \models \varphi$ for a path w and path formula φ. The semantics is defined as:

- $s \models tt \ \forall s \in S$
- $s \models a$ iff $a \in L(s)$
- $s \models \neg\Phi$ iff $s \not\models \Phi$
- $s \models \Phi_1 \wedge \Phi_2$ iff $s \models \Phi_1 \ \wedge \ s \models \Phi_2$
- $s \models P_{\bowtie p}(\varphi)$ iff $Pr(s, \varphi) \bowtie p$
- $w \models \Phi_1 \ \mathcal{U}^{\leq t}\Phi_2$ iff $\exists i \leq t$ s.t. $w[i] \models \Phi_2$ and $w[j] \models \Phi_1$ for all $0 \leq j < i$.

Note that other operators can be derived from these definitions, f.ex. $\Phi_1 \vee \Phi_2 \equiv \neg(\neg\Phi_1 \wedge \neg\Phi_2)$ [33].

2.2.2 Continuous Stochastic Logic

To define properties over CTMCs, we present the Continuous Stochastic Logic (CSL) [2, 4]. The following syntax defines formulae of CSL:

- state formula: $\Phi ::= \ tt \mid a \mid \neg\Phi \mid \Phi \wedge \Phi \mid P_{\bowtie p}(\varphi) \mid S_{\bowtie p}(\Phi)$
- path formula: $\varphi ::= \ X_{[t_0,t_1]}\Phi \mid \Phi \ \mathcal{U}_{[t_0,t_1]}\Phi$

for $a \in AP$,, $\bowtie \in \{\leq, <, \geq, >\}$, $p \in [0,1]$ and $t_0, t_1 \in \mathbb{R}_{\geq 0}$. State formula P defines the probability measure of the paths satisfying Φ, and S defines the steady-state probability for Φ in the interval $\bowtie \ p$. Operators $\neg$ ("not'), $\wedge$ ("and"), $\mathcal{U}$ ("until"), and X ("next") are standard temporal operators.

For the satisfaction relation $\models \subseteq S \times CSL$, the following semantics is defined:

- $s \models tt \ \forall s \in S$
- $s \models a$ iff $a \in L(s)$
- $s \models \neg\Phi$ iff $s \not\models \Phi$
- $s \models \Phi_1 \wedge \Phi_2$ iff $s \models \Phi_1 \wedge s \models \Phi_2$
- $s \models P_{\bowtie p}(\varphi)$ iff $Pr(s, \varphi) \bowtie p$
- $s \models S_{\bowtie p}(\Phi)$ iff $\pi_{S'}(s) \bowtie p$,

where $\pi_{S'}(s)$ denotes the steady-state probability for $S' \subseteq S$ wrt. state s.

2.3 Chemical Reaction Network

Chemical Reaction Networks (CRNs) [15] are the standard formalism to describe the dynamics of biological systems mathematically. Analysing CRNs follows either a deterministic approach, where the dynamics are modelled as a system of ordinary differential equations (ODE), or a stochastic approach, where the CRN is modelled as a CTMC.

Formally, we have a finite set of chemical species $X = \{X_1, ..., X_n\}$ that interact according to a set of chemical reactions $R = \{R_1, ..., R_m\}$. The state vector $\eta_t = (\eta_{t,1}, ..., \eta_{t,n}) \in S = \mathbb{N}^n$ denotes the number of molecules X_i in the system at time t. Each reaction consists of a propensity function λ_{R_i} that determines the rate at which the reaction fires, and an update vector a_i that describes the change of the system's state. The stochastic behaviour of a CRN can be analysed by stochastic simulations [10, 22].

2.4 Statistical Moments of Histograms

One of our defined goals is to analyse the collective response of a biological population based on experimental or simulation data. Mostly, this data is based on simply counting occurrences of different final states the participating agents can have. Therefore, we rely largely on histogram data of populations describing frequencies of final states. Statistical methods to extract useful information from

histograms are mainly based on computing the histogram's statistical moments. We focus on the first two moments, mean and variance.

2.4.1 Mean (first moment)

The population mean μ of a histogram is estimated by the sample mean

$$\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i, \tag{1}$$

where the values x_i of a histogram are the product of the relative frequencies p_i times the observations o_i, and n is the sample size [5]. If the standard deviation σ is known, we obtain the following confidence interval for the mean:

$$\overline{x} \pm z_{\alpha/2}\frac{\sigma}{\sqrt{n}},$$

where we use critical values from the table of normal distribution to account for the standard normal quantiles.
Usually, the standard deviation is unknown, and we need to estimate it before constructing confidence intervals. If the sample size is large, we can simply replace the standard deviation by its estimator s and construct the confidence interval accordingly, $\overline{x} \pm z_{\alpha/2}\frac{s}{\sqrt{n}}$.
For small samples sizes, the critical values are taken from the student's $t-$distribution with $n-1$ degrees of freedom and we compute

$$\overline{x} \pm t_{\alpha/2}\frac{s}{\sqrt{n}} \tag{2}$$

to obtain a $(1-\alpha)100\%$ confidence interval for the population mean [5].

2.4.2 Variance (second moment)

We estimate the sample variance s^2 for the population variance σ^2 of an underlying histogram using

$$s^2 = \frac{1}{n-1}\sum_{i}(x_i - \overline{x})^2, \tag{3}$$

where n denotes the sample size, x_i the values of the histogram and $\overline{x}$ the mean value of the histogram [5].

Since the distribution of s^2 is not symmetric, the corresponding confidence interval for the population variance is not calculated in the same way as the interval for the mean. Instead, the critical values come from a Chi-square distribution with $\nu = n - 1$ degrees of freedom. Here, $\chi^2_{\alpha/2}$ denotes the $(1 - \alpha/2)$-quantile, $q_{1-\alpha/2}$. The $(1 - \alpha)100\%$ confidence interval for the population variance is given by

$$\left[\frac{(n-1)s^2}{\chi^2_{\alpha/2}}, \frac{(n-1)s^2}{\chi^2_{1-\alpha/2}} \right], \tag{4}$$

and not of the form "estimator ± margin" due to its non-symmetry [5].

2.5 Running Example - SIR model

As an illustration of the developed methods, we will look at a simple, but popular model that studies the spread of epidemics, called *SIR model* [31]. It models the dynamics of **S**usceptible, **I**nfected, and **R**ecovered individuals for fixed population sizes N, with $N = S + I + R$. Therefore, the SIR model can, in a simplified way, be seen as representation of the behaviour of a collective, although we do not observe a deliberate behaviour, but rather a passive change of the agents' state. Nonetheless, we can leverage its simple structure and dynamics to explain the frameworks in this thesis. The SIR model can be commonly found as use case in model checking papers [e.g., 13, 39, 48], but also as basis for more advanced mathematical models of diseases, for example to analyse the current COVID-19 pandemic [38, 47].

Modelled as a CRN, we get a simple representation of the SIR model with the species $X = \{S, I, R\}$, and the state vector $\eta_t = (S_t, I_t, R_t)$. Only two reactions are considered that follow mass action kinetics:

- infection: $R_1 : \mathrm{S} + \mathrm{I} \xrightarrow{k_i} 2\ \mathrm{I}$
- recovery: $R_2 : \mathrm{I} \xrightarrow{k_r} \mathrm{R}$

Consequently, the propensity functions are

- $f_{R_1}(S_t, I_t, R_t) = k_i \frac{I_t S_t}{N}$
- $f_{R_2}(S_t, I_t, R_t) = k_r I_t$

with the corresponding update vectors $a_1 = (-1, 1, 0)$ and $a_2 = (0, -1, 1)$ [10].

This CRN can also be translated to a parametric CTMC, with which the dynamics can be analysed stochastically [54].

CHAPTER 3

Problem Statement

Dynamics of large, complex systems that exhibit collective behaviour are important to understand, but equally difficult to analyse. While population Markov Chains are typically used to describe and analyse these dynamics, the applied formal verification methods suffer from several shortcomings.

Usually, we have a model and data for a collective of a fixed size, but aim to understand the general mechanism behind an observed phenomenon. Consequently, we need to conduct more experiments to cover more population sizes, and set up the corresponding models, which is both very expensive. When the number of interacting agents in the system increases, the number of states of the system grows enormously, which is known as the *state explosion problem*, and leads to computationally expensive, or even infeasible, analyses [14]. We refer to Chapter 4.2 for more details about probabilistic model checking, which is known as the standard approach in formal verification of biological stochastic systems [11]. Therefore, it is desired to create general models, or model-agnostic methods that do not rely on an explicit description of the whole state space, and are able to predict the collective response for populations of different sizes.

A collective system is known to exhibit some kind of fitness function, i.e. a behaviour that is optimised by the collective, to obtain maximal benefits in terms of food, reproduction, or survival. Inferring this optimised behaviour is of great interest to understand the overall mechanisms a collective exercises. Hence, the developed method should be able to automatically learn the fitness function of a collective from scarce data.

Another limit of formal methods is found in dealing with uncertainties. Uncertainty as a lack of data is mentioned before, but it can also arise as an insufficient model description. Commonly found in models of biological systems, parametric models are not fully specified and involve uncertainty in form of unknown parameters [6]. While there exist approaches in formal methods to analyse parametric Markov Chains, it is still a current problem to find powerful

techniques that give reliable results.

In summary, we can define three research questions regarding the analysis of biological collectives:

1. What is the collective response of populations of different sizes?
2. What is the fitness function the collective tries to optimise?
3. What are the parameters of the underlying model that best describe the collective behaviour? How can we design a model that captures the underlying phenomenon most robustly?

Addressing these questions requires the development of new techniques that overcome the difficulties of formal verification. More precisely, these difficulties refer to dealing with scarce data, uncertain models, and the previously mentioned state explosion problem. In this thesis, we aim to combine the existing approaches with a method from Machine Learning, Gaussian Processes, which are used to efficiently learn unknown functions. Not only are Gaussian Processes capable of dealing with scarce data, but they also provide a quantification of uncertainty as confidence regions together with the estimate. Incorporating uncertainty into the inference is crucial. Data measurements collected from experiments are exposed to observation errors, model inadequacy results from insufficient descriptions of the real-world process [53], and stochastic models generally contain uncertainty as part of their randomness [13]. Gaussian Processes that already provide a quantification of uncertainty are hence a powerful method to get closer to answering the above questions.

In the following chapter, we describe the methodology we use in our framework to derive three workflows that provide analyses concerning the stated research questions. Within the single workflows, we combine standard formal methods with Gaussian Processes to create an innovative approach in the modelling of biological collectives.

CHAPTER 4

Methodology

4.1 Gaussian Process

A Gaussian Process (GP) is a generalisation of a multivariate Gaussian distribution to infinite dimension. It is a non-parametric distribution over a space of functions, and is designed to solve regression and classification problems by approximating unknown functions. Since a GP model is data-driven and can be applied without specifying the underlying distribution beforehand, it surpasses many of the traditional regression and classification methods. Furthermore, the inference does not rely on huge data sets and is powerful even for little data. The predictions of a GP are probabilistic, such that the results provide not only an estimate, but additionally a quantification of uncertainty in form of confidence intervals.
More precisely, in a GP we forego the parametric model and instead define a prior probability distribution over functions directly, from which the posterior distribution can be inferred when data is observed. To define a distribution over functions, we only need to consider the values of the function at a finite set of input values x_n corresponding to the training set and test set data points, such that we can work in a finite space [8]. A kernel-based probabilistic model is set up to learn relationships between observed data and make predictions about new data points.
In general, a GP's characteristics are completely specified by a mean function $m(x)$ and a covariance function $k(x, x')$. Different kernels within the GP model create different covariance functions. Below, the most common kernel functions are introduced [24, 40, 49, 52].

4.1.1 Kernels

Kernels are used as a similarity measure between data points, and rely on the underlying assumption that similar data points should have similar output val-

ues. With the requirement of being positive definite, they compute the covariance matrix Σ.
Each kernel has a number of hyperparameters that specify the precise shape of the covariance function. These hyperparameters are either set to a default value or optimised in the regression workflow to guarantee the best fit of the model to the given training data.
Three of the most commonly used kernel functions are the following [24, 40, 50]:

- *Linear* kernel: $k_{lin}(x, x') = \sigma_b^2 + \sigma^2(x - c)(x' - c)$
  The linear kernel is a non-stationary kernel, which means that the covariance between two points depends on their absolute positions. If all x values were moved, the GP model will produce different predictions. Parameter c determines the point on which all posterior functions hinge. The constant variance σ_b describes how far from 0 the height of the function will be at zero, hence adding an uncertain offset to the model.

- *Radial Basis Function* (RBF) kernel: $k_{rbf}(x, x') = \sigma^2 \exp(-\frac{||x-x'||^2}{2\ell^2})$
  The RBF kernel is a stationary kernel, hence it is invariant to translations and the covariance between two points depends only on their relative positions. Scale factor σ^2 is the variance controlling vertical variation, and describing the average distance of the function away from its mean. Parameter ℓ determines the horizontal lengthscale over which the function varies, and determines the reach of influence on the neighbours. It is also called squared-exponential kernel.

- *Periodic* kernel: $k_{per}(x, x') = \sigma^2 \exp(-\frac{2\sin^2(\pi||x-x'||/p)}{\ell^2})$
  The periodic kernel is a stationary kernel like the RBF kernel, but contains an additional parameter p that controls the periodicity of the function.

Moreover, it is possible to combine two (or more) kernels using *addition* or *multiplication* to receive more high-level structures in the models. This can be especially useful when we want to incorporate some prior knowledge about the data into the function. For example, when we know that the inferred function increases in x, selecting only the RBF kernel would not suffice, since it tends to reverse back to the specified mean value over time. Instead, combining the RBF kernel with a linear kernel could produce the desired behaviour. Both operations are applied in the standard way,

$$k_1 + k_2 = k_1(x, x') + k_2(x, x')$$

$$k_1 \cdot k_2 = k_1(x, x') \cdot k_2(x, x').$$

Multiplication results in a prior with a high value only if both base kernels have a high value, similar to an *AND* operation. In contrast, addition gives a prior with a high value if either of the two base kernels have a high value, similar to an *OR* operation.

Kernels over multi-dimensional inputs can be constructed by adding and multiplying kernels on different dimensions. One popular example is the RBF-ARD kernel, where RBF kernels over d different dimensions with different lengthscales are multiplied:

$$k_{rbf-ard}(x, x') = \prod_{d=1}^{D} \sigma_d^2 \exp(-\frac{||x_d - x'_d||^2}{2\ell_d^2}) = \sigma^2 \exp(-\frac{1}{2}\sum_{d=1}^{D} \frac{||x_d - x'_d||^2}{\ell_d^2}).$$

Such a kernel implements automatic relevance determination (ARD), where each input is associated with a hyperparameter that controls the weight on connections of this input [41]. In the RBF-ARD kernel, the inverse of the lengthscale determines how relevant the input is - the larger the lengthscale, the more irrelevant the input [19].

4.1.2 Gaussian Process Regression

Gaussian Process Regression (GPR) is an innovative Bayesian approach in Machine Learning to describe, learn, and optimise unknown functions. Here, we follow closely the mathematical description of GPR by Rasmussen and Williams [50], and use the other references to complete the overview.

Let the prior be a GP, $f(x) \sim GP(m(x), k(x, x'))$, such that for any finite set of points, this GP defines a joint Gaussian distribution: $p(\mathbf{f}|\mathbf{X}) = \mathcal{N}(\mathbf{f}|\mu, \Sigma)$, where $\mu = (m(x_1), ..., m(x_n))$ and $\Sigma_{ij} = k(x_i, x_j)$. Usually, the mean is set to $m(x) = 0$ since the GP is flexible enough to model the mean sufficiently. The prior is independent of any training data, but specifies some properties of the functions through the choice of the kernel that creates the covariance function. We demonstrate the general framework by our running example, the SIR model, introduced in Section 2.5. Both rates of the model are fixed, $k_i = 0.001$, $k_r = 0.1$, and we vary only the total population size N while keeping initially five infected individuals, $I_0 = 5$, and zero recovered individuals, $R_0 = 0$. The corresponding

CTMC is simulated and the outputs consisting of the final state of the model collected. These states refer to the number of susceptible, infected, and recovered individuals in the system. As a starting point for regression, we define three GP priors with three different kernels. In Figure 1 we show five sampled priors for each of the three kernels that have been evaluated on a test set X_* of $N_* = 200$ linearly-spaced values from 0 to 250.

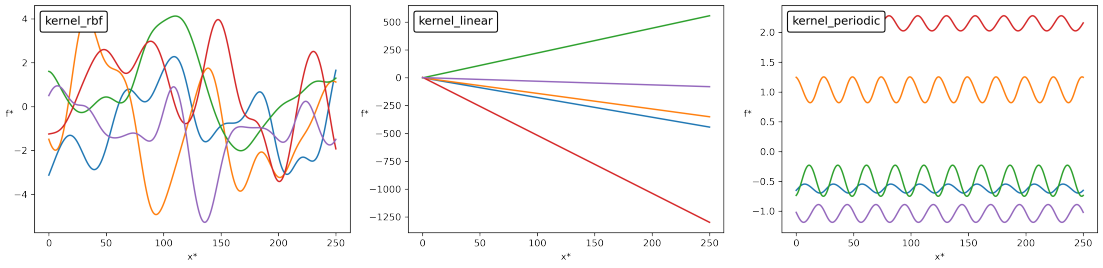


Figure 1 Five functions sampled from each of three GP priors with different kernel functions: RBF, linear, and periodic kernel with randomly chosen parameters $\sigma^2 = 3, \ell = 15, \sigma_b^2 = 2, c = 0.5, p = 25$, evaluated on a test set $X_* = [0, 250]$.

The main goal of GPR is to derive the posterior distribution from training points to find a function that best explains the training data, and predicts function values for new test data points with a high accuracy. Let X be the training data set with known function values $\mathbf{f}$, and X_* the test data set for which we want to predict the corresponding function outputs $\mathbf{f}_*$. For the SIR example, we choose a training set of $X = [25, 50, 75, 100, 125, 150, 175, 200]$ different population sizes N, and collect the corresponding outputs $\mathbf{f} = [0.2412, 0.1764, 0.178, 0.2636, 0.3356, 0.4446, 0.6366, 0.7548]$ reflecting the mean proportions of recovered people for each population size, see Figure 2. We aim to predict the proportion of recovered individuals for other population sizes.

The joint distribution of training and test outputs is given by

$$\begin{bmatrix} f \\ f_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu \\ \mu_* \end{bmatrix}, \begin{bmatrix} \Sigma & \Sigma_* \\ \Sigma_*^T & \Sigma_{**} \end{bmatrix} \right), \tag{5}$$

where $\mu = m(x_i),\ i = 1, ..., n$, denotes the training mean values and analogously μ_* the test mean values. The covariance matrix Σ is evaluated at all pairs of training points, Σ_* at training and test points, and Σ_{**} at test points. Now we are interested in the posterior distribution that is obtained by conditioning the joint

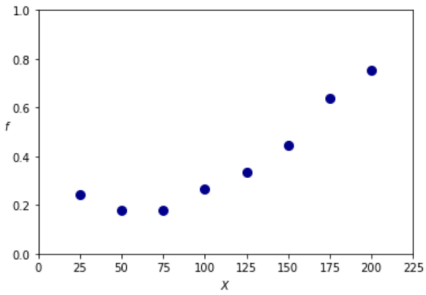


Figure 2 Training data set for the SIR model with population size as input X and mean proportion of recovered individuals as output $\mathbf{f}$.

Gaussian prior distributions on the observations:

$$\mathbf{f}_*|X_*, X, \mathbf{f} \sim \mathcal{N}(\Sigma_*^T \Sigma^{-1} f,\ \Sigma_{**} - \Sigma_*^T \Sigma^{-1} \Sigma_*). \tag{6}$$

By evaluating the mean and covariance we can derive the function values $\mathbf{f}_*$ from the posterior distribution. Evaluating two times the standard deviation of each test point around the mean generates 95% confidence regions [24, 40, 49].

Figure 3 shows the posterior distributions derived for the SIR data for three different kernels, and three combinations of kernels with pre-defined hyperparameters. Five functions are sampled from each GP posterior and shown together with the mean function and the 95% confidence region.

Following this approach creates functions that all go directly through the training points, if possible for the specified kernel and parameters [40]. However, the function space should not be restricted by this constraint, since the training points usually do not reflect the ground truth. Normally, we have noisy training data due to observation errors and limited sample sizes [53]. We can incorporate this uncertainty in GPR, if we assume that the observational noise is normally distributed. Then, the posterior is still a GP and the inference therefore analytically tractable. We have training data $y = f(x) + \epsilon$ with $f \sim GP(0, \Sigma)$ and $\epsilon \sim \mathcal{N}(0, \sigma_f^2 I)$. The noise variance σ_f^2 is independently added to each observation, so we get

$$p(y|f) = \mathcal{N}(y|f, \sigma_f^2 I).$$

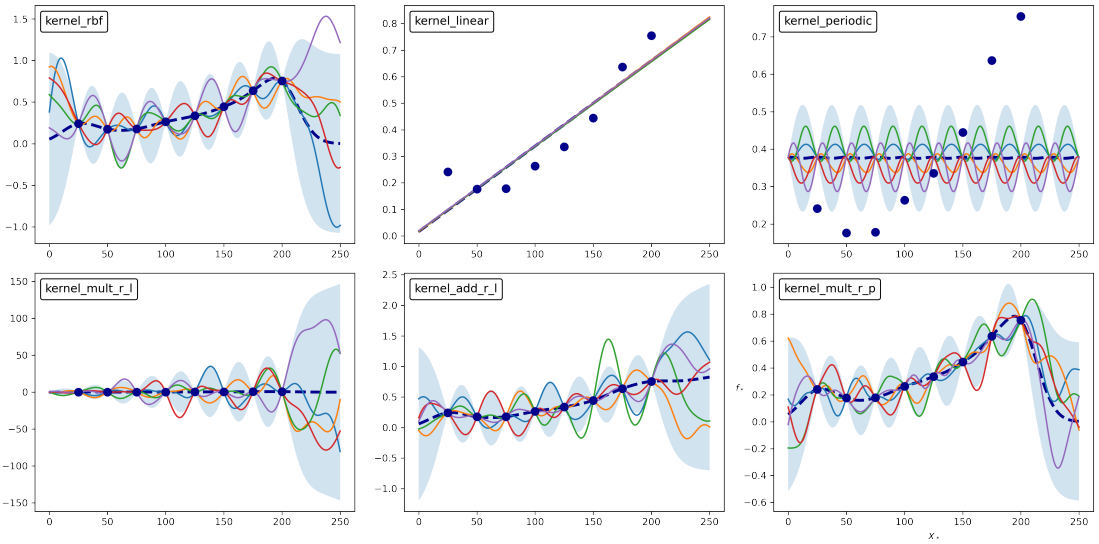


Figure 3 Posterior distributions of SIR model obtained for following kernels: RBF, linear, periodic, multiplication of RBF and linear, addition of RBF and linear, and multiplication of RBF and periodic kernel. Parameters are fixed at $\sigma^2 = 0.3$, $\ell = 15$, $\sigma_b^2 = 1, c = 0.5, p = 25$. Coloured functions represent five samples from the posterior, the blue dashed function is the mean, and the shaded area is the 95% confidence region.

The joint distribution of training and test values changes to:

$$\begin{bmatrix} y \\ f_* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} \Sigma_y & \Sigma_* \\ \Sigma_*^T & \Sigma_{**} \end{bmatrix}\right)$$

with $\Sigma_y := \Sigma + \sigma_f^2 I$. Deriving the posterior distribution follows the same schema as before,

$$f_* | X_*, X, y \sim \mathcal{N}(\Sigma_*^T \Sigma_y^{-1} y,\ \Sigma_{**} - \Sigma_*^T \Sigma_y^{-1} \Sigma_*). \tag{7}$$

In the SIR example, we compute margins of estimation errors around the outputs $\mathbf{f}$ and treat them as noise values σ_f. The resulting posterior distributions are shown in Figure 4, where we observe that the sampled functions, as well as the mean function, do not necessarily go directly through the data points, and that the uncertainty around the data points is increased.

Finally, we can optimise the kernels' hyperparameters to improve the accuracy of predictions in GPR. As standard practice, we follow an empirical Bayesian approach to maximise the marginal likelihood

$$p(y|X) = \int p(y|f, X) p(f|X) df.$$

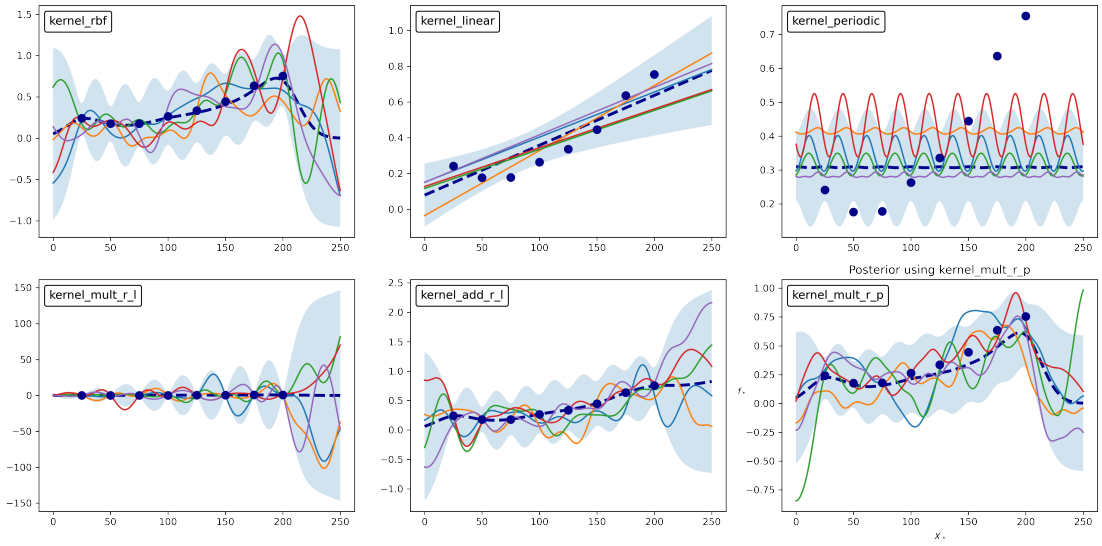


Figure 4 Posterior distributions of SIR model with added noise σ_f^2 around the training points. Following kernels are shown: RBF, linear, periodic, multiplication of RBF and linear, addition of RBF and linear, and multiplication of RBF and periodic kernel. Parameters are fixed at $\sigma^2 = 0.3, \ell = 15$, $\sigma_b^2 = 1$, $c = 0.5$, $p = 25$. Coloured functions represent five samples from the posterior, the blue dashed function is the mean, and the shaded area is the 95% confidence region.

The log marginal likelihood can be rewritten as

$$\log\, p(y|X) = \log\, \mathcal{N}(y|0, \Sigma_y) = -\frac{1}{2} y \Sigma_y^{-1} y - \frac{1}{2} \log\, |\Sigma_y| - \frac{1}{2} N \log\, (2\pi), \qquad (8)$$

where the first term is a data fit term, the second term a model complexity term, and the third term a constant. Minimising the negative log marginal likelihood with respect to the hyperparameters of a kernel gives us an optimised posterior distribution [40].

We apply the optimisation to each kernel of the SIR example and show the resulting posterior mean functions together with the 95% confidence regions in Figure 5.

4.1.3 Gaussian Process Classification

A different approach is needed when the underlying task is to predict a class label to a data point, instead of a continuous output. In this thesis, Gaussian Process Classification (GPC) is applied to a binary classification problem, where we observe class labels $\mathbf{y} \in [0, 1]$ for inputs values X. Again, we refer the interested reader to the book by Rasmussen and Williams [50] for detailed mathematical derivations of the described methods.

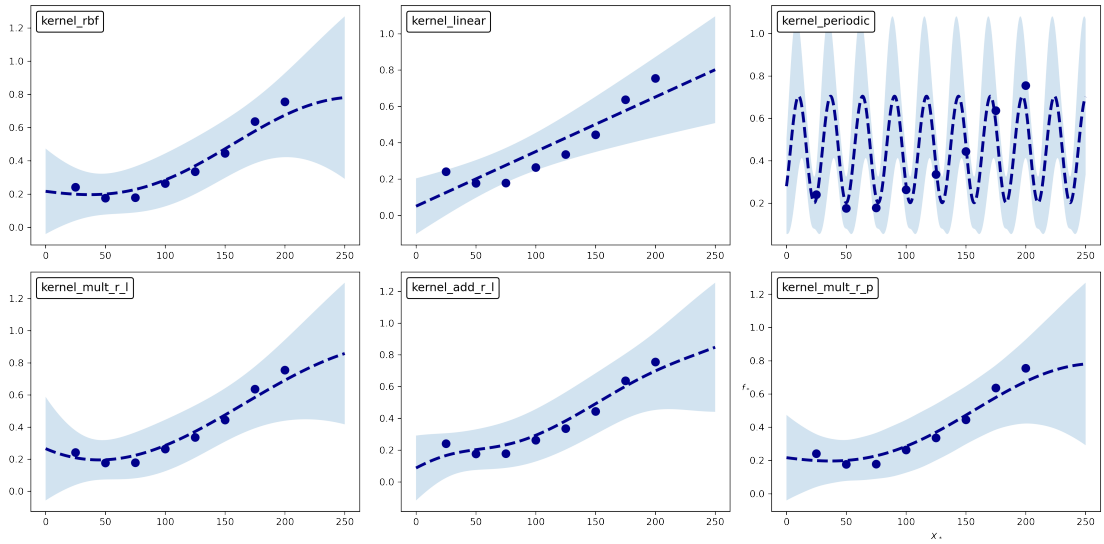


Figure 5 Posterior distributions of SIR model after optimisation of the kernels' hyperparameters. Following kernels are shown: RBF, linear, periodic, multiplication of RBF and linear, addition of RBF and linear, and multiplication of RBF and periodic kernel. The blue dashed function is the mean, and the shaded area is the 95% confidence region.

The main workflow of GPC is similar to GPR. After defining a GP Prior over a suitable function space, the functional form of the likelihood is determined to approximate the posterior distribution. Here, the goal is to get an estimate of a class probability for unknown data points from Boolean observations [50].

The probability of belonging to a certain class at an input value x is related to the value of some latent function $f(x)$ at this location. We do not observe values of f itself, only the inputs X and class labels $\mathbf{y}$, but need this function to define the GP model. In the first step of GPC, a GP prior is placed over the latent function $f(x)$. In order to obtain probabilities instead of real values, the prior is then squashed with the inverse probit transformation. In mathematics, the probit function is equal to the inverse of the standard normal cumulative distribution function (CDF). To map real numbers to probabilities, we need to compute the inverse probit transformation, i.e. the CDF, like this:

$$\Phi(z) = \int_{-\infty}^{z} \mathcal{N}(x|0,1)dx = \frac{1}{2} + \frac{1}{2} \cdot erf\left(\frac{x}{\sqrt{2}}\right),$$

where erf is the Gauss error function, defined as $erf(z) = \frac{2}{\sqrt{\pi}} \int_0^z e^{-t^2}\ dt$ [42].

This means we obtain a prior on class probabilities $\pi(x) \triangleq p(y = 1|x) = \Phi(f(x))$. Usually, the RBF kernel is the preferred choice as kernel function because of its flexibility, and the mean of the prior is again set to zero. See Figure 6 for an example of a GP prior, where we evaluate the RBF kernel and squash it with the inverse probit transformation on a test set X_* of $N_* = 50$ linearly-spaced values from 0 to 15.

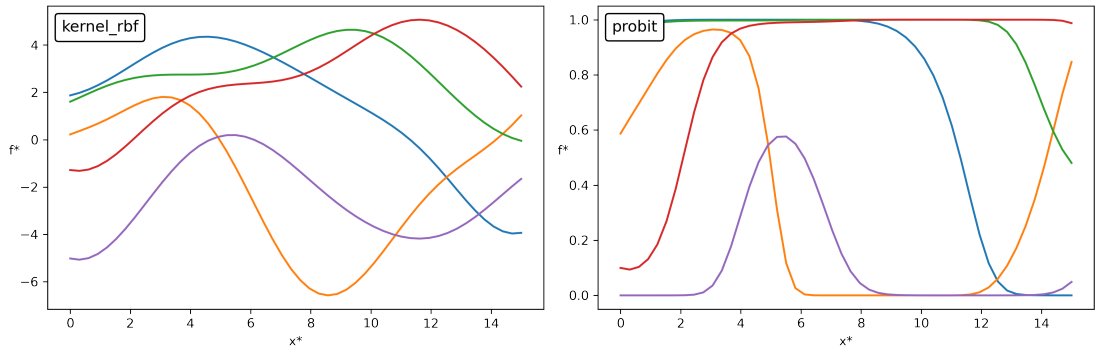


Figure 6 Five functions sampled from a GP prior. Left: Prior with RBF kernel function. Right: The same functions, squashed with the inverse probit transformation to map real values to probabilities. Parameters are fixed at $\sigma^2 = 7, \ell = 3$.

In the next step, the distribution of the latent variable corresponding to a test case is computed with

$$p(f_*|X, \mathbf{y}, x_*) = \int p(f_*|X, x_*, \mathbf{f}) p(\mathbf{f}|X, \mathbf{y}) d\mathbf{f}. \tag{9}$$

This distribution contains the posterior over the latent variables as the product of a normalisation term containing the marginal likelihood, the prior and the likelihood,

$$p(\mathbf{f}|X, \mathbf{y}) = \frac{1}{p(\mathbf{y}|X)} p(\mathbf{f}|X) \prod_{i=1}^{n} p(y_i|f_i), \tag{10}$$

where we will write $Z = p(\mathbf{y}|X)$.
Then, we can calculate predictions for a class probability like this:

$$\overline{\pi}_* \triangleq p(y_* = 1|X, \mathbf{y}, x_*) = \int \Phi(f_*) p(f_*|X, \mathbf{y}, x_*) df_*. \tag{11}$$

In contrast to GPR, the observations are Boolean and the likelihood of the model depends on a Bernoulli random variable, for which we use the probit function, i.e. $p(y_i|f_i) = \Phi(f_i y_i)$. Therefore, the corresponding likelihood is non-Gaussian,

which makes the integral of the posterior distribution in Equation 9 intractable. The non-Gaussian joint posterior thus needs to be approximated by a Gaussian one. Two analytic approximations are mainly used, *Laplace approximation* and *Expectation Propagation*. While both methods are applicable in this case of GPC, we will only focus on the latter one, Expectation Propagation [50], which we will describe below.

Again, we demonstrate the application of GPC on the SIR example. We simulate one trajectory of the model for each of the previously specified population size and check if the trajectory satisfies the CSL property:

$$\varphi := (tt\ \mathcal{U}_{0,10}\ (I > 6)),$$

stating that there are more than 6 infected individuals after $t = 10$ seconds. Figure 7 shows the training data set X of population sizes and the corresponding function outputs $\mathbf{f} \in [0, 1]$ reflecting the satisfaction of property φ. The aim is to predict the satisfaction probability of φ for other population sizes.

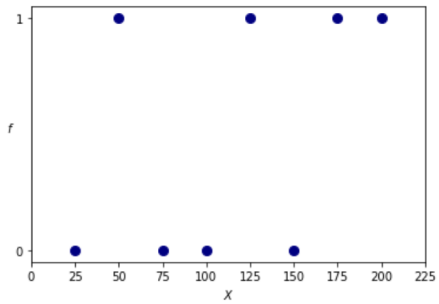


Figure 7 Training data set for the SIR model with population size as input X and satisfaction of property φ as output $\mathbf{f}$. One trajectory was sampled for each population size.

Expectation Propagation

The Expectation Propagation (EP) algorithm is a Bayesian inference method to compute an approximation to probability distributions. The non-Gaussian observation likelihoods in Equation 10 are replaced by univariate Gaussian terms

$$p(y_i|f_i) \simeq t_i(f_i|\tilde{Z}_i, \tilde{\mu}_i, \tilde{\sigma}_i^2) \triangleq \tilde{Z}_i \mathcal{N}(f_i|\tilde{\mu}_i, \tilde{\sigma}_i^2)$$

with site parameters $\tilde{Z}_i, \tilde{\mu}_i$, and $\tilde{\sigma}_i^2$. The product of these independent local likelihoods is defined by

$$\prod_{i=1}^{n} t_i(f_i|\tilde{Z}_i, \tilde{\mu}_i, \tilde{\sigma}_i^2) = \mathcal{N}(\tilde{\boldsymbol{\mu}}, \tilde{\Sigma}) \prod_i \tilde{Z}_i,$$

where $\tilde{\boldsymbol{\mu}}$ is the vector of $\tilde{\mu}_i$ and $\tilde{\Sigma}_{ii} = \tilde{\sigma}_i^2$.

Then, we approximate the posterior distribution by

$$q(\mathbf{f}|X, \mathbf{y}) \triangleq \frac{1}{Z_{EP}} p(\mathbf{f}|X) \prod t_i(f_i|\tilde{Z}_i, \tilde{\mu}_i, \tilde{\sigma}_i^2) = \mathcal{N}(\boldsymbol{\mu}, \Sigma) \tag{12}$$

with $\boldsymbol{\mu} = \Sigma\tilde{\Sigma}^{-1}\tilde{\boldsymbol{\mu}}$, $\Sigma = (K^{-1} + \tilde{\Sigma}^{-1})^{-1}$, and $Z_{EP} = q(\mathbf{y}|X)$ being the EP's approximation to the normalisation term of Equation 10.

Choosing the parameters of the local approximating distributions t_i requires iterating four steps to update individual t_i. The approximation for f_i contains three terms: the prior $p(\mathbf{f}|X)$, the local approximate likelihoods t_j for all $j \neq i$, and the exact likelihood for i, $p(y_i|f_i) = \Phi(y_i f_i)$. The four update steps are:

1. Start from some current approximate posterior and leave out one site index i. In detail, we remove the approximate likelihood term corresponding to i and marginalise all other variables. The resulting distribution is called the *cavity distribution* $q_{-i}(f_i) \triangleq \mathcal{N}(f_i|\mu_{-i}, \sigma_{-i}^2)$ with

$$\mu_{-i} = \sigma_{-i}^2 \cdot (\frac{\mu}{\sigma^2} - \frac{\tilde{\mu}}{\tilde{\sigma}^2}), \text{ and}$$

$$\sigma_{-i}^2 = \frac{1}{\sigma^{-2} - \tilde{\sigma}^{-2}}.$$

2. Combine the cavity distribution with the exact likelihood for i, $p(y_i|f_i)$, to get the non-Gaussian *tilted distribution*.

3. Choose a Gaussian approximation to the non-Gaussian marginal $\hat{q}(f_i) \triangleq \hat{Z}_i \, \mathcal{N}(\hat{\mu}_i, \hat{\sigma}_i^2) \simeq q_{-i}(f_i)p(y_i|f_i)$, such that the Kullback-Leibler (KL) distance to the tilted distribution is minimised. For this, we use moment matching to find the Gaussian with the lowest KL divergence. We get the following desired posterior marginal moments:

$$\hat{Z} = \Phi(z),$$

$$\hat{\mu} = \mu_{-i} + \frac{y\ \sigma_{-i}^2 \mathcal{N}(z)}{\Phi(z)\sqrt{1+\sigma_{-i}^2}}, \text{ and}$$

$$\hat{\sigma}^2 = \sigma_{-i}^2 - \frac{\sigma_{-i}^4 \mathcal{N}(z)}{(1+\sigma_{-i}^2)\Phi(z)}(z + \frac{\mathcal{N}(z)}{\Phi(z)})$$

with $z = \frac{y\ \mu_{-i}}{\sqrt{1+\sigma_{-i}^2}}$.

4. Update the EP approximation by replacing the initial approximation likelihood term with the new term t_i. The new parameters of the approximation are

$$\tilde{\mu}_i = \tilde{\sigma}_i^2(\hat{\sigma}_i^{-2}\hat{\mu}_i - \sigma_{-i}^{-2}\mu_{-i}), \text{ and}$$

$$\tilde{\sigma}_i^2 = \frac{1}{\hat{\sigma}_i^{-2} - \sigma_{-i}^{-2}}.$$

We need to apply these steps iteratively until the moments of the EP approximation do not change significantly, i.e. are indistinguishable from the original ones. After convergence, we update the approximate posterior with Equation 12.

Finally, predictions can be computed using the predictive mean

$$\mathbb{E}_q[f_*|X, y, x_*] = k_*^T(K + \tilde{\Sigma})^{-1}\tilde{\mu}$$

and the predictive variance

$$\mathbb{V}_q[f_*|X, y, x_*] = k(x_*, x_*) - k_*^T(K + \tilde{\Sigma})^{-1}k_*$$

for the latent variable f_*. The predictive probability is given by

$$q(y_* = 1|X, y, x_*) = \phi\left(\frac{\mathbb{E}_q[f_*|X, y, x_*]}{\sqrt{1 + \mathbb{V}_q[f_*|X, y, x_*]}}\right).$$

Confidence bounds are calculated in a standard way,

$$CI_q = \phi\left(\frac{\mathbb{E}_q[f_*|X, y, x_*] \pm 1.96 \cdot \sqrt{\mathbb{V}_q[f_*|X, y, x_*]}}{\sqrt{1 + \mathbb{V}_q[f_*|X, y, x_*]}}\right),$$

for 95% confidence bounds around the mean [13, 40, 50].

Figure 8 illustrates the posterior distribution derived for the SIR data using GPC with the EP algorithm. The covariance matrices are computed with the RBF

kernel and the training data set consists of one observation of satisfaction per input point. The mean predictive probability is shown together with the 95% confidence region for a test set X_* from 0 to 250.

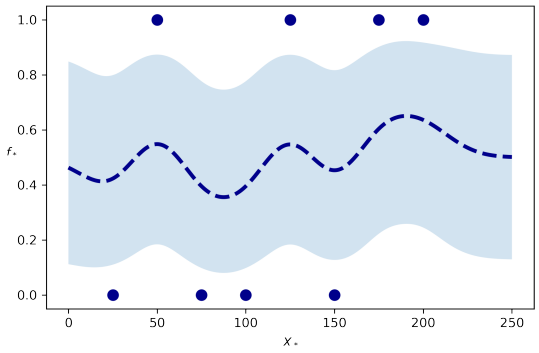


Figure 8 GPC posterior distribution of SIR model obtained for RBF kernel with fixed parameters at $\sigma^2 = 0.5$, $\ell = 17$, showing the satisfaction of property φ with 1 observation per input point. Blue dashed function is the mean predictive probability and the shaded area is the 95% confidence region.

4.2 Probabilistic Model Checking

Analysing biological systems that exhibit stochastic behaviour often requires to set up a stochastic model, such as DTMC, CTMC, or CRN, and subsequently examine its properties and performance.
Probabilistic Model Checking (PMC) is a powerful technique in the automatic formal verification of stochastic systems [3]. It performs an algorithmic analysis whether a system satisfies a certain quantitative property and is implemented in prominent model checking tools such as PRISM [34]. For a specific model $\mathcal{M}$, usually some kind of Markov Chain, and a property φ, the probabilistic model checker investigates if the model satisfies the property, $P(\mathcal{M} \models \varphi)$. The result is either Boolean, as a ”true” or ”false” label, or a numerical value returning the probability of satisfaction. Properties are normally specified as a formula in a language of probabilistic temporal logic, see Section 2.2.1 [32, 35]. Refer to the survey by Katoen [30] for a clear overview of the main probabilistic models, algorithms, and applications of PMC.

There are mainly two different techniques in PMC. Analytical methods compute the path probability of the formula as the probability of the set of trajectories

that satisfy the formula. The computations are based on graph-based algorithms, automata-based methods with Rabin automata, and solving linear equations [35]. Therefore, these methods are very complex and computationally expensive to execute.

Statistical methods simulate the model by repeatedly drawing independent trajectories from the system, and then statistically estimate the satisfaction probability as the average of satisfactions [13]. A simulation algorithm called SSA [21] is typically used to generate trajectories of CTMCs or CRNs. Then, for each trajectory a model checking algorithm determines if the property is satisfied or not, which is followed by a statistical estimation of the satisfaction probability [11].

One elementary problem of PMC is called the *state explosion problem*, shortly mentioned in Chapter 3. The number of states of a system grows exponentially in the size of memory because each state usually represents the global system status at a timepoint. This is especially problematic in the modelling of biological collectives, since population Markov Chains model the interactions and behaviours of independent agents, and increasing the population size leads to an explosive growth of the state space [14]. Therefore, analysing systems quickly becomes challenging and computationally expensive, and new algorithms are being developed to overcome this difficulty [17, 36]. Statistical, simulation-based methods try to address this problem, but encounter difficulties in maintaining accuracy [11, 32].

4.3 Smoothed Model Checking

Analytical and statistical methods of PMC rely on the assumption that the model is fully specified. However, especially when modelling biological systems, parameters of the model are usually not known beforehand and it is of interest to consider varying parameters in the analysis of the model. Model checking uncertain stochastic systems and inferring parameters is therefore a desirable aim in modelling biological collectives [11].

Furthermore, as mentioned in the previous section, methods of PMC quickly reach their limits when analysing growing systems because of the state explosion problem. Therefore, we can only check properties of larger stochastic models for a few, specified parameters.

To overcome both of these limitations, Bortolussi, Milios and Sanguinetti proposed a novel approach called *Smoothed Model Checking (SMMC)* [13]. The goal of this method is to find a statistical estimate of the satisfaction probability of a property φ as a function of the uncertain parameters θ. Standard model checking tools can only provide point-wise estimates of the satisfaction function, but are unable to obtain a global approximation of the underlying function. Gaussian Processes are used to construct an approximation of the satisfaction function from only few observations of trajectories (not) satisfying the property. Then, the satisfaction probability of the formula can be predicted at all possible parameter values together with the related uncertainty. This approach is shown to be accurate and also faster than standard statistical model checking methods, while requiring considerably less simulations [13].

In summary, the aim of SMMC is to evaluate the whole satisfaction function from few samples to predict the satisfaction of a property for an uncertain CTMC efficiently and accurate.

Mathematically, the goal of SMMC is to find an estimate of the satisfaction function $f_\varphi(\theta) = P(\mathcal{M}_\theta \models \varphi)$ for an uncertain CTMC $\mathcal{M}_\theta$ and a property φ that is specified by Metric Interval Temporal Logic (MITL) [37], a different language of temporal logic. A Bayesian algorithm is developed to carry out model checking for all possible θ from just a few observations. Observations are based on simulations of trajectories of the model for fixed parameters θ. For each trajectory, we collect if φ is satisfied, i.e. $\mathcal{M}_\theta \models \varphi$, or not satisfied, i.e. $\mathcal{M}_\theta \not\models \varphi$. Therefore, the actual observations are Boolean and the probability of the observations being 1 is a Bernoulli distribution with parameter $f_\varphi(\theta)$. Inferring the satisfaction function of this problem corresponds to a classification problem, for which GPC, introduced in Section 4.1.3, is suitable. However, to improve the accuracy of smoothed model checking, several trajectories are collected for each input point, such that the observations follow a Binomial random variable $Binomial(m, f_\varphi(\theta))$ for m trajectories.

Then, the training set consists of the satisfactions of the property for independent trajectories sampled from the system for a few input points $\theta_1, ..., \theta_k$. Performing intermediate estimations of the observations would increase the uncertainty of the results, so the GPC algorithm is adapted to exploit the direct use of the exact statistical model of the process through multiple observations per input point.

In the third step of EP, the matching moments of the posterior are computed including the exact likelihood, see Section 4.1.3. Since our observations contain multiple values per input point, we need to apply Gauss-Hermite quadrature to compute the matching moments with the exact observation values. This method is commonly used for approximations of the expectation of a normally distributed variable.

In this way, the estimate of the satisfaction function is more accurate because the exact statistical model is used instead of an approximation, while only few observations per input point are required.

To illustrate the power and highlight the possibility of SMMC, we look again at the running example of the SIR model. For the same property introduced in Section 4.1.3, we now sample 10 trajectories per input point and collect how often the model satisfies the property. The training data is shown in Figure 9, where the corresponding function outputs $\mathbf{f} \in [0, 1]$ reflect the statistical estimate of the satisfaction probability.

Note that the algorithm uses multiple observations per input point for the classification, collecting the satisfactions for each trajectory. After running the adapted GPC, we get the mean predictive posterior distribution and 95% confidence regions shown in Figure 9. Additionally, the results of PMC for 49 values between 5 and 250 are plotted to compare the estimates.

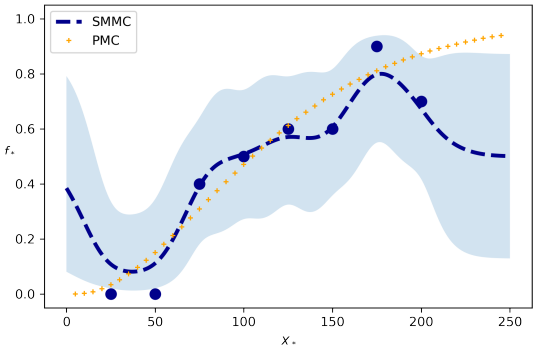


Figure 9 SMMC posterior distribution of SIR model using the RBF kernel with fixed parameters at $\sigma^2 = 0.5$, $\ell = 17$. Training data consists of satisfactions of property φ for 10 trajectories per input point. Blue dots are the statistical averages of satisfactions over observations, blue dashed function is the mean predictive probability and the shaded area is the 95% confidence region. The orange plus symbols are the results of probabilistic model checking for 40 input values between 5 and 250.

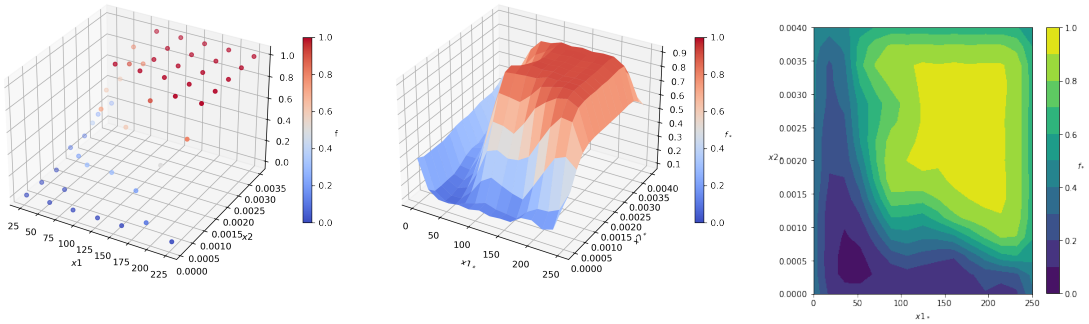


Figure 10 2-dimensional SMMC of SIR model for varying population size $x1 \in [25, 225]$ and varying infection rate $x2 \in [0.0001, 0.0035]$. Posterior distribution is derived with the RBF-ARD kernel with fixed parameters at $\sigma^2 = 0.5$, $\ell = [17, 0.00034]$. Left: training data consisting of satisfactions of property φ for 10 trajectories per input point. Centre: Predictive posterior probabilities for test set in 3D plot. Right: Predictive posterior probabilities for test set in contour plot.

Smoothed model checking can also be applied to two-dimensional training data, when there are two uncertain parameters of the model. The only difference is in the choice of the kernel. For two dimensions, the RBF-ARD kernel with two different lengthscales that was introduced in Section 4.1.1 is most favourable.
We simulate the SIR model and vary not only the population size, but additionally the infection rate k_i. For seven different population sizes and seven different infection rates, we get a total of 49 training points. After running the two-dimensional SMMC, we get the predictive posterior probabilities illustrated in the centre and right plot of Figure 10.

4.4 Model Selection

Both presented analyses are based on GPs that are essentially defined by the chosen kernel function. After developing several models with different kernels in GPR, we need to decide which of the models is the best one. This is called the model selection problem and important to solve to get the best possible results in terms of accuracy [40].
Popular model selection techniques in the area of Machine Learning exploit the vast amount of data that is used to train and test the model. However, in the analysis of biological collectives, there is usually not much data available and the estimates of the performances will not be reliable when using the standard methods. For this

case, cross-validation (CV) is a useful choice, where the available training data is split into K folds.
Then, for each fold $k \in \{1, ..., K\}$, the model is trained on all folds but the k-th one, and tested on the k-th fold. In this thesis, we use a special case of CV called Leave-One-Out Cross-Validation (LOOCV), where we set the number of folds to $K = N$, with N being the size of the data set. Each training data point is used as a test point, all together as test set. The summary statistics of the test set gives an overall evaluation of the goodness of the model. Therefore, we measure the amount of error with the mean squared error (MSE),

$$MSE = \frac{\sum_i (y_i - f_{*_i})^2}{n}$$

with y_i being the observations, f_{*_i} the predictions, and n the size of the test set, for each kernel. The advantages of LOOCV are that it provides reliable and unbiased estimates of the model's performance, even for small data sets [40].
Referring to the running example, we apply LOOCV to the posterior distributions with optimised hyperparameters that were shown in Figure 5. Following the previous explanations, we select the model with the linear kernel, since this gives us the smallest MSE (0.3554).

For SMMC, we have only implemented one kernel, so we cannot perform model selection for this method. However, we compute the MSE of the difference between predictions and observations to evaluate the quality of the model. For the SIR model, we obtain the following results:
MSE= 0.2580 for the GPC posterior shown in Figure 8,
MSE= 0.1454 for the SMMC posterior shown in Figure 9, and
MSE= 0.3475 for the two-dimensional SMMC posterior shown in Figure 10.

CHAPTER 5
Framework

In Chapter 3, we stated three research questions for the presented thesis that are related to three tasks: predict the collective response, infer the fitness function, and infer the parameters of an uncertain model. Here, we explain the three developed workflows to address these questions. Note that the applicability of the analyses depends not only on the research question, but also on the available data and models.
Therefore, we developed a flexible approach in the analysis of biological collectives that works basically with two main techniques: Gaussian Process Regression and Smoothed Model Checking. The project is available on GitHub [1] and structured as follows:

- `data`: experimental data of collectives, simulated data of a CRN and CTMC
- `figures`: illustrations of data and results
- `models`: PRISM models and property specifications; files to run StochNetV2
- `notebooks`: Jupyter notebooks for GPR and SMMC
- `src`: main implementation in Python: GPR, SMMC and model selection

Three Jupyter notebooks show the application of GPR, SMMC, and two-dimensional SMMC in answering the three research questions. Following theoretical explanations, we show step by step the results for the SIR running example in each notebook to illustrate the workflows and the applicability of the methods.

The main implementation is written in Python 3.9.7. Since there are different experimental contexts for which we want to apply our methods (see Chapter 6), we need to consider different input data and models. Therefore, the Python script can

[1] `https://github.com/juliakln/MasterThesis`

automatically read text files containing the input data, and also invoke the model-checking tool PRISM via command line. Beyond that, we provide a bash script that runs StochNetV2 [51], a tool for simulating stochastic reaction networks. In the following, the three work flows to address the outlined research questions are described in more detail.

5.1 Prediction of Collective Response

Experiments to study the collective behaviour of biological populations oftentimes give only results about final, discrete states of the whole collective, that can be interpreted as collective response. When we have the corresponding population Markov Chain, we can map these observations to the chain's BSCCs. Usually, the collective response is obtained for a fixed population size, and deriving the response for other population sizes requires to carry out a new experiment, or to re-engineer and re-run the underlying stochastic model.

In the first application, we use Gaussian Process Regression (GPR), introduced in Section 4.1.2, to predict the behaviour of a collective over varying population sizes. We assume that the available data is based on counting the occurrences of different final states of the participating agents. Therefore, we rely on histogram data of populations of different sizes from which we want to predict the behaviour of populations of other sizes. See Figure 11 for the data set we used for the SIR example. As collective response, we analyse the proportion of infected individuals compared to the total population size in the system after the equilibrium is reached. The data set contains of histograms for population sizes of $n = [25, 50, 75, 100, 125, 150, 175, 200]$ individuals, where each histogram is based on 100 observations.

Predicting a whole distribution is difficult and computationally challenging. Therefore, we choose the first two statistical moments of the histograms as indicative values for the collective response and training data to GPR. According to Equation 1 and Equation 3, the mean and variance of the histograms are computed and used as training function values $\mathbf{f}$. With Equation 2 and 4 we compute margins for the histograms' mean and variance values to apply them as noise values σ_f.

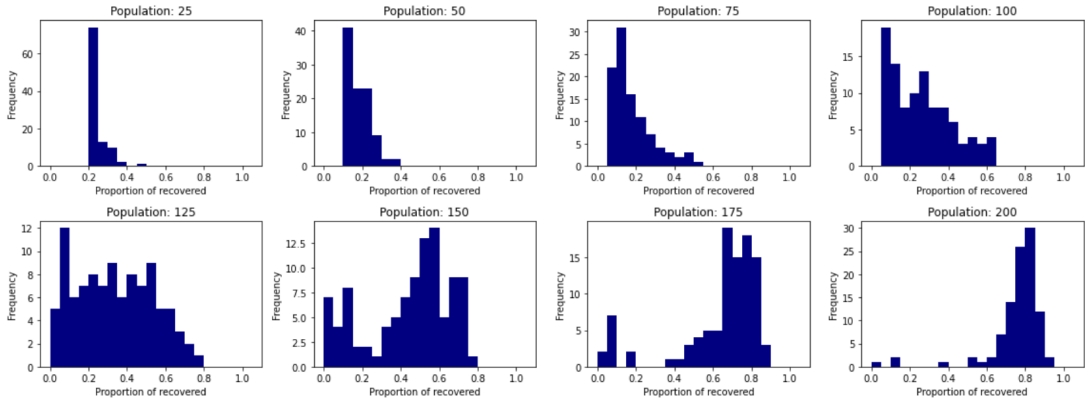


Figure 11 Histograms of the SIR model counting the frequencies of infected individuals as proportions of the total population size. Population sizes vary in $n = [25, 50, 75, 100, 125, 150, 175, 200]$, and each histogram consists of 100 samples. Rates were fixed at $k_i = 0.001$ and $k_r = 0.1$, with initially 5 infected persons.

GPR is run separately for the mean values and variances, and for all available kernels to obtain the most reliable function. The implemented kernels are: linear, RBF, and periodic kernel, addition and multiplication of two kernels. Hyperparameters of all kernels are optimised before the predictive posterior distributions are derived. As a result, the posterior distributions of all (combinations of) kernels are visualised together with the 95% confidence region. In addition, the script returns the kernel with the best fit according to LOOCV and reports the corresponding MSE.

5.2 Inference of Fitness Function

Identifying the fitness function of a collective is a crucial aspect in understanding observed behavioural phenomena. Ordinarily, a population tries to optimise a specific behaviour in order to maximise its benefits. With the second framework, we aim to infer the fitness function of a collective for given data or a specified model. In other words, we want to find out if there is a pattern in the collective response over populations of different sizes which explains the exhibited behaviour.

First, we need to define the property, i.e. fitness function, we want to analyse. The property can either be based on a countable measure of histogram data, or defined in the temporal logic PCTL or CSL when the corresponding model is given.

Furthermore, the property includes one unknown parameter t that defines the specific fitness function, and is thus regarded the target quantity of this approach. Next, a possible range of values of t is determined. For some chosen values in this range, we apply Smoothed Model Checking, introduced in Section 4.3, to obtain the satisfaction probability over varying population sizes. Note that the original implementation of SMMC is part of the JAVA tool U-Check [12], and we stick closely, but with adaptations, to their algorithm in the presented framework.
As input to SMMC, we either have the number of observations of the histograms for which the property is satisfied, or the output of PRISM after model-checking the property. The analysis is run using the RBF kernel with fixed hyperparameters, usually set to $\sigma^2 = \frac{\max(\mathbf{f})-\min(\mathbf{f})}{2}$ and $\ell = \frac{\max(X)-\min(X)}{10}$, as proposed in the tool U-Check [12]. As a result, the posterior distribution of the satisfaction probability for a specific value of t over varying population sizes is visualised together with the 95% confidence region.

In the last step, we compare the posterior distributions for all chosen values of t to find the most probable fitness function. Three values of the distribution are considered:

- the mean value μ to ensure high satisfaction probabilities,
- the standard deviation σ to measure the variability, and
- the coefficient of variation $c_v = \frac{\sigma}{\mu}$ to combine both measures.

Since the posterior distributions are based on the same kind of data ranges, c_v is not interpreted in the normal way, as measure of the variation. The standard deviations are enough to describe the divergence of the data. However, c_v includes the value of the mean, such that a higher mean value gives a lower c_v. In this way, we can find the distribution with little variation and high probabilities using the coefficient of variation. Some researchers, like Faria Filho et al. [20], state that the c_v is mostly classified according to a scale found in a work by Pimentel Gomes [45]. More specifically, a $c_v < 0.1$ is considered low, $0.1 \leq c_v < 0.2$ medium, $0.2 \leq c_v < 0.3$ high, and $c_v \geq 0.3$ very high.
Selecting a value of t that gives a posterior distribution with low c_v results in a plausible fitness function of the collective with small variation but high probabilities.

5.3 Parameter Inference

In the third approach, we consider the case where we have set up a model that is not fully specified, i.e. a parametric Markov Chain, and want to infer its parameters with respect to a pre-defined fitness function. Sometimes, an expert of the biological system that is analysed knows the mechanism of a phenomenon and asks for a computer model that simulates this real-world mechanism as accurately as possible. In this case, the aim is to find an unknown parameter of the model that explains some individual behaviour in the real word. Alternatively, we can look for parameters of the model that maximise the probability for a defined property.

First, we need to define a behaviour that should be exhibited reliably over populations of different sizes as a verifiable property φ. Then, we select two input values to our framework: varying population sizes and the unknown parameter θ, or two unknown parameters. For some chosen values of both input parameters, we apply two-dimensional SMMC to get the satisfaction probability of the property over two varying inputs.

The RBF-ARD kernel is the most common choice for two-dimensional GP models. We fix the hyperparameters as in the previous workflow, whereby parameter ℓ now consists of two values for both input parameters. As a result, the posterior distribution of the satisfaction probability of φ over varying population sizes and the unknown parameter θ is visualised as a contour, and a 3D plot.

Finally, we can choose parameter θ that gives overall high satisfaction probabilities of φ for different population sizes.

5.4 Computational Remarks

Rasmussen and Williams [50] give some hints on how to implement GPR and GPC more robustly. These suggestions can be found in the presented framework and are shortly described to explain deviations of the code from the mathematical foundations in Section 4.1.

In Equation 6, the inverse of the covariance matrix Σ is required to derive the posterior distribution in GPR. To avoid numerically unstable results, we use the Cholesky decomposition

$$LL^T = \Sigma,$$

where L is the lower triangular matrix, and called the Cholesky factor. Then, we can solve for $\alpha = \Sigma^{-1} f = L^{-T} L^{-1} f$, which is used to get the posterior mean

$$\mathbb{E}_q[\mathbf{f}_*|X_*, X, \mathbf{f}] = \Sigma_*^T \alpha$$

and the posterior variance

$$\mathbb{V}_q[\mathbf{f}_*|X_*, X, \mathbf{f}] = \Sigma_{**} - \Sigma_*^T L^{-T} L^{-1} \Sigma_*.$$

Accordingly, we apply the Cholesky decomposition for the hyperparameter optimisation in GPR to ensure numerical stability. The log marginal likelihood in Equation 8 can be rewritten as

$$\log\, p(y|X) = -\frac{1}{2} y^T \alpha - \sum_i \log\, L_{ii} - \frac{1}{2} N \log\, (2\pi)$$

with the previously obtained values for α and L.

A third application of the Cholesky decomposition is found in the third part of the framework, where two-dimensional SMMC is implemented. After defining a prior GP, it can be desired to generate a sample from this distribution to visualise a possible function of this prior. For a multivariate Gaussian $\mathcal{N}(\mu, \Sigma)$, the decomposition of Σ is $LL^T = \Sigma$. Then, a vector $u \sim \mathcal{N}(0, I)$ of length μ is generated. Finally, computing $x = \mu + Lu$ gives a function with the desired mean μ and covariance Σ.

All computations are implemented by closely following the described mathematical derivations without utilising other packages. However, to obtain the hyperparameter optimisation in GPR, we minimise the negative log marginal likelihood in Equation 8 using Python's *SciPy.optimize.minimze* module [2]. We use the *L-BFGS-B* solver to compute the minimal negative log marginal likelihood with respect to the hyperparameters.

[2] `https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.minimize.html#rdd2e1855725e-6`

CHAPTER 6

Bee Case Study

In the following chapter, we test the three workflows described in the previous Chapter 5 in different experimental contexts. We look at a phenomenon found in honeybees, which was illustrated by Hajnal et al. [26]. Biological collectives of honeybees exhibit a social feedback mechanism when being exposed to a threat. To successfully defend their territory, some of the bees become aggressive, decide to sting, and consequently die. During stinging, an alarm pheromone P is released to warn the other bees and increase their aggressiveness. However, if the aggressiveness increased endlessly, eventually all bees of the population would die. Therefore, there needs to be some regulatory mechanism that prevents the colony from becoming extinct, while still being able to effectively defend against the threat. The underlying mechanism of social feedback is still unknown, and it is of great interest to explain this phenomenon. One possible regulation rule could be that the single bee is aware of the fraction of its population that has already stung, and makes its decision based on that proportion. Of course, there also exist more complex hypotheses, for example that the alarm pheromone P does not endlessly increase the bee's aggressiveness, but rather as some S-shaped function, and in that way influences the observed collective behaviour.
In this thesis, we aim to infer the general collective response of bee populations, find the most probable fitness function in terms of the proportion of the population that needs to be kept alive, and find parameters of uncertain models that best explain a specified hypothesis.

To emphasise the wide applicability of the presented model-agnostic implementation, we will demonstrate analyses of three different experimental contexts, depending on the given data and model:

- experimental data, no model
- simulated data, CRN

- parametric DTMC.

Hereafter, all three contexts are described in more detail and the applied work flows are illustrated on analyses of specific examples.

6.1 Experimental Data

Thanks to a biologist from the University of Konstanz who conducted experiments with honeybees, we can test our methods on real-world observations of a biological collective. The previously described phenomenon was tested in three different experiments, called experiment A, B, and C. In each experiment, bee populations of different sizes were put into an arena, where they were exposed to a threat. After some time has passed, the experimenter collected the number of stinging bees by counting the dead bees in the arena. This procedure was repeated several times for each population size. Hence, we get a histogram with the frequencies of stinging bees of each population size for each experiment. See Figure 12 for the detailed outcomes of all three experiments.

First, we apply GPR to predict the collective response of populations of different sizes as the number of stinging bees. As described in the previous chapter, we compute the mean and variance of each histogram, corresponding to the mean and variance of stinging bees for each population size, and use these values as input to the algorithm. As a result, we get a prediction of the collective response for different population sizes. In Figure 13, we show the mean predictive posterior distributions for the mean and variance for all three experiments. We optimised the kernels' hyperparameters and select the kernel with the smallest MSE according to LOOCV.

We obtain the prediction for a specific population size by evaluating the posterior distribution on this data point. For example, for a population size of $n = 7$, the algorithm predicts the following values:

Experiment A: $\mu_A[7] = 1.6897 \pm 0.8753,\ \sigma_A^2[7] = 3.3985 \pm 1.4113$
Experiment B: $\mu_B[7] = 1.6791 \pm 0.8362,\ \sigma_B^2[7] = 2.9536 \pm 1.3157$
Experiment C: $\mu_C[7] = 1.9841 \pm 0.5433,\ \sigma_C^2[7] = 2.9034 \pm 1.1156.$

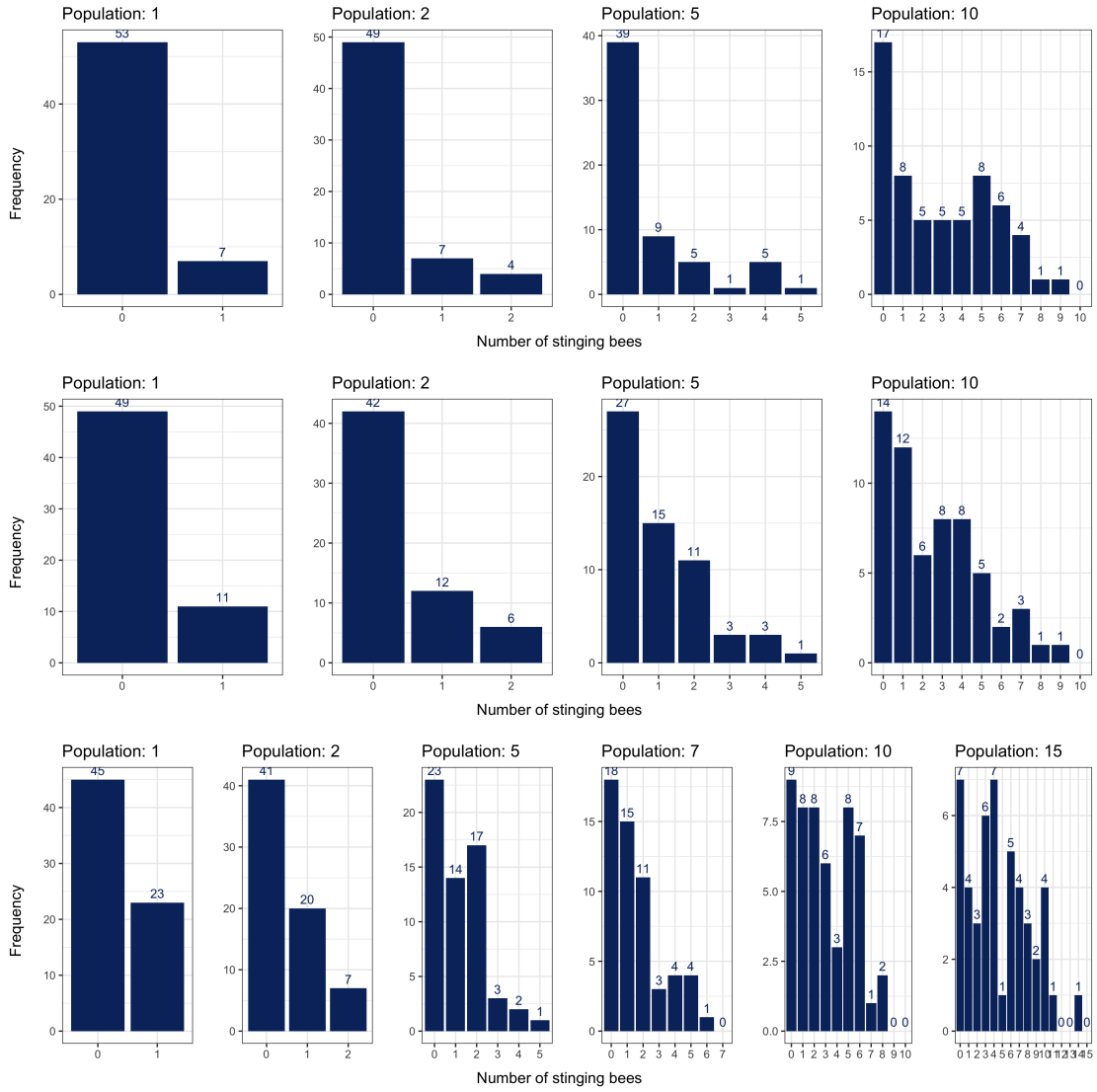
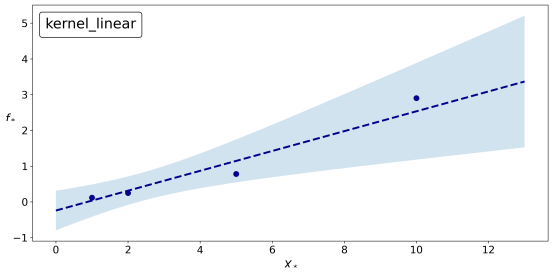
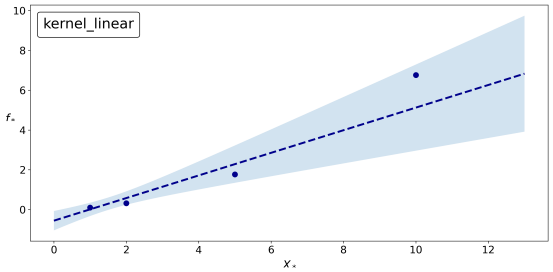


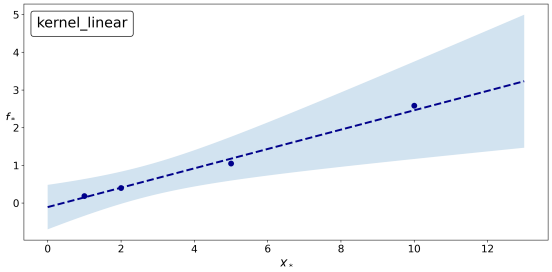
Figure 12 Experimental data of honeybees for experiments A, B, and C. Histograms count the number of stinging bees for each population size.
Experiment A: $N = [1, 2, 5, 10]$ bees, sample size $M = 60$.
Experiment B: $N = [1, 2, 5, 10]$ bees, sample size $M = 60$.
Experiment C: $N = [1, 2, 5, 7, 10, 15]$ bees, sample sizes $M = [68, 68, 60, 56, 52, 48]$.



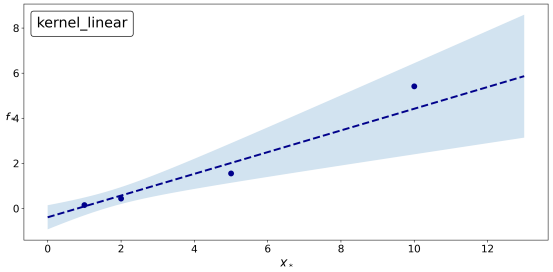
(a) GPR posterior of mean for Experiment A



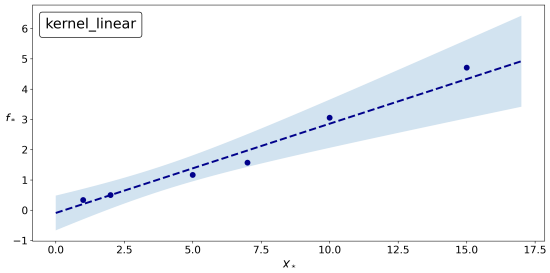
(b) GPR posterior of variance for Experiment A



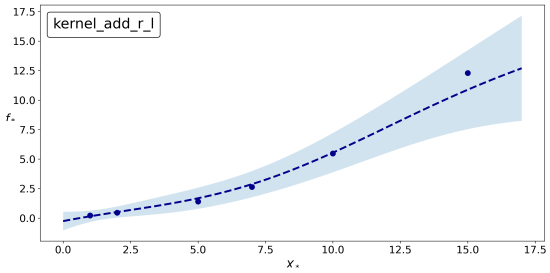
(c) GPR posterior of mean for Experiment B



(d) GPR posterior of variance for Experiment B



(e) GPR posterior of mean for Experiment C

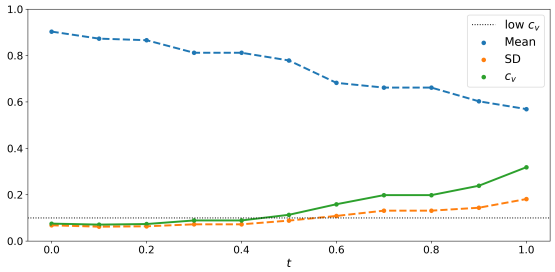


(f) GPR posterior of variance for Experiment C

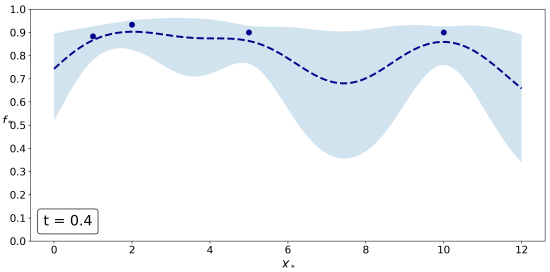
Figure 13 Posterior distributions for mean and variance of histograms for experimental data. Points are training data points, dashed lines are predictive means and shaded areas are 95% confidence regions. Best results according to model selection using LOOCV:
Experiment A: Mean with linear kernel ($MSE = 0.86284$), variance with linear kernel ($MSE = 4.28144$).
Experiment B: Mean with linear kernel ($MSE = 0.18207$), variance with linear kernel ($MSE = 1.86957$).
Experiment C: Mean with linear kernel ($MSE = 0.17253$), variance with addition of RBF and linear kernel ($MSE = 4.42797$).

Following this analysis, we aim to infer the fitness function of the collective. In this case, we want to investigate if there is always a certain proportion of the population defending the whole collective. The corresponding property can be rephrased as: *at least* $(100 \cdot t)\%$ *of the colony survives* for an unknown threshold t that represents the proportion of surviving bees. Therefore, we first analyse the

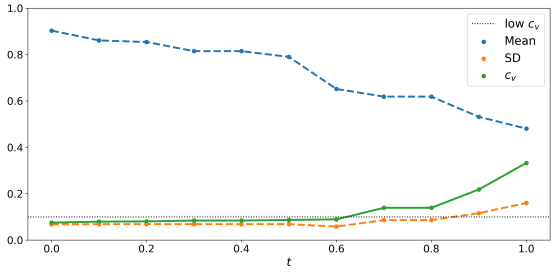
satisfaction of this property for different thresholds $t \in \mathbb{R}$ over varying population sizes. Subsequently, we extract the largest t for which the fitness function gives stable, reliable results according to the satisfaction probabilities. Here, we aim for the largest t, because our definition of the property assures that it is satisfied for all lower values of t as well. As described in Section 5.2, we compare the coefficients of variation, and state that a low $c_v < 0.1$ represents a distribution with little variation.



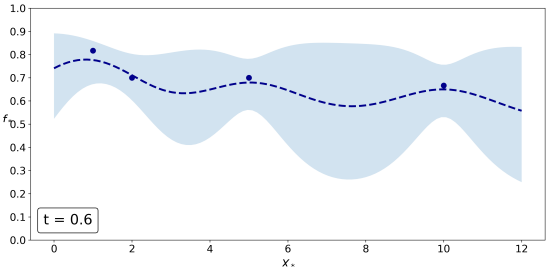
(a) Coefficient of variation for Experiment A



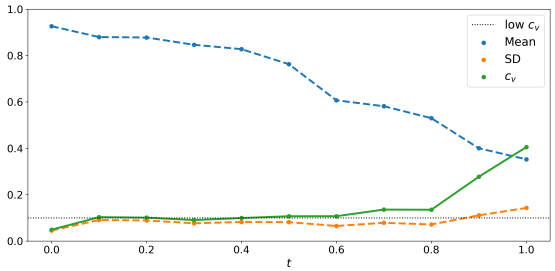
(b) SMMC posterior for Experiment A



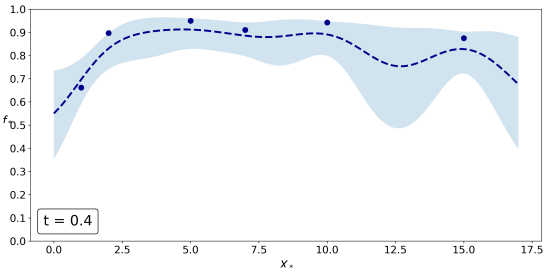
(c) Coefficient of variation for Experiment B



(d) SMMC posterior for Experiment B



(e) Coefficient of variation for Experiment C



(f) SMMC posterior for Experiment C

Figure 14 SMMC of experimental data to analyse which fitness function the collective tries to optimise. Left: mean, standard deviation, and coefficient of variation of posterior distributions over varying t as measure of divergence. Right: SMMC posterior for largest t with $c_v(t) < 0.1$, showing the predictive posterior mean and 95% confidence regions.

We use SMMC to analyse the satisfaction probability for 11 equally-spaced thresholds $t \in [0, 1]$. The inputs to SMMC are the number of observations for which the property is satisfied, read from the respective histogram. The covariance matrices are calculated with the RBF kernel with fixed parameters at $\sigma^2 = 0.25$ and $\ell = 1.5$ for experiment A and B, and $\sigma^2 = 0.2$ and $\ell = 1.75$ for experiment C. In Figure 14, we show on the left side the coefficients of variation c_v for different thresholds t together with the mean and standard deviation of the posterior distribution. On the right side, the SMMC posterior of the largest value of t with $c_v(t) < 0.1$ is visualised.

The results indicate the most plausible value of t to be $t = 0.4$ for experiment A, $t = 0.6$ for experiment B, and $t = 0.4$ for experiment C, what corresponds to the fitness function that at least 40% (60%) of the colony survives. Put differently, we obtain relatively high satisfaction probabilities over all population sizes between 0 and 12 (17) with these detected values for t. The corresponding MSE values are $\text{MSE}_A = 0.0988$, $\text{MSE}_B = 0.0923$, and $\text{MSE}_C = 0.0688$.

6.2 Simulated Data of CRN

Second, we consider the context where we have an underlying model in form of a CRN that we cannot perform classical model checking on, but simulate it to generate data. The CRN is implemented and simulated using a tool called StochNetV2 that was developed for stochastic simulations with CRNs [51]. Originally, as part of StochNetV2, the system contained four reactions, but here we adapt it to containing only two reactions. The advantage of this CRN model is that we can simply simulate it for different amounts of species (here: bees), instead of having to create a new model every time. Bee behaviour is modelled by the following two reactions:

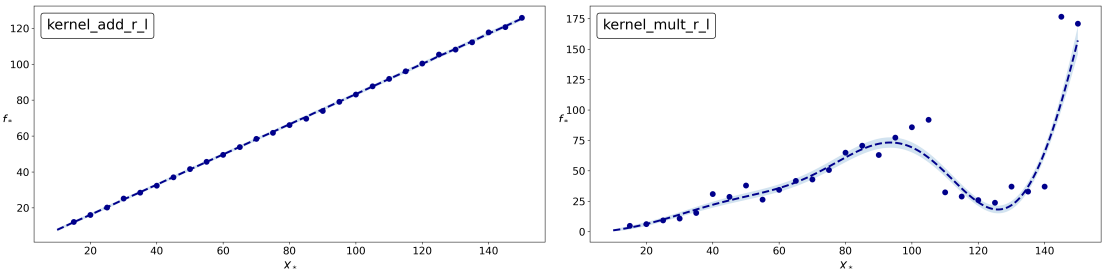
1. Stinging: $Bee + P \xrightarrow{k_1} BeeD + 2P$
2. Pheromone degradation: $P \xrightarrow{k_2} \varnothing$

with Bee denoting bees, $BeeD$ dead bees, and P the amount of pheromone in the system. Therefore, we have a stochastic model that simulates the real-world phenomenon of the social feedback mechanism in honeybees.
Simulating the CRN one time for one specific initial setting corresponds to one trial of an experiment. After the simulation, we get the final number of living

bees, dead bees, and pheromone as a result.

First, we simulate the CRN with fixed parameters to analyse the collective response for different population sizes. We assume that the stinging rate depends on the number of bees in the system, hence $k_1 = \frac{1}{N}$, and a constant pheromone degradation rate of $k_2 = 1$. The CRN is run $M = 100$ times for each of 28 different population sizes $N \in [15, 150]$ with the initial number of pheromone being equal to the number of bees, $P(0) = Bee(0) = N$. As a result, the final number of stinging bees after each simulation is saved to obtain histogram data similar to the experimental data in Section 6.1.

GPR is applied to predict the collective response as the number of stinging bees for populations of different sizes. See Figure 15 for the predictive posterior distributions for mean and variance of the number of stinging bees for different population sizes. Hyperparameters were optimised and the best kernel was selected using LOOCV.



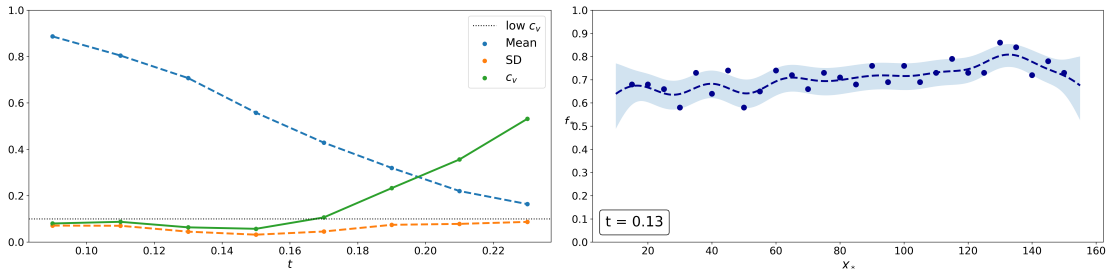
(a) GPR posterior of mean for CRN (b) GPR posterior of variance for CRN

Figure 15 Posterior distributions for mean and variance of results for simulated CRN data. Points are training data points, dashed lines are predictive means and shaded areas are 95% confidence regions. Best results according to model selection using LOOCV: Mean with addition of RBF and linear kernel ($MSE = 0.28419$), variance with multiplication of RBF and linear kernel ($MSE = 528.17589$).

The posterior distribution of the mean values indicates a linear trend of the number of stinging bees in relation to the total population size. This result confirms the expected behaviour of the CRN according to the scaling of the stinging rate. In contrast, the posterior distribution of the variances has an unexpected shape. Instead of following a linear trend as well, since we get less samples per bin for larger population sizes and would need to counteract with

larger sample sizes, the curve drops for population sizes around $N = 120$, and then increases again.

Next, we analyse the CRN with SMMC using the same data set to examine the fitness function the collective tries to optimise. Again, we aim to infer the unknown threshold t of the property: *at least* $(100 \cdot t)\%$ *of the colony survives.* For eight different thresholds $t \in [0.09, 0.23]$, we compute the number of simulation runs which satisfy the property to generate the input to SMMC. The covariance matrix is calculated with the RBF kernel with fixed parameters at $\sigma^2 = 0.15$ and $\ell = 10$. In Figure 16 (a), the coefficients of variation c_v are shown together with mean and standard deviation of the resulting posterior distributions. Note that for the largest value of t with $c_v(t) < 0.1$, which is $t = 0.15$, the corresponding mean value is only slightly above 0.5. Therefore, one previous value of t is more probable, and Figure 16 (b) displays the SMMC posterior of $t = 0.13$. This result indicates that the most plausible fitness function with respect to the simulated data for this CRN is that at least 13% of the colony survives.



(a) Coefficient of variation for CRN

(b) SMMC posterior for CRN

Figure 16 SMMC of simulated CRN data to analyse which fitness function the collective tries to optimise. Left: mean, standard deviation, and coefficient of variation of posterior distributions over varying $t \in [0.09, 0.23]$ as measure of divergence. Right: SMMC posterior for second largest t with $c_v(t) < 0.1$, showing the predictive posterior mean (MSE = 0.0087) and 95% confidence regions.

Finally, we demonstrate how to use the framework for finding parameters of a stochastic model. This application is useful when we have expert knowledge about the biological phenomenon, and want to design a model that best represents this dynamics. Assume that an expert of honeybees declares that a colony always tries to keep at least 11% of the population alive, and that our CRN is capable of capturing the behaviour of bees. Then, our goal is to find the stinging parameter k_1 for which the specified fitness function is most robustly satisfied over varying

population sizes. With this parameter, we can design a model that best represents the real-world dynamics of a collective of honeybees.

The following approach is executed: With a fixed pheromone degradation rate of $k_2 = 1$, and 15 equally-spaced stinging rates $k_1 \in [0.01, 0.07]$, the CRN is run for $M = 1000$ simulations for 15 different population sizes $N \in [15, 85]$. Hence, we get a training set for two uncertain parameters, k_1 and N, that contains the final number of bees, dead bees, and pheromone after each simulation run. For the proposed fitness function, we set $t = 0.11$, and compute the number of runs satisfying the property. These outcomes are analysed using two-dimensional SMMC to check the fitness function *at least 11% of the colony survives* for varying population sizes and varying stinging rates. The RBF-ARD kernel is used with fixed hyperparameters at $\sigma^2 = 0.0005$ and $\ell = [10, 0.01]$.

In Figure 17, the left plot shows the 225 training points for SMMC. In the right plot, the resulting posterior distribution is visualised.

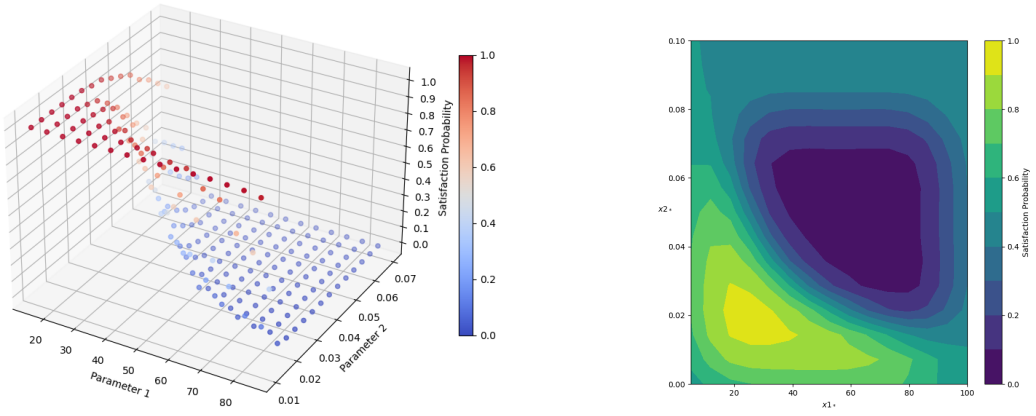


Figure 17 2-dimensional SMMC of CRN for varying population size $x1 \in [15, 85]$ and varying stinging rate $x2 \in [0.01, 0.07]$. Posterior distribution is derived with the RBF-ARD kernel (MSE = 0.3219) with fixed parameters at $\sigma^2 = 0.0005$, $\ell = [10, 0.01]$. Left: training data consisting of satisfactions of the fitness function with threshold $t = 0.11$ for 1000 observations per input point. Right: Predictive posterior probabilities for test set in contour plot.

The results suggest that a scaling of the stinging rate by $k_1 = \frac{1}{N}$ that was implemented in the first two examples of this section, could be not enough to achieve the proposed fitness function. Instead, k_1 should start at a lower value for small population sizes, and then slowly decrease to achieve a robust satisfaction probability of the fitness function.

6.3 Parametric Markov Chain

In the last experimental context, we cover the application of our workflows to an uncertain stochastic model. See Figure 18 for a parametric population DTMC that models the collective behaviour of a population of three bees, first published by Hajnal et al. [26]. The DTMC contains three parameters, q_0, q_1, and q_2, and four BSCCs that represent the final states of the agents, in which a living bee is denoted by "0", and a stinging bee by "1".

The model is implemented in PRISM and various properties written in PCTL can be verified. Analysing the collective response for different population sizes cannot be easily done, since the model would need to be re-defined for every population size to capture the dynamics. Note that this refers to one of the weaknesses mentioned for PMC, and that additionally stochastic models for larger populations would suffer from the state space explosion problem.

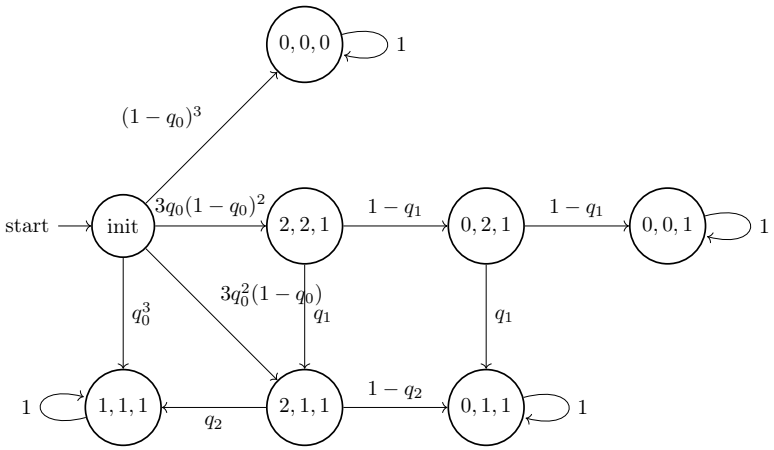


Figure 18 Population parametric DTMC for colony of $n = 3$ bees.

However, we can use SMMC to find parameters of the chain such that a given property is satisfied with a high probability.

First, we want to find out how the value of q_0 influences the satisfaction probability of finally having two stinging, and one not stinging bee. Therefore, we check the property $P_{=?}\ F\ (0, 1, 1)$, i.e. eventually being in the BSCC $(0, 1, 1)$, as a function of q_0. Parameters q_1 and q_2 are fixed at $q_1 = 0.7$ and $q_2 = 0.2$. We simulate the DTMC for 15 values of $q_0 \in [0, 1]$ and generate $M = 50$ trajectories for each value of q_0 to obtain the training data for SMMC. The RBF kernel is used to calculate the covariance matrix with fixed hyperparameters at $\sigma^2 = 0.05$ and $\ell = 0.1$.

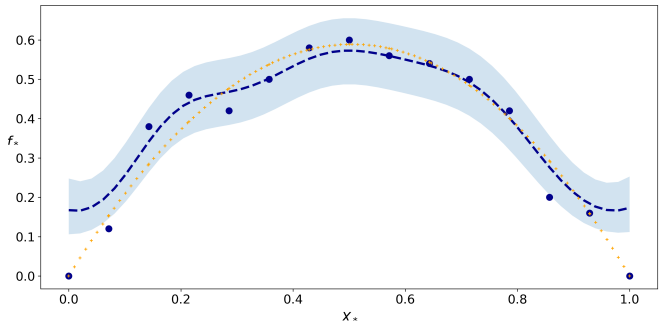


Figure 19 SMMC posterior distribution (MSE = 0.06435) of parametric DTMC modelling the satisfaction probability of eventually two stinging bees for varying parameter $q_0 \in [0, 1]$. Points are training data points, dashed line is predictive mean, and shaded area is 95% confidence region. The orange plus symbols are the result of probabilistic model checking for 100 values of $q_0 \in [0, 1]$.

Looking at the posterior distribution in Figure 19, we see that the highest probabilities are obtained for q_0 being around 0.5. The results of PMC, obtained for 100 values of $q_0 \in [0, 1]$, confirm this assumption that $q_0 = 0.5$ gives the highest satisfaction probability for having eventually two stinging bees.

Accordingly, we can use two-dimensional SMMC to infer parameters q_0 and q_1 at the same time. Again, we fix $q_2 = 0.2$, and check the property to eventually have two stinging bees in the system. $M = 50$ trajectories are generated for the DTMC with 10 equally-spaced values of $q_0 \in [0.1, 1]$ and $q_1 \in [0, 1.1]$. The resulting training set for two uncertain parameters consists of 100 data points. The RBF-ARD kernel is used with fixed hyperparameters at $\sigma^2 = 0.05$ and $\ell = [0.1, 0.1]$ In Figure 20, the left plot shows the satisfaction probabilities of the training points, and the right plot the resulting posterior distribution.

Note that the highest satisfaction probabilities are achieved for q_0 being around 0.43, and q_1 around 0.91. This means that we can model the outcome of two stinging bees most reliably by setting $q_0 = 0.43$ and $q_1 = 0.91$, if we know that $q_2 = 0.7$.

6.4 Relationship of Stochastic and Deterministic Model

In this section, we make a brief remark on the relationship between the stochastic model of the honeybee example, i.e. the CRN, and the deterministic model, i.e. the

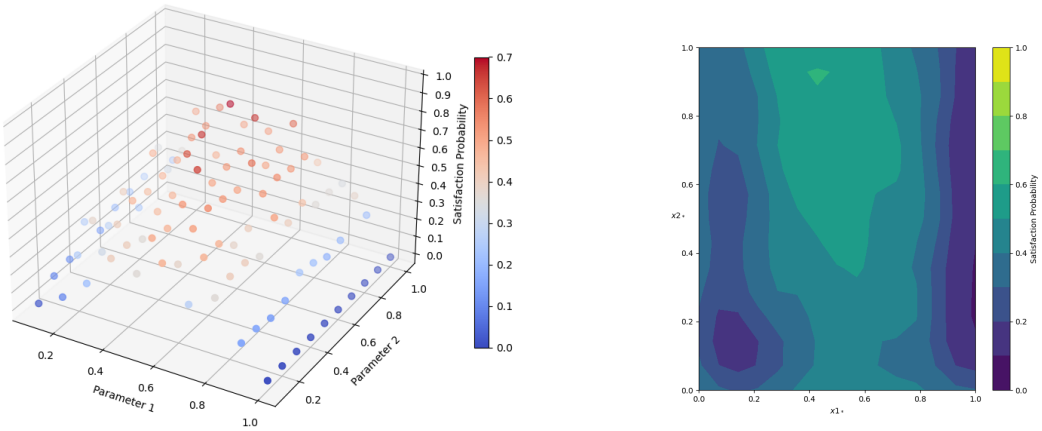


Figure 20 Two-dimensional SMMC of parametric DTMC for varying parameters $q_0 \in [0, 1]$ and $q_1 \in [0, 1]$. Posterior distribution is derived with the RBF-ARD kernel with fixed parameters at $\sigma^2 = 0.05$, $\ell = [0.1, 0.1]$. Left: training data consisting of satisfactions of eventually having two stinging bees for 50 observations per input point. Right: Predictive posterior probabilities for test set in contour plot (MSE = 0.0491).

corresponding system of ODEs. The theoretical background of this relationship was described by Petrov [43].

There are two different ways of looking at the dynamics of a biological system. First, as we did before, we can analyse the stochastic model by simulating data for many trajectories and calculating the expected number of stinging bees after some timestep, for example $t = 100s$. Dividing this expected value by the number of species, here bees, gives a fraction that accounts for the general case, regardless of the total population size.

Second, we can find a numerical solution for the simplest case of the according deterministic model, the ODE model. We can translate the CRN to the ODE model by setting up the deterministic reaction rates. For an initial state $\mathbf{z}_0 = (z_1, ..., z_n) \in \mathbb{R}^n$ and the two consumption vectors $a_1 = (1, 0, 1)$, $a_2 = (0, 0, 1)$, we get the following reaction rates:

$$\tilde{\lambda_1}(\mathbf{z}) = k_1(z_1^{a_{11}} \cdot z_2^{a_{12}} \cdot z_3^{a_{13}}) = k_1(z_1^1 \cdot z_2^0 \cdot z_3^1) = k_1(z_1 z_3)$$
$$\tilde{\lambda_2}(\mathbf{z}) = k_2(z_1^{a_{21}} \cdot z_2^{a_{22}} \cdot z_3^{a_{23}}) = k_2(z_1^0 \cdot z_2^0 \cdot z_3^1) = k_2\ z_3$$

From this, we infer three coupled differential equations, satisfying $\mathbf{z}_0$:

$$\frac{d}{dt}z_1(t) = v_{11}\tilde{\lambda_1}(\mathbf{z}(t)) + v_{12}\tilde{\lambda_2}(\mathbf{z}(t)) = -1k_1z_1(t)z_3(t)$$
$$\frac{d}{dt}z_2(t) = v_{21}\tilde{\lambda_1}(\mathbf{z}(t)) + v_{22}\tilde{\lambda_2}(\mathbf{z}(t)) = 1k_1z_1(t)z_3(t)$$
$$\frac{d}{dt}z_3(t) = v_{31}\tilde{\lambda_1}(\mathbf{z}(t)) + v_{32}\tilde{\lambda_2}(\mathbf{z}(t)) = 1k_1z_1(t)z_3(t) - 1k_2z_3(t)$$

In the simplest case, there is initially one bee and one amount of pheromone in the system, $\mathbf{z}_0 = (1, 0, 1)$, and the two reactions both happen with rate $k_1 = k_2 = 1$. The numerical solution for the resulting system is shown in Figure 21. The system stabilises with around 0.1589 bees, 0.8414 dead bees and 0 amount of pheromone.

In order to compare this with the stochastic model, we need to convert the rates. Usually, stochastic rates depend on the volume of a system, and can therefore be scaled by the number of species in the system. Hence, we scale the stochastic reaction rate for stinging by the total number of bees in the system, i.e. $c_1 = k_1/N$. Note that this is the rate we implemented in Section 6.2 to analyse the CRN. For the initial state $\mathbf{x}_0 = (x_1, ..., x_n) \in \mathbb{N}^n = (100, 0, 100)$, we simulate the stochastic system 10000 times and collect the number of bees, dead bees and pheromone. In Figure 21, we visualise the expected numbers divided by $N = 100$ over time, and obtain around 0.1626 bees, 0.8374 dead bees, and 0 amount of pheromone for the stabilised system.

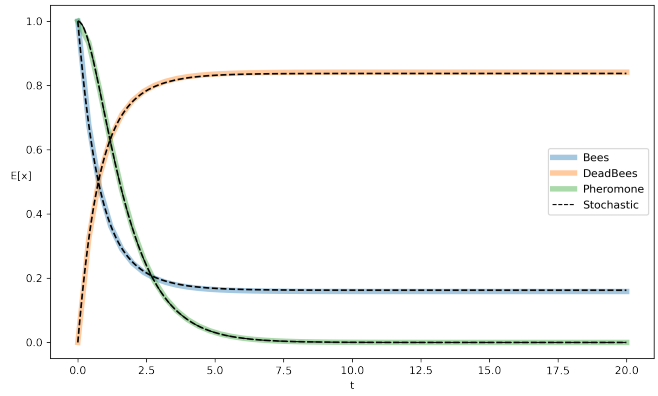


Figure 21 Solutions of ODE model (coloured lines) up to time $t = 20$ with initially 1 bee, 0 dead bees, and 1 amount of pheromone, compared to expected mean of stochastic model (black dashed lines) for 100 bees, 0 dead bees, and 100 amount of pheromone, scaled by $1/N$.

These results suggests that the limit of the stochastic model with a scaled rate of $c_1 = k_1/N$ is similar to the solution of the ODE model. In biological terms, this would indicate that each single bee is aware of the fraction of bees that have already stung in its colony. In turn, this supports the hypothesis that the fitness function of the collective is simply to keep a certain fraction of the population alive. In Section 6.2, we concluded that a scaling of the stinging rate by $c_1 = k_1/N$ would not suffice to achieve the desired dynamics of the system. One reason could be that the sample size we used for SMMC was not large enough to cover the stochasticity of the CRN, but a solid interpretation of the observed results would require a more detailed analysis.

CHAPTER 7
Discussion

Modelling and analysing collective behaviour is a complex task that involves different types of uncertainty, whether due to a lack of data, insufficient expert knowledge, or the stochasticty inherent in the behaviour itself, and hence the underlying model. In this thesis, we show applications of Gaussian Processes, a promising approach in Machine Learning, in combination with formal methods to model the collective behaviour of animal populations. The wide applicability of the approach is demonstrated on three different frameworks addressing the three research questions stated in Chapter 3 in three experimental contexts. In the following, we will discuss the results of the three workflows, advantages and limitations of the applied methodology, and possible directions for future research to complement the work.

Predicting the collective response using GPR is a first step towards extracting the behaviour of a collective from scarce data. We utilised summary statistics of experimental and simulated data of final states of agents as information about the performed response. Validation indicates good results with low MSEs for the predictions of mean and variance values of histograms for different population sizes. Therefore, we can use this method to get reliable predictions of the general response of the collective from only few observations. Without having to conduct new experiments for other population sizes, or building new models, we can infer a reliable estimate of the performed behaviour.

In the second framework, we used SMMC, an adaptation of GPC, to infer the fitness function a collective tries to optimise. We analysed the minimum number of agents exhibiting a behaviour as fitness function. The posterior distribution was derived for different parameters of the fitness function to find the parameter that results in overall high probabilities over different population sizes. Here we can see the advantage of SMMC in providing a smooth satisfaction function over all possible parameter values instead of only point-wise estimates. We computed a coefficient of variation to select the most plausible parameter of the fitness func-

tion according to a posterior distribution with low variation and high satisfaction probabilities. This approach enables us to get more knowledge about higher-level behaviour of a collective from observations of a few, discrete population sizes. Again, the results indicate good predictions of the satisfaction probabilities of the fitness function for populations of different sizes, as well as a reliable estimate of the parameter of the fitness function.
Finally, we aimed to infer the parameters of an underlying, uncertain model to represent an observed real-world phenomenon. For a proposed fitness function, we have shown that the algorithms finds a parameter of the model for which the satisfaction probability is most stable across different population sizes. Furthermore, we derived values for two parameters of a model such that a specified property is satisfied with high probability. Therefore, we are able to design models according to expert knowledge that exhibit a desired behaviour using two-dimensional SMMC.

All in all, we achieve high flexibility using model-agnostic methods based on Gaussian Processes. Whether we have experimental data, simulated data, a CRN, or an uncertain DTMC or CTMC, we can apply the introduced workflows to infer information about the behaviour of a collective, or to design a model that represents this behaviour. Having a non-parametric approach gives the advantage of getting good results without the need to specify the underlying distribution beforehand. The flexibility is also evident in the wide area of problems that can be solved, and is partly favoured by the automatic integration of PRISM. Although we focused on predicting the mean and variance of histograms using GPR, we can also apply this method on any other measurable entity. Similarly, applying SMMC is not restricted to fitness functions of the form *at least x% survive*, but can be used to analyse all kinds of verifiable properties of the system. In the running example, we looked at a temporal property of a CTMC, whereas in the case study we checked a PCTL property of a parametric DTMC. Therefore, we have shown with the presented frameworks how to combine the standard methods of formal verification with Gaussian Processes to achieve a flexible and powerful method in the analysis of biological collectives.

The superiority of GPR and GPC over other regression and classification methods is also visible in the automatic quantification of uncertainty. The methods provide not only an estimate as the mean predictive posterior distribution, but additionally confidence regions around the mean. Beyond that, compared to traditional neural

networks, Gaussian Processes seem to yield even better prediction accuracies [16]. In the examples presented in Chapter 6, we can clearly see that the uncertainty of the predictions decreases for larger sample sizes. However, having the confidence regions of the results allows us to make solid statements about the predictions even for small data sets.
The power of Gaussian Processes is reported in other studies that state, for example, that "it's hard to make the case that a GP shouldn't be involved as a component in a larger analysis [...] where ultimately limited knowledge of the modelling context can be met by a touch of flexibility" [23], hence emphasising the importance of uncertainty quantification in analyses of larger systems.

In spite of the power of the proposed frameworks, we have also encountered some limitations. First, predicting the mean and variance of histograms gives indeed a good approximation of the general behaviour, but there is only a restricted interpretation of the results possible. While we could predict the overall shape of the histogram for a known underlying functional form, for example a Normal distribution, we cannot give more detailed information about the exhibited behaviours of the collective without knowing the distribution.
A problem mentioned for the Expectation Propagation algorithm used in SMMC is that there is no guarantee that the iterations will converge [40], yet no major problems are reported. Choosing appropriate hyperparameters of the kernel is one way to tackle this issue. Due to time limitations, the hyperparameters were not optimised in the presented implementation of SMMC. Therefore, we cannot be completely sure that the obtained results reflect the best possible predictions. In future improvements of the framework it is desired to implement the optimisation algorithm to get more reliable results. Additionally, the results of SMMC can be significantly improved by collecting more data. Increasing the number of test points could also produce more smoothly functions, and therefore more precise predictions.

Beyond that, a possible direction for future work is to incorporate a weighing factor for more relevant value ranges. For example, we could shift the focus to smaller population sizes, and decrease the influence of values for larger population sizes, or vice versa.
More sophisticated is the future goal of improving the automatic learning of unknown parameters of a model, as well as the structure of the model itself. This would enable us to produce even more relevant results with less data. Gaussian

Processes seem powerful enough to achieve this goal, but the exact possibilities still need to be investigated. In addition, learning the underlying fitness function only from data, without previous assumptions, is a desired destination for future research.

In summary, combining Gaussian Processes with methods from formal verification seems like a promising direction towards modelling and analysing collective behaviour. Achieving high flexibility with the Machine Learning approach together with the sound methods for model verification results in powerful techniques that are applicable to various kinds of problem statements and experimental contexts. Furthermore, challenges of standard formal verification methods, especially PMC, can be faced, and established methods improved. We have shown different applications depending on the research question and available data, demonstrating the wide applicability of the proposed frameworks.

Bibliography

[1] Takumi Akazaki. Falsification of Conditional Safety Properties for Cyber-Physical Systems with Gaussian Process Regression. In *Runtime Verification*, volume 10012, pages 439–446. Springer International Publishing, Cham, 2016.

[2] Adnan Aziz, Kumud Sanwal, Vigyan Singhal, and Robert Brayton. Verifying continuous time Markov chains. In *International Conference on Computer Aided Verification*, volume 1102, pages 269–276, Berlin, Heidelberg, 1996. Springer.

[3] Christel Baier and Joost-Pieter Katoen. *Principles of Model Checking*. Principles of Model Checking. MIT Press, 2008.

[4] Christel Baier, Joost-Pieter Katoen, and Holger Hermanns. Approximative Symbolic Model Checking of Continuous-Time Markov Chains. In *International Conference on Concurrency Theory*, volume 1664, pages 146–161, Berlin, Heidelberg, 1999. Springer.

[5] Michael Baron. *Probability and statistics for computer scientists*. Chapman and Hall/CRC, 2014.

[6] Ezio Bartocci, Luca Bortolussi, Laura Nenzi, and Guido Sanguinetti. On the Robustness of Temporal Properties for Stochastic Models. *Electronic Proceedings in Theoretical Computer Science*, 125:3–19, 2013. doi: 10.4204/EPTCS.125.1.

[7] Ezio Bartocci, Luca Bortolussi, Laura Nenzi, and Guido Sanguinetti. System design of stochastic models using robustness of temporal properties. *Theoretical Computer Science*, 587:3–25, 2015. doi: 10.1016/j.tcs.2015.02.046.

[8] Christopher M. Bishop and Nasser M. Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, New York, 2006.

[9] Eric Bonabeau. Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences of the United States of America*, 99(3):7280–7287, 2002. doi: 10.1073/pnas.082080899.

[10] Luca Bortolussi and Luca Palmieri. Deep Abstractions of Chemical Reaction Networks. In *International Conference on Computational Methods in Systems Biology*, volume 11095, pages 21–38, Cham, 2018. Springer.

[11] 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 Berlin Heidelberg, 2013. doi: 10.1007/978-3-642-40196-1_7.

[12] Luca Bortolussi, Dimitrios Milios, and Guido Sanguinetti. U-Check: Model Checking and Parameter Synthesis Under Uncertainty. In *International Conference on Quantitative Evaluation of Systems*, pages 89–104, Cham, 2015. Springer. doi: 10.1007/978-3-319-22264-6_6.

[13] Luca Bortolussi, Dimitrios Milios, and Guido Sanguinetti. Smoothed model checking for uncertain continuous-time Markov chains. *Information and Computation*, 247:235–253, 2016.

[14] Luca Bortolussi, Roberta Lanciani, and Laura Nenzi. Model checking Markov population models by stochastic approximations. *Information and Computation*, 262:189–220, 2018.

[15] Luca Cardelli, Marta Kwiatkowska, and Luca Laurenti. Stochastic analysis of chemical reaction networks using linear noise approximation. *Biosystems*, 149:26–33, 2016.

[16] Tao Chen, Kunn Hadinoto, Wenjin Yan, and Yifei Ma. Efficient meta-modelling of complex process simulations with time–space-dependent outputs. *Computers & Chemical Engineering*, 35(3):502–509, 2011. doi: 10.1016/j.compchemeng.2010.05.013.

[17] Edmund M. Clarke, Thomas A Henzinger, Helmut Veith, and Roderick Bloem. *Handbook of model checking*, volume 10. Springer, Cham, 2018.

[18] Donald L. Deangelis and Stephanie G. Diaz. Decision-Making in Agent-Based Modeling: A Current Review and Future Prospectus. *Frontiers in Ecology and Evolution*, 6:237, 2019. doi: 10.3389/fevo.2018.00237.

[19] David Duvenaud. *Automatic model construction with Gaussian processes*. Doctoral dissertation, University of Cambridge, 2014. Publisher: Apollo - University of Cambridge Repository.

[20] De Faria Filho, An Dias, Alc Veloso, Cfd Bueno, Fap Couto, Jb Matos Júnior, Kzo Barreto, Pa Rodrigues, and Wa Carneiro. Classification of coefficients of variation in experiments with commercial layers. *Revista Brasileira de Ciência Avícola*, 12(4):255–257, 2010. doi: 10.1590/S1516-635X2010000400006.

[21] Daniel T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *The Journal of Physical Chemistry*, 81(25):2340–2361, 1977. doi: 10.1021/j100540a008.

[22] Daniel T Gillespie. Stochastic simulation of chemical kinetics. *Annu. Rev. Phys. Chem.*, 58:35–55, 2007.

[23] Robert B. Gramacy. *Surrogates: Gaussian process modeling, design, and optimization for the applied sciences*. Chapman and Hall/CRC, 2020.

[24] Jochen Görtler, Rebecca Kehlbeck, and Oliver Deussen. A visual exploration of gaussian processes. *Distill*, 4(4):e17, 2019.

[25] Ernst Moritz Hahn, Holger Hermanns, and Lijun Zhang. Probabilistic reachability for parametric Markov models. *International Journal on Software Tools for Technology Transfer*, 13(1):3–19, 2011.

[26] Matej Hajnal, Morgane Nouvian, David Šafránek, and Tatjana Petrov. Data-Informed Parameter Synthesis for Population Markov Chains. In *International Workshop on Hybrid Systems Biology*, pages 147–164, Cham, 2019. Springer. doi: 10.1007/978-3-030-28042-0_10.

[27] Hans Hansson and Bengt Jonsson. A logic for reasoning about time and reliability. *Formal Aspects of Computing*, 6(5):512–535, 1994. doi: 10.1007/BF01211866.

[28] John Jackson, Luca Laurenti, Eric Frew, and Morteza Lahijanian. Formal Verification of Unknown Dynamical Systems via Gaussian Process Regression. *arXiv:2201.00655 [cs, eess]*, 2021.

[29] Sebastian Junges, Erika Ábrahám, Christian Hensel, Nils Jansen, Joost-Pieter Katoen, Tim Quatmann, and Matthias Volk. Parameter synthesis for markov models. *arXiv preprint arXiv:1903.07993*, 2019.

[30] Joost-Pieter Katoen. The probabilistic model checking landscape. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 31–45, 2016.

[31] William Ogilvy 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. doi: 10.1098/rspa.1927.0118.

[32] Marta Kwiatkowska, Gethin Norman, and David Parker. Quantitative Analysis With the Probabilistic Model Checker PRISM. *Electronic Notes in Theoretical Computer Science*, 153(2):5–31, 2006. doi: 10.1016/j.entcs.2005.10.030.

[33] 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, Berlin, Heidelberg, 2007.

[34] 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, Berlin, Heidelberg, 2011. Springer.

[35] Marta Kwiatkowska, Gethin Norman, and David Parker. Probabilistic model checking: Advances and applications. In *Formal System Verification*, pages 73–121. Springer, Cham, 2018.

[36] Axel Legay, Kim Larsen, Uli Fahrenberg, and Benoît Delahaye. Refinement and Difference for Probabilistic Automata. *Logical Methods in Computer Science*, 10(3):11, 2014. doi: 10.2168/LMCS-10(3:11)2014.

[37] Oded Maler and Dejan Nickovic. Monitoring Temporal Properties of Continuous Signals. In *Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems*, volume 3253, pages 152–166. Springer, Berlin, Heidelberg, 2004.

[38] Paolo Milazzo. Analysis of COVID-19 Data with PRISM: Parameter Estimation and SIR Modelling. In *From Data to Models and Back*, pages 123–133. Springer, Cham, 2021. doi: 10.1007/978-3-030-70650-0_8.

[39] Gareth W. Molyneux and Alessandro Abate. ABC(SMC)^2: Simultaneous Inference and Model Checking of Chemical Reaction Networks. In *Inter-*

national Conference on Computational Methods in Systems Biology, pages 255–279, 2020. doi: 10.1007/978-3-030-60327-4_14.

[40] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

[41] Radford M. Neal. *Bayesian Learning for Neural Networks*. Number 118 in Lecture Notes in Statistics. Springer, New York, NY, 1996.

[42] Jagdish K. Patel and Campbell B. Read. *Handbook of the normal distribution*, volume 150. CRC Press, 1996.

[43] Tatjana Petrov. *Formal reductions of stochastic rule-based models of biochemical systems*. PhD thesis, ETH Zurich, 2013. `http://hdl.handle.net/20.500.11850/74227`.

[44] Paul Piho and Jane Hillston. Active and sparse methods in smoothed model checking. *arXiv:2104.09940 [cs, eess]*, April 2021.

[45] Frederico Pimentel Gomes. *Curso de estatística experimental*. Nobel, São Paulo, 13 edition, 2000.

[46] Elizabeth Polgreen, Viraj B Wijesuriya, Sofie Haesaert, and Alessandro Abate. Data-efficient Bayesian verification of parametric Markov chains. In *International Conference on Quantitative Evaluation of Systems*, pages 35–51. Springer, 2016.

[47] Zhaozhi Qian, Ahmed M. Alaa, and Mihaela van der Schaar. When and How to Lift the Lockdown? Global COVID-19 Scenario Analysis and Policy Assessment using Compartmental Gaussian Processes. *Advances in Neural Information Processing Systems*, pages 10729–10740, 2020.

[48] Arvind Ramanathan, Chad A. Steed, and Laura L. Pullum. Verification of Compartmental Epidemiological Models Using Metamorphic Testing, Model Checking and Visual Analytics. In *2012 ASE/IEEE International Conference on BioMedical Computing (BioMedCom)*, pages 68–73. IEEE, 2012. doi: 10.1109/BioMedCom.2012.18.

[49] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer school on machine learning*, pages 63–71, Berlin, Heidelberg, 2003. Springer.

[50] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian processes for machine learning*, volume 2 of *3*. MIT Press, Cambridge, MA, 2006.

[51] Denis Repin, Nhat-Huy Phung, and Tatjana Petrov. StochNetV2: A Tool for Automated Deep Abstractions for Stochastic Reaction Networks. In *International Conference on Quantitative Evaluation of Systems*, pages 27–32, Cham, 2020. Springer.

[52] Eric Schulz, Maarten Speekenbrink, and Andreas Krause. A tutorial on Gaussian process regression: Modelling, exploring, and exploiting functions. *Journal of Mathematical Psychology*, 85:1–16, 2018. doi: 10.1101/095190.

[53] Juan Zhang, Junping Yin, and Ruili Wang. Basic Framework and Main Methods of Uncertainty Quantification. *Mathematical Problems in Engineering*, 2020, 2020.

[54] Milan Češka, Petr Pilař, Nicola Paoletti, Luboš 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, Berlin, Heidelberg, 2016. Springer. doi: 10.1007/978-3-662-49674-9_21.