ABC(SMC)^2: Simultaneous inference and model checking of chemical reaction networks

Gareth W. Molyneux and Alessandro Abate

Department of Computer Science, University of Oxford, UK
firstname.lastname@cs.ox.ac.uk

Abstract. We present an approach that simultaneously infers model parameters while statistically verifying properties of interest to chemical reaction networks, which we observe through data and model as parametrised continuous-time Markov Chains. The new approach simultaneously integrates learning models from data, done by likelihood-free Bayesian inference, specifically Approximate Bayesian Computation, with formal verification over models, done by statistically model checking logical specifications expressed in CSL. The approach generates a probability (or credibility calculation) on whether the underlying chemical reaction network satisfies the CSL property of interest.

1 Introduction

Contribution We introduce a framework that integrates Bayesian inference and formal verification that additionally employs supervised machine learning, which allows for the model-based probabilistic verification of data-generating stochastic biological systems. The methodology perform data-driven inference of accurate models, which can contribute to the verification of whether or not the underlying stochastic system satisfies a given formal property of interest. Verification entails the estimation of the probability that models of the system satisfy a formal specification. Our framework accommodates partially known systems that might only generate finite, noisy observations. These systems are captured by parametric models, with uncertain rates within a known stoichiometry.

Related Work Bayesian inference techniques [10, 11] have been applied extensively to biological systems [41, 48]. Exact inference is in general difficult due to the intractability of the likelihood function, which has led to to likelihood-free methods such as Approximate Bayesian Computation (ABC) [43, 47]. [21] computes the probability that an underlying stochastic system satisfies a given property using data produced by the system and leveraging system's models. Along this line of work, the integration of verification of parameterised discrete-time Markov chains and Bayesian inference is considered in [37], with an extension to Markov decision processes in [38]. Both [37, 38] work with small finite-state models with fully observable traces, which allows the posterior probability distribution to be calculated analytically and parameters to be synthesised symbolically. On the contrary, here we work with partially observed data and stochastic models with intractable likelihoods, and must rely on likelihood-free methods and statistical parameter synthesis procedures. Building on previous work [35], which allowed for the likelhood-free Bayesian Verification of systems, the following framework is applicable to a wider variety of stochastic models.

Both probabilistic and statistical model checking have been applied to biological models [30,31,50], with tools for parameter synthesis [12,13]. Although the parameter synthesis approach in [12] rigorously calculates the satisfaction probability over the whole parameter space, this suffers from scalability issues. A Bayesian approach to statistical model checking is considered in [26] and partly inspires this work. Parametric verification has been considered from a statistical approach underpinned by Gaussian Processes: smoothed Model checking [6] provides an estimate of the satisfaction probability with uncertainty estimates, and has been used for parameter estimation from Boolean observations [8] and for parameter synthesis [9]. [3] proposes a methodology that, given a reachability specification, computes a related probability distribution on the parameter space, and an automaton-based adaptation of the ABC method is introduced to estimate it. Using the ABC method for model selection has been researched extensively in [2,46].

Approach Our framework is as follows (Section 3). Given a property of interest, a class of parametrised models and data from the underlying system, we simultaneously infer parameters and perform model-based statistical model checking. We then use a supervised machine learning method to determine regions of the parameter space that relates to models verifying the given property of interest. We integrate the generated posterior over these parameter synthesis regions, to quantify a probability (or credible calculation) on whether or not the system satisfies the given property. We apply this framework to Chemical Reaction Networks (CRNs) [22,48] (Section 4), which can be modelled by parametrised continuous-time Markov Chains [27], which represent the data-generating biological system (CRN). We argue that the alternative use of CRN data for black-box statistical model checking would be infeasible.

2 Background

2.1 Parametric Continuous-Time Markov Chains

Although our methodology can be applied to a number of stochastic models, in view of the applications of interest we work with discrete-state, continuous-time Markov chains [27].

Definition 1 (Continuous-time Markov Chain). *A continuous-time Markov chain (CTMC) $\mathcal{M}$ is a tuple (S, R, AP, L), where;*

- *S is a finite, non-empty set of states,*
- *s_0 is the initial state of the CTMC,*
- *$R : S \times S \to \mathbb{R}_{\geq 0}$ is the transition rate matrix, where $R(s, s')$ is the rate of transitioning from state s to state s',*

- *$L : S \to 2^{AP}$ is a labelling function mapping each state, $s \in S$, to the set $L(s) \subseteq AP$ of atomic propositions AP, that hold true in s.*

For the models in this paper, we assume s_0 is unique and deterministically given. The transition rate matrix R governs the dynamics of the overall model.

Definition 2 (Path of a CTMC). *Let $\mathcal{M} = (S, R, AP, L)$ be a CTMC. An infinite path of a CTMC $\mathcal{M}$ is a non-empty sequence $s_0t_0s_1t_1\dots$ where $R(s_i, s_{i+1}) > 0$ and $t_i \in \mathbb{R}_{>0}$ for all $i \geq 0$. A finite path is a sequence $s_0t_0s_1t_1\dots s_{k-1}t_{k-1}s_k$ such that s_k is absorbing. The value t_i represents the amount of time spent in the state s_i before jumping to the next state in the chain, namely state s_{i+1}. We denote by $\omega(i)$ the ith state of path ω, namely s_i, and $\omega@(t) = s_i$ the state occupied at time t. We denote by $Path^{\mathcal{M}}(s)$ the set of all (infinite or finite) paths of the CTMC $\mathcal{M}$ starting in state s. A trace of a CTMC is the mapping of a path through the labelling function L.*

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

Definition 3 (Parametric CTMC). *Let Θ be a parameter space. A parametric Continuous-time Markov Chain (pCTMC) over θ is a tuple (S, R_θ, AP, L):*

- *S, s_0, AP and L are as in Definition 1, and*
- *$\theta = (\theta_1, \dots, \theta_k)$ is the vector of parameters, taking values in a compact hyperrectangle $\Theta \subset \mathbb{R}^k_{\geq 0}$,*
- *$R_\theta : S \times S \to \mathbb{R}[\theta]$ is the parametric rate matrix, where $\mathbb{R}[\theta]$ denotes a set of polynomials over $\mathbb{R}^+$ with variables θ_k, $\theta \in \Theta$.*

Given a pCTMC and a parameter space Θ, we denote with $\mathcal{M}_\Theta$ the set $\{\mathcal{M}_\theta, \theta \in \Theta\}$ where $\mathcal{M}_\theta = (S, R_\theta, AP, L)$ is the instantiated CTMC obtained by replacing the parameters in R with their valuation in θ. So a standard CTMC is induced by selecting a specific parameter $\theta \in \Theta$: the sampled paths of an instantiated pCTMC $\mathcal{M}_\theta$ are denoted by ω_θ and are defined similarly to ω.

In this paper we work with Chemical Reaction Networks (CRNs), which have dynamics that can be modelled by CTMCs.

Definition 4 (Chemical Reaction Network). *A Chemical Reaction Network (CRN) $\mathcal{C}$ is a tuple $(M, X, W, \mathcal{R}, \boldsymbol{v})$, where*

- *$M = \{m_1, \dots, m_n\}$ is the set of n species,*
- *$X = (X_1, ..., X_n)$ is a vector where each X_i represents the number of molecules of each species i. $X \in W$, with $W \subseteq \mathbb{N}^N$ the state space,*
- *$\mathcal{R} = \{r_1, \dots, r_k\}$ is the set of chemical reactions, each of the form $r_j = (\boldsymbol{v}_j, \alpha_j)$, with $\boldsymbol{v}_j$ the stoichiometry vector of size n and $\alpha_j = \alpha_j(X, \upsilon_j)$ is the propensity or rate function,*

- *$\boldsymbol{\upsilon} = (\upsilon_1, \ldots, \upsilon_k)$ is the vector of (kinetic) parameters, taking values in a compact hyperrectangle $\Upsilon \subset \mathbb{R}^k$.*

Each reaction j of the CRN is represented as $r_j : \sum_{i=1}^{n} u_{j,i} m_i \xrightarrow{\alpha_j} \sum_{i=1}^{n} u'_{j,i} m_i$, where $u_{j,i}$ ($u'_{j,i}$) is the amount of species m_i consumed (produced) by reaction r_j. CRNs are used to model many biological processes and at the cellular level, can be modelled by CTMCs if we consider each state of the pCTMC to be a unique combination of the number of molecules, taking some initial molecule count X_0 to be the initial state of the pCTMC, $s_0 = X_0$. Parametrising the reaction rates within a CRN results in a parametric CRN (pCRN), which can be modelled as a pCTMC. For the rest of this paper, with a slight abuse in notation, we will let $\mathcal{M}_\theta$ be the pCTMC that represents a pCRN, where θ are the kinetic rates.

2.2 Properties - Continuous Stochastic Logic

We wish to verify properties over CRNs and their pCTMC models. We employ a time-bounded fragment of *continuous stochastic logic* (CSL) [1,30].

Definition 5. *Let ϕ be a CSL formula interpreted over states $s \in S$ of a parametrised model $\mathcal{M}_\theta$, and φ be a formula over its paths. Its syntax is*

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

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

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

$P_{\sim\zeta}[\varphi]$ holds if the probability of the path formula φ being satisfied from a given state meets $\sim \zeta$. Path formulas are defined by combining state formulas through temporal operators: $X^I\phi$ is true if ϕ holds in the next state whenever the next state of the Markov chain is reached at time $\tau \in I = [t, t']$, while $\phi_1 U^I \phi_2$ is true if ϕ_2 is satisfied at some $\tau \in I$ and ϕ_1 holds at all preceding time instants [30].

We define a *satisfaction function* to capture how the satisfaction probability of a given property over a model paths relates to its parameters and initial state.

Definition 6 (Satisfaction Function). *Let ϕ be a CSL formula, $\mathcal{M}_\theta$ be a parametrised model over a space Θ, s_0 is the initial state, and $Path^{\mathcal{M}_\theta}(s_0)$ is the set of all paths generated by $\mathcal{M}_\theta$ with initial state s_0. Denote by $\Lambda_\phi : \theta \to [0,1]$ the satisfaction function such that*

$$\Lambda_\phi(\theta) = P\left(\{\omega \in Path^{\mathcal{M}_\theta}(s_0) \models \varphi\} \mid \omega(0) = s_0\right), \tag{1}$$

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

2.3 Bayesian Inference

Given a set of observations or data, y_{obs}, a parametrised stochastic or deterministic model, $\mathcal{M}_\theta$, and prior information, the task of Bayesian inference is to learn the true parameter by means of its probability distribution. Prior beliefs about the model parameters, expressed through a probability distribution $\pi(\theta)$, are updated via y_{obs}, where assumptions on the model are encoded into the likelihood function $p(y_{obs}|\theta)$. Using Bayes' theorem, the posterior distribution is obtained as $\pi(\theta|y_{obs}) = p(y_{obs}|\theta)\pi(\theta)/\pi(y_{obs})$. When likelihood functions are intractable one can resort to likelihood-free methods, such as Approximate Bayesian Computation (ABC) [43], to approximate this posterior as $\pi_{ABC}(\theta|y_{obs}) \approx \pi(\theta|y_{obs})$.

Approximate Bayesian Computation ABC methods [43] produce an approximation to the posterior probability distribution when the likelihood $p(y|\theta)$ is intractable. The likelihood is approximated by matching simulated data $y \sim p(y|\theta)$ with the observed data y_{obs}, according to some function of the distance $\|y - y_{obs}\|$ or correspondingly over summary statistics of the simulated and observed data, namely $\|s - s_{obs}\|$.

Ideally, the observations y_{obs} are directly mapped to the variables of the model, which is endowed with sufficient statistics y. However, in many real world settings, model variables cannot be fully observed and outputs y are perturbed by noise due to measurement error. Since it is in general hard to identify a finite-dimensional set of sufficient statistics, it is common and computationally advantageous to use (insufficient) summary statistics $s = S(y)$, where function S performs a simplification of the signals y (e.g., averaging, smoothing, or sampling), which ideally are so that $\pi(\theta|y_{obs}) = \pi(\theta|s_{obs})$ [39].

The procedure generates samples $\theta^* \sim \pi(\theta)$, each of which is handled as follows: generating simulated data $y \sim p(y|\theta)$, the proposed sample θ^* is accepted if $\|y - y_{obs}\| \leq h$ for some $h \geq 0$, $h \in \mathbb{R}^+$, and rejected if $\|y - y_{obs}\| > h$. This procedure is equivalent to drawing a sample (θ, y) from the joint distribution

$$\pi_{ABC}(\theta, y|y_{obs}) \propto K_h(\|y - y_{obs}\|)p(y|\theta)\pi(\theta), \tag{2}$$

where $K_h(u)$ is a standard smoothing kernel function [42], which depends on a predetermined distance h and on $u = \|y - y_{obs}\|$. A standard choice we use for the smoothing kernel function is the indicator function, where $K_h(\|y - y_{obs}\|) = 1$ if $\|y - y_{obs}\| \leq h$, and $K_h(\|y - y_{obs}\|) = 0$ otherwise. Accordingly, the ABC approximation to the true posterior distribution is

$$\pi_{ABC}(\theta|y_{obs}) = \int \pi_{ABC}(\theta, y|y_{obs})dy. \tag{3}$$

As $h \to 0$, the samples from the true posterior distribution are obtained [43]:

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

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

$$\pi_{ABC}(\theta|s_{obs}) \propto \int K_h(\|s - s_{obs}\|)p(y|\theta)\pi(\theta)dy, \tag{4}$$

where, by slight abuse of notation, $K_h(\|s - s_{obs}\|)$ is defined as above.

Approximate Bayesian Computation - Sequential Monte Carlo The major issue with standard ABC is that if the prior $\pi(\theta)$ differs from the posterior distribution, $p(\theta|y_{obs})$, then the acceptance rates, namely the rates at which sampled parameters are accepted, will be low, thus resulting in more proposed parameters and associated simulations, which leads to an increase in computational burden. Approximate Bayesian Computation - Sequential Monte Carlo (ABCSMC) [46] techniques are developed to mitigate this issue. ABCSMC algorithms [45, 46] (cf. Appendix A) are designed to overcome this burden by constructing a sequence of slowly-changing intermediate distributions, $f_m(\theta)$, $m = 0, \ldots, M$, where $f_0(\theta) = \pi(\theta)$ is the initial sampling distribution and $f_M(\theta) = f(\theta)$ is the target distribution of interest, namely the approximated posterior, $\pi_{ABC}(\theta|s_{obs})$. A population of particles or samples from generation m, $\theta_m^{(i)}$, where $i = 1, \ldots, N$ is the number of particles, are propagated between these distributions sequentially, so that these intermediary distributions act as an importance sampling scheme [43], which is a technique used to sample from a distribution that over-weights specific regions of interest. This technique attempts to bridge the gap between the prior $\pi(\theta)$ and the (unknown) posterior $\pi(\theta|s_{obs})$. In the ABCSMC framework, a natural choice for the sequence of intermediary distributions is

$$f_m(\theta) = \pi_{ABC}^{h_m}(\theta, s|s_{obs}) \propto K_{h_m}(\|s - s_{obs}\|)p(y|\theta)\pi(\theta), \tag{5}$$

where $m = 0, \ldots, M$ and h_m is a monotonically decreasing sequence, namely such that $h_m > h_{m+1} \geq 0$. As above, K_{h_m} is the standard smoothing kernel, which now depends on the distance h_m. We expect that $\lim_{h_m \to 0} \pi_{ABC}^{h_m}(\theta|s_{obs}) = \pi(\theta|s_{obs})$ [43], and that, the more samples N are generated, the more accurate the approximated quantity will become.

A key part of the ABCSMC scheme is the generation of samples θ^* and the setting of weights (which is typical for other importance sampling schemes). Sample θ^* is initially ($m = 0$) taken from the prior and subsequently ($m > 0$) sampled from the intermediary distributions $f_{m-1}(\theta)$ through its corresponding weights (see below), as parameter $\theta_{m-1}^{(j)}$. Afterwards, θ^* is perturbed into θ^{**} by a kernel, $F_m(\theta^{**}|\theta^*)$. For the perturbed parameter, θ^{**}, a number of B_t simulations y_b, and in turn s^b, are generated from $p(y|\theta^{**})$, and the quantity $b_t(\theta^{**}) =$

$\sum_{b=1}^{B_t} K_{h_m}(\|s^b - s_{obs}\|)$ is calculated. If $b_t(\theta^{**}) = 0$, then θ^{**} is discarded and we resample θ^* again. Otherwise, the accepted θ^{**} results in the pair $\{\theta_m^{(i)}, w_m^{(i)}\}$, where the corresponding weights $w_m^{(i)}$ are set to

$$w_m^{(i)} = \begin{cases} b_t\left(\theta_m^{(i)}\right), & \text{if } m = 0 \\ \dfrac{\pi\left(\theta_m^{(i)}\right) b_t\left(\theta_m^{(i)}\right)}{\sum_{j=1}^{N} w_{m-1}^{(j)} F_m\left(\theta_m^{(i)} | \theta_{m-1}^{(j)}\right)}, & \text{if } m > 0 \end{cases} \tag{6}$$

and later normalised after calculating for each ith particle, $i = 1, \ldots, N$. If B_t is large, the estimate of $\pi_{ABC}(\theta|s_{obs})$ is accurate, which implies the acceptance probability is accurate but at the cost of many Monte Carlo draws. However, if B_t is small, the acceptance probability is highly variable but cheaper to evaluate [5].

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

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

2.4 Statistical Model Checking with the Massart Algorithm

Statistical model checking (SMC) techniques are used to estimate the validity of quantitative properties of probabilistic systems by simulating traces from an executable model of the system [32]. Unlike precise (up to numerics) probabilisitic model checking, SMC results are typically attained with statistical precision and can come, in particular, with confidence bounds (denoted below by δ) [14, 34]. In this work, we require Monte Carlo simulations to estimate the probability of properties of interest with a user-defined degree of accuracy (below ϵ). This can be obtained via standard concentration inequalities, such as the Chernoff [14] or the Okamoto [36] bounds. We wish to estimate a probability $\hat{\Lambda}_\phi(\theta)$ that approximates the unknown $\Lambda_\phi(\theta)$ within an absolute error ϵ and with a $(1 - \delta)$ confidence lower bound, namely

$$P(|\hat{\Lambda}_\phi(\theta) - \Lambda_\phi(\theta)| > \epsilon) \leq \delta. \tag{7}$$

For instance, the Okamoto bound ensures that drawing $n \geq n_{\mathcal{O}} = \lceil \frac{1}{2\epsilon^2} \log \frac{2}{\delta} \rceil$ simulations, results in an estimate $\hat{\Lambda}_\phi(\theta)$ with a statistical guarantee as in (7), where $\delta = 2\exp\left(-2n\epsilon^2\right)$.

In this work, we leverage the sharper Massart bounds [33]: we use the Sequential Massart algorithm [24,25] (described below), which progressively defines confidence intervals of the estimated probability and then applies the Massart bounds [33]. Massart bounds depend on the unknown probability $\Lambda_\phi(\theta)$ that we are estimating, which forces one to numerically evaluate with certainty an interval in which $\Lambda_\phi(\theta)$ evolves. Let us denote by $C(\Lambda_\phi(\theta), I)$ the coverage of $\Lambda_\phi(\theta)$ by a confidence interval I, i.e., the probability that $\Lambda_\phi(\theta) \in I$.

Theorem 1 (Absolute-Error Massart Bound with Coverage [25]). *Let $\hat{\Lambda}_\phi(\theta)$ be the probability estimated from n Monte Carlo simulations, ϵ be a given error, $\hat{\Lambda}^L_\phi(\theta)$ and $\hat{\Lambda}^U_\phi(\theta)$ be the lower and upper bounds of a confidence interval $I = [\hat{\Lambda}^L_\phi(\theta), \hat{\Lambda}^U_\phi(\theta)]$ and I^c be its complement within $[0, 1]$. The Massart bound is defined as*

$$P(|\hat{\Lambda}_\phi(\theta) - \Lambda_\phi(\theta)| > \epsilon) \leq 2 \exp\left(-n\epsilon^2 h_a(\Lambda_\phi(\theta), \epsilon)\right) + C(\Lambda_\phi(\theta), I^c), \qquad (8)$$

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

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

Theorem 2 ([25]). *Let α be a coverage parameter chosen such that $\alpha < \delta$ and $C(\Lambda_\phi(\theta), I^c) < \alpha$. Under the conditions of Theorem 1, a Monte Carlo algorithm $\mathcal{A}$ that outputs an estimate $\hat{\Lambda}_\phi(\theta)$ fulfils the condition in* (7) *if $n > \lceil \frac{1}{h_a(\Lambda_\phi(\theta),\epsilon)\epsilon^2} \log \frac{2}{\delta-\alpha} \rceil$.*

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

2.5 Bayesian Verification

In this work we extend the Bayesian Verification framework (cf. Fig. 1) introduced in [35], which addresses the following problem. Consider a data generating stochastic system $\mathcal{S}$, namely a CRN, where we denote the generated data as y_{obs}. We are interested in verifying a CSL property of interest ϕ over system $\mathcal{S}$ using sampled observations of the underlying system, y_{obs}, or a summary statistics $s_{obs} = S(y_{obs})$ thereof. We assume this goal cannot be reliably attained by means of statistical techniques directly applied on data y_{obs}. We thus plan to integrate model-based techniques (formal verification) with the use of data (Bayesian inference).

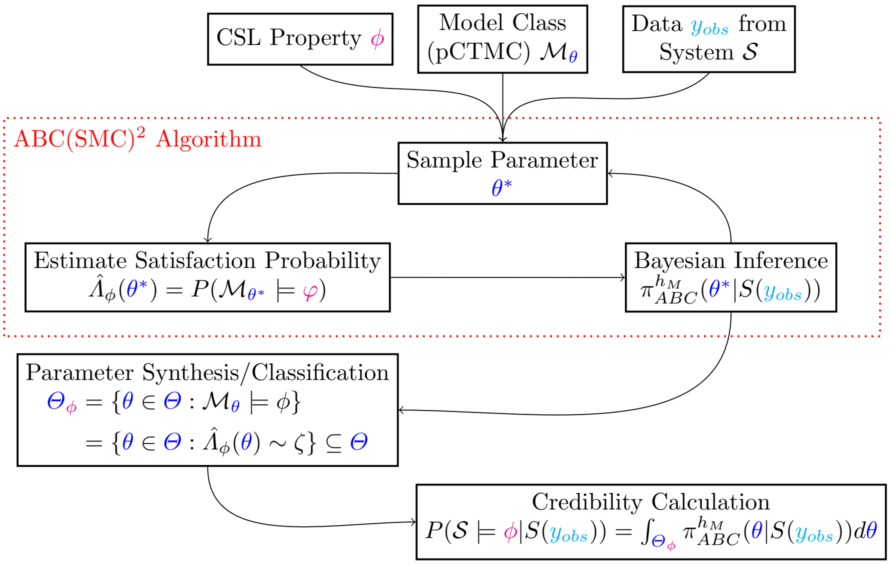

CSL Property ϕ
Model Class (pCTMC) $\mathcal{M}_\theta$
Data y_{obs} from System $\mathcal{S}$
ABC(SMC)^2 Algorithm
Sample Parameter θ^*
Estimate Satisfaction Probability $\hat{\Lambda}_\phi(\theta^*) = P(\mathcal{M}_{\theta^*} \models \varphi)$
Bayesian Inference $\pi^{h_M}_{ABC}(\theta^*|S(y_{obs}))$
Parameter Synthesis/Classification $\Theta_\phi = \{\theta \in \Theta : \mathcal{M}_\theta \models \phi\}$ $= \{\theta \in \Theta : \hat{\Lambda}_\phi(\theta) \sim \zeta\} \subseteq \Theta$
Credibility Calculation $P(\mathcal{S} \models \phi | S(y_{obs})) = \int_{\Theta_\phi} \pi^{h_M}_{ABC}(\theta|S(y_{obs}))d\theta$

Fig. 1. Bayesian Verification via ABC(SMC)^2.

Assume we have sufficient knowledge to propose a parametric model that adequately describes the underlying system, $\mathcal{M}_\theta$. We employ Bayesian inference to learn the posterior probability distribution of the model, namely $\pi(\theta|s_{obs})$ from (possibly scarce) data s_{obs}. We also use this parametric model to formally verify the property of interest ϕ, as follows. We synthesise two complementary parameter regions, $\Theta_\phi = \{\theta \in \Theta : \mathcal{M}_\theta \models \phi\}$ and $\Theta_{\neg\phi} = \{\theta \in \Theta : \mathcal{M}_\theta \not\models \phi\}$. We then integrate the inferred posterior probability distribution over Θ_ϕ to obtain the credibility calculation, which represents the probability that the underlying system $\mathcal{S}$ satisfies the property:

$$\mathcal{C} = P(\mathcal{S} \models \phi | s_{obs}) = \int_{\Theta_\phi} \pi(\theta|s_{obs})\mathrm{d}\theta, \tag{9}$$

which, if needed, can be estimated as needed via Monte Carlo methods. A complementary result can be drawn over $\Theta_{\neg\phi}$. The full procedure and further details are presented in [35] and summarised in Appendix B.

The limitations of the Bayesian Verification framework of [35] lie in the parameter synthesis part. Parameter synthesis of pCTMCs is considered in the work of [12], and accelerated by means of GPU processing in [13]. This and related probabilistic approaches to parameter synthesis are limited to finite-state systems that can be easily uniformised. In many practical applications they do not scale to realistic models. To address this limitation, we resort to statistical approaches (via SMC) for the parameter synthesis, similar to [9]. We formally integrate SMC techniques into the algorithm that perform Bayesian inference.

More precisely, we utilise the simulations needed in the ABCSMC algorithm to perform SMC, which yields the estimation of the probability of satisfying the property of interest, $\hat{\Lambda}_{\phi}(\theta)$. Whilst the ABCSMC algorithm rejects parts of the sampled parameters, we propose to retain these samples, and their corresponding simulations, to provide a classification of the parameter space. This is done by means of support vector machines. With these statistically-estimated parameter regions, we complete the Bayesian Verification framework, as per (9). The new framework (detailed in the next section), which employs models to extract information from the observation data s_{obs}, is now entirely based on simulations, which makes it usable with models of different size and structure.

3 ABC(SMC)^2: Approximate Bayesian Computation - Sequential Monte Carlo with Statistical Model Checking

We address the scalability limitations of our previous work [35], and specifically the parameter synthesis part: in [35] the synthesis was calculated symbolically, which practically limited the applicability to CTMCs with small state spaces and a few parameters. We incorporate statistical model checking within the Bayesian inference framework and estimate parameter regions. We name the modified algorithm Approximate Bayesian Computation - Sequential Monte Carlo with Statistical Model Checking: ABC(SMC)^2.

In the ABCSMC scheme (Algorithm 2 in Appendix A), a total of B_t simulations are performed for each sampled parameter θ^{**}, whether the sample is retained or not towards the approximate posterior $\pi^{h_M}_{ABC}(\theta|s_{obs})$: this leads to a considerable amount of wasted computational effort. We propose instead to statistically model check (SMC) each of the sampled parametrised models by means of the generated simulations, whilst parameter inference on the model is run (ABCSMC); we shall use the outcome of this algorithm for our Bayesian Verification framework, by classifying the parameter synthesis regions using statistical approaches.

At any of the M iterations, for each sampled point, $\theta^{**} \in \Theta$, we estimate the probability, $\hat{\Lambda}_{\phi}(\theta^{**}) \approx \Lambda_{\phi}(\theta^{**})$, with statistical guarantees, that an instantiated model $\mathcal{M}_{\theta^{**}}$ satisfies a given property of interest ϕ, namely $P(\mathcal{M}_{\theta^{**}} \models \varphi) = \hat{\Lambda}_{\phi}(\theta^{**})$. We then proceed with the ABCSMC algorithm as normal, calculating whether the sampled parameter θ^{*} contributes to the approximate posterior. In addition to producing samples $\{\theta^{(i)}_{h_M}, w^{(i)}_{h_M}\}$, which allows one to construct an approximation to the posterior distribution $\pi^{h_M}_{ABC}(\theta|s_{obs})$, the algorithm outputs $\left\{\theta^{**}, \hat{\Lambda}_{\phi}\left(\theta^{**}\right), \hat{\Lambda}^{L}_{\phi}\left(\theta^{**}\right), \hat{\Lambda}^{U}_{\phi}\left(\theta^{**}\right)\right\}$ for all the sampled parameters θ^{**} (whether accepted or not). These values are later used to train an SVM classifier to generate the parameter synthesis regions. We shall then integrate the approximate posterior over the parameter synthesis regions, to obtain a credibility calculation.

3.1 ABC(SMC)^2

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

$$\{\theta_M^{(i)}, w_M^{(i)}\} \sim \pi_{ABC}^{h_M}(\theta|s_{obs}) \propto \int K_{h_M}\left(\|s - s_{obs}\|\right) p(y|\theta)\pi(\theta)dy, \tag{10}$$

where $i = 1, \ldots, N$ is the number of particles used to approximate the posterior. For each parameter θ^{**}, simulation data is generated from the model $y_b \sim p(y|\theta^{**})$ to calculate $s^b = S(y_b)$, for a total of B_t times. In ABCSMC, if the sample is rejected, all the information gathered from simulation is simply discarded, which is a waste of performed calculations.

Instead, we propose using the corresponding simulations $y_b \sim p(y|\theta^{**})$ to calculate s^b and estimate $\Lambda_\phi(\theta^{**}) \approx \hat{\Lambda}_\phi(\theta^{**})$. We utilise the sequential Massart algorithm [25] presented in the previous section for this SMC procedure. We replace the number of simulations for each sampled parameter, B_t, with the calculated minimum number of samples estimated in the sequential Massart algorithm [25], $B_t = n \leq n_{\mathcal{O}}$, to calculate an estimated probability $\hat{\Lambda}_\phi(\theta^{**})$ with accuracy and confidence. We sample θ^{**} a total of R times, whether or not these samples are accepted as samples from the posterior at any generation m. For these sampled parameters, $\theta^{(r)}$, $r = 1, \ldots, R$, we estimate the corresponding probabilities corresponding mean estimated probabilities $\hat{\Lambda}_\phi\left(\theta^{(r)}\right)$ and $(1-\delta)$ uncertainty bounds: $\left\{\theta^{(r)}, \hat{\Lambda}_\phi\left(\theta^{(r)}\right), \hat{\Lambda}_\phi^L\left(\theta^{(r)}\right), \hat{\Lambda}_\phi^U\left(\theta^{(r)}\right)\right\}$, where $r = 1, \ldots, R$. Here R depends on the acceptance rate of the sampled parameters $\theta^{(r)}$, where $R \geq N \times M$, where N is the number of particles to sample and M is the total number of generations of the ABCSMC scheme. From this new algorithm, we obtain a set of weighted parameter vectors from the final generation M, $\{\theta_M^{(i)}, w_M^{(i)}\} \sim \pi_{ABC}^{h_M}(\theta|s_{obs})$, where $i = 1, \ldots, N$ as well as R sampled parameters and their corresponding estimated probabilities $\left\{\theta^{(r)}, \hat{\Lambda}_\phi\left(\theta^{(r)}\right), \hat{\Lambda}_\phi^L\left(\theta^{(r)}\right), \hat{\Lambda}_\phi^U\left(\theta^{(r)}\right)\right\}_{r=1}^{R}$.

We present the ABC(SMC)^2 scheme in Algorithm 1, with the $MASSART$ function corresponding to the Absolute-Error Massart Algorithm presented in Appendix C. The ABC(SMC)^2 algorithm takes as inputs a property of interest, ϕ, a prior probability distribution $\pi(\theta)$ an absolute-error tolerance ϵ as well as a coverage parameter α and confidence value δ. The output of the algorithm is a set of weighted parameter vectors $\left\{\theta_M^{(i)}, w_M^{(i)}\right\} \sim \pi_{ABC}^{h_M}(\theta|s_{obs})$ and a set of parameter vectors with corresponding estimated probability of satisfying a property of interest, $\left\{\theta^{(r)}, \hat{\Lambda}_\phi(\theta)^{(r)}, \hat{\Lambda}_\phi^L(\theta)^{(r)}, \hat{\Lambda}_\phi^U(\theta)^{(r)}\right\}$, which will be utilised for approximate parameter synthesis, as is discussed in the next section.

3.2 Approximate Parameter Synthesis via Statistical MC

The aim of parameter synthesis is to partition the parameter space Θ according to the satisfaction of the CSL property ϕ. Unlike the PMC-based synthesis in [35] (recalled in Sec. 2.5), we utilise a statistical approach to classify the

Algorithm 1 ABC(SMC)^2

Input:

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

Output:

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

```
1: Set $r = 0$
2: for $m = 0, \ldots, M$: do
3:   for $i = 0, \ldots, N$: do
4:     if $m = 0$ then
5:       Sample $\theta^{**} \sim \pi(\theta)$
6:     else
7:       Sample $\theta^*$ from the previous population $\{\theta_{m-1}^{(i)}\}$ with weights $\{w_{m-1}^{(i)}\}$ and perturb the
         particle to obtain $\theta^{**} \sim F_m(\theta|\theta^*)$
8:     end if.
9:     if $\pi(\theta^{**}) = 0$ then
10:      goto line 3
11:    end if
12:    Calculate $\left(\left\{\hat{\Lambda}_\phi(\theta^{**}), [\hat{\Lambda}_\phi^L(\theta^{**}), \hat{\Lambda}_\phi^U(\theta^{**})\right\}, B_t, \sum_{b=1}^{B_t} K_{h_m}(\|s^b - s_{obs}\|), \bar{d}\right)$ from the
       modified Massart Algorithm: $MASSART(\epsilon, \delta, \alpha, h_m, \theta^{**}, s_{obs})$
13:    Calculate $b_t(\theta^{**}) = \frac{1}{B_t}\sum_{b=1}^{B_t} K_{h_m}(\|s^b - s_{obs}\|)$
14:    Set $\left(\theta^{(r)}, \hat{\Lambda}_\phi\left(\theta^{(r)}\right), \hat{\Lambda}_\phi^L\left(\theta^{(r)}\right), \hat{\Lambda}_\phi^U\left(\theta^{(r)}\right)\right) = \left(\theta^{**}, \hat{\Lambda}_\phi(\theta^{**}), \hat{\Lambda}_\phi^L(\theta^{**}), \hat{\Lambda}_\phi^U(\theta^{**})\right)$
15:    $r \leftarrow r + 1$
16:    if $b_t(\theta^{**}) = 0$ then
17:      goto line 3
18:    end if
19:    Set $\theta_m^{(i)} = \theta^{**}$, $\bar{d}_m^{(i)} = \bar{d} = \frac{1}{B_t}\sum_{b=1}^{B_t}\|s^b - s_{obs}\|$ and calculate
20:    $$w_m^{(i)} = \begin{cases} b_t\left(\theta_m^{(i)}\right), & \text{if } m = 0 \\ \dfrac{\pi\left(\theta_m^{(i)}\right) b_t\left(\theta_m^{(i)}\right)}{\sum_{j=1}^N w_{m-1}^{(j)} F_m\left(\theta_m^{(i)}|\theta_{m-1}^{(j)}\right)}, & \text{if } m > 0 \end{cases}$$
21:  end for
22:  Normalise weights: $w_m^{(i)} \leftarrow w_m^{(i)} / \left(\sum_{i=1}^N w_m^{(i)}\right)$
23:  Set $h_{m+1} = (\upsilon/N)\sum_{i=1}^N \bar{d}_m^{(i)}$
24: end for
25: return $\left\{\left(\theta_M^{(i)}, w_M^{(i)}\right)\right\}_{i=1}^N$, $\left\{\theta^{(r)}, \hat{\Lambda}_\phi\left(\theta^{(r)}\right), \hat{\Lambda}_\phi^L\left(\theta^{(r)}\right), \hat{\Lambda}_\phi^U\left(\theta^{(r)}\right)\right\}_{r=1}^R$
```

parameter space, akin to [9]. So instead of employing the true satisfaction probability $\Lambda_\phi(\theta) \sim \zeta$ to determine Θ_ϕ (and its complement), we use $\hat{\Lambda}_\phi(\theta^{(r)})$, a statistical approximation computed at each sampled parameter point $\theta^{(r)}$. Evidently, recalling the confidence parameter δ, we should compute $\hat{\Lambda}_\phi(\theta^{(r)}) \sim \zeta \pm \epsilon$ (where the sign ± depends on the direction of the inequality $\sim$).

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

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

Notice that these formulas are a function of $\theta \in \Theta$. Since in the ABC(SMC)^2 procedure we generate a finite number of parameter samples $\theta^{(r)}$, which are biased towards the sought posterior distribution, there might be areas of the parameter space Θ that are insufficiently covered. We thus resort to supervised learning techniques to globally classify parameter synthesis regions. We utilise support vector machines (SVMs) [15,44] as a supervised learning classification technique. We train the SVM classifier on the data produced from the ABC(SMC)^2 algorithm, $\{\theta^{(r)}, \hat{\Lambda}_\phi(\theta^{(r)}), \hat{\Lambda}^L_\phi(\theta^{(r)}), \hat{\Lambda}^U_\phi(\theta^{(r)})\}$ where $r = 1, \ldots, R$. The SVM which is trained on this data then provides a non-linear classifying function, $\xi_\phi(\theta)$, where $\xi_\phi(\theta) = 1$ if $\theta \in \Theta_\phi, -1$ if $\theta \in \Theta_{\neg\phi}$, and 0 if $\theta \in \Theta_{\mathcal{U}}$.

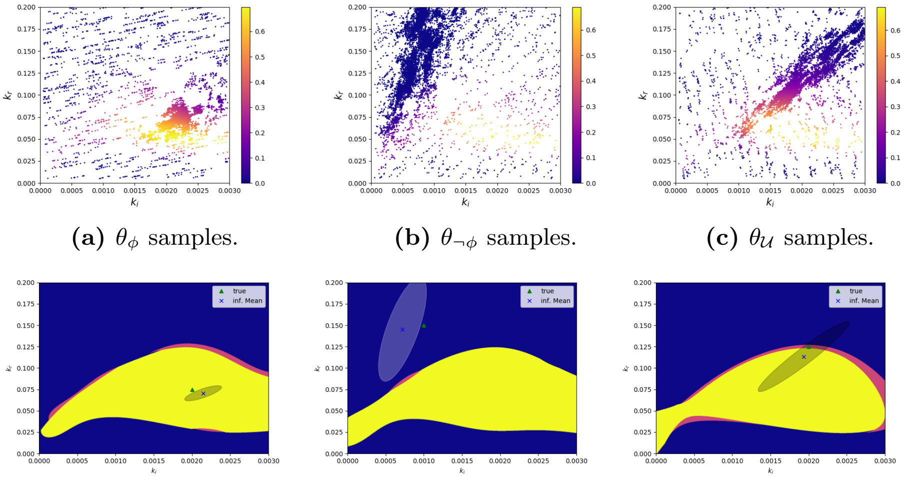
4 Experiments

Experimental Setup All experiments have been run on an Intel(R) Xeon(R) CPU E5-1660 v3 @ 3.00GHz, 16 cores with 16GB memory. ABC(SMC)^2 is coded in C++, while Python is used for the SVM classifier. Parameter synthesis is done via GPU-accelerated PRISM [13] and is shown in Figure 5a.

SIR System and Parameterised Model Towards an accessible explanation of the ABC(SMC)^2 algorithm, we consider the stochastic SIR epidemic model [28], which has the same structure (stoichiometry over species counts) as CRNs [13]. The model describes the dynamics of three epidemic types, a susceptible group (S), an infected group (I), and a recovered group of individuals (R) - here we let S, I and R evolve via the rules

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

This is governed by the rate parameters $\theta = (k_i, k_r)$, and each state of the pCTMC describes the combination of the number of each type (S, I, R) (this equates to molecule/species counts in CRNs). The initial state of the pCTMC is $s_0 = (S_0, I_0, R_0) = (95, 5, 0)$. We wish to verify the following property, $\phi = P_{>0.1}((I > 0)U^{[100,150]}(I = 0))$, i.e. whether, with a probability greater than 0.1, the infection dies out within a time interval between $t = 100$ and $t = 150$ seconds. We confine our parameters to the set $\Theta = [k_i^\perp, k_i^\top] \times [k_r^\perp, k_r^\top] = [5 \times 10^{-5}, 0.003] \times [0.005, 0.2]$. We generate observation data from the SIR model



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

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

Fig. 2. Bayesian Verification results from ABC(SMC)^2 for Case θ_ϕ (2a and 2d), Case $\theta_{\neg\phi}$(2b and 2e), and Case $\theta_{\mathcal{U}}$ (2c and 2f). Sampled points θ with estimated probabilities $\hat{\Lambda}_\phi(\theta)$ (2a, 2b and 2c). Inferred posterior $\pi^{h_M}_{ABC}(\theta|S(y_{obs}))$ and parameter regions (2d, 2e and 2f).

with three different parameter choices, corresponding to the CTMCs $\mathcal{M}_{\theta_\phi}$, $\mathcal{M}_{\theta_{\neg\phi}}$ and $\mathcal{M}_{\theta_{\mathcal{U}}}$, where $\theta_\phi = (0.002, 0.075)$, $\theta_{\neg\phi} = (0.001, 0.15)$ and $\theta_{\mathcal{U}} = (0.002, 0.125)$. From Figure 5a, we see that $\theta_\phi \in \Theta_\phi$, $\theta_{\neg\phi} \in \Theta_{\neg\phi}$, and finally $\theta_{\mathcal{U}}$ is near the borderline. These models will correspond to three "true" underlying stochastic systems $\mathcal{S}$, with associated observation data. For each instance, we work with observed data y_{obs} that is sampled at a finite number of time steps. The observed data consists of only 5 simulated traces, observed at 10 time points. The summary statistics $S(y_{obs}) = s_{obs}$ is the average of the 5 traces. It is worth emphasising that with so few observation traces, black-box SMC (directly based on observation traces, not on model-generated simulations) would be hopeless.

Application of ABC(SMC)^2 Algorithm Our algorithm outputs samples from the approximated posterior and their corresponding weights, $\left\{\theta^{(i)}_M, w^{(i)}_M\right\} \sim \pi^{h_M}_{ABC}(\theta|s_{obs})$ where $i = 1, \ldots, N$. By the strong law of large numbers, letting $\bar{\theta}_M = \sum_{i=1}^N \theta^{(i)}_M w^{(i)}_M$, $P\left(\lim_{N\to\infty} \sum_{i=1}^N w^{(i)}_M \theta^{(i)}_M - \mathbb{E}[\bar{\theta}_M] = 0\right) = 1$. Thus we assume that the approximated posterior can be modelled by a multivariate Normal distribution, $\pi^{h_M}_{ABC}(\theta|s_{obs}) \approx \mathcal{N}(\bar{\theta}_M, \Sigma_M)$, where the mean is given by $\bar{\theta}_M$ and the elements of the empirical covariance matrix are defined as

$$\Sigma_{M\,jk} = \frac{1}{1 - \sum_{i=1}^N (w^{(i)}_M)^2} \sum_{i=1}^N w^{(i)}_M \left(\theta^{(i)}_M - \bar{\theta}_M\right)_j \left(\theta^{(i)}_M - \bar{\theta}_M\right)_k .$$

We choose the number of samples to be $N = 500$; the number of sequential steps to be $M = 20$; the kernel function $K_h(u)$ to be a simple indicator function, i.e. $K_h(u) = 1$ if $u < h$, $K_h(u) = 0$ otherwise; the rate at which the thresholds h_m decrease to be $\upsilon = 0.5$; and the summary statistic $s = S(y)$ is chosen to be the sample mean of the simulations and of the observations. The perturbation kernel $F_m(\theta^{**}|\theta^*)$ is chosen to be a multivariate Normal distribution, so that $\theta^{**} \sim \mathcal{N}(\theta^*, 2\Sigma_{m-1})$, where the covariance is twice the second moment computed over the accepted weights and particles at step $m-1$, namely $\left\{\theta^{(i)}_{m-1}, w^{(i)}_{m-1}\right\}$, where $i = 1, \ldots, N$. For further details on alternative choices for threshold sequences, summary statistics and perturbation kernels, see [4,16,17,40,43]. We choose $\pi(\theta)$ to be a uniform prior over Θ.

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

Outcomes of ABC(SMC)^2 Algorithm For the three case studies, the inferred mean $\bar{\theta}_M$, covariance Σ_M, total number of sampled parameters ($\theta^{(r)}$, $r = 1, \ldots, R$) and resulting credibility calculation are given in Table 1, with corresponding runtimes in Table 3. Figures 2d, 2e and 2f plot the inferred posterior, showing the mean (denoted by ×) and 2 standard deviations from the mean (corresponding ellipse around the mean), as well as the true parameter value (△). In Case θ_ϕ, we can assert , with a parameter synthesis based off a confidence of $(1-\delta) = 0.95$ and absolute-error $\epsilon = 0.01$, that the underlying stochastic system $\mathcal{S}$ does indeed satisfy the property of interest, as the credibility calculation gives $P(\mathcal{S} \models \phi | S(y_{obs})) = 1$. Case $\theta_{\neg\phi}$ has a low probability of satisfying the property of interest ($P(\mathcal{S} \models \phi | S(y_{obs})) = 0.0054$), whereas for Case $\theta_{\mathcal{U}}$ the inferred mean converges to the true mean that we would expect the estimated probability of satisfying the property to converge to, which is 0.5.

Table 3, and Figure 3 suggest that simulation times are largely dependent on the estimated probabilities, $\hat{\Lambda}_\phi(\theta)$: the closer the estimated probabilities are to 0.5, the larger the number of simulations required, see Table 2. To improve the runtime of Case $\theta_{\mathcal{U}}$, we would need to reduce variance and improve the accuracy of the inferred parameters, for instance by increasing the number of observed data points y_{obs} or with an alternative choice of either the summary statistics chosen or of the perturbation kernels [17].

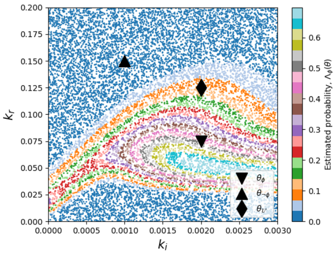
5 Future work

We plan to leverage ongoing research on approximation techniques to speed up simulations for CRNs [7, 19, 22, 47], as our framework is reliant on simultaneously learning and formally verifying by simulating models of interest. The

CRN simulations [49], ABCSMC [23] and the SMC [25] algorithm and thus, the ABC(SMC)^2 scheme, can easily be parallelised. We plan to apply the framework to different model classes, such as stochastic differential equations [18, 20] and incorporating the Bayesian model selection problem [29, 46].

Table 1. Inferred posterior and Bayesian Verification Results.

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



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

Fig. 3 & Table 2. True parameter values with corresponding estimated probabilities using SMC (15000 uniform samples), and number of SMC simulations used in ABC(SMC)^2 .

Table 3. Runtimes for algorithms.

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

References

1. Aziz, A., Sanwal, K., Singhal, V., Brayton, R.: Verifying continuous time Markov chains. In: Alur, R., Henzinger, T.A. (eds.) Computer Aided Verification. pp. 269–276. Springer Berlin Heidelberg, Berlin, Heidelberg (1996)
2. Barnes, C.P., Silk, D., Sheng, X., Stumpf, M.P.: Bayesian design of synthetic biological systems. Proceedings of the National Academy of Sciences 108(37), 15190–15195 (2011)
3. Bentriou, M., Ballarini, P., Cournède, P.H.: Reachability design through approximate bayesian computation. In: International Conference on Computational Methods in Systems Biology. pp. 207–223. Springer (2019)
4. Bonassi, F.V., West, M., et al.: Sequential monte carlo with adaptive weights for approximate bayesian computation. Bayesian Analysis 10(1), 171–187 (2015)
5. Bornn, L., Pillai, N.S., Smith, A., Woodard, D.: The use of a single pseudo-sample in approximate bayesian computation. Statistics and Computing 27(3), 583–590 (2017)
6. Bortolussi, L., Milios, D., Sanguinetti, G.: Smoothed model checking for uncertain continuous-time Markov chains. Inf. Comput. 247(C), 235–253 (Apr 2016)
7. Bortolussi, L., Palmieri, L.: Deep abstractions of chemical reaction networks. In: International Conference on Computational Methods in Systems Biology. pp. 21–38. Springer (2018)
8. Bortolussi, L., Sanguinetti, G.: Learning and designing stochastic processes from logical constraints. In: International Conference on Quantitative Evaluation of Systems. pp. 89–105. Springer (2013)
9. Bortolussi, L., Silvetti, S.: Bayesian statistical parameter synthesis for linear temporal properties of stochastic models. In: Beyer, D., Huisman, M. (eds.) Tools and Algorithms for the Construction and Analysis of Systems. pp. 396–413. Springer International Publishing, Cham (2018)
10. Box, G., Tiao, G.: Bayesian Inference in Statistical Analysis. Wiley Classics Library, Wiley (1973)
11. Broemeling, L.: Bayesian Inference for Stochastic Processes. CRC Press (2017)
12. Ceska, M., Dannenberg, F., Paoletti, N., Kwiatkowska, M., Brim, L.: Precise parameter synthesis for stochastic biochemical systems. Acta Inf. 54(6), 589–623 (2014)
13. Ceska, M., Pilar, P., Paoletti, N., Brim, L., Kwiatkowska, M.Z.: PRISM-PSY: precise gpu-accelerated parameter synthesis for stochastic systems. In: Tools and Algorithms for the Construction and Analysis of Systems - 22nd International Conference, TACAS 2016, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2016, Eindhoven, The Netherlands, April 2-8, 2016, Proceedings. pp. 367–384 (2016)
14. Chernoff, H., et al.: A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. The Annals of Mathematical Statistics 23(4), 493–507 (1952)
15. Cortes, C., Vapnik, V.: Support-vector networks. Machine learning 20(3), 273–297 (1995)
16. Del Moral, P., Doucet, A., Jasra, A.: An adaptive sequential monte carlo method for approximate bayesian computation. Statistics and Computing 22(5), 1009–1020 (2012)
17. Filippi, S., Barnes, C.P., Cornebise, J., Stumpf, M.P.: On optimality of kernels for approximate bayesian computation using sequential monte carlo. Statistical applications in genetics and molecular biology 12(1), 87–107

18. Gardiner, C.: Stochastic Methods: A Handbook for the Natural and Social Sciences. 13, Springer-Verlag Berlin Heidelberg, 4 edn.
19. Gillespie, D.T.: Approximate accelerated stochastic simulation of chemically reacting systems. The Journal of Chemical Physics 115(4), 1716–1733 (2001)
20. Gillespie, D.T., Gillespie, D.T.: The chemical Langevin equation The chemical Langevin equation 297(2000) (2000)
21. Haesaert, S., den Hof, P.M.J.V., Abate, A.: Data-driven and model-based verification: a Bayesian identification approach. CoRR abs/1509.03347 (2015)
22. Higham, D.J.: Modeling and simulating chemical reactions. SIAM Review 50(2), 347–368 (May 2008)
23. Jagiella, N., Rickert, D., Theis, F.J., Hasenauer, J.: Parallelization and high-performance computing enables automated statistical inference of multi-scale models. Cell systems 4(2), 194–206 (2017)
24. Jegourel, C., Sun, J., Dong, J.S.: Sequential schemes for frequentist estimation of properties in statistical model checking. In: International Conference on Quantitative Evaluation of Systems. pp. 333–350. Springer (2017)
25. Jegourel, C., Sun, J., Dong, J.S.: On the sequential massart algorithm for statistical model checking. In: International Symposium on Leveraging Applications of Formal Methods. pp. 287–304. Springer (2018)
26. Jha, S.K., Clarke, E.M., Langmead, C.J., Legay, A., Platzer, A., Zuliani, P.: A Bayesian approach to model checking biological systems. In: Degano, P., Gorrieri, R. (eds.) Computational Methods in Systems Biology. pp. 218–234. Springer Berlin Heidelberg, Berlin, Heidelberg (2009)
27. Karlin, S., Taylor, H., Taylor, H., Taylor, H., Collection, K.M.R.: A First Course in Stochastic Processes. No. v. 1, Elsevier Science (1975)
28. Kermack, W.: A contribution to the mathematical theory of epidemics. Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences 115(772), 700–721 (1927)
29. Kirk, P., Thorne, T., Stumpf, M.P.: Model selection in systems and synthetic biology. Current opinion in biotechnology 24(4), 767–774 (2013)
30. Kwiatkowska, M., Norman, G., Parker, D.: Stochastic model checking. In: Bernardo, M., Hillston, J. (eds.) Formal Methods for the Design of Computer, Communication and Software Systems: Performance Evaluation (SFM'07). LNCS (Tutorial Volume), vol. 4486, pp. 220–270. Springer (2007)
31. Kwiatkowska, M., Thachuk, C.: Probabilistic model checking for biology. In: Software Safety and Security. NATO Science for Peace and Security Series - D: Information and Communication Security, IOS Press (2014)
32. Legay, A., Delahaye, B., Bensalem, S.: Statistical model checking: An overview. In: International conference on runtime verification. pp. 122–135. Springer (2010)
33. Massart, P.: The tight constant in the dvoretzky-kiefer-wolfowitz inequality. The annals of Probability pp. 1269–1283 (1990)
34. Metropolis, N., Ulam, S.: The monte carlo method. Journal of the American statistical association 44(247), 335–341 (1949)
35. Molyneux, G.W., Wijesuriya, V.B., Abate, A.: Bayesian Verification of Chemical Reaction Networks. arXiv preprint arXiv:2004.11321 (2020)
36. Okamoto, M.: Some inequalities relating to the partial sum of binomial probabilities. Annals of the institute of Statistical Mathematics 10(1), 29–35 (1959)
37. Polgreen, E., Wijesuriya, V.B., Haesaert, S., Abate, A.: Data-efficient Bayesian verification of parametric Markov chains. In: Quantitative Evaluation of Systems - 13th International Conference, QEST 2016, Quebec City, QC, Canada, August 23-25, 2016, Proceedings. pp. 35–51 (2016)

38. Polgreen, E., Wijesuriya, V.B., Haesaert, S., Abate, A.: Automated experiment design for data-efficient verification of parametric Markov decision processes. In: Quantitative Evaluation of Systems - 14th International Conference, QEST 2017, Berlin, Germany, September 5-7, 2017, Proceedings. pp. 259–274 (2017)

39. Prangle, D.: Summary statistics in approximate bayesian computation. arXiv preprint arXiv:1512.05633 (2015)

40. Prangle, D., et al.: Adapting the abc distance function. Bayesian Analysis 12(1), 289–309 (2017)

41. Schnoerr, D., Sanguinetti, G., Grima, R.: Approximation and inference methods for stochastic biochemical kinetics: a tutorial review. Journal of Physics A: Mathematical and Theoretical 50(9), 093001 (2017)

42. Sisson, S., Fan, Y., Beaumont, M.: Overview of abc. Handbook of Approximate Bayesian Computation pp. 3–54 (2018)

43. Sisson, S.A., Fan, Y., Beaumont, M.: Handbook of approximate Bayesian computation. Chapman and Hall/CRC (2018)

44. Smola, A.J., Schölkopf, B.: A tutorial on support vector regression. Statistics and computing 14(3), 199–222 (2004)

45. Toni, T., Stumpf, M.P.: Simulation-based model selection for dynamical systems in systems and population biology. Bioinformatics 26(1), 104–110 (2010)

46. Toni, T., Welch, D., Strelkowa, N., Ipsen, A., Stumpf, M.P.: Approximate bayesian computation scheme for parameter inference and model selection in dynamical systems. Journal of the Royal Society Interface 6(31), 187–202 (2008)

47. Warne, D.J., Baker, R.E., Simpson, M.J.: Simulation and inference algorithms for stochastic biochemical reaction networks: from basic concepts to state-of-the-art. Journal of the Royal Society Interface 16(151), 20180943 (2019)

48. Wilkinson, D.: Stochastic Modelling for Systems Biology, Second Edition. Chapman & Hall/CRC Mathematical and Computational Biology, Taylor & Francis (2011)

49. Zhou, Y., Liepe, J., Sheng, X., Stumpf, M.P., Barnes, C.: Gpu accelerated biochemical network simulation. Bioinformatics 27(6), 874–876 (2011)

50. Zuliani, P.: Statistical model checking for biological applications. International Journal on Software Tools for Technology Transfer 17(4), 527–536 (Aug 2015)

A Approximate Bayesian Computation - Sequential Monte Carlo (ABCSMC) Algorithm

Algorithm 2 ABCSMC

Input:

- Prior $\pi(\theta)$ and data-generating likelihood function $p(y_{obs}|\theta)$
- A kernel function $K_h(u)$ and scale parameter $h > 0$ where $u = \|y - y_{obs}\|$
- $N > 0$, number of particles used to estimate posterior distributions
- Sequence of perturbation kernels $F_m(\theta|\theta^*)$, $m = 1, \ldots, M$
- A quantile $\upsilon \in [0, 1]$ to control the rate of decrease of h_m
- Summary statistic function $s = S(y)$
- $B_t > 0$, number of simulations per sampled particle. For stochastic systems $B_t > 1$

Output:

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

```
1: for $m = 0, \ldots, M$: do
2:     for $i = 0, \ldots, N$: do
3:         if $m = 0$ then
4:             Generate $\theta^{**} \sim \pi(\theta)$
5:         else
6:             Generate $\theta^*$ from the previous population $\{\theta_{m-1}^{(i)}\}$ with weights $\{w_{m-1}^{(i)}\}$ and perturb
               the particle to obtain $\theta^{**} \sim F_m(\theta|\theta^*)$
7:         end if
8:         if $\pi(\theta^{**}) = 0$ then
9:             goto line 3
10:        end if
11:        for $b = 1, \ldots, B_t$ : do
12:            Generate $y_b \sim p(y|\theta^{**})$
13:            Calculate $s^b = S(y_b)$
14:        end for
15:        Calculate $b_t(\theta^{**}) = \sum_{b=1}^{B_t} K_{h_m}(\|s^b - s_{obs}\|)$
16:        if $b_t(\theta^{**}) = 0$ then
17:            goto line 3
18:        end if
19:        Set $\theta_m^{(i)} = \theta^{**}$, $\bar{d}_m^{(i)} = \frac{1}{B_t}\sum_{b=1}^{B_t}\|s^b - s_{obs}\|$ and calculate
20:
           $$w_m^{(i)} = \begin{cases} b_t\left(\theta_m^{(i)}\right), & \text{if } t = 0 \\ \dfrac{\pi\left(\theta_m^{(i)}\right) b_t\left(\theta_m^{(i)}\right)}{\sum_{j=1}^N w_{m-1}^{(j)} F_m\left(\theta_m^{(i)}|\theta_{m-1}^{(j)}\right)}, & \text{if } t > 0 \end{cases}$$
21:    end for
22:    Normalise weights: $w_m^{(i)} \leftarrow w_m^{(i)} / \left(\sum_{i=1}^N w_m^{(i)}\right)$
23:    Set $h_{m+1} = (\upsilon/N)\sum_{i=1}^N \bar{d}_m^{(i)}$
24: end for
25: return $\left\{\left(\theta_M^{(i)}, w_M^{(i)}\right)\right\}_{i=1}^N$
```

B Bayesian Verification Framework

There are 3 aspects to the Bayesian Verification framework. The Bayesian inference, parameter synthesis and probability or credibility calculation. The infer-

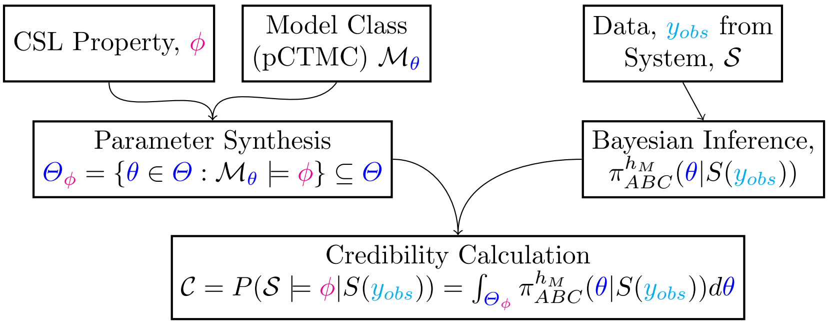

CSL Property, ϕ
Model Class (pCTMC) $\mathcal{M}_\theta$
Data, y_{obs} from System, $\mathcal{S}$
Parameter Synthesis $\Theta_\phi = \{\theta \in \Theta : \mathcal{M}_\theta \models \phi\} \subseteq \Theta$
Bayesian Inference, $\pi^{h_M}_{ABC}(\theta | S(y_{obs}))$
Credibility Calculation $\mathcal{C} = P(\mathcal{S} \models \phi | S(y_{obs})) = \int_{\Theta_\phi} \pi^{h_M}_{ABC}(\theta | S(y_{obs})) d\theta$

Fig. 4. Bayesian Verification Framework of [35].

ence technique we use has been covered in the main text and here we focus on the parameter synthesis and the probability calculation.

B.1 Credibility Calculation

In the final phase of the approach, a probability estimate is computed corresponding to the satisfaction of a CSL specification formula ϕ by a system of interest such that $\mathcal{S} \models \phi$, which we denote as the credibility. To calculate the credibility that the system satisfies the specified property, we integrate the posterior distribution $\pi(\theta | y_{obs})$ over the feasible set of parameters Θ_ϕ:

Definition 7. *Given a CSL specification ϕ and observed data y_{obs} and $s_{obs} = S(y_{obs})$ from the system $\mathcal{S}$, the probability that $\mathcal{S} \models \phi$ is given by*

$$\mathcal{C} = P(\mathcal{S} \models \phi | s_{obs}) = \int_{\Theta_\phi} \pi(\theta | s_{obs}) d\theta, \tag{11}$$

where Θ_ϕ denotes the feasible set of parameters.

C Absolute-Error Massart Algorithm

Here we present the slightly modified Sequential Massart Algorithm with Absolute Error. The outputs of Algorithm 3 are $\hat{\Lambda}_\phi(\theta)$, the total number of simulation undertaken B_t, the sum of the kernel smoothing functions $\sum_{b=1}^{B_t} K_{h_m}(\|s^b - s_{obs}\|)$ and the mean summary statistic produced from n simulations, $\bar{d}$. The algorithm is slightly modified to consider the distance function that is crucial for the ABC-SMC aspect of the algorithm.

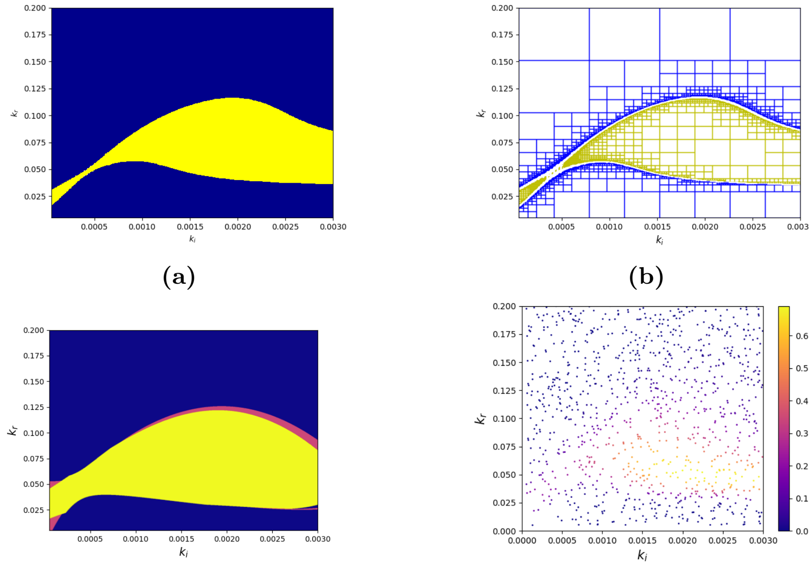
D Parameter Synthesis: A Motivating Comparison

The PRISM-based parameter synthesis technique dissects the parameter space into 14413 grid regions (cf. Figure 5b), which results in calculating the satisfaction probability at 57652 points.

Algorithm 3 Modified Absolute-Error Sequential Massart Algorithm

```
Input:
– Absolute-error value $\epsilon$, a confidence parameter $\delta$ and coverage parameter $\alpha$.
– Current distance threshold $h_m$.
– Sampled parameter $\theta^{**}$.
– True data $s_{obs}$
– CSL specification $\phi$
Output:
– Estimated probability $\hat{\Lambda}_\phi(\theta^{**})$ with corresponding bounds $[\Lambda^L_\phi(\theta^{**}), \Lambda^U_\phi(\theta^{**})]$.
– Sum of kernel smoothing functions $\sum_{b=1}^{B_t} K_{h_m}(\|s^b - s_{obs}\|)$.
– Mean summary statistic from $B_t$ simulations $\bar{d}$.

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



(a) **(b)** **(c)** **(d)**

Fig. 5. The set Θ_ϕ, is shown in yellow (lighter colour), meanwhile $\Theta_{\neg\phi}$, is shown in blue (darker colour) $\Theta_{\neg\phi}$. The undecided areas, $\Theta_{\mathcal{U}}$ (if any) are shown in magenta.
(5a) Parameter regions synthesised by GPU-Accelerated PRISM [13]. (5b) Gridding scheme.
(5c) Parameter regions from SVM classification using 1000 samples from a uniform distribution. (5d) Estimated probabilities $\Lambda_\phi(\theta^*)$.

Instead, we consider sampling 1000 points from a Uniform distribution over the parameter space. We run the Massart algorithm at each point to obtain an estimated probability with corresponding $(1-\delta)$ confidence bounds, where $\delta = 0.05$. With these samples and probabilities, we classify parameter regions with an SVM, which results in Figure 5c,with corresponding estimated probabilities in Figure 5d. The runtimes presented in Table D suggest that we obtain a good approximation of the parameter synthesis region in half the time of the GPU-accelerated PRISM tool, which could be further improved if we parallelised the computation [25]. These considerations have led us to embed the statistical parameter synthesis in the parameter inference algorithm.

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