Smoothed model checking for uncertain Continuous-Time Markov Chains ☆

Luca Bortolussi [a,b,c,*], Dimitrios Milios [d], Guido Sanguinetti [d,e]

[a] Department of Mathematics and Geosciences, University of Trieste, Italy
[b] Modelling and Simulation Group, University of Saarland, Germany
[c] CNR/ISTI, Pisa, Italy
[d] School of Informatics, University of Edinburgh, United Kingdom
[e] SynthSys, Centre for Synthetic and Systems Biology, University of Edinburgh, United Kingdom

Keywords:
Model checking
Uncertainty
Continuous-Time Markov Chains
Gaussian Processes

A B S T R A C T

We consider the problem of computing the satisfaction probability of a formula for stochastic models with parametric uncertainty. We show that this satisfaction probability is a smooth function of the model parameters under mild conditions. This enables us to devise a novel Bayesian statistical algorithm which performs model checking simultaneously for all values of the model parameters from observations of truth values of the formula over individual runs of the model at isolated parameter values. This is achieved by exploiting the smoothness of the satisfaction function: by modelling explicitly correlations through a prior distribution over a space of smooth functions (a Gaussian Process), we can condition on observations at individual parameter values to construct an analytical approximation of the function itself. Extensive experiments on non-trivial case studies show that the approach is accurate and considerably faster than naive parameter exploration with standard statistical model checking methods.

1. Introduction

Computational verification of logical properties by model checking is one of the great success stories of theoretical computer science, with profound practical implications. Robust and mature tools such as PRISM [1] and MRMC [2] implement probabilistic model checking, enabling the quantification of the truth probability of a formula for a wide variety of stochastic models and logics. Recently, the applicability of model checking has been greatly extended by the development of randomized algorithms which leverage the possibility of drawing large number of samples from generative models such as Continuous-Time Markov Chains (CTMCs), by means of stochastic simulation algorithms such as SSA [3]. Statistical model checking (SMC, [4,5]) repeatedly draws independent samples (runs/ trajectories) from the model to estimate satisfaction probabilities as averages of satisfactions on individual runs; by the law of large numbers, these averages will converge to the true probability in the limit when the sample size becomes large, and general asymptotic results permit to bound (probabilistically) the estimation error.

☆ We thank Botond Cseke for sharing with us his code for Expectation Propagation, Alberto Policriti for highlighting the analogy with Smoothed Complexity Analysis, and Jane Hillston and Vincent Danos for useful conversations. D.M. and G.S. acknowledge support from the ERC under grant MLCS 306999. L.B. acknowledges partial support from EU-FET project QUANTICOL (nr. 600708), by FRA-UniTS, and by the German Research Council (DFG) as part of the Cluster of Excellence on Multimodal Computing and Interaction at Saarland University.

* Corresponding author.
E-mail address: luca@dmi.units.it (L. Bortolussi).

Both analytical and statistical tools for model checking however start from the premises that the underlying mathematical model is fully specified (or at least that a mechanism to draw independent and identically distributed samples exists in the case of SMC). This is both conceptually and practically problematic: models are abstractions of reality informed by domain expertise. Condensing the domain expertise in a single vector of parameter values is at best an approximation. We are particularly interested in modelling biological systems: in this important application area for formal methods, parametric uncertainty is the norm, and reasoning about how the qualitative properties of a system may vary while varying parameters is often of paramount interest. While parameter estimation from measurements has seen considerable progress in recent years [6–10], uncertainty can never be completely eliminated. In such cases, acceptance of the inherent uncertainty is the natural way forward, and alternative semantics such as Interval Markov Chains [11] or Constraint Markov Chains [12] should be preferred. Model checking methodologies for these semantics are however in their infancy and mostly rely on a reduction to continuous-time Markov Decision Processes [13], obtaining upper and lower bounds on the satisfaction probability [11,14]. Other approaches such as [15,16] provide provably correct piecewise constant upper and lower bounds on the satisfaction probability, but rely on a computationally intensive use of numerical transient analysis.

In this paper we define a novel, quantitative approach to model checking uncertain CTMCs. We start by defining the *satisfaction function* of a formula, the natural extension of the concept of satisfaction probability of a formula to the case of CTMCs with parametric uncertainty. We prove that, under mild conditions, the satisfaction function is a smooth function of the uncertain parameters of the CTMC. We then propose a novel statistical model checking approach which leverages this smoothness to transfer information on the satisfaction of the formula at nearby values of the uncertain parameters. We show that the satisfaction function can be approximated arbitrarily well by a sample from a *Gaussian Process* (GP), a non-parametric distribution over spaces of functions, and use the GP approach to obtain an analytical approximation to the satisfaction function. This enables us to predict the value (and related uncertainty) of the satisfaction probability *at all values of the uncertain parameters* from individual model simulations at a finite (and generally rather small) number of distinct parameter values. We term this whole approach *smoothed model checking* in analogy with the recently proposed smoothed complexity analysis, another traditional domain of discrete mathematics where embedding problems in a continuous framework has proved highly valuable [17]. We show on three non-trivial examples that smoothed model checking can provide an accurate estimation of the satisfaction function with considerably fewer simulations than naive exploration by SMC, providing in our examples computational savings of an order of magnitude for equivalent accuracy.

The rest of the paper is organised as follows: in the next section we introduce the mathematical background on formulae and model checking, proving in the following section that the satisfaction function is a smooth function of model parameters. We then introduce our smoothed model checking framework and discuss the statistical tools needed. We demonstrate the power of our approach on three nontrivial examples, and conclude the paper by discussing the broader implications of our approach.

2. Background

In this section, we briefly review continuous-time Markov chains, Metric Interval Temporal Logic, and statistical model checking approaches.

2.1. Continuous-Time Markov Chains

A Continuous-Time Markov Chain (CTMC) $\mathcal{M}$ is a Markovian (i.e. memoryless) stochastic process defined on a finite or countable state space S and evolving in continuous time [18]. A CTMC on S is fully characterised by its infinitesimal generator matrix $Q \in \mathbb{R}^{|S|\times|S|}$ [19]. For any $i, j \in S$, the entry Q_{ij} denotes the rate parameter of an exponential distribution that governs the time of transitioning from i to j.

Instead of representing the generator Q explicitly, we will specifically consider population models of interacting agents [20], which can be easily represented by:

- a vector of population variables $\vec{X} = (X_1, \ldots, X_n)$, counting the number of entities of each kind, and taking values in $S \subseteq \mathbb{N}^n$;
- a finite set of reaction rules, describing how the system state can change. Each rule η is a tuple $\eta = (\vec{r}_\eta, \vec{s}_\eta, \tau_\eta)$. $\vec{r}_\eta$ (respectively $\vec{s}_\eta$) is a vector encoding how many agents are consumed (respectively produced) in the reaction, so that $\vec{v}_\eta = \vec{s}_\eta - \vec{r}_\eta$ gives the net change of agents due to the reaction. $\tau_\eta = \tau_\eta(\vec{X}, \theta)$ is the rate function, associating to each reaction the rate of an exponential distribution, depending on the global state of the model and on a d dimensional vector of *model parameters*, θ. Reaction rules are easily visualised in the chemical reaction style, as follows:

  $$r_1 X_1 + \ldots r_n X_n \xrightarrow{\tau(\vec{X},\theta)} s_1 X_1 + \ldots s_n X_n$$

  For the population models we consider, the entries of the generator matrix can be reconstructed as follows:

  $$Q_{ij} = \sum_\eta \tau_\eta(\vec{X}, \theta)$$

  where $i = \vec{X}$ and $j = \vec{X} + \vec{v}_\eta$.

The dependency on θ is often crucial, with qualitatively different dynamics arising at different values of θ (for an example, see the infection models described in Section 5.1). We use the notation $\mathcal{M}_\theta$ to stress such a dependency.

Sample trajectories of a CTMC are (usually integer-valued) piecewise constant cadlag functions, with jumps distributed exponentially in time (see e.g. [21] for further mathematical details). Hence, we can think of CTMCs as a collection of random variables $\vec{X}(t)$ on the state space S, indexed by time $t \in [0, T]$, or as random functions $\vec{X}_{0:T}$ from $[0, T]$ to S (which are necessarily piecewise constant due to the countable nature of S). In this second sense, a CTMC is equivalent to a measure on the (infinite dimensional) space of trajectories of the system, which are individually denoted by $\vec{x}_{0:T}$.

Running example – Poisson process: Part I We demonstrate the CTMC notation introduced on a Poisson process. The state of the system is represented by the population vector $\vec{X} = (X_N)$, which contains a single variable that counts the number of arrival events N. An arrival is described by the following reaction:

Arrival: $\emptyset \xrightarrow{r} N$

where r denotes the constant arrival rate, whose value is uncertain.

2.2. Metric Interval Temporal Logic

We will specify properties of CTMC trajectories by Metric Interval Temporal Logic (MITL), see [22]. MITL is a linear temporal logic for continuous time trajectories, and we will consider its time-bounded fragment. The choice of MITL is justified because time-bounded linear time properties are natural when reasoning about systems like biological ones, for which we can only observe single time-bounded realisations. However, our method can be straightforwardly applied to any other *time-bounded linear time* specification formalism equipped with a monitoring routine.

Formally, the syntax of MITL is given by the following grammar:

$$\varphi ::= \mathtt{tt} \mid \mu \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \mathbf{U}_{[T_1,T_2]} \varphi_2,$$

where $\mathtt{tt}$ is the true formula, conjunction and negation are the standard boolean connectives, and there is only one temporal modality, the time-bounded until $\mathbf{U}_{[T_1,T_2]}$. Atomic propositions μ identify subsets of the state space. In practice, we consider them to be (non-linear) inequalities on population variables.

A MITL formula is interpreted over a real valued function of time $\vec{x}$, and its satisfaction relation is given by the classical point-wise semantics, see e.g. [22]. For instance, the rule for the temporal modality states that $\vec{x}, t \models \varphi_1 \mathbf{U}_{[T_1,T_2]} \varphi_2$ if and only if $\exists t_1 \in [t+T_1, t+T_2]$ such that $\vec{x}, t_1 \models \varphi_2$ and $\forall t_0 \in [t, t_1)$, $\vec{x}, t_0 \models \varphi_1$. Temporal modalities like time-bounded eventually and always can be defined in the usual way from the until operator: $\mathbf{F}_{[T_1,T_2]}\varphi \equiv \mathtt{tt}\mathbf{U}_{[T_1,T_2]}\varphi$ and $\mathbf{G}_{[T_1,T_2]}\varphi \equiv \neg\mathbf{F}_{[T_1,T_2]}\neg\varphi$. MITL can be interpreted in the probabilistic setting [5,23,9] by computing the path probability $Pr(\varphi = \mathtt{tt}|\mathcal{M}_\theta)$ of a formula φ, $Pr(\varphi = \mathtt{tt}|\mathcal{M}_\theta) = Pr(\{\vec{x}_{0:T} \mid \vec{x}_{0:T}, 0 \models \varphi\}|\mathcal{M}_\theta)$, i.e. the probability of the set of (time-bounded) CTMC trajectories that satisfy the formula.[1]

2.3. Model checking

Model checking algorithms for MITL formulae against a CTMC model are essentially of two kinds. Numerical algorithms [23] are very complex, and severely suffer from state space explosion. A more feasible alternative for population models is Statistical Model Checking (SMC [5,4]), which is implemented in widely used model checking tools such as PRISM [1] or MRMC [2]. SMC approaches estimate the probability of a MITL formula by combining simulation and statistical inference tools. More precisely, given a CTMC $\mathcal{M}_\theta$ with fixed parameters θ, time-bounded CTMC trajectories are sampled by standard simulation algorithms, like SSA [3], and monitoring algorithms for MITL [22] are used to asses if the formula φ is satisfied for each sampled trajectory. In this way, one generates samples from a Bernoulli random variable Z_φ, equal to 1 if and only if φ is true. SMC then uses standard statistical tools, either frequentist [4] or Bayesian [5], to estimate the satisfaction probability $Pr(\varphi|\mathcal{M}_\theta)$ or to test if $P(\varphi|\mathcal{M}_\theta) > q$ with a given confidence level α.

3. Uncertain CTMCs and the satisfaction function

In this section, we define the main object of our study, the satisfaction function of a formula for uncertain CTMCs, and characterise its differentiability. We call a family of CTMC models indexed by a parameter vector an *uncertain CTMC*. Our interest is to quantify how the satisfaction of MITL formulae against CTMCs drawn from an uncertain CTMC depends on the unknown parameters. The following defines the central object of study in our work.

[1] We assume implicitly that T is sufficiently large so that the truth of φ at time 0 can always be established from $\vec{x}$. The minimum of such times can be easily deduced from the formula φ itself, see [5,22].

Definition 1. Let $\mathcal{M}_\theta$ be an uncertain CTMC with parameters $\theta \in D$ where D is a compact subset of $\mathbb{R}^d$, and let φ be a MITL formula. The *satisfaction function* $f_\varphi : D \to [0,1]$ associated with φ is

$$f_\varphi(\theta) = Pr(\varphi = \mathtt{tt} | \mathcal{M}_\theta)$$

i.e., with each value θ in the space of parameters D it associates the satisfaction probability of φ for the model with that parameter value.

Before proceeding to show that $f_\varphi(\theta)$ is a smooth function of the model parameters, we observe that classic SMC can be applied only to models with a fixed value of parameters. Estimating the whole satisfaction function f_φ by SMC would require a potentially large number of evaluations; while these can be performed in parallel, the overall number of simulations needed for accurate estimation would certainly be very large.

3.1. Smoothness of the satisfaction function

The following lemma is standard but useful (see for instance [5]). Recall that a CTMC $\mathcal{M}_\theta$ induces a measure μ_θ over the space of trajectories of the system.

Lemma 1. *Let $\mathcal{M}_\theta$ be a CTMC and φ be a MITL formula. The subset of trajectories where the formula is satisfied is a measurable set under μ_θ.*

We are now ready to prove our main theoretical results. An important characterisation of the satisfaction function is given in the following theorem:

Theorem 1. *Let φ be a MITL formula and let $\mathcal{M}_\theta$ be an uncertain CTMC indexed by the variable $\theta \in D$. Denote as $\boldsymbol{\tau}(\vec{X}, \theta)$ the transition rates of the CTMCs and assume that these depend smoothly on the parameters θ and polynomially on the state vector of the system $\vec{X}$. Then, the satisfaction function of φ is a smooth function of the parameters, $f_\varphi \in \mathcal{C}^\infty(D)$.*

Proof. The proof is reported in Appendix A. □

Running example – Poisson process: Part II In the Poisson process example, we consider the following simple formula:

$$\varphi = \mathbf{G}_{[0,1]}(X_N < 4) \tag{1}$$

which will be true for trajectories where the number of arrivals X_N is always less than four in the time interval between 0 and 1. The satisfaction probability of the formula in (1) can be statistically evaluated for different values of the arrival rate r. Fig. 1 shows the results of performing statistical model checking, where r was allowed to vary over an order of magnitude from 0.5 to 5. In the left-side figure, we have used *five* samples to evaluate the satisfaction probability, while *twenty* samples were used in the right-side figure. For this particular property of the Poisson process, an analytic expression can be derived for the satisfaction function; that will be the cumulative probabilities of having up to 3 arrivals:

$$f(r) = \sum_{i=0}^{3} \frac{r^i}{i!} e^{-r} \tag{2}$$

As expected, we see that increasing the number of samples results in better approximation of the satisfaction function.

4. Smoothed model checking

In this section, we introduce in detail the smoothed model checking approach and the statistical tools we use. We will start first from a high-level description of the method in the next section, filling in the statistical details in the following ones.

4.1. A high-level view

Given an uncertain CTMC $\mathcal{M}_\theta$ depending on a vector of parameters $\theta \in \mathcal{D}$ and a MITL formula φ, our goal is to find a *statistical estimate* of the satisfaction probability of φ as a function of θ, i.e. of the satisfaction function $f_\varphi(\theta) = P(\varphi | \mathcal{M}_\theta)$, $\theta \in \mathcal{D}$. As in all statistical model checking algorithms, our statistical estimation will be based on monitoring the satisfaction of the formula on sample trajectories of the system. However, as our system depends on a continuous vector of parameters, we will necessarily be able to *noisily* observe the function f_φ only at a few input points $\theta_1, \ldots, \theta_k$, our *training set*. Given such information, our task is to construct a statistical model that, for *any* value $\theta^* \in \mathcal{D}$, will permit us to *compute efficiently* an *estimate of* $f_\varphi(\theta^*)$ and a *confidence interval* for such a prediction. Notice that standard SMC tools will only be able to

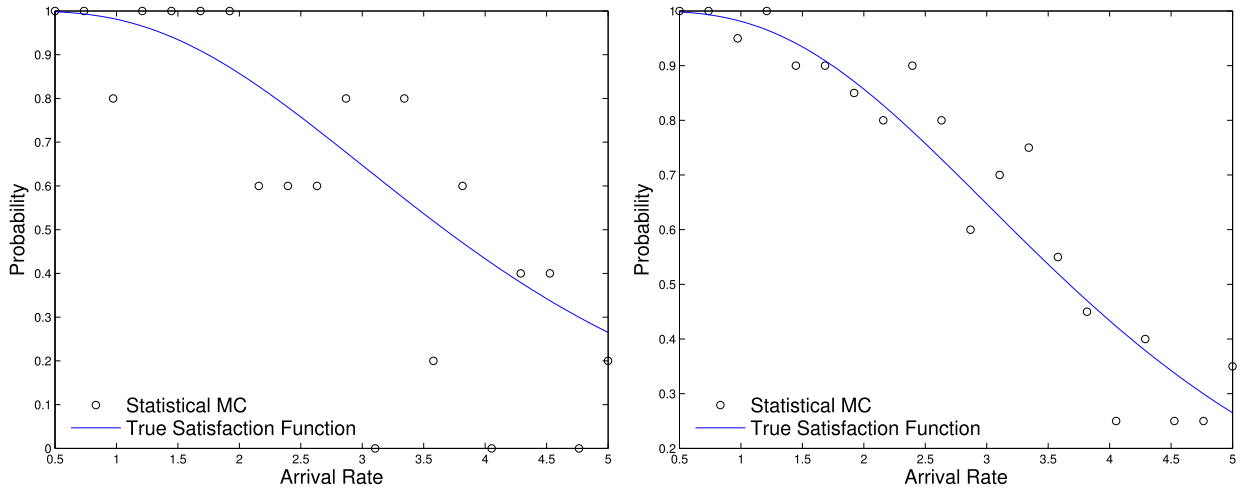

Fig. 1. Statistical estimation of the satisfaction function for Poisson process from five samples per input value (left) and twenty samples per value (right).

provide point-wise estimates of the satisfaction function; to obtain a global approximation, a possibly prohibitive number of SMC evaluations would be needed (for example on a fine grid in parameter space).

To find an efficient and mathematically sound solution to this problem, we will adopt a Bayesian approach; the main ingredients are as follows:

- we first choose a *prior distribution* over a suitable *function space*; this distribution must be sufficiently expressive as to contain the satisfaction function in its support;
- we determine the functional form of the *likelihood*, i.e. how the probability of the observed satisfaction values at individual parameters depends on the (unknown) true value of the satisfaction probability at that point;
- leveraging Bayes's theorem, we then compute an approximation to posterior distribution over functions, given the observations. Evaluating the statistics of the induced posterior distribution on the function values at point θ^* we obtain the desired estimate and confidence interval.

The technical difficulties of the procedure centre around the need to construct and manipulate efficiently distributions over functions. Here we rely on Gaussian Processes (GP), a flexible and computationally tractable non-parametric class of distributions which provide an ideal prior distribution for our problem [24]. The choice of the likelihood model is instead dictated by the nature of the problem: our observations are obtained by evaluating a property φ on a single trajectory generated from a stochastic model $\mathcal{M}_\theta$. Hence our actual observations are boolean, and the probability of the observations being 1 is a Bernoulli distribution with parameter $f_\varphi(\theta)$. To improve estimation accuracy, we will generally draw $m \geq 1$ observation for each θ in the training set, so that observations are actually drawn from a Binomial random variable $Binomial(m, f_\varphi(\theta))$. Notice that, in statistical model checking, an approximation of $f_\varphi(\theta)$ would be computed directly from observations of such a binomial variable; the accuracy of the estimate would only be guaranteed in the limit of $m \to \infty$. In a GP context, there is no need to perform this intermediate estimation: we can *directly use the binomial observation model* in Bayes's theorem. In this way, we are using the *exact* statistical model of the process, converging to the true function in the limit of a large number of observations. The strength of the approach, however, is in the fact that we generally obtain good approximations with few samples m per each of few input points $\theta_1, \ldots, \theta_k$, which makes the method very efficient from a computational point of view.

The statistically minded reader may have noticed that, as we are observing a set of true/ false labels per each input point, this problem looks similar to a classification problem (with the crucial difference that in a classification problem we observe only one single label per point and we do not have an exact statistical model of the observations). This similarity enables us to leverage the vast algorithmic repertoire developed for GP classification; in particular, our solution to the inference of satisfaction functions uses a modified version of the Expectation Propagation algorithm, which is the state-of-the-art in GP classification [24].

In the remainder of this section we will introduce in turn each element of the algorithm, and conclude by providing practical advice on its use and implementation.

4.2. Bayesian inference

In this paper, we adopt a Bayesian machine learning approach. Bayesian methods offer substantial advantages, including a principled treatment of noise and a mathematically consistent way to quantify the resulting uncertainty in model-based

estimates. Bayesian methods have already been employed in statistical model checking [5]: there, the aim was to incorporate prior beliefs about truth probabilities, as well as to regularize to better handle rare events.

The fundamental insight of Bayesian statistics is that uncertainty quantification is a two step process: given an observable, uncertain quantity θ, we must first construct a probability distribution $p(\theta)$, the *prior distribution*, which encapsulates our beliefs about θ prior to any observations been taken. The choice of the prior distribution is a fundamental modelling step, and affords considerable flexibility: prior distributions can range from uninformative priors, to priors encapsulating basic properties of the variables (e.g. positivity, continuity) to detailed mechanistic priors informed by hard scientific knowledge. The second component of a Bayesian model consists of the *likelihood function* or *noise model*, $p(\hat{\theta}|\theta)$, i.e. a probabilistic model of how the actual observations $\hat{\theta}$ depend on the value of the uncertain quantity. The likelihood effectively models the noise inherent in the observation process: while in some occasions these may be reasonably modelled from knowledge of the observation process, in other cases simple parametric choices such as Gaussian are made for computational simplicity.

Once both prior distribution and likelihood are defined, the rules of probability provide a way of computing the *posterior probability* through the celebrated *Bayes' Theorem*

$$p(\theta|\hat{\theta}) = \frac{p(\hat{\theta}|\theta)p(\theta)}{\sum_{\theta} p(\hat{\theta}|\theta)p(\theta)}. \tag{3}$$

The posterior probability precisely quantifies the uncertainty in our quantity of interest θ resulting from both our prior beliefs and our noisy observations of its value. Unfortunately, computing the normalisation constant in equation (3) is often computationally impossible in all but the simplest cases, and much research in Machine Learning and Computational Statistics is devoted to efficient algorithms to accurately approximate posterior distributions.

Over the rest of this section, we shall discuss how the Bayesian framework can be used effectively to devise a statistical model checking algorithm for uncertain CTMCs. We separately introduce the family of prior models we employ, how this can be combined with observations of the satisfaction of formulae over individual runs of the model, and finally how an accurate and efficient approximation of the posterior distribution can be computed.

4.3. Prior modelling – Gaussian processes

In this paper we are interested in estimating the satisfaction function of a formula from instances of its satisfaction on individual runs at discrete parameter values. Our theoretical analysis in Theorem 1 enabled us to conclude that the satisfaction function is a smooth function of its arguments, the model parameters: a natural choice of prior distribution over smooth functions is a *Gaussian Process* (GP [24]). Intuitively, one can realise a random function as a linear combination of basis functions with random coefficients:

$$f(x) = \sum_{i \in \mathcal{I}} w_i \xi_i(x),$$

where $\mathcal{I} \subset \mathbb{N}$ is a set of indices and $\xi_i(x)$ are the basis functions. If we choose the coefficients (weights) w_i to be sampled from a normal distribution, the resulting random functions are draws from a GP. Formally, the definition of a GP is as follows:

Definition 2. A GP is a collection of random variables indexed by an input variable x such that every finite dimensional marginal distribution is a multivariate normal distribution.

In practice, a sample from a GP is a random function; the random vector obtained by evaluating a sample function at a finite set of points $x_1, \ldots, x_N$ is a multivariate Gaussian random variable. A GP is uniquely defined by its *mean* and *covariance* functions, denoted by $\mu(x)$ and $k(x, x')$; the mean vector (covariance matrix) of the finite dimensional marginals are given by evaluating the mean (covariance) function on every (pair of) point in the finite sample. Naturally, by subtracting the mean function to any sample function, we can always reduce ourselves to the case of *zero mean* GPs; in the following, we will adopt this convention and ignore the mean function.

How is the GP covariance related to the basis function description? Consider a random function $f(x) = \sum_{i \in \mathcal{I}} w_i \xi_i(x)$. By definition,

$$k(x, x') = \langle f(x) f(x') \rangle = \sum_{i,j} \xi_i(x) \xi_j(x') \langle w_i w_j \rangle \tag{4}$$

so that the covariance function is determined by the covariance of the weights and by products of basis functions evaluated at the two points. In practice, it is more convenient to specify a GP by directly specifying its covariance function (the so-called function space view), rather than using the explicit basis function construction; nevertheless, basis functions are still useful in order to prove properties of GPs.

A popular choice for the covariance function, which we will also use, is the *squared exponential* covariance function

$$k(x, x') = \sigma^2 \exp\left[-\frac{(x - x')^2}{\lambda^2} \right]$$

with two hyper-parameters: the amplitude σ^2 and the characteristic length scale λ^2. It is easy to show that this covariance function corresponds to selecting as basis functions a set of Gaussian shaped curves of standard deviation λ and centred at all points in the input space [24].

The kernel or covariance function endows the space of samples from a GP with a metric: this is an example of a Reproducing Kernel Hilbert Space (RKHS). A complete characterisation of such spaces is non-trivial; for our purposes, however, it is sufficient to show that their expressivity is sufficient to approximate a satisfaction function by a sample from a GP. We restrict to the squared exponential covariance, although our result holds for the more general class of universal covariance functions (for a precise definition of universality see [25]). We then have the following result:

Theorem 2. *Let f be a continuous function over a compact domain $D \in \mathbb{R}^p$. For every $\epsilon > 0$, there exists a sample ψ from a GP with squared exponential covariance such that*

$$\|f - \psi\|_2 \leq \epsilon$$

where $\|\cdot\|_2$ denotes the L_2 norm.

Proof. This follows directly from the universal property of the square exponential kernel, as discussed in [25].[2] □

4.4. GP posterior prediction

The results of Theorems 1 and 2 jointly imply that the satisfaction function of a formula can be approximated arbitrarily well by a sample from a GP, justifying the use of GPs as priors for the satisfaction function. To see how this fact enables a Bayesian statistical model checking approach *directly at the level of the satisfaction function*, we need to explain the basics of posterior computation in GP models. Let x denote the input value and let $\hat{\mathbf{f}} = \{\hat{f}_1, \ldots, \hat{f}_N\}$ denote observations of the values of the unknown function f at input points $x_1, \ldots, x_N$. We are interested in computing the distribution over f at a *new* input point x^* *given* the observed values $\hat{\mathbf{f}}$, $p(f(x^*)|\hat{\mathbf{f}})$. A priori, we know that the true function values at any finite collection of input points is Gaussian distributed, hence

$$p\big(f(x^*), f(x_1), \ldots, f(x_N)\big) = \mathcal{N}(\boldsymbol{\mu}, \Sigma)$$

with $\boldsymbol{\mu}$ and Σ obtained from the mean and covariance function as explained in the previous subsection. This prior distribution can be combined with likelihood models for the observations, $p(\hat{f}|f)$ in a Bayesian fashion to yield a joint posterior

$$p\big(f(x^*), f(x_1), \ldots, f(x_N)|\hat{\mathbf{f}}\big) = \frac{1}{Z} p\big(f(x^*), f(x_1), \ldots, f(x_N)\big) \prod_i p\big(\hat{f}_i|f(x_i)\big)$$

where Z is a normalisation constant. The desired posterior predictive distribution can then be obtained by integrating out (*marginalising*) the true function values $f(x_1), \ldots, f(x_N)$

$$p\big(f(x^*)|\hat{\mathbf{f}}\big) = \int \prod_{i=1}^{N} df(x_i)\, p\big(f(x^*), f(x_1), \ldots, f(x_N)|\hat{\mathbf{f}}\big). \tag{5}$$

Equation (5) plays a central role in non-parametric function estimation; the inference procedure outlined above goes under the name of *GP regression*. It is important to note that, in the case of Gaussian observation noise, the integral in equation (5) can be computed in closed form. Further details are given e.g. in [24].

Important remark GP regression provides an analytical expression for the predicted mean and variance of the unknown function at all input points

4.5. Likelihood model

In our case, observations of the satisfaction function are made through boolean evaluations of a formula over individual trajectories at isolated parameter values. Therefore, the Gaussian observation noise cannot be applied directly in this case, meaning that a closed form solution to the inference problem cannot be found.[3] However, as we shall see, a closed form,

[2] The proof of the universality of the square exponential kernel is nontrivial and technical. A simpler, constructive proof of this theorem can be obtained by observing that any smooth function can be approximated arbitrarily well with a polynomial, and then use the fact that the integral operator with squared-exponential kernel has bounded inverse on polynomials. Therefore, the weight vector must be bounded, and hence have finite probability under a GP.

[3] In principle, one could appeal to the Central Limit Theorem and approximate a Gaussian observation model by using as observations averages of formula evaluations over numerous trajectories for each values of the parameters. Such a procedure however would incur significant computational overheads.

accurate approximation of the posterior can be found. The satisfaction of a formula φ over a trajectory generated from a specific parameter value θ is a Bernoulli random variable with success probability $f_\varphi(\theta)$. In order to map this probability to the real numbers, we introduce the *inverse probit* transformation

$$\psi(f) = g \Leftrightarrow f = \int_{\infty}^{g} \mathcal{N}(0,1) \quad \forall f \in [0,1], g \in \mathbb{R}$$

where $\mathcal{N}(0,1)$ is the standard Gaussian distribution with mean zero and variance 1. The function $g_\varphi(\theta) = \psi(f_\varphi(\theta))$ is by construction a smooth, real valued function of the model parameters, and can therefore be modelled as a draw from a GP.

We can summarise the inference problem as follows: our data would consist of D binary evaluations of satisfaction at each of P parameter values. At each parameter value, binary evaluation represent independent draws from the same Bernoulli distribution with success probability $f_\varphi(\theta)$. The (inverse probit) transform of the success probability is a smooth function of the parameters and is assigned a GP prior. The overall joint probability of the observations $\mathcal{O}$ and of the satisfaction function $f_\varphi(\theta)$ would be given by

$$p\big(\mathcal{O}, f_\varphi(\theta)\big) = GP\big(\psi\big(f_\varphi(\theta)\big)\big) \prod_{i=1}^{D} \prod_{j=1}^{P} \text{Bernoulli}\big(O_{i,j} | f_\varphi(\theta_j)\big) \tag{6}$$

Computing the posterior distribution the satisfaction probability at a new parameter value $f_\varphi(\theta^*)$ solves the statistical model checking problem.

Before discussing how the posterior computation problem can be solved, we make the following observations:

1. The procedure described above is an *exact* probabilistic model, in the sense that both the Bernoulli and the GP modelling step use provable properties of the unknown function (by virtue of Theorem 1). No further approximations are introduced.
2. Computing the posterior distribution introduces an approximation, however it provides an *analytical estimate* (with uncertainty) of the satisfaction probability at all parameter values (not only the ones explored through simulation)
3. The posterior inference problem is akin to a *classification* problem, where we seek to assign the probability that a binary output will be 1 to a point in input space. The difference from a standard classification problem is that in this case we admit the possibility of having multiple labels observed for the same input point. This observation enables us to effectively exploit the vast amount of research on GP classification over the last decade to devise an efficient and accurate algorithms for statistical model checking.

4.6. Approximate posterior computation

GP classification has been intensely studied over the last fifteen years in machine learning; a key difference from the regression case is that exact computation of the posterior probability is not possible. However, several highly accurate approximate schemes have been proposed over the last decade in the machine learning literature: here we use the Expectation Propagation (EP) approach [26,27], which has been shown to combine high accuracy with strong computational efficiency. Importantly, this approach still provides an analytic approximation to the whole satisfaction function in terms of basis functions.

We highlight here the basic ideas behind the EP algorithm; for a more detailed discussion see e.g. [24]. The EP algorithm has its origins in the statistical physics of disordered systems [26]; it is a general algorithm that computes a Gaussian approximation to probabilistic models of the form

$$p(\mathbf{x}|\mathbf{y}) = p_0(\mathbf{x}) \prod_i t_i(y_i, x_i) \tag{7}$$

where $p_0(\mathbf{x})$ is a multivariate Gaussian distribution coupling all x_i variables (so called *site variables*) and t_i can be general univariate distributions. Models of this form are frequently encountered in machine learning and are termed *latent Gaussian models*: the common setup (which is indeed our setting in equation (6)) is that the p_0 term represents a prior distribution, with the t_i terms representing non-Gaussian observation likelihoods. The EP approximation inherits the same structure of the original model in equation (7), with the likelihood terms replaced by univariate Gaussian terms

$$q(\mathbf{x}|\mathbf{y}) = p_0(\mathbf{x}) \prod_i \tilde{t}_i(y_i, x_i). \tag{8}$$

The algorithm then proceeds iteratively as follows:

1. Given an estimate of the joint EP approximation in equation (8), chose a site index i, remove the approximate likelihood term corresponding to site i, and marginalise all other variables. The resulting distribution

$$q_c(x_i) = \int \prod_{i \neq j} dx_j q(\mathbf{x}|\mathbf{y}) \tilde{t}^{-1}(y_i, x_i)$$

   is called the *cavity distribution*. This term, inherited from statistical physics, is intuitively interpreted as the influence site i would feel from the other sites *if it was not there*.
2. The exact likelihood term corresponding to site i is reintroduced forming the so-called *tilted distribution*, $q_c(x_i)t_i(x_i, y_i)$ (which is in general not normalised).
3. The EP approximation is updated by replacing the initial approximate likelihood term for site i with a new term $\tilde{t}_i^{new}(x_i, y_i)$ obtained by finding the univariate Gaussian which matches the moments of the tilted distribution.
4. This update procedure is applied iteratively to all sites until the moment of the EP approximation do not change.

A formal justification of the EP algorithm is non-trivial: [28] showed that the EP free energy does indeed minimise a Gibbs free energy associated with the statistical model, however their argument is non-trivial and out of the scope of this paper. Importantly, extensive empirical studies have confirmed the excellent accuracy of the EP algorithm for posterior computation in GP classification models [29], so that EP is effectively the state-of-the-art approximate inference algorithm in this field.

4.7. Practical considerations

Smoothed model checking relies on the density of the RKHS within the space of smooth functions; this restricts somewhat the choice of covariance function that can be used to model the satisfaction function. Nevertheless, several possible choices still remain besides the squared exponential kernel we use; we refer the interested reader to [24]. Each of these covariance functions is equipped with some hyperparameters; while the value of the hyperparameters does not affect the theoretical guarantees (i.e. the RKHS is dense in the smooth functions regardless of the kernel hyperparameters), their value may have practical repercussions on the quality of the reconstruction from a finite sample. Automatic estimators for hyperparameters can be obtained by type-II maximum likelihood methods [24] and these may be very useful when the dimensionality of the space is high, so that empirical, nonparametric estimates are difficult to obtain.

An important issue is the computational complexity of Smoothed Model Checking. As we will see in Section 5, leveraging the smoothness of the satisfaction function can dramatically reduce the number of simulations required for estimation; this however comes at a cost. GP regression involves the inversion of the covariance matrix estimated at all pairs of parameter values; if we have n points in our grid of parameter values, this comes at a complexity $O(n^3)$. This complexity can be alleviated by using sparse approximations (again intensely studied in the last ten years, see [24] for a review). Further savings may be obtained by an intelligent choice of grid points, using strategies such as Latin Hypercube Sampling (see e.g. [9]).

Our theoretical results Theorems 1 and 2 ensure that the statistical model we employ for the satisfaction function is correct; therefore, powerful arguments from asymptotic statistics guarantee that, when the number of samples increases, the estimated satisfaction function (GP posterior mean) will approach the true satisfaction function. Regarding the number of samples that are necessary to achieve a certain level of accuracy, our advice is to monitor the estimated predictive uncertainty (GP posterior variance). The predictive uncertainty also diminishes as the number of samples increases, and monitoring of this quantity would enable the user to obtain the desired level of precision.

Running example – Poisson process: Part III We have seen previously how the satisfaction probability of the formula in (1) could be estimated statistically at a discrete set of parameter values for the Poisson process example. We shall now use this data as input to the smoothed model checking approach, where we place a GP prior with squared exponential covariance with amplitude 1 and characteristic length scale 1. GP hyper-parameters were not optimised in this case. The GP posterior, which constitutes an analytical approximation to the satisfaction function, can be seen in Fig. 2 for 5 and 20 observations per input value (left and right figures correspondingly). More specifically, we plot the GP posterior mean and the associated 95% confidence intervals. We also plot the true satisfaction function of (1), as given in (2).

We observe that smoothed model checking is consistently more accurate than the naïve SMC approach, although the very same SMC data has been used as input. This can be attributed to the fact that we have transferred information across neighbouring values, by exploiting the smoothness of satisfaction function. Moreover, the approximation quality of smoothed model checking is improved as we increase the number of samples. A more extensive evaluation of the quality and reliability of our approach follows in the experiments of Section 5.

4.8. Implementation

An implementation of our approach is available in the U-check tool [30], which offers a framework for formal analysis of models under parametric uncertainty. U-check can perform smoothed model checking for models specified in formal

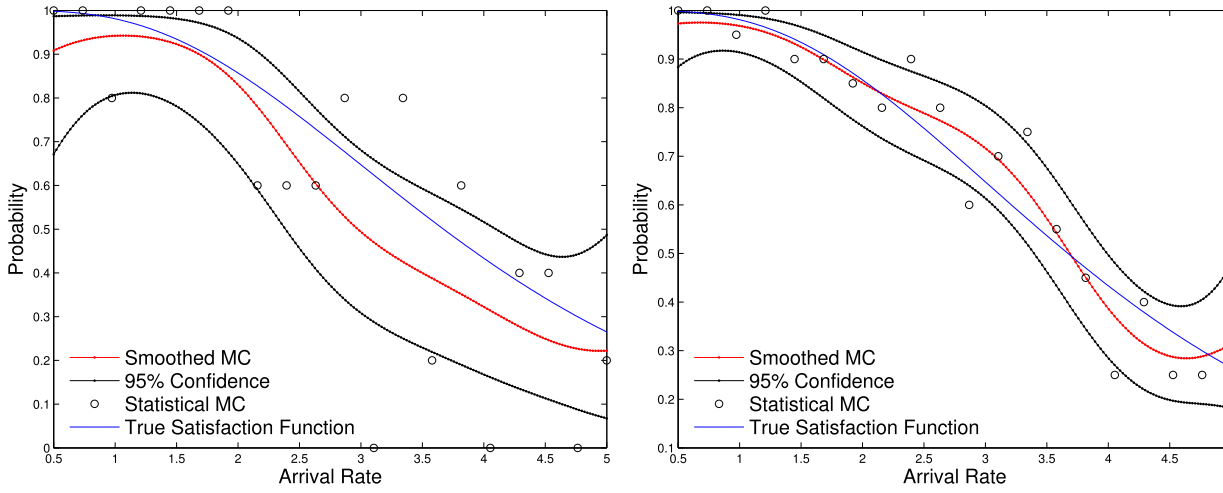

Fig. 2. Estimated satisfaction function for Poisson process from five samples per input value (left) and twenty samples per value (right). Blue: true satisfaction function; red: mean estimate; black: estimated 95% confidence intervals. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

modelling languages such as PRISM [1] and Bio-PEPA [31]; there is also support for stochastic hybrid models specified in the SimHyA modelling language [32]. U-check is available to download at https://github.com/dmilios/U-check.

5. Experiments

In this section, we present a series of experiments that demonstrate the potential of smoothed model checking.

5.1. Network epidemics

As a first, simpler example, we consider the classical SIR model of the spread of an epidemic in a population of fixed size [33], in which an initial population of susceptible nodes can be infected after contact with an infectious individual. Infected nodes can recover after some time, and become immune from the infection; such models play an important role not only in epidemiology, but also in computer science, for instance as models of the spread of software worms. Here we consider the case of permanent immunisation.

This system is modelled as a population CTMC, in which the state space is described by a vector $\vec{X}$ of three variables, counting how many nodes are susceptible (X_S), infected (X_I), and recovered (X_R). The dynamics of the CTMC are described by a list of transitions rules, or reactions, together with their rate functions, as discussed in Section 2.1.

Infection: $S + I \xrightarrow{k_i} I + I$, with rate function $k_i X_S X_I$;
Recovery: $I \xrightarrow{k_r} R$, with rate function $k_r X_I$;

Since immunisation is permanent, the epidemics extinguishes after a finite amount of time with probability one. However, the time of extinction depends on the parameters of the process in a non-trivial way. As for the transient dynamics before extinction, there are two possible behaviours depending on the *basic reproductive number* $R_0 = \frac{k_i}{k_r}$. If $R_0 < 1$, the epidemics extinguishes very quickly, while if $R_0 > 1$, there is an outbreak and a large fraction of the population can get infected. In this example, we consider the following MITL property, which states that the extinction happens between time 100 and 120:

$$\varphi = (X_I > 0)\mathbf{U}_{[100,120]}(X_I = 0) \tag{9}$$

We used smoothed model checking to explore each of the two parameters individually. We varied the infection rate k_i in the interval $[0.005, 0.3]$, holding k_r fixed to $k_r = 0.05$, while k_r has been varied within $[0.005, 0.2]$ with $k_i = 0.12$. In each case, we sampled 20 runs from a grid of 200 points evenly distributed in the parameter domain, for a total of 4000 runs, and applied smoothed model checking to obtain a prediction of the satisfaction probability of φ as a function of the parameter. Results are shown in Fig. 3, where we also plot the 95% confidence bounds of the estimate. The predicted satisfaction function is compared with point-wise estimates of the satisfaction probability using standard statistical model checking with 5000 runs per point. As we can see, our method provides an accurate reconstruction of the probability function, and a good estimation of the associated uncertainty, from a small number of simulation runs.

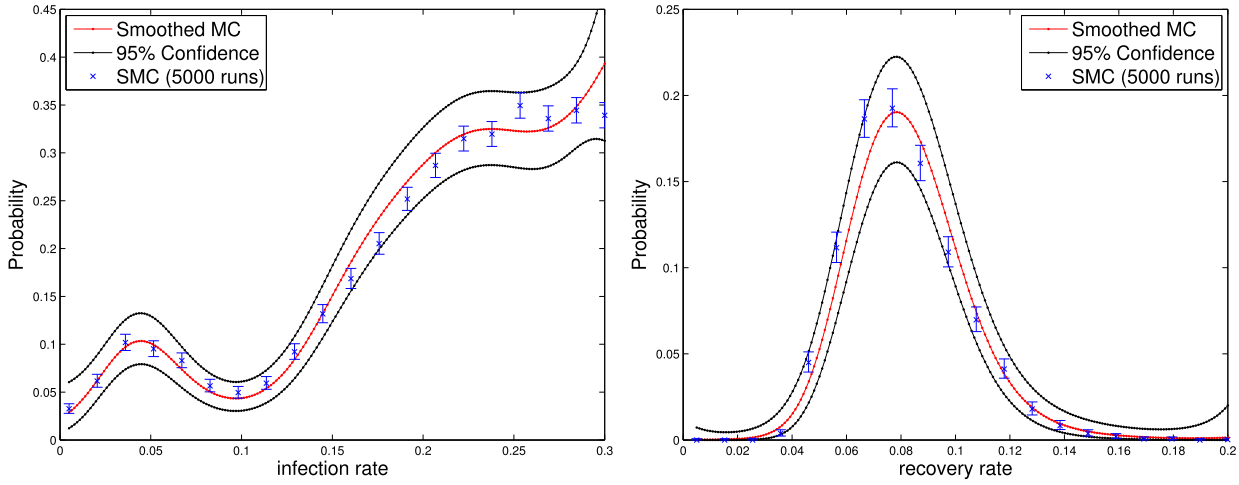

Fig. 3. Smoothed model checking reconstruction of the satisfaction probability of the extinction formula (9), as a function of $k_i \in [0.005, 0.3]$ with fixed $k_r = 0.05$ (left) and of $k_r \in [0.005, 0.2]$ with fixed $k_i = 0.12$ (right). The red line is the reconstructed probability and the black curves are the 95% confidence bounds. The blue crosses are estimates (and confidence bounds) of the satisfaction probability from standard SMC with 5000 runs per parameter value. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 1
RMSE for the extinction formula (SIR model); the k_i and k_r parameters have been explored simultaneously. A grid of 256 parameter values and a varying number of observations per value has been used. The true satisfaction function was approximated via SMC using 10000 simulation runs. The RMSE values presented are the average of 5 independent experiment iterations.

Obs. per value	RMSE			
	Bayesian SMC	Smoothed MC		
		64 points	100 points	256 points
5	0.1251 ± 0.013	0.0614 ± 0.015	0.0559 ± 0.011	0.0402 ± 0.007
10	0.0887 ± 0.010	0.0416 ± 0.022	0.0354 ± 0.011	0.0244 ± 0.003
20	0.0619 ± 0.006	0.0395 ± 0.020	0.0253 ± 0.007	0.0247 ± 0.010
50	0.0393 ± 0.006	0.0294 ± 0.012	0.0187 ± 0.005	0.0134 ± 0.002
100	0.0269 ± 0.003	0.0231 ± 0.004	0.0189 ± 0.003	0.0119 ± 0.001
200	0.0198 ± 0.002	0.0203 ± 0.005	0.0139 ± 0.001	0.0102 ± 0.002

Table 1 presents a more extensive evaluation of our method. The model parameters $k_i = [0.005, 0.3]$ and $k_r = [0.005, 0.12]$ have been explored simultaneously; we have estimated the satisfaction probability of (9) via smoothed model checking for a grid of 256 evenly distributed parameter values. We have then calculated the squared differences between the predicted and the true satisfaction probabilities, where the latter have been approximated via SMC using 10000 simulation runs. We have finally recorded the *Root-Mean-Square Error* (RMSE), which is the square root of the average over the specified grid of values. We report the average RMSE values for 5 independent experiment iterations, plus-minus 2 times the standard deviation observed. The objective of this experiment is to examine how the performance of smoothed MC is affected by the quantity and quality of the training data. We varied the number of training parameter locations on grids of 64, 100 and 256 points, and also varied the number of simulation runs per parameter value 5 to 200. As a comparison, we present the results obtained a Bayesian SMC approach with beta prior *at the actual test parameter values*, for varying number of simulation runs. As expected, the accuracy of smoothed MC increases as we increase either the points in the grid, or the number of simulations per point.

It is worth commenting that the confidence regarding the estimated satisfaction probability is also increased. Table 2 summarises the average width of the 95% confidence intervals as given by the GP for the smoothed MC method. Our method is consistently more accurate than Bayesian SMC for the same number of simulations; remarkably, for grids of 256 points, smoothed MC achieves the same accuracy as Bayesian SMC with an order of magnitude less simulation runs per value. It is important to stress that, as well as providing increased accuracy, smoothed MC also provides an analytical approximation to the whole satisfaction function, which is not available for standard SMC. This means that we can get an estimate of the satisfaction probability outside the training points, at practically no additional computational cost. Furthermore, the width of the confidence interval will be similar to that of nearby training points. On the contrary, SMC requires to perform additional simulations at the new point, incurring a much larger computational burden.

Finally, Table 3 presents information on the running times of performing Smoothed MC for the SIR model. We compare our approach with Bayesian SMC using 100 simulation runs, whose accuracy has been roughly similar to Smoothed MC with 10 simulation runs and 256 points, as seen in Table 1. For the particular model considered, stochastic simulation has

Table 2
Confidence intervals for the extinction formula (SIR model) as given by smoothed MC for varying number of observations per value.

Obs. per value	Avg. Width of 95% C.I.		
	64 points	100 points	256 points
5	0.1522	0.1209	0.1062
10	0.1249	0.1071	0.0777
20	0.0975	0.0796	0.0583
50	0.0700	0.0577	0.0393
100	0.0505	0.0433	0.0296
200	0.0373	0.0310	0.0231

Table 3
Running times of model checking the extinction formula (SIR model). The k_i and k_r parameters have been explored simultaneously over a grid of 256 parameter values. The experiments have been performed in an Intel® Xeon® E5-1620 v3 @ 3.50 GHz PC running Linux.

Method		64 points	100 points	256 points
Smoothed MC	SMC (10 runs)	0.05 sec	0.12 sec	0.24 sec
	Hyperparam. Opt.	5 sec	8 sec	34 sec
	GP Prediction	0.4 sec	0.8 sec	2.5 sec
	Total	5.45 sec	8.92 sec	36.74 sec
Bayesian SMC (100 runs)*		1.2 sec		

* Bayesian SMC with 100 runs results in roughly similar RMSE values to smoothed MC with 10 runs and 256 points, as seen in Table 1.

been inexpensive, therefore the decrease in the number of simulation runs did not pay off from a practical perspective. The computational overhead of Smoothed MC has been significantly larger than any savings achieved, due to the cost of hyperparameter optimisation. This however can be attributed to the simplicity of the model, for which simulation is very fast. In the rest of this section, we view examples where stochastic simulation constitutes a computational bottleneck in parameter exploration, and therefore Smoothed MC delivers a substantial decrease in the analysis time. Most importantly, we stress that Smoothed MC produces an analytical approximation to the satisfaction function; any overhead is bound to be constant, regardless of the number of points at which the satisfaction probability needs to be explored.

5.2. Viral infection

In this section, we consider is the viral infection model appeared in [34], whose stiff behaviour renders stochastic simulation particularly expensive. It is therefore an appropriate example to demonstrate the computational benefits of smoothed model checking. The model is described by the following set of reactions:

$$
\begin{aligned}
r_1&: \quad \text{nucleotides} + T \xrightarrow{k_1} G + T\\
r_2&: \quad \text{nucleotides} + G \xrightarrow{k_2} T\\
r_3&: \quad \text{nucleotides} + \text{amino acids} + T \xrightarrow{k_3} S + T\\
r_4&: \quad T \xrightarrow{k_4} \emptyset\\
r_5&: \quad S \xrightarrow{k_5} \emptyset\\
r_6&: \quad G + S \xrightarrow{k_6} V
\end{aligned}
$$

It is assumed that nucleotides and amino acids are available at constant concentrations, and their contributions to the rate functions are encoded in the model parameters. The system keeps track of the populations of three species: the viral template T, the viral genome G, and the viral structural protein S. In the initial state, we assume 1 molecule for T and zero for the rest of the species. In this experiment, we vary coefficients c_n and c_a which control the concentrations of nucleotides and amino acids correspondingly. By default we have $c_n = 1$ and $c_a = 1$, which correspond to the default kinetic constants of the model. The rest of the parameters are summarised in Table 4.

In this experiment, we measure the probability of a delayed infection, i.e. the viral genome population grows significantly, after it has remained under a certain threshold for a considerable amount of time. This property is captured by the following formula:

$$\varphi = \mathbf{G}_{[0,50]} X_G \leq 10 \ \wedge \ \mathbf{F}_{[50,200]} X_G > 100 \tag{10}$$

Table 4
Rate functions and default parameter values for the viral model.

Reaction	Rate function	Default parameter
r_1	$k_1 X_T c_n$	$k_1 = 1$
r_2	$k_2 X_G c_n$	$k_2 = 0.025$
r_3	$k_3 X_T c_n c_a$	$k_3 = 100$
r_4	$k_4 X_T$	$k_4 = 0.25$
r_5	$k_5 X_S$	$k_5 = 0.2$
r_6	$k_6 X_G X_S$	$k_6 = 7.5 \times 10^{-6}$

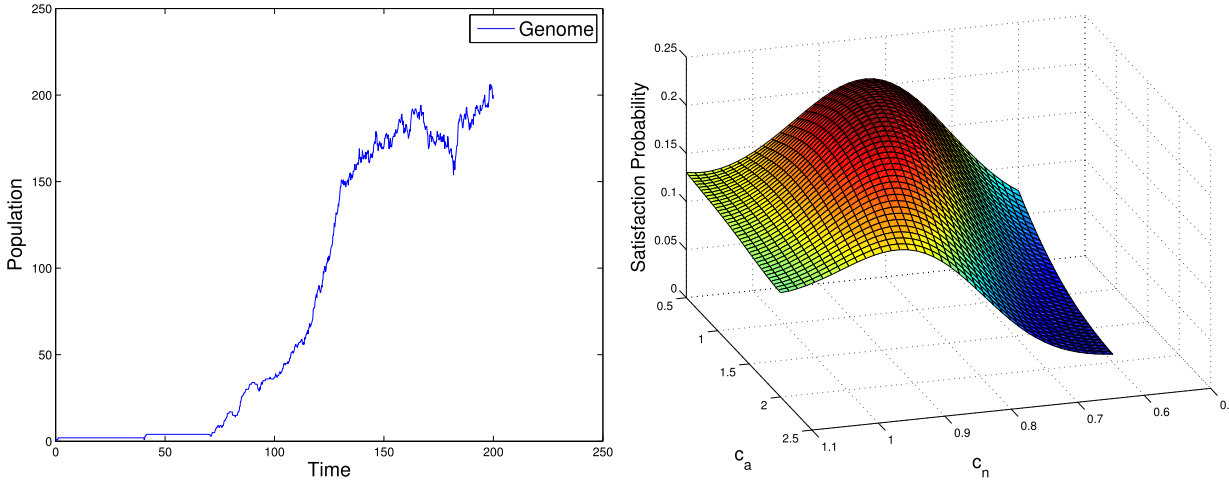

Fig. 4. Left: A sample trajectory from the viral model that satisfies the formula in (10). Right: Smoothed model checking reconstruction of the satisfaction probability of (10), as a function of $c_n \in [0.6, 1.1]$ and $c_a \in [0.5, 2]$.

Table 5
RMSE for the viral model; c_n and c_a have been explored simultaneously. A grid of 256 parameter values and a varying number of observations per value has been used. The true values were approximated via SMC using 2000 simulation runs. The MSE values presented are the average of 5 independent experiment iterations.

Obs. per value	RMSE		
	Bayesian SMC	Smoothed MC	
		100 points	256 points
5	0.1540 ± 0.007	0.0485 ± 0.007	0.0347 ± 0.023
10	0.1073 ± 0.008	0.0454 ± 0.004	0.0201 ± 0.005
20	0.0783 ± 0.005	0.0324 ± 0.019	0.0167 ± 0.006
50	0.0472 ± 0.004	0.0169 ± 0.004	0.0133 ± 0.002
100	0.0346 ± 0.001	0.0130 ± 0.002	0.0119 ± 0.002
200	0.0261 ± 0.003	0.0111 ± 0.000	0.0095 ± 0.002

An example of a trajectory that satisfies this property can be seen in Fig. 4. We estimate the probability that the formula in (10) is satisfied for $c_n \in [0.6, 1.1]$ and $c_a \in [0.5, 2]$. The reconstruction of the satisfaction probability as a function of both of these parameters can be seen on the right-hand side of Fig. 4.

A more extensive experimental evaluation of our method is summarised in Table 5, which reports the errors observed when estimating the satisfaction probability of (10) for a grid of 256 points in the parameter space. The true probabilities have been approximated via SMC using 2000 simulation runs. The RMSE values reported are the averages of 5 independent experiment iterations, plus-minus 2 times the standard deviation observed. We demonstrate that increasing the number of samples improves approximation quality in two ways: we vary the number of observations per value from 5 to 200, and the size of the training set for the GP (100 and 256 points). Compared against Bayesian SMC, our approach has been significantly more accurate for a given number of samples. This is translated to significant computational savings, as demonstrated in Table 6, where the execution times of our method is compared with a Bayesian SMC approach that achieves similar accuracy. Smoothed MC achieves a certain level of accuracy with fewer simulation runs.

Table 6
Running times of model checking the viral model. c_n and c_a have been explored simultaneously over a grid of 256 parameter values. The experiments have been performed in an Intel® Xeon® E5-1620 v3 @ 3.50 GHz PC running Linux.

Method		100 points	256 points
Smoothed MC	SMC (10 runs)	49 sec	132 sec
	Hyperparam. Opt.	15 sec	54 sec
	GP Prediction	1 sec	4 sec
	Total	65 sec	190 sec
Bayesian SMC (100 runs)*		1204 sec	

* Bayesian SMC with 100 runs results in similar RMSE to smoothed MC with 10 runs and 100 points, as seen in Table 5.

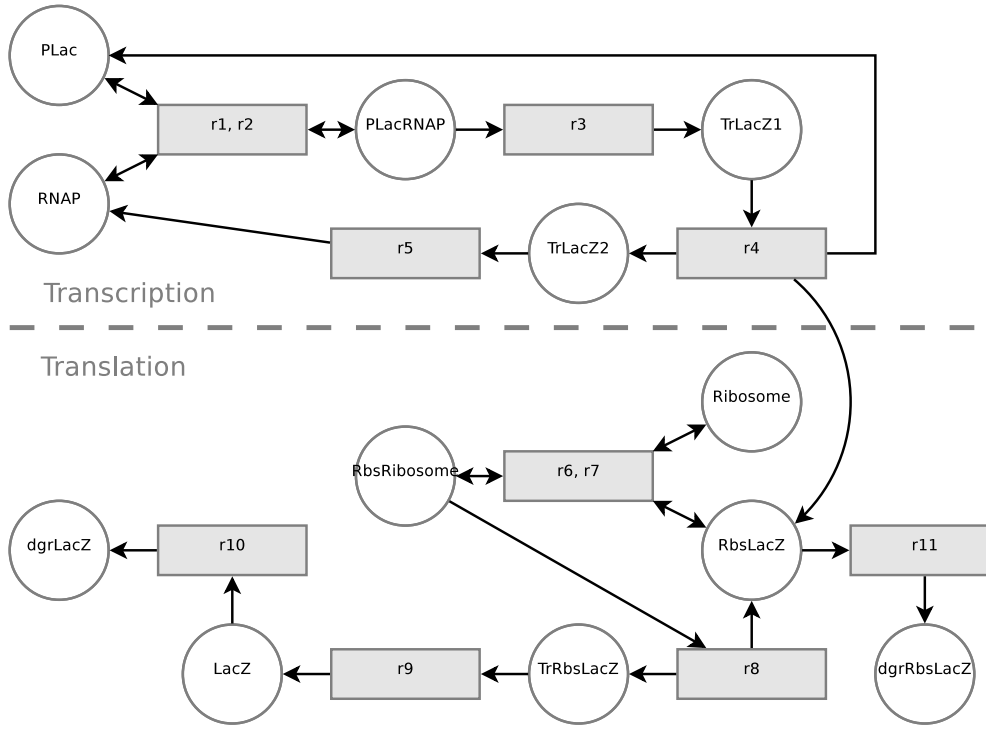

Fig. 5. Schematic representation of LacZ expression.

5.3. *Bursting behaviour in prokaryotic gene expression*

Systems biology forms a major application area of modern formal modelling. This relatively recent development was brought about by novel experimental techniques that demonstrated the stochasticity of biology at the single cell level [35], and by pioneering work linking formal finite state machines to biological processes [36]. These resulted in a strong demand for tools to model and reason formally on biological systems at the microscopic scale. However, biological processes usually depend on large numbers of uncertain parameters, creating a need for reasoning in the presence of parametric uncertainty. A prominent example is given by a CTMC model of prokaryotic gene expression introduced in [37], which describes LacZ protein synthesis in E. coli. The primary purpose of the original model was to investigate how stochasticity depended on model parameters, and a computationally intensive brute-force exploration was presented in [37].

A schematic representation of the model is depicted in Fig. 5. The rate functions and the default values for the rate constants are summarised in Table 7. The first two reactions describe the binding and dissociation of RNA polymerase (RNAP) with the promoter (PLac).

r_1: $\quad \text{PLac} + \text{RNAP} \xrightarrow{k_1} \text{PLacRNAP}$

r_2: $\quad \text{PLacRNAP} \xrightarrow{k_2} \text{PLac} + \text{RNAP}$

The reactions that follow model RNA transcription, and the release of the promoter and the RNA polymerase. The product of these reactions is the ribosome binding site for LacZ (RbsLacZ). The transcription initiation frequency effectively depends on

Table 7
Rate functions and default parameter values for the LacZ model.

Reaction	Rate function	Default parameter
r_1	$k_1 X_{\text{PLac}} X_{\text{RNAP}}$	$k_1 = 0.17$
r_2	$k_2 X_{\text{PLacRNAP}}$	$k_2 = 10$
r_3	$k_3 X_{\text{PLacRNAP}}$	$k_3 = 1$
r_4	$k_4 X_{\text{TrLacZ1}}$	$k_4 = 1$
r_5	$k_5 X_{\text{TrLacZ2}}$	$k_5 = 0.015$
r_6	$k_6 X_{\text{Ribosome}} X_{\text{RbsLacZ}}$	$k_6 = 0.17$
r_7	$k_7 X_{\text{RbsRibosome}}$	$k_7 = 0.45$
r_8	$k_8 X_{\text{RbsRibosome}}$	$k_8 = 0.4$
r_9	$k_9 X_{\text{TrRbsLacZ}}$	$k_9 = 0.015$
r_{10}	$k_{10} X_{\text{LacZ}}$	$k_{10} = 6.42 \times 10^{-5}$
r_{11}	$k_{11} X_{\text{RbsLacZ}}$	$k_{11} = 0.3$

the promoter strength; in the experiments, this is decreased by increasing the RNA polymerase dissociation rate (k_2 kinetic constant).

r_3: $\text{PLacRNAP} \xrightarrow{k_3} \text{TrLacZ1}$
r_4: $\text{TrLacZ1} \xrightarrow{k_4} \text{RbsLacZ} + \text{PLac} + \text{TrLacZ2}$
r_5: $\text{TrLacZ2} \xrightarrow{k_5} \text{RNAP}$

The attachment of a ribosome to a binding site, and the corresponding dissociation, are given as follows:

r_6: $\text{Ribosome} + \text{RbsLacZ} \xrightarrow{k_6} \text{RbsRibosome}$
r_7: $\text{RbsRibosome} \xrightarrow{k_7} \text{Ribosome} + \text{RbsLacZ}$

The following reactions describe the last two steps in gene expression: translation and protein synthesis. In the experiments, the translation initiation frequency is decreased by increasing the ribosome dissociation rate (k_7 kinetic constant).

r_8: $\text{RbsRibosome} \xrightarrow{k_8} \text{TrRbsLacZ} + \text{RbsLacZ}$
r_9: $\text{TrRbsLacZ} \xrightarrow{k_9} \text{LacZ}$

Finally, the reactions that follow model the degradation of the LacZ protein and the transcribed RNA correspondingly.

r_{10}: $\text{LacZ} \xrightarrow{k_{10}} \text{dgrLacZ}$
r_{11}: $\text{RbsLacZ} \xrightarrow{k_{11}} \text{dgrRbsLacZ}$

As reported in [37], decreasing the transcription initiation rate results in a behaviour that is characterised by irregular "bursts" of gene expression. We formalise the concept of burst as a MITL formula which monitors rapid increases in LacZ counts, followed by long periods of lack of protein production. The resulting formula is

$$\varphi = \mathbf{F}_{[16000,21000]}(\Delta X_{\text{LacZ}} > 0 \ \wedge \ \mathbf{G}_{[10,2000]} \Delta X_{\text{LacZ}} \leq 0) \tag{11}$$

which will be true if a burst of gene expression is present in a particular time-window, from 16000 to 21000 seconds.

While models such as this one are highly parametrised, it is often desirable to explore the behaviour at the model while a limited number of control parameters are varied. This is motivated both by the different precision to which parameters can be measured, and, more importantly, the fact that some parameters may be directly manipulated in a synthetic biology application. The parameters explored here are the RNA polymerase dissociation rate k_2, and the ribosome dissociation rate k_7. We vary both $k_2 \in [1000, 10000]$ and $k_7 \in [45, 450]$ simultaneously. The reconstruction of the satisfaction probability as a function of these two parameters can be seen in Fig. 6. Notice how the figure immediately reveals a non-trivial correlation structure between the parameters, with high bursting probability being achieved for anti-correlated parameters (either high k_2 and low k_7, or vice-versa). From the modelling point of view, the ability of quickly identifying such dependency structures could be a very valuable tool.

Table 8 summarises the RMSE for estimating the satisfaction probability of the expression burst formula in (11) for a grid of 256 points; the true probabilities have been approximated via SMC using 2000 simulation runs. We report the average RMSE values for 5 independent experiment iterations, plus-minus 2 times the standard deviation observed. Smoothed MC has been applied using grids of 100 and 256 points as the training set for the GP. In both cases, smoothed MC is much more accurate than Bayesian SMC for the same number of observations per value. As a consequence, it is possible to get accurate estimates for the satisfaction probability with only few simulation runs per parameter value. The computational savings of smoothed MC are reflected in Table 9, where we compare the execution times of our method with a Bayesian SMC approach

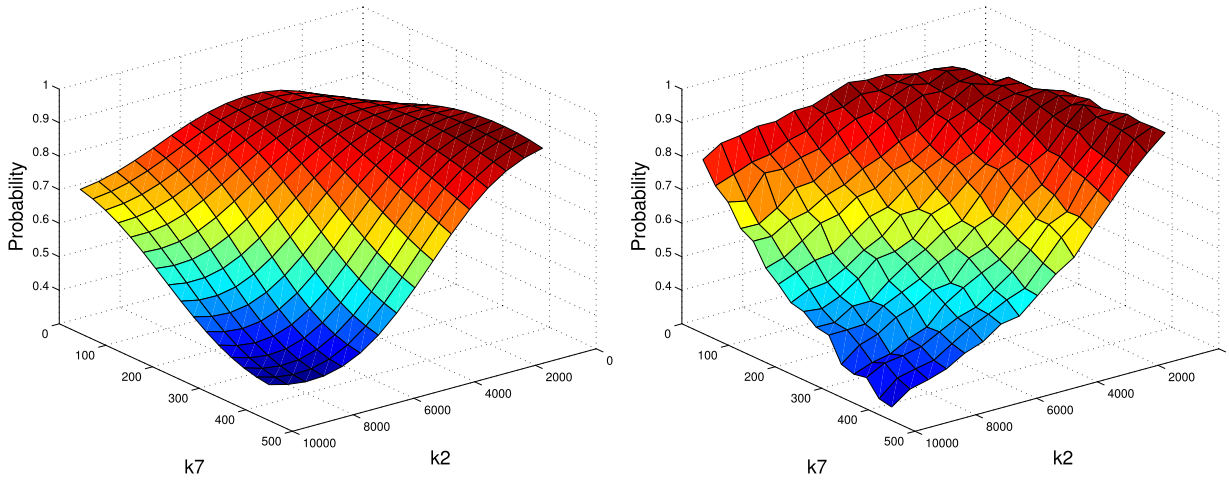

Fig. 6. Left: smoothed model checking reconstruction of the satisfaction probability of the expression burst formula in (11), as a function of $k_2 \in [1000, 10000]$ and $k_7 \in [45, 450]$; a grid of 100 points was used as training set for the GP. Right: the same satisfaction probability function as estimated by Bayesian statistical model checking.

Table 8
RMSE for the expression burst formula (LacZ model); the k_2 and k_7 parameters have been explored simultaneously. A grid of 256 parameter values and a varying number of observations per value has been used. The true values were approximated via SMC using 2000 simulation runs. The MSE values presented are the average of 5 independent experiment iterations.

Obs. per value	RMSE		
	Bayesian SMC	Smoothed MC	
		100 points	256 points
5	0.1819 ± 0.017	0.0513 ± 0.033	0.0390 ± 0.011
10	0.1246 ± 0.008	0.0412 ± 0.006	0.0322 ± 0.003
20	0.0954 ± 0.010	0.0375 ± 0.011	0.0248 ± 0.003
50	0.0588 ± 0.005	0.0266 ± 0.005	0.0200 ± 0.004
100	0.0427 ± 0.006	0.0200 ± 0.002	0.0159 ± 0.001
200	0.0309 ± 0.003	0.0171 ± 0.002	0.0141 ± 0.002

Table 9
Running times of model checking the expression burst formula (LacZ model). The k_2 and k_7 parameters have been explored simultaneously over a grid of 256 parameter values. The experiments have been performed in an Intel® Xeon® E5-1620 v3 @ 3.50 GHz PC running Linux. Bayesian SMC with 100 runs results in roughly the same RMSE with smoothed MC with 10 runs, as seen in Table 8.

Method		100 points	256 points
Smoothed MC	SMC (10 runs)	31 sec	110 sec
	Hyperparam. Opt.	8 sec	30 sec
	GP Prediction	1 sec	2 sec
	Total	40 sec	142 sec
Bayesian SMC (100 runs)*		1100 sec	

* Bayesian SMC with 100 runs results in roughly the same RMSE with smoothed MC with 10 runs, as seen in Table 8.

that achieves roughly similar accuracy in terms of RMSE. The cost of SMC is dominated by simulation, therefore decreasing the simulation runs for smoothed MC resulted in a great reduction of the analysis time.

6. Conclusions

Verification of logical properties over uncertain stochastic processes is an important task in theoretical computer science. This paper offers contributions on two different levels to this model checking problem: from the theoretical point of view, it refocuses the question of model checking from estimating a number (the satisfaction probability of a formula) to estimating a *function*, which we term the satisfaction function of the formula. Our main theoretical result characterises the satisfaction

function as a differentiable function of the parameters of the process, under mild conditions. From the practical point of view, this smoothness result offers a powerful new way to perform statistical model checking: intuitively, the power of statistical model checking, deriving from the law of large numbers, can be increased by simultaneous sampling at nearby values of the parameters. We show that Gaussian Processes provide a natural prior distribution over smooth functions which can be employed in a Bayesian statistical model checking framework to evaluate a whole satisfaction function from a relatively small number of samples. We term this novel approach to model checking *Smoothed Model Checking*, and show in an empirical section that indeed this approach can be extremely efficient and accurate.

Model checking of parametric models has been considered in [38] for *Discrete-Time Markov Chains* (DTMCs); the method has been further developed into software tools [39,40]. These approaches involve constructing symbolic representations for the satisfaction probability of temporal logic properties. Such a strategy is not readily applicable for CTMCs, unless a (possibly expensive) uniformisation scheme is applied. However, our method can be easily adapted to parametric DTMCs, provided that the transition probabilities are smooth functions of the parameters in question. The advantage of a statistical approach, such as ours, is the ability to have more expressive power in terms of temporal logic, including nested temporal operators and time-bounded properties. The exploration of this possibility is an interesting direction for future work.

The availability of quick methods for estimating a satisfaction function could be of considerable use in tackling other computational problems: for example, such a method could be used to inform system design approaches, or to guide effectively and efficiently importance sampling strategies for SMC [41] to identify combinations of parameters giving large probability to the rare event we wish to estimate. We stress, however, that the purpose of importance sampling methods is somewhat complementary to our aim: while importance sampling focuses on producing accurate estimates of rare event probabilities at specific values of the parameters, our method provides a global analytical approximation to the satisfaction function. To our knowledge, such global approximations have not been studied so far, and could represent a novel direction in statistical model checking. An interesting direction could be to combine our method with computationally intensive methods which provide strong error guarantees such as [16]. This could result into a provably correct parameter synthesis tool which quickly identifies good candidate regions of the parameter space via smoothed MC, to be then characterised more rigorously using the method of [16].

The cross fertilisation of ideas from machine learning and model checking is increasingly being exploited in the development of novel algorithms in both fields. GP-based methods in particular are becoming part of the formal modelling repertoire [9,42,43]. A practical limitation shared by all GP-based methods is the cubic complexity of the matrix inversion operations needed in GP prediction. Scaling our method to large spaces of parameters would therefore necessitate the use of approximate methods to alleviate the computational burden; we envisage that ideas from sparse GP prediction could be very effective in this scenario [24].

Appendix A. Proofs

Theorem 1. *Let φ be a MITL formula and let $\mathcal{M}_\theta$ be an uncertain CTMC indexed by the variable $\theta \in D$. Denote as $\boldsymbol{\tau}(\vec{X}, \theta)$ the transition rates of the CTMCs and assume that these depend smoothly on the parameters θ and polynomially on the state vector of the system $\vec{X}$. Then, the satisfaction function of φ is a smooth function of the parameters, $f_\varphi \in \mathcal{C}^\infty(D)$.*

Proof. We begin the proof by elucidating the topology of the space of trajectories of the system.

Structure of space of trajectories Let $\mathcal{T}$ denote the space of trajectories of the system starting at $t = 0$ and ending at $t = T$. W.l.o.g. assume that from any state of the system there are at most R different types of transitions that can occur (in a biochemical example, these would be all the possible reactions in the system). We notice that each trajectory in $[0, T]$ can be uniquely defined by specifying the number of transitions k, the sequence of types of transitions $r_i \quad i = 1, \ldots, k$ and the times of the transitions $t_i, \quad i = 1, \ldots, k$. This implies that $\mathcal{T}$ is a countable union of finite dimensional spaces $\mathcal{T}_k$ corresponding to trajectories where exactly k transitions have occurred. Each of the $\mathcal{T}_k$ is in itself the union of R^k copies of $[0, T]^k$; each of these copies corresponds to a sequence of k transitions (there are R^k such sequences), and a point in $[0, T]^k$ determines the times at which the transitions happened. Notice that, given a trajectory of the system, i.e. a point in $\mathcal{T}$, the formula φ is either true or false; this implies that the *set* of trajectories which satisfy the formula φ is independent of the model parameters, hence we only need to show that the measure on the set of trajectories depends smoothly on the parameters.

Measure on the set of trajectories – state-independent rates The topology of the space of trajectories corresponds to the well known factorisation of the measure over the space of trajectories; denoting a trajectory $\tau = [(r_1, t_1), \ldots, (r_k, t_k)]$, we have that

$$p(\tau) = p(k)p(r_1, \ldots, r_k|k)p(t_1, \ldots, t_k|r_1, \ldots, r_k).$$

In the case of state-independent rates, $p(k)$ is a Poisson distribution over the number of transitions (with mean μ given by the inverse of the sum of the rates times T), $p(r_1, \ldots, r_k|k)$ is the probability of the choice of the k transition types (this is a rational function with positive denominator in the rates) and $p(t_1, \ldots, t_k|r_1, \ldots, r_k)$ is the product of the exponential

probabilities of the k waiting times. This density is clearly smooth w.r.t. the parameters (rates); to prove the integrability of its derivative, we have to show that the absolute value of the derivative is bound by a quantity which is integrable. As the domains $[0, T]$ and D are bounded, all we have to verify is that the derivatives do not grow too fast as $k \to \infty$ to ensure integrability. This is easily verified directly for the three terms in the derivative:

1. $\frac{\partial p(k)}{\partial \theta} = p(k-1)\frac{\partial \mu}{\partial \theta}$ where we exploited the fact that the derivative of a Poisson probability of k events w.r.t. the mean is equal to the probability of $k-1$ events.
2. $\frac{\partial p(r_1,\ldots,r_k|k)}{\partial \theta} < kM$ where M is a constant, as $p(r_1,\ldots,r_k|k)$ is a rational function with positive denominator and both numerator and denominator polynomials of degree at most k in each of the rates.
3. $\frac{\partial p(t_1,\ldots,t_k|r_1,\ldots,r_k)}{\partial \theta} < kLp(t_1,\ldots,t_k|r_1,\ldots,r_k)$ where L is a constant depending on the rates and on T, as $p(t_1,\ldots,t_k|r_1,\ldots,r_k)$ is a product of exponentials with exponents linear in the rates divided by a normalising constant which is a monomial of degree at most k in the rates.

Therefore, in the case where the rates do not depend on the state of the system, all derivative terms grow at most linearly with k, which means they are still integrable once multiplied by the Poisson density $p(k)$.

Measure on the set of trajectories – state-dependent rates In the case where the rates depend polynomially on the state of the system, the argument above can be modified in a straightforward manner to show that each derivative term grows at most as k^{c+1}, where c is the maximum polynomial order of the rates (w.r.t. the state variables). As a polynomial function is still integrable when multiplied by the Poisson density $p(k)$, by the same argument we obtain that the derivative of the density is still integrable over the space of trajectories, hence the first derivative of the satisfaction function exists.

We notice that repeated applications of the derivative operator still result in polynomial growth of the derivatives of the density with the number of transitions k. By the same argument, all the derivatives of finite order of the satisfaction function exist, hence $f_\varphi \in C^\infty(D)$. □

References

[1] M. Kwiatkowska, G. Norman, D. Parker, PRISM 4.0: verification of probabilistic real-time systems, in: Proc. of Computer Aided Verification, in: Lect. Notes Comput. Sci., vol. 6806, Springer, 2011, pp. 585–591.
[2] J.-P. Katoen, M. Khattri, I.S. Zapreevt, A Markov reward model checker, in: Proc. of Quantitative Evaluation of Systems, IEEE Computer Society, 2005, pp. 243–244.
[3] D.T. Gillespie, Exact stochastic simulation of coupled chemical reactions, J. Phys. Chem. 81 (25) (1977) 2340–2361.
[4] H.L. Younes, R.G. Simmons, Statistical probabilistic model checking with a focus on time-bounded properties, Inf. Comput. 204 (9) (2006) 1368–1409.
[5] P. Zuliani, A. Platzer, E.M. Clarke, Bayesian statistical model checking with application to Simulink/Stateflow verification, in: Proc. of Hybrid Systems: Computation and Control, ACM, 2010, pp. 243–252.
[6] M. Opper, G. Sanguinetti, Variational inference for Markov jump processes, in: Proc. of Advances in Neural Information Processing Systems, 2007, pp. 1105–1112.
[7] E. Bartocci, R. Grosu, P. Katsaros, C.R. Ramakrishnan, S.A. Smolka, Model repair for probabilistic systems, in: Proc. of Tools and Algorithms for the Construction and Analysis of Systems, in: Lect. Notes Comput. Sci., vol. 6605, Springer, 2011, pp. 326–340.
[8] A. Andreychenko, L. Mikeev, D. Spieler, V. Wolf, Approximate maximum likelihood estimation for stochastic chemical kinetics, EURASIP J. Bioinform. Syst. Biol. 2012 (1) (2012) 1–14.
[9] L. Bortolussi, G. Sanguinetti, Learning and designing stochastic processes from logical constraints, in: Proc. of Quantitative Evaluation of Systems, in: Lect. Notes Comput. Sci., vol. 8054, Springer, 2013, pp. 89–105.
[10] E. Bartocci, L. Bortolussi, L. Nenzi, G. Sanguinetti, System design of stochastic models using robustness of temporal properties, Theor. Comput. Sci. 587 (2015) 3–25.
[11] J.-P. Katoen, D. Klink, M. Leucker, V. Wolf, Three-valued abstraction for continuous-time Markov chains, in: Proc. of Computer Aided Verification, in: Lect. Notes Comput. Sci., vol. 4590, 2007, pp. 311–324.
[12] B. Caillaud, B. Delahaye, K.G. Larsen, A. Legay, M.L. Pedersen, A. Wasowski, Constraint Markov chains, Theor. Comput. Sci. 412 (34) (2011) 4373–4404.
[13] C. Baier, H. Hermanns, J.-P. Katoen, B.R. Haverkort, Efficient computation of time-bounded reachability probabilities in uniform continuous-time Markov decision processes, Theor. Comput. Sci. 345 (1) (2005) 2–26.
[14] M. Benedikt, R. Lenhardt, J. Worrell, LTL model checking of interval Markov chains, in: Proc. of Tools and Algorithms for the Construction and Analysis of Systems, in: Lect. Notes Comput. Sci., vol. 7795, 2013, pp. 32–46.
[15] L. Brim, M. Češka, S. Dražan, D. Šafránek, Exploring parameter space of stochastic biochemical systems using quantitative model checking, in: Proc. of Computer Aided Verification, in: Lect. Notes Comput. Sci., vol. 8044, 2013, pp. 107–123.
[16] M. Češka, F. Dannenberg, M. Kwiatkowska, N. Paoletti, Precise parameter synthesis for stochastic biochemical systems, in: Proc. of Computational Methods in Systems Biology, in: Lect. Notes Comput. Sci., vol. 8859, Springer, 2014, pp. 86–98.
[17] D.A. Spielman, S.-H. Teng, Smoothed analysis: an attempt to explain the behavior of algorithms in practice, Commun. ACM 52 (10) (2009) 76–84.
[18] R. Durrett, Essentials of Stochastic Processes, Springer, 2012.
[19] J.R. Norris, Markov Chains, Camb. Ser. Stat. Probab. Math., Cambridge University Press, 1997.
[20] L. Bortolussi, J. Hillston, D. Latella, M. Massink, Continuous approximation of collective systems behaviour: a tutorial, Perform. Eval. 70 (5) (2013) 317–349.
[21] P. Billingsley, Probability and Measure, Wiley, Hoboken, N.J., 2012.
[22] O. Maler, D. Nickovic, Monitoring temporal properties of continuous signals, in: Proc. of Formal Modelling and Analysis of Timed Systems, in: Lect. Notes Comput. Sci., vol. 3253, Springer, 2004, pp. 152–166.
[23] T. Chen, M. Diciolla, M. Kwiatkowska, A. Mereacre, Time-bounded verification of CTMCs against real-time specifications, in: Proc. of Formal Modelling and Analysis of Timed Systems, in: Lect. Notes Comput. Sci., vol. 6919, Springer, 2011, pp. 26–42.
[24] C.E. Rasmussen, C.K.I. Williams, Gaussian Processes for Machine Learning, MIT Press, 2006.
[25] I. Steinwart, On the influence of the kernel on the consistency of support vector machines, J. Mach. Learn. Res. 2 (2002) 67–93.

[26] M. Opper, O. Winther, Gaussian processes for classification: mean-field algorithms, Neural Comput. 12 (11) (2000) 2655–2684.
[27] T.P. Minka, Expectation Propagation for approximate Bayesian inference, in: Proc. of Uncertainty in Artificial Intelligence, Morgan Kaufmann Publishers Inc., 2001, pp. 362–369.
[28] T. Heskes, M. Opper, W. Wiegerinck, O. Winther, O. Zoeter, Approximate inference techniques with expectation constraints, J. Stat. Mech. Theory Exp. (2005).
[29] M. Kuss, C.E. Rasmussen, Assessing approximate inference for binary Gaussian process classification, J. Mach. Learn. Res. 6 (2005) 1679–1704.
[30] L. Bortolussi, D. Milios, G. Sanguinetti, U-check: model checking and parameter synthesis under uncertainty, in: Proc. of Quantitative Evaluation of Systems, in: Lect. Notes Comput. Sci., vol. 9259, 2015, pp. 89–104.
[31] F. Ciocchetta, J. Hillston, Bio-PEPA: a framework for the modelling and analysis of biological systems, Theor. Comput. Sci. 410 (33–34) (2009) 3065–3084.
[32] L. Bortolussi, V. Galpin, J. Hillston, Hybrid performance modelling of opportunistic networks, in: Proc. of Quantitative Aspects of Programming Languages, 2012, pp. 106–121.
[33] H. Andersson, T. Britton, Stochastic Epidemic Models and Their Statistical Analysis, Springer, 2000.
[34] E.L. Haseltine, J.B. Rawlings, Approximate simulation of coupled fast and slow reactions for stochastic chemical kinetics, J. Chem. Phys. 117 (15) (2002) 6959.
[35] M. Elowitz, A. Levine, E. Siggia, P. Swain, Stochastic gene expression in a single cell, Science 297 (5584) (2002) 1183–1186.
[36] J. Fisher, T. Henzinger, Executable cell biology, Nat. Biotechnol. 25 (11) (2007) 1239–1249.
[37] A.M. Kierzek, STOCKS: STOChastic kinetic simulations of biochemical systems with Gillespie algorithm, Bioinformatics 18 (3) (2002) 470–481.
[38] C. Daws, Symbolic and parametric model checking of discrete-time Markov chains, in: Proc. of Theoretical Aspects of Computing, in: Lect. Notes Comput. Sci., vol. 3407, Springer, 2005, pp. 280–294.
[39] E.M. Hahn, H. Hermanns, B. Wachter, L. Zhang, PARAM: a model checker for parametric Markov models, in: Proc. of Computer Aided Verification, in: Lect. Notes Comput. Sci., vol. 6174, Springer, 2010, pp. 660–664.
[40] C. Dehnert, S. Junges, N. Jansen, F. Corzilius, M. Volk, H. Bruintjes, J.-P. Katoen, E. Ábrahám, PROPhESY: a PRObabilistic ParamEter SYnthesis tool, in: Proc. of Computer Aided Verification, in: Lect. Notes Comput. Sci., vol. 9206, Springer, 2015, pp. 214–231.
[41] C. Jegourel, A. Legay, S. Sedwards, Cross-entropy optimisation of importance sampling parameters for statistical model checking, in: Proc. of Computer Aided Verification, in: Lect. Notes Comput. Sci., vol. 7358, 2012, pp. 327–342.
[42] E. Bartocci, L. Bortolussi, G. Sanguinetti, Data-driven statistical learning of temporal logic properties, in: Proc. of Formal Modelling and Analysis of Timed Systems, in: Lect. Notes Comput. Sci., vol. 8711, Springer, 2014, pp. 23–37.
[43] A. Legay, S. Sedwards, Statistical abstraction boosts design and test efficiency of evolving critical systems, in: Proc. of Leveraging Applications of Formal Methods, Verification and Validation, Technologies for Mastering Change, ISoLA, in: Lect. Notes Comput. Sci., vol. 8802, 2014, pp. 4–25.