Machine learning methods in Statistical Model Checking and System Design

Guido Sanguinetti[1,2]

[1]Adaptive and Neural Computation, School of Informatics
University of Edinburgh
[2]SynthSys, Synthetic and Systems Biology, University of Edinburgh

joint work with D. Milios (Edinburgh), and Luca Bortolussi (Saarbruecken/ Trieste)

RV Tutorial, 22/09/15

Outline

Broad picture

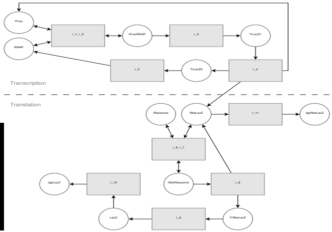
Machine learning and formal methods

Formal methods give us tools to run and reason about models. Machine learning gives us methods to refine models in the light of observations.

Ultra-condensed summary of this talk

Some formal modelling questions can be viewed as function estimation problems, for which machine learning methods yield efficient solutions.

Example 1



BIOINFORMATICS

Vol. 18 no. 3 2002
Pages 470–481

STOCKS: STOChastic Kinetic Simulations of biochemical systems with Gillespie algorithm

Andrzej M. Kierzek

Institute of Biochemistry and Biophysics, Polish Academy of Sciences, Pawinskiego 5a, 02-106 Warszawa, Poland

Received on June 5, 2001; revised on August 22, 2001; accepted on October 4, 2001

We build a model of a real system to investigate properties of the trajectories of the system.

Example 2

ELSEVIER

Electronic Notes in Theoretical Computer Science

Volume 310, 5 January 2015, Pages 179–199

Proceedings of the Seventh International Workshop on the Practical Application of Stochastic Modelling (PASM)

Open Access

Finding Optimal Timetables for Edinburgh Bus Routes

Ludovica Luisa Vissat

Allan Clark, Stephen Gilmore[1]

Sales pitch

- Building models of real-world systems involves uncertainty (typically in the parameters, but also in the structure)
- Verifying/ designing properties on uncertain models is unfeasible in a brute force way

Sales pitch

- Building models of real-world systems involves uncertainty (typically in the parameters, but also in the structure)
- Verifying/ designing properties on uncertain models is unfeasible in a brute force way
- Rest of the talk: use a smoothness result and treat the problem as a prediction/ optimisation problem in machine learning

The problems tackled

Satisfaction under Uncertainty Problem

Can we compute satisfaction probabilities of formal properties for uncertain models?

The problems tackled

Satisfaction under Uncertainty Problem

Can we compute satisfaction probabilities of formal properties for uncertain models?

Parameter Estimation problem

Can we learn model parameters from qualitative data, i.e. from observations of truth values of formal (temporal logic) properties?

The problems tackled

Satisfaction under Uncertainty Problem

Can we compute satisfaction probabilities of formal properties for uncertain models?

Parameter Estimation problem

Can we learn model parameters from qualitative data, i.e. from observations of truth values of formal (temporal logic) properties?

Synthesis problem

Can we learn model parameters satisfying (as robustly as possible) a set of formal (temporal logic) specifications?

Population CTMC

We consider population CTMC models, that are CTMC models in the biochemical reactions style.

Population CTMC

We consider population CTMC models, that are CTMC models in the biochemical reactions style.

State space

The state space is described by a set of n variables $\mathbf{X} = X_1, \ldots, X_n \in \mathbb{N}$, each counting the number of agents (jobs, molecules, ...) of a given kind.

Population CTMC

We consider population CTMC models, that are CTMC models in the biochemical reactions style.

State space

The state space is described by a set of n variables $\mathbf{X} = X_1, \ldots, X_n \in \mathbb{N}$, each counting the number of agents (jobs, molecules, ...) of a given kind.

Transitions

The dynamics is given by a set of chemical reactions, of the form

$$s_1 X_1 + \ldots + s_n X_n \to r_1 X_1 + \ldots + r_n X_n,$$

with a rate given by a function $f(\mathbf{X}, \theta)$, depending on the system variables and on a set of parameters θ.

Metric Interval Temporal Logic

- We express qualitative properties by Metric (interval) Temporal Logic.

Metric Interval Temporal Logic

- We express qualitative properties by Metric (interval) Temporal Logic.
- Linear time: in experiments we observe single realisations.

Metric Interval Temporal Logic

- We express qualitative properties by Metric (interval) Temporal Logic.
- Linear time: in experiments we observe single realisations.
- Metric bounds: we can observe a system only for a finite amount of time.

Syntax

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

As customary: $\mathbf{F}^{[T_1,T_2]}\varphi \equiv \mathtt{tt}\,\mathbf{U}^{[T_1,T_2]}\varphi$, $\mathbf{G}^{[T_1,T_2]}\varphi \equiv \neg\mathbf{F}^{[T_1,T_2]}\neg\varphi$.

MITL semantics

Semantics

μ are (non-linear) inequalities on vectors of n variables

- $\mathbf{x}, t \models \mu$ if and only if $\mu(\mathbf{x}(t)) = \mathtt{tt}$;
- $\mathbf{x}, t \models \varphi_1 \mathbf{U}^{[T_1, T_2]} \varphi_2$ if and only if $\exists t_1 \in [t + T_1, t + T_2]$ such that $\mathbf{x}, t_1 \models \varphi_2$ and $\forall t_0 \in [t, t_1]$, $\mathbf{x}, t_0 \models \varphi_1$.

MITL semantics

Semantics

μ are (non-linear) inequalities on vectors of n variables

- $\mathbf{x}, t \models \mu$ if and only if $\mu(\mathbf{x}(t)) = \mathtt{tt}$;
- $\mathbf{x}, t \models \varphi_1 \mathbf{U}^{[T_1,T_2]} \varphi_2$ if and only if $\exists t_1 \in [t + T_1, t + T_2]$ such that $\mathbf{x}, t_1 \models \varphi_2$ and $\forall t_0 \in [t, t_1], \mathbf{x}, t_0 \models \varphi_1$.

As customary: $\mathbf{F}^{[T_1,T_2]}\varphi \equiv \mathtt{tt}\mathbf{U}^{[T_1,T_2]}\varphi$, $\mathbf{G}^{[T_1,T_2]}\varphi \equiv \neg\mathbf{F}^{[T_1,T_2]}\neg\varphi$.

Can have both qualitative (boolean) and quantitative (robustness) semantics.

MITL semantics

Semantics

μ are (non-linear) inequalities on vectors of n variables

- $\mathbf{x}, t \models \mu$ if and only if $\mu(\mathbf{x}(t)) = \mathtt{tt}$;
- $\mathbf{x}, t \models \varphi_1 \mathbf{U}^{[T_1,T_2]} \varphi_2$ if and only if $\exists t_1 \in [t + T_1, t + T_2]$ such that $\mathbf{x}, t_1 \models \varphi_2$ and $\forall t_0 \in [t, t_1]$, $\mathbf{x}, t_0 \models \varphi_1$.

As customary: $\mathbf{F}^{[T_1,T_2]}\varphi \equiv \mathtt{tt}\mathbf{U}^{[T_1,T_2]}\varphi$, $\mathbf{G}^{[T_1,T_2]}\varphi \equiv \neg\mathbf{F}^{[T_1,T_2]}\neg\varphi$.

Can have both qualitative (boolean) and quantitative (robustness) semantics.

Example

$\mathbf{G}^{[0,100]}(X_I < 40) \wedge \mathbf{F}^{[0,60]}\mathbf{G}^{[0,40]}(5 \leq X_I \leq 20) \wedge \mathbf{G}^{[30,50]}(X_I > 30)$ encodes transient behaviour.

Model Checking

- Model checking: Given a stochastic process and a formula, compute the probability ϖ of the formula being true

Model Checking

- Model checking: Given a stochastic process and a formula, compute the probability ϖ of the formula being true
- Problem: almost always analytically impossible/ often computationally unfeasible

Model Checking

- Model checking: Given a stochastic process and a formula, compute the probability ϖ of the formula being true
- Problem: almost always analytically impossible/ often computationally unfeasible
- Statistical model checking: ϖ can be estimated as an average of the truth values T_i of the formula over many samples
- Can be given a Bayesian flavour by specifying priors $p(\varpi)$ and estimating *a posteriori* $p(\varpi|T_i)$ using the fact that $p(T_i|\varpi)$ is Bernoulli and Bayes' theorem
- This is equivalent to introducing pseudocounts$\rightarrow$ useful when rare events could be present

Outline

Uncertainty in models

- Quantitative models require parameters. Is domain expertise sufficient to select a particular value of the parameters (out of an uncountable set)?
- Certainly not in many applications like systems biology
- What are the implications for model checking?
- Satisfaction probabilities of formulae are defined for fully specified models.

All model checking algorithms rely on a fully specified model

Some further definitions

Uncertain CTMC

Consider a Population CTMC model $\mathcal{M}$ depending on a set of d parameters θ. We only know $\theta \in D$, but not their precise value. We call $\mathcal{M}$ an Uncertain CTMC (notation: $\mathcal{M}_\theta$ for fixed θ).

Some further definitions

Uncertain CTMC

Consider a Population CTMC model $\mathcal{M}$ depending on a set of d parameters $\boldsymbol{\theta}$. We only know $\boldsymbol{\theta} \in D$, but not their precise value. We call $\mathcal{M}$ an Uncertain CTMC (notation: $\mathcal{M}_{\boldsymbol{\theta}}$ for fixed $\boldsymbol{\theta}$).

Satisfaction function

Suppose we have a UCTMC $\mathcal{M}$, $\boldsymbol{\theta} \in \mathcal{D}$, and a Metric Temporal Logic formula φ. The the satisfaction probability of φ w.r.t. $\mathcal{M}$ is a function of $\boldsymbol{\theta}$:

$$p(\varphi \mid \boldsymbol{\theta}) = Prob\{\mathcal{M}_{\boldsymbol{\theta}}, x, 0 \models \varphi\}.$$

We call $p(\varphi \mid \boldsymbol{\theta})$ the satisfaction function.

Some further definitions

Uncertain CTMC

Consider a Population CTMC model $\mathcal{M}$ depending on a set of d parameters $\boldsymbol{\theta}$. We only know $\boldsymbol{\theta} \in D$, but not their precise value. We call $\mathcal{M}$ an Uncertain CTMC (notation: $\mathcal{M}_{\boldsymbol{\theta}}$ for fixed $\boldsymbol{\theta}$).

Satisfaction function

Suppose we have a UCTMC $\mathcal{M}$, $\boldsymbol{\theta} \in \mathcal{D}$, and a Metric Temporal Logic formula φ. The the satisfaction probability of φ w.r.t. $\mathcal{M}$ is a function of $\boldsymbol{\theta}$:

$$p(\varphi \mid \boldsymbol{\theta}) = Prob\{\mathcal{M}_{\boldsymbol{\theta}}, x, 0 \models \varphi\}.$$

We call $p(\varphi \mid \boldsymbol{\theta})$ the satisfaction function.

Goal: can we find statistical methods to approximate $p(\varphi \mid \boldsymbol{\theta})$ analytically?

Key property of satisfaction function

Theorem (Bortolussi, Milios and G.S. 2014)

Consider a Population CTMC model $\mathcal{M}$ whose transition rates depend polynomially on a set of d parameters $\boldsymbol{\theta} \in \mathcal{D}$. Let φ be a MITL formula. The satisfaction function $p(\varphi \mid \boldsymbol{\theta}) : \mathcal{D} \to [0, 1]$ is a smooth function of the parameters $\boldsymbol{\theta}$.

Key property of satisfaction function

Theorem (Bortolussi, Milios and G.S. 2014)

Consider a Population CTMC model $\mathcal{M}$ whose transition rates depend polynomially on a set of d parameters $\boldsymbol{\theta} \in \mathcal{D}$. Let φ be a MITL formula. The satisfaction function $p(\varphi \mid \boldsymbol{\theta}) : \mathcal{D} \to [0, 1]$ is a smooth function of the parameters $\boldsymbol{\theta}$.

This means that we can **transfer** information across neighbouring points. Can we define a statistical model checking technique which simultaneously computes the *whole satisfaction function*?

Smoothness proof (sketch)

- Use the fact that the space of trajectories of a CTMC is a countable union of finite dimensional spaces (indexed by the finite sequence of transitions)
- Write explicitly the measure on each finite dimensional space
- Show that the series of derivatives converges

Random functions

- Bayesian SMC uses the fact the satisfaction probability of a formula given a model is a number in $[0, 1]$, and prior distributions on numbers between $[0, 1]$ exist (Beta distribution)
- The object we are interested in is a smooth function. How do you construct probability distributions over random functions?

Random functions

- Bayesian SMC uses the fact the satisfaction probability of a formula given a model is a number in $[0, 1]$, and prior distributions on numbers between $[0, 1]$ exist (Beta distribution)
- The object we are interested in is a smooth function. How do you construct probability distributions over random functions?
- Simplest idea: fix a set of functions $\xi_i(\boldsymbol{\theta}) \quad i = 1, \ldots, N$ (basis functions)
- Take a linear combination f of the basis function with random coefficients w_i, $f = \sum_i w_i \xi_i(\boldsymbol{\theta})$
- If the coefficients are (multivariate) Gaussian distributed, the value of f at a point $\hat{\boldsymbol{\theta}}$ is Gaussian distributed

Important exercise

Let $\varphi_1(x), \ldots, \varphi_N(x)$ be a fixed set of functions, and let $f(x) = \sum w_i \varphi_i(x)$. If $\mathbf{w} \sim \mathcal{N}(0, I)$, compute:

1. The single-point marginal distribution of $f(x)$
2. The two-point marginal distribution of $f(x_1), f(x_2)$

Important exercise

Let $\varphi_1(x), \ldots, \varphi_N(x)$ be a fixed set of functions, and let $f(x) = \sum w_i \varphi_i(x)$. If $\mathbf{w} \sim \mathcal{N}(0, I)$, compute:

1. The single-point marginal distribution of $f(x)$
2. The two-point marginal distribution of $f(x_1), f(x_2)$

- Obviously both distributions are zero-mean Gaussians
- To compute the (co)variance, take products and expectations and remember that $\langle w_i w_j \rangle = \delta_{ij}$
- Defining $\boldsymbol{\varphi}(x) = (\varphi_1(x), \ldots, \varphi_N(x))$, we get that

$$\langle f(x_i) f(x_j) \rangle = \boldsymbol{\varphi}(x_i)^T \boldsymbol{\varphi}(x_j)$$

The Gram matrix

- Generalising the exercise to more than two points, we get that *any* finite dimensional marginal of this process is multivariate Gaussian
- The covariance matrix of this function is given by evaluating a function of two variables at all possible pairs
- The function is defined by the set of basis functions

$$k(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)$$

- The covariance matrix is often called *Gram matrix* and is (necessarily) symmetric and positive definite
- Bayesian prediction in regression then is essentially the same as computing conditionals for Gaussians (more later)

Main limitation of basis function regression

- Choice of basis functions inevitably impacts what can be predicted
- Suppose one wishes the basis functions to tend to zero as $x \to \infty$. What would we predict for very large input values?

Main limitation of basis function regression

- Choice of basis functions inevitably impacts what can be predicted
- Suppose one wishes the basis functions to tend to zero as $x \to \infty$. What would we predict for very large input values?
- Very large input values will have predicted outputs near zero with high confidence!
- Ideally, one would want a prior over functions which would have the same uncertainty everywhere

Stationary variance

- We have seen that the variance of a random combination of functions depends on space as $\sum \varphi_i^2(x)$
- Given any compact set, (e.g. hypercube with centre in the origin), we can find a finite set of basis functions s.t. $\sum \varphi_i^2(x) = const$ (partition of unity, e.g. triangulations or smoother alternatives)
- We can construct a sequence of such sets which covers the whole of $\mathbb{R}^D$ in the limit
- Therefore, we can construct a sequence of priors which all have constant prior variance across all space
- Covariances would still be computed by evaluating a Gram matrix (and need not be constant)

Function space view

- The argument before shows that we can put a prior over infinite-dimensional spaces of functions s.t. all finite dimensional marginals are multivariate Gaussian
- The constructive argument, often referred to as *weights space view*, is useful for intuition but impractical
- It does demonstrate the existence of truly infinite dimensional Gaussian processes
- Once we accept that Gaussian processes exist, we are better off proceeding along a more abstract line

Gaussian Processes

Gaussian Processes are a natural extension of the multivariate normal distribution to infinite dimensional spaces of functions.

Gaussian Processes

Gaussian Processes are a natural extension of the multivariate normal distribution to infinite dimensional spaces of functions.

A GP is a probability measure over the space of continuous functions (over a suitable input space) such that the random vector obtained by evaluating a sample function at a finite set of points follows a multivariate normal distribution.

Gaussian Processes

Gaussian Processes are a natural extension of the multivariate normal distribution to infinite dimensional spaces of functions.

A GP is a probability measure over the space of continuous functions (over a suitable input space) such that the random vector obtained by evaluating a sample function at a finite set of points follows a multivariate normal distribution.

A GP is uniquely defined by its mean and covariance functions, denoted by $\mu(x)$ and $k(x, x')$:

$$f \sim \mathcal{GP}(\mu, k) \leftrightarrow \mathbf{f} = (f(x_1), \ldots, f(x_N)) \sim \mathcal{N}(\boldsymbol{\mu}, K),$$

$$\boldsymbol{\mu} = (\mu(x_1), \ldots .\mu(x_N)), \quad K = (k(x_i, x_j))_{i,j}$$

The Radial Basis Function Kernel

The kernel function is the most important ingredient of GP (the mean function is usually taken to be zero).

The Radial Basis Function Kernel

The kernel function is the most important ingredient of GP (the mean function is usually taken to be zero).

Radial Basis Function kernel

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

It depends on two hyper-parameters, the amplitude γ and the lengthscale λ. Sample functions from a GP with RBF covariance are with probability 1 infinitely differentiable functions.

Fact

Any smooth function can be approximated to arbitrary precision by a sample from a GP with RBF covariance

Bayesian prediction with GPs

- The joint probability of function values at a set of points is multivariate Gaussian
- If I have observations $y_i \quad i = 1, \ldots, N$ of the function at inputs x_i, what can I say of the function value at a new point x_*?
- By Bayes' theorem, we have

$$p(f_*|\mathbf{y}) \propto \int df(\mathbf{x}) p(f_*, f(\mathbf{x})) p(\mathbf{y}|f(\mathbf{x})) \tag{1}$$

  where $f(\mathbf{x})$ is the vector of function values at the input points
- If $p(\mathbf{y}|f(\mathbf{x}))$ is Gaussian, then we have a regression task and the integral in (1) can be computed analytically

Regression calculations

- By the definition of GPs we know that the joint $p(f_{new}, f(\mathbf{x}))$ in equation (1) is a zero-mean multivariate Gaussian with covariance

$$\Sigma = \begin{pmatrix} k_{**} & \mathbf{k}_* \\ \mathbf{k}_* & K \end{pmatrix} \tag{2}$$

where $k_{**} = k(x_*, x_*)$, $\mathbf{k}_{*j} = k(x_*, x_j)$ and $K_{ij} = k(x_i, x_j)$

Regression calculations

- By the definition of GPs we know that the joint $p(f_{new}, f(\mathbf{x}))$ in equation (1) is a zero-mean multivariate Gaussian with covariance

$$\Sigma = \begin{pmatrix} k_{**} & \mathbf{k}_* \\ \mathbf{k}_* & K \end{pmatrix} \tag{2}$$

  where $k_{**} = k(x_*, x_*)$, $\mathbf{k}_{*j} = k(x_*, x_j)$ and $K_{ij} = k(x_i, x_j)$
- Supposing the observations are i.i.d. Gaussian $p(\mathbf{y}|f(\mathbf{x})) = \mathcal{N}(0, \sigma^2 I)$ we can obtain the joint distribution for the observations and the new value

$$p(f_*, \mathbf{y}) = \mathcal{N}(0, \Sigma_{\mathbf{y}}) \quad \Sigma_{\mathbf{y}} = \begin{pmatrix} k_{**} & \mathbf{k}_* \\ \mathbf{k}_* & K + \sigma^2 I \end{pmatrix} \tag{3}$$

Regression calcs cont'd

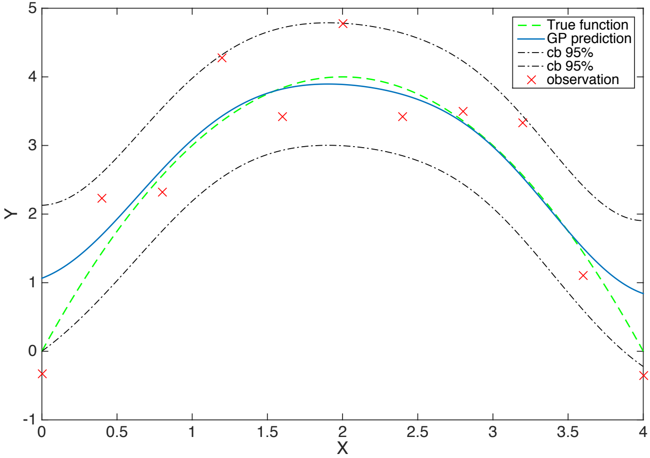
- Conditioning on observations means fixing the values of the variables **y** to the actual measurements in equation (3)
- The posterior distribution is then obtained by completing the square and using the partitioned inverse formula (see e.g. Rasmussen and Williams, Gaussian Processes for Machine learning, MIT press 2006)
- The final result is that $p(f_*|\mathbf{y}) = \mathcal{N}(m, v)$ with

$$m = \mathbf{k}_*^T \Sigma_{\mathbf{y}}^{-1} \mathbf{y} \qquad v = k_{**} - \mathbf{k}_*^T \Sigma_{\mathbf{y}}^{-1} \mathbf{k}_* \tag{4}$$

Observations

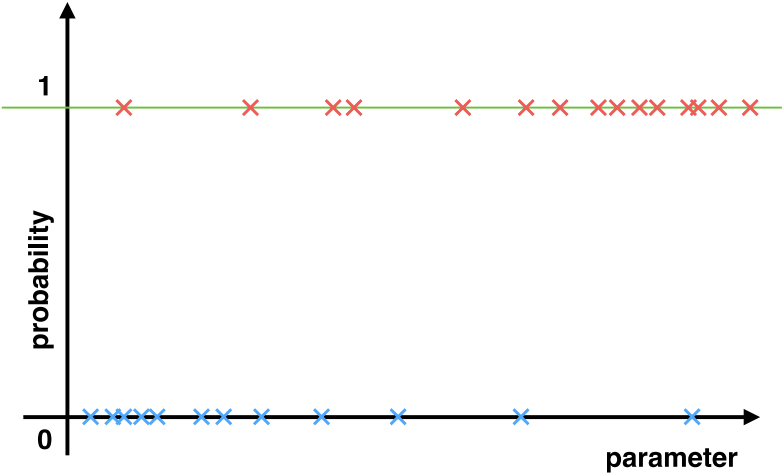
- Equations (4) provide the predictive distribution *analytically* at all points in space
- The expectation is always a linear combination of the observations; far away points contribute less to the prediction (being multiplied by the fast decaying term k_{*j}
- Addition of a new observation *always* reduces uncertainty at all points $\rightarrow$ vulnerable to outliers
- **MAIN PROBLEM**: GP prediction relies on a matrix inversion which scales cubically with the number of points!
- Sparsification methods have been proposed but in high dimension GP regression is likely to be tricky nevertheless

GP regression - example

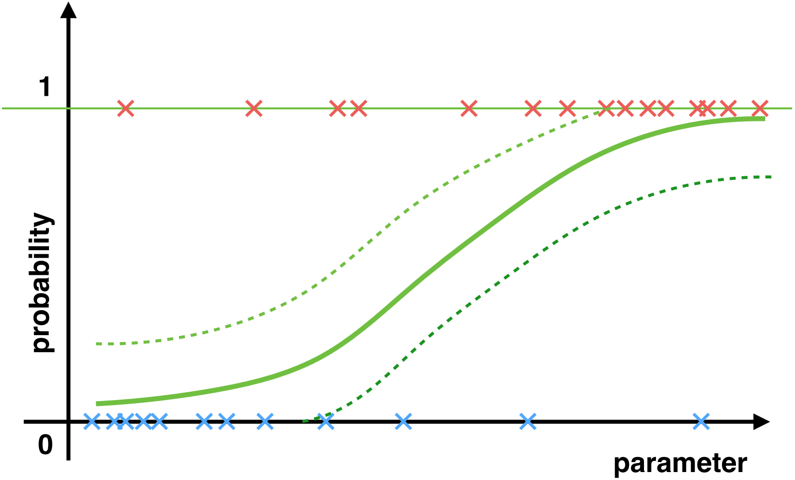


Smoothed model checking: An intuitive view

Smoothed model checking: An intuitive view



Smoothed model checking: An intuitive view



GP prediction from binary observations

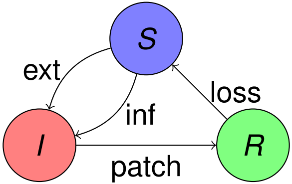
- Equation (1) holds whatever $p(\mathbf{y}|f(\mathbf{x}))$.
- In the model checking case, we have that the truth of a formula at a specified parameter value is a boolean variable following a Bernoulli distribution
- We can encode this in a GP prediction mechanism and work directly with (few) truth observations at a set of parameter values
- The integral in (1) is no longer analytically computable, we use Expectation-Propagation (EP) an established fast approximate inference method

Smoothed model checking

We call the following statistical model checking algorithm *Smoothed model checking*

1. Input a set of parameter values and a number of observations per parameter value
2. Perform (approximate) GP prediction to obtain estimates of satisfaction function and uncertainties
3. If uncertainty is too high, increase the resolution of the parameter grid/ number of observations and repeat.
4. Return estimated satisfaction function and uncertainties

Network Epidemics



We investigate a SIR-like network epidemics model, with a network of 100 nodes having three states: susceptible (X_S), infected (X_I), and patched (X_R). The dynamics is described by the following reactions:

External infection: $S \xrightarrow{k_e} I$, with rate function $k_e X_S$;

Internal infection: $S + I \xrightarrow{k_i} I + I$, with rate function $k_i X_S X_I$;

Patching: $I \xrightarrow{k_r} R$, with rate function $k_r X_I$;

Immunity loss: $R \xrightarrow{k_s} S$, with rate function $k_s X_R$;

Example: Epidemics

SIR model

We investigate the dependence of truth probability of the property

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

on the infection rate: k_I and recovery rate: k_R. We use ten simulations at each of ten parameter values.

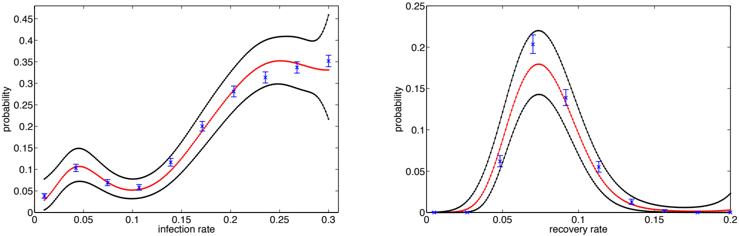
Example: Epidemics

SIR model

We investigate the dependence of truth probability of the property

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

on the infection rate: k_I and recovery rate: k_R. We use ten simulations at each of ten parameter values.



(left: function of k_I, right: function of k_R. Blue dots, SMC for 5000 runs per parameter value)

Example: Epidemics

SIR model - two free parameters

We estimate $p(\varphi \mid k_I, k_R)$ from 2560 traces for the same property.

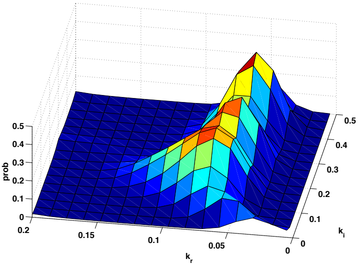
$$\varphi = (X_I > 0)\mathbf{U}^{[100,120]}(X_I = 0)$$

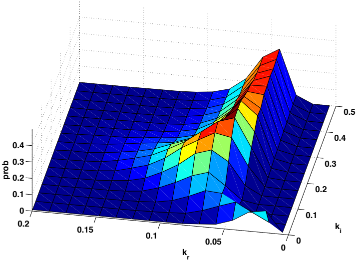
Example: Epidemics

SIR model - two free parameters

We estimate $p(\varphi \mid k_I, k_R)$ from 2560 traces for the same property.

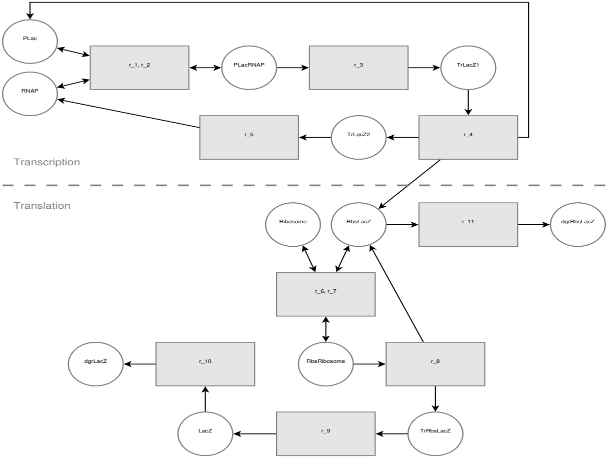
$$\varphi = (X_I > 0)\mathbf{U}^{[100,120]}(X_I = 0)$$





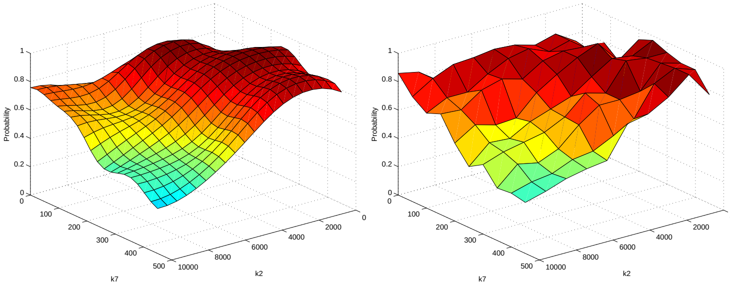
(left: smoothed MC, right SMC on 16x16 points at 5000 runs per point. Time SMC= 580× smoothed MC)

Example: LacZ operon



Stochastic model of celebrated regulatory circuit in *E. coli* anabolism (Kierzek,2002).

Example: LacZ operon



Model checking the probability of bursting expression

$$\varphi = \mathbf{F}^{[16000,21000]}\left(\Delta X_{\mathrm{LacZ}} > 0 \ \wedge \ \mathbf{G}^{[10,2000]}\Delta X_{\mathrm{LacZ}} \leq 0\right)$$

as we vary k_2 and k_7. Left smoothed MC, right SMC (100 draws per parameter value).

Quantitative evaluation

Table: 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

Outline

Robust Design with Temporal Logic

Qualitative data

We consider a set of system requirements specified as (linear) temporal logic formulae.

Robust Design with Temporal Logic

Qualitative data

We consider a set of system requirements specified as (linear) temporal logic formulae.

Problem: Robust Synthesis/ Design

Find the set of model parameters that such that the model satisfies the requirements as robustly as possible.

Robust Design with Temporal Logic

Qualitative data

We consider a set of system requirements specified as (linear) temporal logic formulae.

Problem: Robust Synthesis/ Design

Find the set of model parameters that such that the model satisfies the requirements as robustly as possible.

Notion of robustness

This requires a proper notion of robustness of satisfaction for a temporal logic formula.

Signal Temporal Logic

STL is metric linear time logic for real-valued signals.

STL Syntax

Given a (primary) real-valued signal $x[t] = (x_1[t], ..., x_n[t])$, $t \in \mathbb{R}_{\geqslant 0}$, $x_i \in \mathbb{R}$, the *STL syntax* is given by

$$\varphi := \mu \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \, \mathbf{U}^{[a,b]} \, \varphi_2$$

Signal Temporal Logic

STL is metric linear time logic for real-valued signals.

STL Syntax

Given a (primary) real-valued signal $x[t] = (x_1[t], ..., x_n[t])$, $t \in \mathbb{R}_{\geqslant 0}$, $x_i \in \mathbb{R}$, the *STL syntax* is given by

$$\varphi := \mu \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \, \mathbf{U}^{[a,b]} \, \varphi_2$$

- $\mu : \mathbb{R}^n \to \mathbb{B}$ is an atomic predicate s.t. $\mu(x) := (y(x) \geqslant 0)$,

Signal Temporal Logic

STL is metric linear time logic for real-valued signals.

STL Syntax

Given a (primary) real-valued signal $x[t] = (x_1[t], ..., x_n[t])$, $t \in \mathbb{R}_{\geqslant 0}$, $x_i \in \mathbb{R}$, the *STL syntax* is given by

$$\varphi := \mu \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \, \mathbf{U}^{[a,b]} \, \varphi_2$$

- $\mu : \mathbb{R}^n \to \mathbb{B}$ is an atomic predicate s.t. $\mu(x) := (y(x) \geqslant 0)$,
- $y : \mathbb{R}^n \to \mathbb{R}$ a real-valued function, the secondary signal.

Signal Temporal Logic

STL is metric linear time logic for real-valued signals.

STL Syntax

Given a (primary) real-valued signal $x[t] = (x_1[t], ..., x_n[t])$, $t \in \mathbb{R}_{\geqslant 0}$, $x_i \in \mathbb{R}$, the *STL syntax* is given by

$$\varphi := \mu \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \, \mathbf{U}^{[a,b]} \, \varphi_2$$

- $\mu : \mathbb{R}^n \to \mathbb{B}$ is an atomic predicate s.t. $\mu(x) := (y(x) \geqslant 0)$,
- $y : \mathbb{R}^n \to \mathbb{R}$ a real-valued function, the secondary signal.

As usual, $\mathbf{F}^{[a,b]}\varphi := \mathtt{tt}\,\mathbf{U}^{[a,b]}\varphi$ and $\mathbf{G}^{[a,b]}\varphi := \neg\mathbf{F}^{[a,b]}\neg\varphi$.

Signal Temporal Logic

STL Quantitative Semantics

The quantitative satisfaction function ρ is defined by

$$
\begin{aligned}
\rho(\mu, x, t) &= y(x[t]) \qquad \text{where } \mu \equiv (y(x[t]) \geqslant 0) \\
\rho(\neg\varphi, x, t) &= -\rho(\varphi, x, t) \\
\rho(\varphi_1 \wedge \varphi_2, x, t) &= \min(\rho(\varphi_1, x, t), \rho(\varphi_2, x, t)) \\
\rho(\varphi_1\, \mathcal{U}_{[a,b)} \varphi_2, x, t) &= \max_{t' \in t+[a,b]} (\min(\rho(\varphi_2, x, t')), \min_{t'' \in [t,t']} (\rho(\varphi_1, x, t''))).
\end{aligned}
$$

This satisfaction score can be computed efficiently for piecewise linear signals, see the Breach Matlab Toolbox

Satisfability of Stochastic Models

The boolean semantics of an STL formula φ can be easily extended to stochastic models.
as customary, by measuring the probability of the set of trajectories of the CTMC that satisfy the formula:

$$P(\varphi) = \mathbb{P}\{\mathbf{x} \mid \mathbf{x} \models \varphi\}.$$

Satisfability of Stochastic Models

The boolean semantics of an STL formula φ can be easily extended to stochastic models.
as customary, by measuring the probability of the set of trajectories of the CTMC that satisfy the formula:

$$P(\varphi) = \mathbb{P}\{\mathbf{x} \mid \mathbf{x} \models \varphi\}.$$

- The stochastic process $\mathbf{X}(t)$ is interpreted as a random variable $\mathbf{X}$ over the space of trajectories $\mathcal{D}$.

Satisfability of Stochastic Models

The boolean semantics of an STL formula φ can be easily extended to stochastic models.
as customary, by measuring the probability of the set of trajectories of the CTMC that satisfy the formula:

$$P(\varphi) = \mathbb{P}\{\mathbf{x} \mid \mathbf{x} \models \varphi\}.$$

- The stochastic process $\mathbf{X}(t)$ is interpreted as a random variable $\mathbf{X}$ over the space of trajectories $\mathcal{D}$.
- The set of trajectories that satisfy an STL formula φ can be thought as a measurable function

$$I_\varphi : \mathcal{D} \to \{0, 1\}, \text{ s.t. } I_\varphi(\mathbf{x}) = 1 \quad \Leftrightarrow \quad \mathbf{x} \models \varphi.$$

Satisfability of Stochastic Models

The boolean semantics of an STL formula φ can be easily extended to stochastic models.
as customary, by measuring the probability of the set of trajectories of the CTMC that satisfy the formula:

$$P(\varphi) = \mathbb{P}\{\mathbf{x} \mid \mathbf{x} \models \varphi\}.$$

- The stochastic process $\mathbf{X}(t)$ is interpreted as a random variable $\mathbf{X}$ over the space of trajectories $\mathcal{D}$.
- The set of trajectories that satisfy an STL formula φ can be thought as a measurable function

$$I_\varphi : \mathcal{D} \to \{0,1\}, \text{ s.t. } I_\varphi(\mathbf{x}) = 1 \quad \Leftrightarrow \quad \mathbf{x} \models \varphi.$$

Hence,

$$\mathbb{P}(I_\varphi(\mathbf{X}) = 1) = \mathbb{P}(\{\mathbf{x} \in \mathcal{D} \mid I_\varphi(\mathbf{x}) = 1\}) = \mathbb{P}\{\mathbf{x} \mid \mathbf{x} \models \varphi\} = P(\varphi)$$

$$\mathbb{P}(I_\varphi(\mathbf{X}) = 1) = \mathbb{P}(\{\mathbf{x} \in \mathcal{D} \mid I_\varphi(\mathbf{x}) = 1\}) = \mathbb{P}(I_\varphi^{-1}(1))$$

Robustness of Stochastic Models

The quantitative satisfaction function

$$\rho(\varphi, \mathbf{x}) : \mathcal{D} \to \mathbb{R}$$

(with respect to the σ-algebra induced from the Skorokhod topology in $\mathcal{D}$),

induces a real-valued random variable $R_\varphi(\mathbf{X})$ with probability distribution

$$\mathbb{P}\Big(R_\varphi(\mathbf{X}) \in [a,b]\Big) = \mathbb{P}\left(\mathbf{X} \in \{\mathbf{x} \in \mathcal{D} \mid \rho(\varphi, \mathbf{x}, 0) \in [a,b]\}\right)$$

Robustness of Stochastic Models

The quantitative satisfaction function

$$\rho(\varphi, \mathbf{x}) : \mathcal{D} \to \mathbb{R}$$

(with respect to the σ-algebra induced from the Skorokhod topology in $\mathcal{D}$),

induces a real-valued random variable $R_\varphi(\mathbf{X})$ with probability distribution

$$\mathbb{P}\Big(R_\varphi(\mathbf{X}) \in [a, b]\Big) = \mathbb{P}\left(\mathbf{X} \in \{\mathbf{x} \in \mathcal{D} \mid \rho(\varphi, \mathbf{x}, 0) \in [a, b]\}\right)$$

Indicators:

- $\mathbb{E}(R_\varphi)$ (The average robustness degree)
- $\mathbb{E}(R_\varphi \mid R_\varphi > 0)$ and $\mathbb{E}(R_\varphi \mid R_\varphi < 0)$ (The conditional average)

Schlögl system

CTMC model of a Schlögl system.

Biochemical reactions of the Schlögl model:

Reaction	rate constant	init pop
$A + 2X \rightarrow 3X$	$k_1 = 3 \cdot 10^{-7}$	$X(0) = 247$
$3X \rightarrow A + 2X$	$k_2 = 1 \cdot 10^{-4}$	$A(0) = 10^5$
$B \rightarrow X$	$k_3 = 1 \cdot 10^{-3}$	$B(0) = 2 \cdot 10^5$
$X \rightarrow B$	$k_4 = 3.5$	

Schlögl system

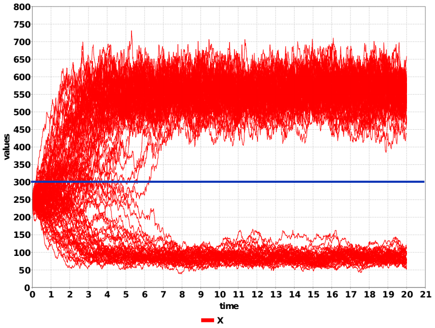
CTMC model of a Schlögl system.

Biochemical reactions of the Schlögl model:

Reaction	rate constant	init pop
$A + 2X \rightarrow 3X$	$k_1 = 3 \cdot 10^{-7}$	$X(0) = 247$
$3X \rightarrow A + 2X$	$k_2 = 1 \cdot 10^{-4}$	$A(0) = 10^5$
$B \rightarrow X$	$k_3 = 1 \cdot 10^{-3}$	$B(0) = 2 \cdot 10^5$
$X \rightarrow B$	$k_4 = 3.5$	

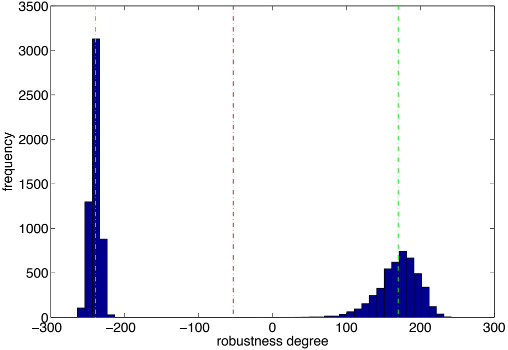
Simulation of the Schlogl model (100 runs):

starting close to the boundary of the basin of attraction, the bistable behaviour is evident



Schlögl system

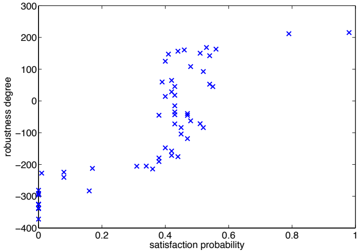
STL formula : $\varphi : \mathbf{F}^{[0,T_1]}\mathbf{G}^{[0,T_2]}(X \geq k_t) \quad k_t = 300$



Statistical estimation of φ: $p = 0.4583$ (10000 runs, error ± 0.02 at 95% confidence level).

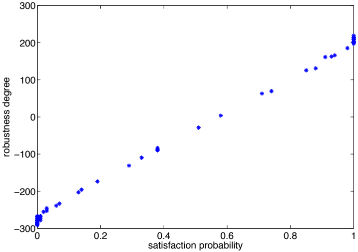
Schlögl system

Satisfaction probability versus average robustness degree



Varying threshold k_t

Correlation around 0.8386, dependency seems to follow a sigmoid shaped curve.



Varying k_3

Correlation around 0.9718 with an evident linear trend.

The System Design Problem

"Given a stochastic model, depending on a set of parameters $\theta \in \boldsymbol{K}$, and a specification φ (STL formula), find the parameter combination θ^* s.t. the system satisfies φ as robustly as possible".

The System Design Problem

"Given a stochastic model, depending on a set of parameters $\boldsymbol{\theta} \in \boldsymbol{K}$, and a specification φ (STL formula), find the parameter combination $\boldsymbol{\theta}^*$ s.t. the system satisfies φ as robustly as possible".

Solution strategy:

- **rephrase** it as an unconstrained optimisation problem: we maximise the expected robustness degree.

The System Design Problem

"Given a stochastic model, depending on a set of parameters $\boldsymbol{\theta} \in \boldsymbol{K}$, and a specification φ (STL formula), find the parameter combination $\boldsymbol{\theta}^*$ s.t. the system satisfies φ as robustly as possible".

Solution strategy:

- **rephrase** it as an unconstrained optimisation problem: we maximise the expected robustness degree.
- **evaluate** the function to optimise using statistical model checking with a fixed number of runs

The System Design Problem

"Given a stochastic model, depending on a set of parameters $\boldsymbol{\theta} \in \boldsymbol{K}$, and a specification φ (STL formula), find the parameter combination $\boldsymbol{\theta}^*$ s.t. the system satisfies φ as robustly as possible".

Solution strategy:

- **rephrase** it as an unconstrained optimisation problem: we maximise the expected robustness degree.
- **evaluate** the function to optimise using statistical model checking with a fixed number of runs
- **solve** the optimisation problem using the Gaussian Process-Upper Confidence Bound optimisation (GP-UCB)

The optimisation problem

- We need to maximise the expected robustness.

The optimisation problem

- We need to maximise the expected robustness.
- Each evaluation of this function is costly (we obtain it by SMC, need to run SSA many times).

The optimisation problem

- We need to maximise the expected robustness.
- Each evaluation of this function is costly (we obtain it by SMC, need to run SSA many times).
- Each evaluation of this function is noisy (we estimate the value by SMC - noise is approximatively gaussian).

The optimisation problem

- We need to maximise the expected robustness.
- Each evaluation of this function is costly (we obtain it by SMC, need to run SSA many times).
- Each evaluation of this function is noisy (we estimate the value by SMC - noise is approximatively gaussian).

We need to maximise an unknown function, which we can observe with noise, with the minimal number of evaluations.

The optimisation problem

- We need to maximise the expected robustness.
- Each evaluation of this function is costly (we obtain it by SMC, need to run SSA many times).
- Each evaluation of this function is noisy (we estimate the value by SMC - noise is approximatively gaussian).

We need to maximise an unknown function, which we can observe with noise, with the minimal number of evaluations.

We use the GP-UCB algorithm (Srinivas et al 2012)

The GP-UCB algorithm

Basic Idea

Use GP regression to emulate the unknown function, and to explore the region near the maximum of the posterior mean. Doing this naively $\Rightarrow$ trapped in local optima.

The GP-UCB algorithm

Basic Idea

Use GP regression to emulate the unknown function, and to explore the region near the maximum of the posterior mean. Doing this naively $\Rightarrow$ trapped in local optima.

Balance Exploration and Exploitation

Maximise an upper quantile of the distribution, obtained as mean value plus a constant times the standard deviation:

$$x_{t+1} = \mathrm{argmax}_x \left[\mu_t(x) + \beta_t var_t(x)\right]$$

Then x_t is added to the observation points.

The GP-UCB algorithm

Basic Idea

Use GP regression to emulate the unknown function, and to explore the region near the maximum of the posterior mean. Doing this naively ⇒ trapped in local optima.

Balance Exploration and Exploitation

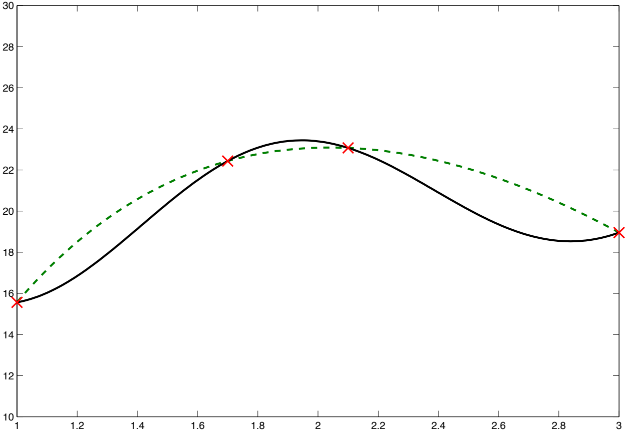
Maximise an upper quantile of the distribution, obtained as mean value plus a constant times the standard deviation:

$$x_{t+1} = \mathrm{argmax}_x \left[\mu_t(x) + \beta_t var_t(x)\right]$$

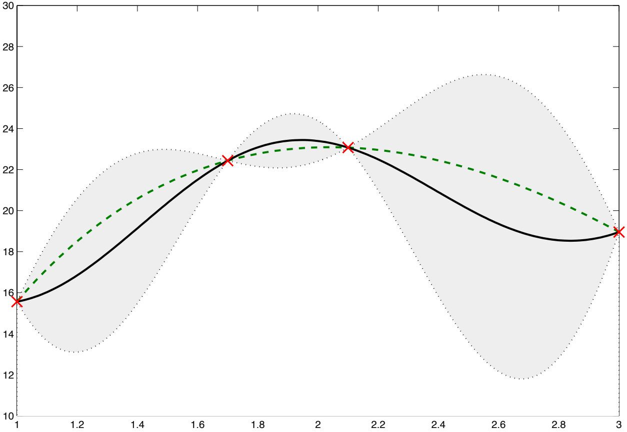
Then x_t is added to the observation points.

The algorithm has a convergence guarantee in terms of regret bounds (for slowly increasing β_t).

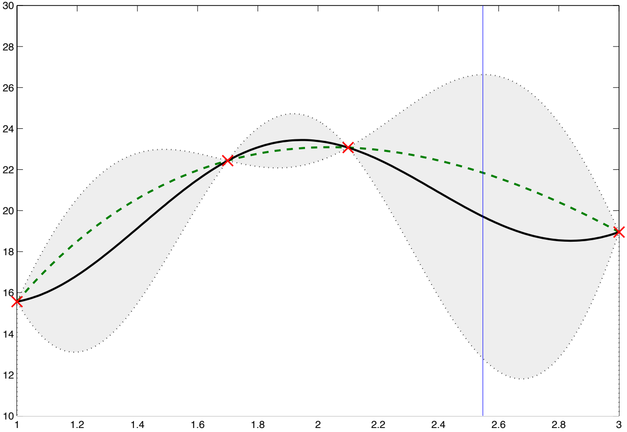
The GP-UCB algorithm - example



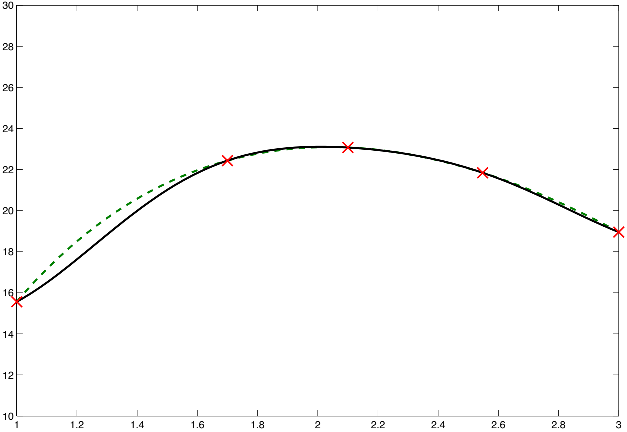
The GP-UCB algorithm - example



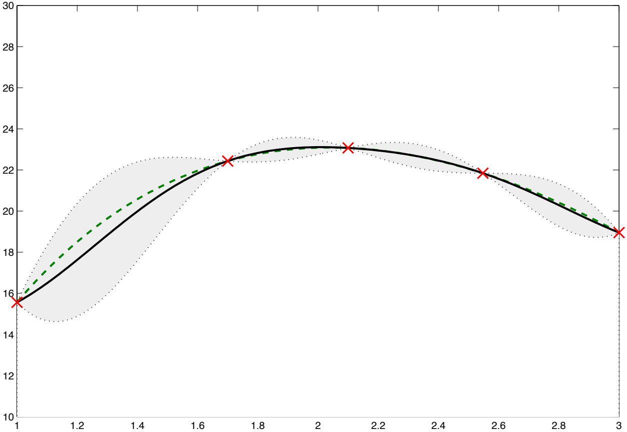
The GP-UCB algorithm - example



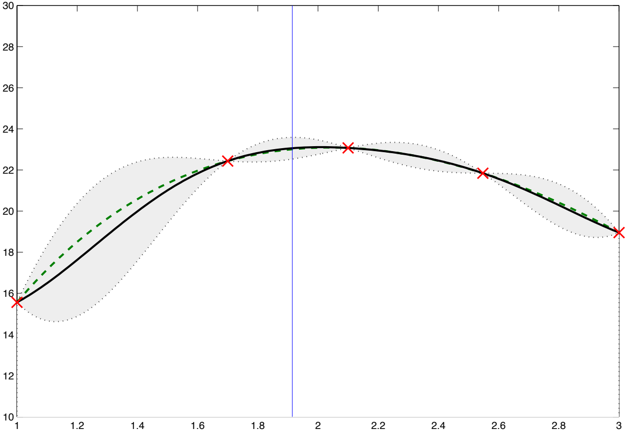
The GP-UCB algorithm - example



The GP-UCB algorithm - example



The GP-UCB algorithm - example

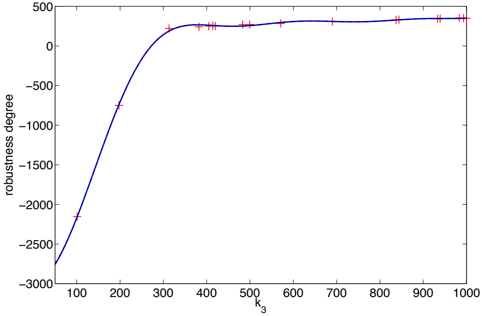


Experimental Results (Schlögl System)

Statistics of the results of ten experiments to optimize the parameter k_3 , for $\varphi : \mathbf{F}^{[0,T_1]}\mathbf{G}^{[0,T_2]}(X \geq k_t)$, in the range $[50, 1000]$:

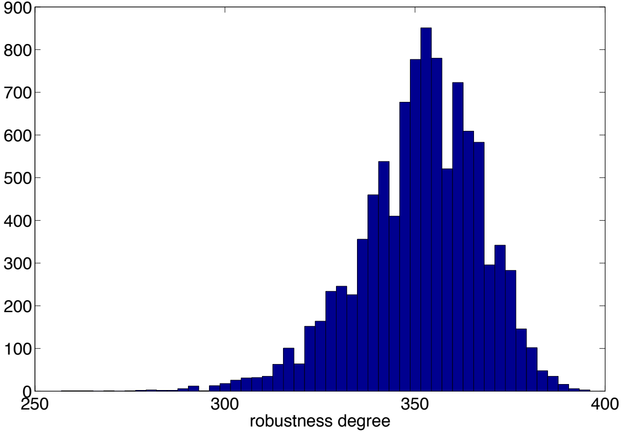
Parameter mean	Parameter range	Mean probability
$k_3 = 997.78$	[979.31 999.99]	1
Average Robustness	Number of function evaluations	Number of simulation runs
348.97	34.4	3440

The emulated robustness function in the optimisation of k3



Experimental Results (Schlögl System)

The distribution of the robustness score for $\varphi : \mathbf{F}^{[0,T_1]}\mathbf{G}^{[0,T_2]}(X \geq k_t)$ with k3 = 999.99, $T_1 = 10$, $T_2 = 15$ and $k_t = 300$



U-check

- Open-source java implementation of smoothed model checking and learning/ designing from qualitative data
- Takes as input models specified as PRISM, BioPEPA or SimHyA
- Outputs results readable/ analysable with MATLAB or GNUplot
- Java source code available at `https://github.com/dmilios/U-check`

Other related things

- Learning parametrisations from formulae truth observations (QEST 2013)
- Learning parametric formulae from time series data (with E. Bartocci, FORMATS 2014)
- Spatio-temporal generalisations of robust system design (with E. Bartocci and L. Nenzi, HSB 2015)
- Statistical abstractions of stiff dynamical systems for efficient simulations (CMSB 2015)

Conclusions

- Uncertainty has implications also for formal analysis techniques

Conclusions

- Uncertainty has implications also for formal analysis techniques
- When there is uncertainty, machine learning is likely to be of use

Conclusions

- Uncertainty has implications also for formal analysis techniques
- When there is uncertainty, machine learning is likely to be of use

Conclusions

- Uncertainty has implications also for formal analysis techniques
- When there is uncertainty, machine learning is likely to be of use
- We proposed methods relying on Gaussian Processes to efficiently compute the probability of a certain property as a function of the parameters, and to train models from qualitative observations.

Conclusions

- Uncertainty has implications also for formal analysis techniques
- When there is uncertainty, machine learning is likely to be of use
- We proposed methods relying on Gaussian Processes to efficiently compute the probability of a certain property as a function of the parameters, and to train models from qualitative observations.
- Similar ideas can be applied to other performance metrics, such as expected rewards.

Conclusions

- Uncertainty has implications also for formal analysis techniques
- When there is uncertainty, machine learning is likely to be of use
- We proposed methods relying on Gaussian Processes to efficiently compute the probability of a certain property as a function of the parameters, and to train models from qualitative observations.
- Similar ideas can be applied to other performance metrics, such as expected rewards.
- We have just released a tool: U-check.

The End

and thanks to the ERC!

Bibliography

- L. Bortolussi, D. Milios and G. S. Smoothed Model Checking for Uncertain Continuous Time Markov Chains. `http://arxiv.org/pdf/1402.1450.pdf`, to appear in Information and Computation.
- E. Bartocci, L. Bortolussi, L. Nenzi, and G. S. On the robustness of temporal properties for stochastic models. *Electronic Proceedings in Theoretical Computer Science*, 125:3–19, Aug. 2013.
- E. Bartocci, L. Bortolussi, L. Nenzi and G. S., System design of stochastic models using robustness of temporal properties, Theor. Comp. Sci. 587,3-25.
- L. Bortolussi and G. Sanguinetti. Learning and designing stochastic processes from logical constraints. In *Proceedings of QEST 2013*, 2013 and LMCS 2015.
- L. Bortolussi, D. Milios and G.S., U-check: Model Checking and Parameter Synthesis under Uncertainty, QEST 2015
- C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for machine learning*. MIT Press, Cambridge, Mass., 2006.
- N. Srinivas, A. Krause, S. M. Kakade, and M. W. Seeger. Information-theoretic regret bounds for gaussian process optimization in the bandit setting. *IEEE Transactions on Information Theory*, 58(5):3250–3265, May 2012.
- A. Donzé and O. Maler. Robust satisfaction of temporal logic over real-valued signals. In *Proc. of FORMATS*, 2010.