

Cyber-Physical Systems

Laura Nenzi

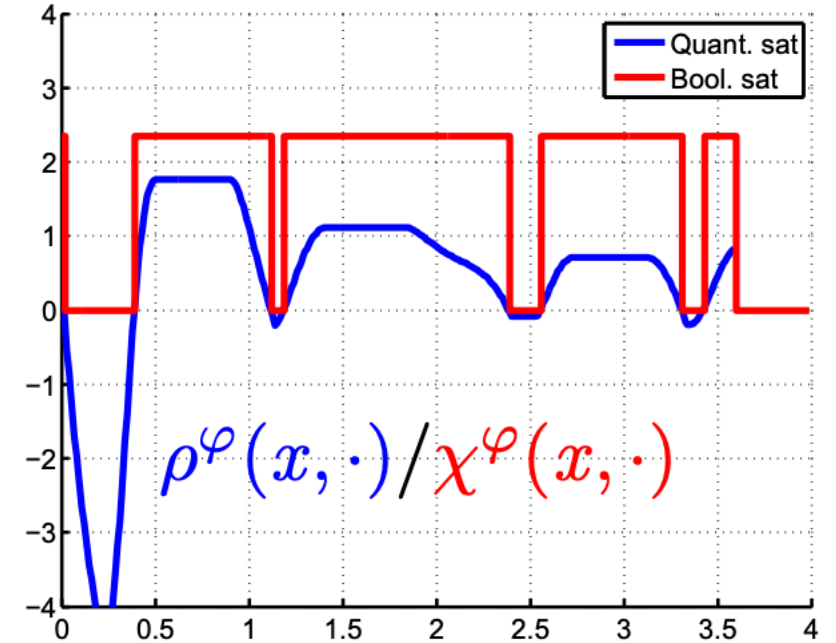
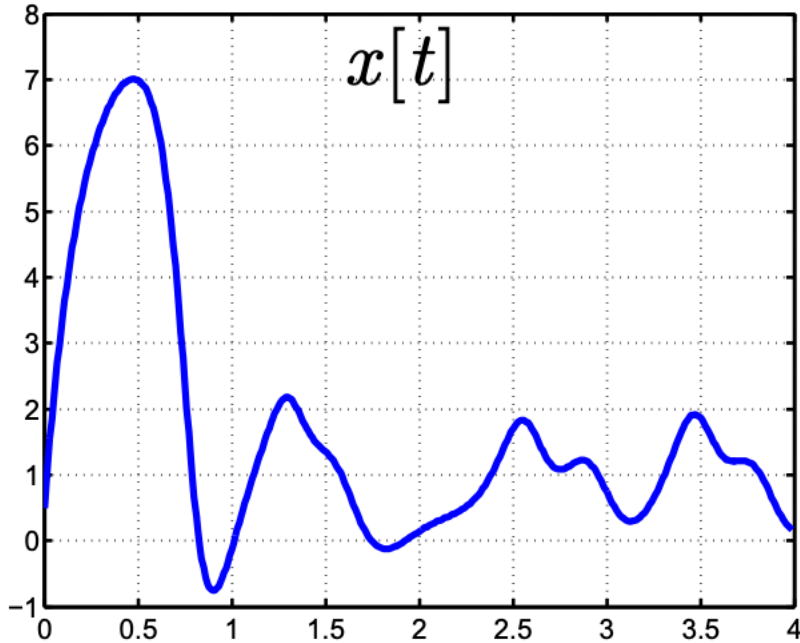
Università degli Studi di Trieste
II Semestre 2019

Lecture 10: STL applications

Terminology

- **Syntax:** A set of syntactic rules that allow us to construct formulas from specific ground terms
- **Semantics:** A set of rules that assign meanings to well-formed formulas obtained by using above syntactic rules
- **Model-checking/Verification:** $M \models \phi \iff \forall \mathbf{x} \in trace(M) \ \beta(\phi, \mathbf{x}, 0) = 1$
- **Monitoring:** computing β for a single trace $\mathbf{x} \in trace(M)$
- **Statistical Model Checking:** “doing statistics” on $\beta(\phi, \mathbf{x}, 0)$ for a finite-subset of $trace(M)$

STL Monitor



An STL monitor is a transducer that transforms x into Boolean or a quantitative signal

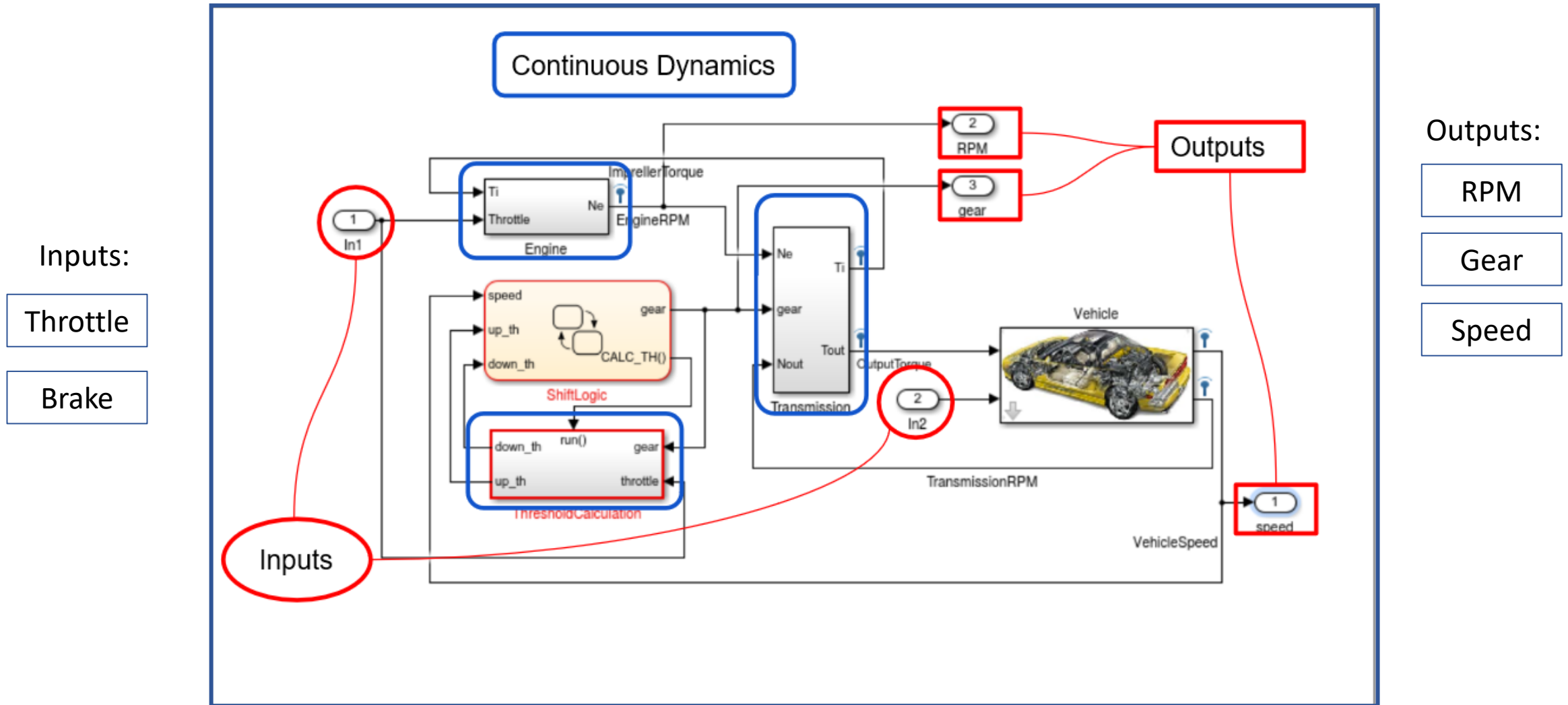
The many uses of STL

- ▶ Requirement-based testing for closed-loop control models
- ▶ Falsification Analysis
- ▶ Parameter Synthesis
- ▶ Mining Specifications/Requirements from Models
- ▶ Online Monitoring
- ▶ ...

Closed-loop Models

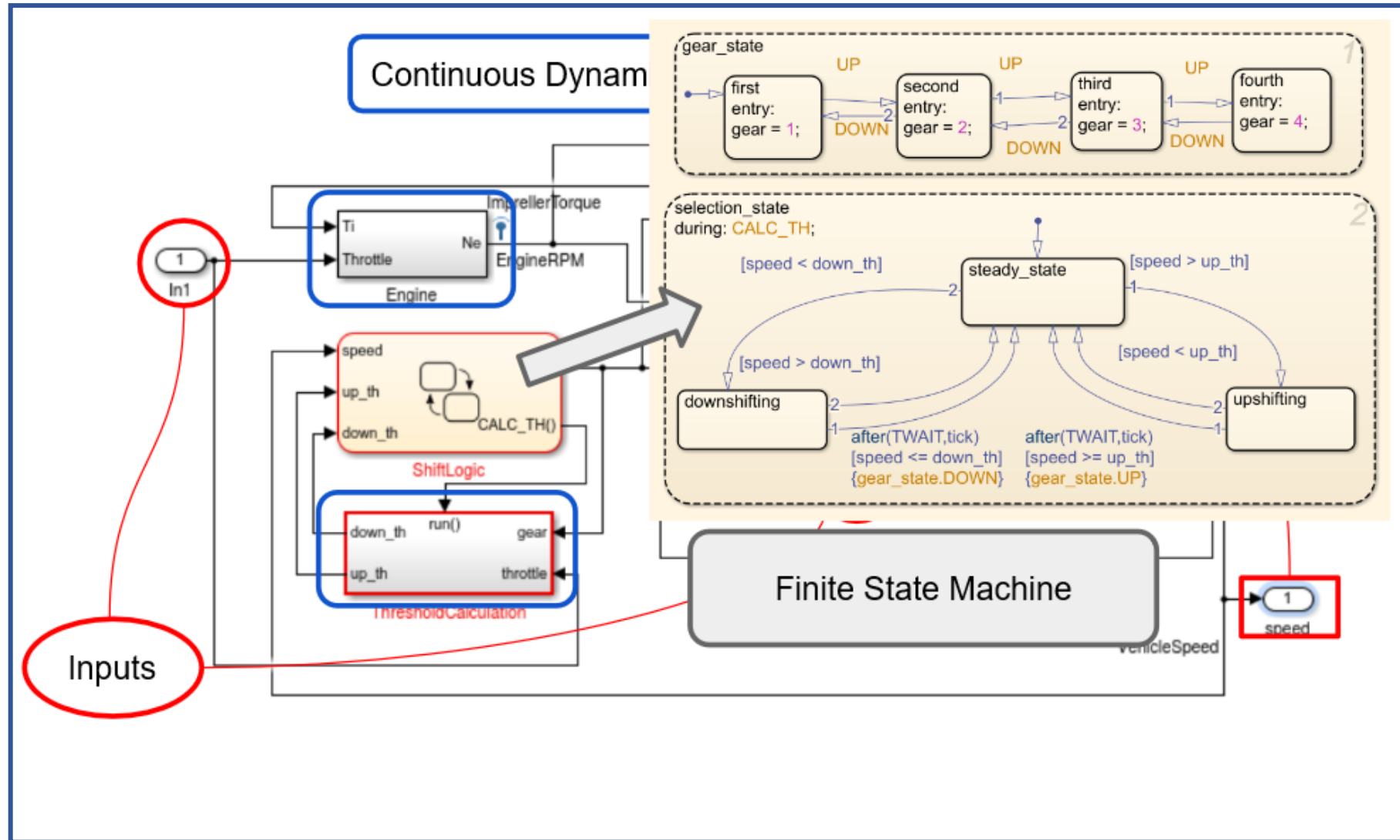
- ▶ Closed-loop Models contain:
 - ▶ Dynamics describing Physical Processes (Plant)
 - ▶ Code describing Embedded Control, Sensing, Actuation
 - ▶ Models of connection between plant and controller (hard-wired vs. wired network vs. wireless communication)

Example

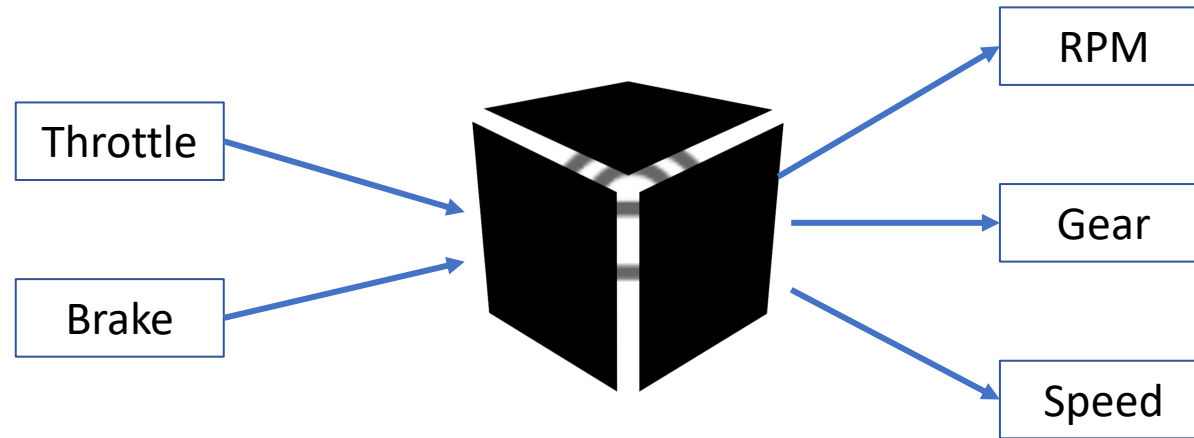


Simulink model of a Car Automatic Gear Transmission Systems

Example

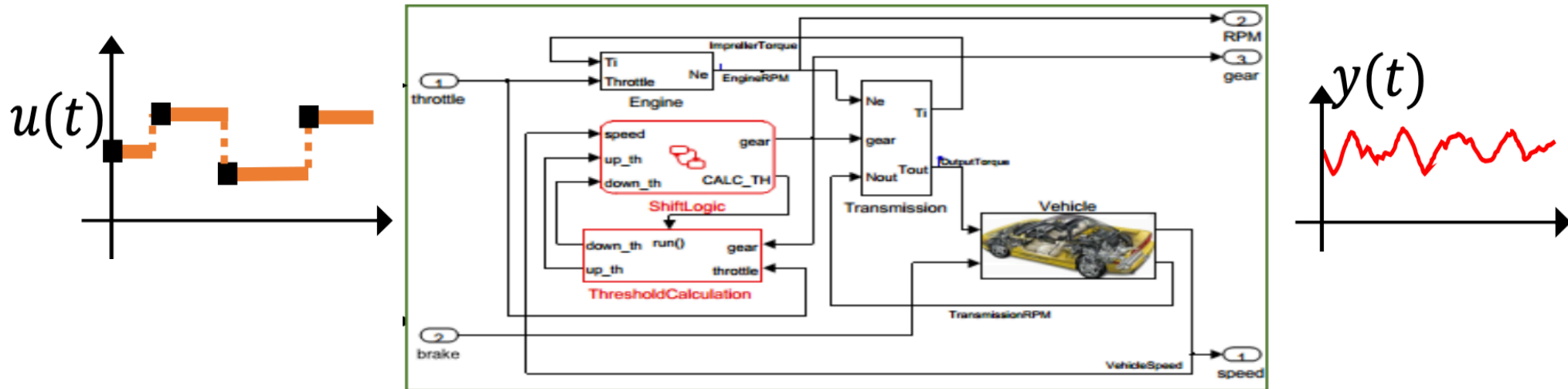


Black Box Assumption



Black Box Assumption

- For simplicity, consider the composed plant model, controller and communication to be a model M that is excited by an input signal $\mathbf{u}(t)$ and produces some output signal $\mathbf{y}(t)$



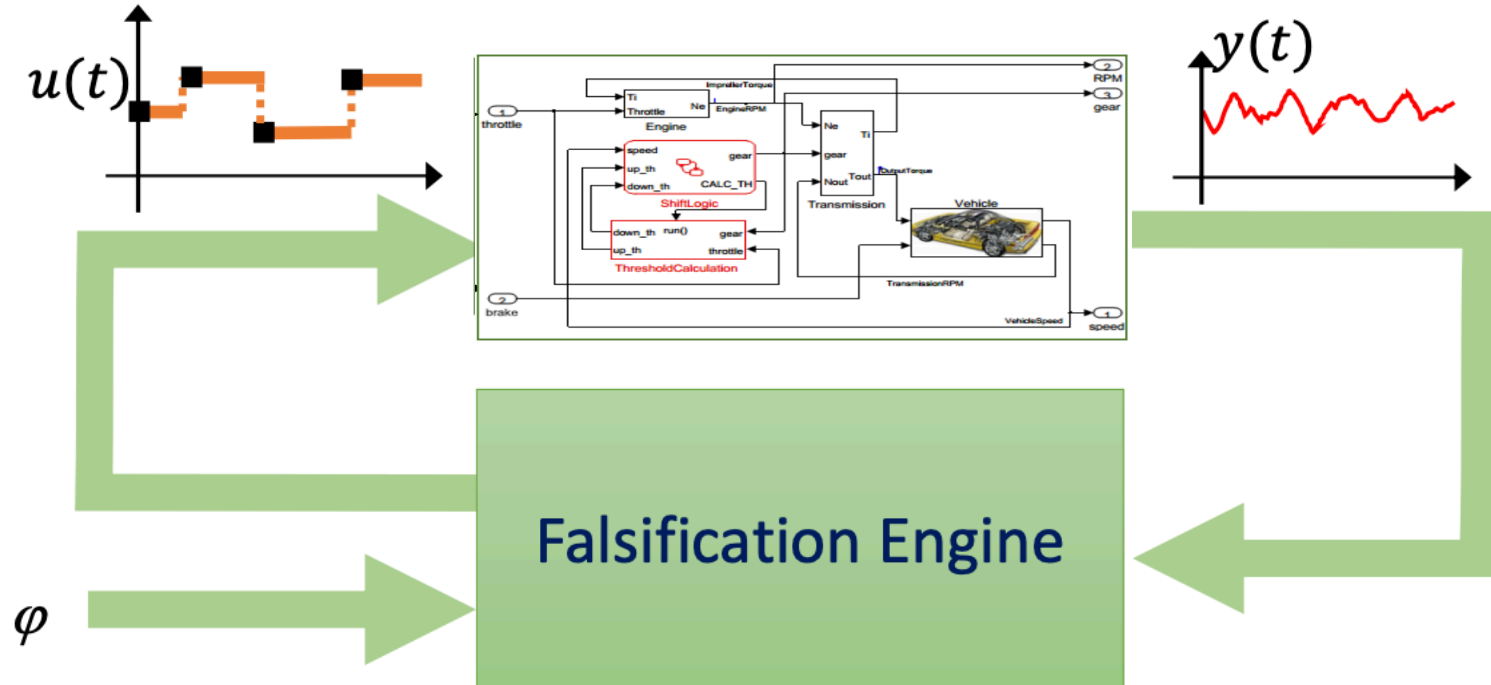
Verification vs. Testing

- ▶ For simplicity, \mathbf{u} is a function from \mathbb{T} to \mathbb{R}^m ; let the set of all possible functions representing input signals be U
- ▶ Verification Problem:
Prove the following: $\forall \mathbf{u} \in U: (\mathbf{y} = M(\mathbf{u})) \models \varphi(\mathbf{u}, \mathbf{y})$
- ▶ Falsification/Testing Problem:
Find a witness to the query: $\exists \mathbf{u} \in U : (\mathbf{y} = M(\mathbf{u})) \not\models \varphi(\mathbf{u}, \mathbf{y})$
- ▶ These formulations are quite general, as we can include the following “*model uncertainties*” as input signals: Initial states, tunable parameters in both plant and controller, time-varying parameter values, noise, etc.,

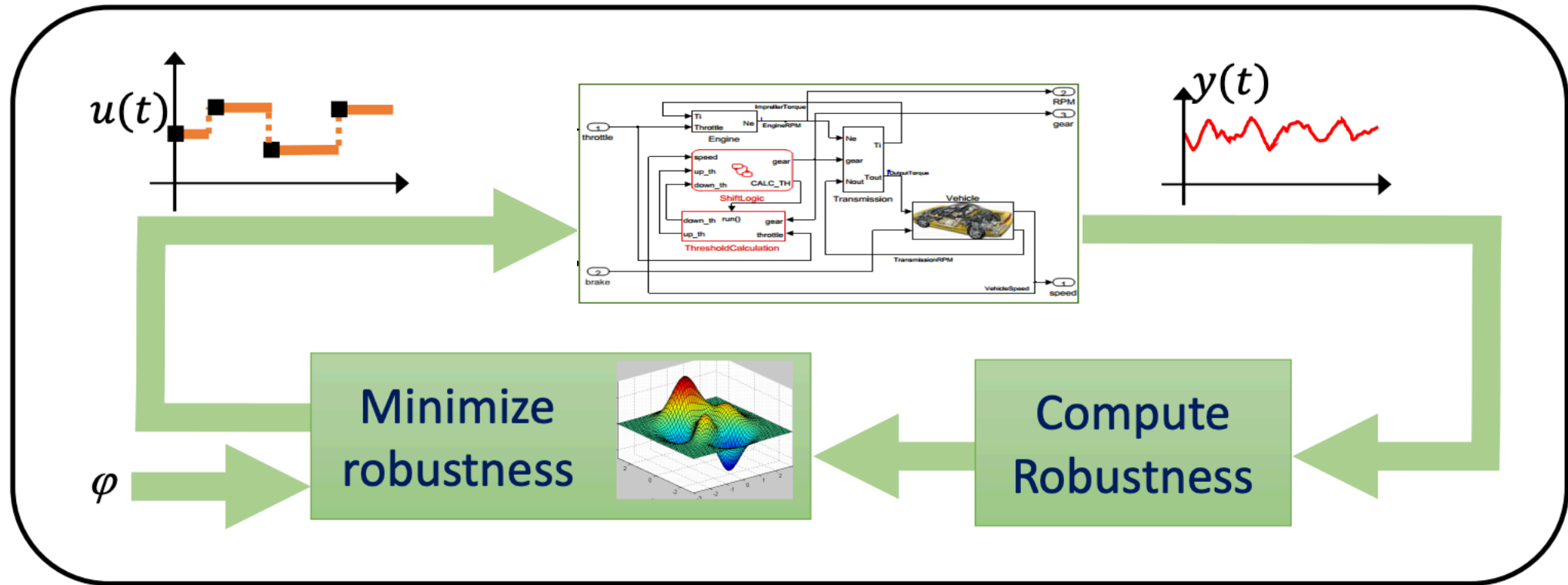
Challenges with real-world systems

- ▶ If plant model, software and communication is simple (e.g. linear models), then we can do formal analysis
- ▶ Most real-world examples have very complex plants, controllers and communication!
- ▶ Verification problem, in the most general case is ***undecidable***
 - ▶ it is proved to be impossible to construct an algorithm that always leads to a correct yes-or-no answer to the problem

Falsification/Testing



Falsification by optimization



Use robustness as a cost function to minimize with Black-box/Global Optimizers

Falsification/Testing

- ▶ Falsification or testing attempts to find one or more \mathbf{u} signals such that $\neg\varphi(\mathbf{u}, M(\mathbf{u}))$ is true.
- ▶ In verification, the set \mathbb{T} (the time domain) could be unbounded, in falsification or testing, the time domain is necessarily bounded, i.e. $\mathbb{T} \subseteq [0, T]$, where T is some finite numeric constant
- ▶ In verification the co-domain of \mathbf{u} , could be an unbounded subset of \mathbb{R}^m , in falsification, we typically consider some compact subset of \mathbb{R}^m
- ▶ For the i^{th} input signal component, let D_i denote its compact co-domain. Then the input signal \mathbf{u} is a function from \mathbb{T} to $D_1 \times \cdots \times D_m$, where $\mathbb{T} \subseteq [0, T]$
In simple words: input signals range over bounded intervals and over a bounded time horizon

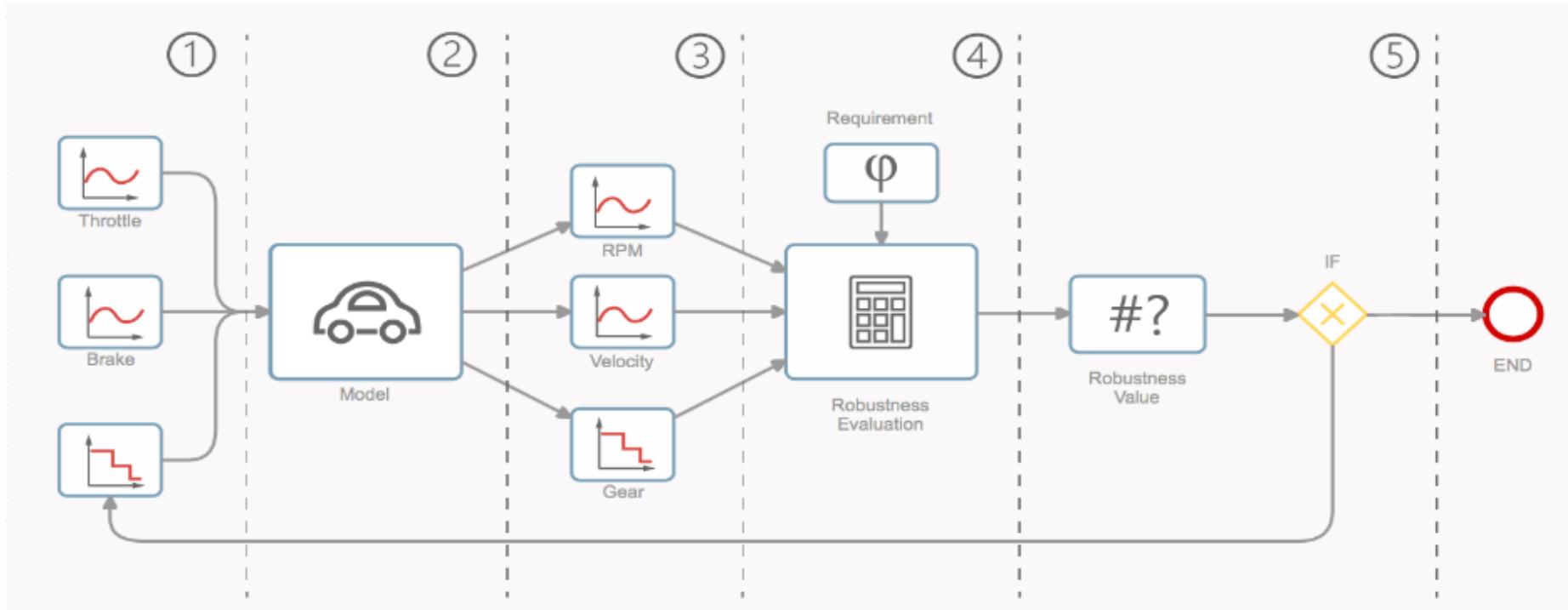
Falsification re-framed

Given:

- ▶ Set of all such input signals : U
- ▶ Input signal \mathbf{u} : function from \mathbb{T} to $D_1 \times \cdots \times D_m$, where $\mathbb{T} \subseteq [0, T]$
- ▶ Model M that maps \mathbf{u} to some signal \mathbf{y} with the same domain as \mathbf{u} , and co-domain some subset of \mathbb{R}^n
- ▶ Property φ that can be evaluated to true/false over given \mathbf{u} and \mathbf{y}

Check: $\exists \mathbf{u} \in U : (\mathbf{y} = M(\mathbf{u})) \not\models \neg\varphi(\mathbf{u}, \mathbf{y})$

Falsification CPS



Goal:

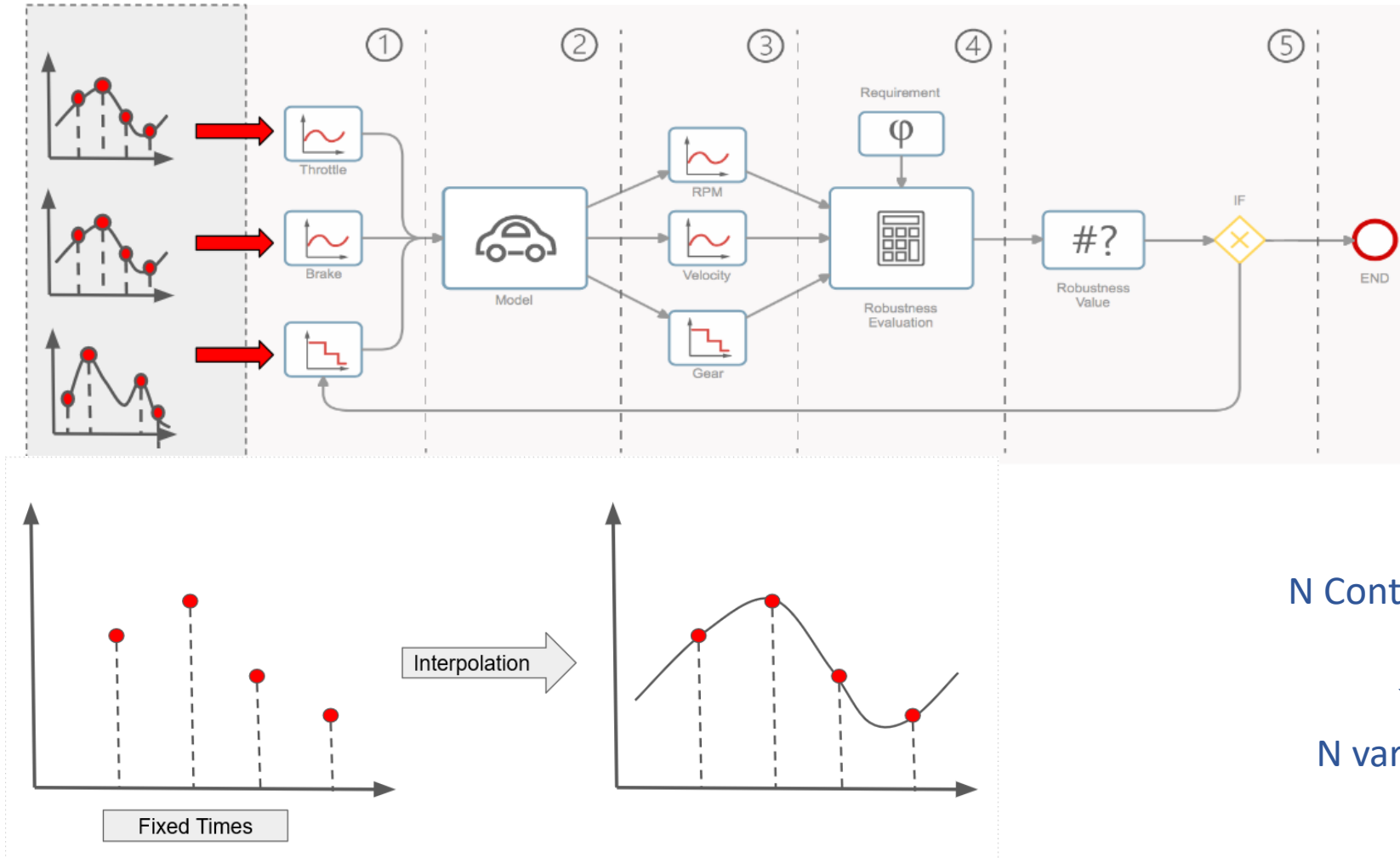
Find the inputs (1) which falsify the requirements (4)

Problems:

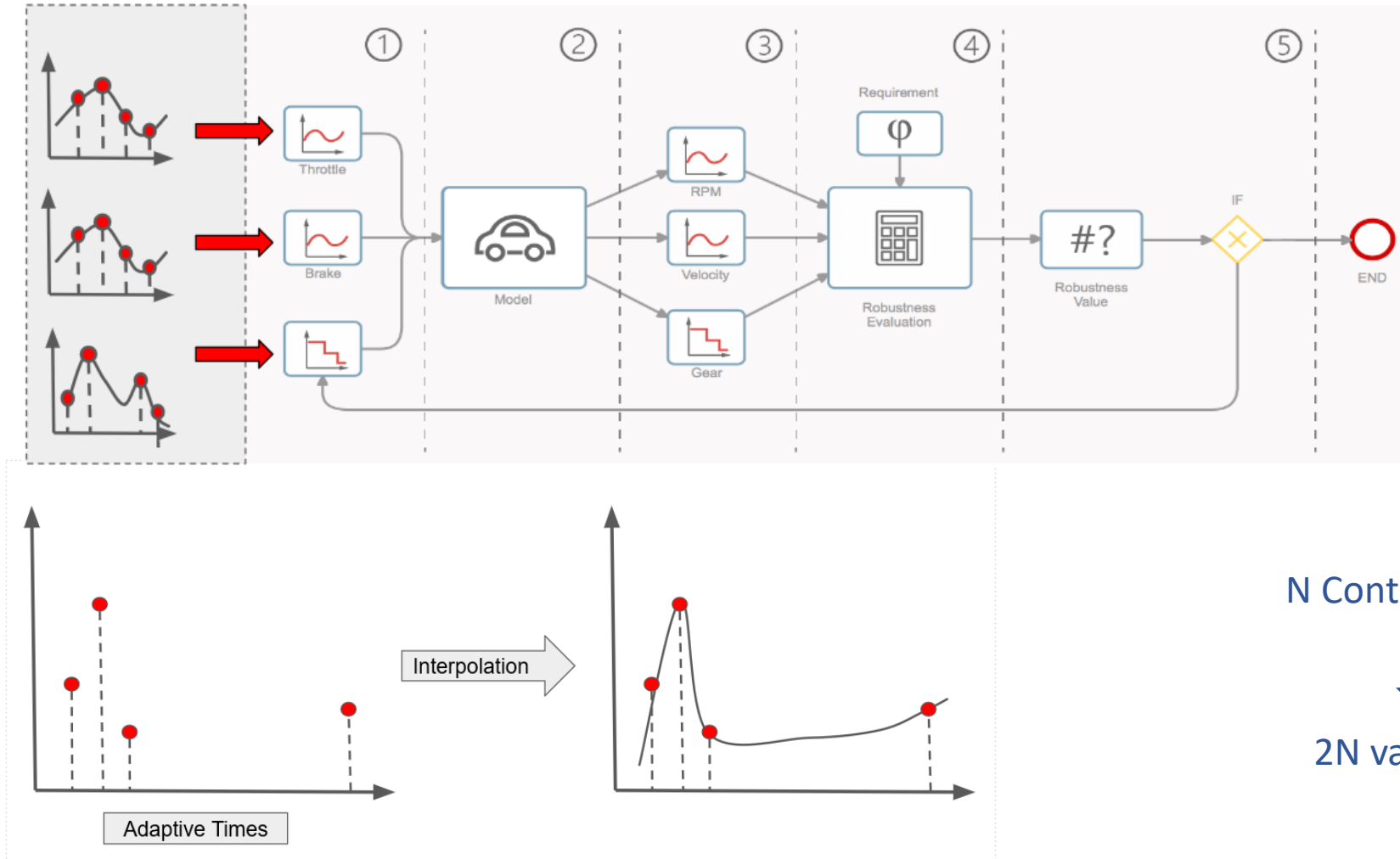
- Falsify with a low number of simulations
- Functional Input Space

Active Learning
Adaptive Parameterization

Adaptive Parameterization



Adaptive Parameterization



N Control points



2N variable

Problem

Finding the trajectories which falsify the requirements, finding $\mathbf{u} \in B$

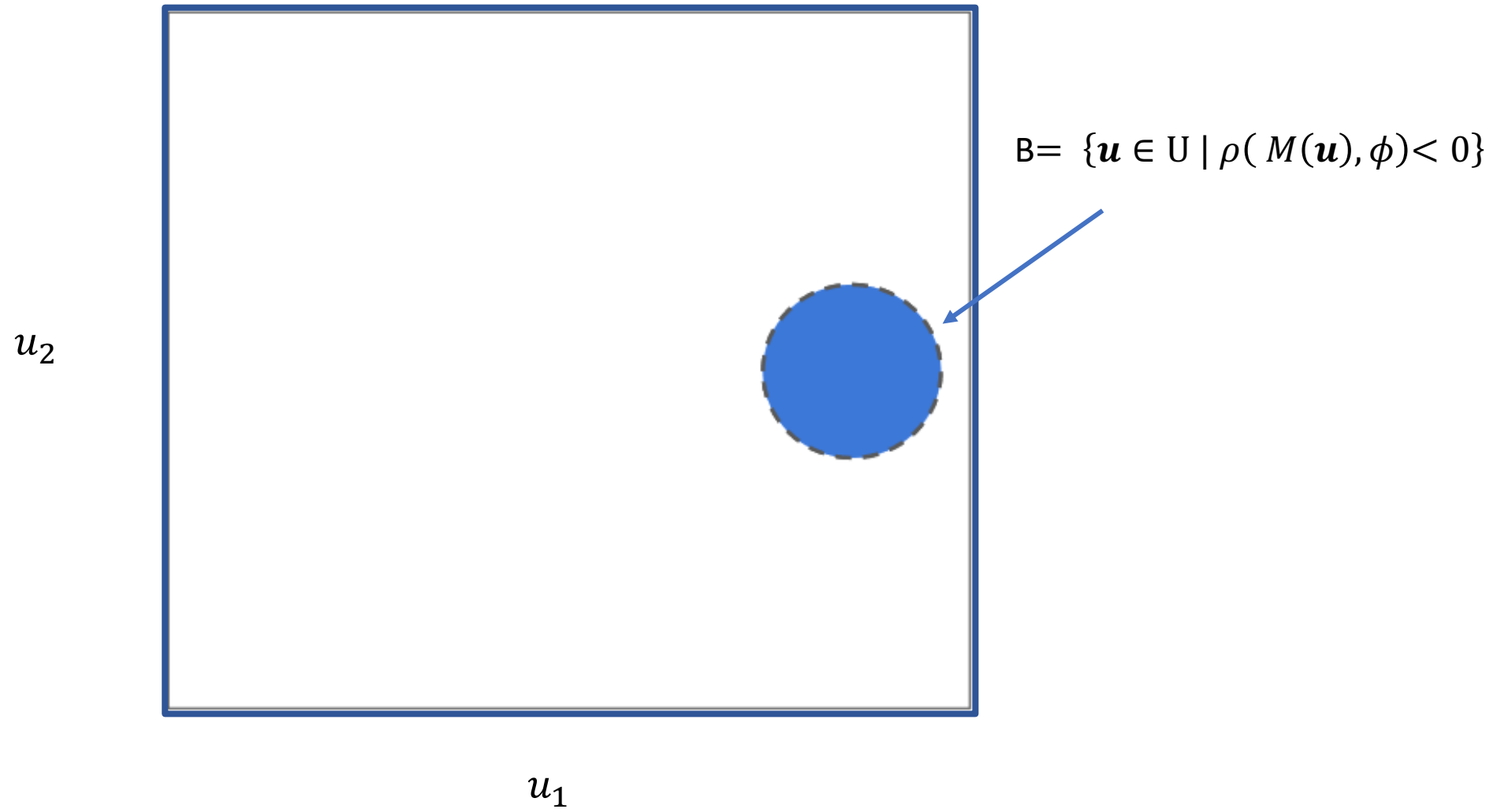
$$B = \{\mathbf{u} \in U \mid \rho(M(\mathbf{u}), \phi) \in (-\infty, 0)\} \subseteq U$$

- **Training Set:** $K = \{\mathbf{u}_i, \rho(M(\mathbf{u}_i), \phi)\}_{i \leq n}$ (the partial knowledge after n iterations)
- **Gaussian Process:** $\rho_K(\mathbf{u}) \sim GP(m_K(\mathbf{u}), \sigma_K(\mathbf{u}))$ (the partial model)

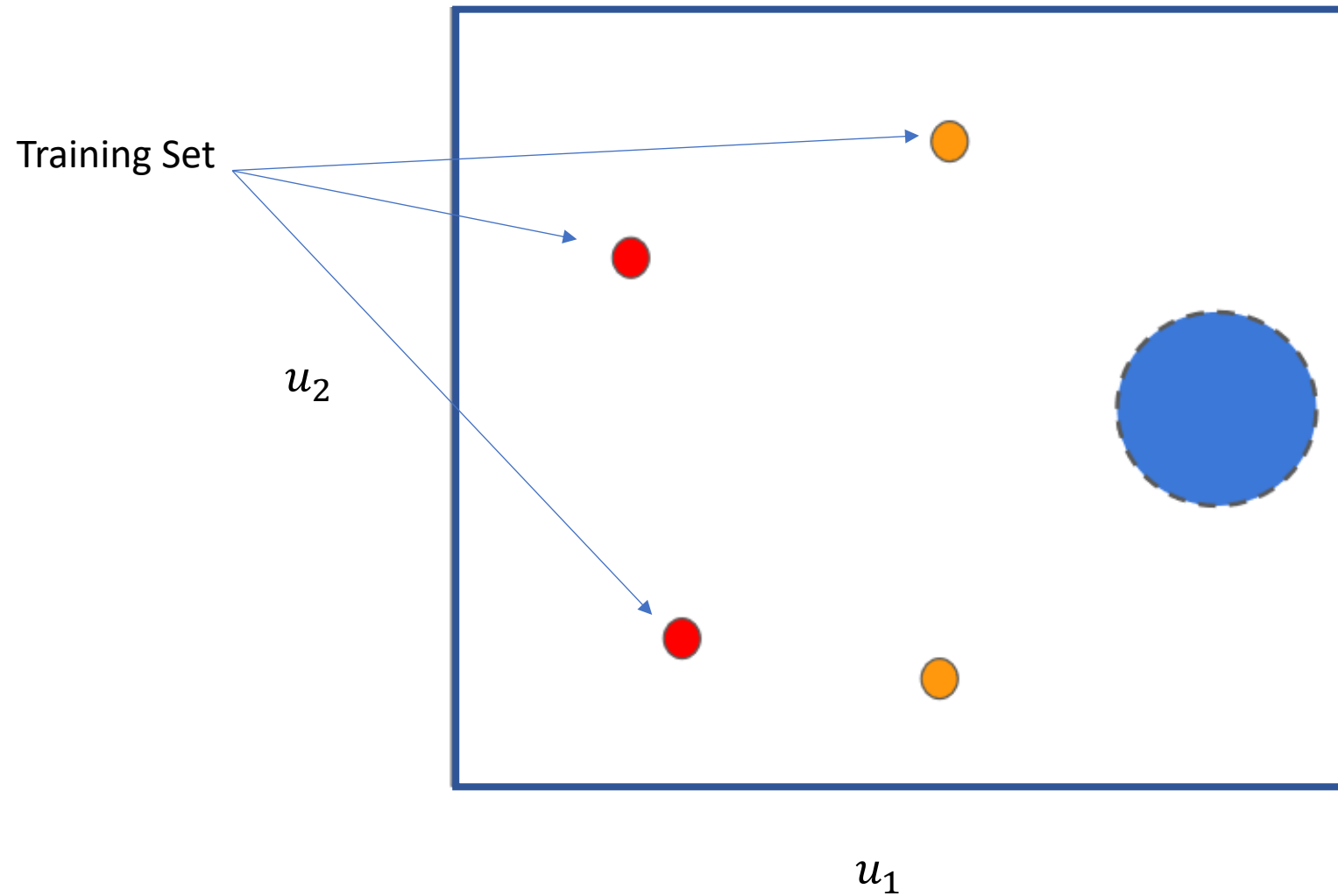
$$P(\rho_K(\mathbf{u}) < 0) = CDF\left(\frac{0 - m_K(\mathbf{u})}{\sigma_K(\mathbf{u})}\right)$$

Idea: implementing an iterative sample strategy in order to increase the probability to sample a point in B, as the number of iterations increases.

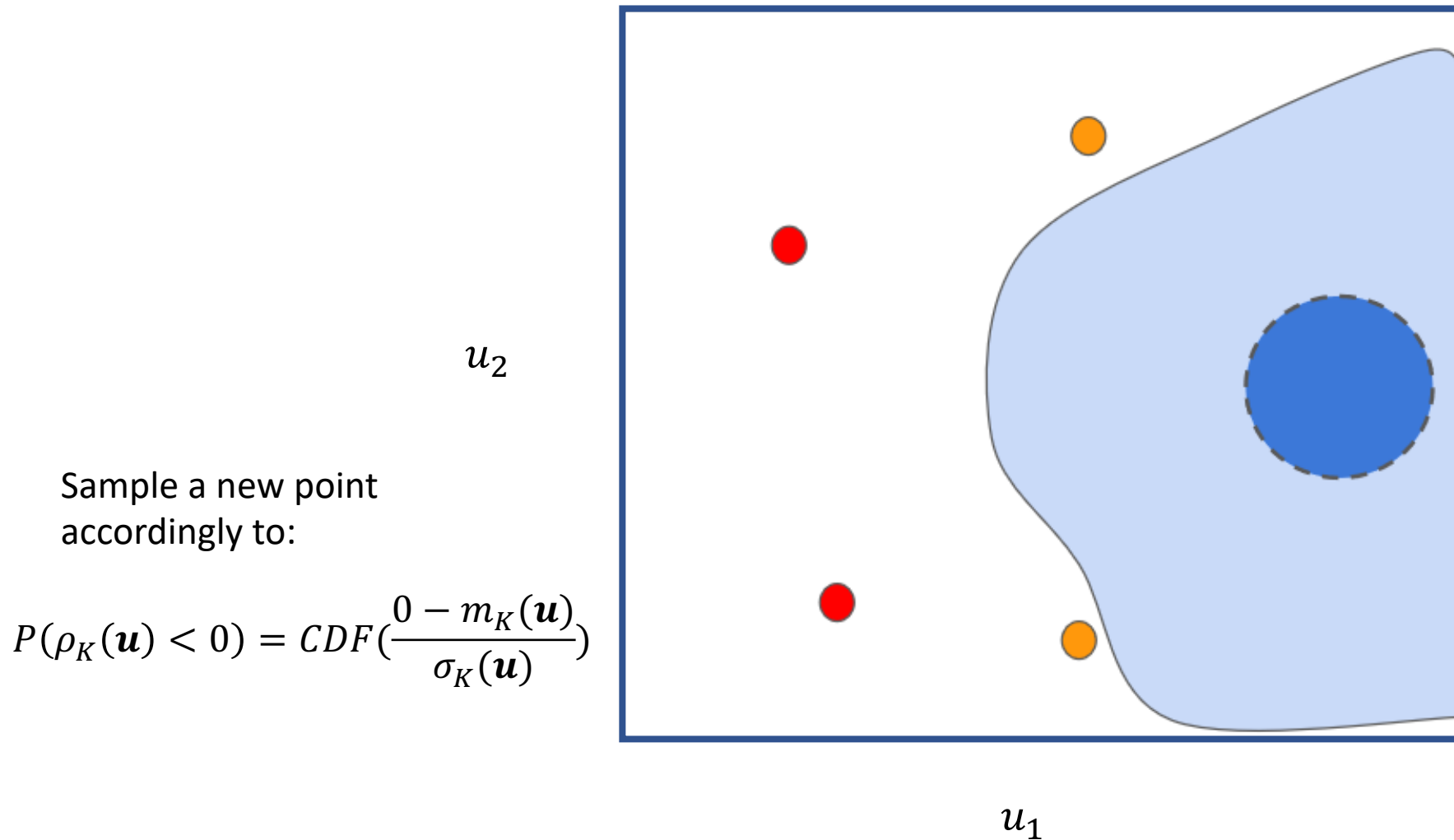
Domain Estimation Algorithm (DEA)



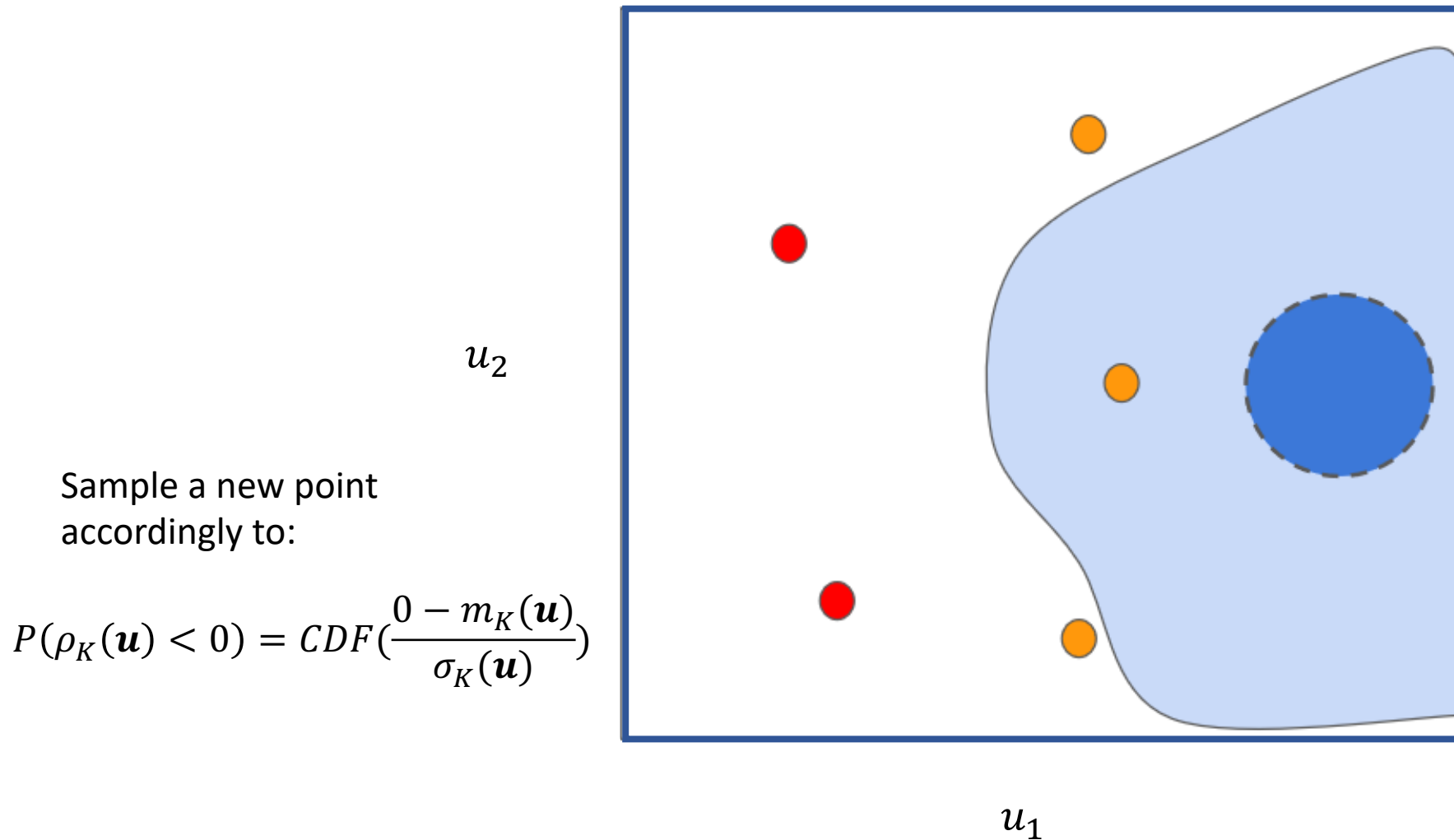
Domain Estimation Algorithm (DEA)



Domain Estimation Algorithm (DEA)



Domain Estimation Algorithm (DEA)

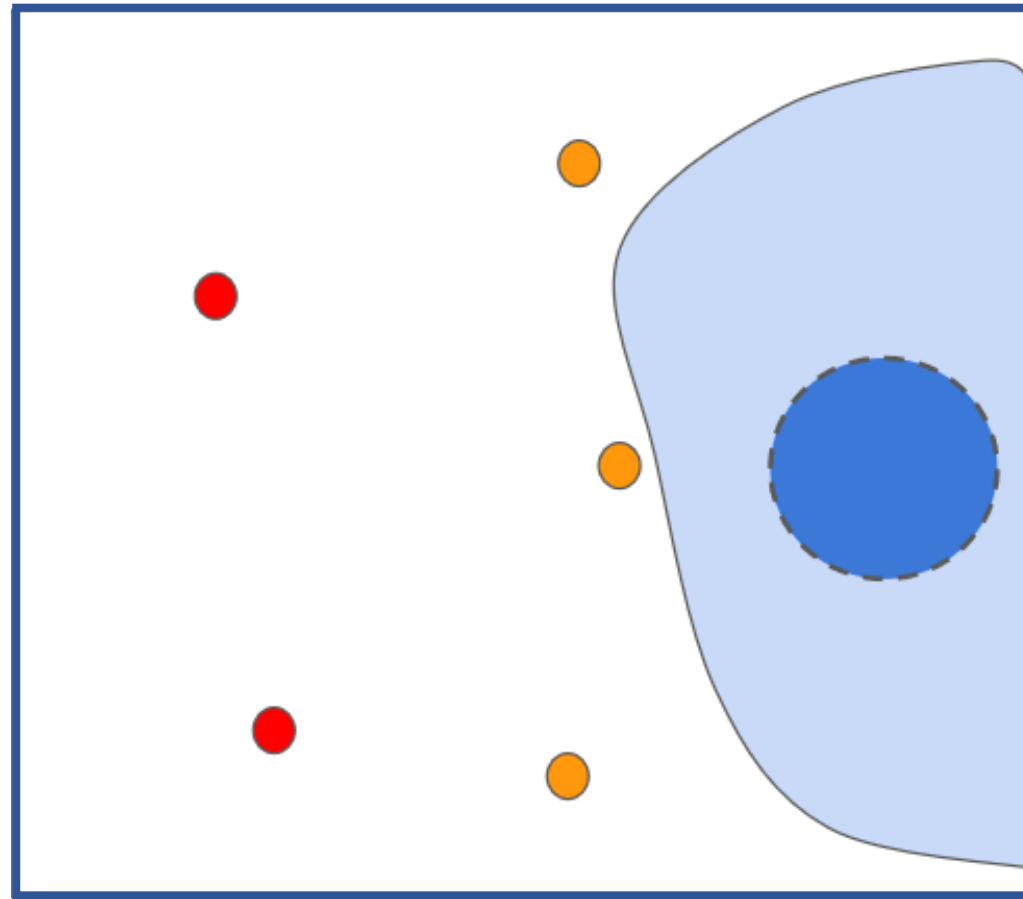


Domain Estimation Algorithm (DEA)

u_2

Sample a new point
accordingly to:

$$P(\rho_K(\mathbf{u}) < 0) = CDF\left(\frac{0 - m_K(\mathbf{u})}{\sigma_K(\mathbf{u})}\right)$$



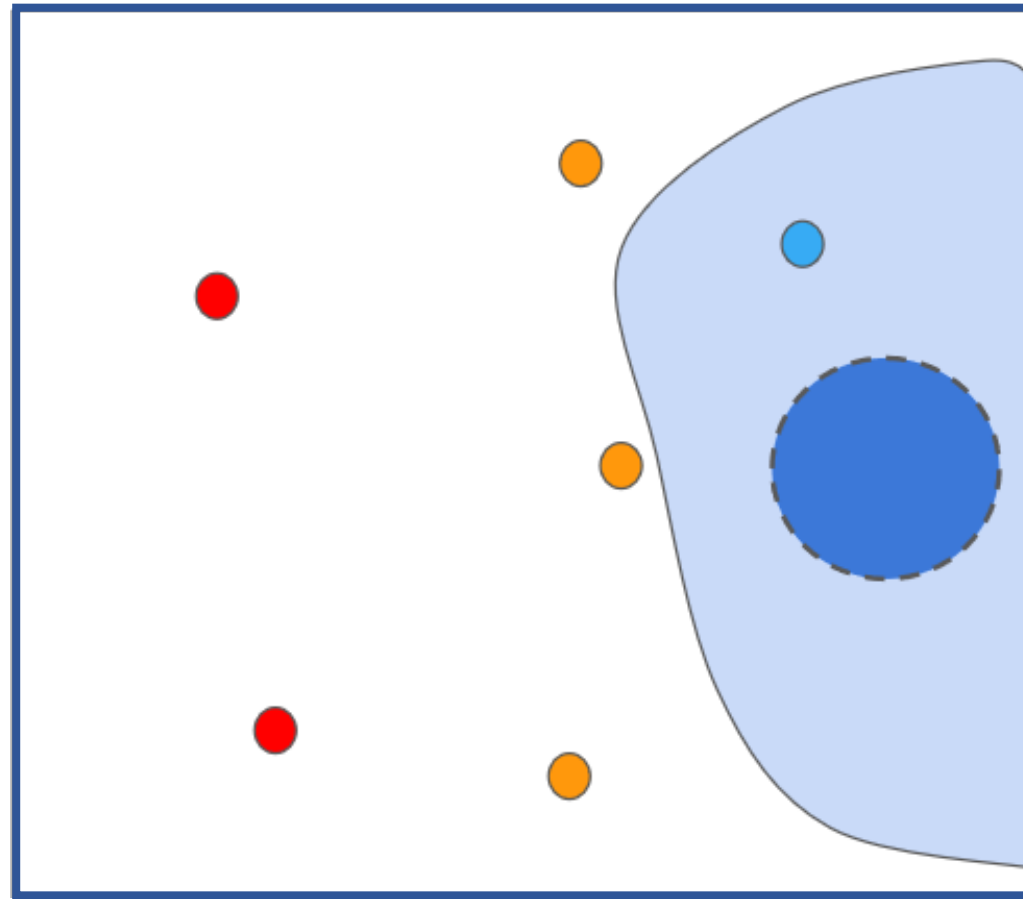
u_1

Domain Estimation Algorithm (DEA)

u_2

Sample a new point
accordingly to:

$$P(\rho_K(\mathbf{u}) < 0) = CDF\left(\frac{0 - m_K(\mathbf{u})}{\sigma_K(\mathbf{u})}\right)$$



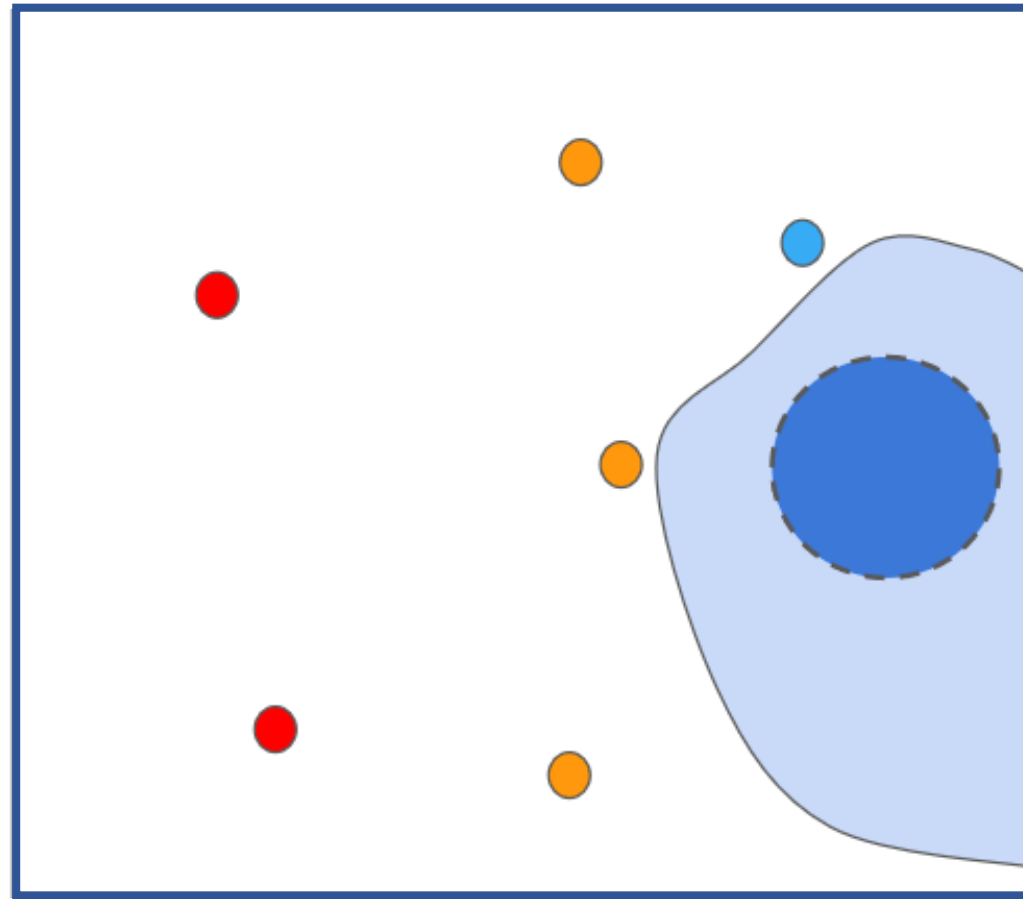
u_1

Domain Estimation Algorithm (DEA)

u_2

Sample a new point
accordingly to:

$$P(\rho_K(\mathbf{u}) < 0) = CDF\left(\frac{0 - m_K(\mathbf{u})}{\sigma_K(\mathbf{u})}\right)$$



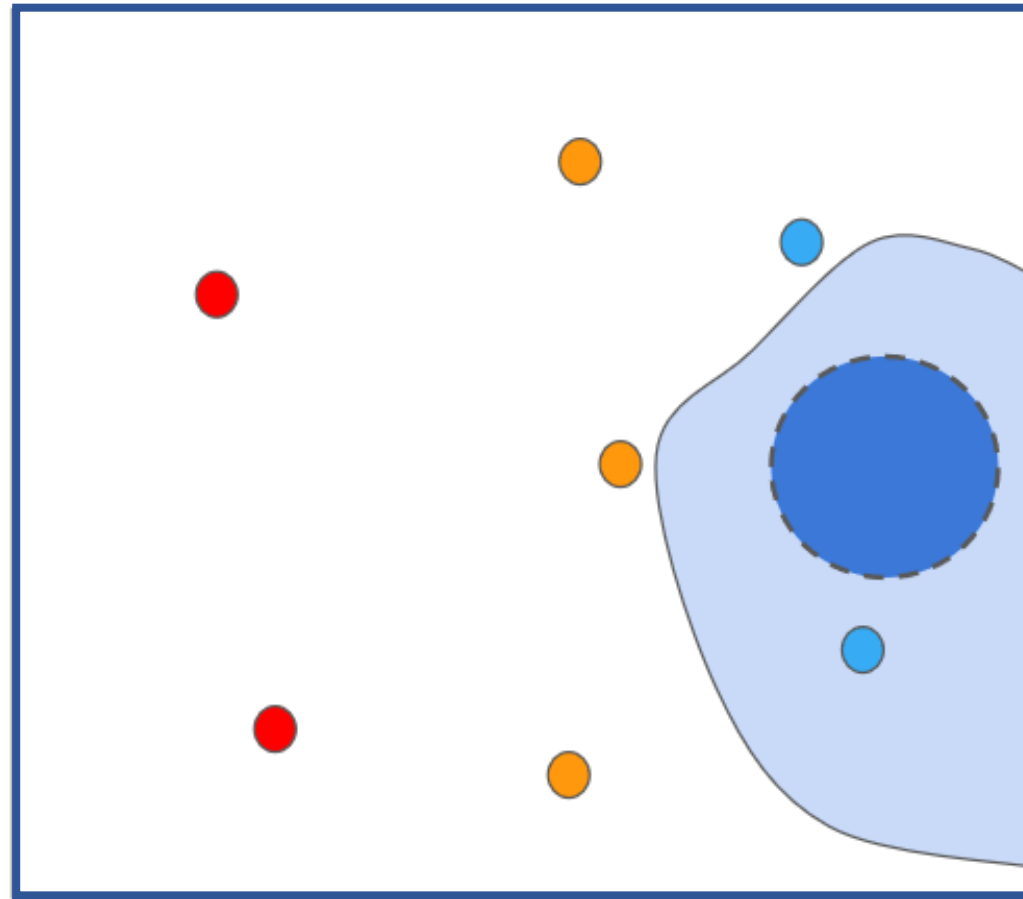
u_1

Domain Estimation Algorithm (DEA)

u_2

Sample a new point
accordingly to:

$$P(\rho_K(\mathbf{u}) < 0) = CDF\left(\frac{0 - m_K(\mathbf{u})}{\sigma_K(\mathbf{u})}\right)$$



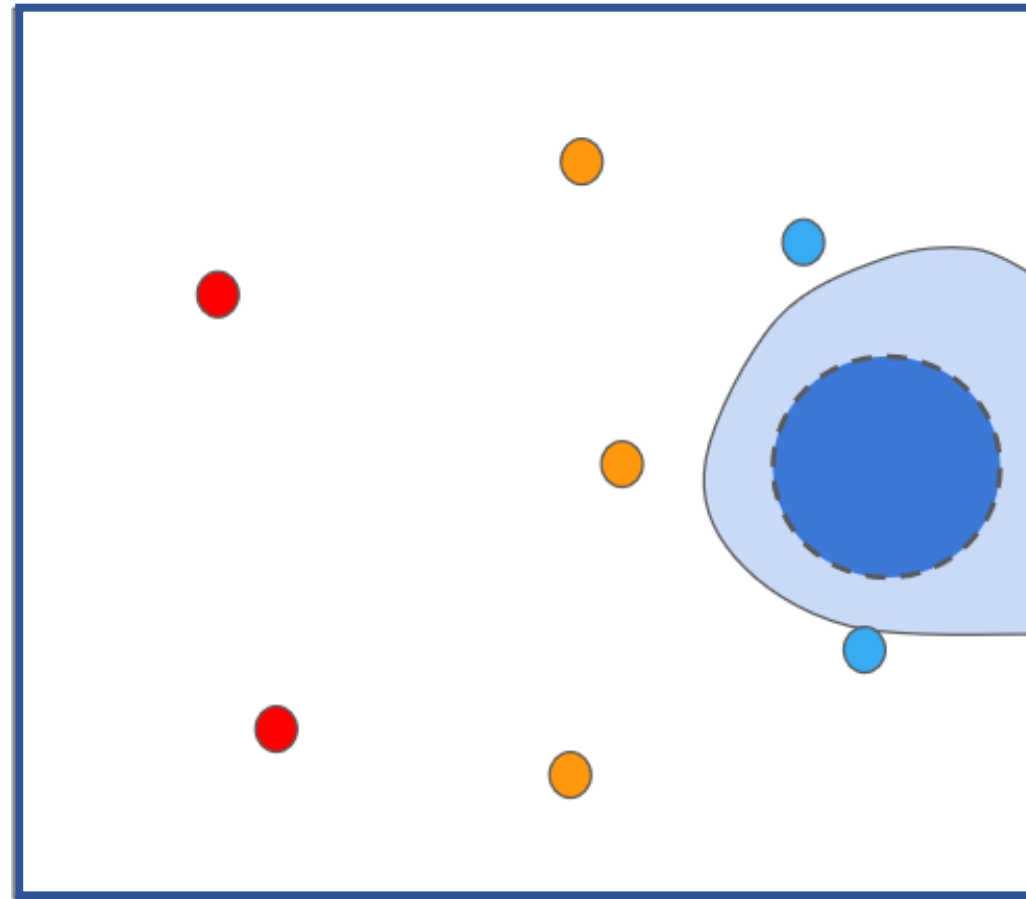
u_1

Domain Estimation Algorithm (DEA)

u_2

Sample a new point
accordingly to:

$$P(\rho_K(\mathbf{u}) < 0) = CDF\left(\frac{0 - m_K(\mathbf{u})}{\sigma_K(\mathbf{u})}\right)$$



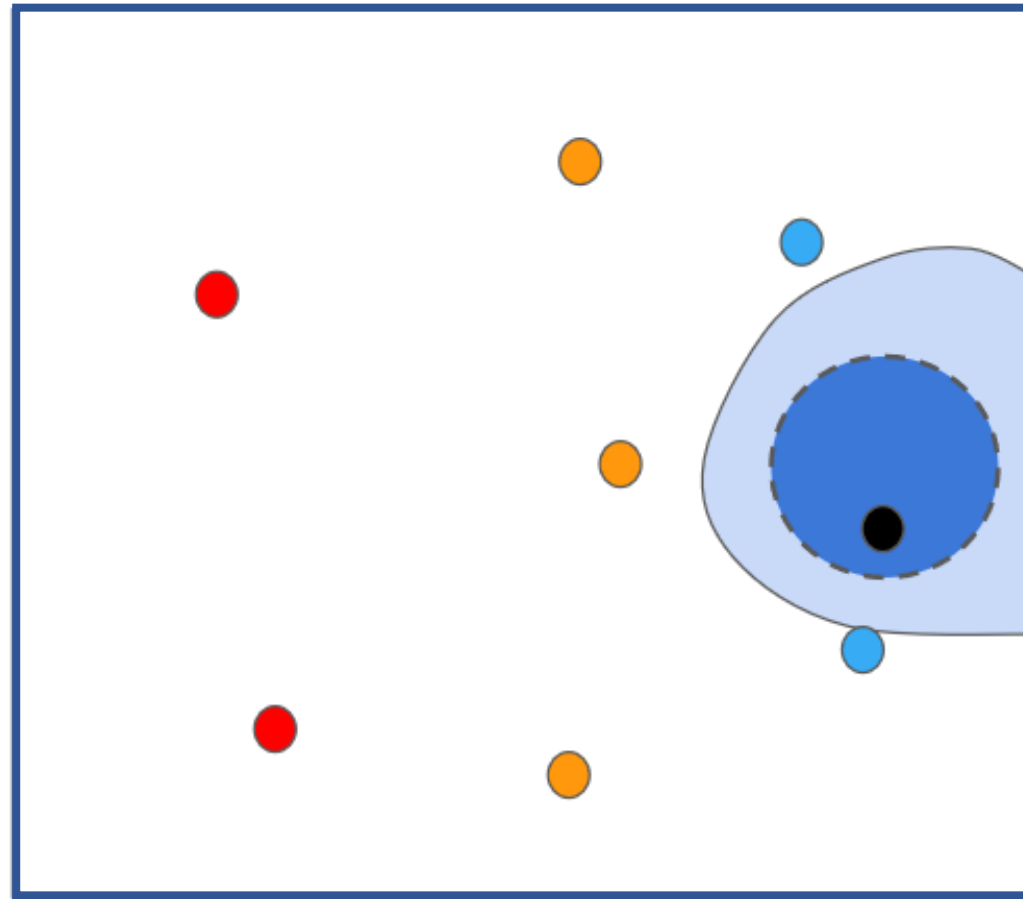
u_1

Domain Estimation Algorithm (DEA)

u_2

Sample a new point
accordingly to:

$$P(\rho_K(\mathbf{u}) < 0) = CDF\left(\frac{0 - m_K(\mathbf{u})}{\sigma_K(\mathbf{u})}\right)$$



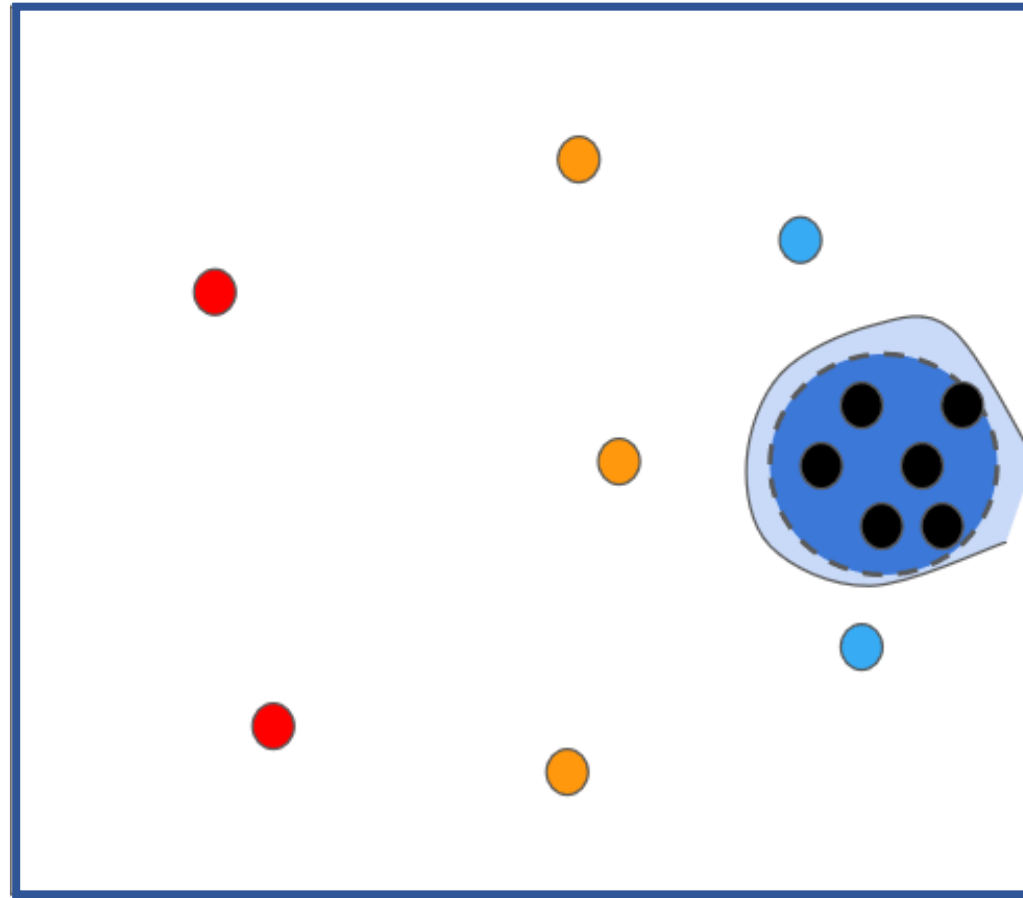
u_1

Domain Estimation Algorithm (DEA)

u_2

Sample a new point
accordingly to:

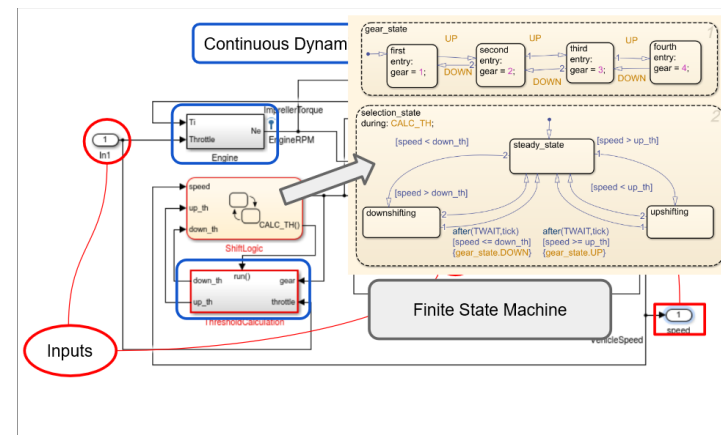
$$P(\rho_K(\mathbf{u}) < 0) = CDF\left(\frac{0 - m_K(\mathbf{u})}{\sigma_K(\mathbf{u})}\right)$$



u_1

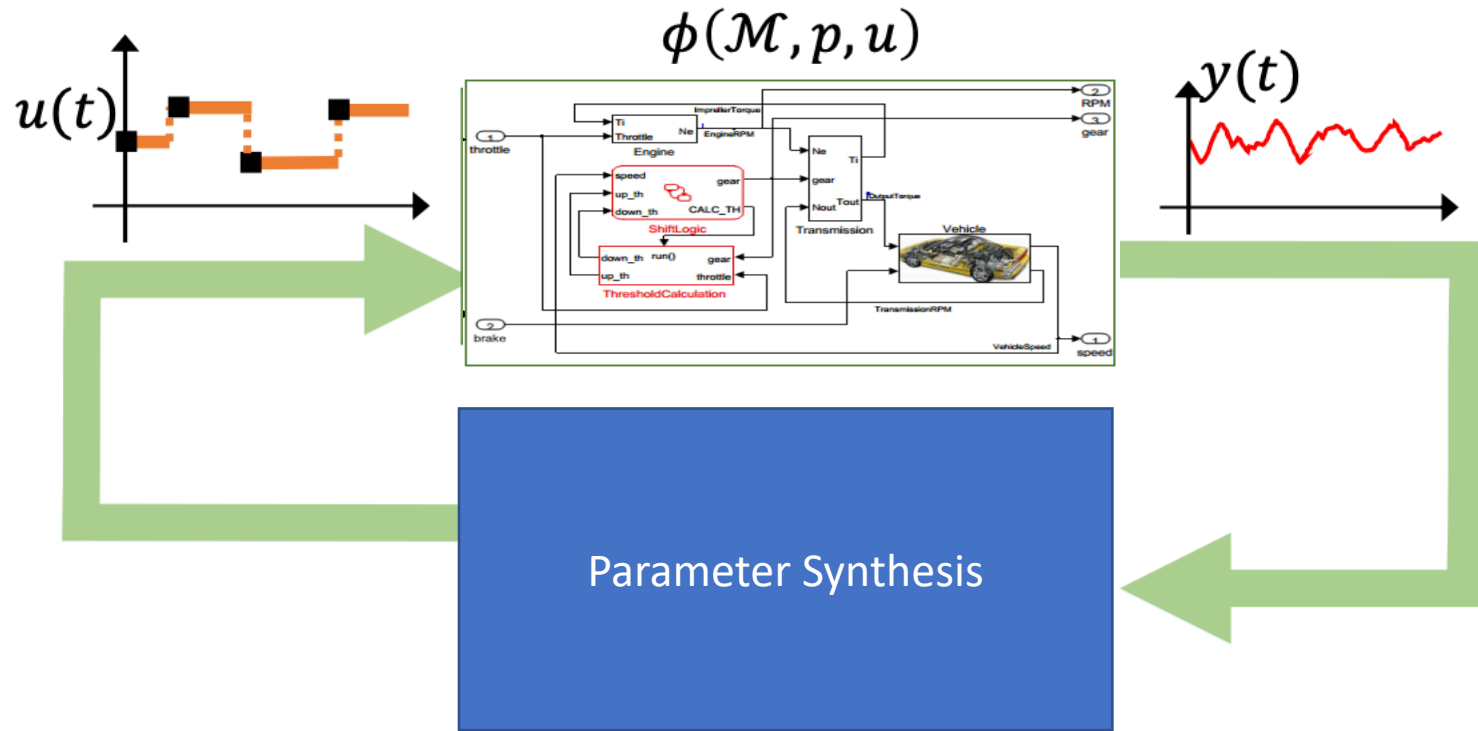
Tests Case & Results

- $\phi_1(\bar{v}, \bar{\omega}) = \mathbf{G}_{[0,30]}(v \leq \bar{v} \wedge \omega \leq \bar{\omega})$ (in the next 30 seconds the engine and vehicle speed never reach $\bar{\omega}$ rpm and \bar{v} km/h, respectively)
- $\phi_2(\bar{v}, \bar{\omega}) = \mathbf{G}_{[0,30]}(\omega \leq \bar{\omega}) \rightarrow \mathbf{G}_{[0,10]}(v \leq \bar{v})$ (if the engine speed is always less than $\bar{\omega}$ rpm, then the vehicle speed can not exceed \bar{v} km/h in less than 10 sec)
- $\phi_3(\bar{v}, \bar{\omega}) = \mathbf{F}_{[0,10]}(v \geq \bar{v}) \rightarrow \mathbf{G}_{[0,30]}(\omega \leq \bar{\omega})$ (the vehicle speed is above \bar{v} km/h than from that point on the engine speed is always less than $\bar{\omega}$ rpm)



Req	Adaptive DEA		Adaptive GP-UCB		S-TaLiRo		Alg
	nval	times	nval	times	nval	times	
ϕ_1	4.42 ± 0.53	2.16 ± 0.61	4.16 ± 2.40	0.55 ± 0.30	5.16 ± 4.32	0.57 ± 0.48	UR
ϕ_1	6.90 ± 2.22	5.78 ± 3.88	8.7 ± 1.78	1.52 ± 0.40	39.64 ± 44.49	4.46 ± 4.99	SA
ϕ_2	3.24 ± 1.98	1.57 ± 1.91	7.94 ± 3.90	1.55 ± 1.23	12.78 ± 11.27	1.46 ± 1.28	CE
ϕ_2	10.14 ± 2.95	12.39 ± 6.96	23.9 ± 7.39	9.86 ± 4.54	59 ± 42	6.83 ± 4.93	SA
ϕ_2	8.52 ± 2.90	9.13 ± 5.90	13.6 ± 3.48	4.12 ± 1.67	43.1 ± 39.23	4.89 ± 4.43	SA
ϕ_3	5.02 ± 0.97	2.91 ± 1.20	5.44 ± 3.14	0.91 ± 0.67	10.04 ± 7.30	1.15 ± 0.84	CE
ϕ_3	7.70 ± 2.36	7.07 ± 3.87	10.52 ± 1.76	2.43 ± 0.92	11 ± 9.10	1.25 ± 1.03	UR

Parameter Synthesis



Parameter Synthesis

Problem

Given a model, depending on a set of parameters $\theta \in \Theta$, and a specification ϕ (STL formula), find the parameter combination θ s.t. the system satisfies ϕ as more as possible



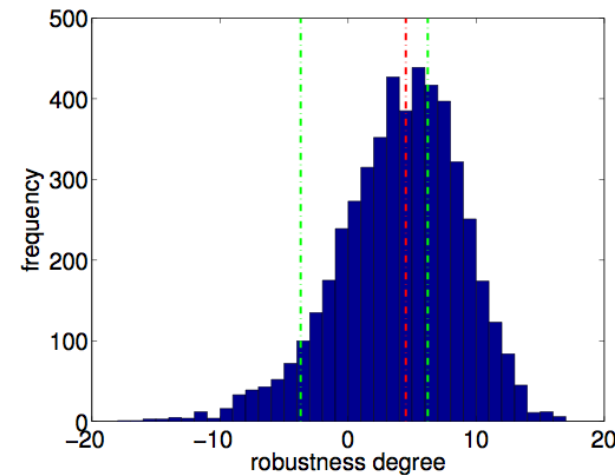
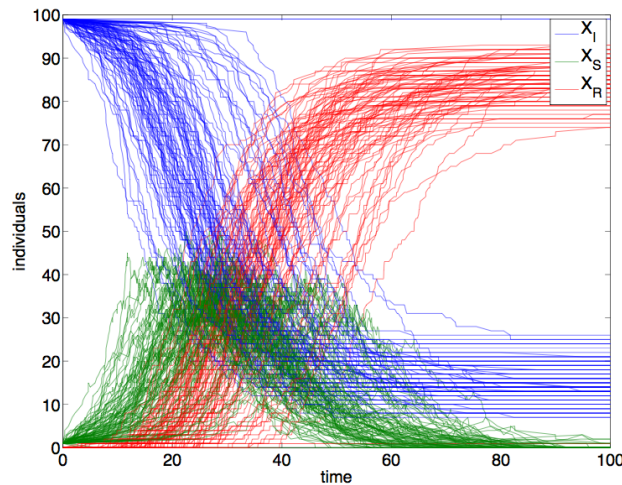
Solution Strategy

- **rephrase** it as a optimisation problem (maximizing ρ)
- **evaluate** the function to optimise
- **solve** the optimisation problem

Parameter Synthesis via Robustness Maximisation

Robustness Distribution

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



Indicators

$$\mathbb{E}(R_\varphi)$$

(the average robustness degree)

$$\mathbb{E}(R_\varphi \mid R_\varphi > 0) \quad \text{and} \quad \mathbb{E}(R_\varphi \mid R_\varphi < 0)$$

(the conditional averages)

Parameter Synthesis

Problem

Find the parameter configuration that maximizes $E[R_\phi](\theta)$, of which we have few costly and noisy evaluations.

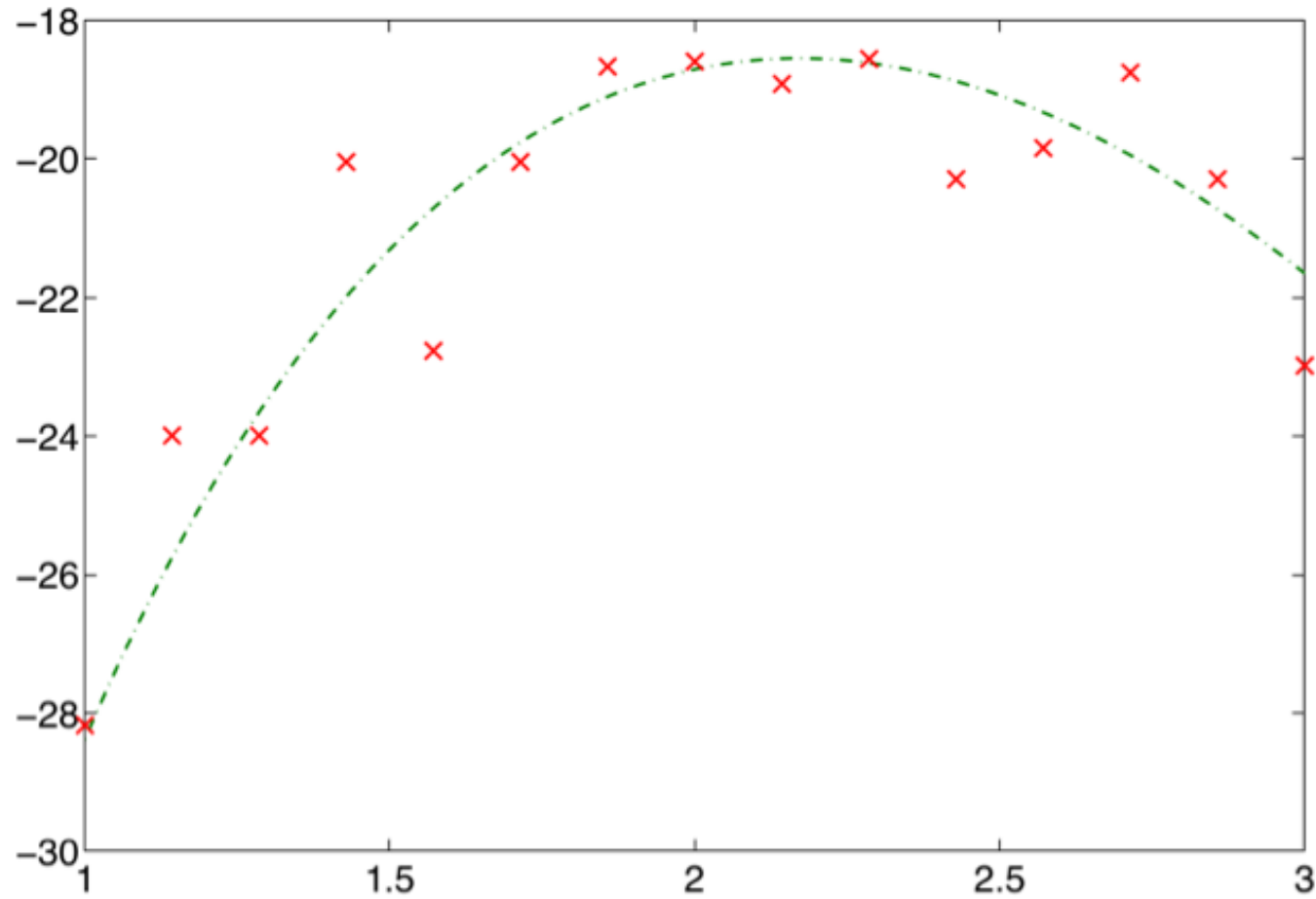


Methodology

1. Sample $\{(\theta_{(i)}, y_{(i)}), i = 1, \dots, n\}$
2. Emulate (**GP Regression**): $E[R_\phi] \sim \text{GP}(\mu, k)$
3. Optimize the emulation via **GP-UCB algorithm**, new $\theta_{(n+1)}$

(1) Sample

Collection of the **training set** $\{(\theta^{(i)}, y^{(i)}), i = 1, \dots, m\}$ for parameters values θ .

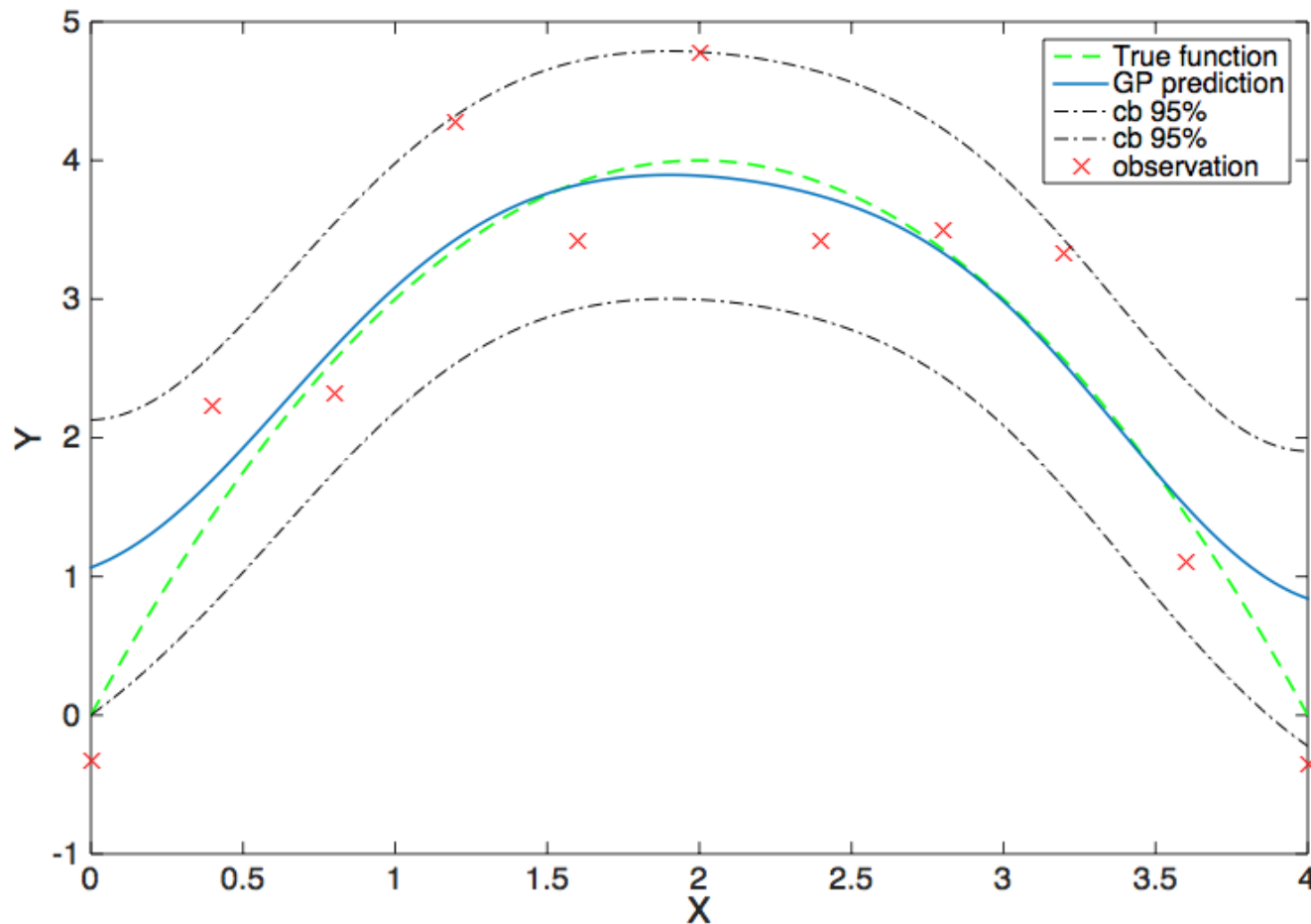


(2) The GP Regression

We have noisy **observations** y of the function value distributed around an unknown **true value** $f(\theta)$ with spherical Gaussian noise

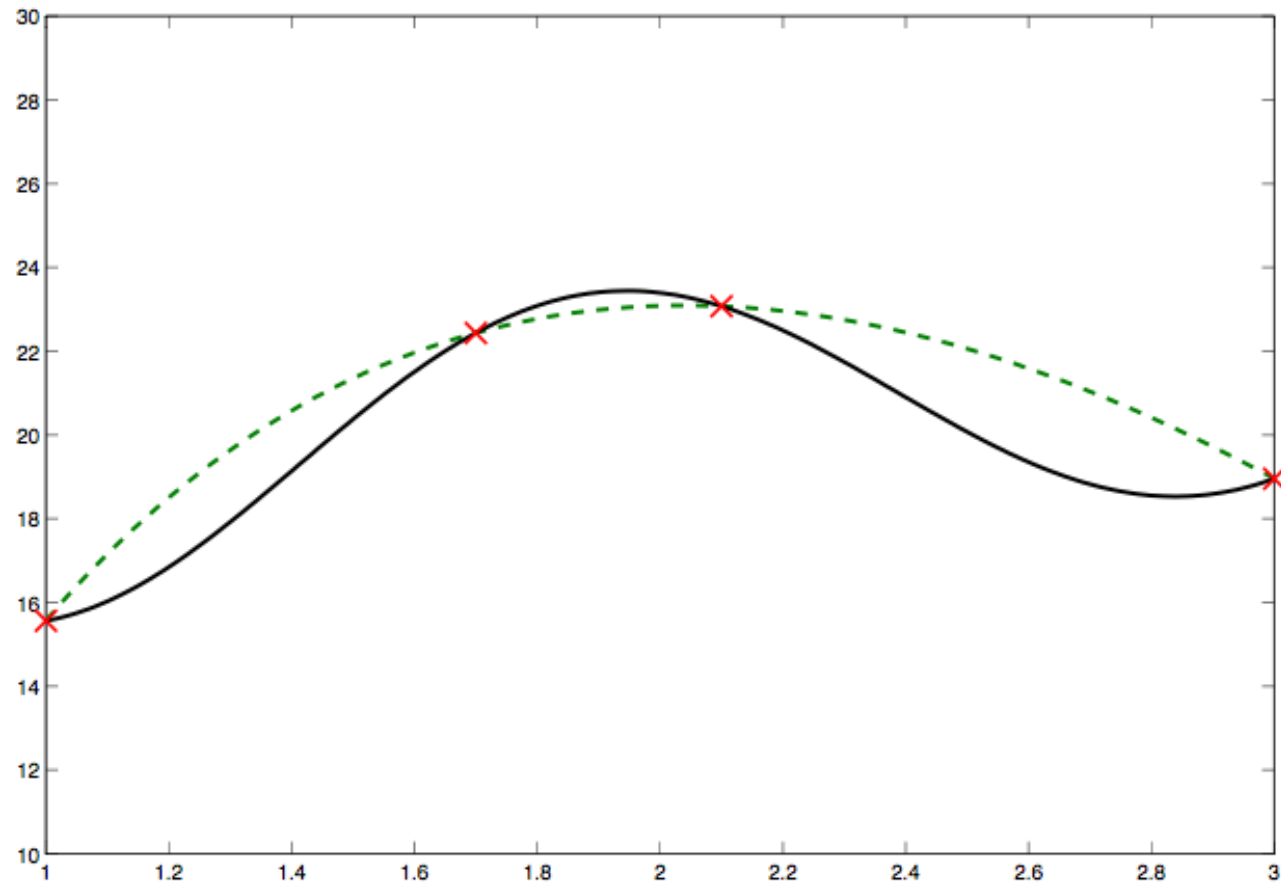
(2) The GP Regression

We have noisy **observations** y of the function value distributed around an unknown **true value** $f(\theta)$ with spherical Gaussian noise



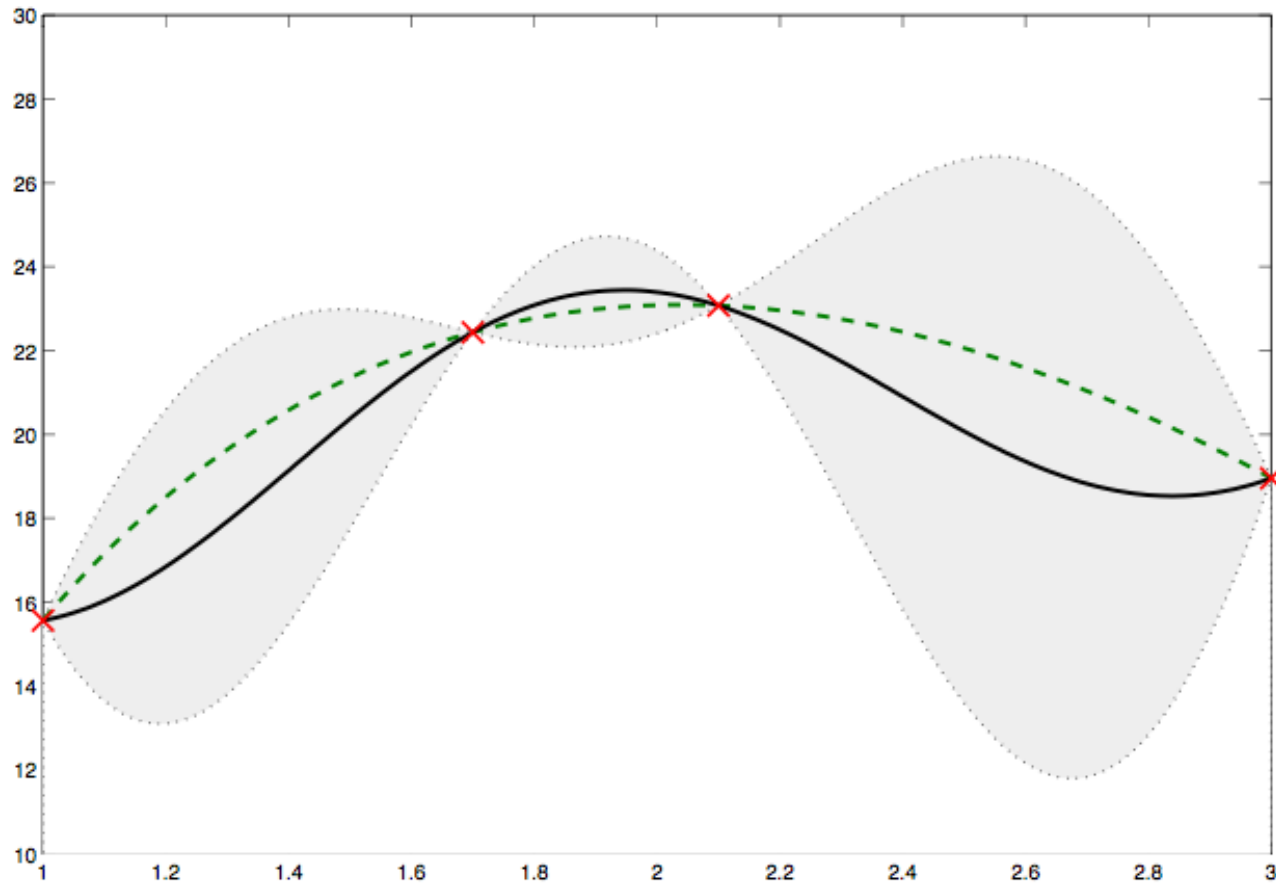
(3) The GP-UCB Algorithm

Balance Exploration and Exploitation: we maximise the **95% upper quantile of the distribution**: $\theta_{t+1} = \operatorname{argmax}_{\theta} [\mu^*(\theta) + \beta_t \sqrt{k^*(\theta, \theta)}]$



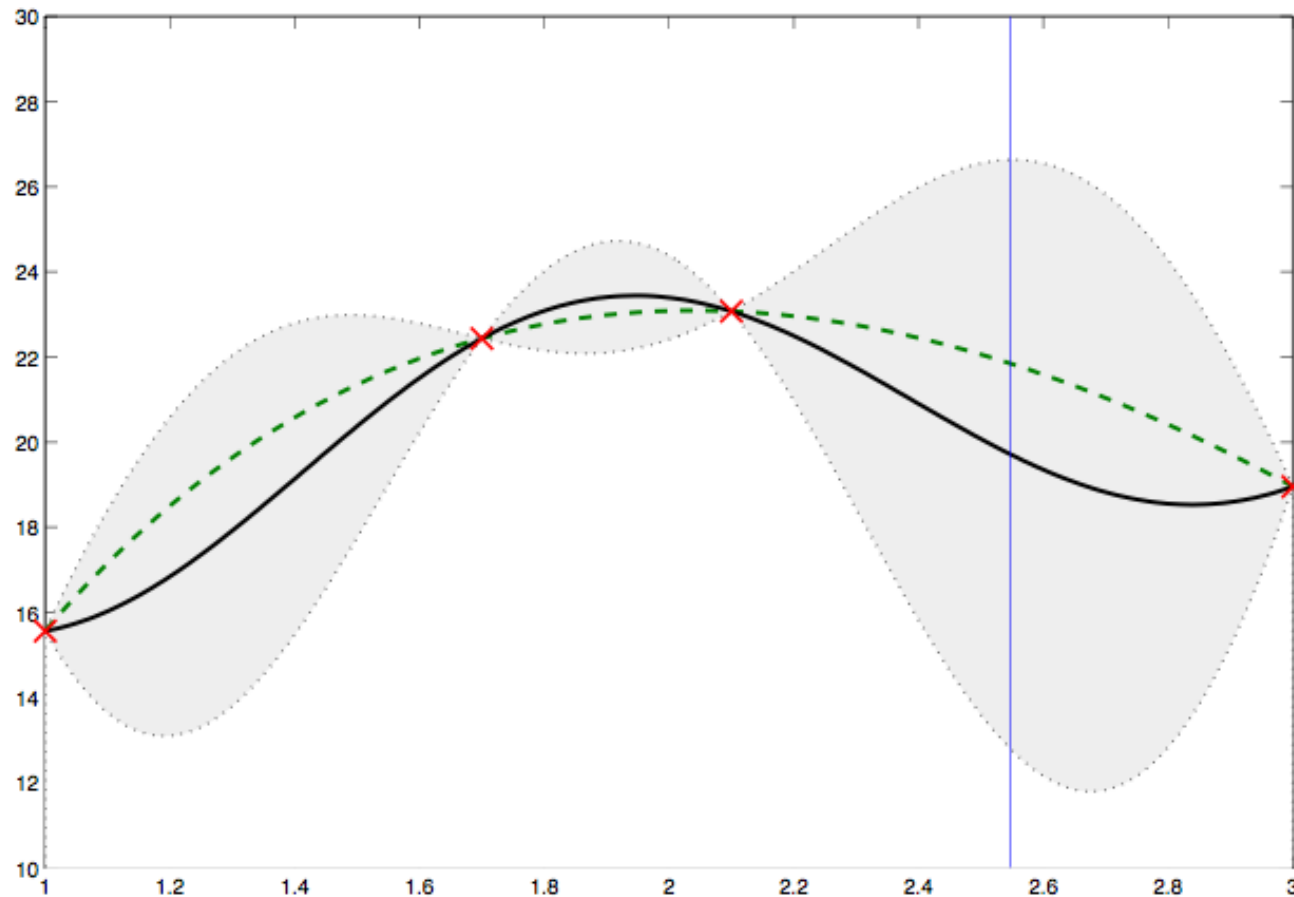
(3) The GP-UCB Algorithm

Balance Exploration and Exploitation: we maximise the **95% upper quantile of the distribution**: $\theta_{t+1} = \operatorname{argmax}_{\theta} [\mu^*(\theta) + \beta_t \sqrt{k^*(\theta, \theta)}]$



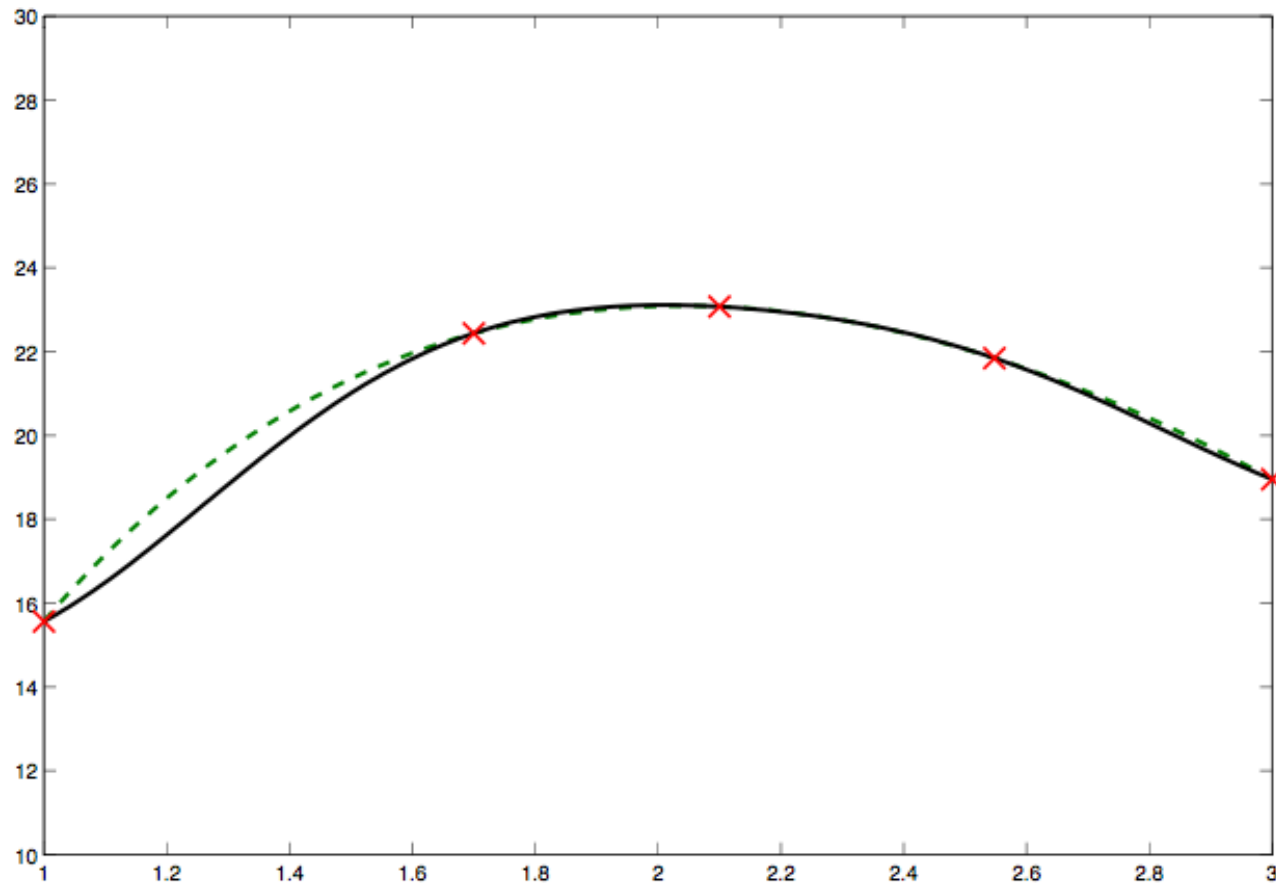
(3) The GP-UCB Algorithm

Balance Exploration and Exploitation: we maximise the **95% upper quantile of the distribution**: $\theta_{t+1} = \operatorname{argmax}_{\theta} [\mu^*(\theta) + \beta_t \sqrt{k^*(\theta, \theta)}]$



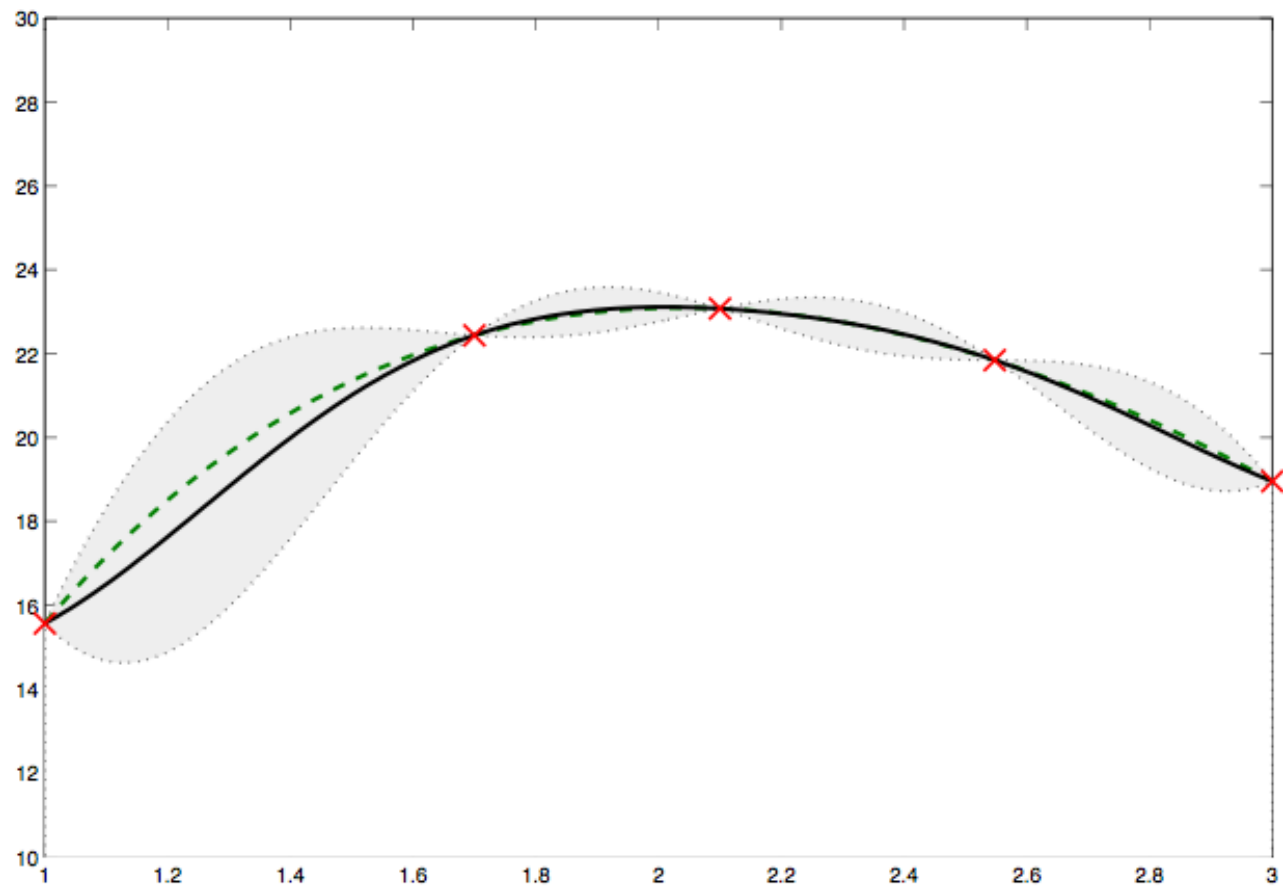
(3) The GP-UCB Algorithm

Balance Exploration and Exploitation: we maximise the **95% upper quantile of the distribution**: $\theta_{t+1} = \operatorname{argmax}_{\theta} [\mu^*(\theta) + \beta_t \sqrt{k^*(\theta, \theta)}]$



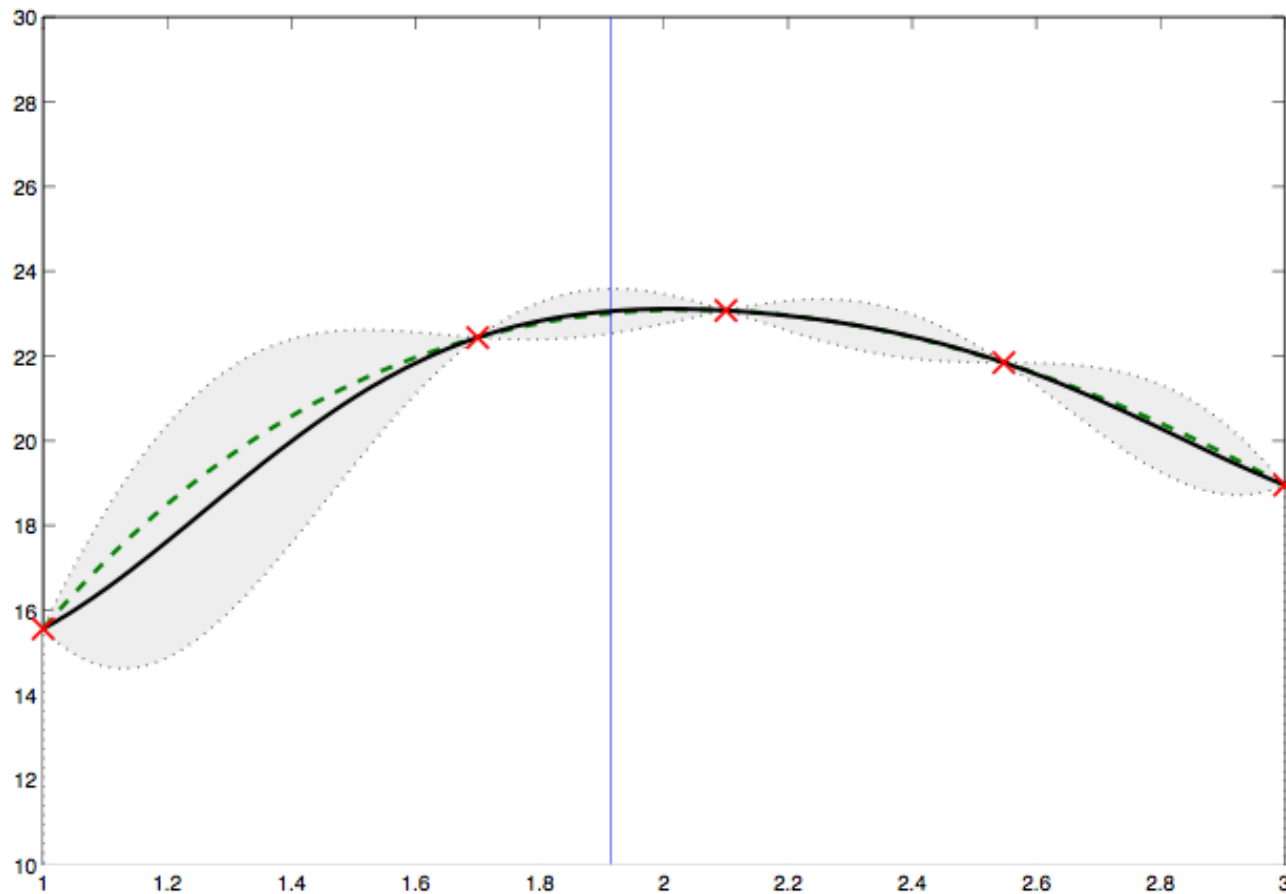
(3) The GP-UCB Algorithm

Balance Exploration and Exploitation: we maximise the **95% upper quantile of the distribution**: $\theta_{t+1} = \operatorname{argmax}_{\theta} [\mu^*(\theta) + \beta_t \sqrt{k^*(\theta, \theta)}]$

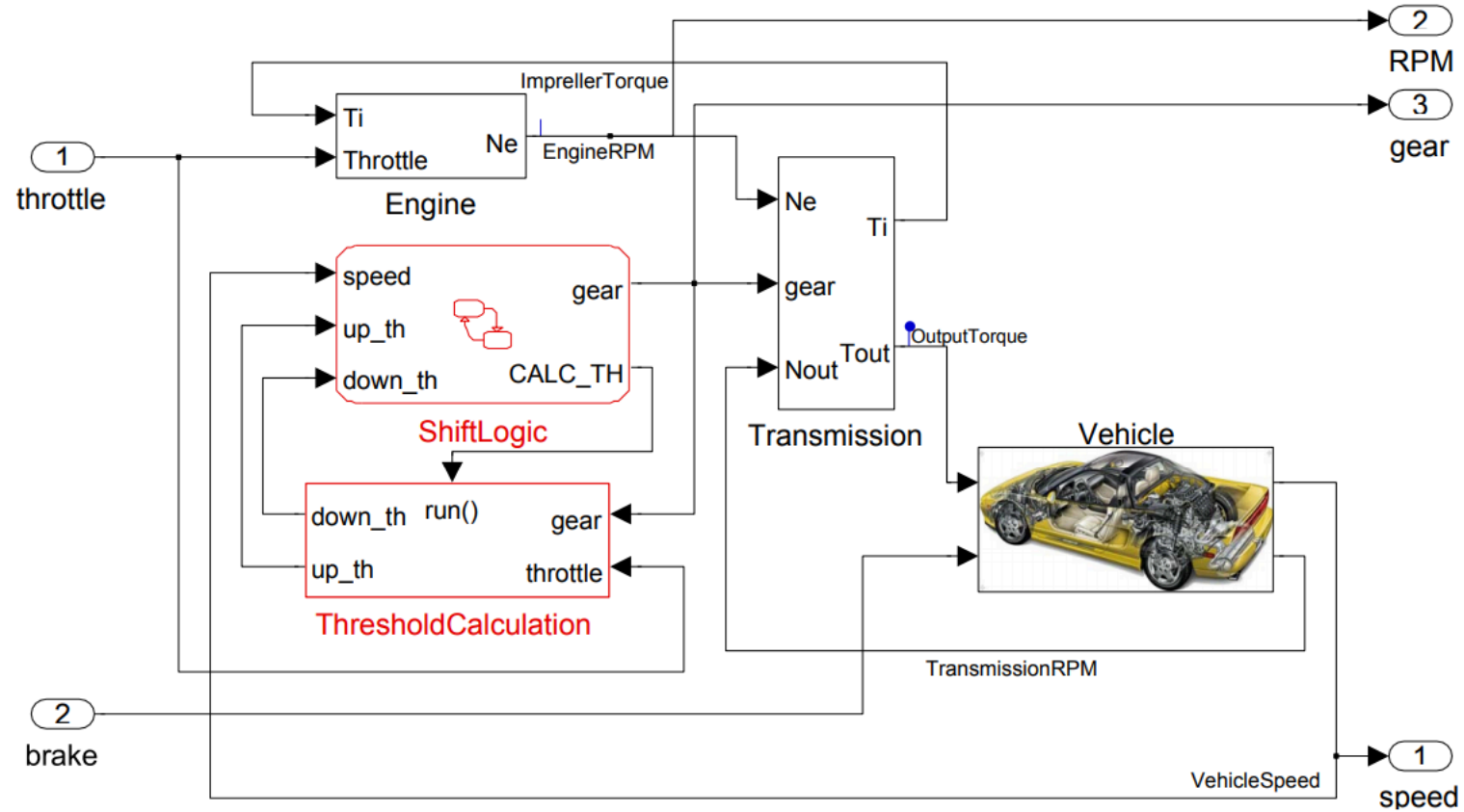


(3) The GP-UCB Algorithm

Balance Exploration and Exploitation: we maximise the **95% upper quantile of the distribution**: $\theta_{t+1} = \operatorname{argmax}_{\theta} [\mu^*(\theta) + \beta_t \sqrt{k^*(\theta, \theta)}]$



Specification Mining



- What is the maximum speed that the vehicle can reach ?
- What is the minimum dwell time in a given gear ?

Parametric Signal Temporal Logic

Definition (PSTL syntax)

$$\phi := (x_i \bowtie \pi) \mid \neg \varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \mathcal{U}_{[\tau_1, \tau_2]} \varphi_2$$

with $\bowtie \in \{>, \leq\}$

- ▶ π is **threshold** parameter
- ▶ τ_1, τ_2 are **temporal** parameters

- ▶ $\mathbb{K} = (\mathcal{T} \times \mathcal{C})$ be the **parameter space**

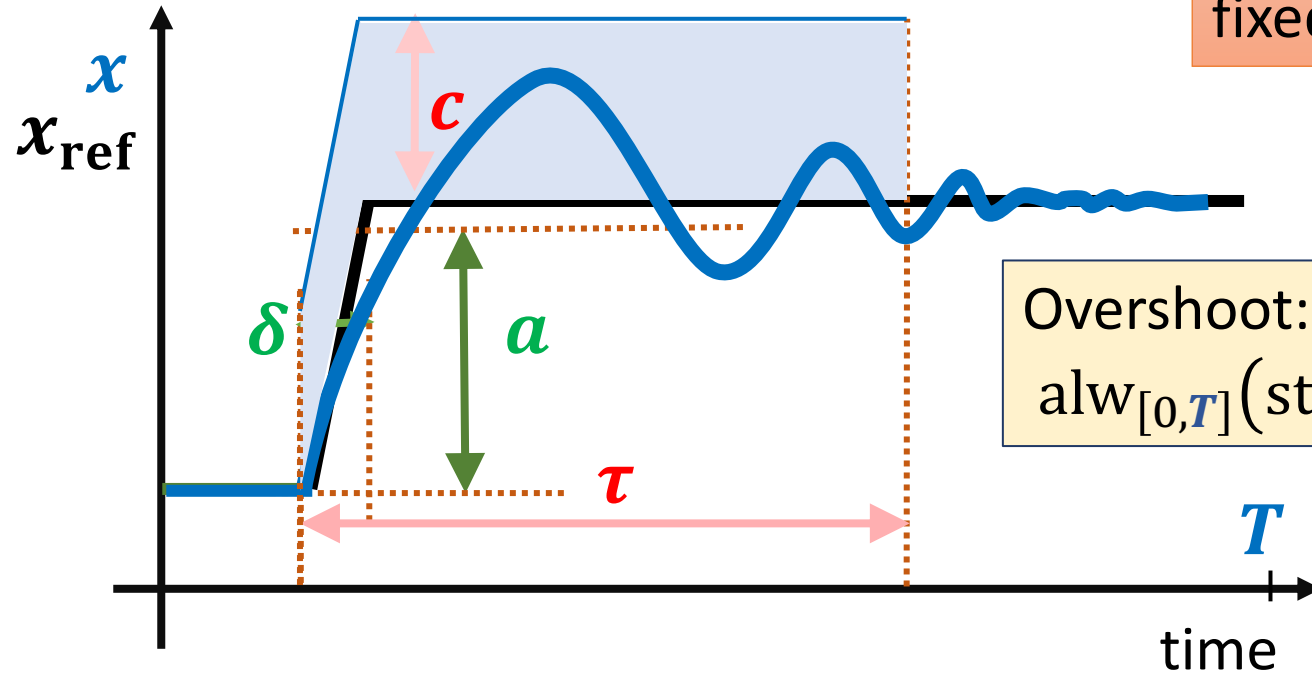
- ▶ $\theta \in \mathbb{K}$ is a **parameter configuration**

e.g., $\phi = \mathcal{F}_{[a,b]}(x_i > k), \theta = (0, 2, 3.5)$ then $\phi_\theta = \mathcal{F}_{[0,2]}(x_i > 3.5)$.

Specification Mining

- ▶ Specification Mining: Try to find values of parameters of a PSTL formula from a given model
- ▶ Why?
 - ▶ Good to know “as-is” properties of the model
 - ▶ Finds worst-case behaviors of the model
 - ▶ Discriminates between regular and anomalous behaviours

Specification Templates using PSTL



In previous lecture, a, c, T, τ, δ were some fixed values, here they represent parameters

Step:

$$\text{step}(y) := y(t + \delta) - y(t) > a$$

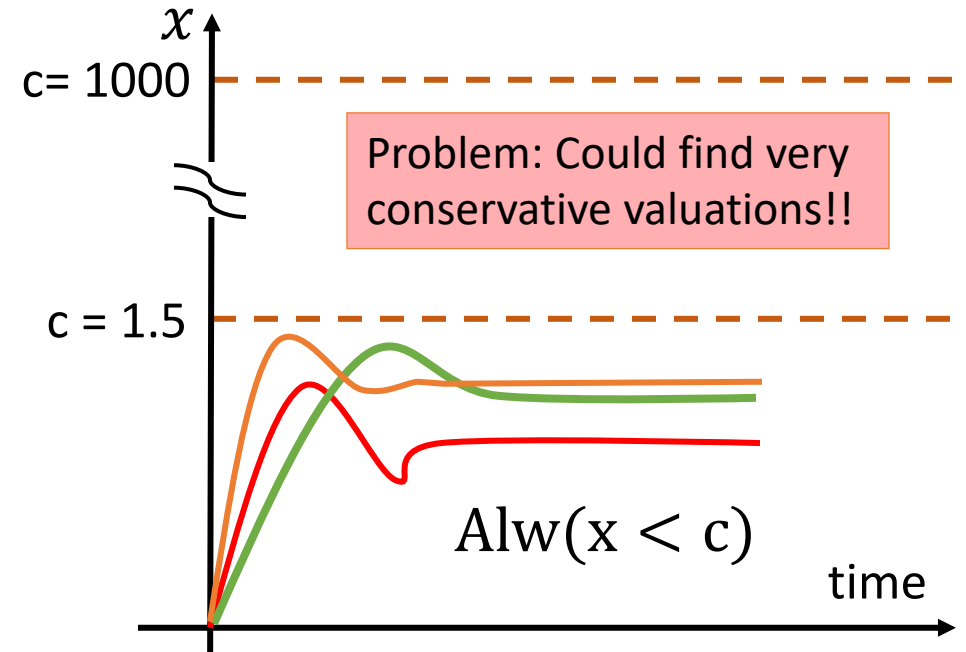
Overshoot:

$$\text{alw}_{[0, T]}(\text{step}(x_{\text{ref}}) \Rightarrow \text{alw}_{[0, \tau]}(x(t) - x_{\text{ref}}(t) < c))$$

Parameter inference for PSTL

- ▶ Given:
 - ▶ PSTL formula $\varphi(\mathbf{p})$, $[\mathbf{p} = (p_1, p_2, \dots, p_m)]$
 - ▶ Traces x_1, \dots, x_n
- ▶ Find:
 - ▶ ~~Valuation~~ $v(\mathbf{p})$ such that: $\forall i : x_i \models \varphi(v(\mathbf{p}))$
 δ -tight valuation
 - and $\exists i : x_i \not\models \varphi(v(\mathbf{p}) \pm \delta)$:
i.e. small perturbation in $v(\mathbf{p})$ makes some trace not satisfy formula

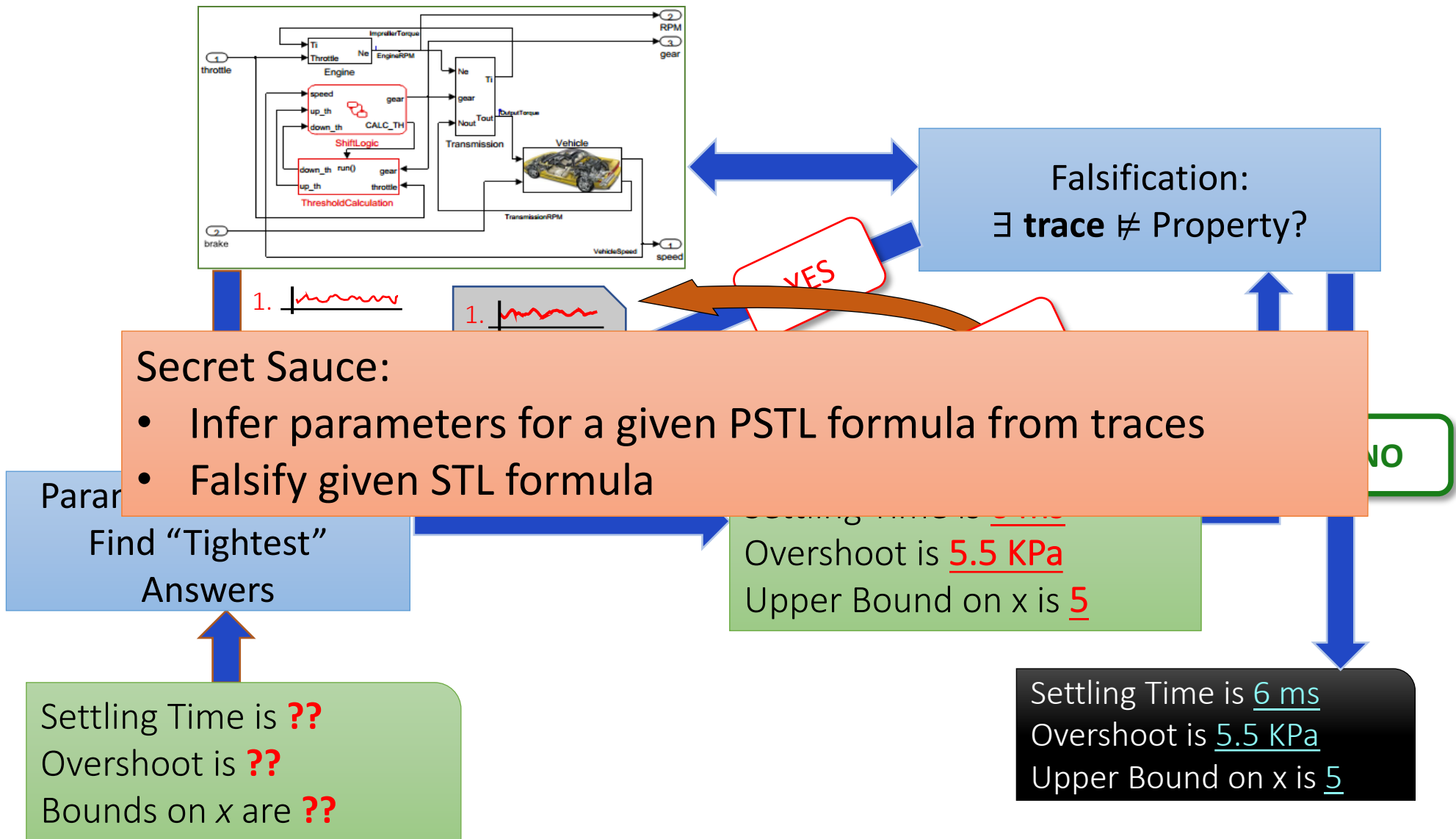
Finding δ -tight valuations hard in general, but **efficient** for **Monotonic PSTL**



formula sat for given valuation \Rightarrow
 \forall greater (or lesser) valuations sat

Binary search on parameter space

Specification Mining

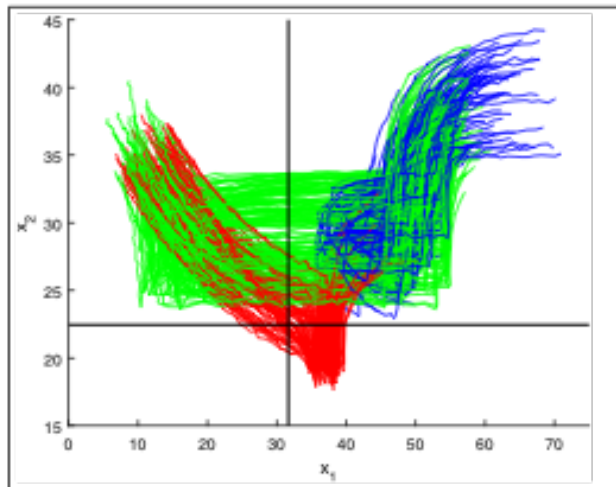


Learning STL classifiers

Goal: Given sets of good and bad trajectories (or generative models), learn STL properties that can separate the two behaviours (a STL classifier)

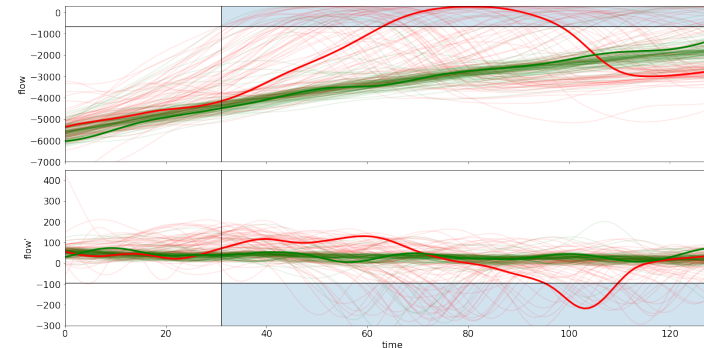
Idea: for a fixed template formula, learn optimally separating parameters by **Bayesian Optimisation**.

$$\phi = ((x_2 > 22.46) \mathcal{U}_{[49,287]} (x_1 \leq 31.65))$$



Maritime surveillance

$$\phi = \mathcal{F}_{[31,130]}((flow \geq -670) \vee (flow' \leq -94))$$



Light entrainment of biological oscillator

Idea: explore formula structure by **genetic programming** on syntactic trees

Problem Formulation

A supervised two-class classification problem

Given a training set of D_p (good) and D_n (bad) find the best ϕ that better separates the two sets.

Discrimination Function

$$G(\phi) = \frac{\mathbb{E}(R_\phi | \vec{X}_p) - \mathbb{E}(R_\phi | \vec{X}_n)}{\sigma(R_\phi | \vec{X}_p) + \sigma(R_\phi | \vec{X}_n)}.$$

Observation: only statistical and noisy evaluations of $G(\phi)$

Goal: maximize $G(\phi)$

ROGE – RObustness GEnetic Algorithm

It is a bi-level optimization algorithm. A **Genetic algorithm** to learn the structure and a **Bayesian optimization algorithm** to learn the parameters.

Require: $\mathcal{D}_p, \mathcal{D}_n, \mathbb{K}, Ne, Ng, \alpha, s$

```
1:  $gen \leftarrow \text{GENERATEINITIALFORMULAE}(Ne, s)$ 
2:  $gen_{\Theta} \leftarrow \text{LEARNINGPARAMETERS}(gen, G, \mathbb{K})$ 
3: for  $i = 1 \dots Ng$  do
4:    $subg_{\Theta} \leftarrow \text{SAMPLE}(gen_{\Theta}, F)$ 
5:    $newg \leftarrow \text{EVOLVE}(subg_{\Theta}, \alpha)$ 
6:    $newg_{\Theta} \leftarrow \text{LEARNINGPARAMETERS}(newg, G, \mathbb{K})$ 
7:    $gen_{\Theta} \leftarrow \text{SAMPLE}(newg_{\Theta} \cup gen_{\Theta}, F)$ 
8: end for
9: return  $gen_{\Theta}$ 
```

$$\phi_{best} = \operatorname{argmax}_{\phi_{\theta} \in gen_{\Theta}} (G(\phi_{\theta}))$$

Learning the Parameters

Problem

Given a PSTL formula ϕ , a parameter space \mathbb{K} , find Θ^* that maximises the discrimination function $G(\phi_{\Theta^*})$.



Methodology

1. Sample $\{(\theta_{(i)}, y_{(i)}), i = 1, \dots, n\}$
2. Emulate (**GP Regression**): $G[R_{\phi}] \sim \text{GP}(\mu, k)$
3. Optimize the emulation via **GP-UCB algorithm**, new $\theta_{(n+1)}$

$\exists \delta$ s.t. $\mathbb{E}(R_{\phi_{\Theta^*}} | \vec{X}_p) > \delta$ and $\mathbb{E}(R_{\phi_{\Theta^*}} | \vec{X}_n) \leq \delta$

Translation. $(\vec{x} - \delta) \Rightarrow \mathbb{E}(R_{\phi_{\Theta^*}} | \vec{X}_p) > 0$ and $\mathbb{E}(R_{\phi_{\Theta^*}} | \vec{X}_n) \leq 0$

Learning the Structure

Problem

Given a set of PSTL formulas gen , find the best ϕ such that ϕ_{Θ} maximises the discrimination function $G(\phi_{\Theta})$.



Methodology

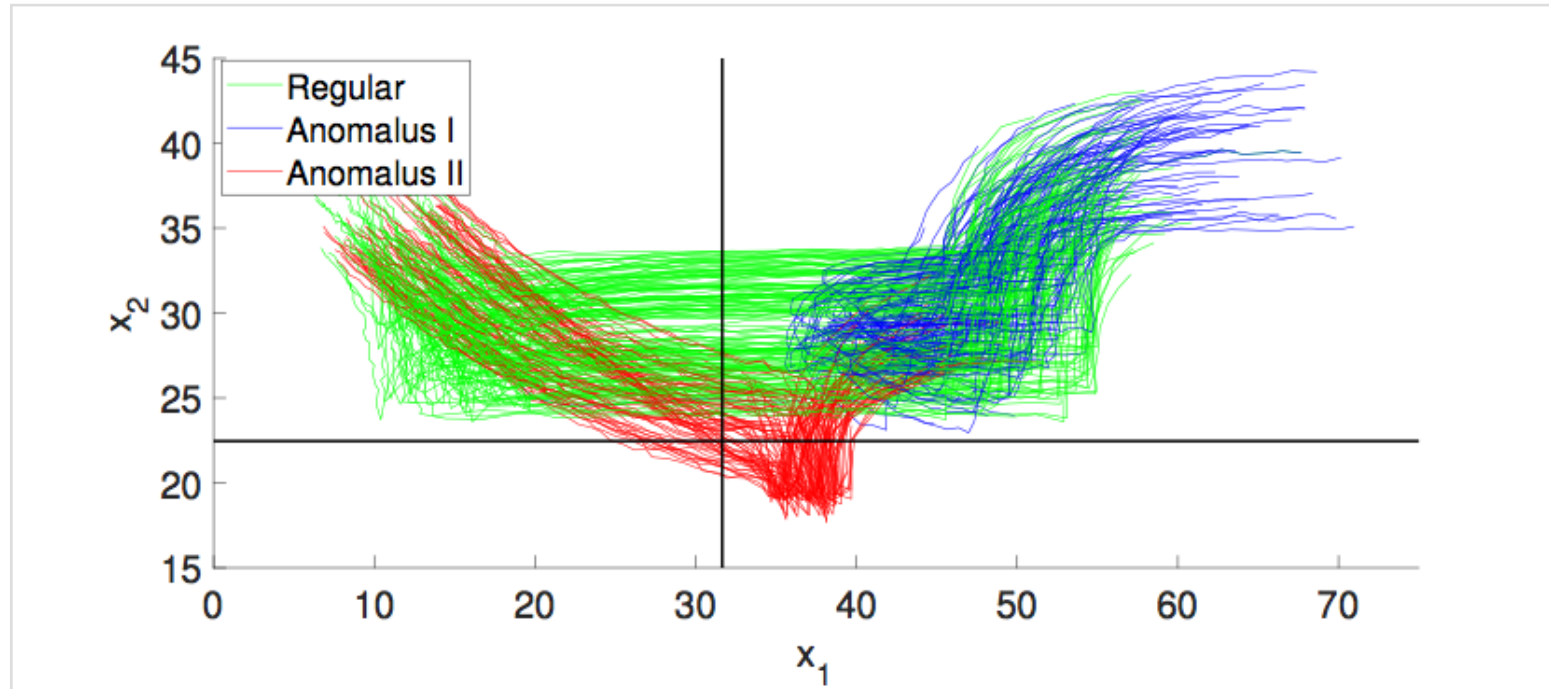
1. **GENERATEINITIALFORMULAE**: $gen = \{\phi_1, \dots, \phi_{N_e}\}$
2. **SAMPLE**(gen_{Θ}, F) = $subg_{\Theta}$, $N_e/2$ formulae, $F(\phi) = G(\phi) - S(\phi)$
3. **EVOLVE**($subg_{\Theta}, \alpha$) = $newg_{\Theta}$, based on two genetic operators, a **recombination** and a **mutation** operator.

Regularization

Formula size penalty $S(\phi)$ and complexity of initial population.

Maritime Surveillance

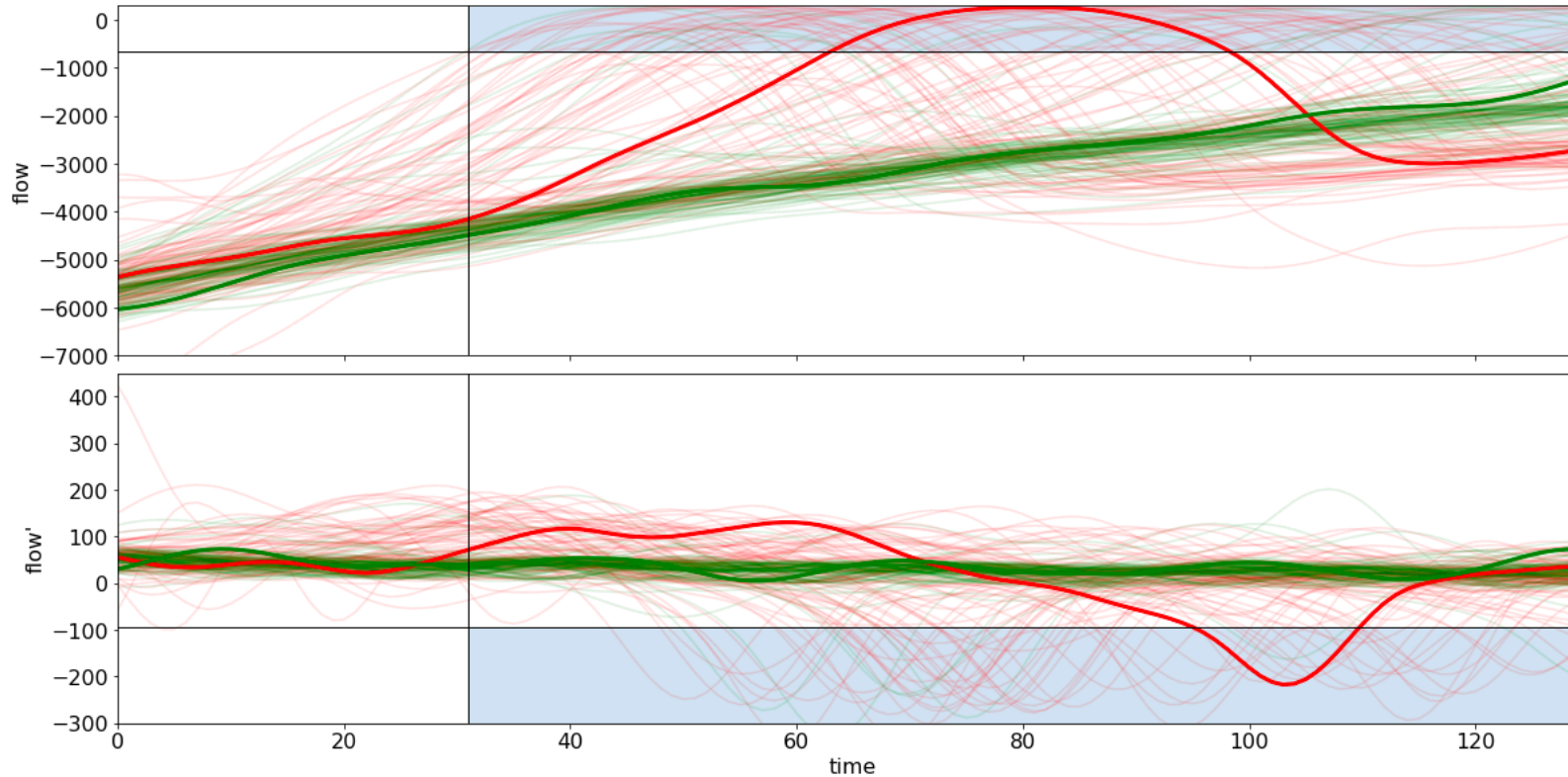
Synthetic dataset of naval surveillance of 2-dimensional coordinates traces of vessels behaviours.



$$\phi_{ROGE} = ((x_2 > 22.46) \mathcal{U}_{[49,287]} (x_1 \leq 31.65))$$

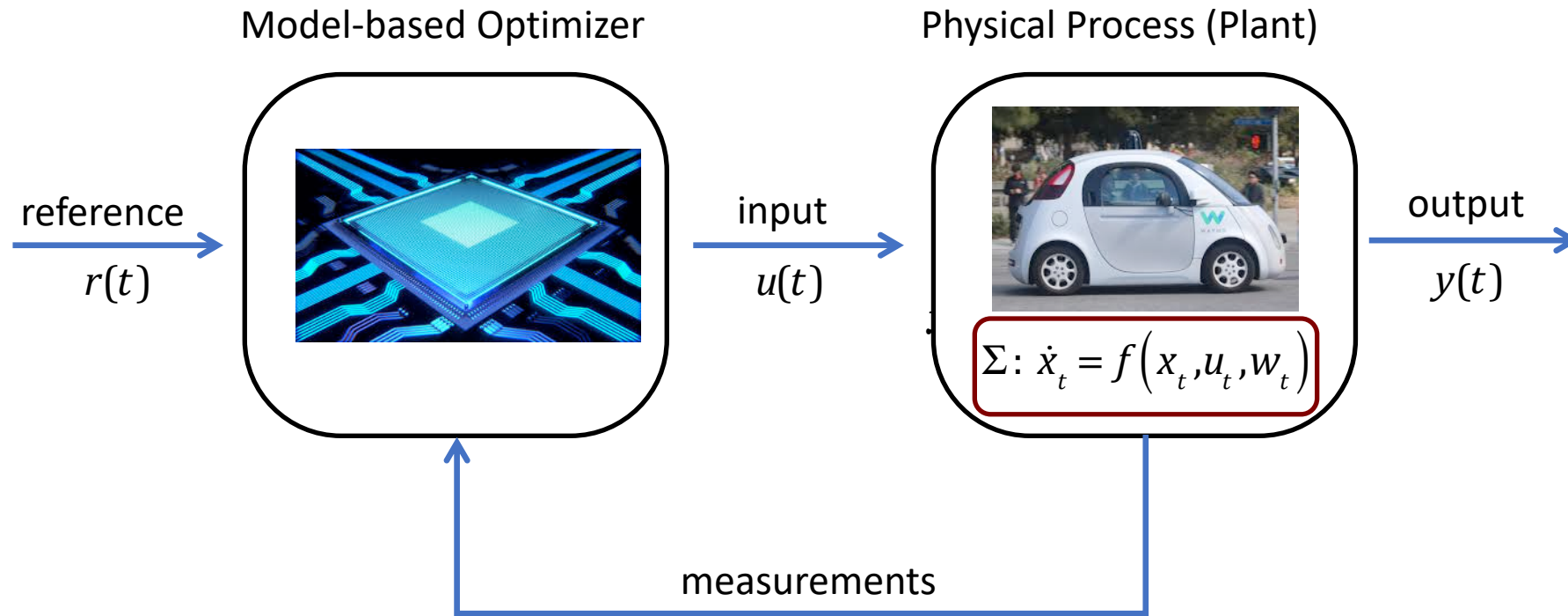
Ineffective Inspiratory Effort (IIE)

The dataset consists of 2-dim traces of flow and its derivative, flow'.



$$\phi = \mathcal{F}_{[31,130]}((flow \geq -670) \vee (flow' \leq -94))|$$

Control Synthesis with STL



The idea is to use the dynamical model of the process to predict its future evolution and optimize consequently the control input signal

Bibliography

Falsification:

- ▶ Silveti S., Policriti A., Bortolussi L. (2017) *An Active Learning Approach to the Falsification of Black Box Cyber-Physical Systems*. IFM 2017. LNCS, vol 10510. Springer, Cham.
- ▶ Several excellent papers on the first development of falsification technology can be found on the web-site of S-TaLiRo : <https://sites.google.com/a/asu.edu/s-taliro/references>
- ▶ Jyotirmoy Deshmukh, Marko Horvat, Xiaoqing Jin, Rupak Majumdar, and Vinayak S. Prabhu. 2017. Testing Cyber-Physical Systems through Bayesian Optimization. *ACM Trans. Embed. Comput. Syst.* 16, 5s, Article 170 (September 2017)
- ▶ Deshmukh, Jyotirmoy, Xiaoqing Jin, James Kapinski, and Oded Maler. Stochastic Local Search for Falsification of Hybrid Systems. In *International Symposium on Automated Technology for Verification and Analysis*, pp. 500-517.

Parameter Synthesis:

- ▶ Ezio Bartocci, Luca Bortolussi, Laura Nenzi, Guido Sanguinetti, System design of stochastic models using robustness of temporal properties. *Theor. Comput. Sci.* 587: 3-25 (2015)
- ▶ Bortolussi L., Silveti S. (2018) *Bayesian Statistical Parameter Synthesis for Linear Temporal Properties of Stochastic Models*. TACAS 2018. LNCS, vol 10806. Springer, Cham

Mining Requirements:

- ▶ Jin, Deshmukh et al. Mining Requirements from Closed-loop Control Models (HSCC '13, IEEE Trans. On Computer Aided Design '15)
- ▶ Nenzi L., Silveti S., Bartocci E., Bortolussi L. (2018) *A Robust Genetic Algorithm for Learning Temporal Specifications from Data*. QEST 2018. LNCS, vol 11024. Springer, Cham.