# An Active Learning Reliability Method for Systems with Partially Defined Performance Functions

Jonathan Sadeghi Romain Mueller John Redford

Five AI Ltd.

jon.sadeghi@five.ai

## Hierarchical Gaussian Process: enables efficient calculation of failure probability for partially defined performance functions

## Summary


- **Partially defined system performance** introduces discontinuous behaviour $\implies$ poor performance of Gaussian Process active learning reliability algorithms which assume continuous performance
- Introduce a **Hierarchical Gaussian Process** composed of two GPs to model partially defined performance.
- Active learning with Hierarchical Gaussian Process outputs better calibrated probabilities $\implies$ obtains **higher accuracy** and **superior convergence**

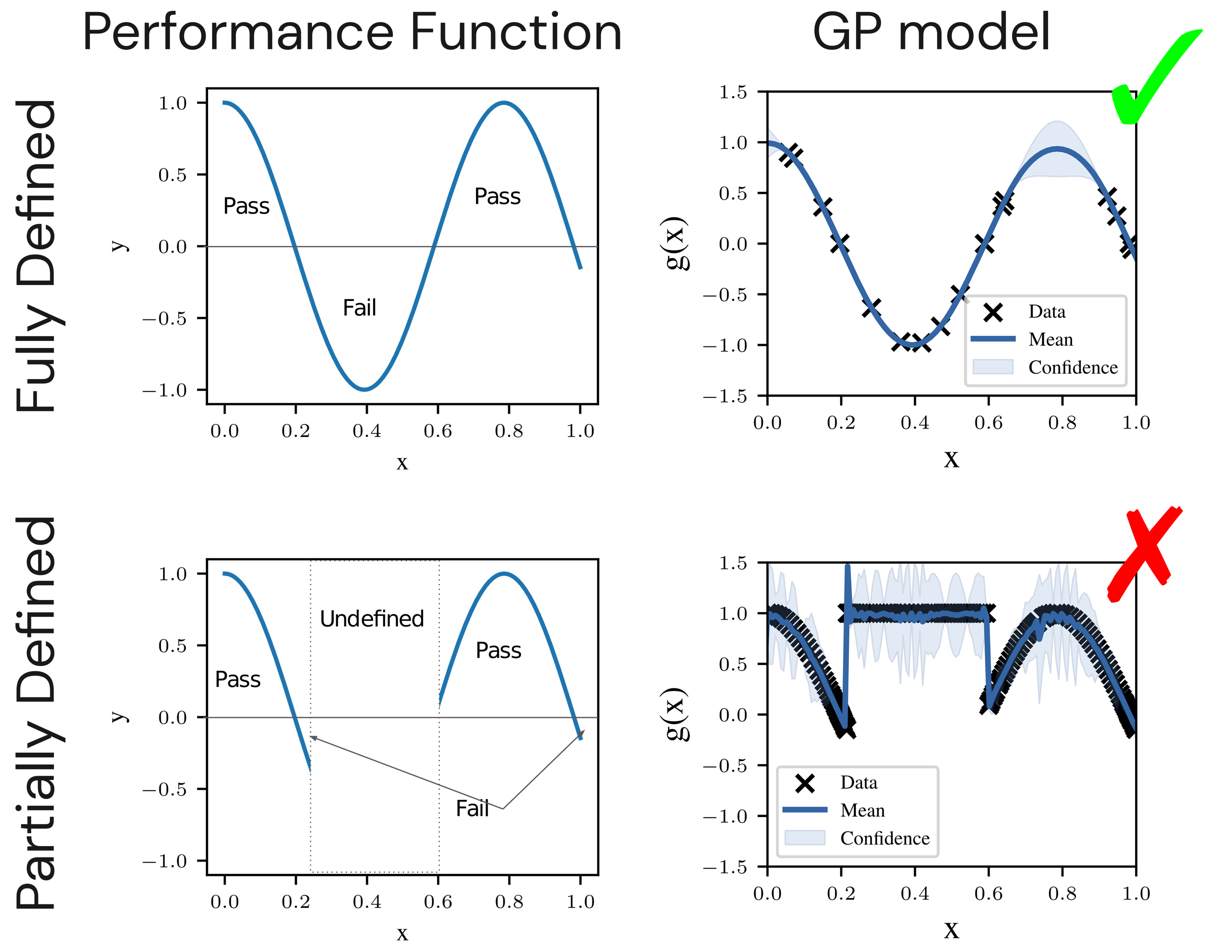




Figure 1: AK-MCS applied to a masked partially defined performance function and a fully defined performance function.

## Adaptive Kriging Monte Carlo Simulation

- Aim to calculate $p_f = \mathbb{E}_{p(\mathbf{x})}\mathbb{1}[g(\mathbf{x}) < 0]$, where $p(\mathbf{x})$ is given distribution of environmental variables, and $g(\mathbf{x})$ is system performance.
- GP regression for $g(\mathbf{x})$ and iteratively query system on most likely misclassified points with AK-MCS.

## Hierarchical Gaussian Process Model

- Introduce a hierarchical Gaussian Process model of the system performance in active learning loop:
  1. Classification GP to predict if y is NaN-valued
  2. Regression GP to predict numerical value of y
- Failure and misclassification probabilities can be readily computed:

$$p(y_*|\mathbf{x}_*, \mathcal{D}) = \begin{cases} \underbrace{p(y_*|\mathbf{x}_*, \mathcal{D}, y_* \neq \text{NaN})}_{\text{Regression GP}}\, p(y_* \neq \text{NaN}|\mathbf{x}_*, \mathcal{D}) & \text{if } y_* \neq \text{NaN}, \\ \underbrace{p(y_* = \text{NaN}|\mathbf{x}_*, \mathcal{D})}_{\text{Classification GP}} & \text{if } y_* = \text{NaN}, \end{cases}$$

$$p_f \approx \mathbb{E}_{p(\mathbf{x}_*)}\mathbb{1}[p(y_* < 0, y_* \neq \text{NaN}|\mathbf{x}_*, \mathcal{D}) > 0.5]$$

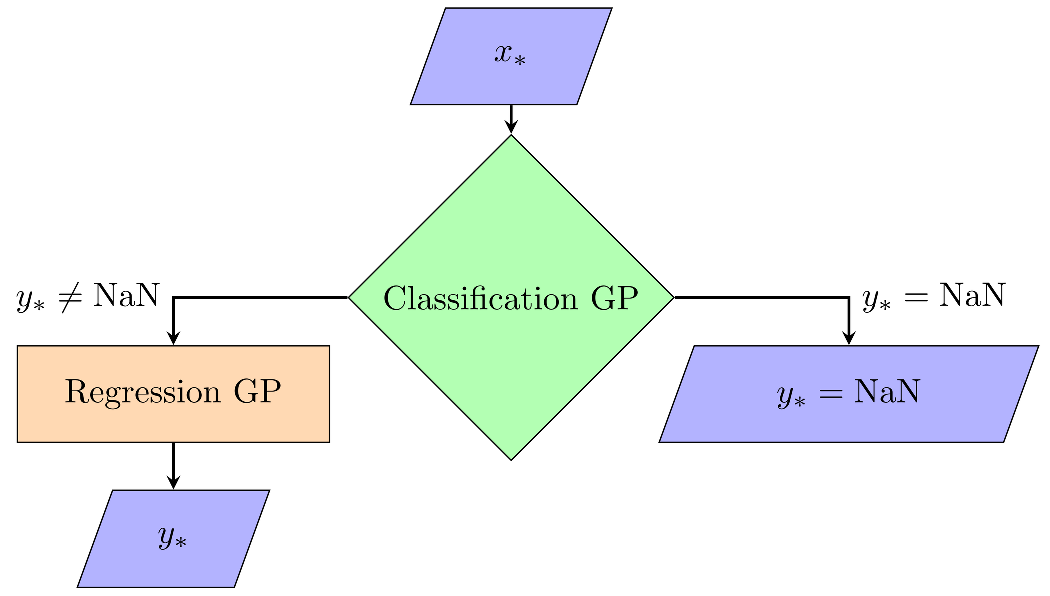



Figure 2: A Hierarchical Gaussian Process Model.

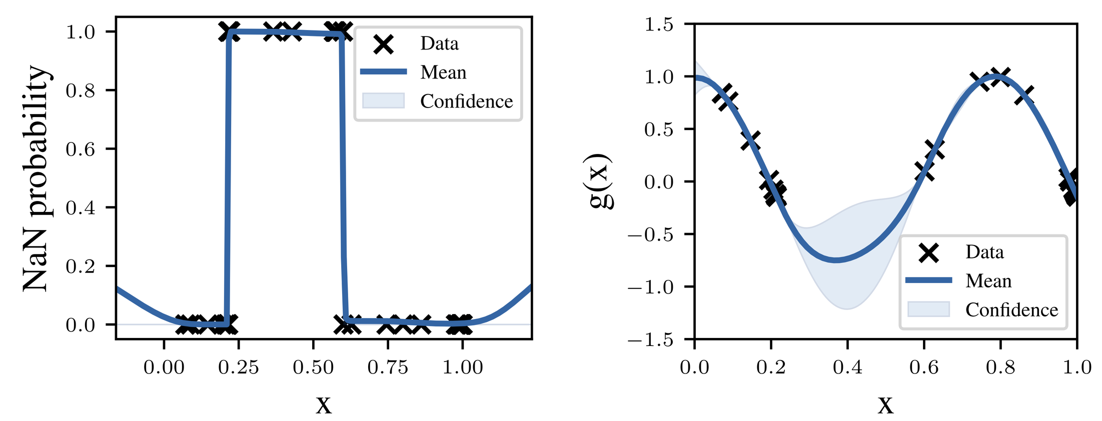
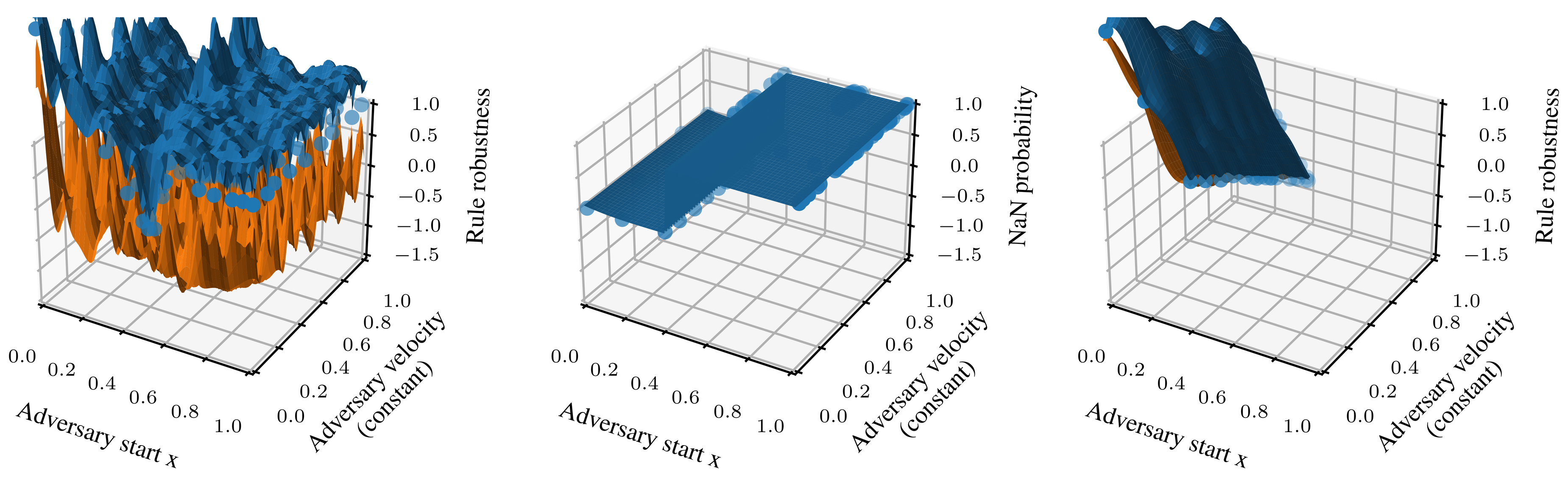
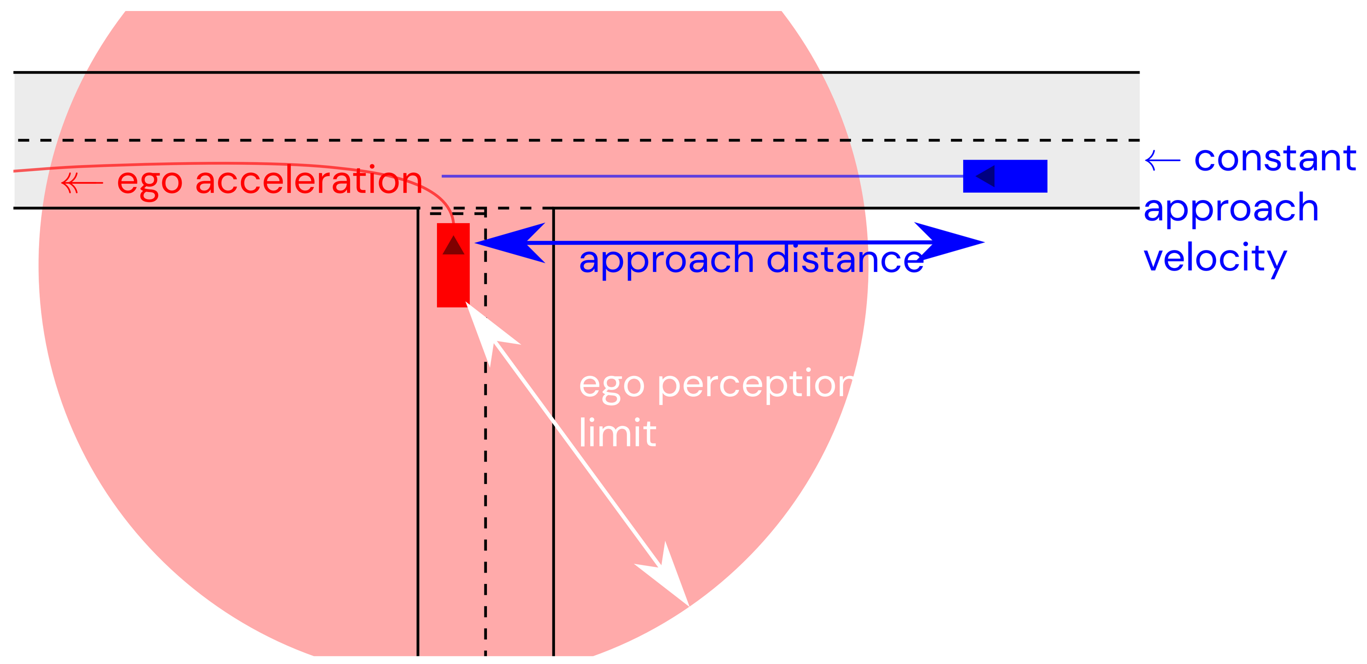



Figure 3: AK-MCS with hierarchical GP on masked cosine function.




(a) Masked GP (b) Hierarchical GP

Figure 4: Trained Models. Hierarchical model produces much **better calibrated probabilities**.




## Autonomous Driving Example Results

- **Task**: Autonomous Driving T-junction merge scenario with vehicle trajectories parameterised by random variables
- **Performance**:
  - longitudinal distance in lane
  - **undefined** if ego doesn't merge

Table 1: Number of iterations was capped at 150 and methods hitting the cap are marked with did not terminate (DNT). Results shown as *mean (standard deviation)* for 5 repeats. Our method **terminates after an appropriate number of iterations**.

| Methodology | $p_f$ | $F_1$score | # Evaluations |
|---|---|---|---|
| Analytic | 0.0382 | N/A | N/A |
| hGP (ours) | 0.038 (0.0019) | 1 (0.0013) | 81 (21) |
| Masked GP | 0.037 (0.0041) | 0.92 (0.063) | DNT |
| GPC | 0.035 (0.0029) | 0.98 (0.0049) | DNT |

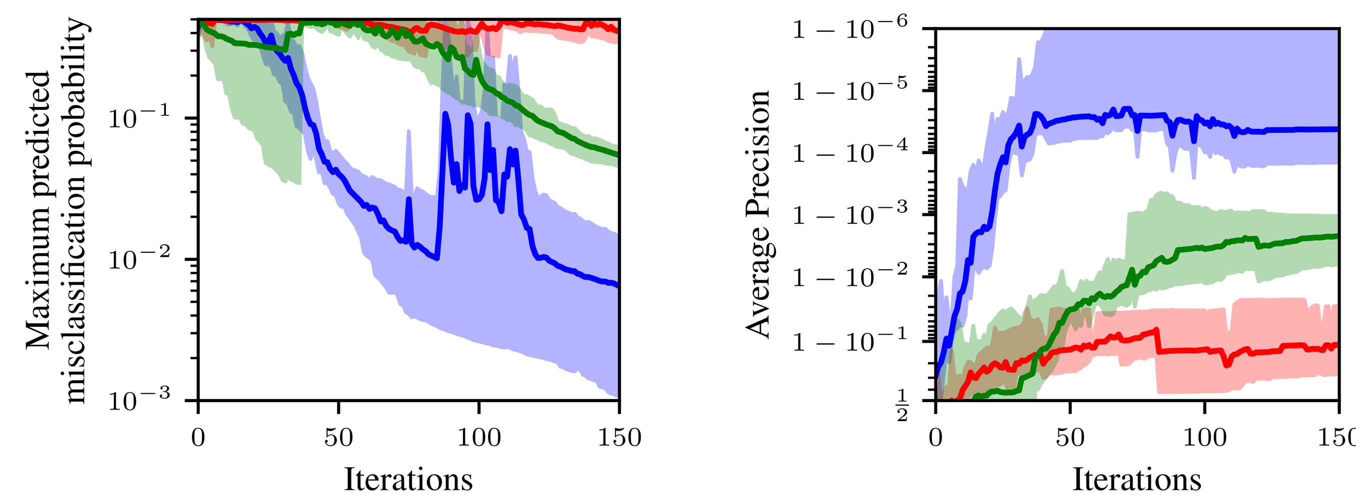



Figure 5: Convergence of $p_f$ and average precision. Baseline models converge erratically, but Hierarchical GP obtained **high average precision** and the predicted misclassification probability decreases quickly.