# Inferring the uncertainty of simulator parameters for Sim2Real and deep RL

Fabio Ramos, 11/08/2019
Joint work with Rafael Possas, Dieter Fox, Carolyn Chen and Yashraj Narang

# WHY TO CARE ABOUT UNCERTAINTY

# BUT WHEN YOU DON'T CARE...

# OUTLINE

1. Characterizing uncertainty
2. Uncertainty at work
3. BayesSIM: Uncertainty for Sim2Real

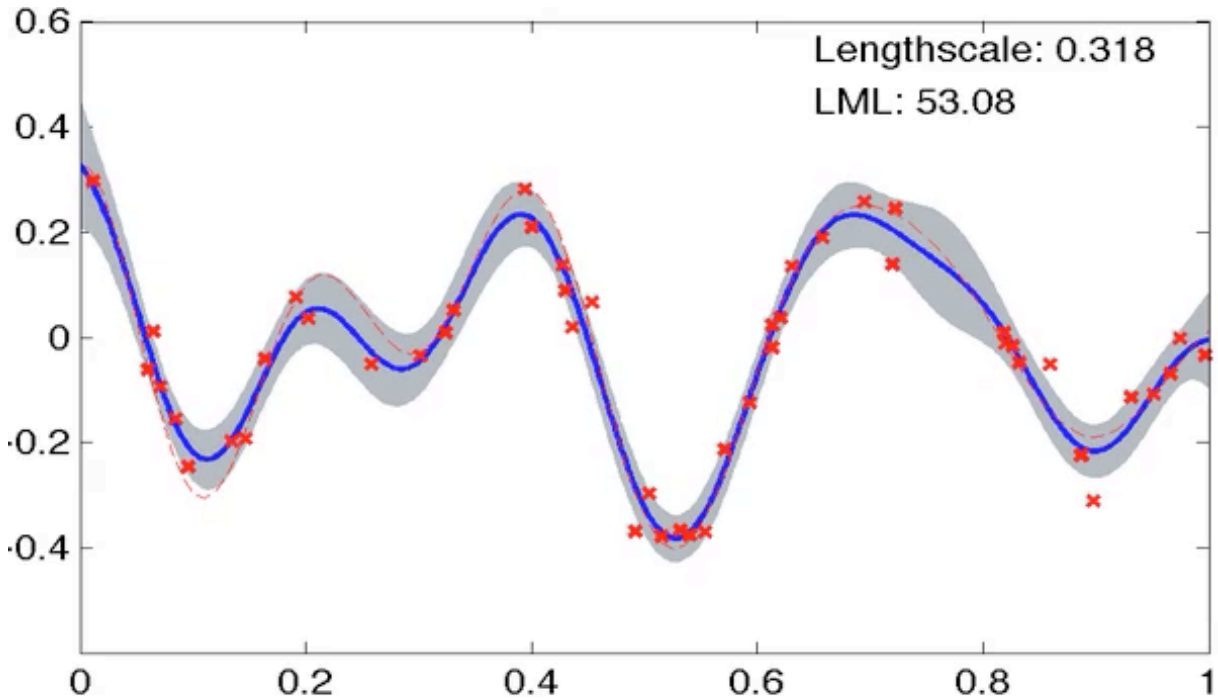
# 1. CHARACTERIZING UNCERTAINTY

# TYPES OF UNCERTAINTY

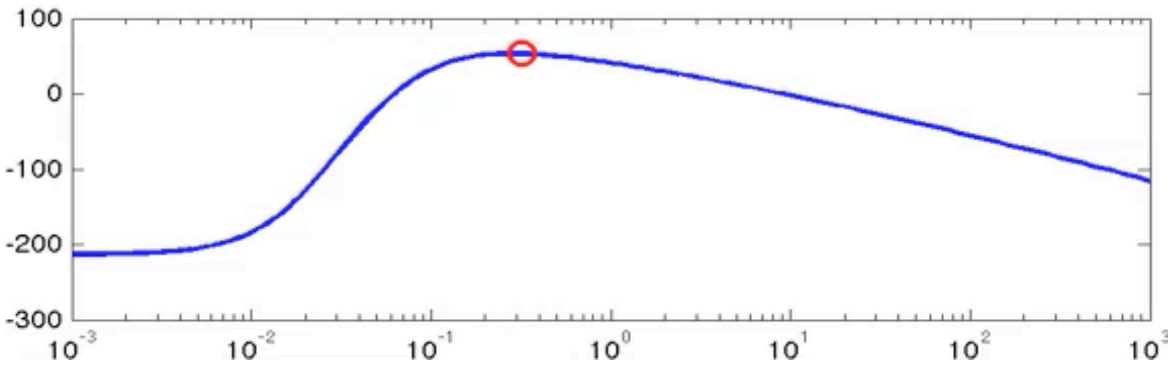

- Epistemic: what we should know but we don’t. For example, model uncertainty, lack of data, simplifications.
- Aleatoric: natural stochasticity of the process. For example, when you run the experiment multiple times, and have a different answer every time.
- Measurement: incapacity to fully observe the variable of interest. For example, noisy sensors.

# UNCERTAINTY IN REGRESSION

Problem




Objective



# 2. UNCERTAINTY AT WORK

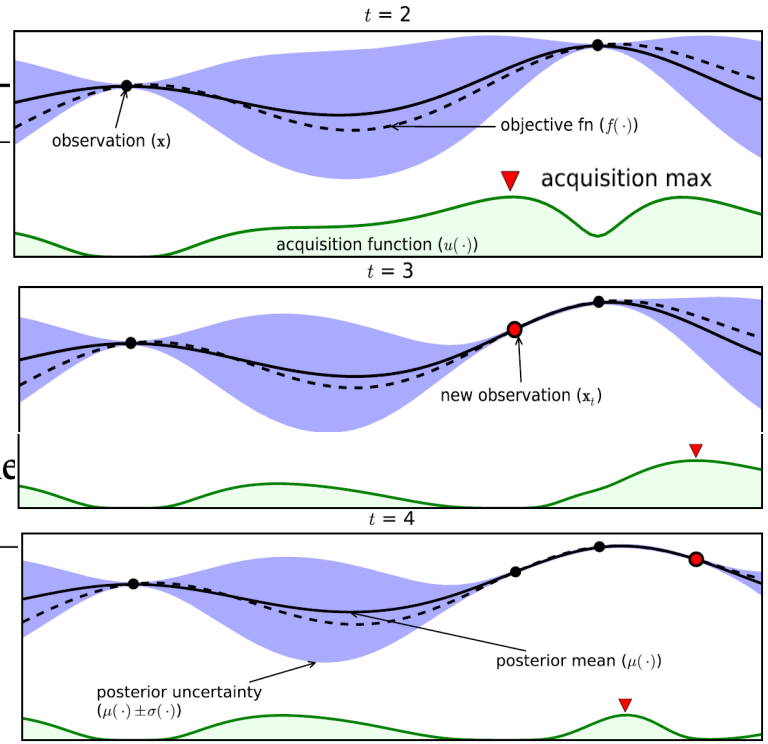
# EPISTEMIC AT WORK

## Bayesian Optimisation

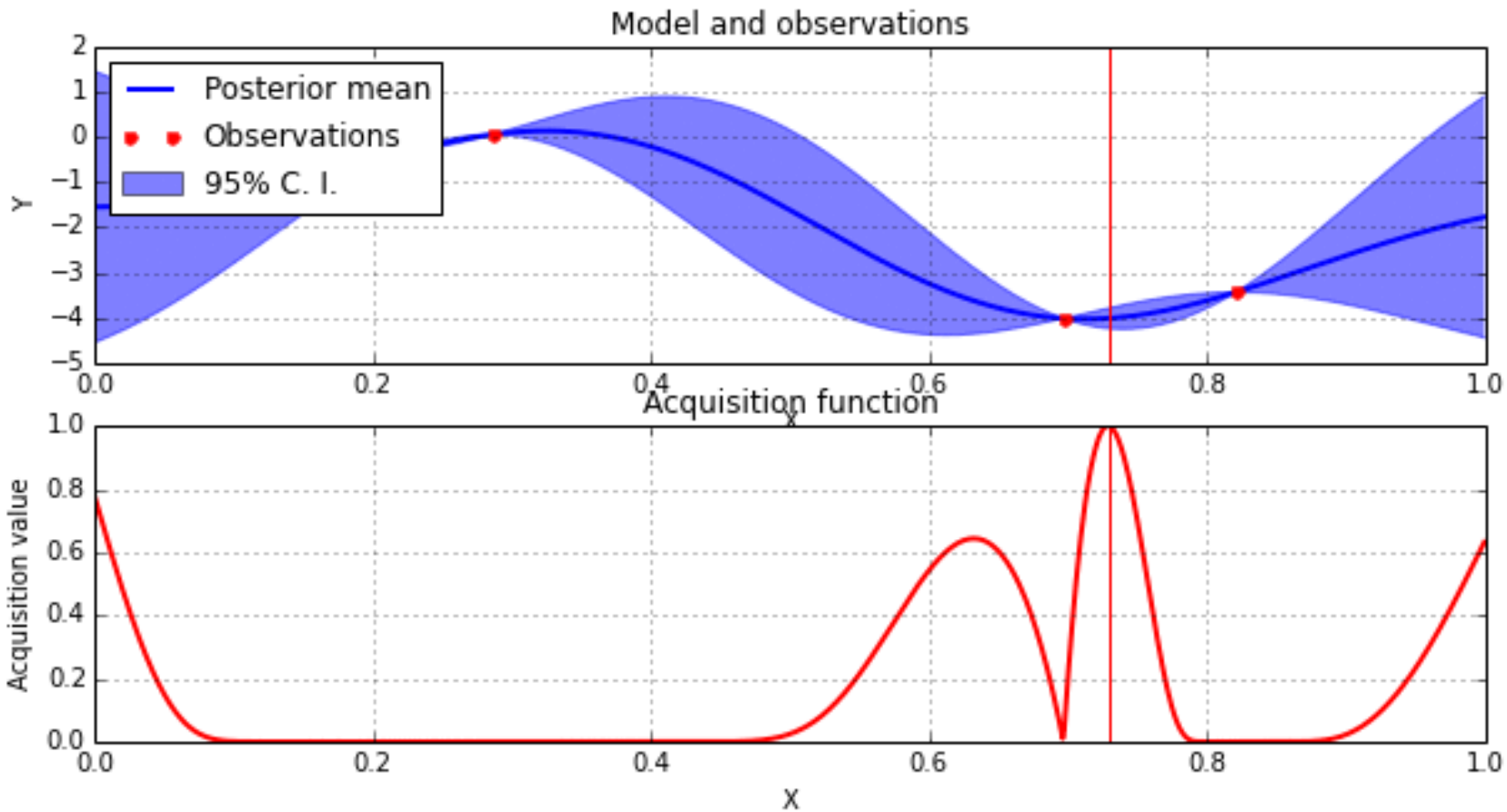
Goal: Find the maximum of an unknown, noisy and costly to evaluate function $f$.
Idea: Choose next sampling location $x$ by maximising an acquisition function $S$ over the domain of the GP model of the function.

**Algorithm 1** Bayesian Optimisation

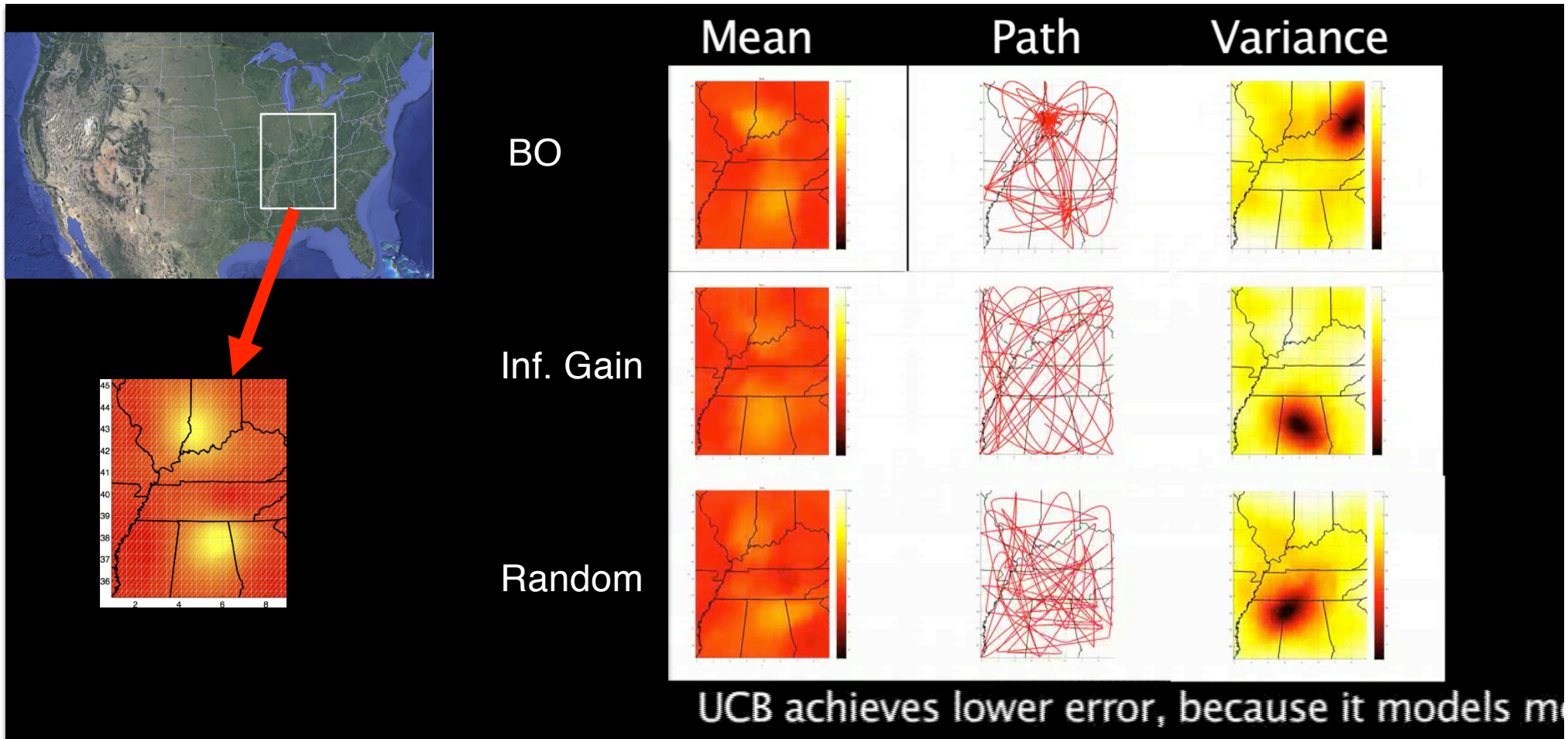
1: $\mathbf{x}_i$: chosen sampling point at iteration $i$.
2: $s$: acquisition function.
3: $f$: unknown function. (terrain roughness)
4: **for** $i = 1, 2, 3, \ldots$ **do**
5: Find $\mathbf{x}_i = \arg\max_{\mathbf{x}} s(\mathbf{x})$
6: Acquire a sample from $f$ at location $\mathbf{x}_i$.
7: Update the GP model of $f$ with the new sample
8: **end for**



# BAYESIAN OPTIMISATION



# PLANNING TO IMPROVE PREDICTIONS



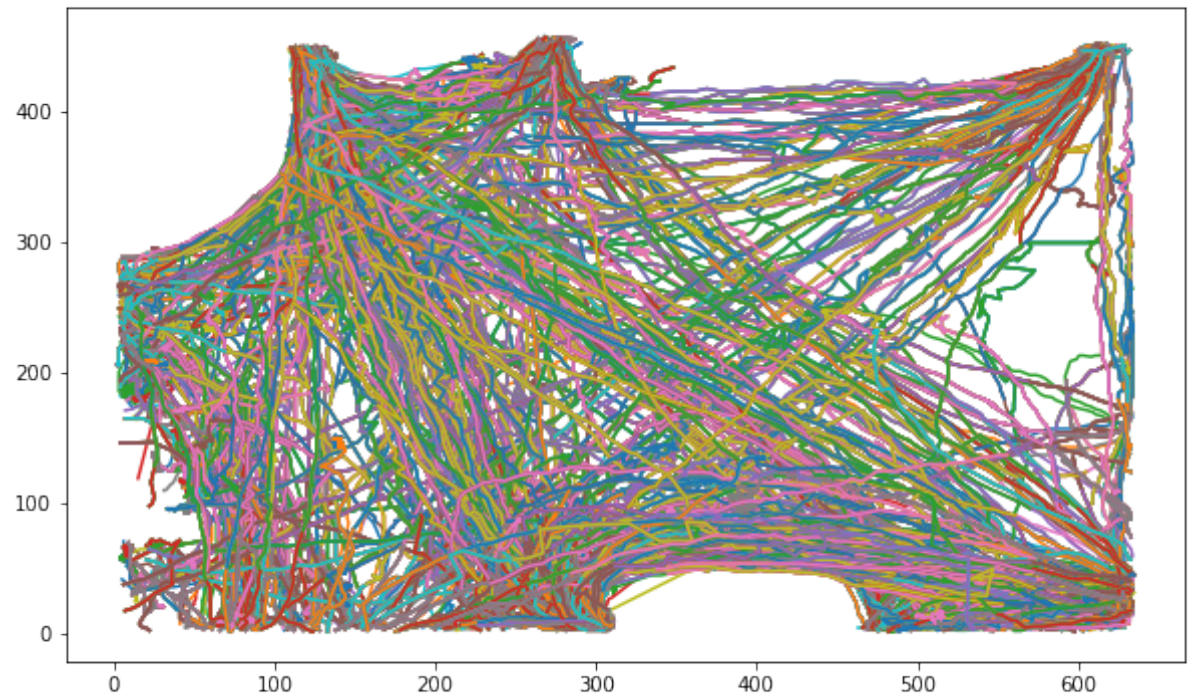

R. Marchant, F. Ramos, Bayesian Optimisation for Informative Continuous Path Planning. In IEEE International Conference on Robotics and Automation (ICRA), 2014.

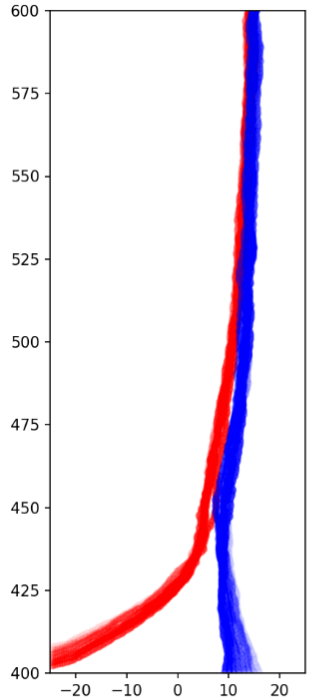
R. Marchant, F. Ramos, S. Sanner, Sequential Bayesian Optimisation for Spatial-Temporal Monitoring. In Uncertainty in Artificial Intelligence (UAI), 2014

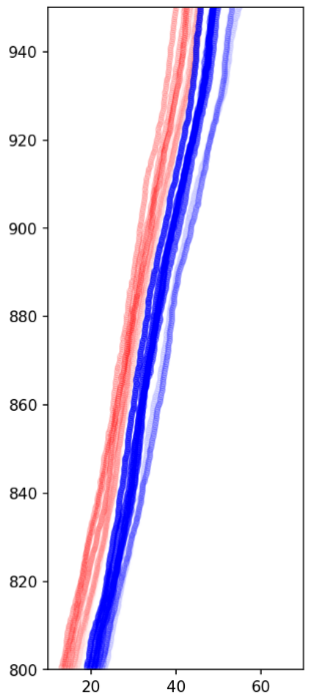
# ALEATORIC AT WORK

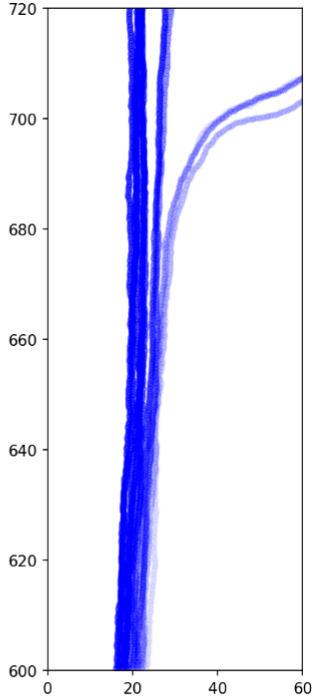
## Learning motion patterns

[Zhi, Ott, Ramos. Kernel Trajectory Maps for Multi-Modal Probabilistic Motion Prediction. CoRL'19]



# KERNEL TRAJECTORY MAPS FOR MULTI-MODAL PROBABILISTIC MOTION PREDICTION

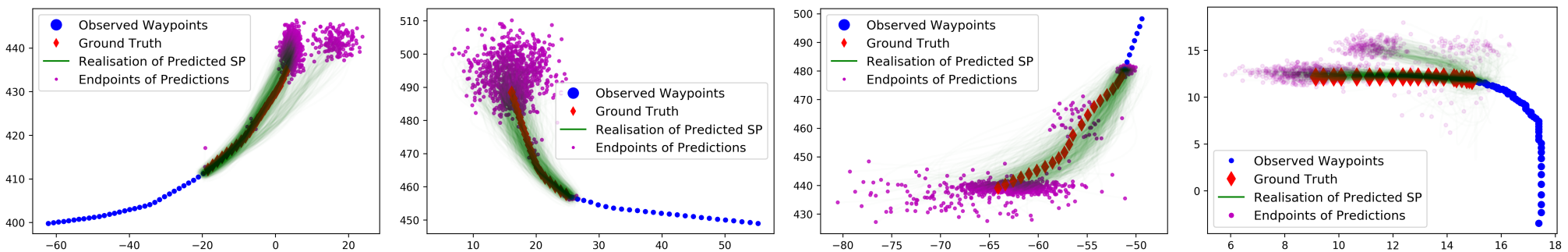
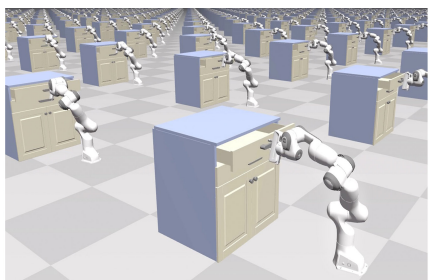



Figure 1: Observed waypoints (blue) and predicted trajectories (green with magenta end-points) sampled from KTM outputs. The ground truth trajectory is indicated in red. The probabilistic and multi-modal nature of KTMs is able to capture the complexity of the motion patterns.

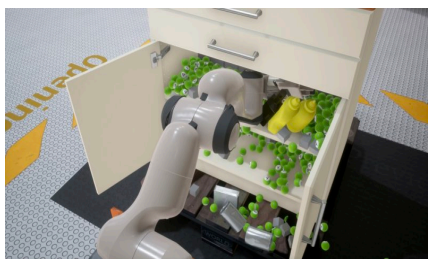
# 3. BAYESSIM: UNCERTAINTY FOR SIM2REAL

# POWERFUL SIMULATORS VS REALITY

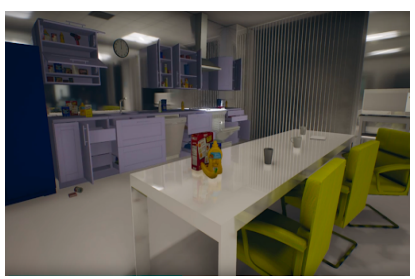
*All simulators are wrong, some are useful*



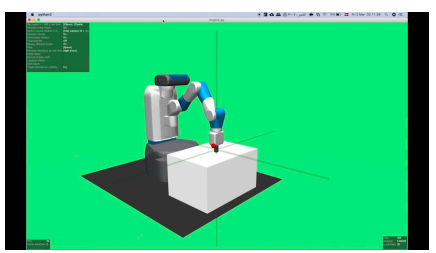
Isaac Gym



PhysX

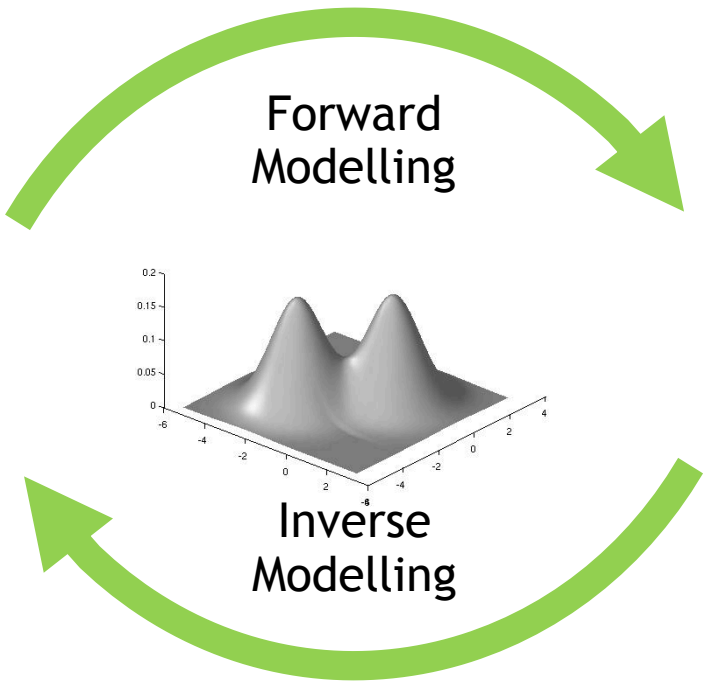


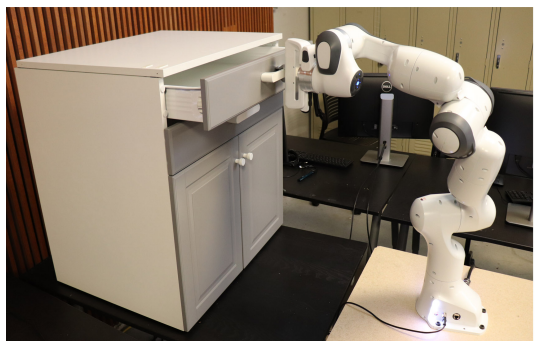
Isaac Sim



Mujoco

Simulators



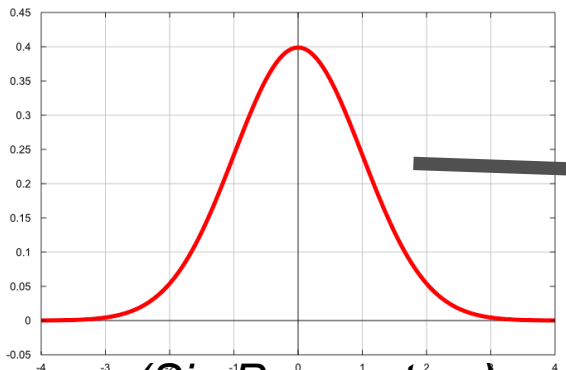
Reality

**How to estimate the uncertainty in simulators?**

# BayesSim

*All simulators are wrong, some are useful*

[Ramos, Possas, Fox. BayesSim: adaptive domain randomization via probabilistic inference for robotics simulators. RSS'19]



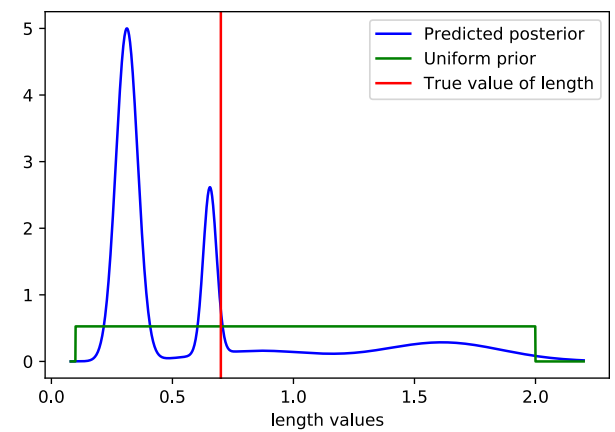

*p(SimParameters)*

Prior

Simulator 4

RealData

*BayesSim*




*p(SimParameters | RealData)*

Posterior

# LIKELIHOOD-FREE INFERENCE

## From generative models to Bayesian inference

Attempts to estimate $p(\boldsymbol{\theta}|\boldsymbol{x} = \boldsymbol{x}^r) \propto p(\boldsymbol{\theta})p(\boldsymbol{x} = \boldsymbol{x}^r|\boldsymbol{\theta})$ , where $\theta$ are simulator parameters.

We treat simulators as black-box generative models, $g(\theta) = \boldsymbol{x}$.

However, the likelihood function, $p(\boldsymbol{x} = \boldsymbol{x}^r|\theta)$, is not available as the simulator is a black box.

Instead, we perform inference by computing

$$\hat{p}(\boldsymbol{\theta}|\mathbf{x} = \mathbf{x}^r) \propto \frac{p(\boldsymbol{\theta})}{\tilde{p}(\boldsymbol{\theta})} q_\phi(\boldsymbol{\theta}|\mathbf{x} = \mathbf{x}^r)$$

where $q_\phi(\boldsymbol{\theta}|\boldsymbol{x} = \boldsymbol{x}^r)$ is a mixture of Gaussians given by $q_\phi(\boldsymbol{\theta}|\mathbf{x}) = \sum_k \alpha_k \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$

$p(\boldsymbol{\theta})$ and $\tilde{p}(\boldsymbol{\theta})$ are the prior and proposal priors respectively.

# MIXTURE DENSITY WITH RFF

## We learn feature maps for the Gaussian mixture model

The mixture model approximating the posterior,

$$q_\phi(\boldsymbol{\theta}|\mathbf{x}) = \sum_k \alpha_k \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

is parametrized by

$$\boldsymbol{\alpha} = \text{softmax}(\mathbf{W}_{\boldsymbol{\alpha}} \Phi(\mathbf{x}) + \mathbf{b}_{\boldsymbol{\alpha}})$$

$$\boldsymbol{\mu}_k = \mathbf{W}_{\boldsymbol{\mu}_k} \Phi(\mathbf{x}) + \mathbf{b}_{\boldsymbol{\mu}_k}$$

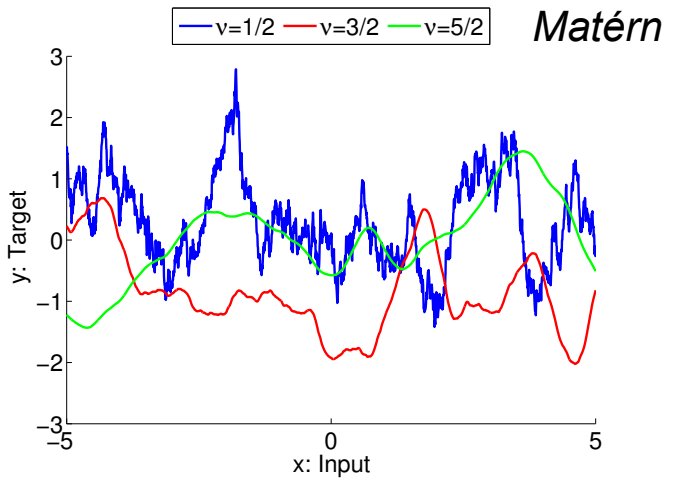
$$\text{diag}(\boldsymbol{\Sigma}_k) = \text{mELU}(\mathbf{W}_{\boldsymbol{\Sigma}_k} \Phi(\mathbf{x}) + \mathbf{b}_{\boldsymbol{\Sigma}_k})$$

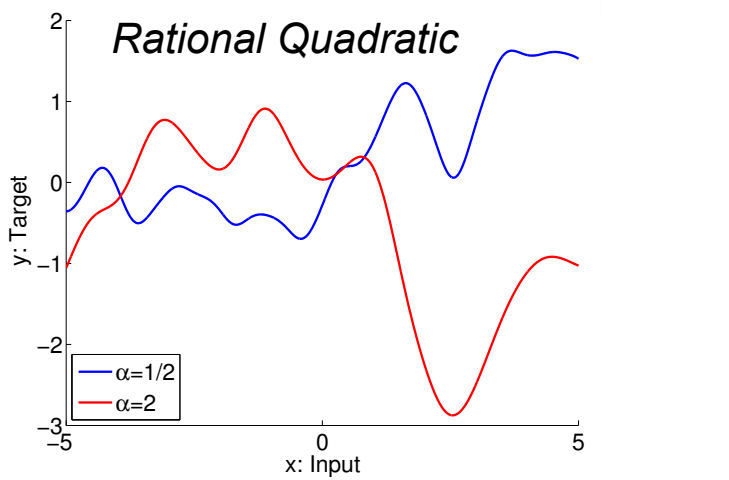
where $\Phi(\boldsymbol{x})$ are random Fourier features.

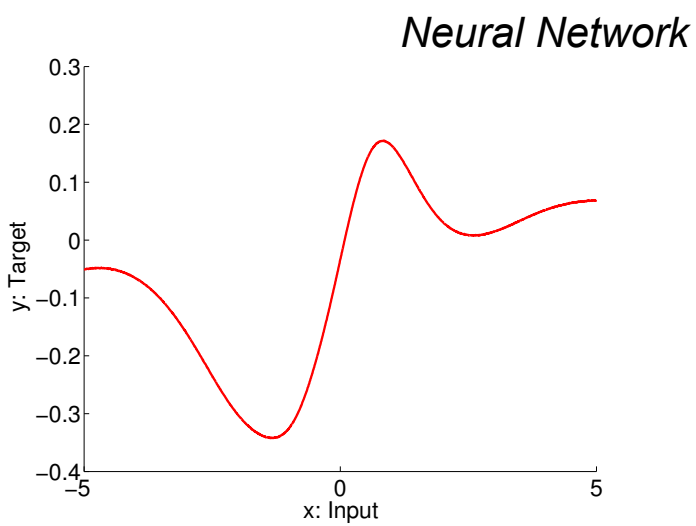
We compute these features and learn all other parameters of the mixture:

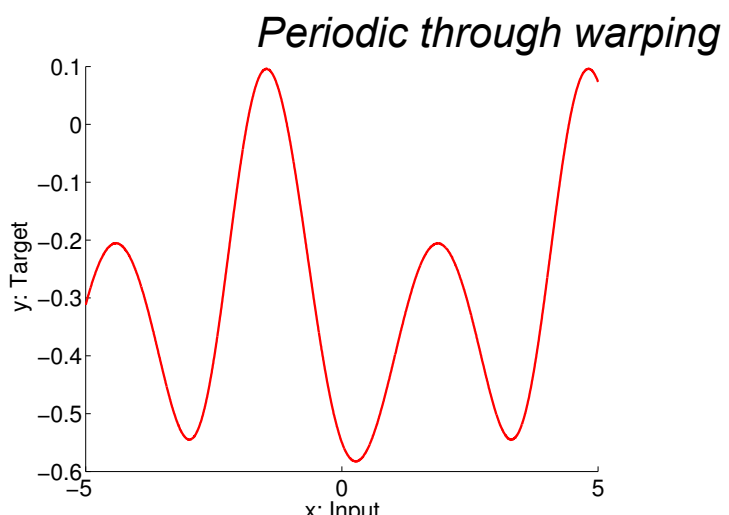
$$\phi = (\mathbf{W}_{\boldsymbol{\alpha}}, \mathbf{b}_{\boldsymbol{\alpha}}, \{\mathbf{W}_{\boldsymbol{\mu}_k}, \mathbf{b}_{\boldsymbol{\mu}_k}, \mathbf{W}_{\boldsymbol{\Sigma}_k}, \mathbf{b}_{\boldsymbol{\Sigma}_k}\}_{k=1}^K)$$

# HILBERT SPACES AS PRIORS



# RANDOM FOURIER FEATURES

## More consistent than Neural Networks for this problem

Following Bochner's Theorem, a stationary kernel, defining a Hilbert space, can be represented in terms of its Fourier transform,

$$k(\boldsymbol{\tau}) = \int \mathcal{K}(\boldsymbol{\omega}) e^{-i\boldsymbol{\omega}\cdot\boldsymbol{\tau}} d\boldsymbol{\omega}$$

where $\tau = \boldsymbol{x} - \boldsymbol{x}'$ and $\mathcal{K}(\boldsymbol{\omega})$ is its spectral representation. This can be seen as an expectation that can approximated as,

$$k(\boldsymbol{\tau}) \approx \frac{1}{N} \sum_{n=1}^{N} (e^{-i\boldsymbol{\omega}_n \mathbf{x}})(e^{-i\boldsymbol{\omega}_n \mathbf{x}'}) = \Phi(\mathbf{x})^T \Phi(\mathbf{x}')$$

and

$$\Phi(\mathbf{x}) = \frac{1}{\sqrt{N}} [\cos(\boldsymbol{\omega}_1 \mathbf{x} + b_1), \ldots, \cos(\boldsymbol{\omega}_n \mathbf{x} + b_n), \\ -i \cdot \sin(\boldsymbol{\omega}_1 \mathbf{x} + b_1), \ldots, -i \cdot \sin(\boldsymbol{\omega}_n \mathbf{x} + b_n)].$$

# RECOVERING THE FINAL RESULT

When the prior is different from the proposal prior, we need to adjust the posterior as follows,

$$\hat{p}(\boldsymbol{\theta}|\mathbf{x}=\mathbf{x}^r) \propto \frac{q_\phi(\boldsymbol{\theta}|\mathbf{x}^r)}{\tilde{p}(\boldsymbol{\theta})}$$

Assuming a Gaussian prior, the posterior is given by

$$\hat{p}(\boldsymbol{\theta}|\mathbf{x}=\mathbf{x}^r) = \sum_k \alpha'_k \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}'_k, \boldsymbol{\Sigma}'_k)$$

where

$$\begin{aligned}
\boldsymbol{\Sigma}'_k &= \left(\boldsymbol{\Sigma}_k^{-1} - \boldsymbol{\Sigma}_0^{-1}\right)^{-1} \\
\boldsymbol{\mu}'_k &= \boldsymbol{\Sigma}_k^{-1}\left(\boldsymbol{\Sigma}_k^{-1}\boldsymbol{\mu}_k - \boldsymbol{\Sigma}_0^{-1}\boldsymbol{\mu}_0\right) \\
\alpha'_k &= \frac{\alpha_k \exp(-\frac{1}{2}\lambda_k)}{\sum_{k'} \alpha_{k'} \exp(-\frac{1}{2}\lambda_{k'})},
\end{aligned}$$

And the coefficients are:

$$\begin{aligned}
\lambda_k = \log\det\boldsymbol{\Sigma}_k - \log\det\boldsymbol{\Sigma}_0 - \log\det\boldsymbol{\Sigma}'_k + \boldsymbol{\mu}_k^T\boldsymbol{\Sigma}_k^{-1}\boldsymbol{\mu}_k \\
- \boldsymbol{\mu}_0^T\boldsymbol{\Sigma}_0^{-1}\boldsymbol{\mu}_0 - \boldsymbol{\mu}_k'^T\boldsymbol{\Sigma}_k'^{-1}\boldsymbol{\mu}'_k.
\end{aligned}$$

# SUFFICIENT STATISTICS

## A fast low dimensional representation for trajectories

Since trajectories of state and action pairs might be very high dimensional, we reduce the dimensionality using some sufficient statistics. Formally, observations

$$\mathbf{x} = \psi(\mathbf{S}, \mathbf{A})$$

where $\mathbf{S} = \{\mathbf{s}^t\}_{t=1}^T$ are states and $\mathbf{A} = \{\mathbf{a}^t\}_{t=1}^T$ are actions.

We use sufficient statistics that are inspired by other problems in likelihood-free inference such as cross correlation between states and actions, means, and variances,

$$\psi(\mathbf{S}, \mathbf{A}) = (\{\langle \boldsymbol{\tau}_i, \mathbf{A}_j \rangle\}_{i=1,j=1}^{D_s, D_a}, \mathrm{E}[\boldsymbol{\tau}], \mathrm{Var}[\boldsymbol{\tau}])$$

# DOMAIN RANDOMIZATION WITH BAYES-SIM

## RL can be trained in simulation and applied to the real robot

In reinforcement learning, we want to learn a policy $\pi(\boldsymbol{s};\beta)$, parametrized by $\beta$, that optimizes the objective,

$$J(\boldsymbol{\beta}) = \mathbb{E}_{\boldsymbol{\theta}}\left[\mathbb{E}_{\boldsymbol{\eta}}\left[\sum_{t-0}^{T-1}\gamma^{(t)} r(\mathbf{s}_t, \mathbf{a}_t)|\boldsymbol{\beta}\right]\right]$$

where $r(\boldsymbol{s}_t, \boldsymbol{a}_t)$ is the reward of taking action $\boldsymbol{a}_t$ in state $\boldsymbol{s}_t$.

We can use the *posterior* produced by BayesSim to estimate the expectation $E_\theta$.

We can *train* the policy using multiple simulators, each *sampled* from the posterior.

The policy obtained is more robust and likely to work on the real robot.

# CART POLE EXAMPLE

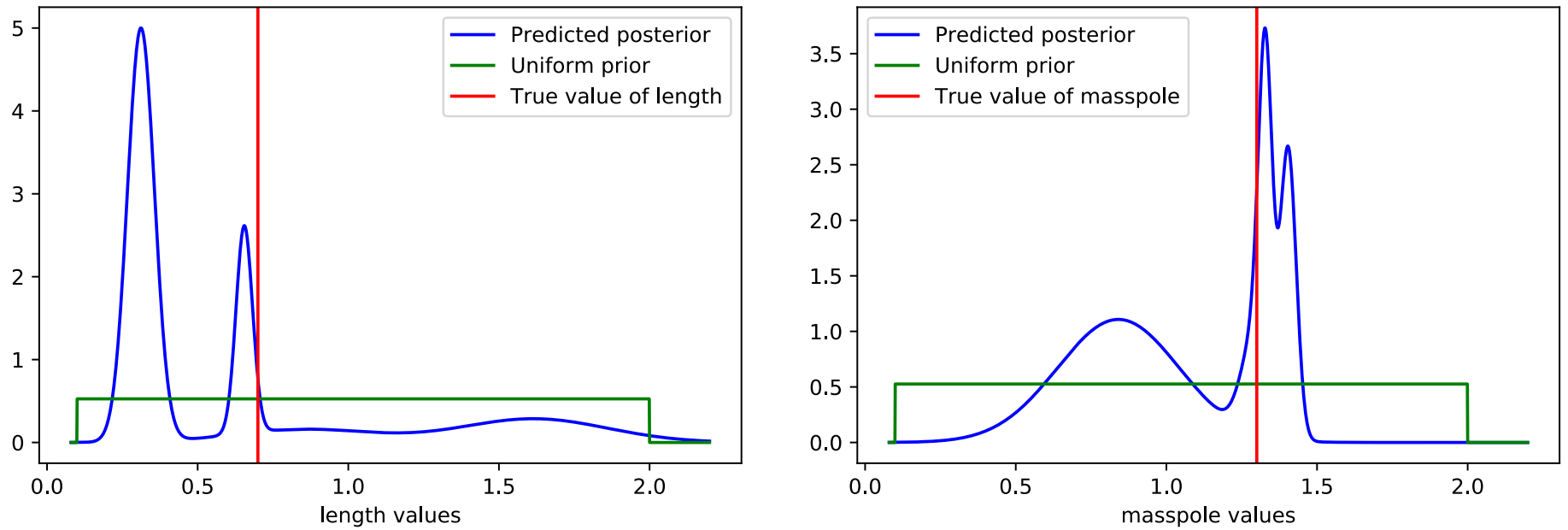
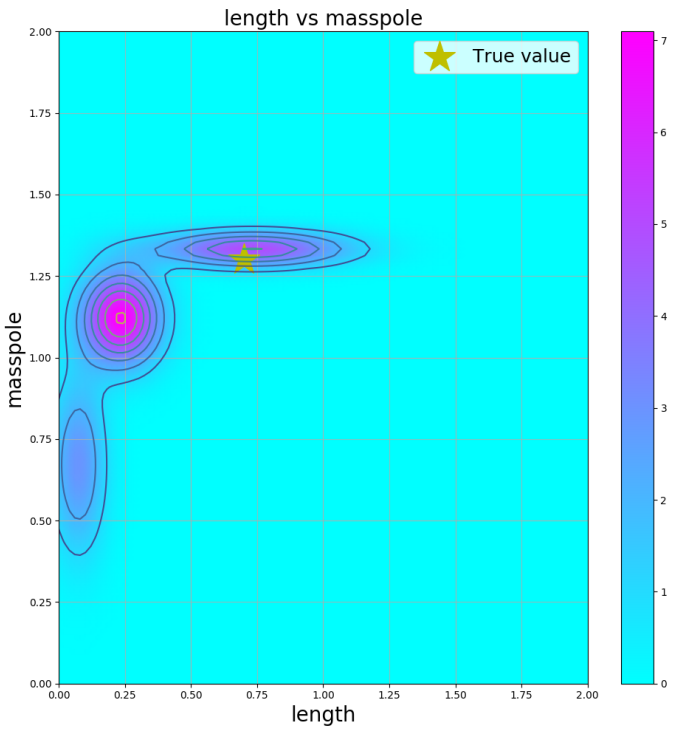
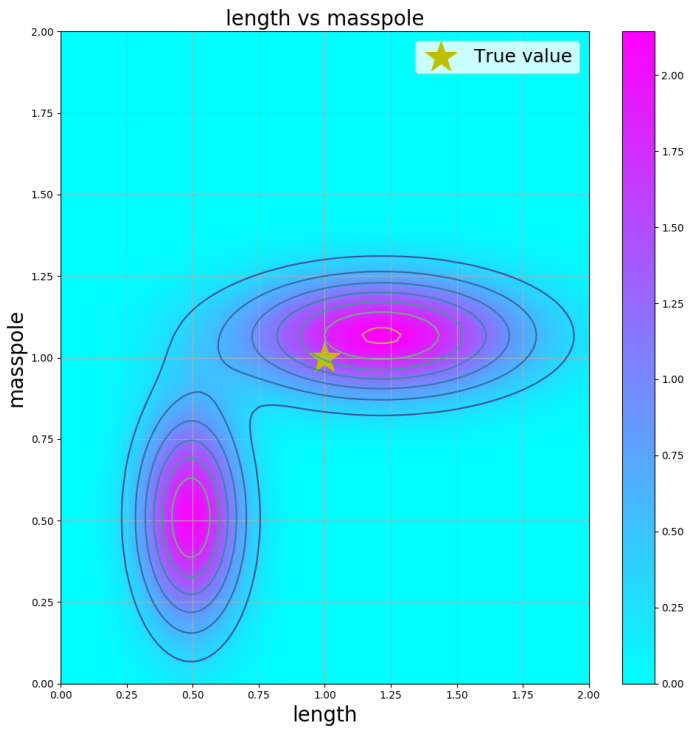
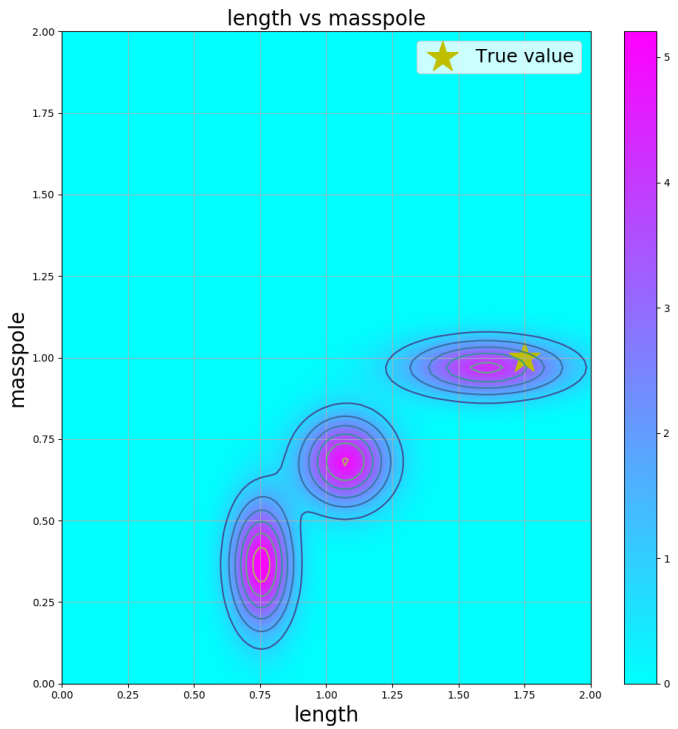



Fig. 2: Example of posterior marginals obtained for the CartPole problem. Left: Posterior for parameter *length*. Right: Posterior for parameters *masspole*. Note that the posterior marginals capture the multimodality of the problem where two explanations seem likely, a longer pole length with a lighter mass or vice versa.

# CART POLE EXAMPLE

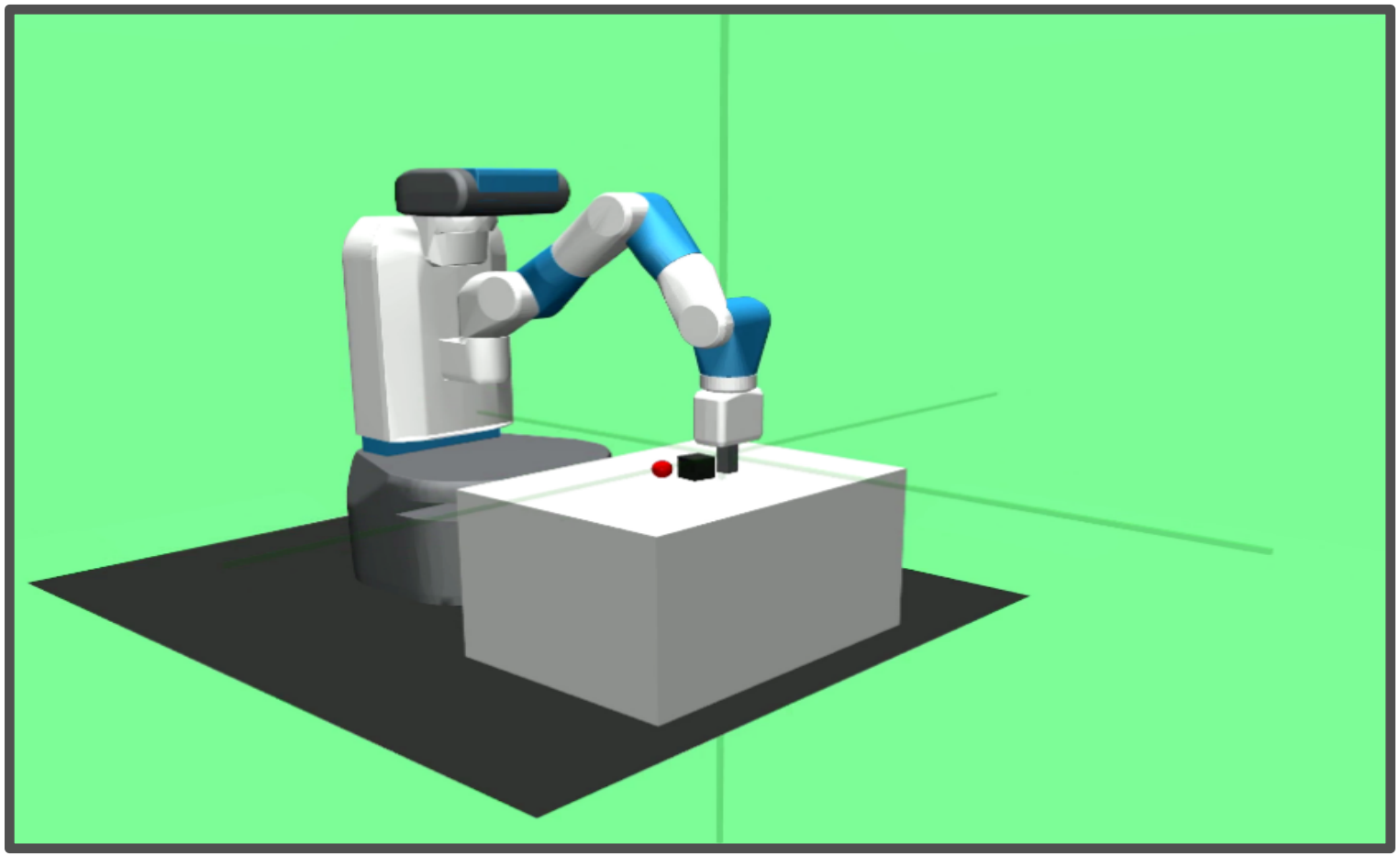


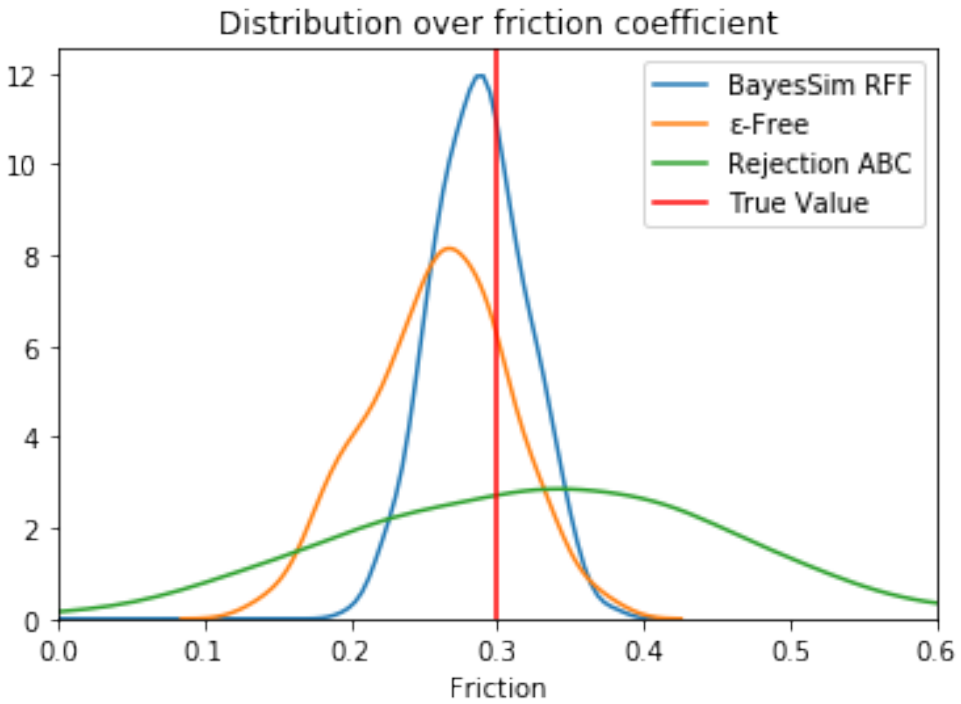



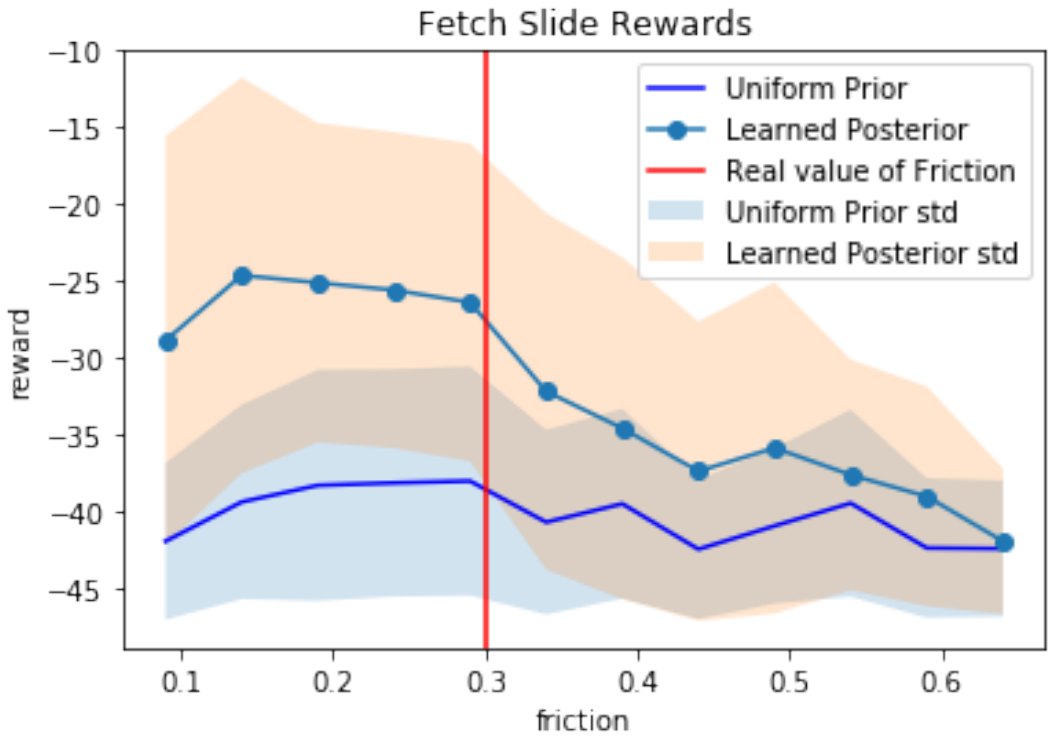
# PUSH AND SLIDE

*All simulators are wrong, some are useful,*
*but **many**, together, can be **much more** useful*

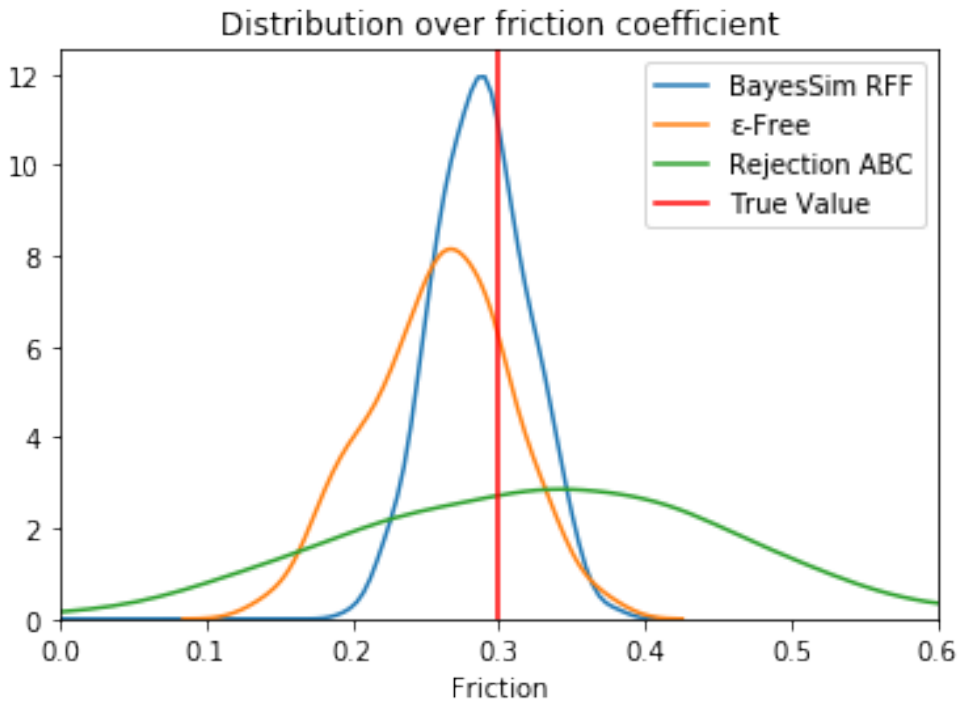


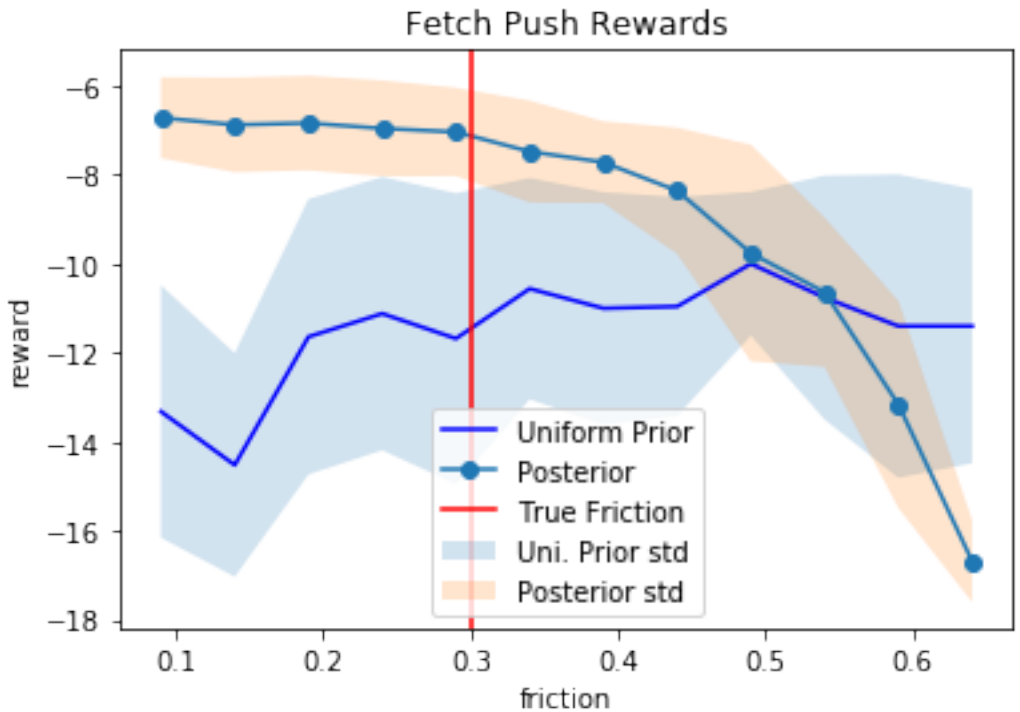
# FETCH SLIDE


Distribution over friction coefficient
BayesSim RFF
ε-Free
Rejection ABC
True Value
Friction
0
2
4
6
8
10
12
0.0
0.1
0.2
0.3
0.4
0.5
0.6


Fetch Slide Rewards
Uniform Prior
Learned Posterior
Real value of Friction
Uniform Prior std
Learned Posterior std
reward
friction
−10
−15
−20
−25
−30
−35
−40
−45
0.1
0.2
0.3
0.4
0.5
0.6

# FETCH PUSH


Distribution over friction coefficient
BayesSim RFF
ε-Free
Rejection ABC
True Value
Friction


Fetch Push Rewards
reward
friction
Uniform Prior
Posterior
True Friction
Uni. Prior std
Posterior std

# RESULTS

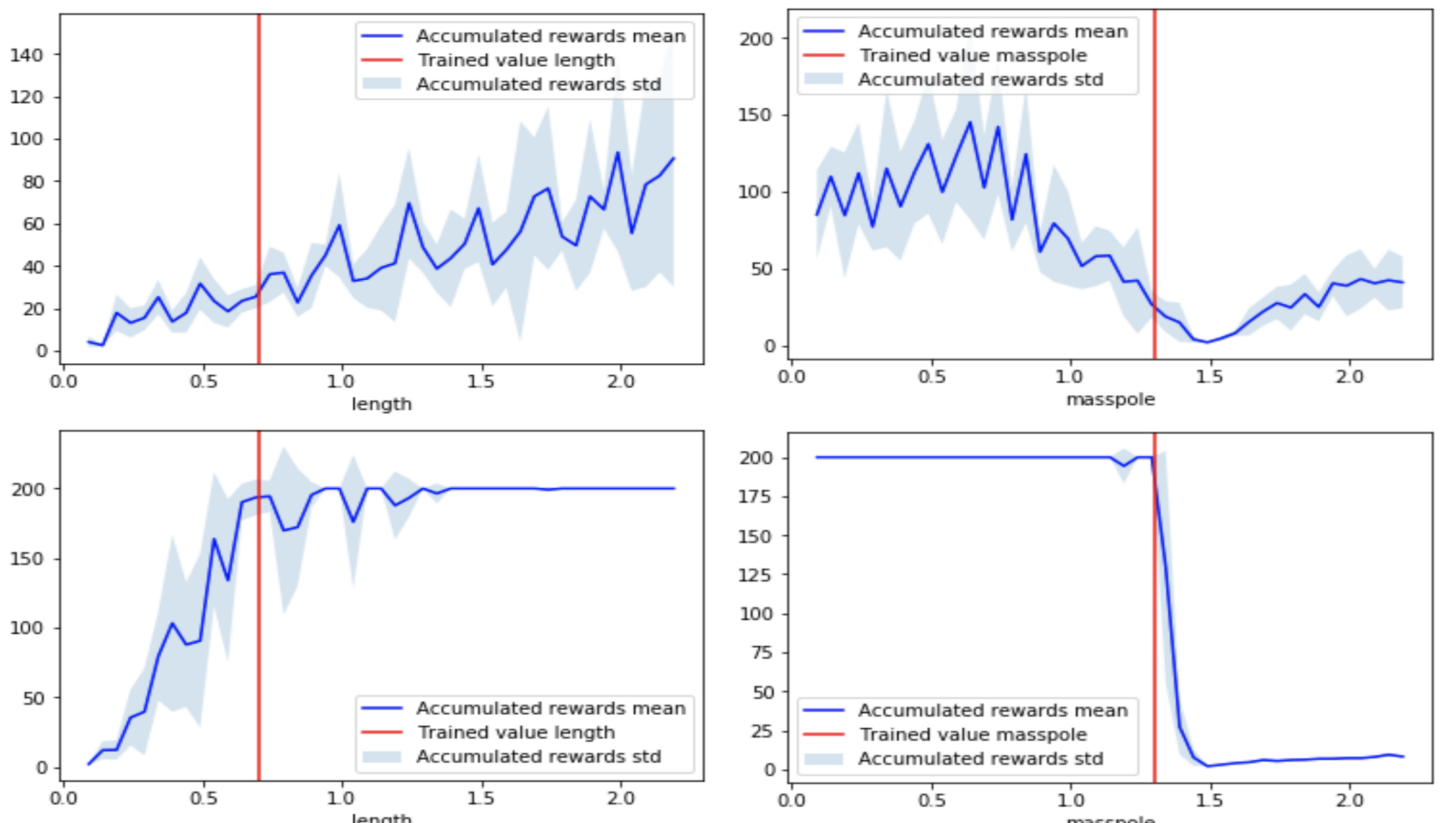



Fig. 4: Accumulated rewards for CartPole policies trained with PPO across parameter values. Top left: Policy trained by randomizing with the prior for parameter *length*. Top right: Policy trained by randomizing with the prior for parameter *masspole*. Bottom left: Policy trained by randomizing with the posterior for parameter *length*. Bottom right: Policy trained by randomizing with the posterior for parameter *masspole*.

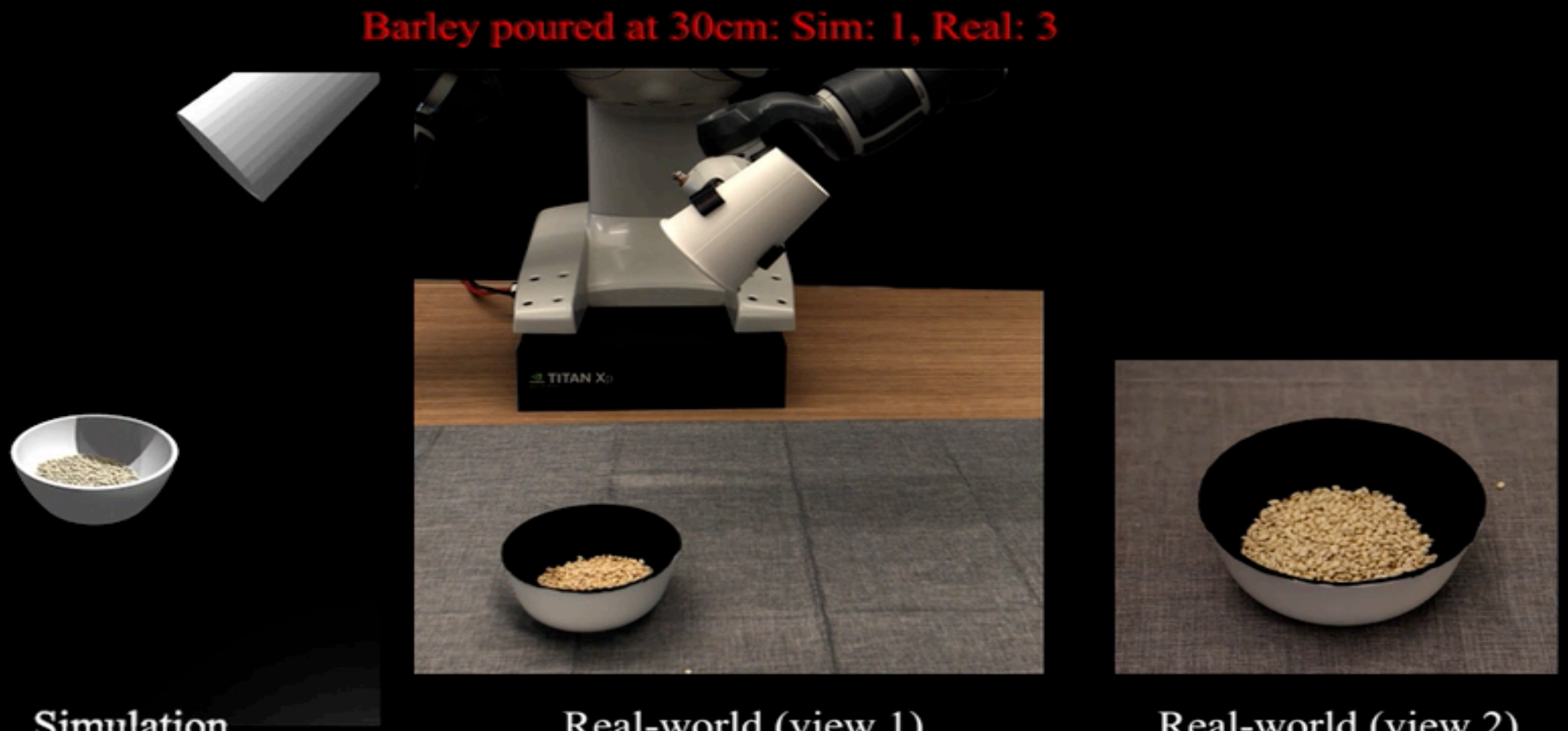
# RESULTS

## Comparison to other likelihood-free methods

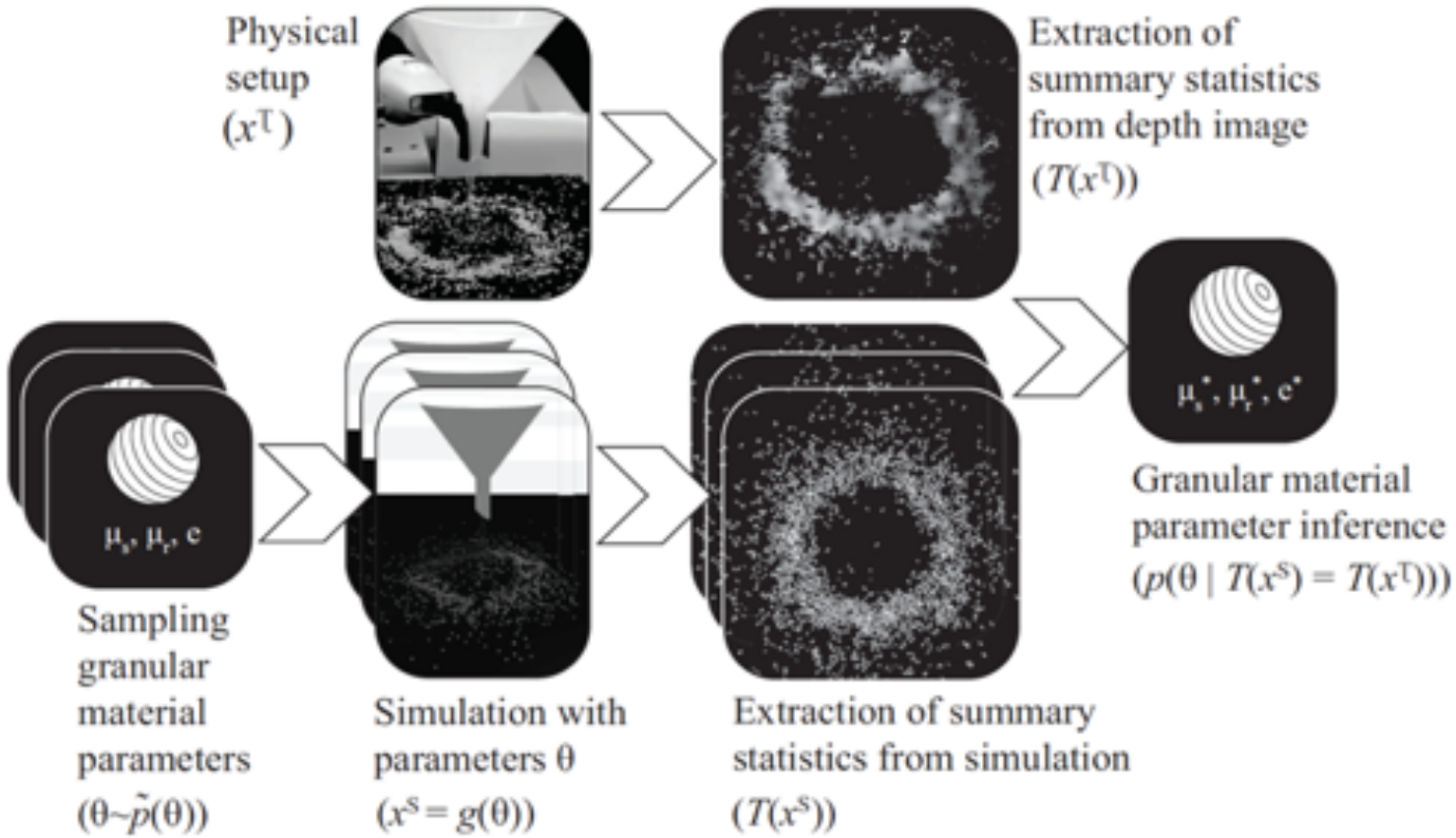
| Problem | Parameter | Uniform prior | Rejection ABC | $\epsilon$-Free | BayesSim RFF | BayesSim NN |
|---|---|---|---|---|---|---|
| CartPole | pole length | $[0.1, 2.0]$ | -0.342±0.15 | **-0.211±0.07** | -0.609±0.39 | -0.657±0.25 |
| | pole mass | $[0.1, 2.0]$ | 0.032±0.21 | 0.056±0.14 | **0.973 ± 0.26** | 0.633± 0.52 |
| Pendulum | dt | $[0.01, 0.3]$ | 2.101±1.04 | 2.307±0.84 | 3.192±0.30 | **3.199±0.17** |
| Mountain Car | power | $[0.0005, 0.1]$ | 3.69±1.21 | 3.800±1.06 | 3.863±0.52 | **3.901±0.2** |
| Acrobot | link mass 1 | $[0.5, 2.0]$ | 1.704±0.82 | 1.883±0.79 | **2.046±0.37** | 1.331±0.22 |
| | link mass 2 | $[0.5, 2.0]$ | 1.832±0.93 | **2.237±0.76** | 0.321±1.85 | 1.513±0.39 |
| | link length 1 | $[0.1, 1.5]$ | **2.421±0.75** | 2.135±0.50 | 2.072±0.76 | 1.856±0.18 |
| | link length 2 | $[0.5, 1.5]$ | -0.521±0.36 | -0.703±0.16 | **-0.148±0.19** | -0.672±0.09 |
| Hopper | lateral friction | $[0.3, 0.5]$ | 3.032±0.43 | 3.154±0.81 | 2.622±0.64 | **3.391±0.08** |
| Fetch Push | friction | $[0.1, 1.0]$ | 1.332±0.54 | 2.013±0.09 | **2.423±0.07** | 2.404±0.05 |
| Fetch Slide | friction | $[0.1, 1.0]$ | 1.014±0.38 | 1.614±0.12 | **2.391±0.06** | 2.111±0.03 |

# INFERRING GRANULAR PARAMETERS

[Matl et al. Inferring the Material Properties of Granular Media for Robotic Tasks. Submitted to ICRA'20]

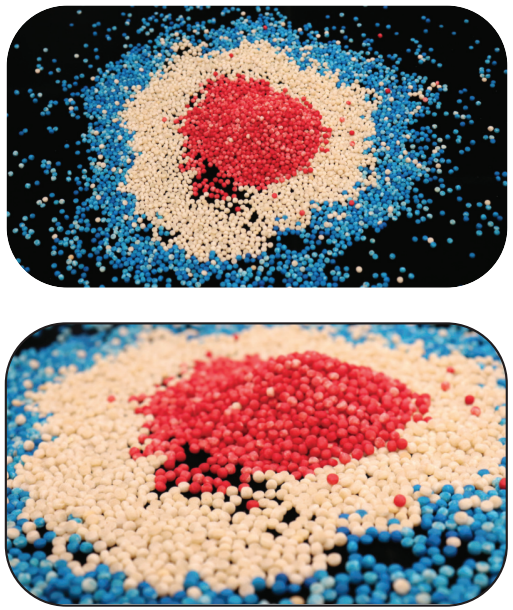


# INFERRING MATERIAL PROPERTIES OF GRANULAR MEDIA



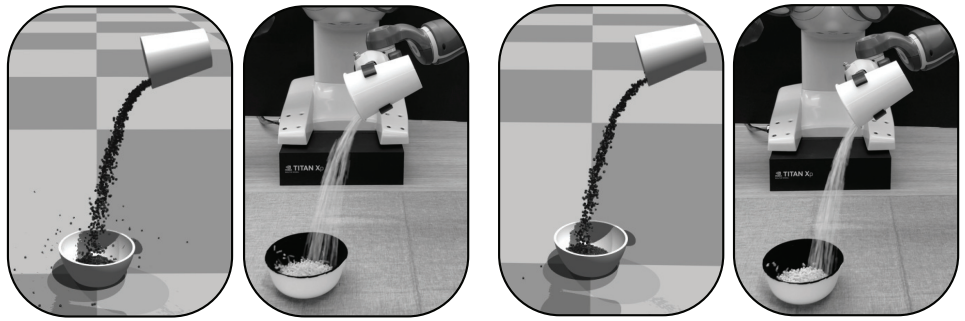
The framework is used to estimate the simulation values for the coefficients of sliding friction, rolling friction, and restitution
for couscous

Inferring pouring height
for ring pattern



Pouring Couscous
Simulation Robot

Pouring Barley
Simulation Robot



Number of grains that bounce out at different heights (3 pours each)

| | | | | |
|---|---|---|---|---|
| 15cm | 0, 0, 0 | 0, 0, 0 | 0, 0, 0 | 0, 0, 0 |
| 30cm | 23, 20, 18 | 12, 15, 8 | 1, 0, 0 | 3, 2, 3 |
| 45cm | 52, 50, 53 | 46, 36, 44 | 4, 12, 10 | 22, 11, 24 |
| 60cm | 133, 90, 116 | 115, 95, 127 | 35, 48, 49 | 33, 40, 42 |

**Fig. 6:** Two granular manipulation tasks were tested. (Left): The height of the funnel is inferred to create a desired ring shape. We demonstrate this by creating a pattern of three concentric rings. (Right): A real-life scene pouring grains into a bowl is recreated in simulation. At various different heights, both the simulation of couscous and barley perform well in estimating the magnitude of grains to leave the cereal bowl.

# TAKE HOME MESSAGES

- Epistemic and aleatoric uncertainty can help to develop algorithms for different purposes (active learning, multi-modal trajectories).
- Bayesian likelihood-free inference is a powerful tool to connect black-box models with data-driven methods.
- BayesSim infers the uncertainty over simulation parameters and use it for robust Sim2Real.

# SUMMARY