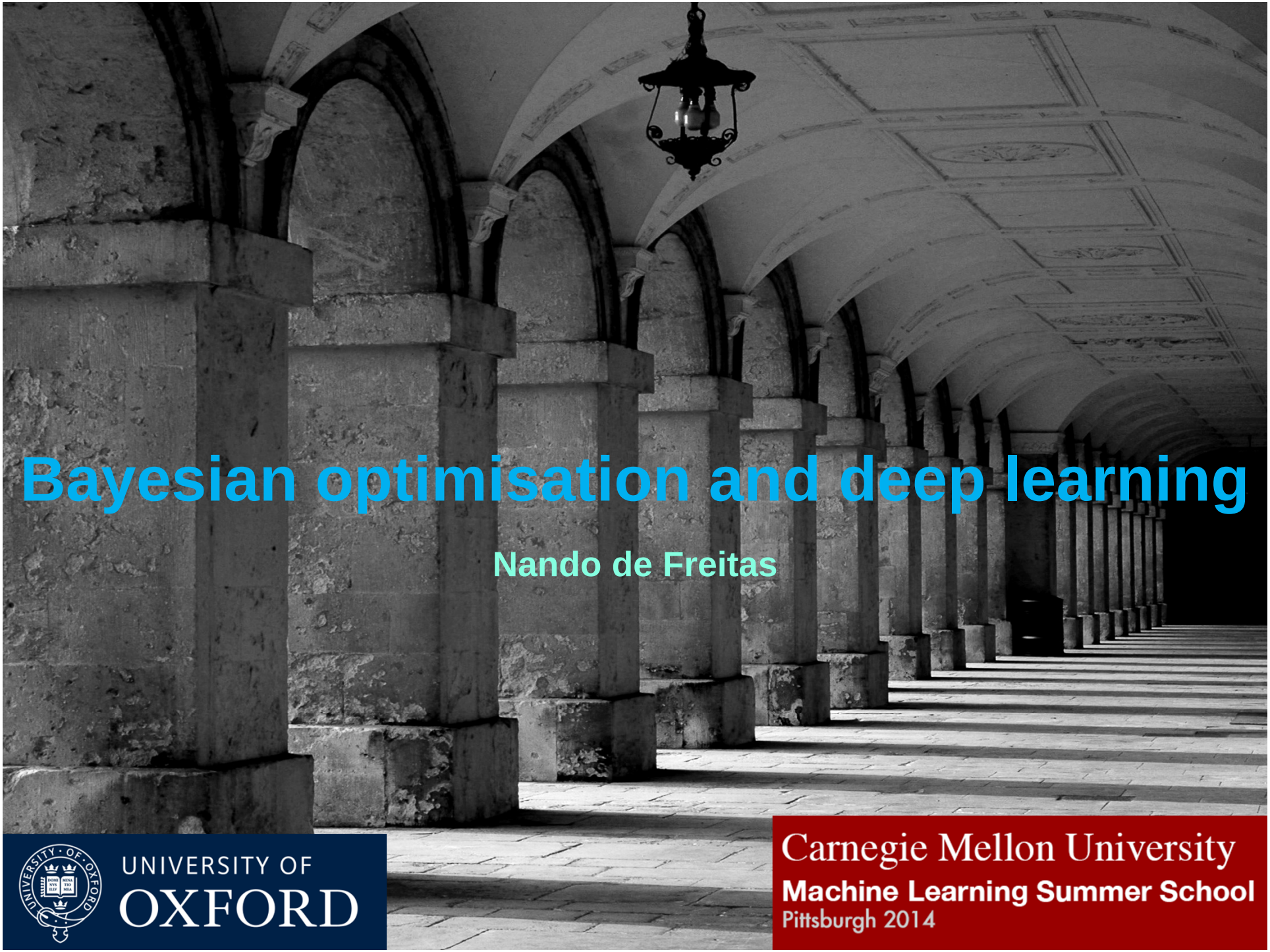


Bayesian optimisation and deep learning

Nando de Freitas

UNIVERSITY OF OXFORD

Carnegie Mellon University

Machine Learning Summer School

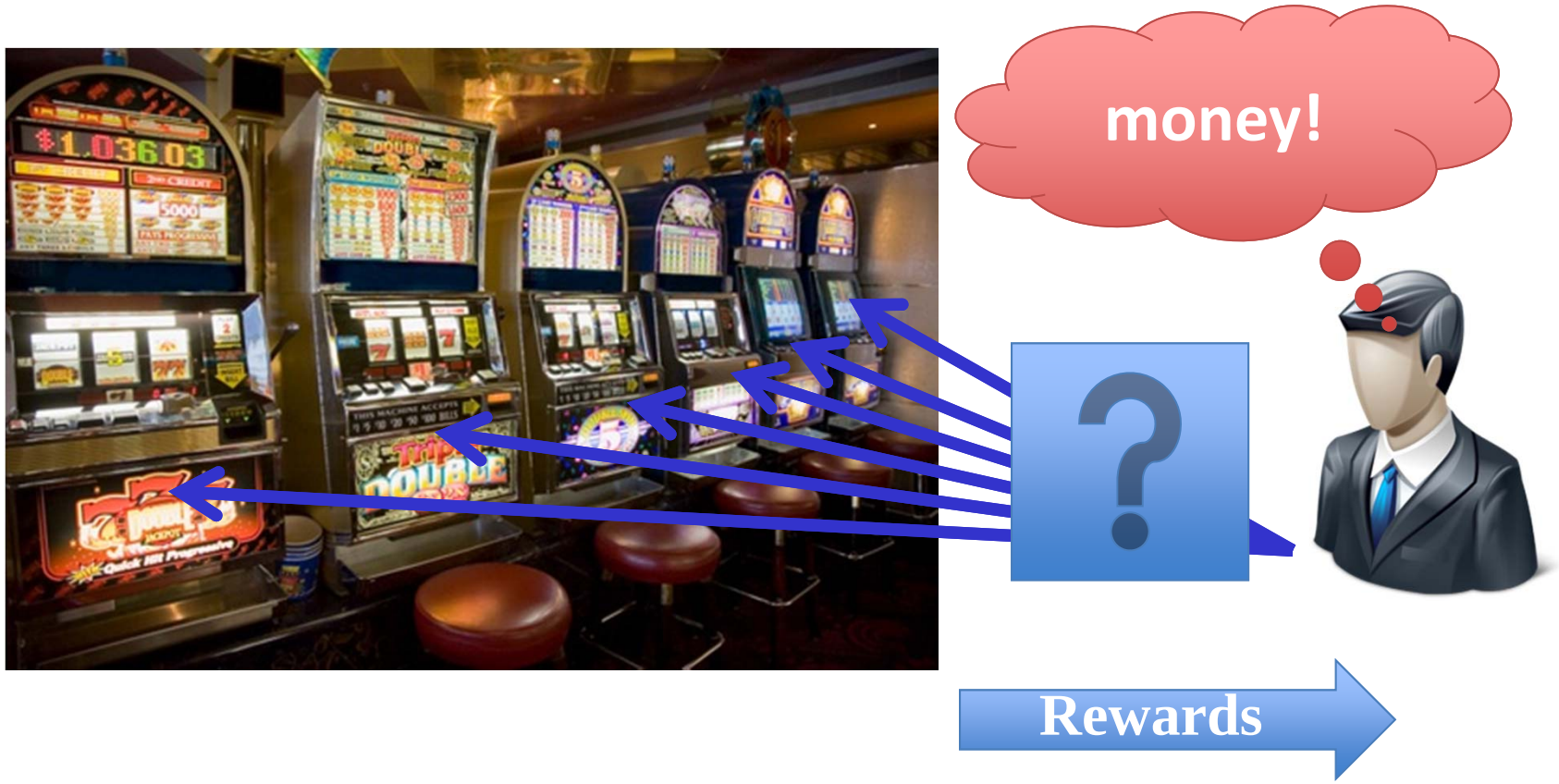
Pittsburgh 2014

Vision for a purpose

o x

[Laurent Itti / Dana Ballard]

Goals and decisions



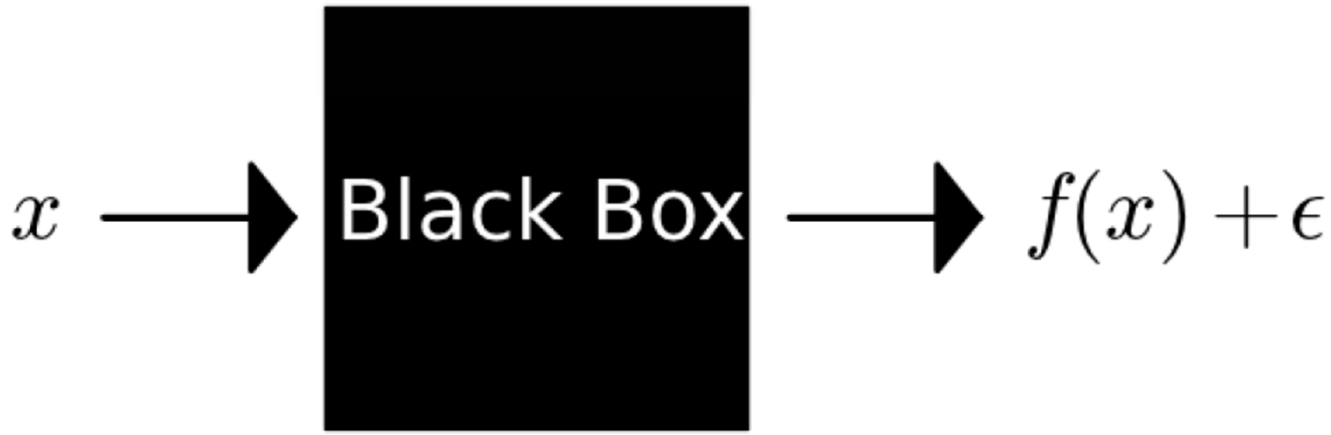
Regret = Player reward – Reward of best action

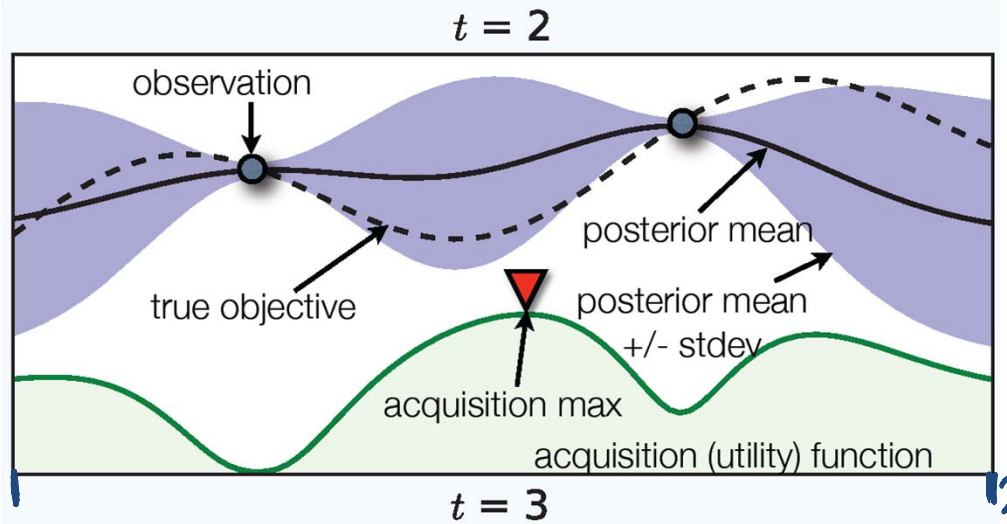
Trade-off between Exploration and Exploitation

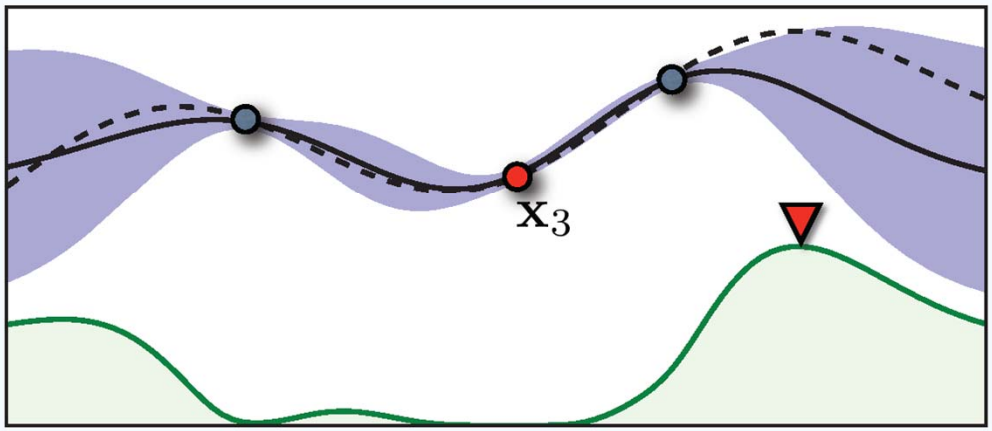
Black-box optimization / Design

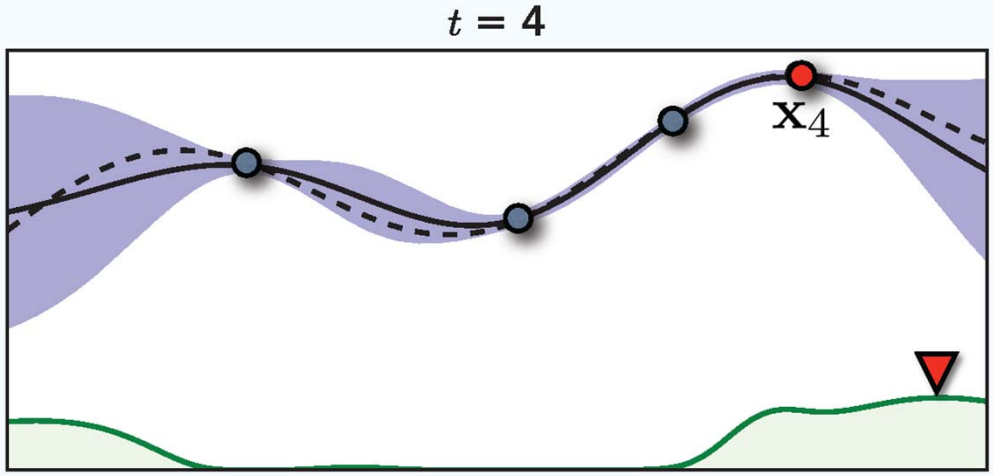
Bayesian Optimization (BO) addresses the following global optimization problem

$$\mathbf{x}^\star = \arg\max_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}).$$









Bayesian optimization

1: **for** $t = 1, 2, \ldots$ **do**

2: Find $\mathbf{x}_t$ by combining attributes of the posterior distribution in a utility function u and maximizing: $\mathbf{x}_t = \text{argmax}_{\mathbf{x}}\, u(\mathbf{x}|\mathcal{D}_{1:t-1})$.

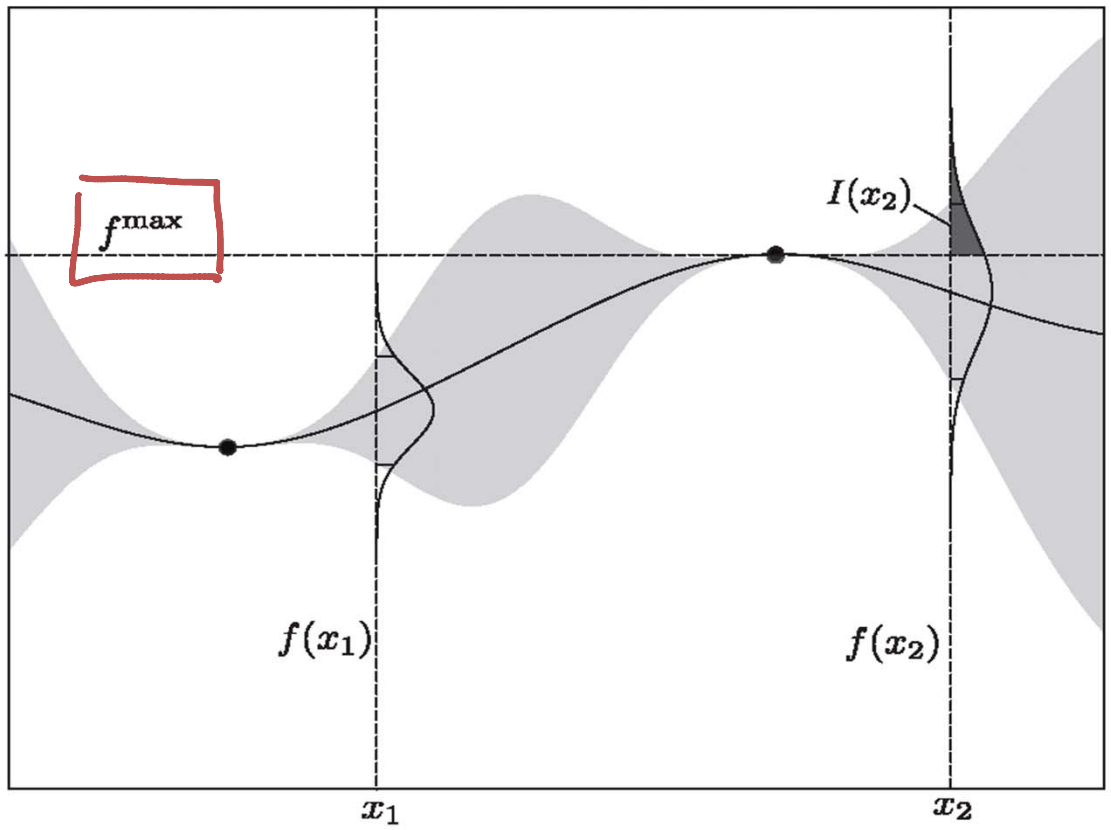
3: Sample the objective function: $y_t = f(\mathbf{x}_t) + \varepsilon_t$.

4: Augment the data $\mathcal{D}_{1:t} = \{\mathcal{D}_{1:t-1}, (\mathbf{x}_t, y_t)\}$ and update the GP.

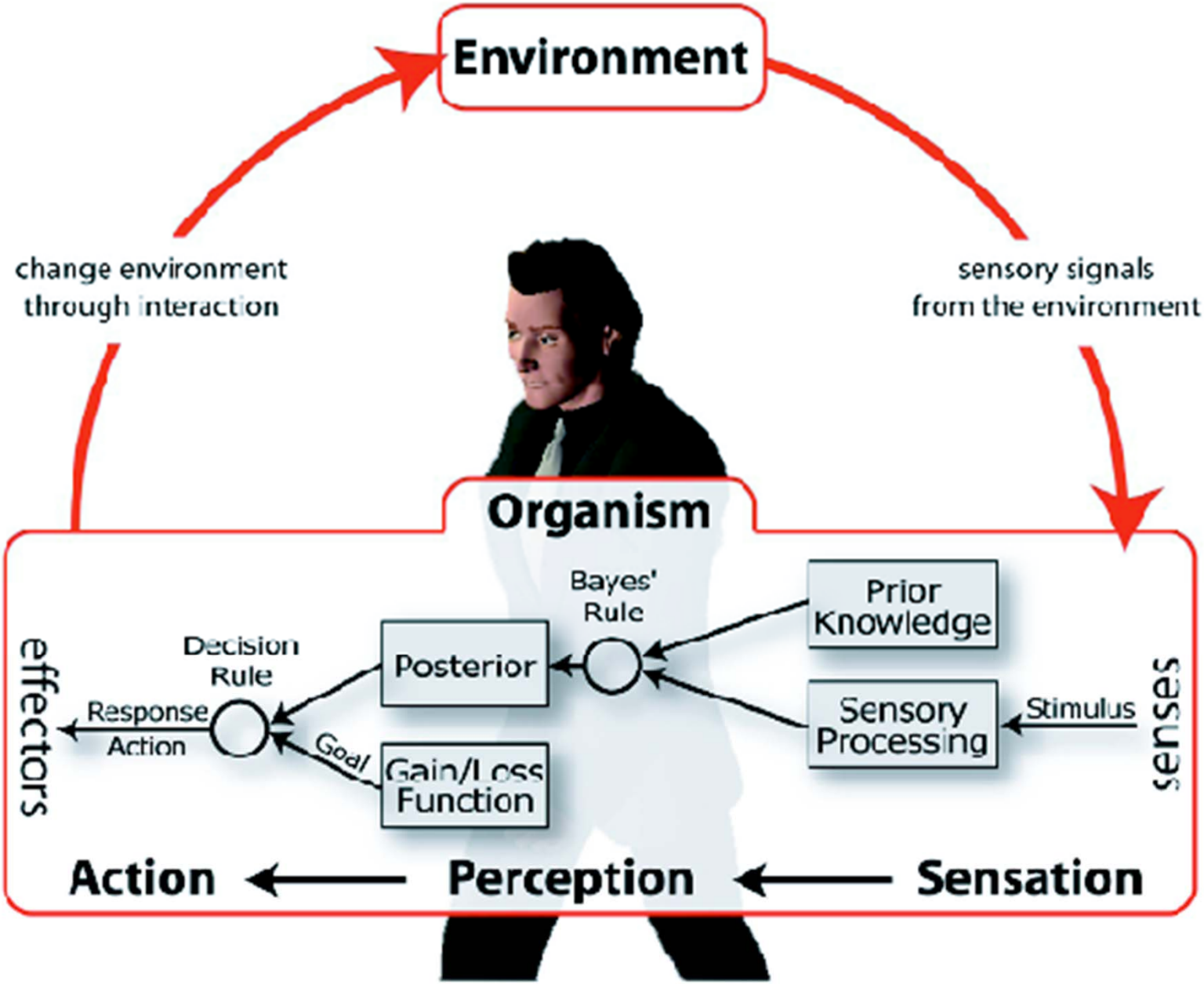
5: **end for**

Probability of Improvement

$$\mathrm{PI}(\mathbf{x}) = P(f(\mathbf{x}) \geq f^{max})$$
$$= \Phi\left(\frac{\mu(\mathbf{x}) - f^{max}}{\sigma(\mathbf{x})}\right)$$



People as Bayesian reasoners



Expected improvement

At iteration *n+1*, choose the point that minimizes the distance to the objective evaluated at the maximum ***x****:

$$\mathbf{x}_{n+1} = \arg\min_{\mathbf{x}} \int \|f_{n+1}(\mathbf{x}) - f(\mathbf{x}^\star)\| p(f_{n+1}|\mathcal{D}_n) df_{n+1}$$

The true objective at the maximum is unknown. To overcome this, Mockus proposed the following EI criterion:

$$\mathbf{x} = \arg\max_{\mathbf{x}} \mathbb{E}(\max\{0, f_{n+1}(\mathbf{x}) - f^{\max}\} \,|\mathcal{D}_n)$$

Some acquisition functions

$$\mu^+ = \text{argmax}_{\mathbf{x}_i \in \mathbf{x}_{1:t}} \mu(\mathbf{x}_i)$$

- Probability of Improvement

$$\text{PI}(\mathbf{x}) = \Phi\left(\frac{\mu(\mathbf{x}) - \mu^+ - \xi}{\sigma(\mathbf{x})}\right)$$

Kushner 1964

- Expected Improvement

$$\text{EI}(\mathbf{x}) = (\mu(\mathbf{x}) - \mu^+ - \xi)\Phi(Z) + \sigma(\mathbf{x})\phi(Z)$$

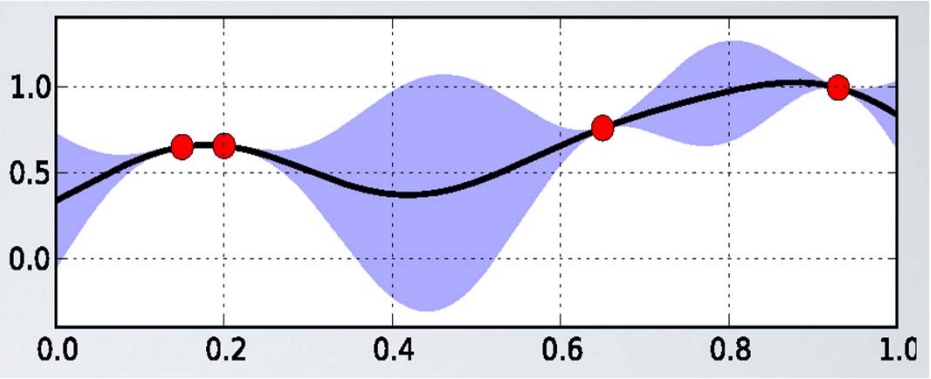
$$Z = \frac{\mu(\mathbf{x}) - \mu^+ - \xi}{\sigma(\mathbf{x})}$$

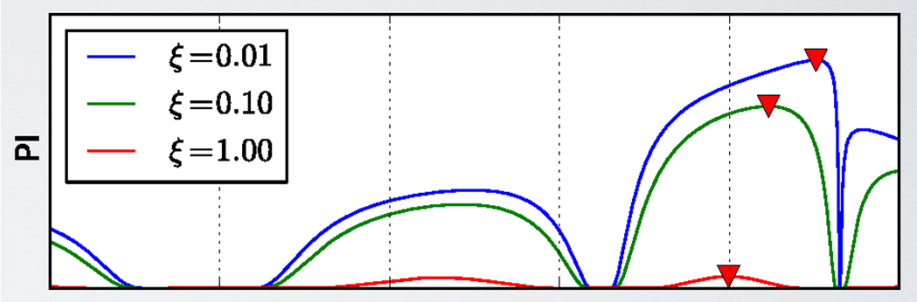
Mockus 1978

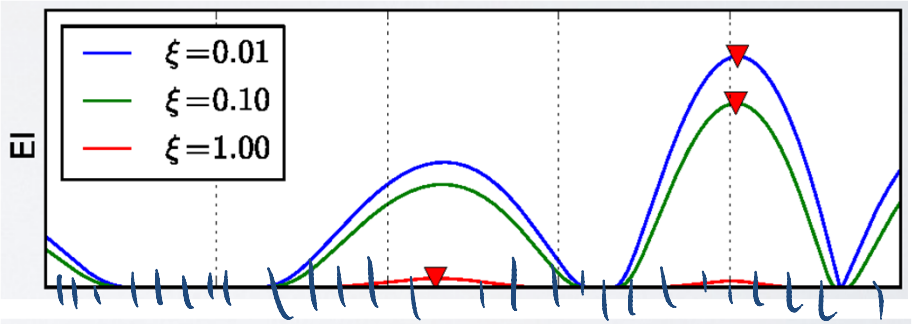
- Upper Confidence Bound

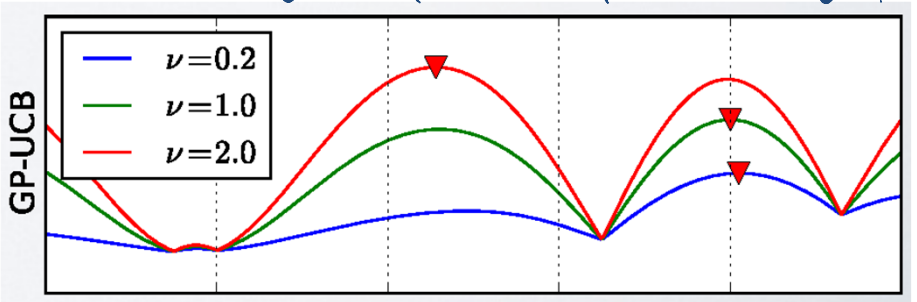
$$\text{GP-UCB}(\mathbf{x}) = \mu(\mathbf{x}) + \sqrt{\nu\tau_t}\sigma(\mathbf{x})$$

Srinivas *et al.* 2010









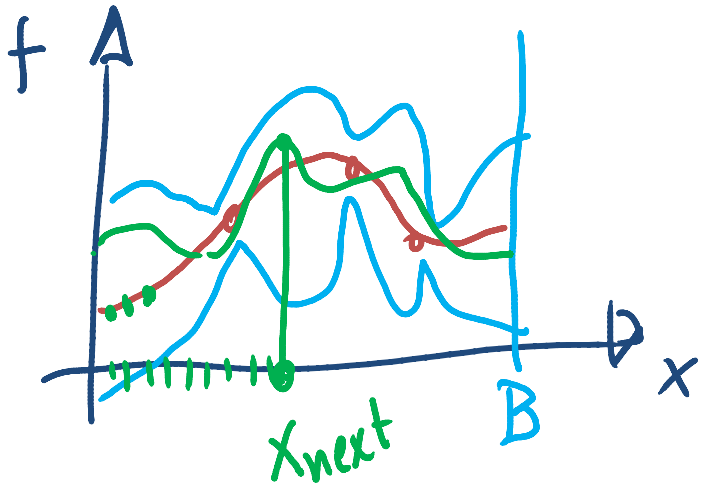
Thompson sampling

Consider the posterior over the location of the minimum:

$$\mathbb{P}(\,\mathrm{d}\mathbf{x}_\star|\mathcal{D}) = \mathbb{P}\left(\arg\min_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}) \in \,\mathrm{d}\mathbf{x}_\star \middle| \mathcal{D}\right)$$

The Thompson (aka probability matching) strategy is to draw a single sample from this distribution at each iteration:

$$\alpha_{\text{THOMPSON}}(\mathcal{D}) = \mathbf{x}_\star^{(0)} \quad \text{where} \quad \mathbf{x}_\star^{(0)} \sim p(\mathbf{x}_\star|\mathcal{D}_t)$$

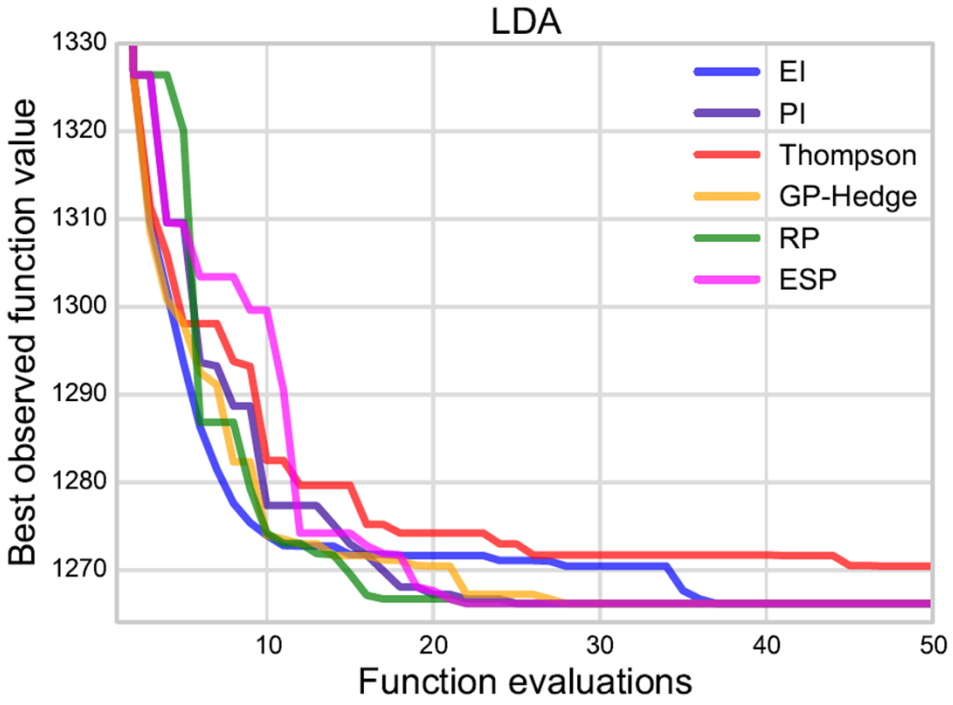


For GPs, this is done using random kitchen sinks or FastFood.

Entropy search portfolios

Random sinks/FastFood can also be used to approximate the expected information over the location of the optimum.

$$\mathbb{E}_{p(y_k|\mathcal{D},\mathbf{x}_k)}[H[p(\mathbf{x}_\star|\tilde{\mathcal{D}}_k)]] \text{ where } \tilde{\mathcal{D}} = \mathcal{D} \cup \{(\mathbf{x}_k, y_k)\}$$



[Lindley,
Vilemonteix et al,
Hennig et al
Shahriari, Hoffman, Wang and NdF]

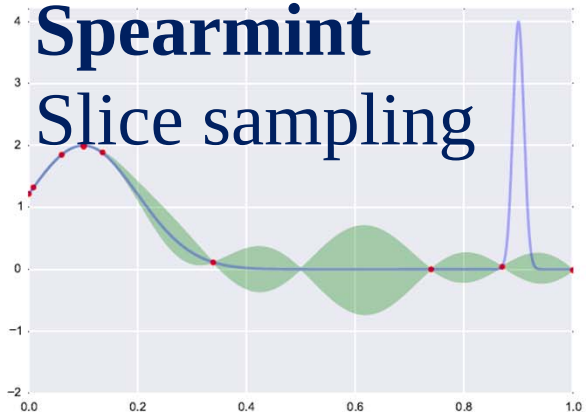
Hyper-parameters of the GP

Learning the hyper-parameters of the GP is important.

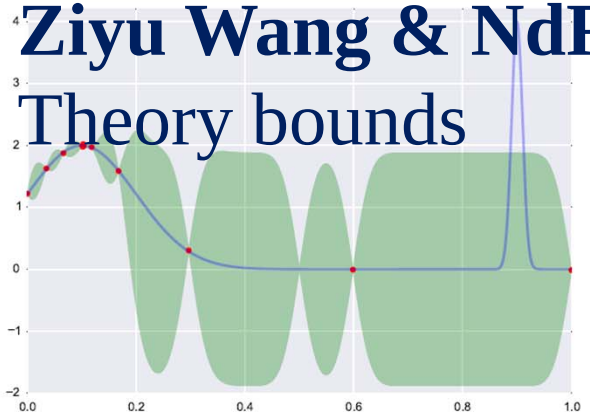
The tuning method must be automatic!

One of the best ways to manage the GP hyper-parameters is to integrate them out, *e.g.* with slice sampling as in **Spearmint**. But this is still dangerous!

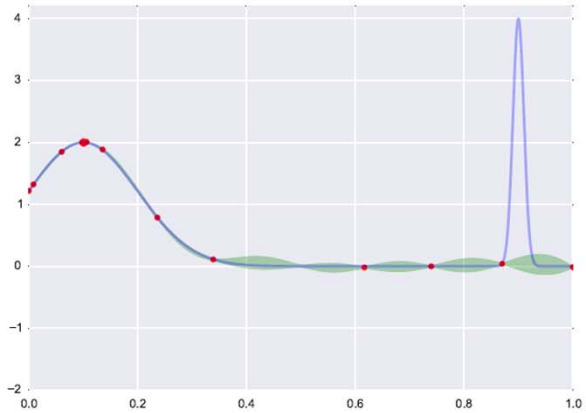
Spearmint
Slice sampling

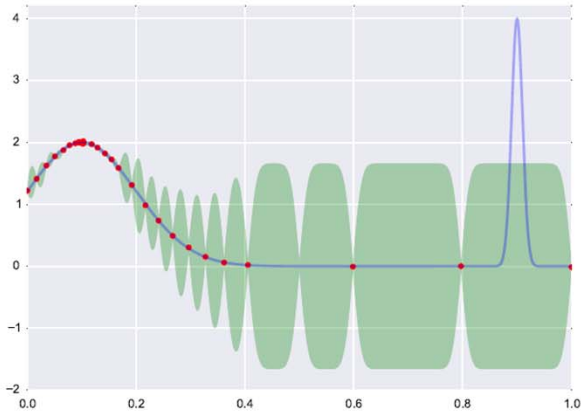


Ziyu Wang & NdF 2014
Theory bounds

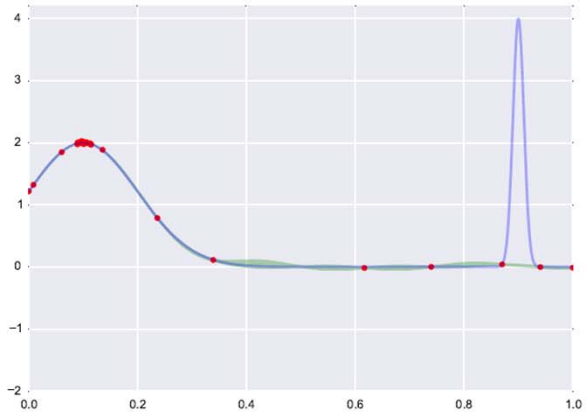


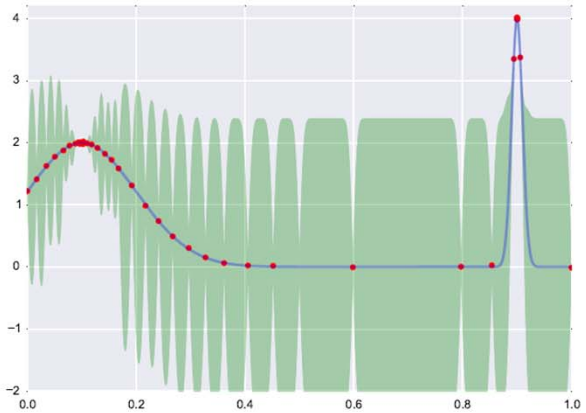
$t = 20$





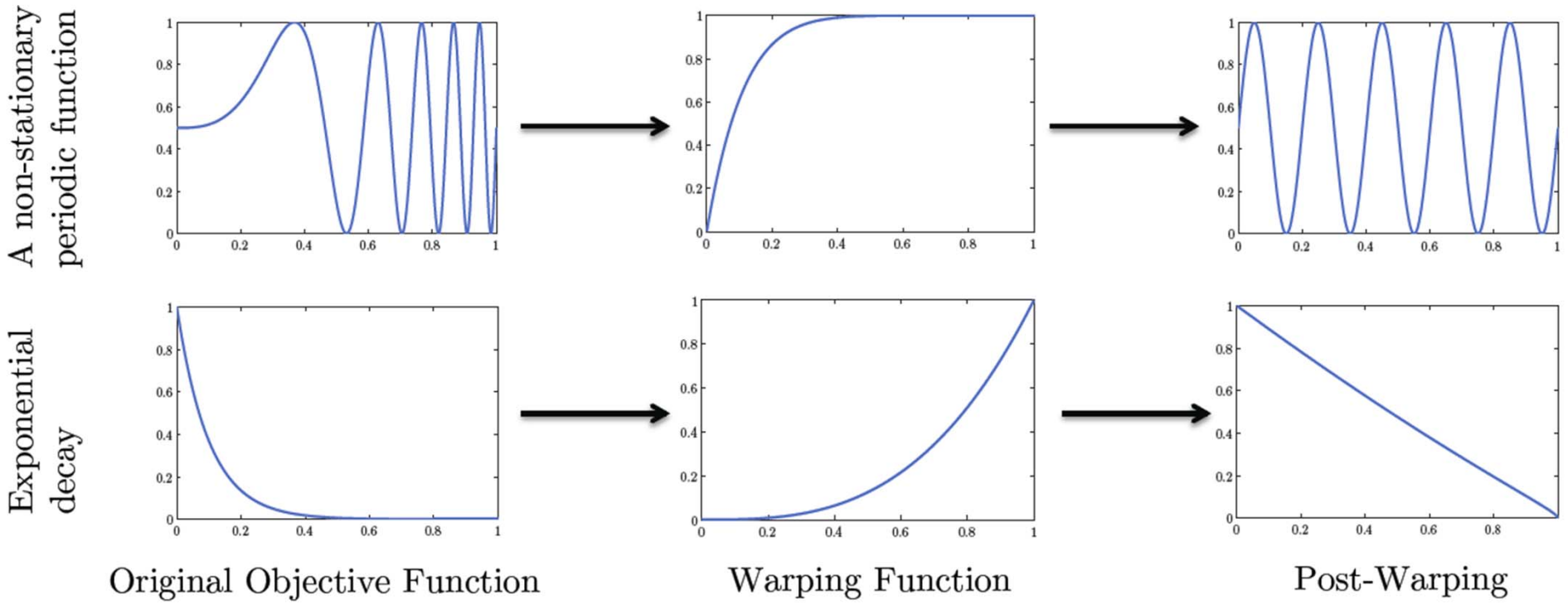
$t = 40$





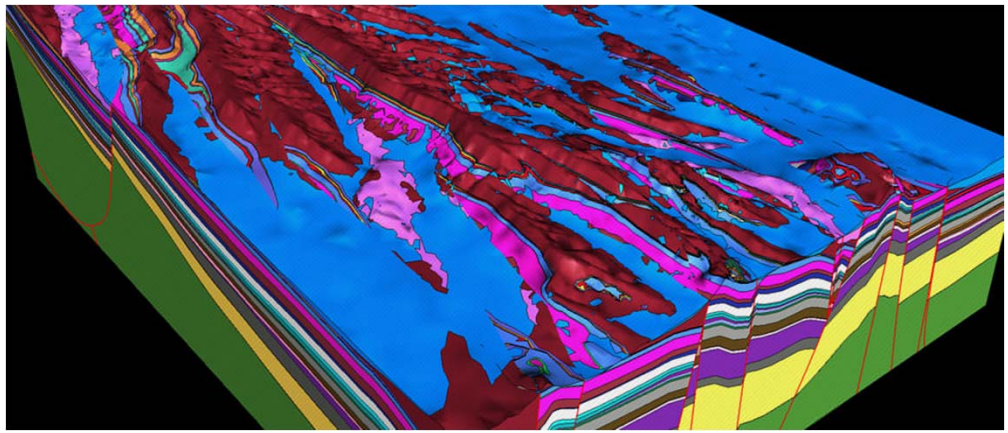
$t = 60$

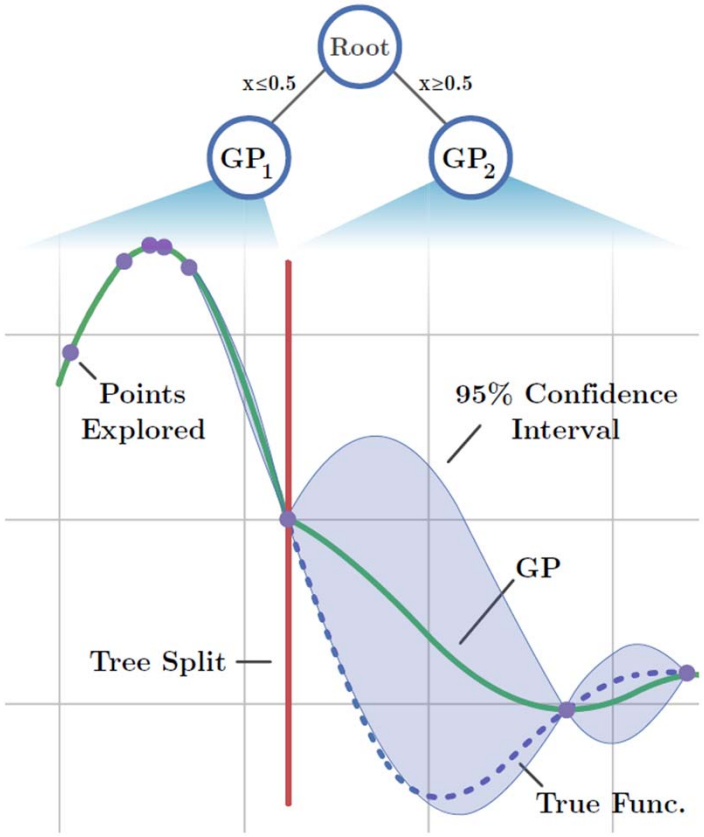
Warping is very useful to deal with heteroskedasticity

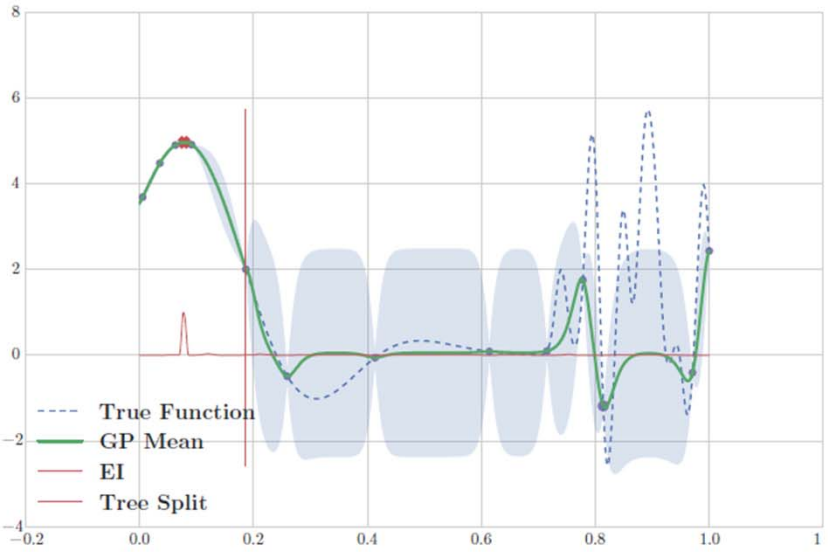


[Jasper Snoek, Kevin Swersky, Rich Zemel, Ryan Adams, 2014]

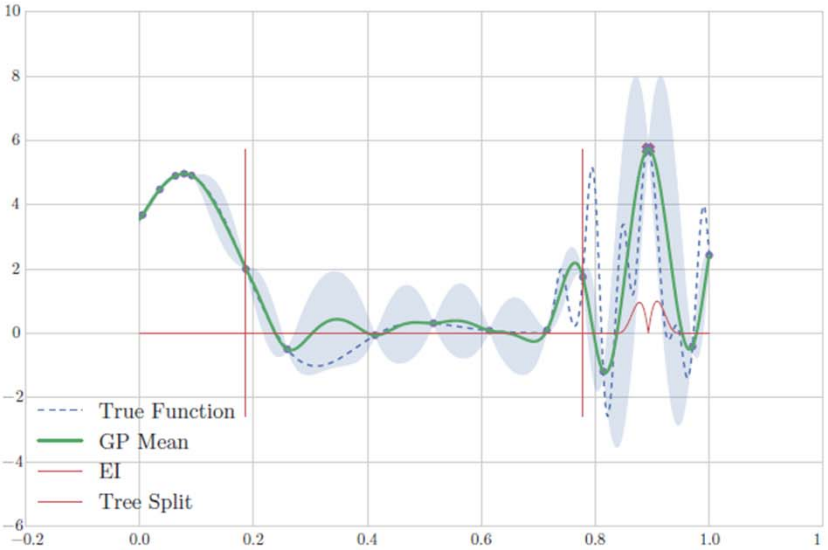
Warping + trees







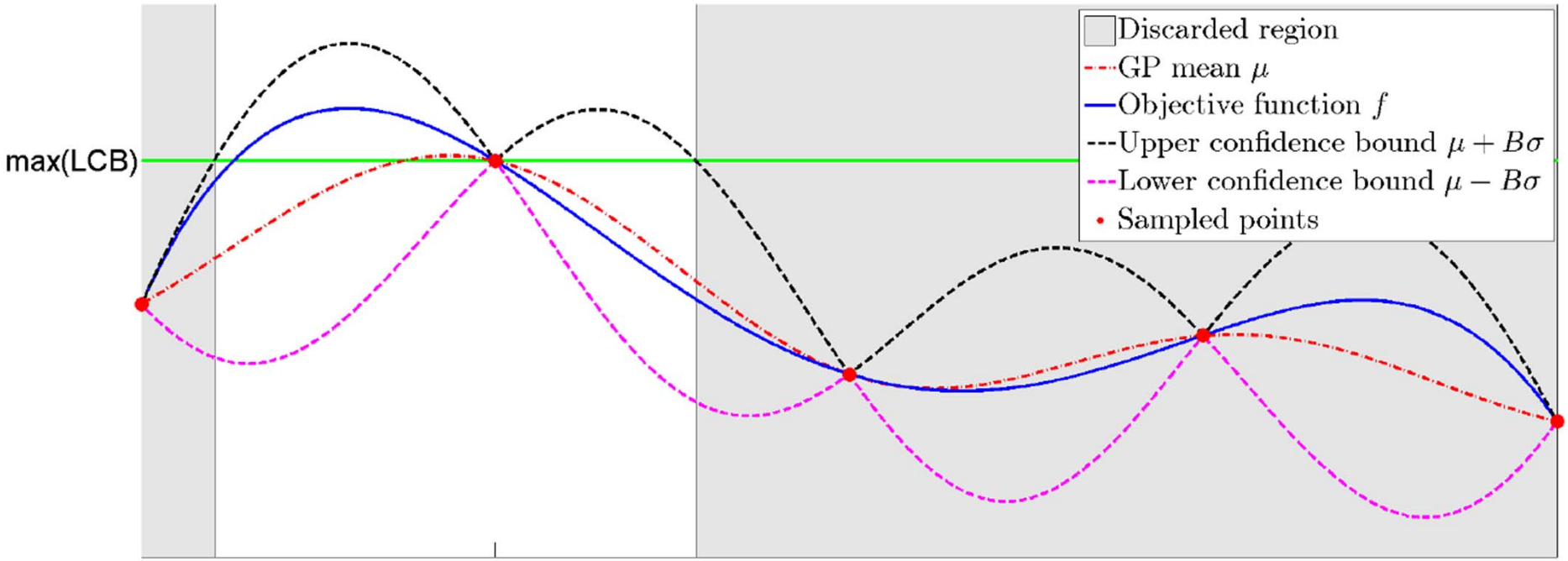
(b) Iteration 14



(d) Iteration 16

[Assael, Wang and NdF]

Analysis of Bayes Opt [dF, Zoghi & Smola, 2012]

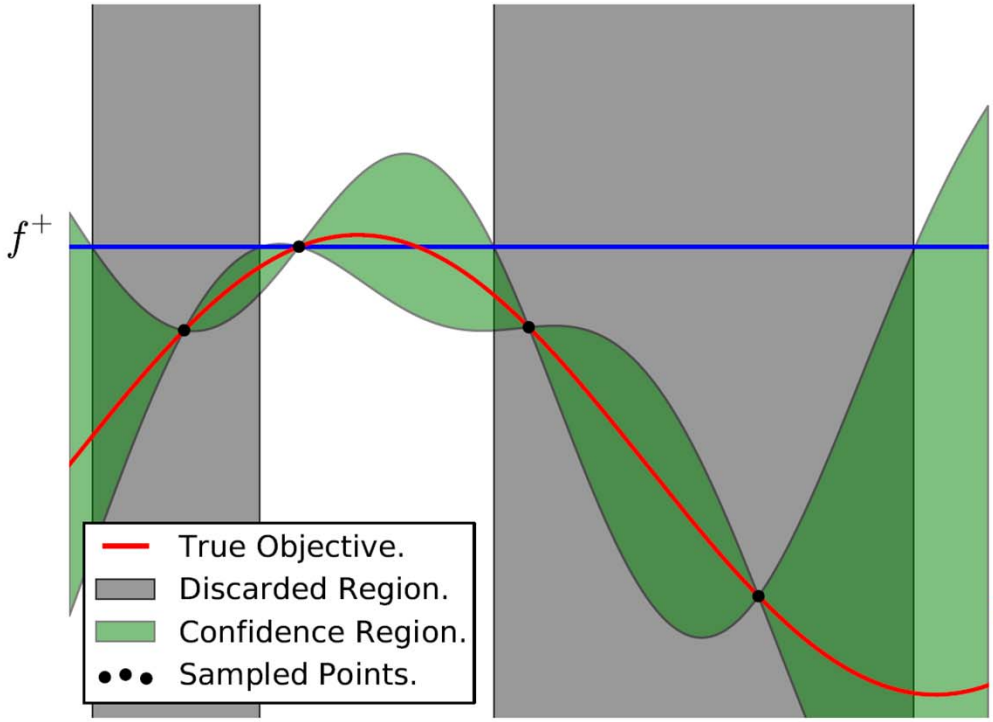


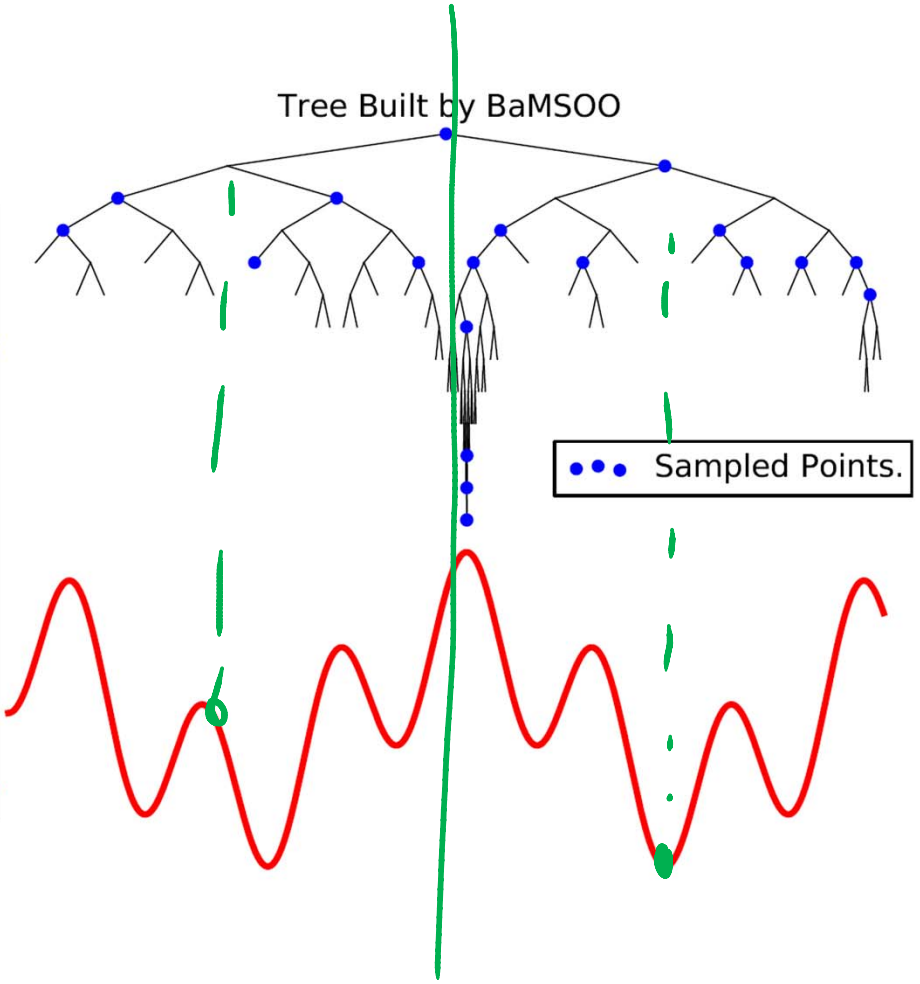
Proposition 1 (Variance Bound) : $\sup_{\mathcal{D}} \sigma_T \leq \frac{Q\delta^2}{4}$

Theorem 2 (Vanishing regret) : $r(x_t) < Ae^{-\frac{\tau t}{(\ln t)^{d/4}}}$

$$r(x_t) = f(x_M) - f(x_t)$$

Multi-scale optimistic optimization

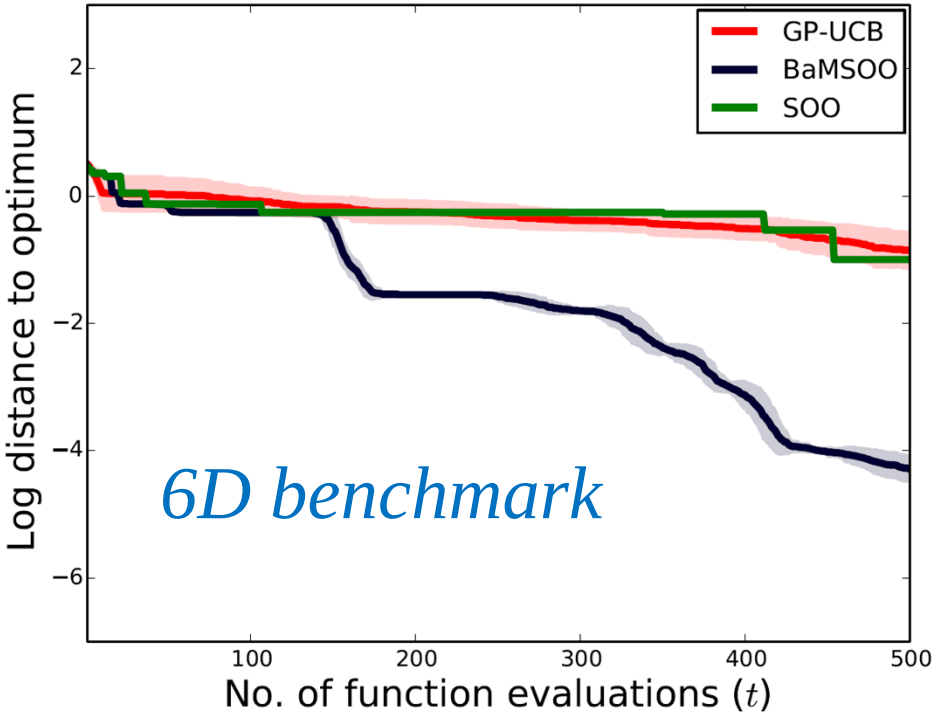


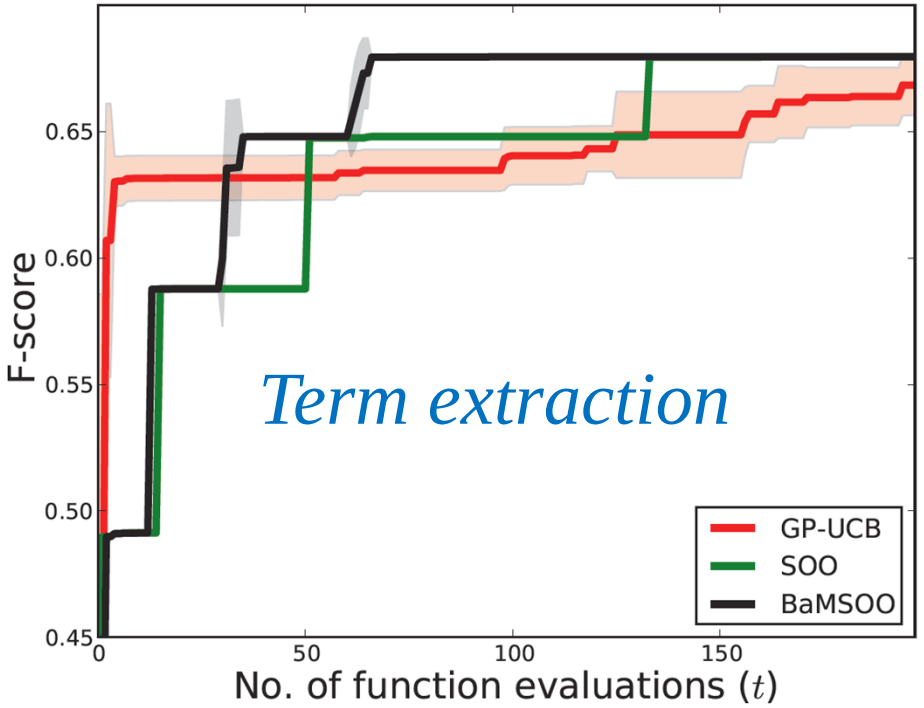


[Remi Munos - SOO, UCT]

[Ziyu Wang et al, 2014]

Multi-scale optimistic optimization

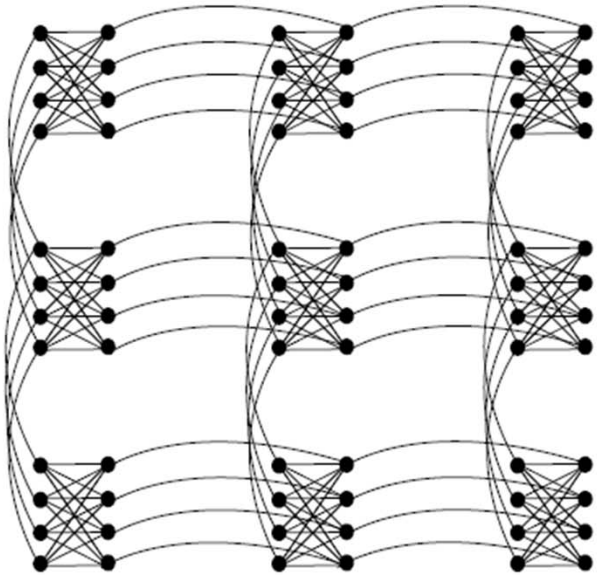


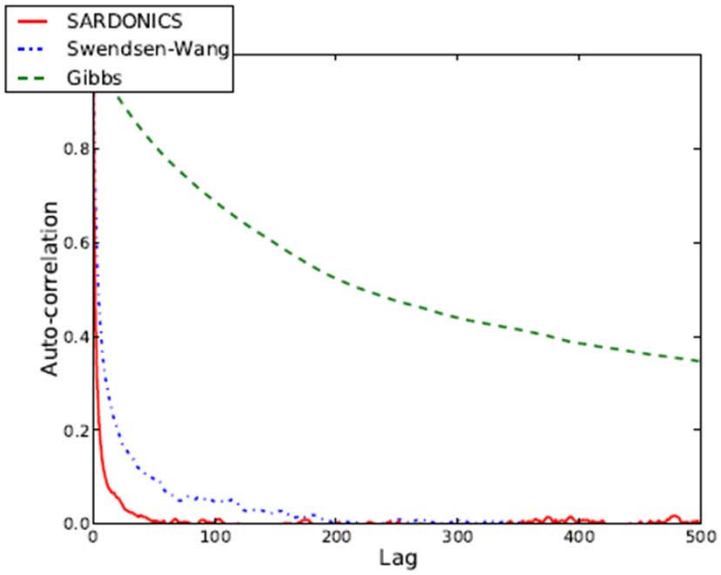


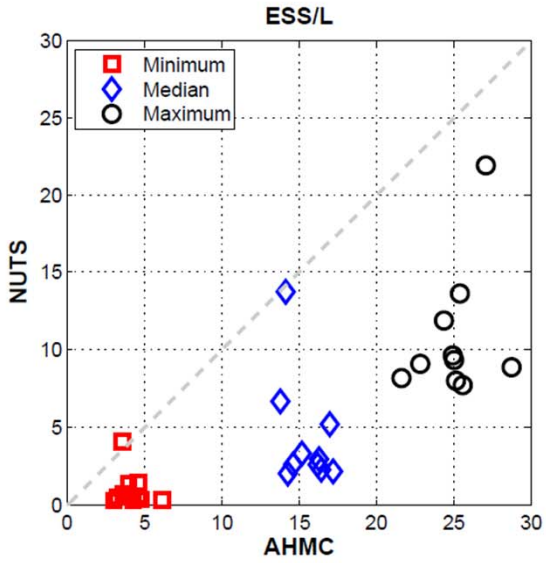
[Ziyu Wang et al, 2014]

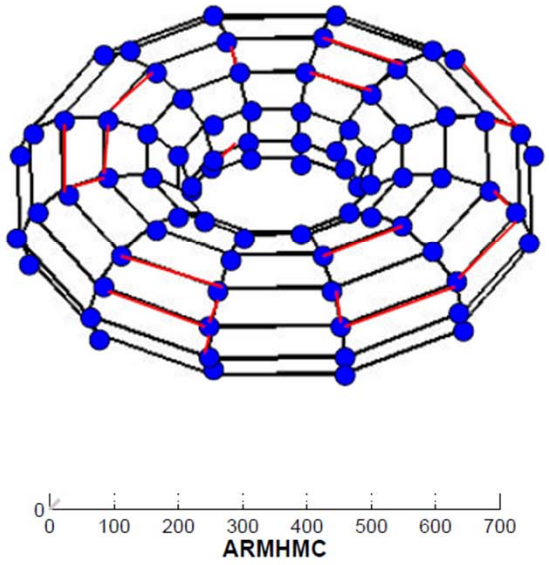
Automatic (Adaptive) Monte Carlo samplers

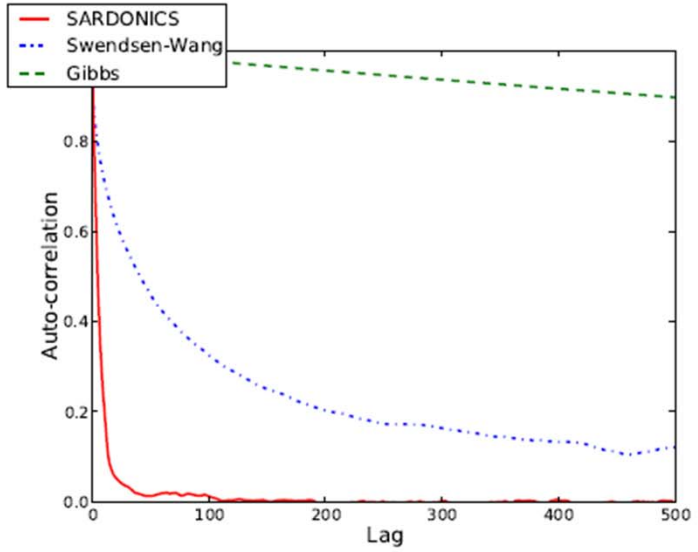
Method
Rios Insua and Muller's reversible-jump MCMC
Mackay's (1992) Gaussia with highest evidence
Neal's (1996) HMC
Neal's (1996) HMC with
Reversible-jump MCMC model by Andrieu et al.
Adaptive HMC (Median
Adaptive HMC (Mean Err





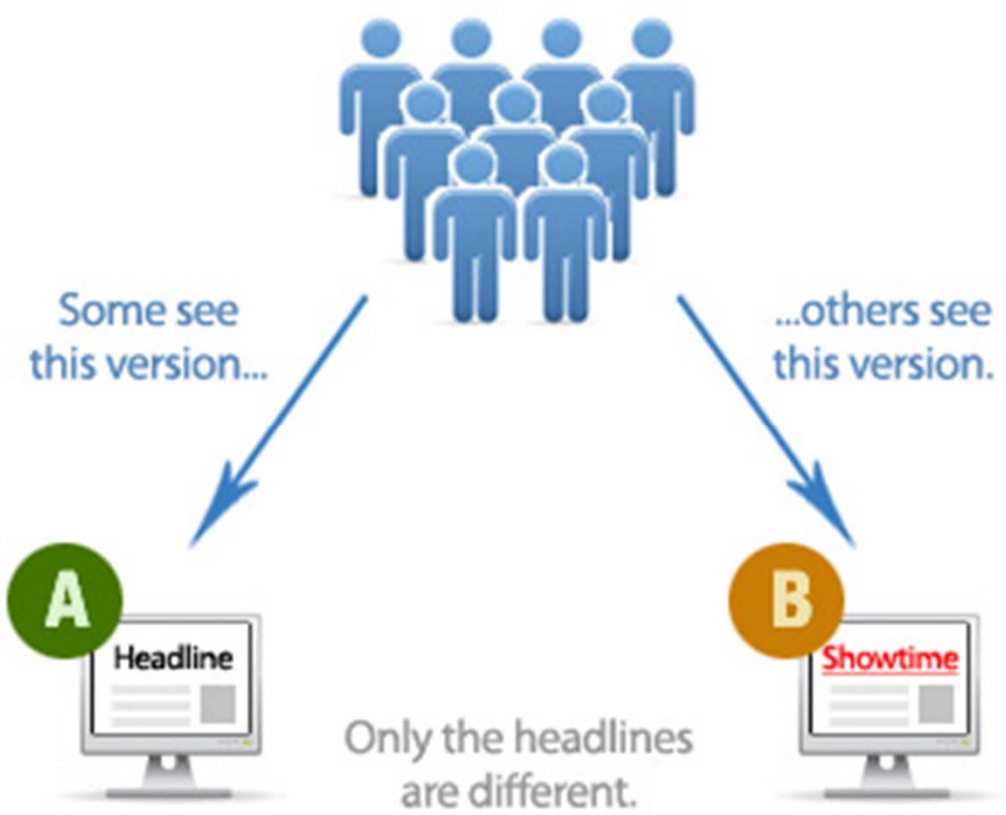






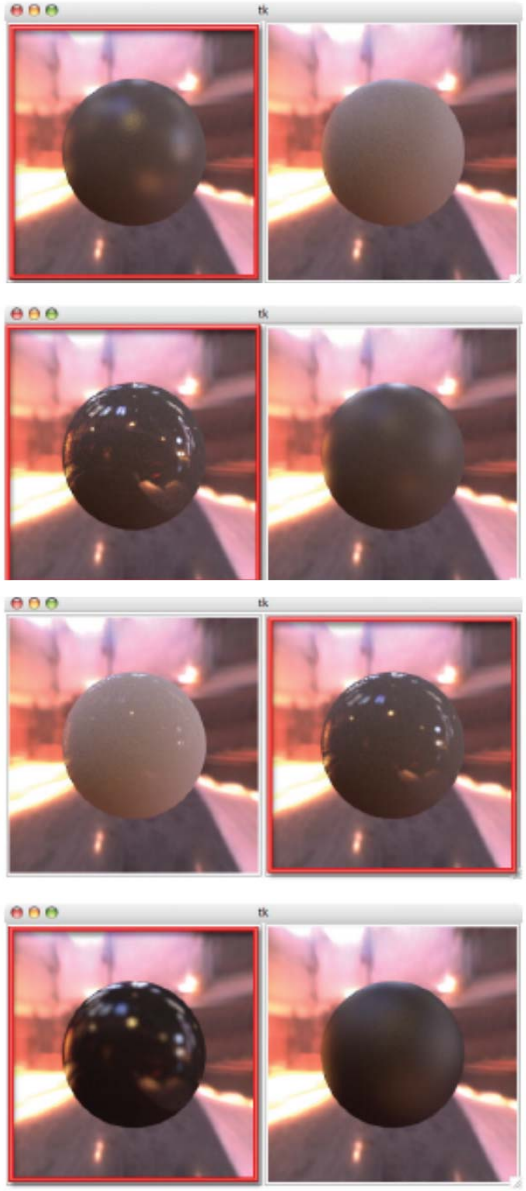
[Wang, Mohamed & dF, 2013]

Analytics, dynamic creative content and A/B testing



[Steve Scott on Bayesian bandits at Google]

animation session

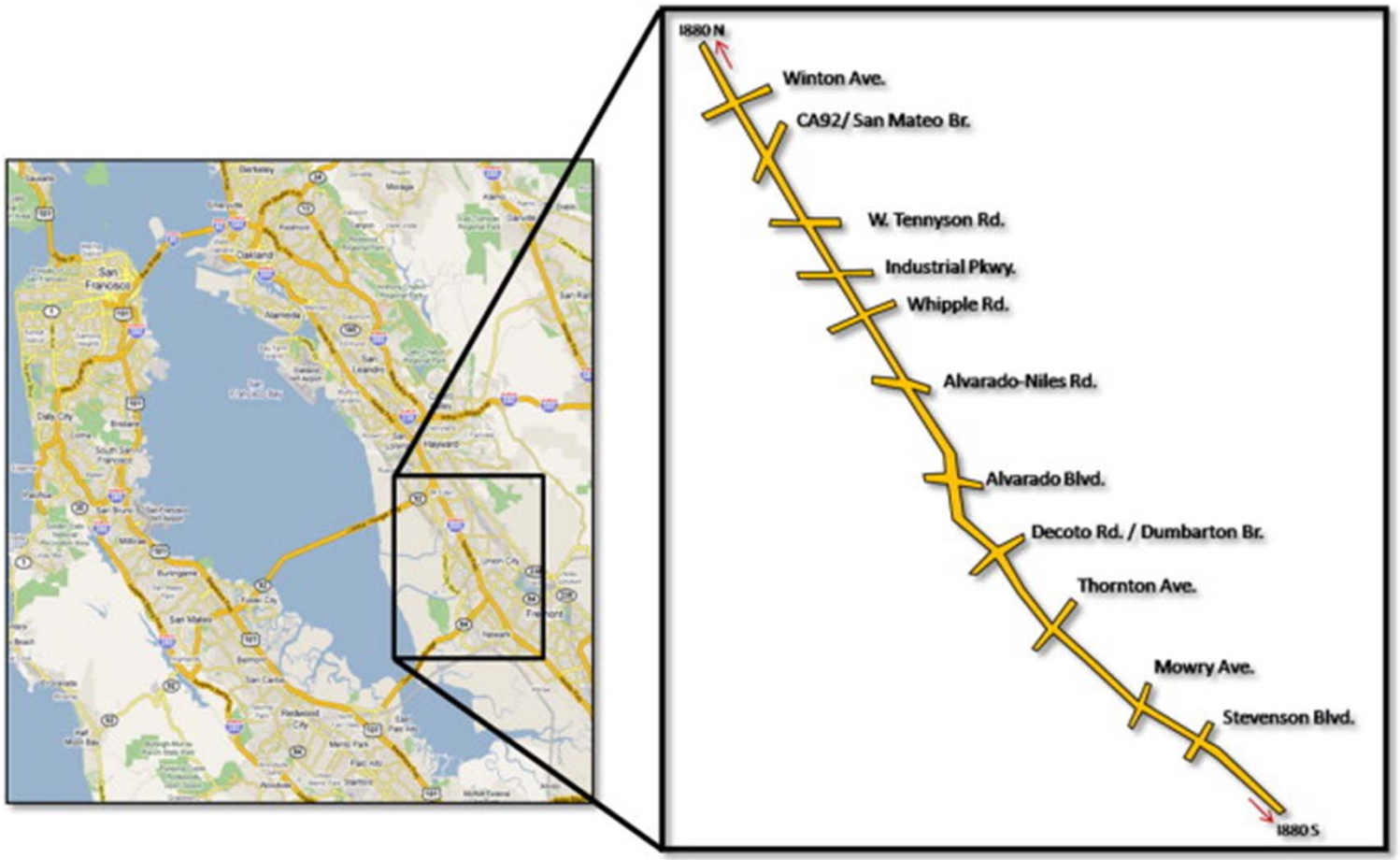


target

[Brochu, Ghosh & dF, 2007. Brochu, Brochu, dF, 2010]

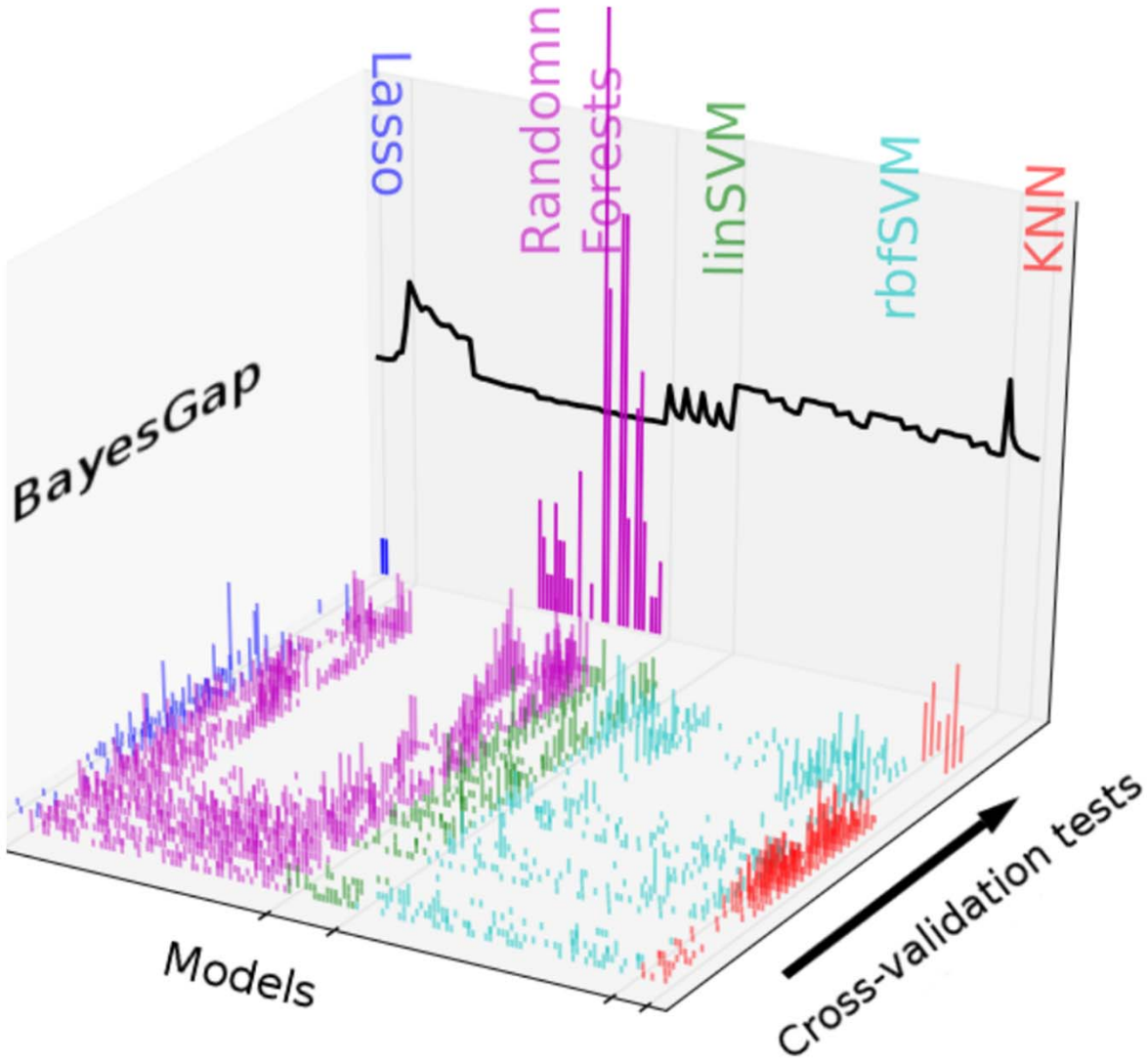
Winner of the SRC competition - SIGGRAPH

Sensor networks



[Andreas Krause et al]

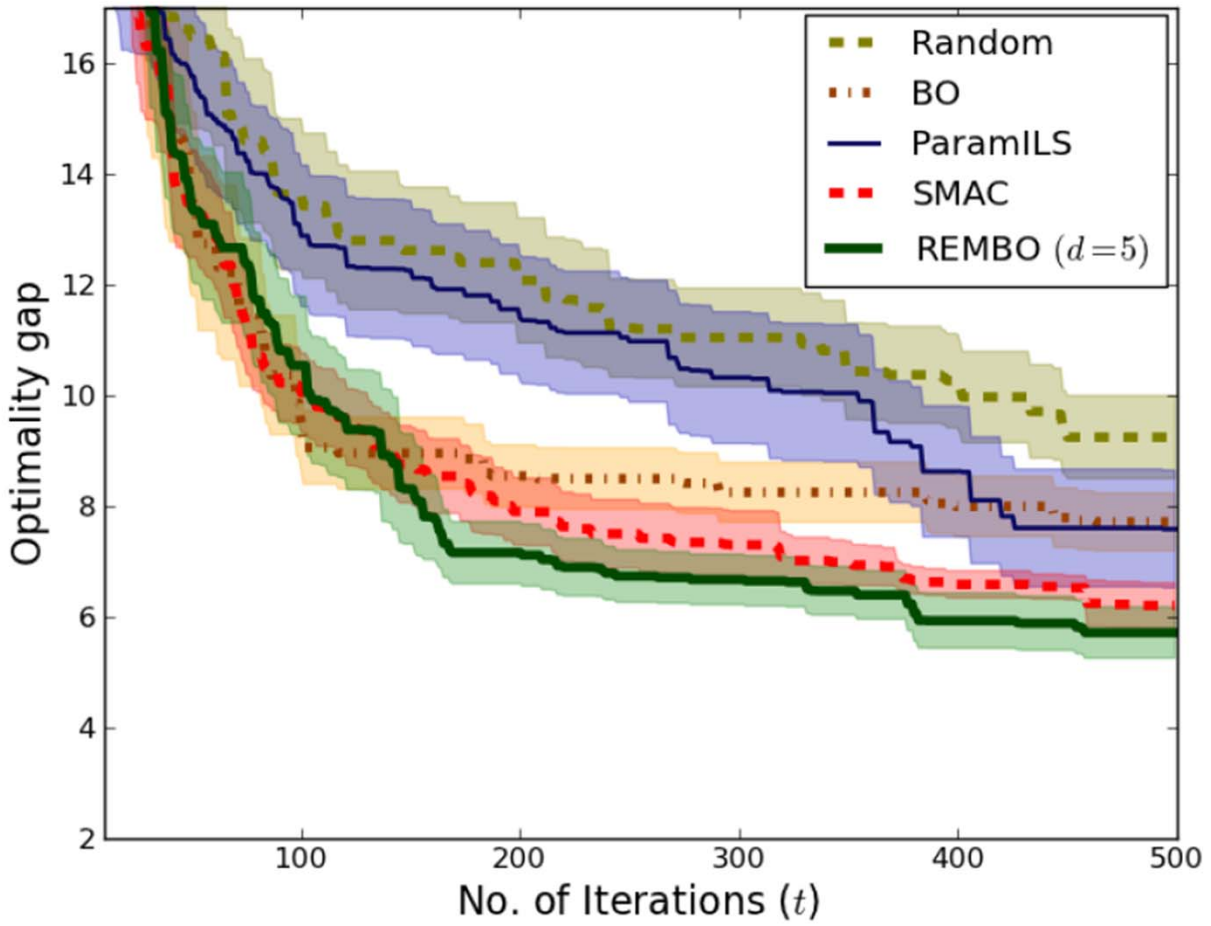
Automatic machine learning

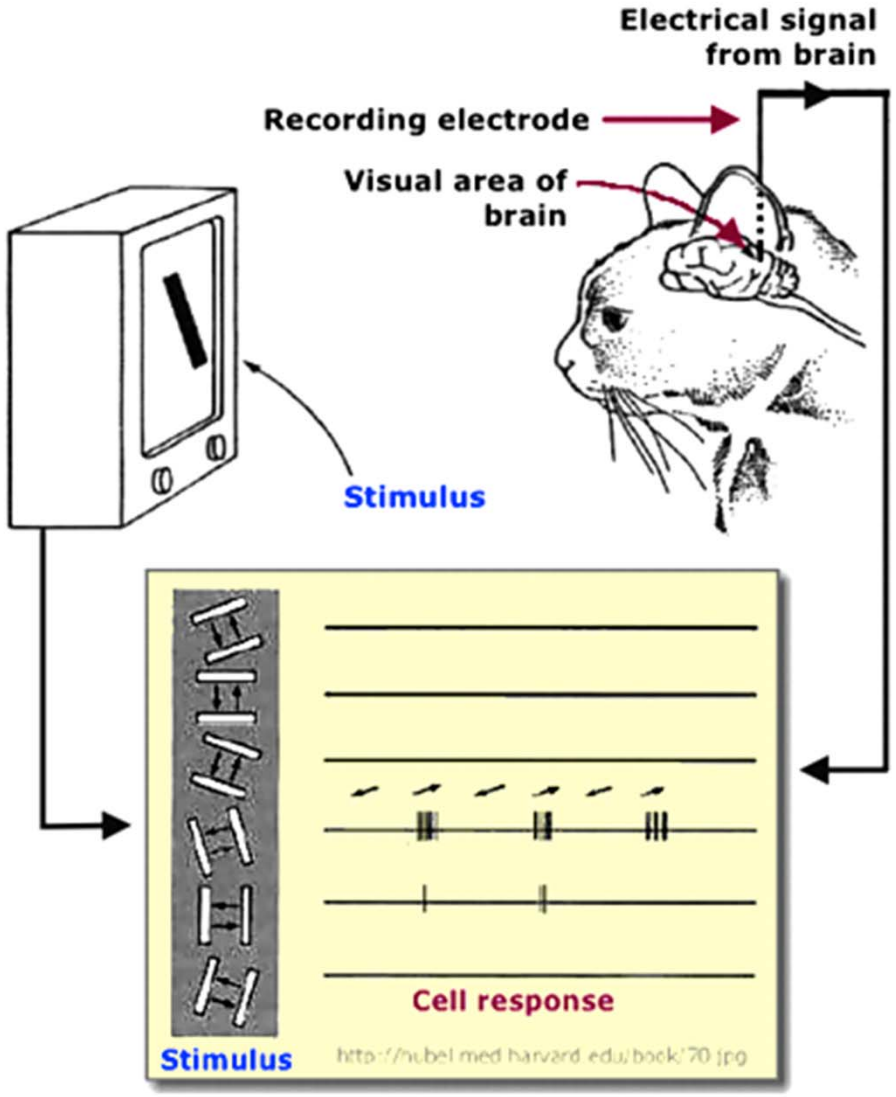


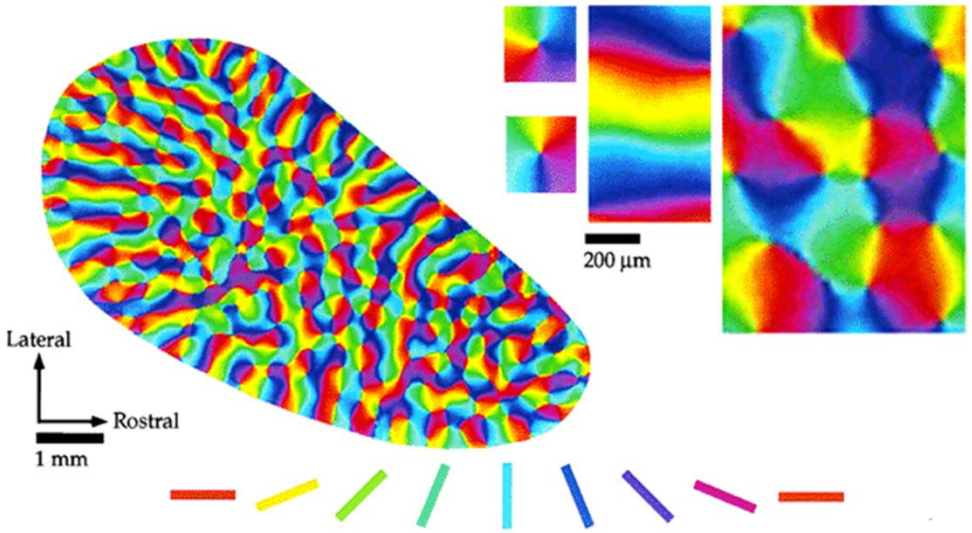
[Hoffman, Shahriari & dF, 2013]

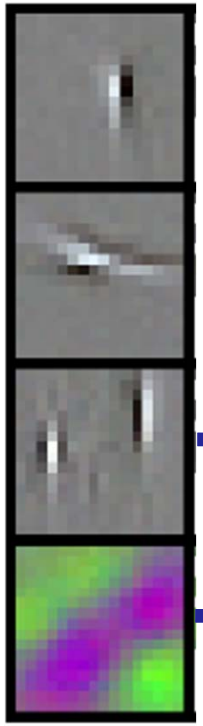
Tuning NP hard problem solvers

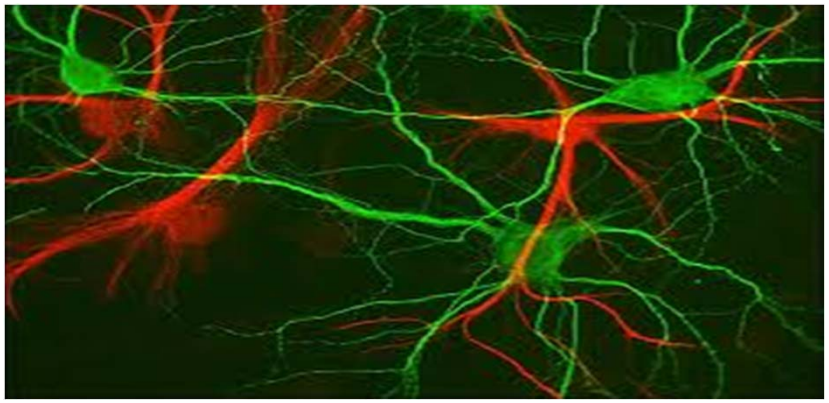
- **lpsolve** is a mixed integer programming solver, downloaded over 40,000 times last year.
- 47 discrete parameters (choices)

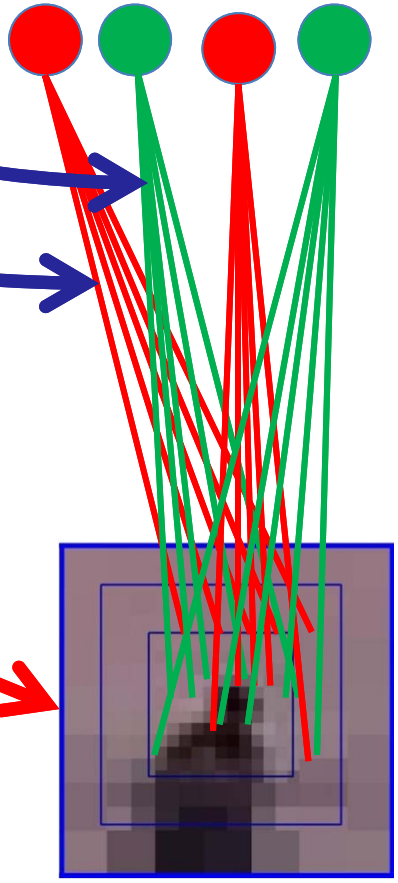


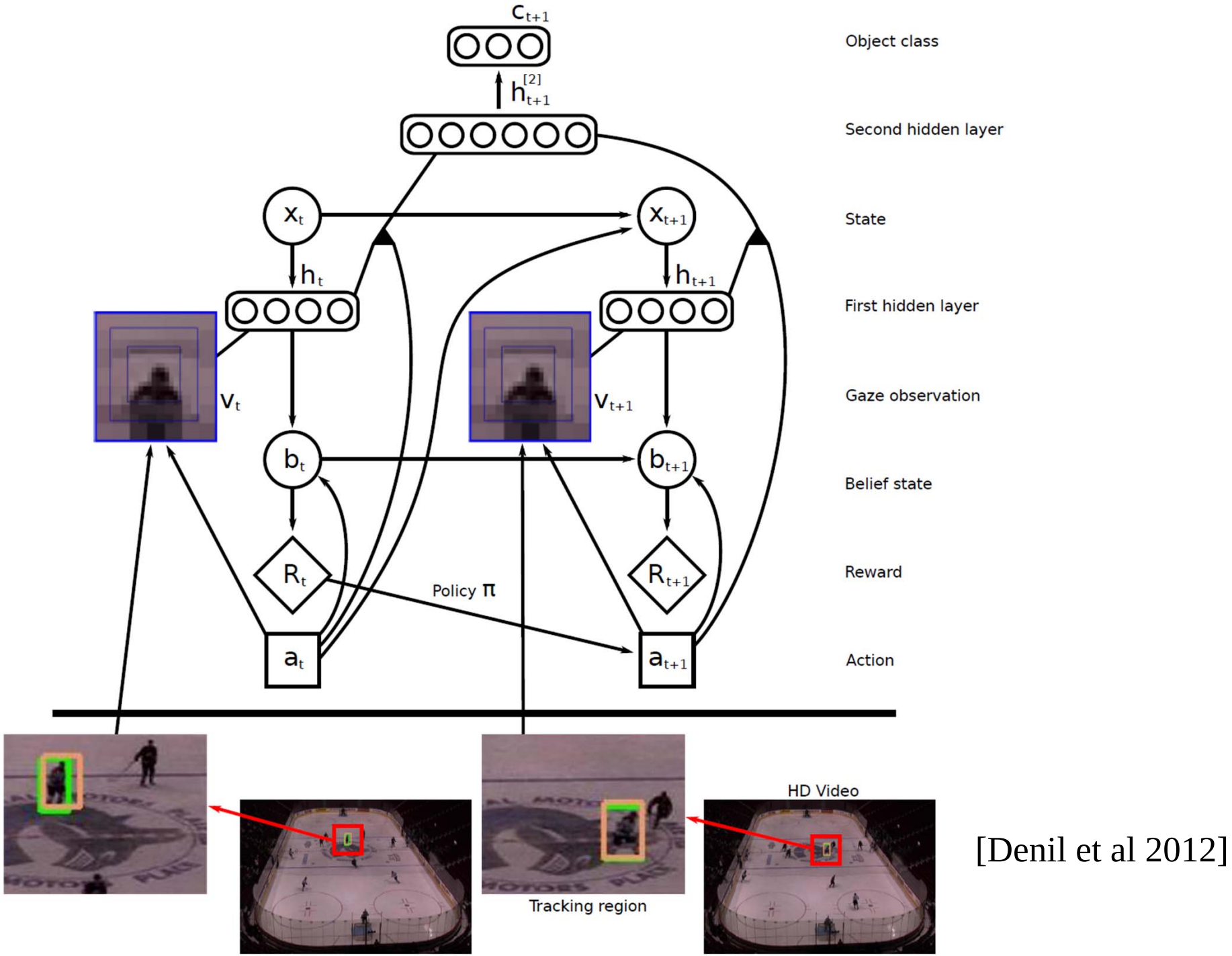

Electrical signal
from brain
Recording electrode
Visual area of
brain
Stimulus
Cell response
Stimulus
http://hubel.med.harvard.edu/book/f70.jpg


200 µm
Lateral
Rostral
1 mm





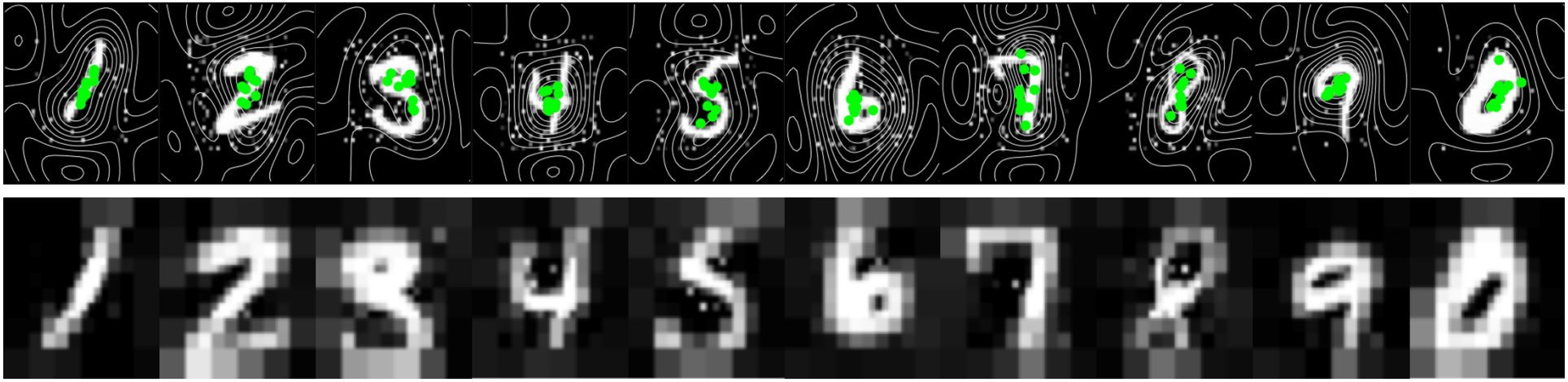




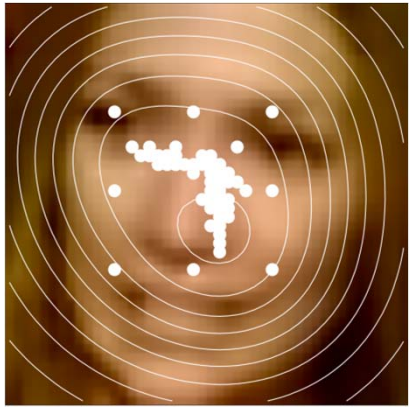
[Denil et al 2012]

GP Policy for tracking

Digits Experiment:



Face Experiment:

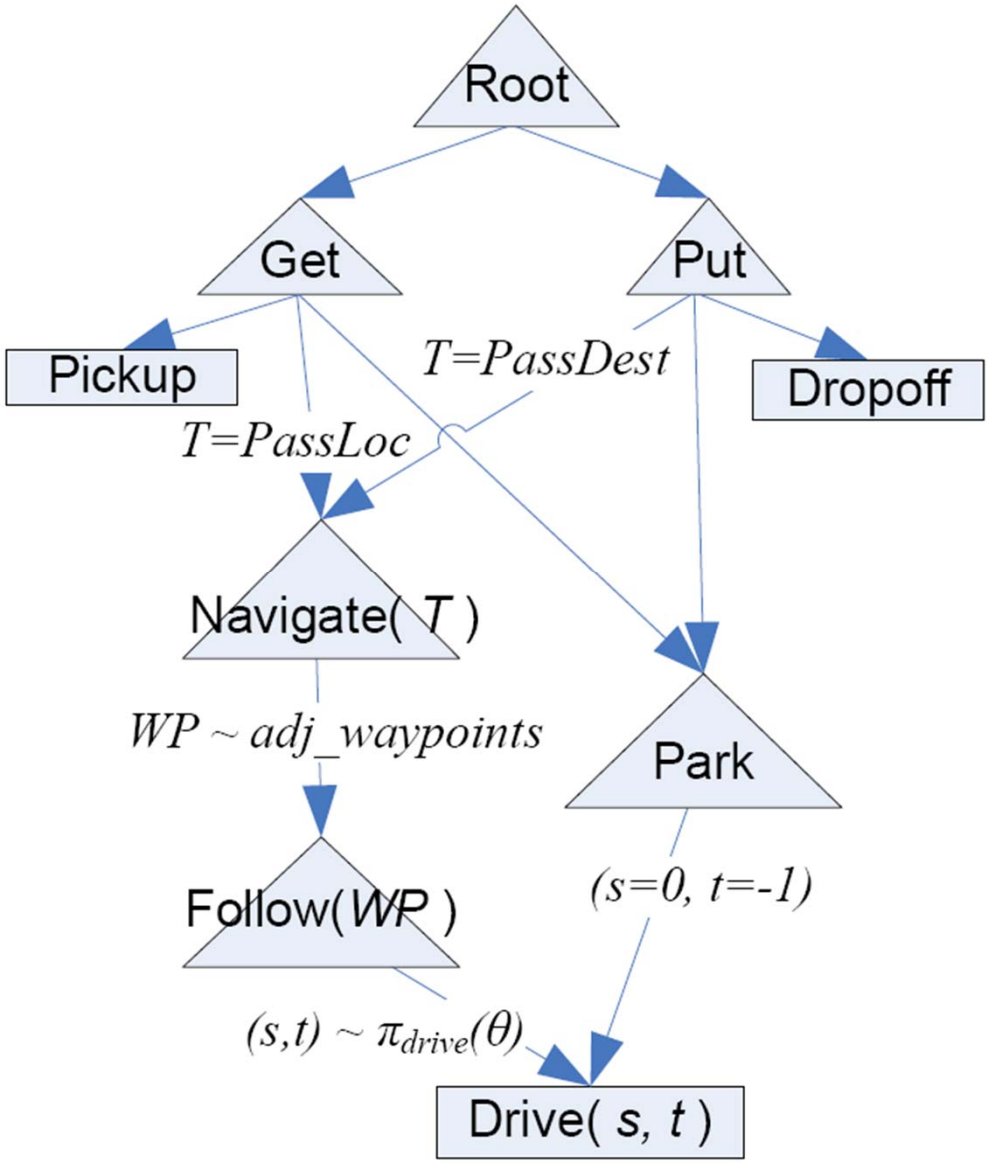


Hierarchical reinforcement learning

High-level model-based learning for deciding when to navigate, park, pickup and dropoff passengers.

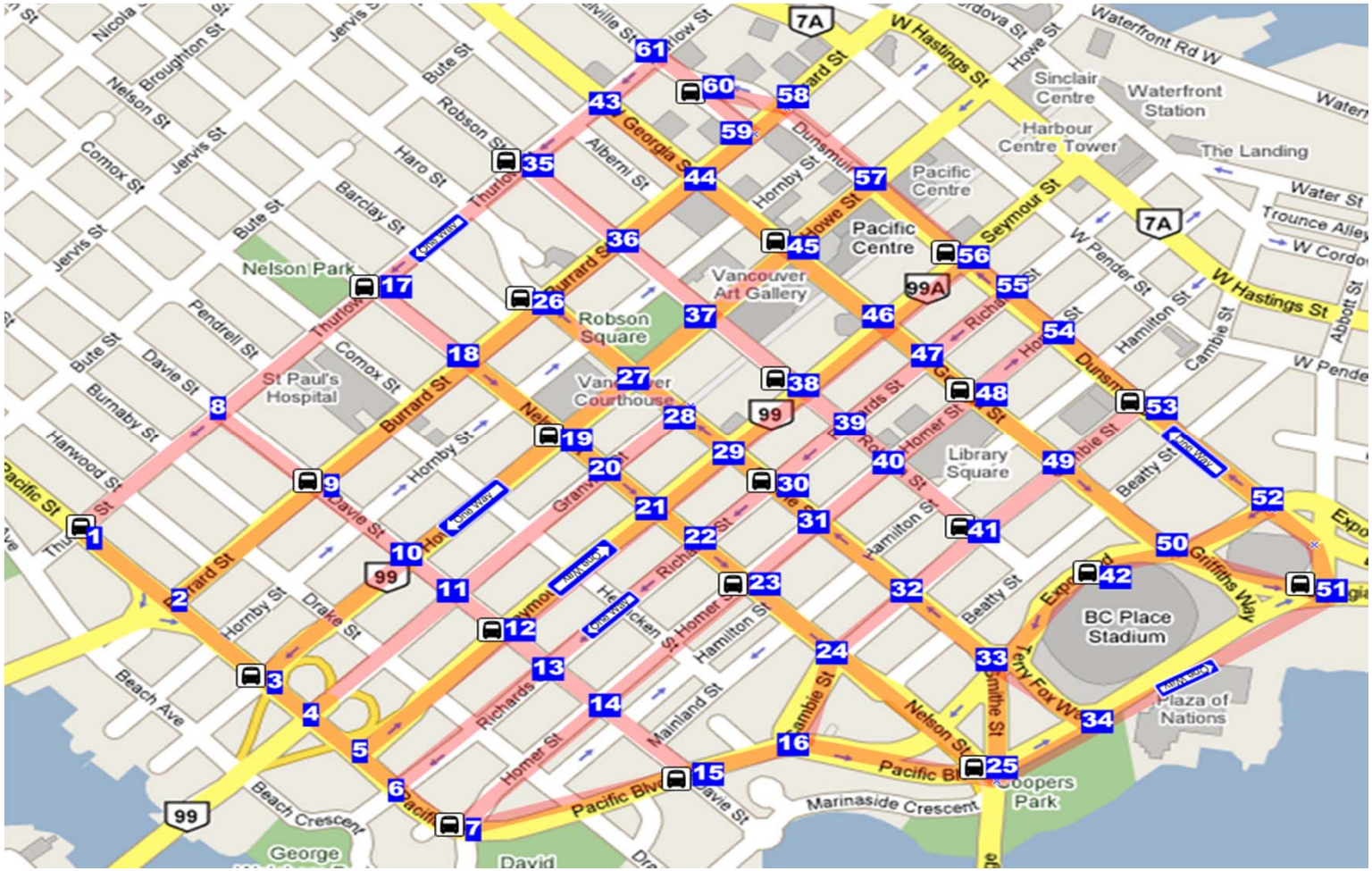
Mid-level active path learning for navigating a topological map.

Low-level active policy optimizer to learn control of continuous non-linear vehicle dynamics.



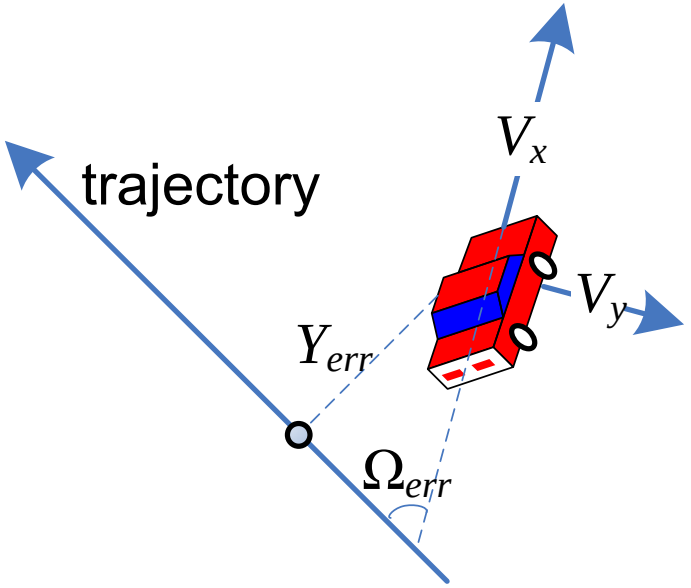
Active Path Finding in Middle Level

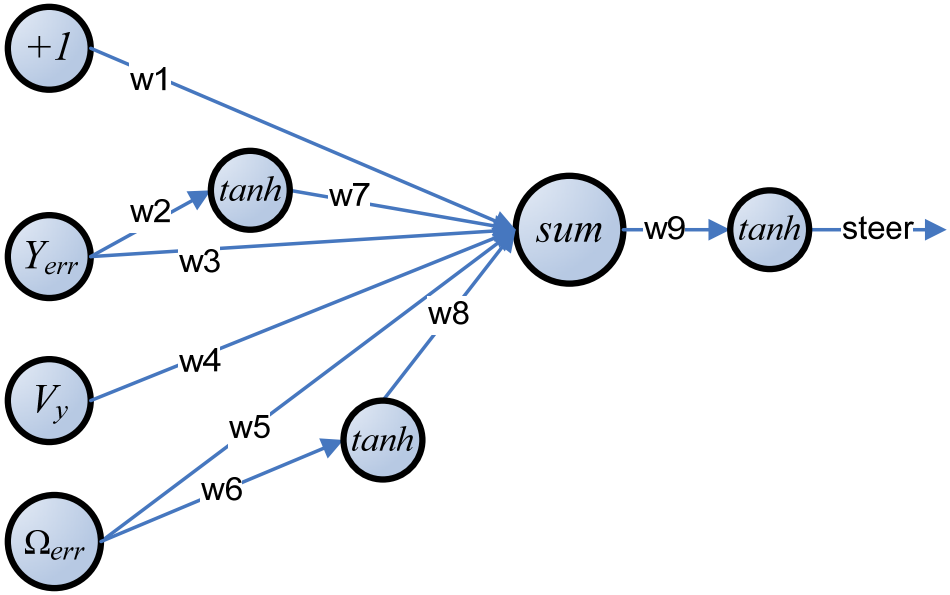
Navigate policy generates sequence of waypoints on a topological map to navigate from a location to a destination.

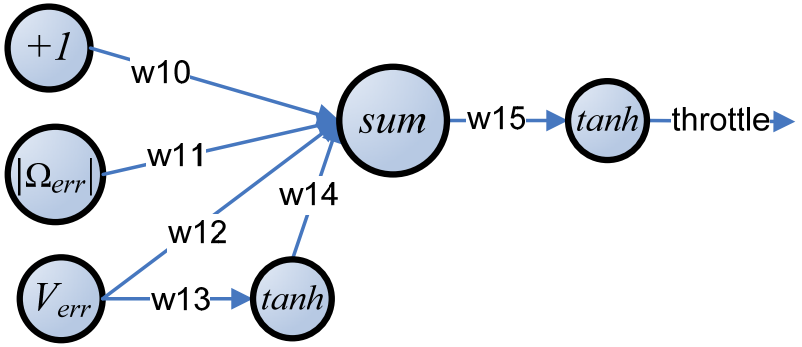


Low-Level: Trajectory following

TORCS: 3D game engine that implements complex vehicle dynamics complete with manual and automatic transmission, engine, clutch, tire, and suspension models.



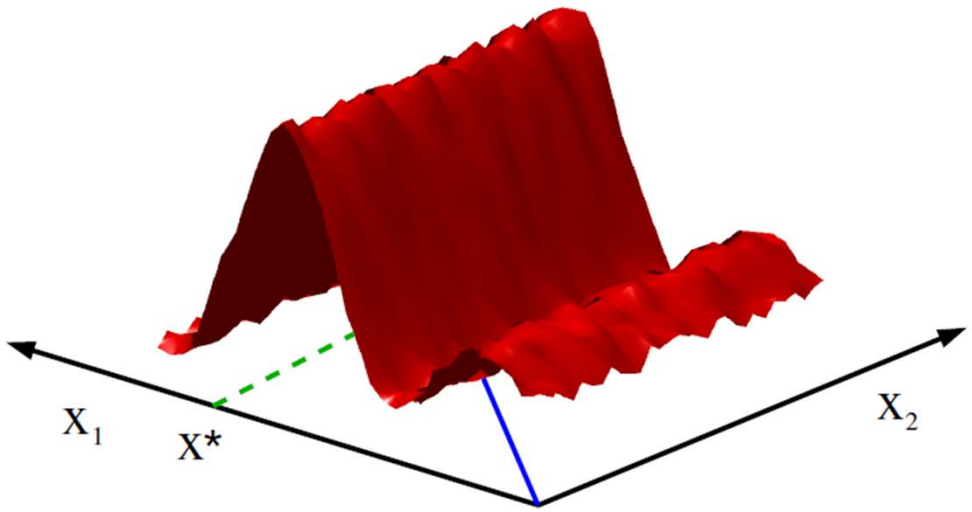


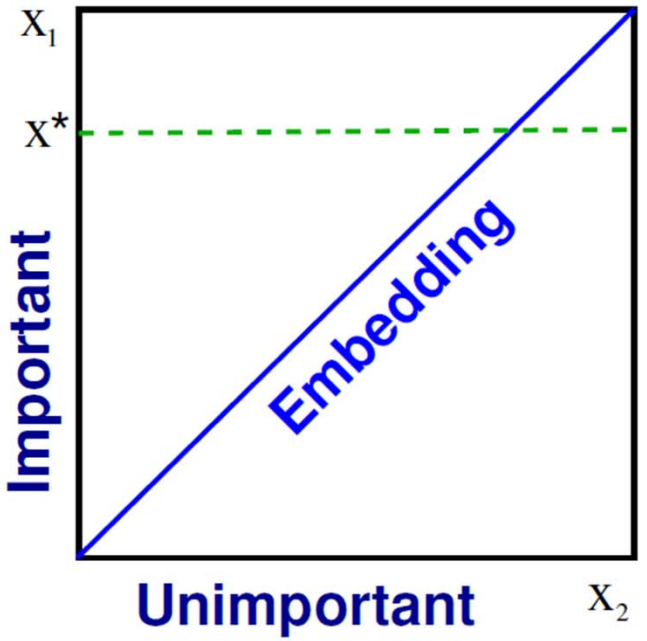


Bayesian optimization was used to find the neural net low-level policy and the value function for the upper levels

Random Embedding Bayesian Optimization

- **Embed** a low dimensional space into the high dimensional one
- Optimize only on the low dimensional space.





[Wang, Zoghi, Matheson, Hutter & dF,
IJCAI 2013 Distinguished paper award]

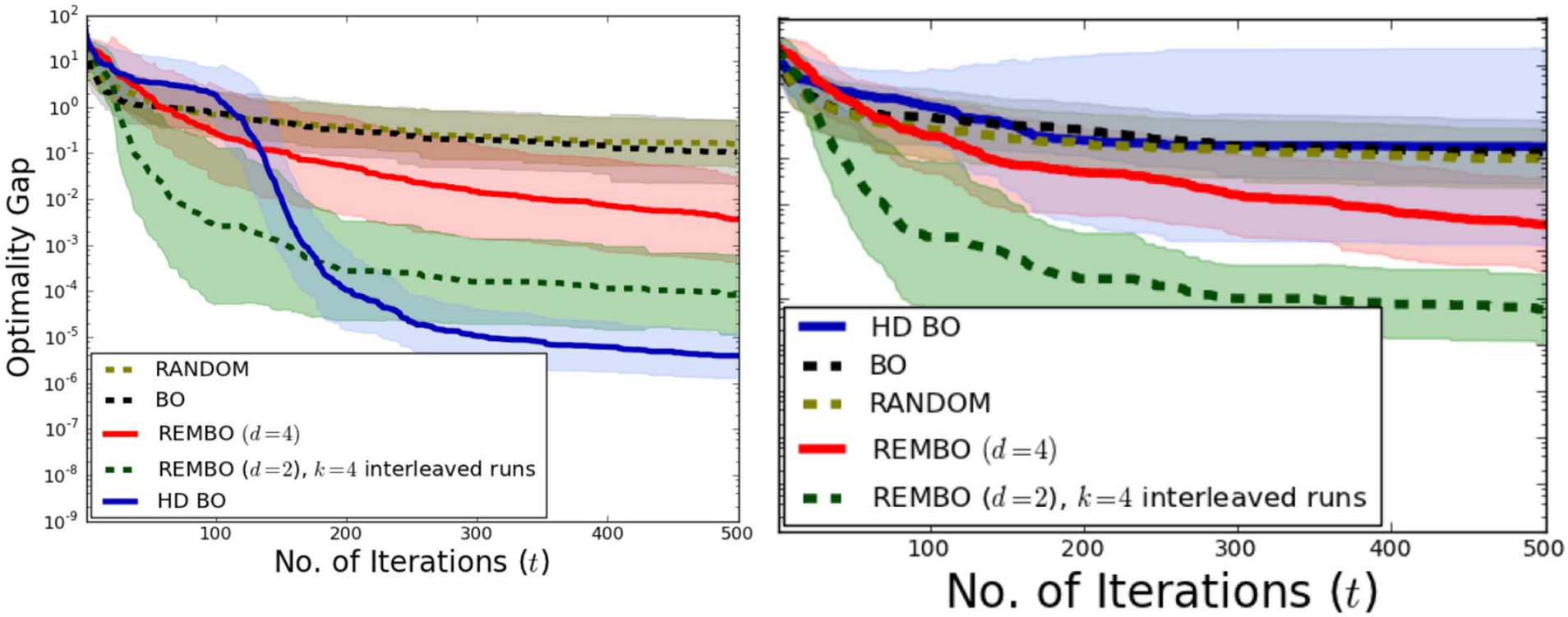
Algorithm 1 Bayesian Optimization

1: **for** $t = 1, 2, \ldots$ **do**
2: Find $\mathbf{x}_{t+1} \in \mathbb{R}^D$ by optimizing the acquisition function u: $\mathbf{x}_{t+1} = \arg\max_{\mathbf{x} \in \mathcal{X}} u(\mathbf{x}|\mathcal{D}_t)$.
3: Augment the data $\mathcal{D}_{t+1} = \{\mathcal{D}_t, (\mathbf{x}_{t+1}, f(\mathbf{x}_{t+1}))\}$
4: **end for**

Algorithm 2 REMBO: Bayesian Optimization with Random Embedding

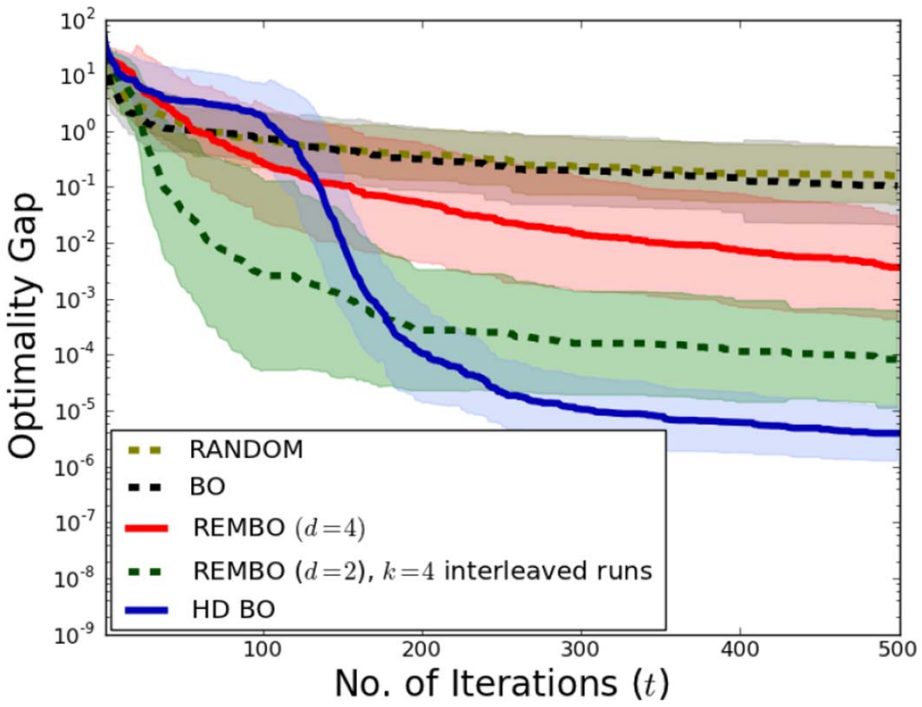
1: Generate a random matrix $\mathbf{A} \in \mathbb{R}^{D \times d}$
2: Choose the bounded region set $\mathcal{Y} \subset \mathbb{R}^d$
3: **for** $t = 1, 2, \ldots$ **do**
4: Find $\mathbf{y}_{t+1} \in \mathbb{R}^d$ by optimizing the acquisition function u: $\mathbf{y}_{t+1} = \arg\max_{\mathbf{y} \in \mathcal{Y}} u(\mathbf{y}|\mathcal{D}_t)$.
5: Augment the data $\mathcal{D}_{t+1} = \{\mathcal{D}_t, (\mathbf{y}_{t+1}, f(\mathbf{A}\mathbf{y}_{t+1})\}$
6: Update the kernel hyper-parameters.
7: **end for**

Rotation invariance

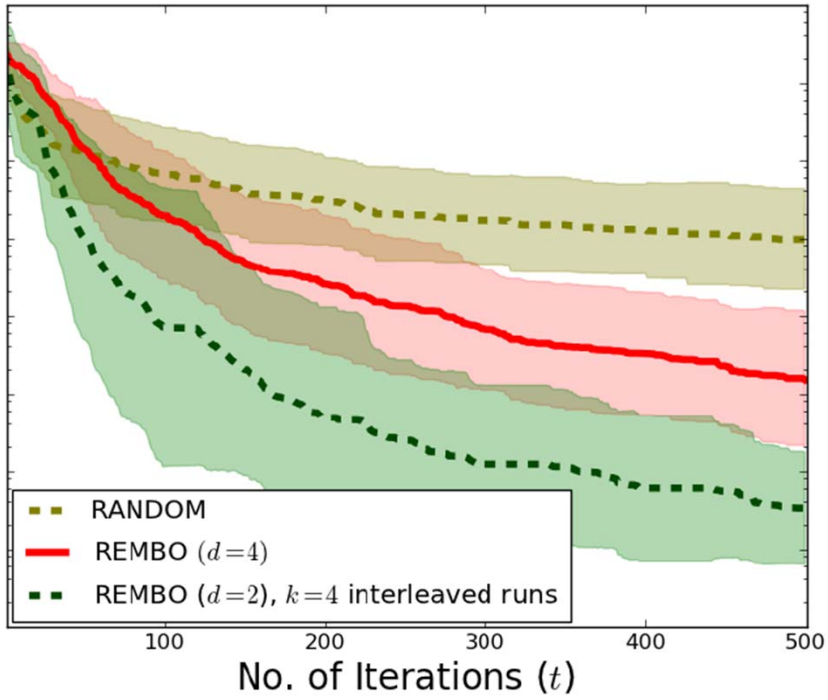


$$D = 25$$

Scaling to over a billion dimensions



$D = 25$



$D = 1\,000\,000\,000$


Thank you
Next lecture: Intro to RL