Sample Efficient Policy Search

Emma Brunskill

15-889e

Fall 2015

Sample Efficient RL

- Objectives
  - Probably Approximately Correct
  - Minimizing regret
  - Bayes-optimal RL
  - **Focus today: Empirical performance**

Last Time: Policy Search Using Gradients

- Gradient approaches only guaranteed to find a local optima
- Finite-difference methods scale with # of parameters needed to represent the policy, but don't require differentiable policy
- Likelihood ratio gradient approaches
  - **Require us to be able to compute gradient analytically**
  - Cost independent of # params
  - Don't need to know dynamics model
  - Benefit from using value function/baseline to reduce variance

Question from Last Class

Theorem (Compatible Function Approximation Theorem)

If the following two conditions are satisfied:

1. *Value function approximator is compatible to the policy*

$$\nabla_w Q_w(s,a) = \nabla_\theta \log \pi_\theta(s,a)$$

2. *Value function parameters w minimise the mean-squared error*

$$\varepsilon = \mathbb{E}_{\pi_\theta}\left[\left(Q^{\pi_\theta}(s,a) - Q_w(s,a)\right)^2\right]$$

Then the policy gradient is exact,

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\ Q_w(s,a)\right]$$

Figure from David Silver

How to Choose a Compatible Value Function Approximator? Ex.

Policy parameterization:

$$\pi(s,a) = \frac{e^{\theta^T \phi_{sa}}}{\sum_b e^{\theta^T \phi_{sb}}}, \qquad \forall s \in \mathcal{S}, s \in \mathcal{A},$$

Compatibility requires that:

$$\frac{\partial f_w(s,a)}{\partial w} = \frac{\partial \pi(s,a)}{\partial \theta} \frac{1}{\pi(s,a)} = \phi_{sa} - \sum_b \pi(s,b)\phi_{sb},$$

Therefore a reasonable choice for value function is

$$f_w(s,a) = w^T \left[\phi_{sa} - \sum_b \pi(s,b)\phi_{sb} \right]$$

Example from Sutton, McAllester, SIngh & Mansour
NIPS 2000

Today: Sample Efficient Policy Search

- Powerful function approximators to represent policy value
  - May not be easy to take derivative
- Like last time, may benefit from exploiting structure (e.g. not completely blackbox optimization

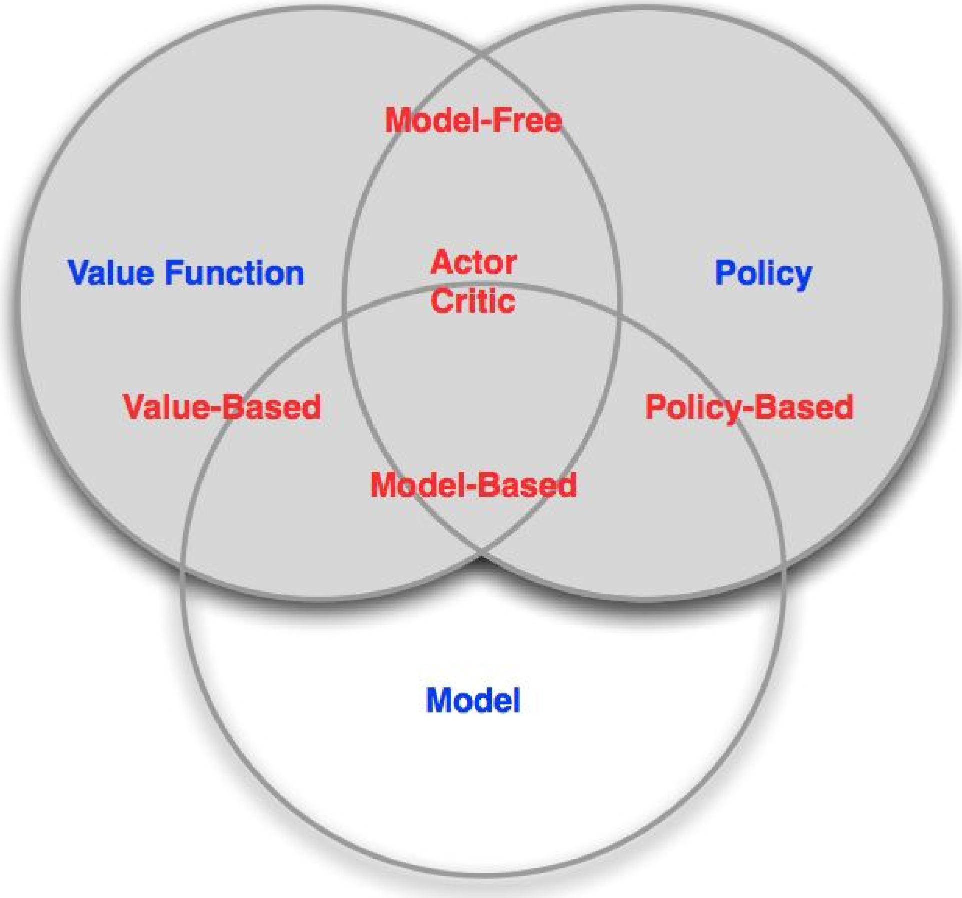


Figure from David Silver

Recall: Gaussian Process to Represent MDP Dynamics/Reward Models

$s' = \Delta + s$

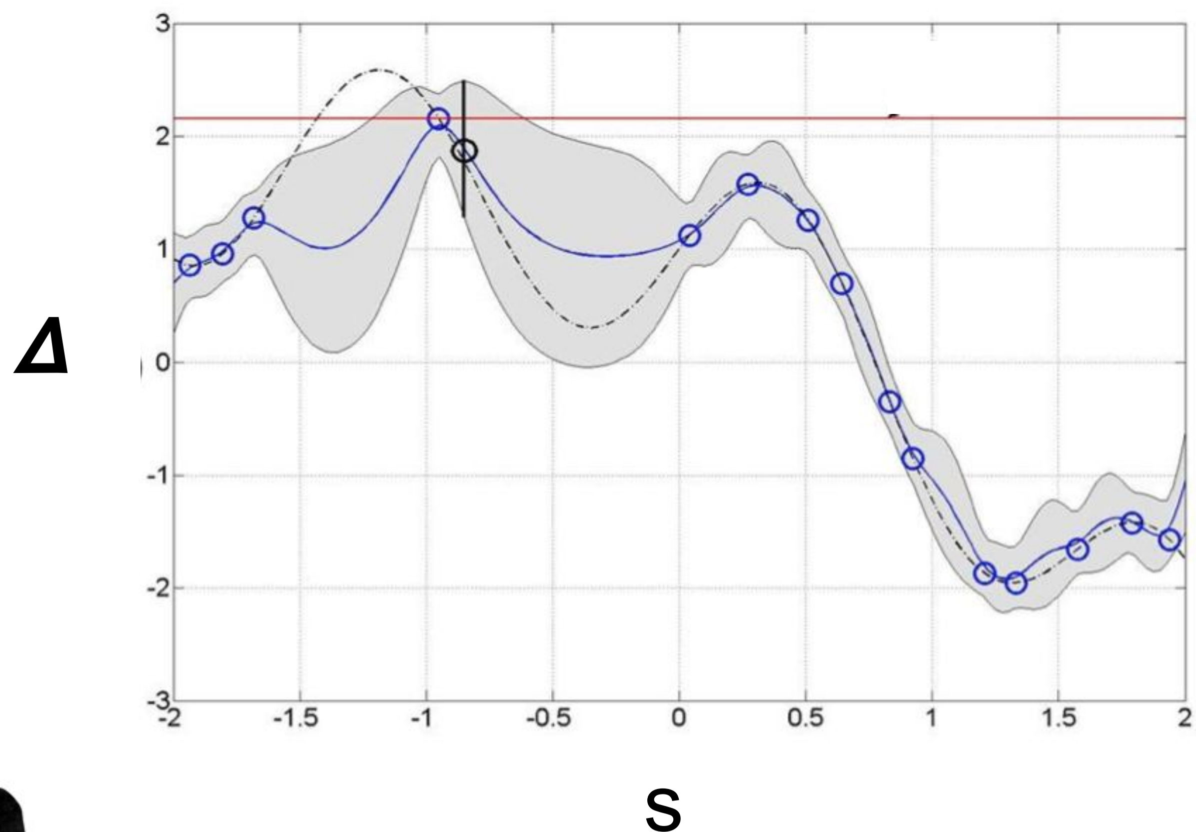


Figure adjusted from Wilson et al. JMLR 2014

Today: Gaussian Process to Represent Value of Parameterized Policy

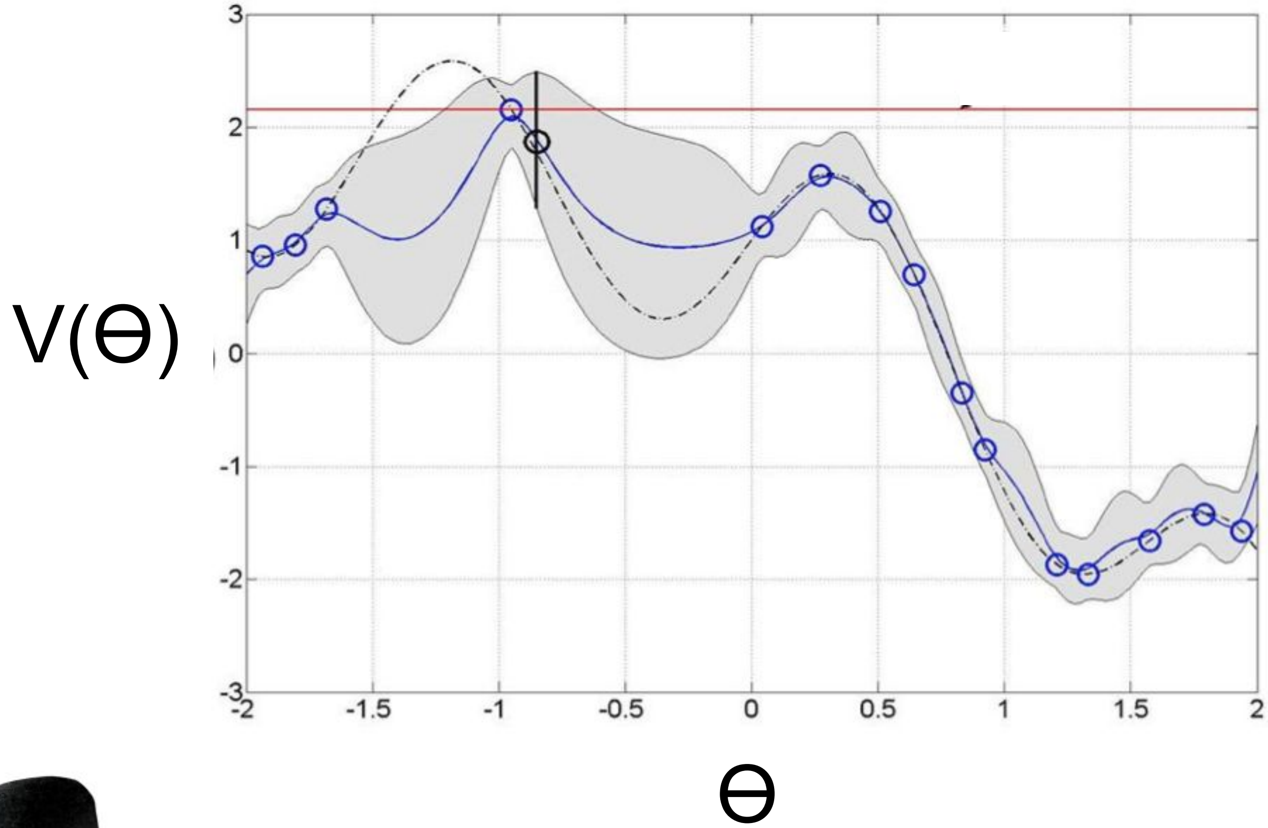


Figure adjusted from Wilson et al. JMLR 2014

Now Generalizing Policy Value (Rather than Model Dynamics/Rewards)

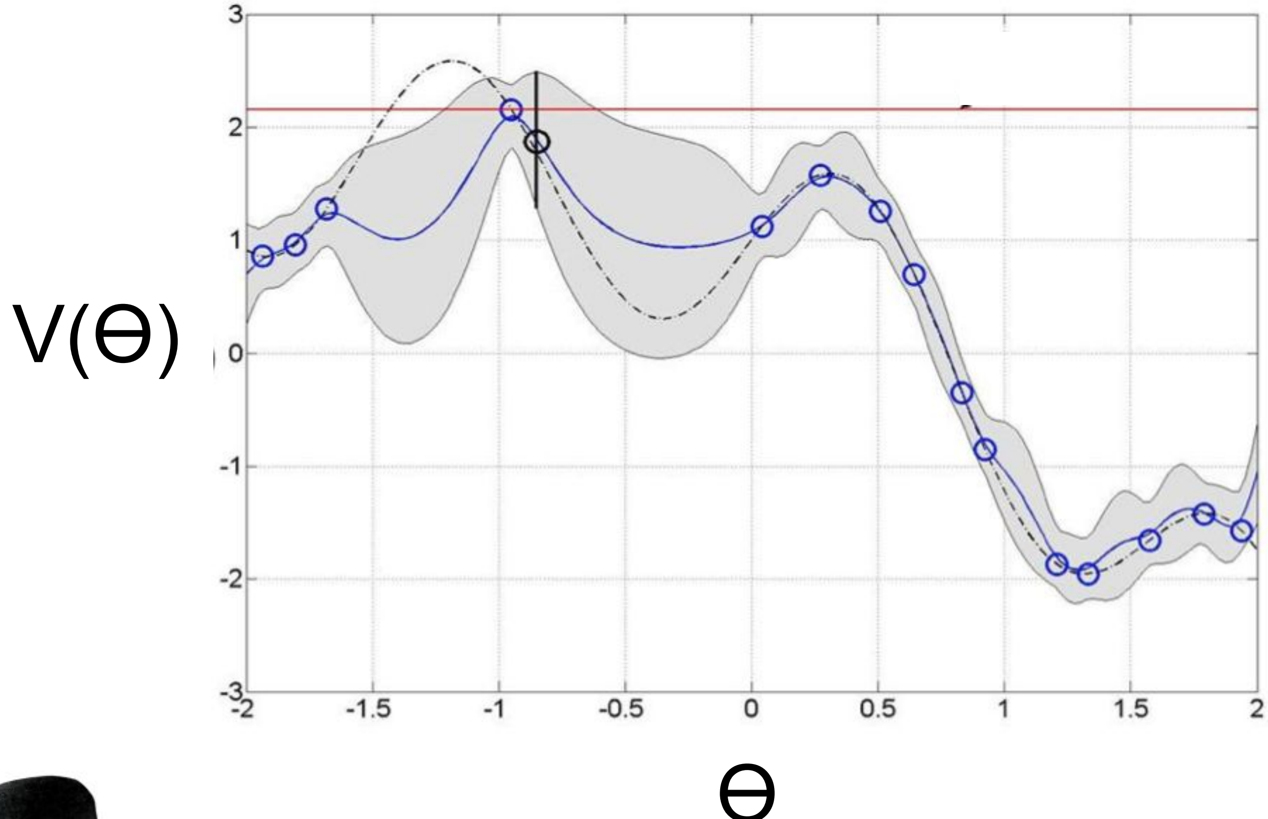


Figure adjusted from Wilson et al. JMLR 2014

Why Use GPs?

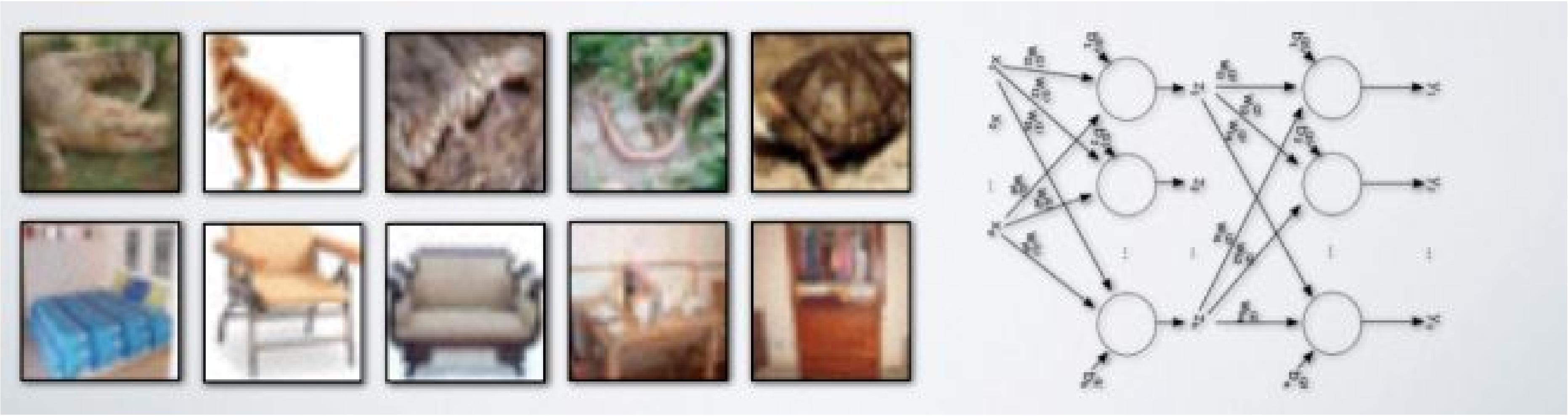
- Used frequently in Bayesian optimization
- Last ~5 years Bayesian optimization has become very influential & useful
- Brief (relevant) digression
- Two big motivations for Bayesian optimization

Motivation 1: ML Parameter Tuning

- ML methods getting more powerful… and complex
- Sophisticated fitting methods
- Often involve tuning many parameters
- Hard for non experts to use
- Performance often substantially impacted by choice of parameters

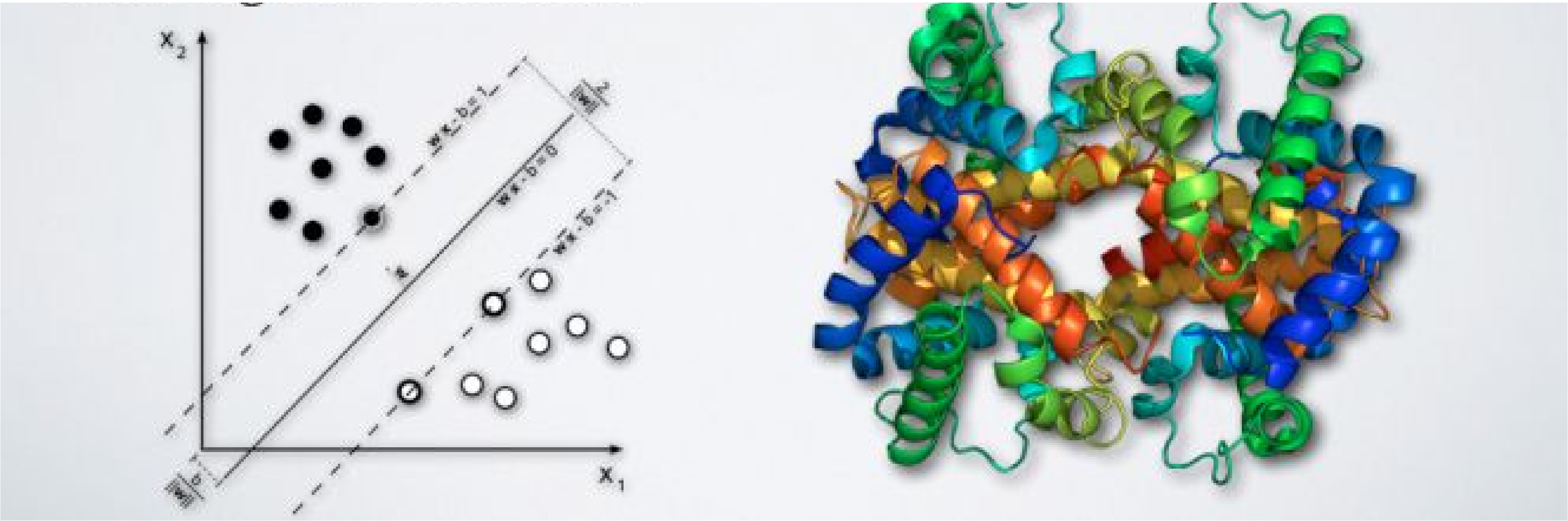
Material from Ryan Adams

Deep Learning



- Big investments by Google, Facebook, Microsoft, etc.
- Many choices: number of layers, weight regularization, layer size, which nonlinearity, batch size, learning rate schedule, stopping conditions

Classification of DNA Sequences

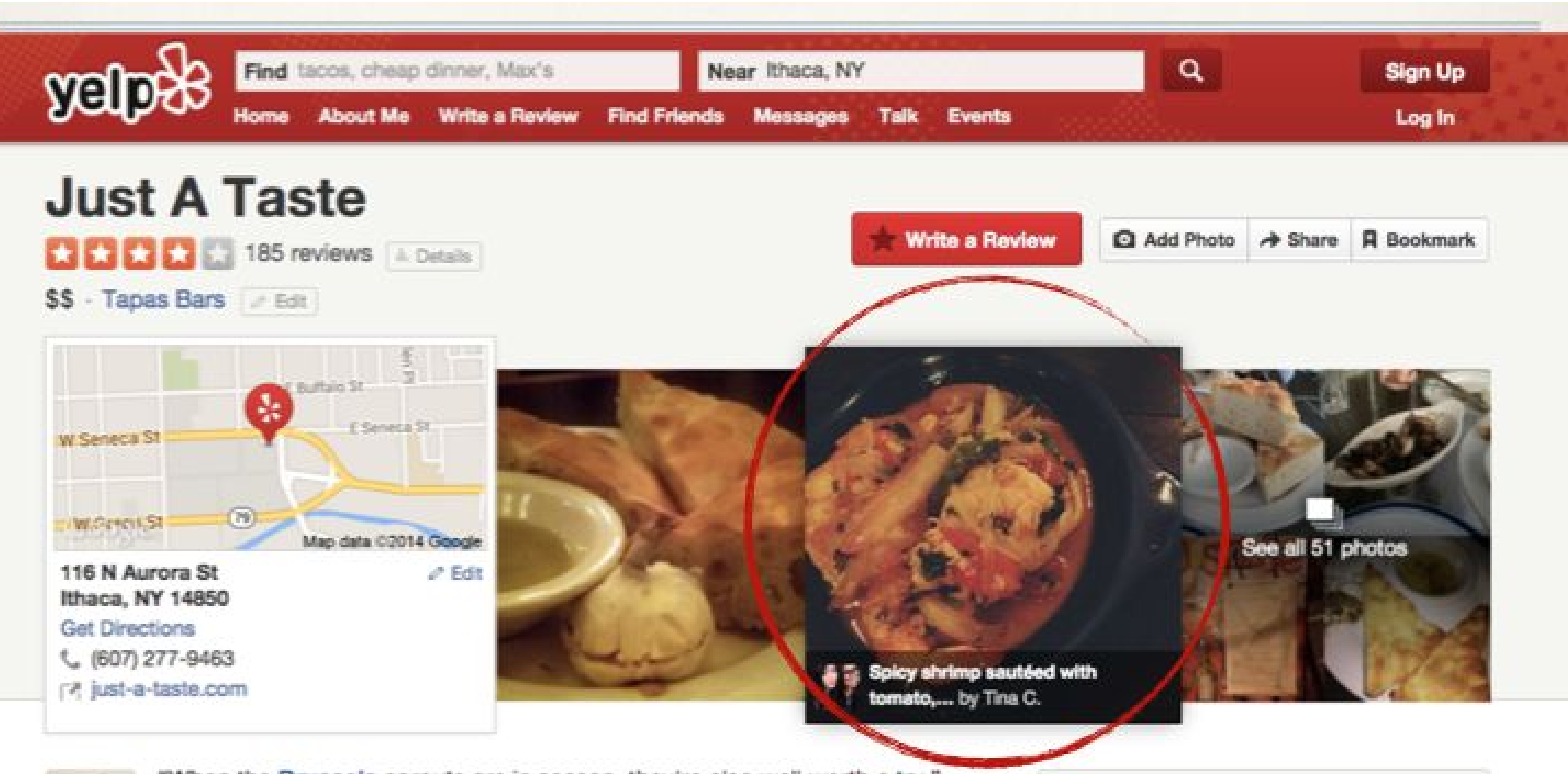


- Predict which DNA sequences will bind with which proteins, Miller et al. (2012)
- Choices: margin param, entropy param, converg. criterion

Motivation 2: Increasing Instances Which Blur Experimentation & Optimization

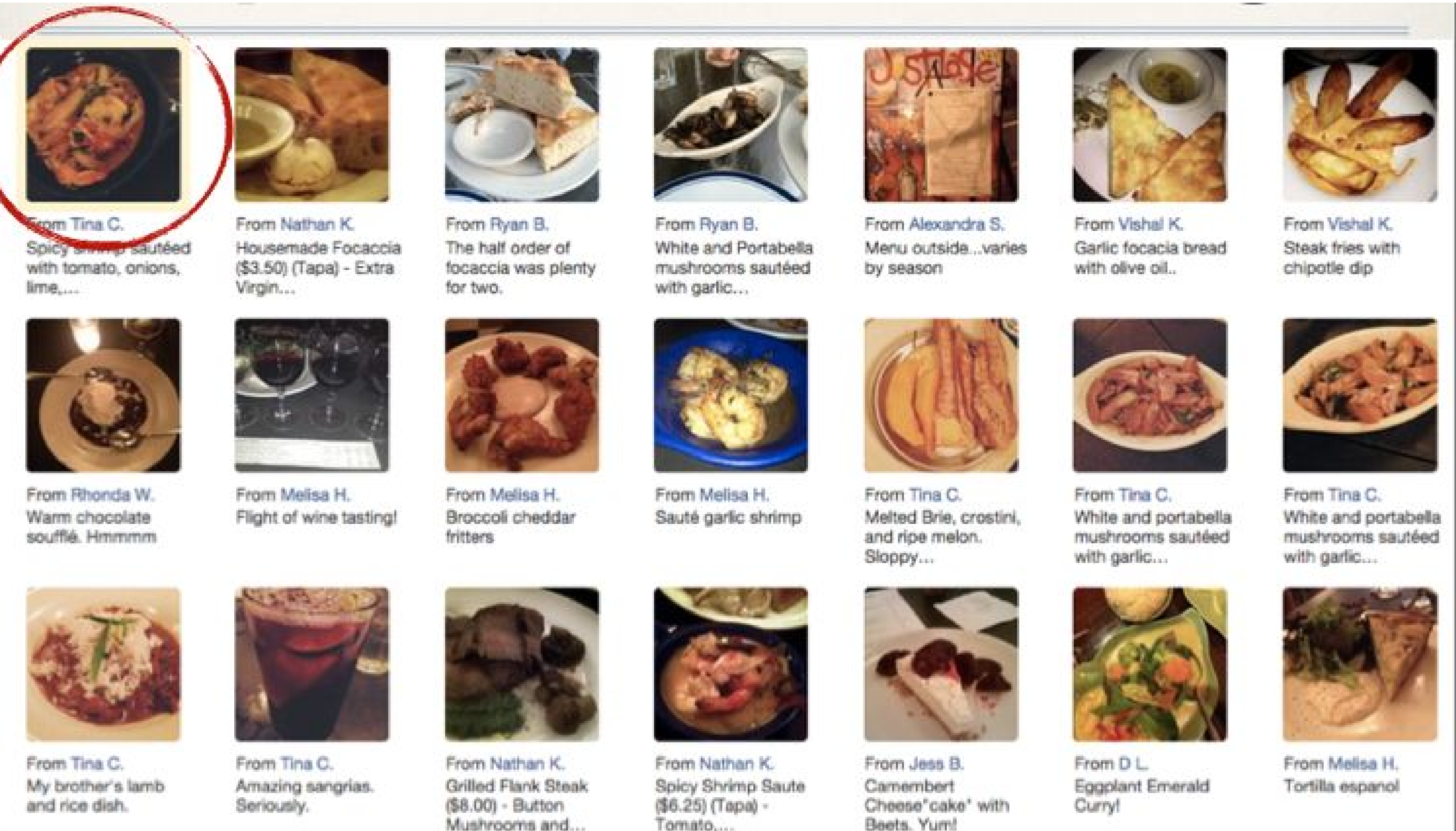
•

Website Design
Ex. What Photo to Put Here?

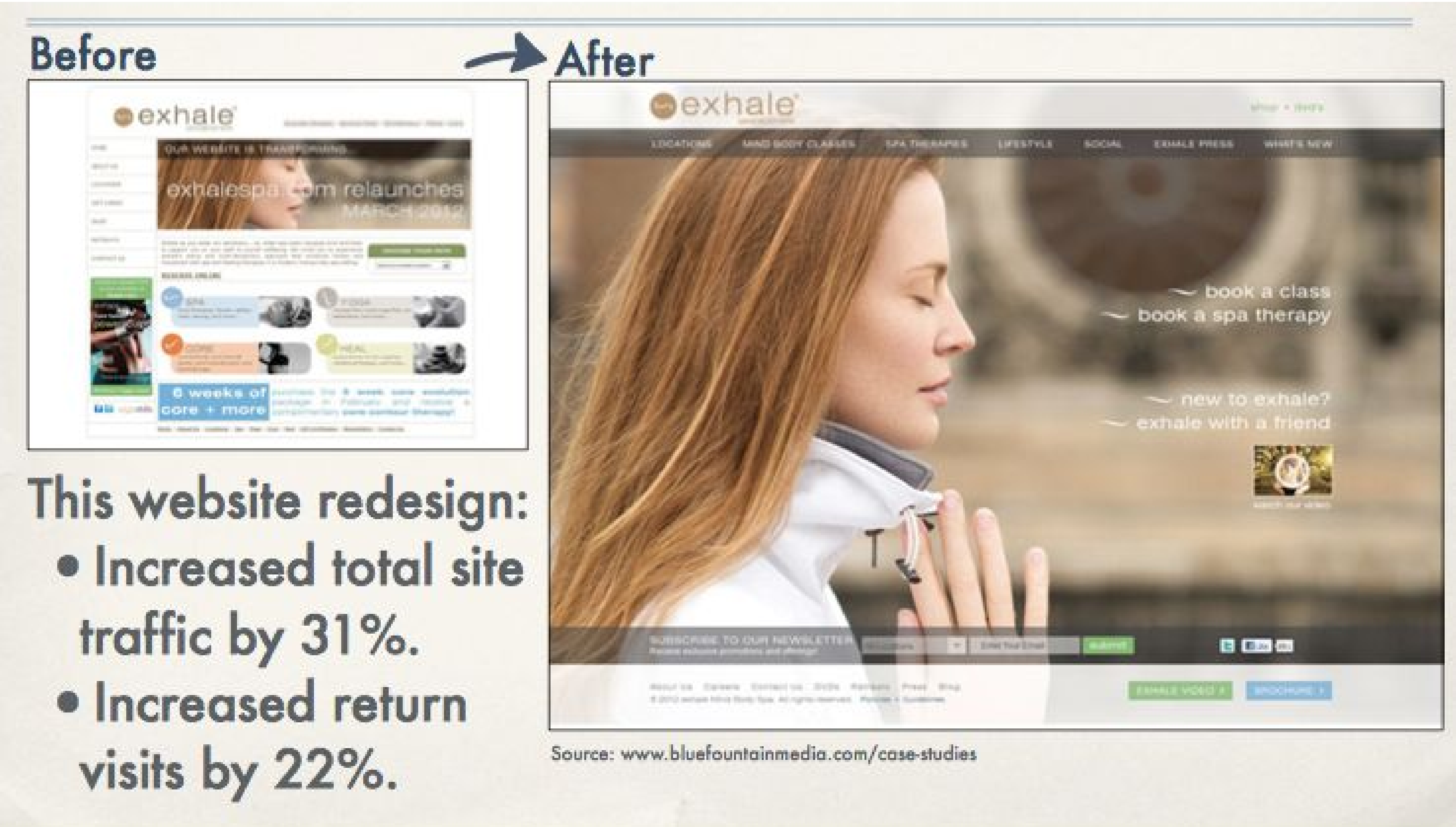


Material from Peter Frazier

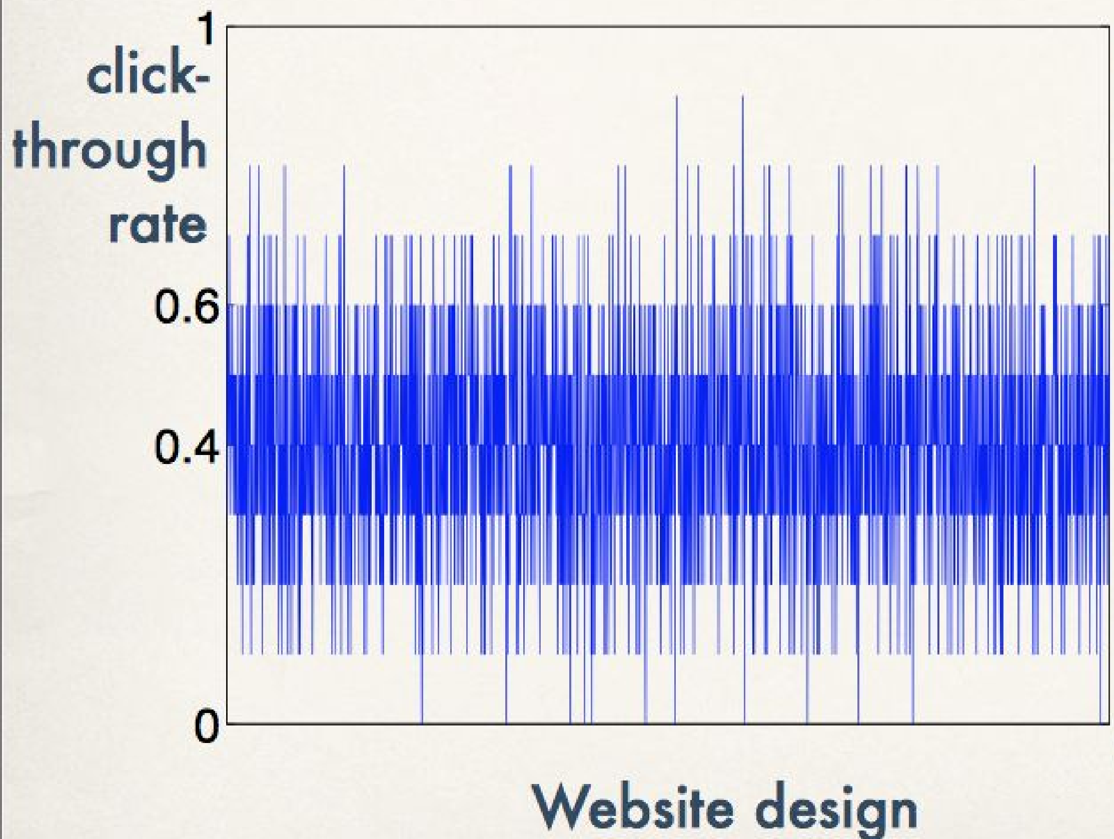
6 Photos on Website: 51*50*49*48*47*46= 12,966,811,200 Options!



Design Choices Matter



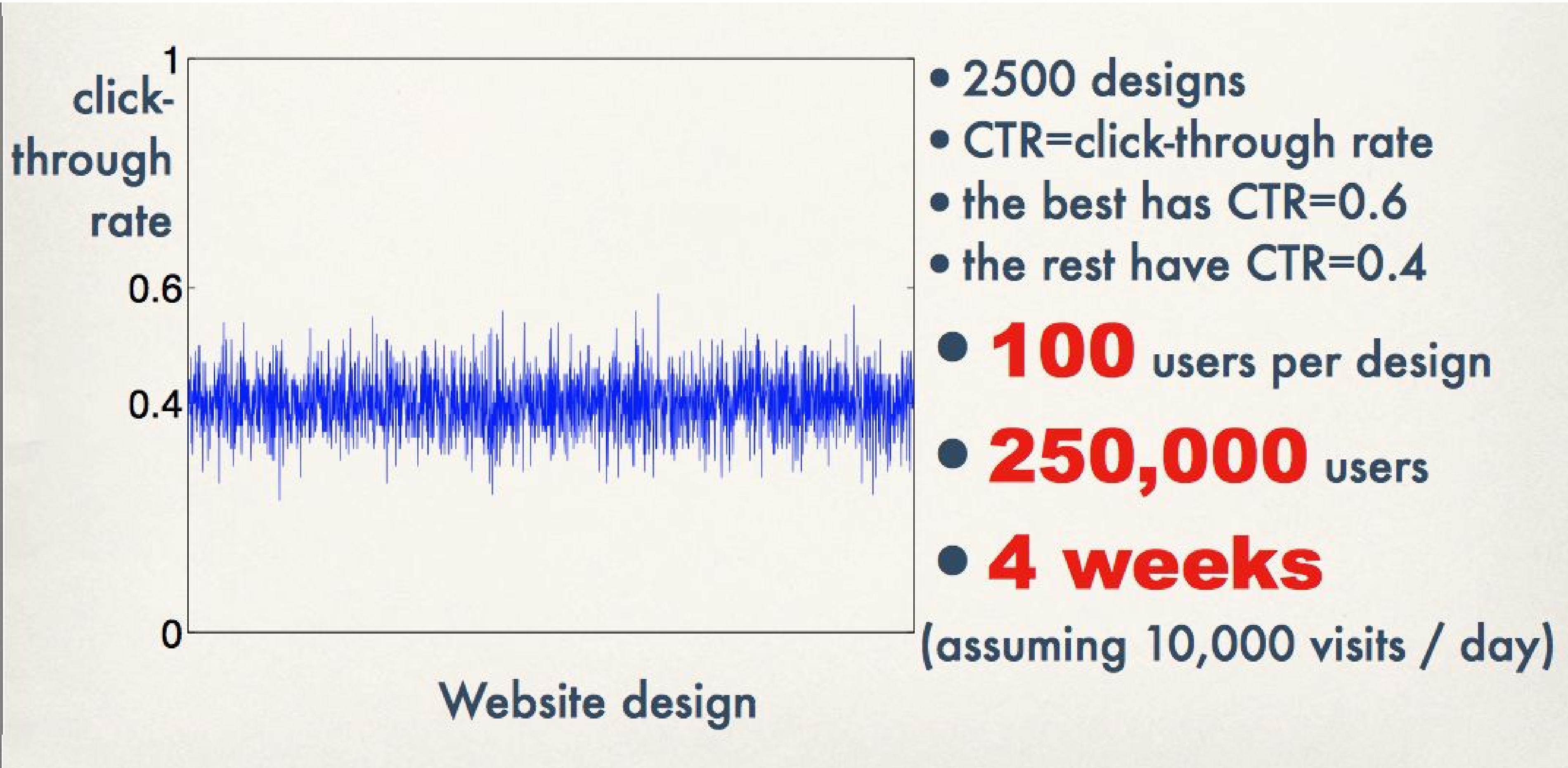
Standard A/B Testing or Experimentation Doesn't Scale



- 2500 designs
- CTR=click-through rate
- the best has CTR=0.6
- the rest have CTR=0.4
- **10** users per design
- **25,000** users overall
- **2.5 days**

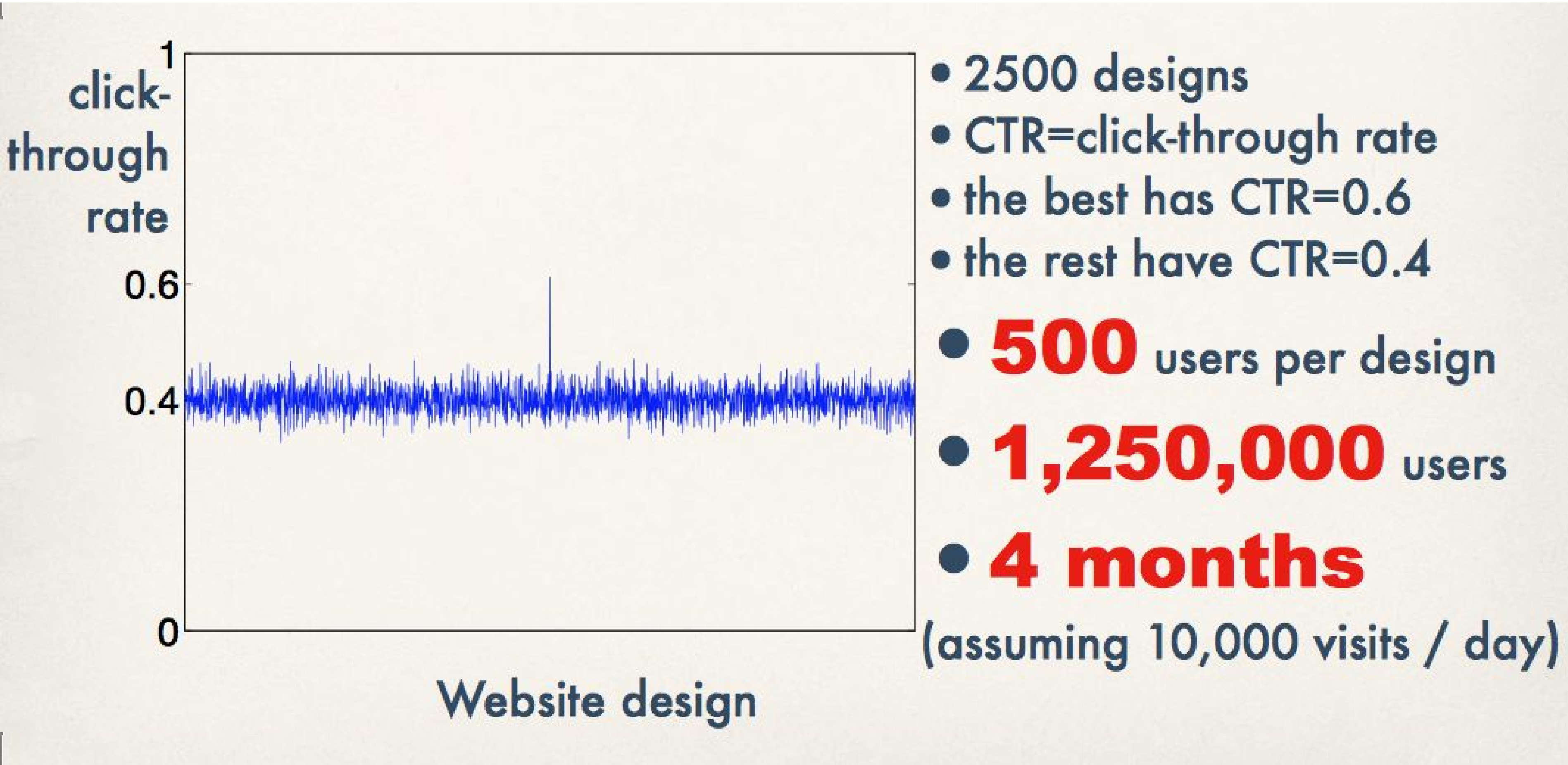
(assuming 10,000 visits / day)

Standard A/B Testing or Experimentation Doesn't Scale

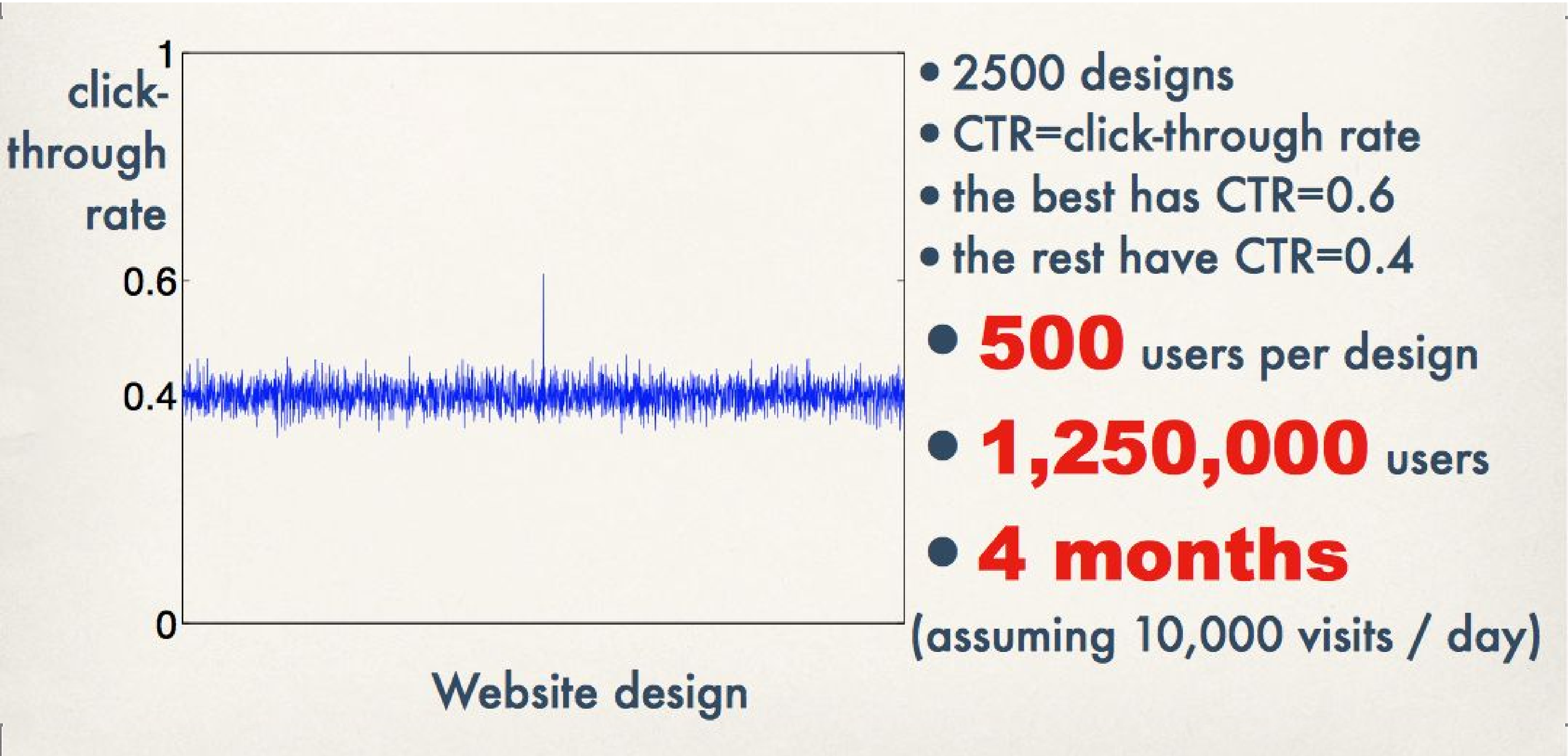


Material from Peter Frazier

Standard A/B Testing or Experimentation Doesn't Scale



Don’t Want to Have Worse Revenue on 1.25 million Users



Why Use GPs?

- Used frequently in Bayesian optimization
- Last ~5 years Bayesian optimization has become very influential & useful
- Brief (relevant) digression
- Two big motivations for Bayesian optimization
  - ML algorithm parameter tuning
  - Large online experimental settings where care about performance (e.g. revenue) while testing

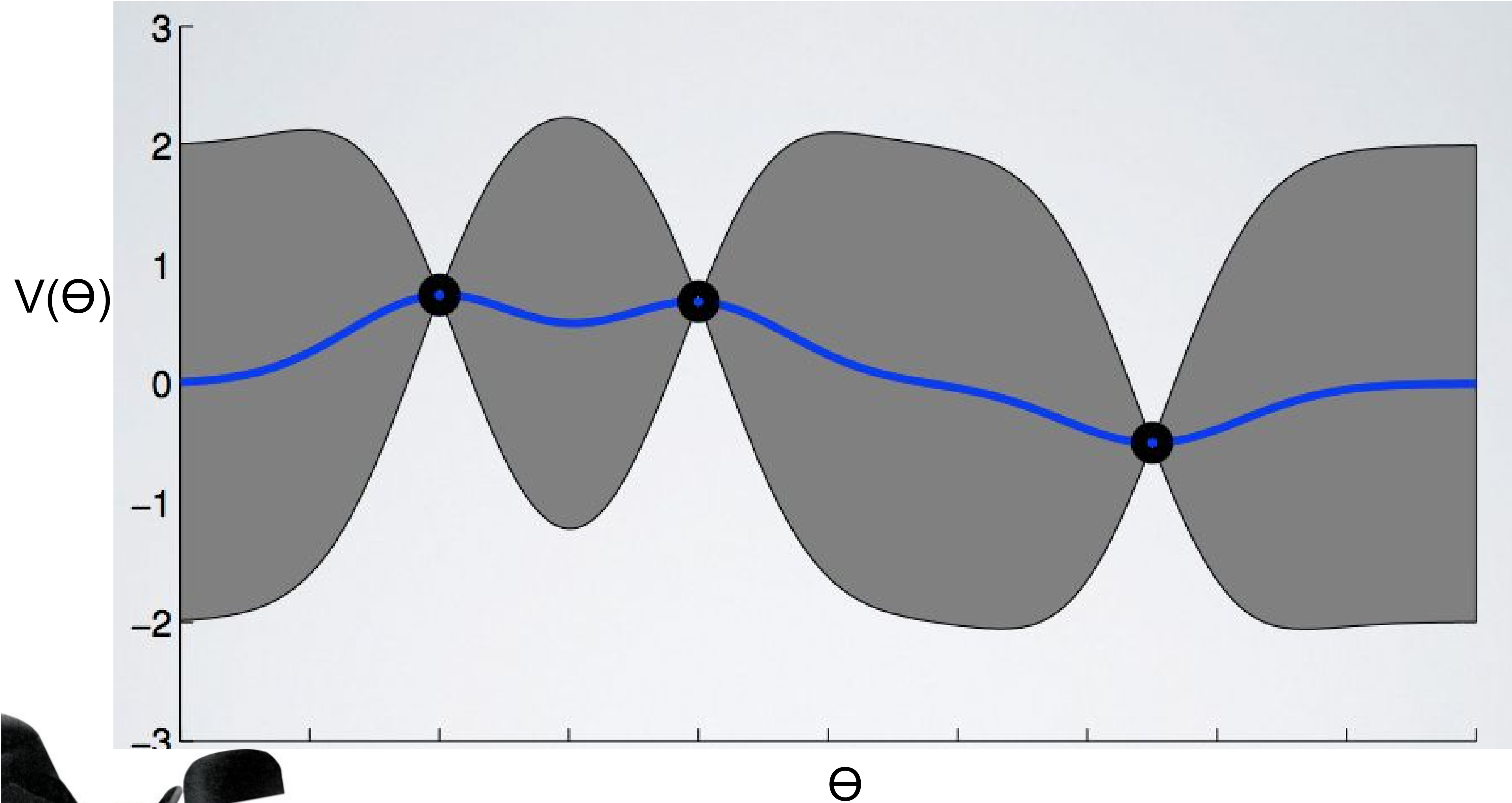
Bayesian Optimization

- Build a probabilistic model for the objective. Include hierarchical structure about units, etc.
- Compute the posterior predictive distribution. Integrate out all the possible true functions.
    - Gaussian process regression popular
- Optimize a cheap proxy function instead. The model is much cheaper than that true objective
- Two key ideas
    - Use model to guide how to search space. Model is an approximation, but when sampling a point in the real world is more costly than computation, very useful
    - Use proxy function to guide how to balance exploration and exploitation

Historical Background of Bayesian Optimization

- Closely related to statistical ideas of optimal design of experiments, dating back to Kirstine Smith in 1918.
- As response surface methods, date back to Box and Wilson in 1951
- As Bayesian optimization, studied first by Kushner in 1964 and then Mockus in 1978.
- Methodologically, it touches on several important machine learning areas: active learning, contextual bandits, Bayesian nonparametrics
- Started receiving serious attention in ML in 2007,
- Interest exploded when it was realized that Bayesian optimization provides an excellent tool for finding good ML hyperparameters.

Bayesian Optimization for Policy Search



Material from Ryan Adams

Bayesian Optimization for Policy Search

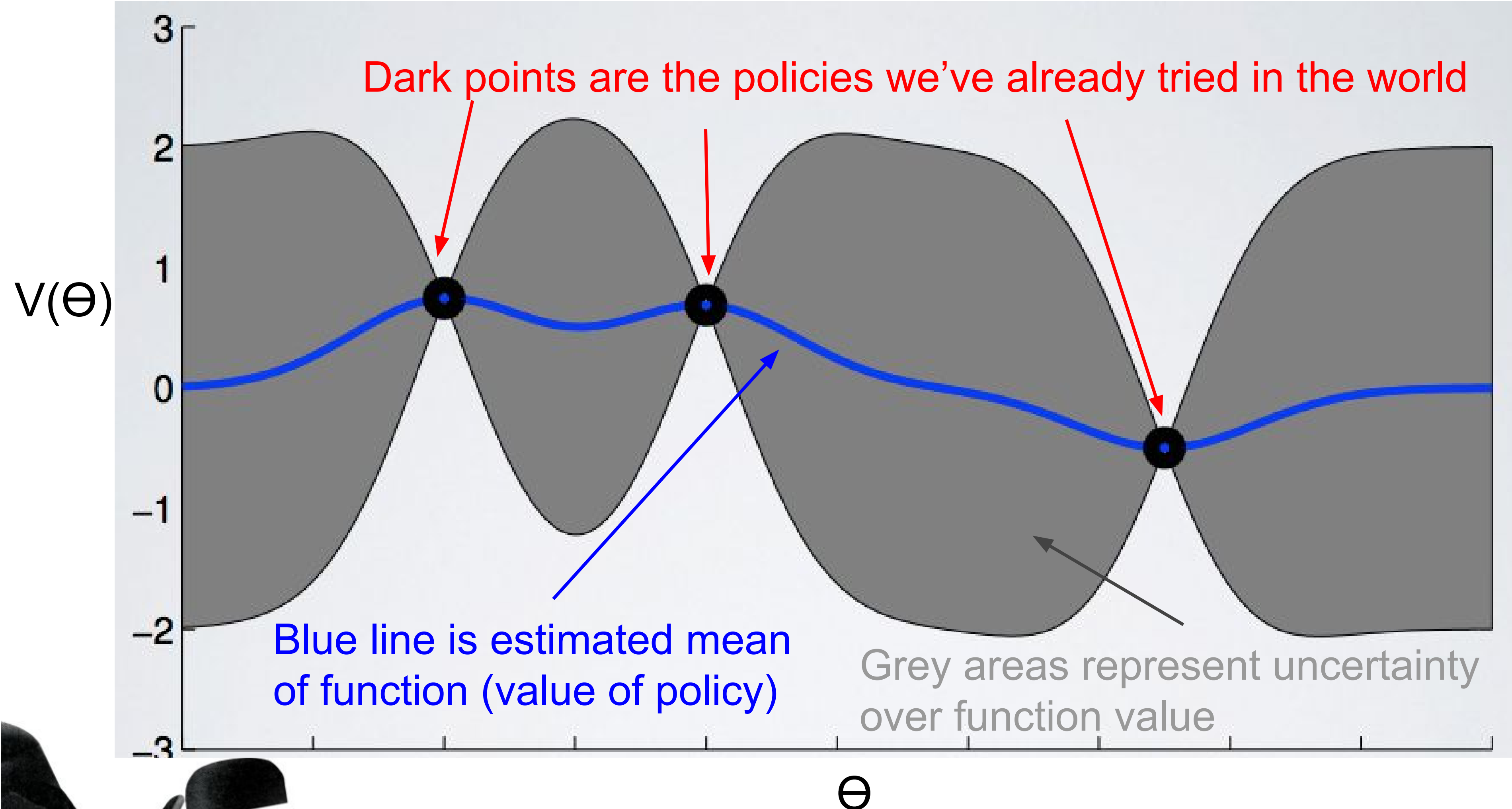


Figure modified from Ryan Adams

Exercise: Where Would You Sample Next (What Policy Would You Evaluate) & Why? What Algorithm Would You Use for Sampling?

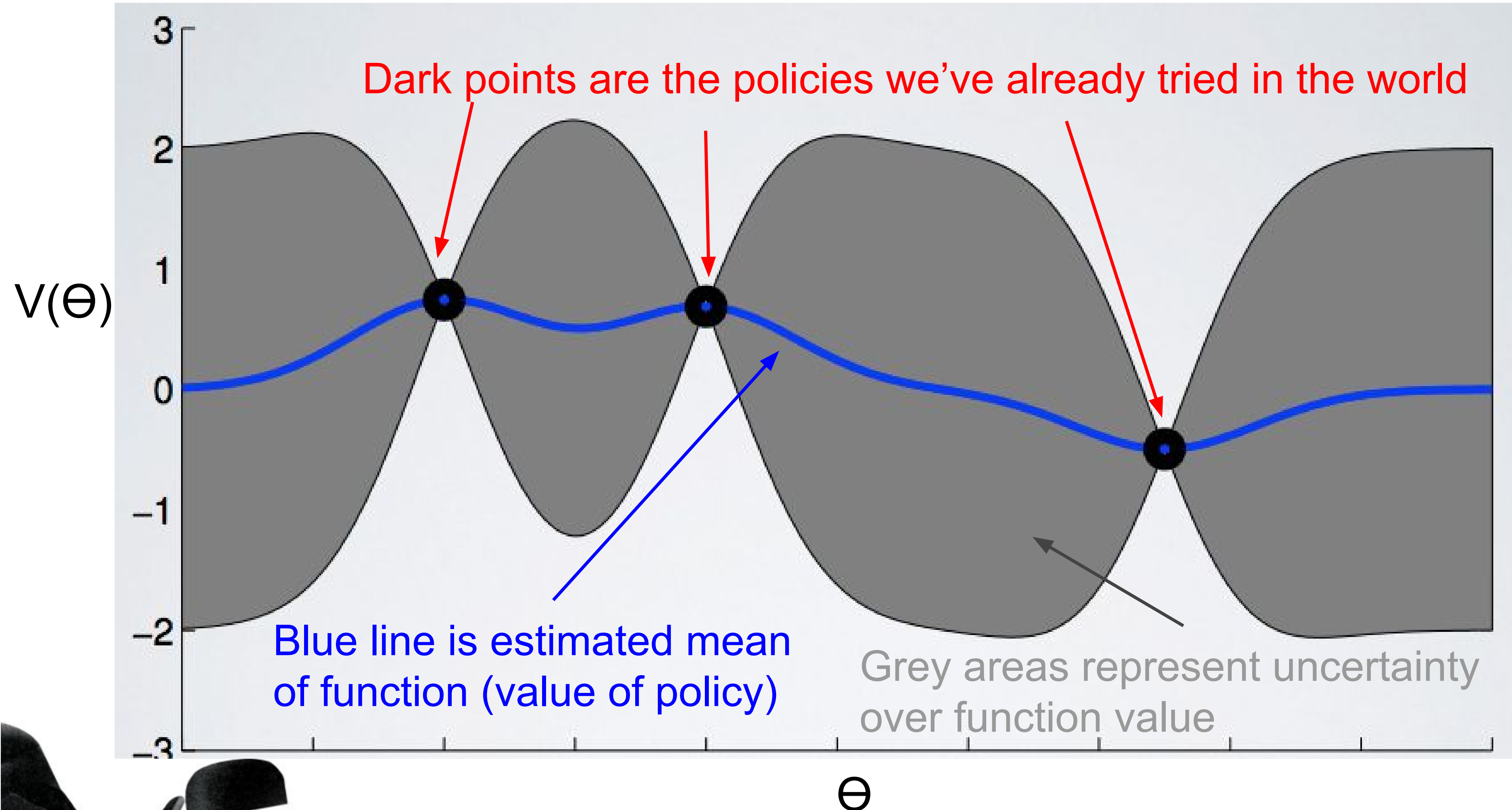


Figure modified from Ryan Adams

Probability of Improvement

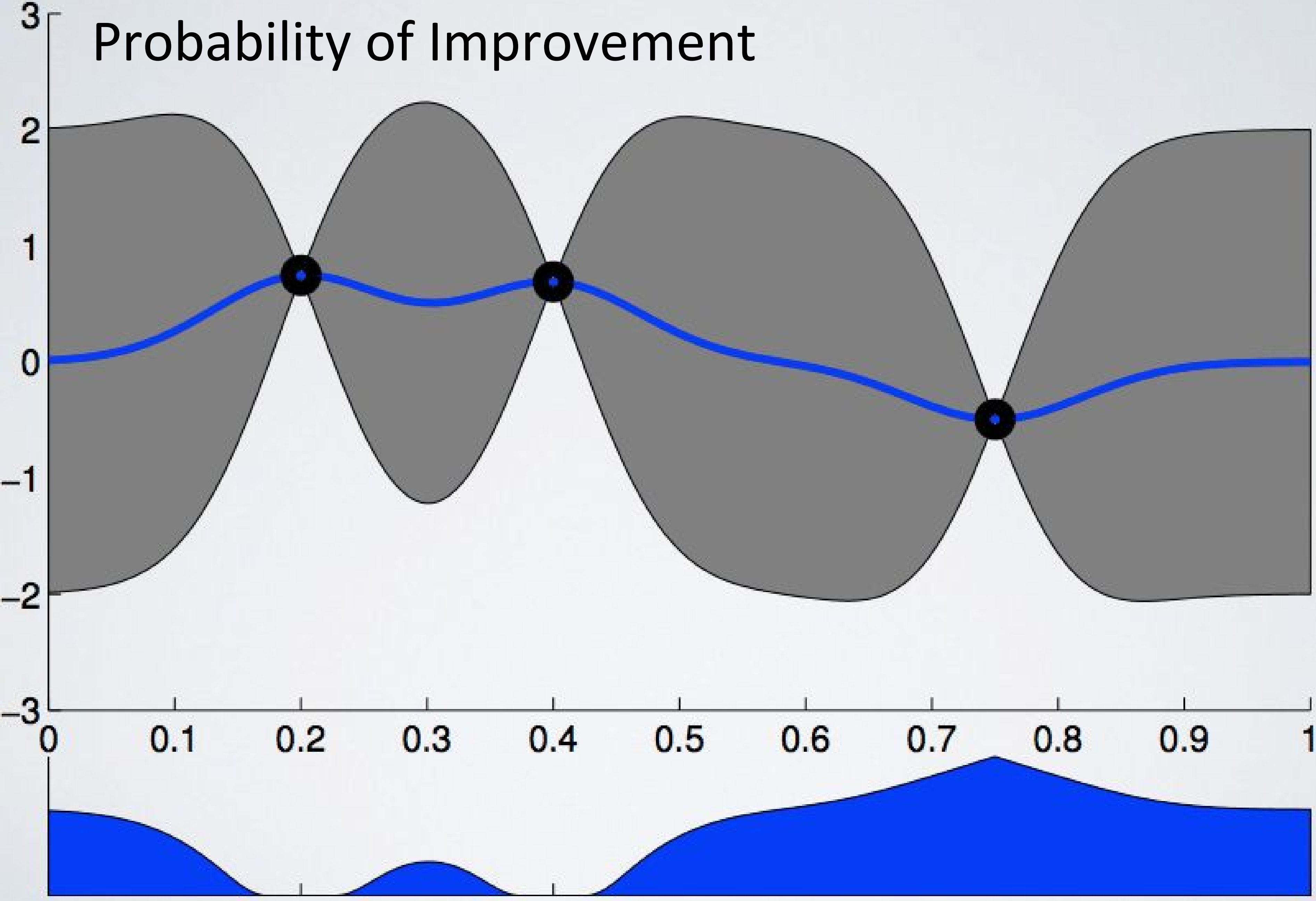


Figure modified from Ryan Adams

Expected Improvement

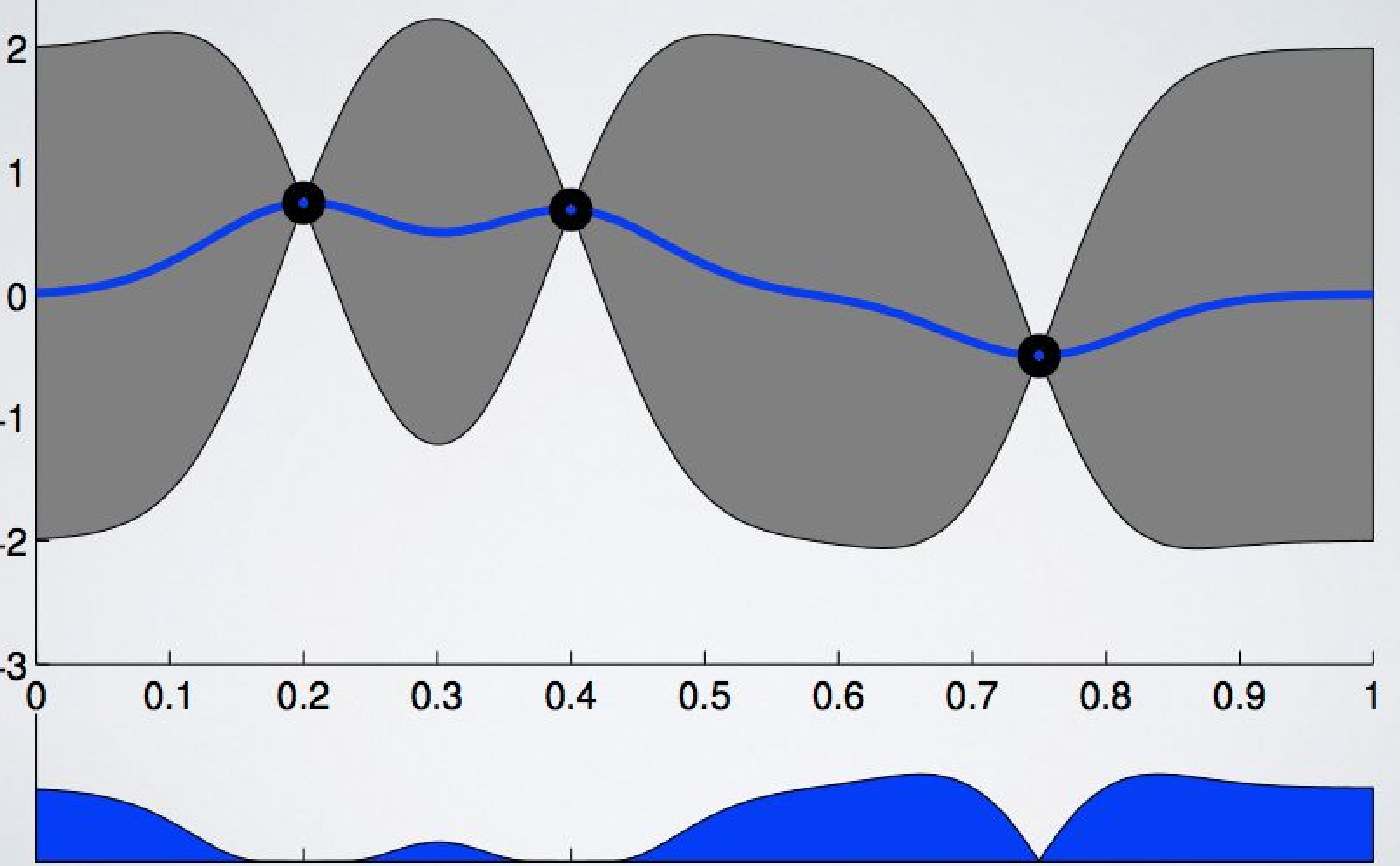


Figure modified from Ryan Adams

Upper/Lower Confidence Bound

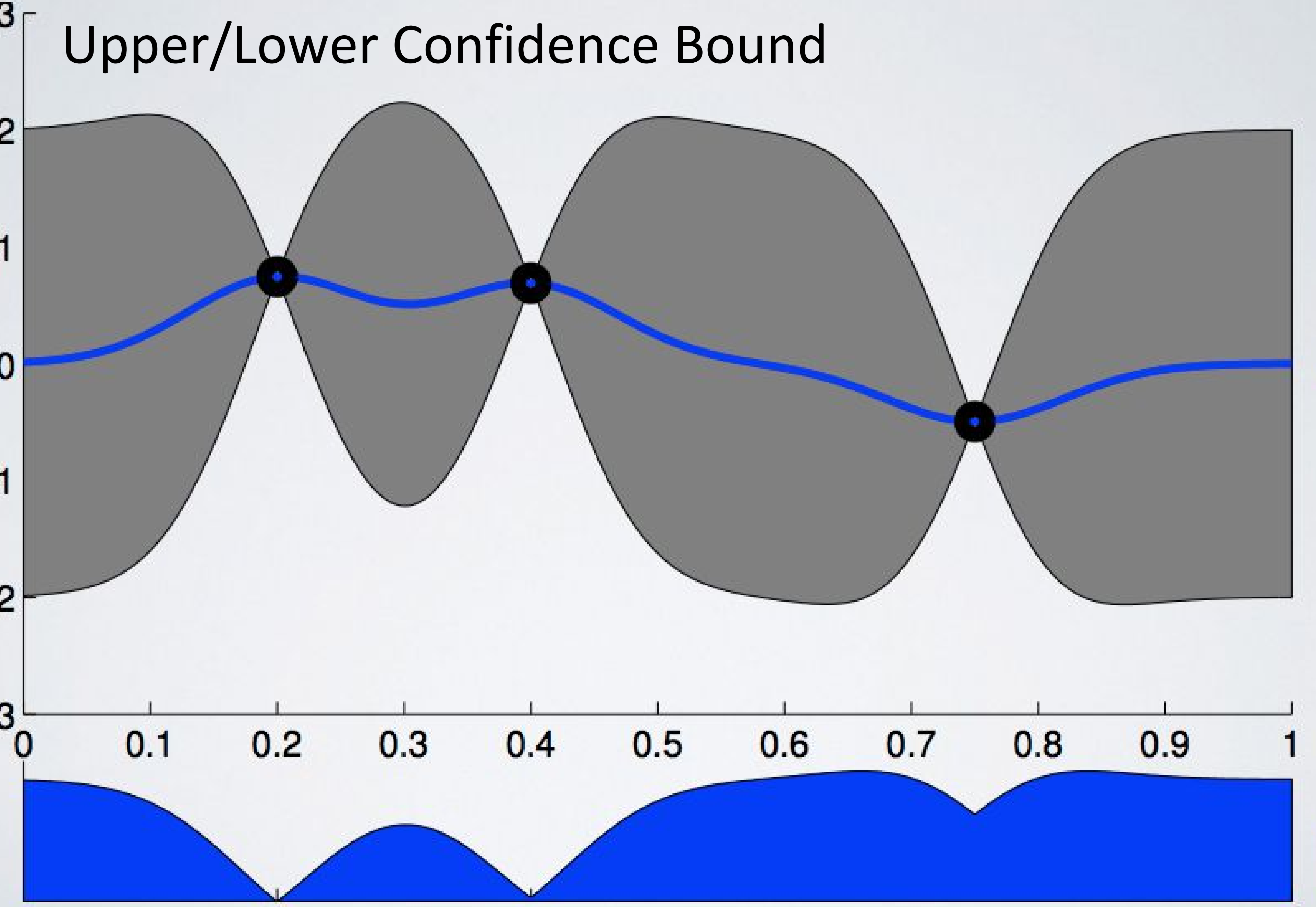
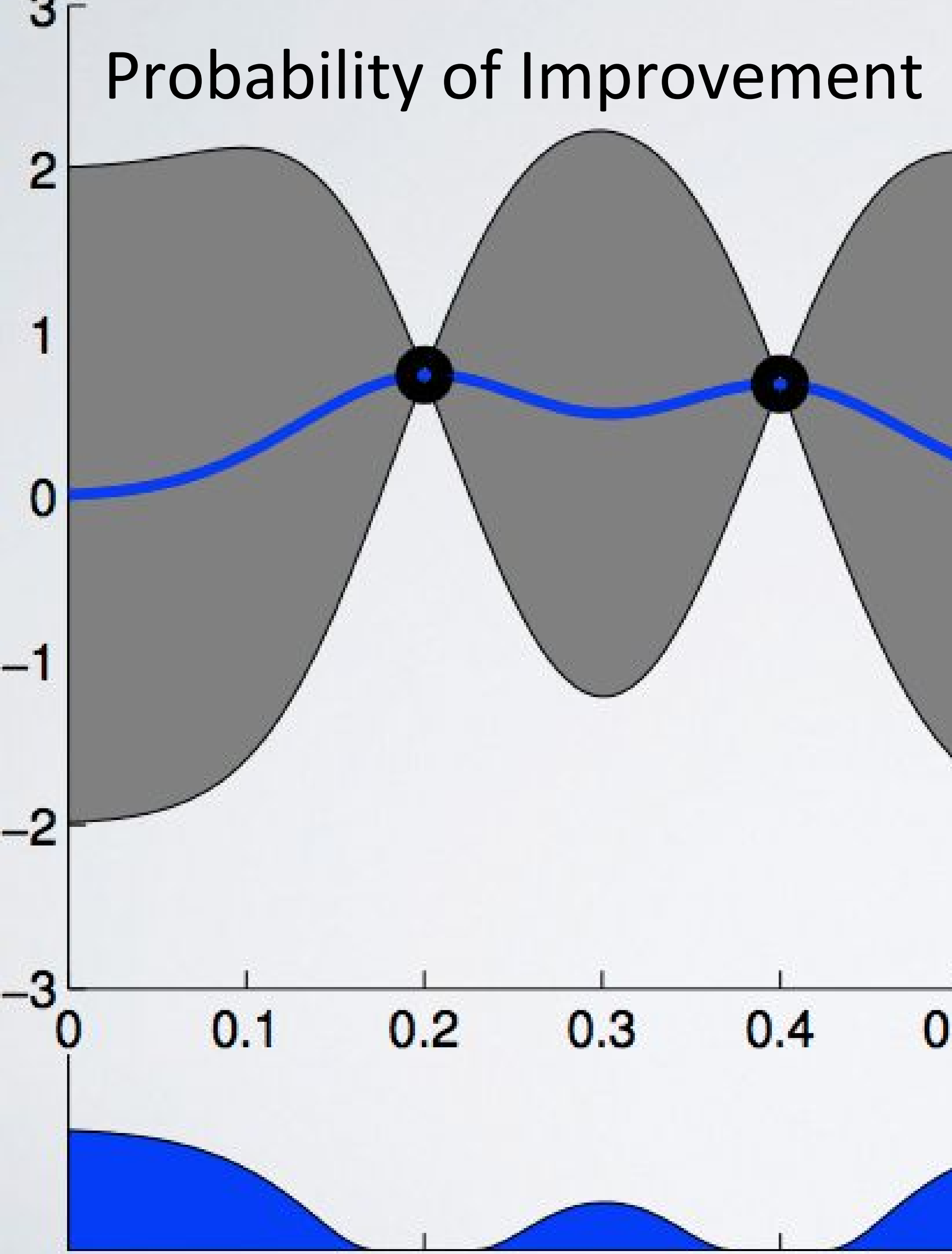


Figure modified from Ryan Adams

Probability of Improvement



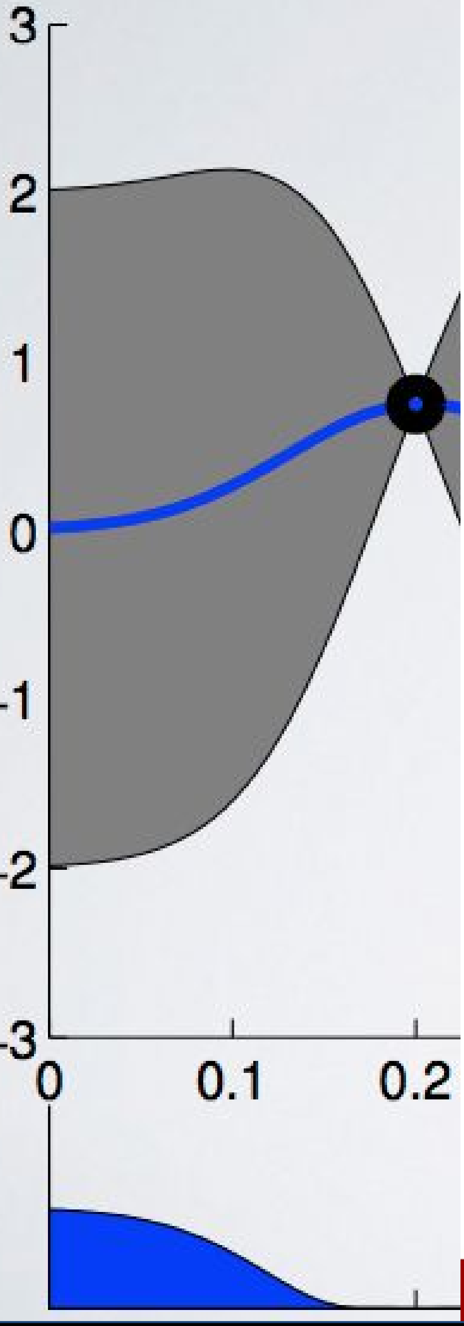
Utility function relative to f’, best point so far

$$u(x) = \begin{cases} 0 & f(x) > f' \\ 1 & f(x) \leq f'. \end{cases}$$

Acquisition function

$$\begin{aligned} a_{\mathrm{PI}}(x) &= \mathbb{E}\big[u(x) \mid x, \mathcal{D}\big] \\ &= \int_{-\infty}^{f'} \mathcal{N}\big(f; \mu(x), K(x, x)\big)\,\mathrm{d}f \\ &= \Phi\big(f'; \mu(x), K(x, x)\big). \end{aligned}$$

Figure modified from Ryan Adams, Equations from http://www.cse.wustl.edu/~garnett/cse515t/files/lecture_notes/12.pdf



Expected Improvement

Utility function relative to f', best point so far

$$u(x) = \max\big(0, f' - f(x)\big)$$

Acquisition function

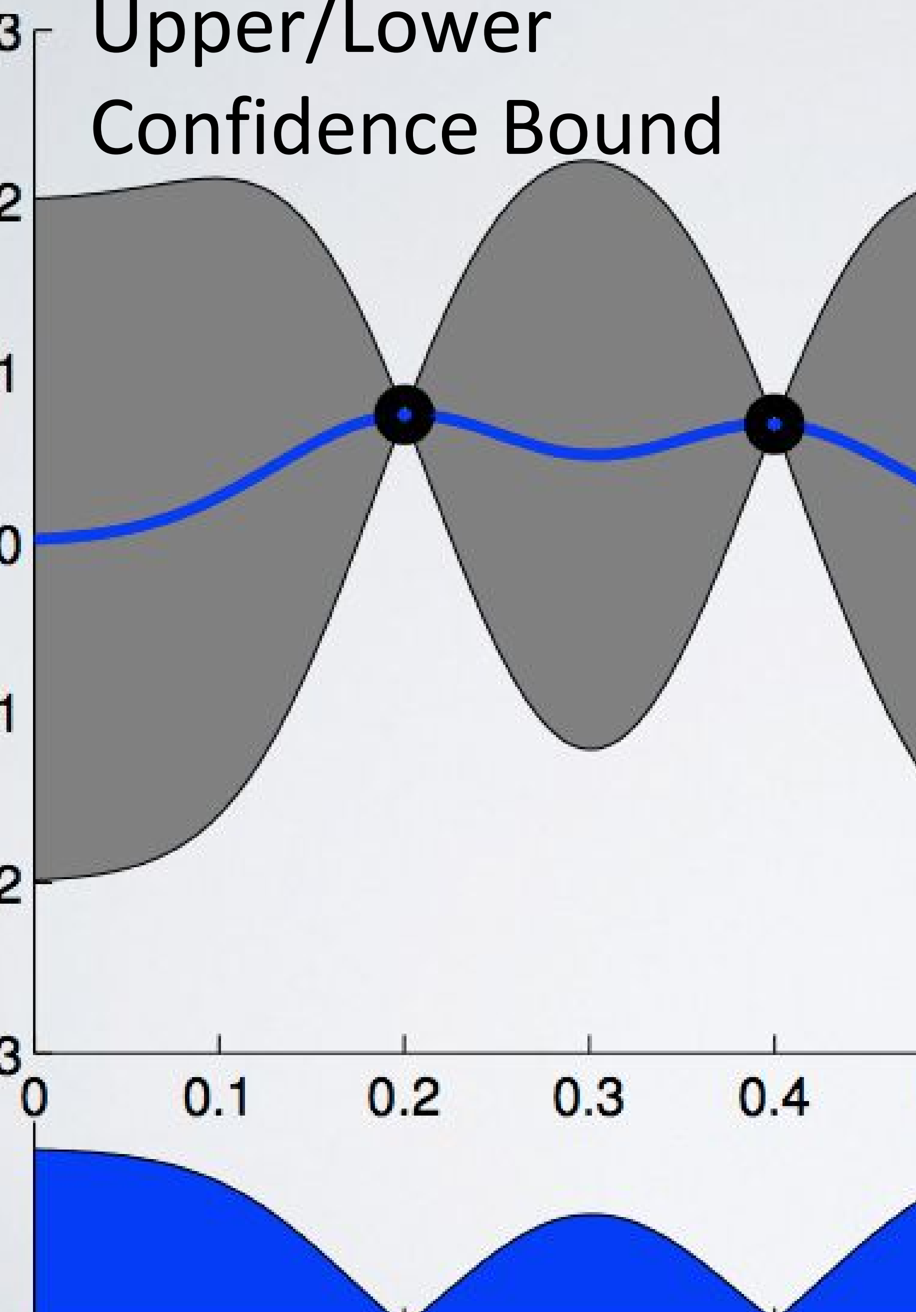
$$a_{\mathrm{EI}}(x) = \mathbb{E}\big[u(x) \mid x, \mathcal{D}\big]$$

$$= \int_{-\infty}^{f'} (f' - f)\, \mathcal{N}\big(f; \mu(x), K(x,x)\big)\, \mathrm{d}f$$

$$= \big(f' - \mu(x)\big)\Phi\big(f'; \mu(x), K(x,x)\big) + K(x,x)\mathcal{N}\big(f'; \mu(x), K(x,x)\big)$$

Figure modified from Ryan Adams, Equations from http://www.cse.wustl.edu/~garnett/cse515t/files/lecture_notes/12.pdf

Upper/Lower Confidence Bound



Acquisition function

$$a_{\text{UCB}}(x;\beta) = \mu(x) - \beta\sigma(x)$$

$$\sigma(x) = \sqrt{K(x,x)}$$

Figure modified from Ryan Adams, Equations from http://www.cse.wustl.edu/~garnett/cse515t/files/lecture_notes/12.pdf

Acquisition Function

- Probability of improvement
- Expected Improvement
- Upper confidence bound
- Other ideas?
- What are the limitations of these?

Policy Search as Black Box Bayesian Optimization

Bayesian Optimization with a Gaussian Process

$\pi = f(\theta_i)$

Execute policy π in environment for T steps, get total reward R

Create new training point $[f(\theta_i),R]$

Policy Search as Bayesian Optimization

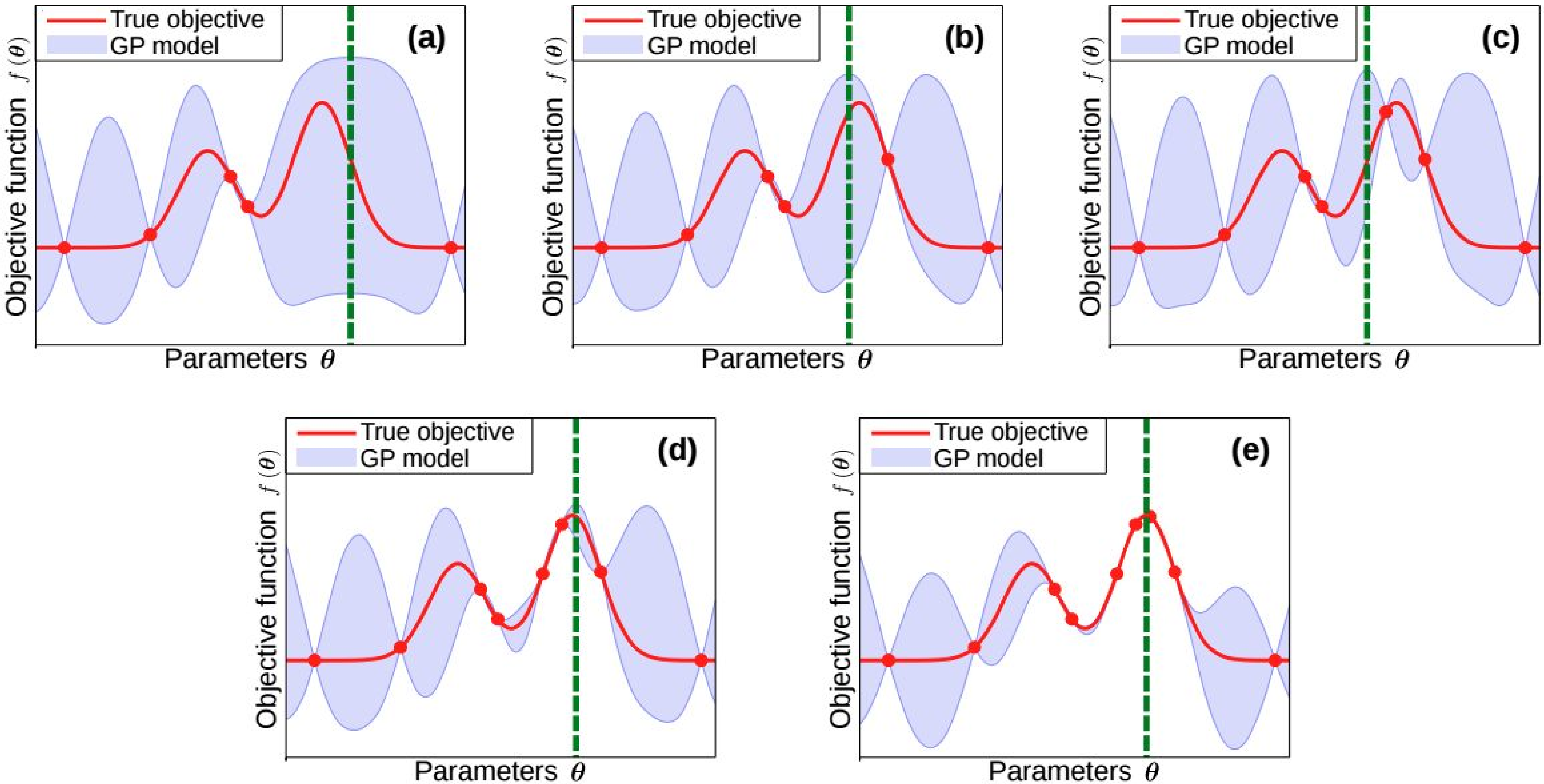


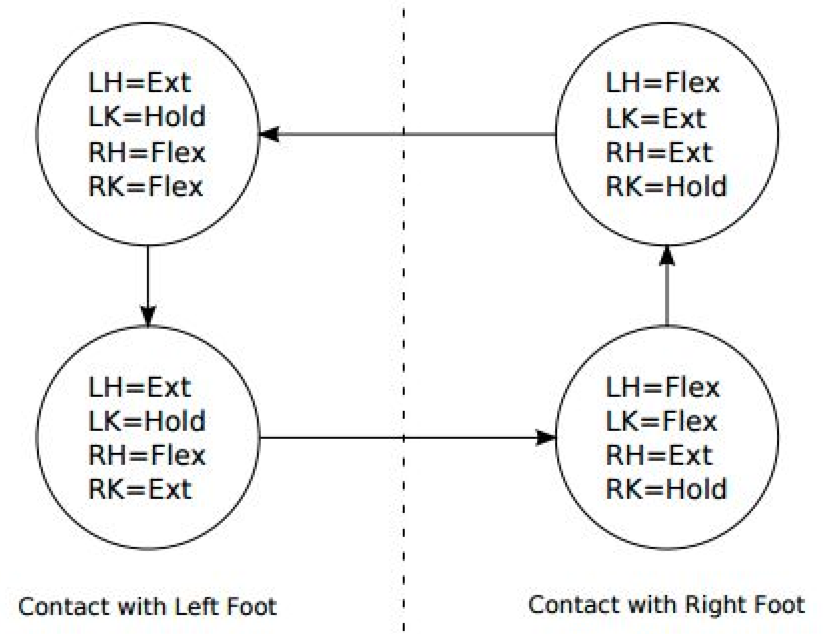
Figure modified from Calandra, Seyfarth, Peters & Deissenroth 2015

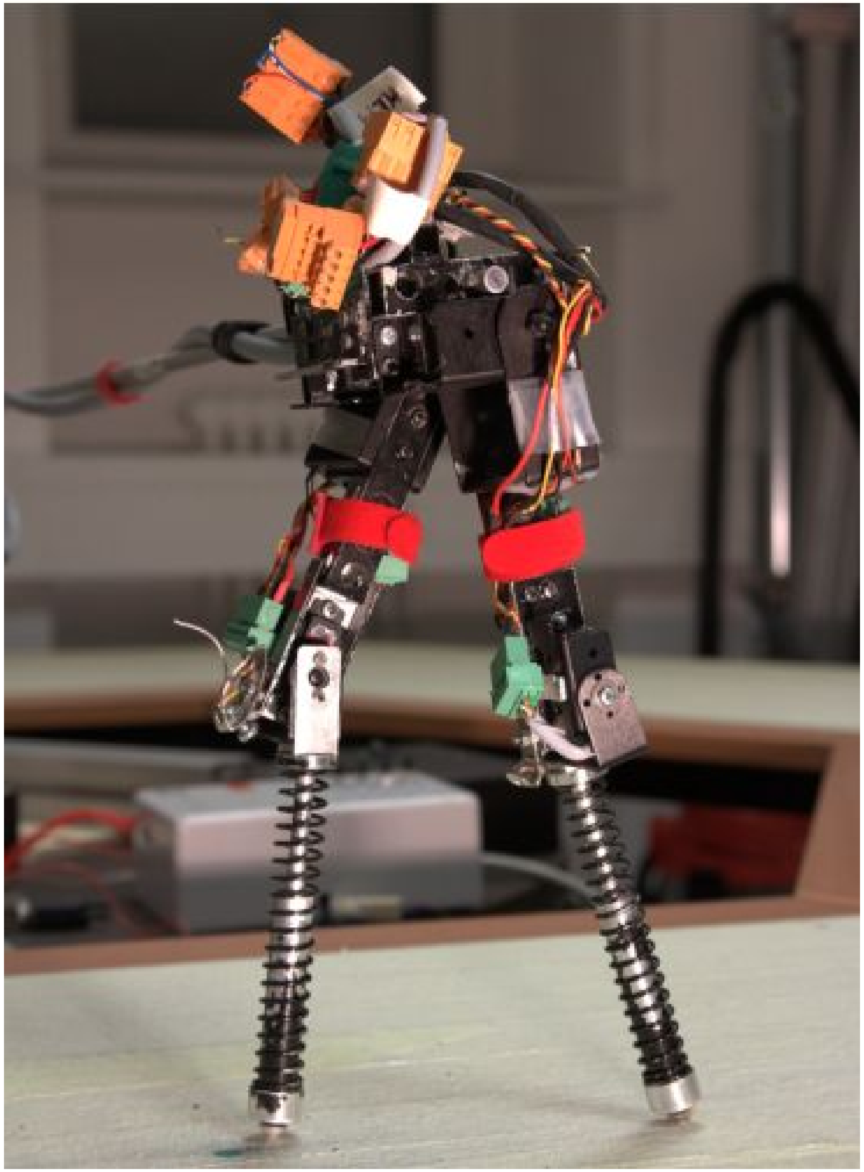
Carnegie Mellon University

Gait Optimization

- GP Policy search
- Reduced samples needed to find a fast walk by about 3x
- Lizotte et al. IJCAI 2007

More Gait Optimization





Gait parameters: 4 threshold values of the FSM (two for each leg) & 4 control signals applied during extension and flexion (separately for knees and hips).

Figures modified from Calandra, Seyfarth, Peters & Deissenroth 2015 videos

Why is this Suboptimal?

Bayesian Optimization
with a Gaussian Process

$\pi = f(\theta_i)$

Execute policy π in
environment for T steps, get
total reward R

Create new
training point
$[f(\theta_i), R]$

Throwing Away All Information But Policy Parameter & Total Reward from Trajectory.
Also Ignores Structure of Relationship Between Policies

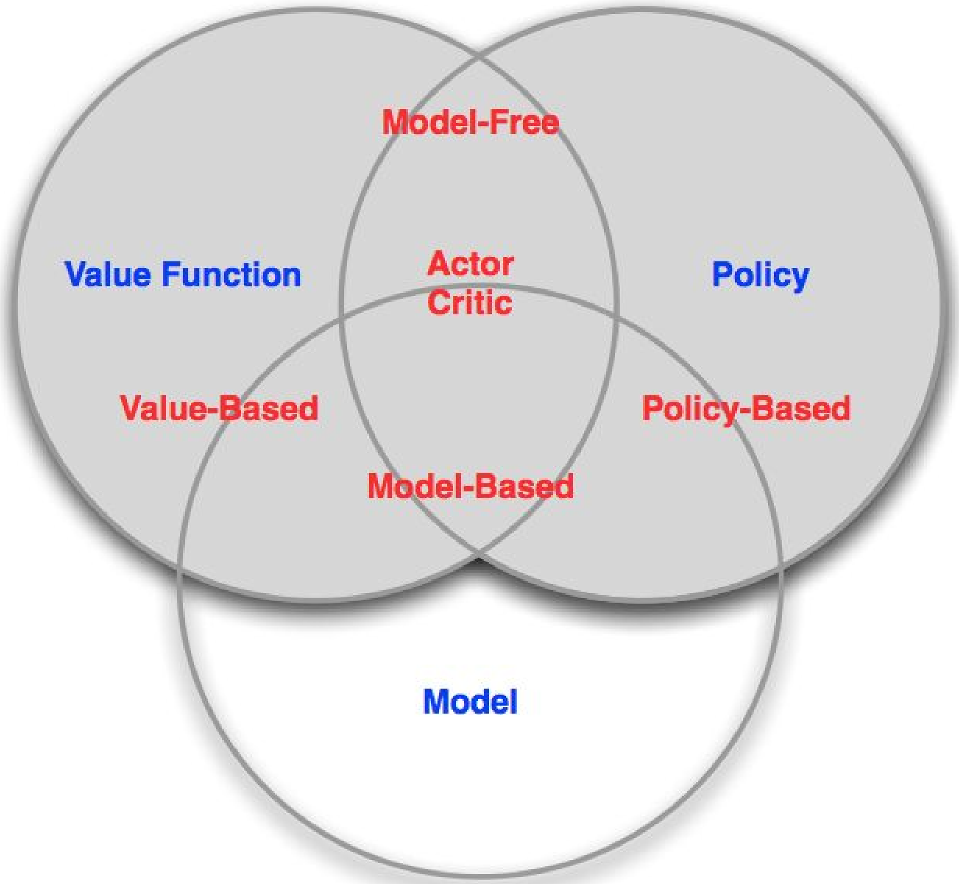
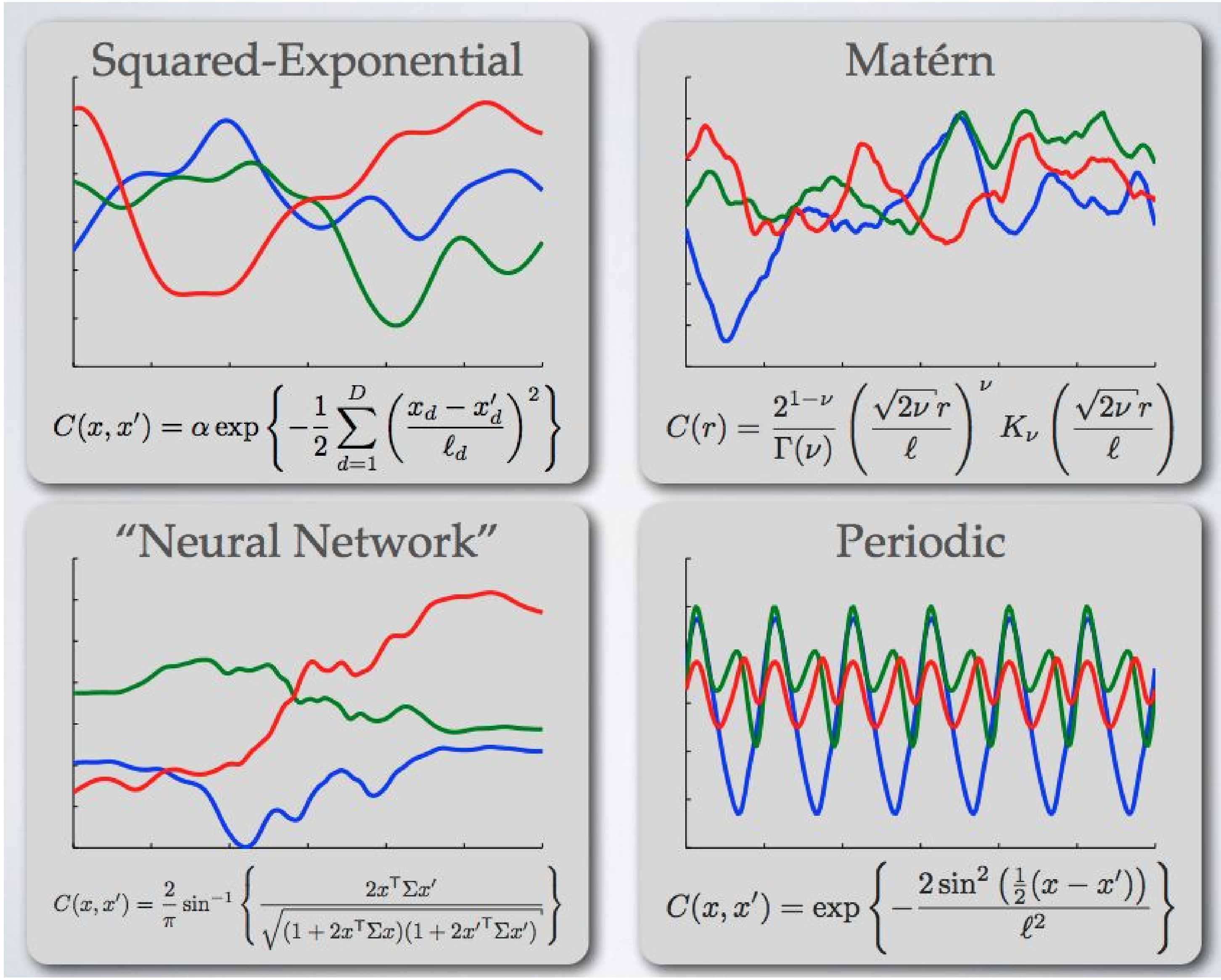


Figure from David Silver

Covariance function: Key choice for GP

When should 2 policies be considered "close"?



Figures from Ryan Adams

Behavior Based Kernel
(Wilson et al. JMLR 2014)

$$D(\theta_i, \theta_j) = \sqrt{KL(\underbrace{P(\xi|\theta_i)}_{\text{Probability of trajectory under policy }\Theta_i}||P(\xi|\theta_j))} + \sqrt{KL(P(\xi|\theta_j)||P(\xi|\theta_i))}$$

KL divergence

$$K(\theta_i, \theta_j) = \exp(-\alpha \cdot D(\theta_i, \theta_j))$$

Behavior Based Kernel
(Wilson et al. JMLR 2014)

$$D(\theta_i, \theta_j) = \sqrt{KL(\underbrace{P(\xi|\theta_i)}_{\text{Prob. trajectory under policy } \Theta_i}||P(\xi|\theta_j))} + \sqrt{KL(P(\xi|\theta_j)||P(\xi|\theta_i))}$$

KL divergence

$$\hat{D}(\theta_i, \theta_j) = \sum_{\xi \in \xi_i} \log\left(\frac{P(\xi|\theta_i)}{P(\xi|\theta_j)}\right) + \sum_{\xi \in \xi_j} \log\left(\frac{P(\xi|\theta_j)}{P(\xi|\theta_i)}\right)$$

Behavior Based Kernel (BBK)
(Wilson et al. JMLR 2014)

$$D(\theta_i, \theta_j) = \sqrt{KL(\underbrace{P(\xi|\theta_i)}_{\text{Prob. trajectory under policy } \Theta_i}||P(\xi|\theta_j))} + \sqrt{KL(P(\xi|\theta_j)||P(\xi|\theta_i))}$$

KL divergence

$$\hat{D}(\theta_i, \theta_j) = \sum_{\xi \in \xi_i} \log\left(\frac{P(\xi|\theta_i)}{P(\xi|\theta_j)}\right) + \sum_{\xi \in \xi_j} \log\left(\frac{P(\xi|\theta_j)}{P(\xi|\theta_i)}\right)$$

Do we need to know the dynamics model?

Throwing Away All Information But Policy Parameter & Total Reward from Trajectory.
Also Ignores Structure of Relationship Between Policies

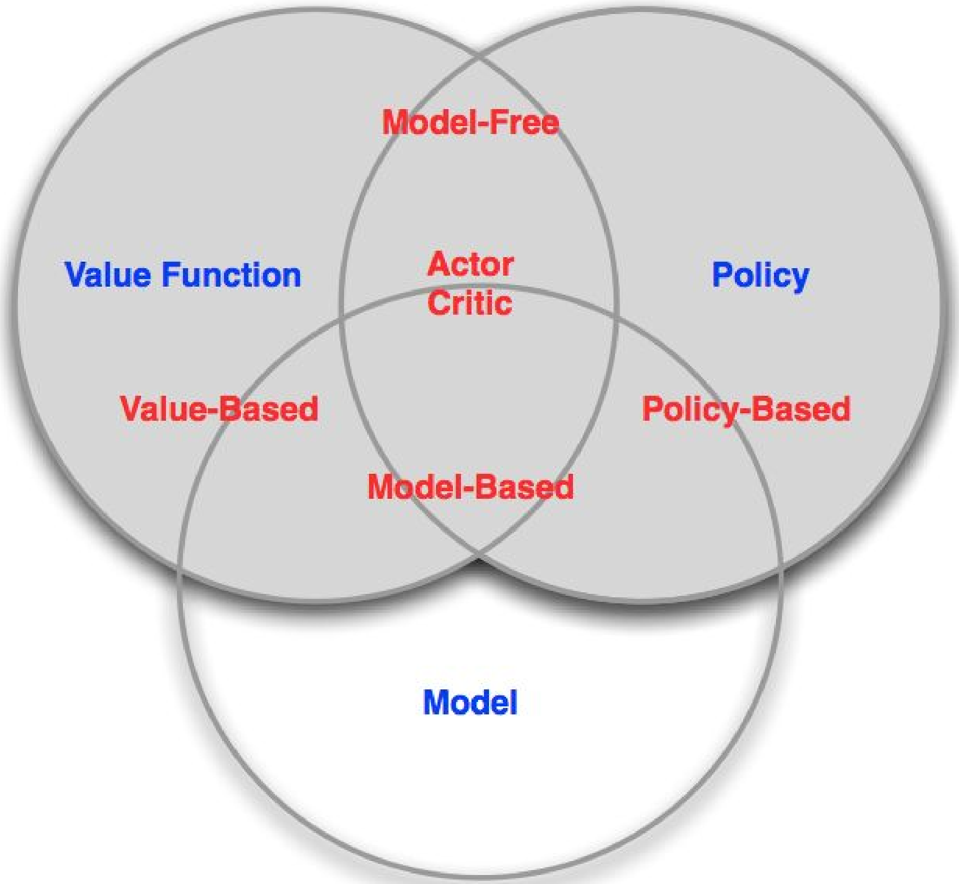


Figure from David Silver

Model-Based Bayesian Optimization Algorithm (MBOA) (Wilson et al. JMLR 2014)

Bayesian Optimization
with a Gaussian Process

$\pi = f(\theta_i)$

Use data to build a model.
Use to accelerate learning

Create new
training point
$[f(\theta_i), R]$

Execute policy π in
environment for T steps, get
total reward R

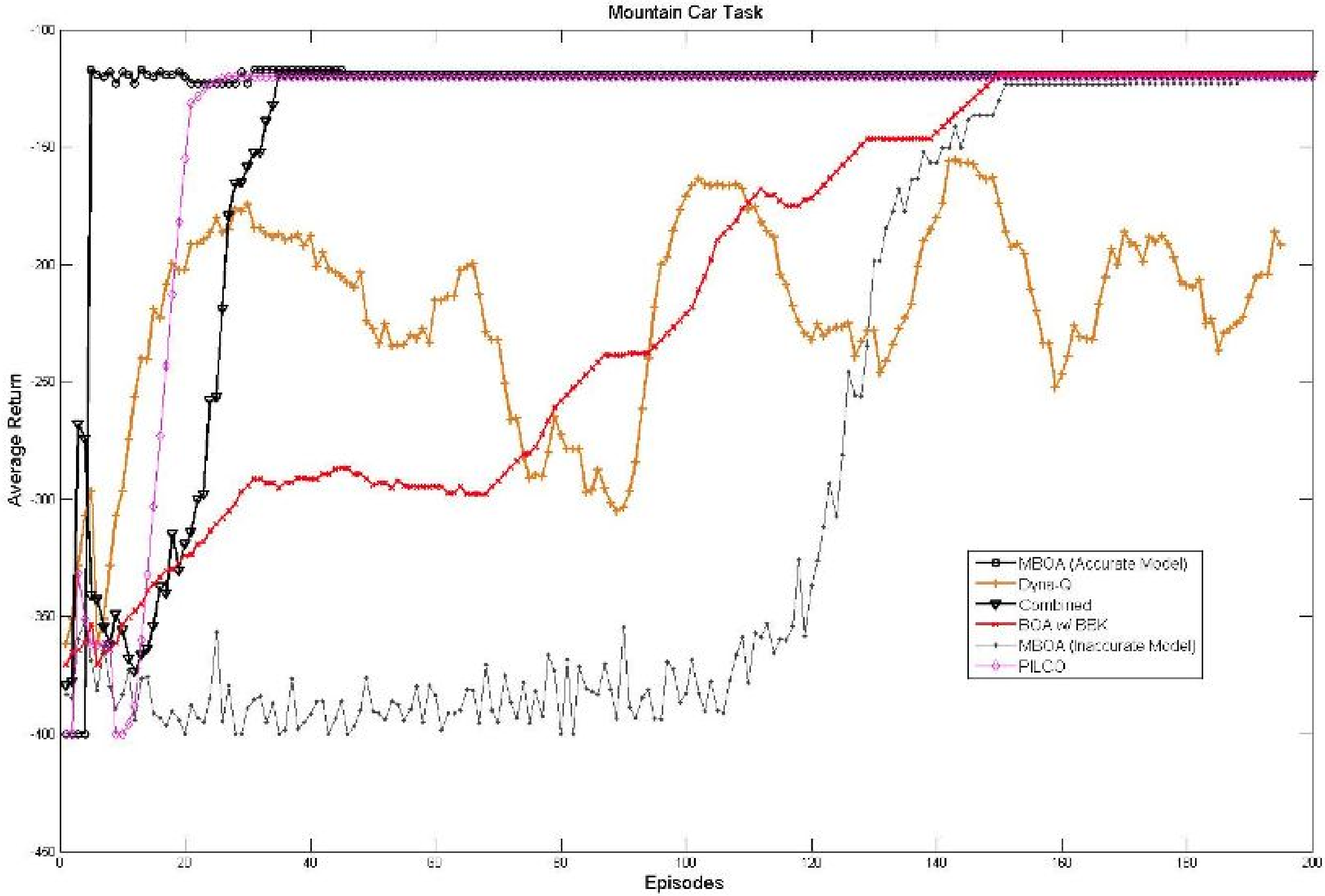


Figure from WIlson, Fern & Tadepalli
JMLR 2014

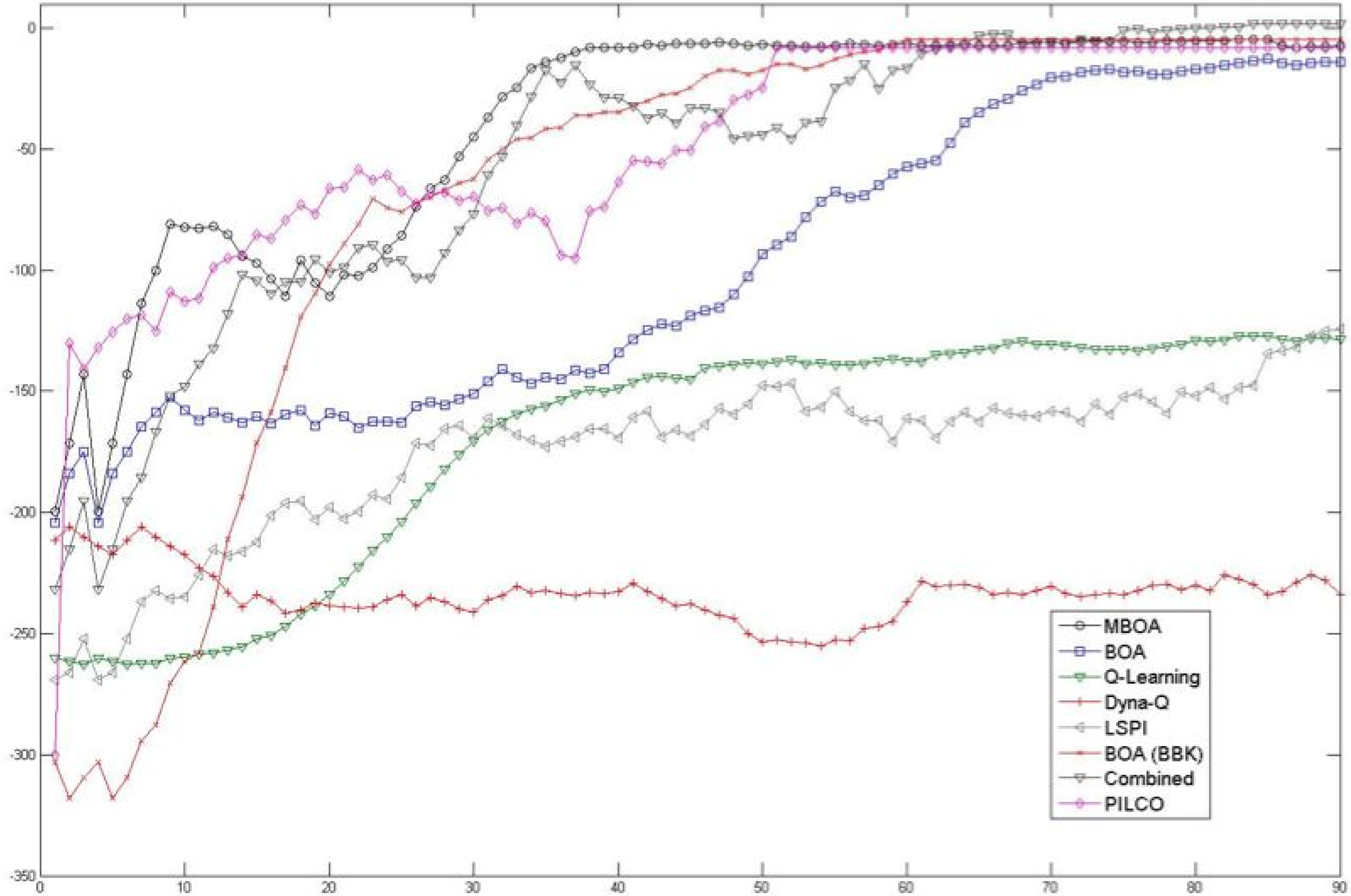


Figure from WIlson, Fern & Tadepalli
JMLR 2014

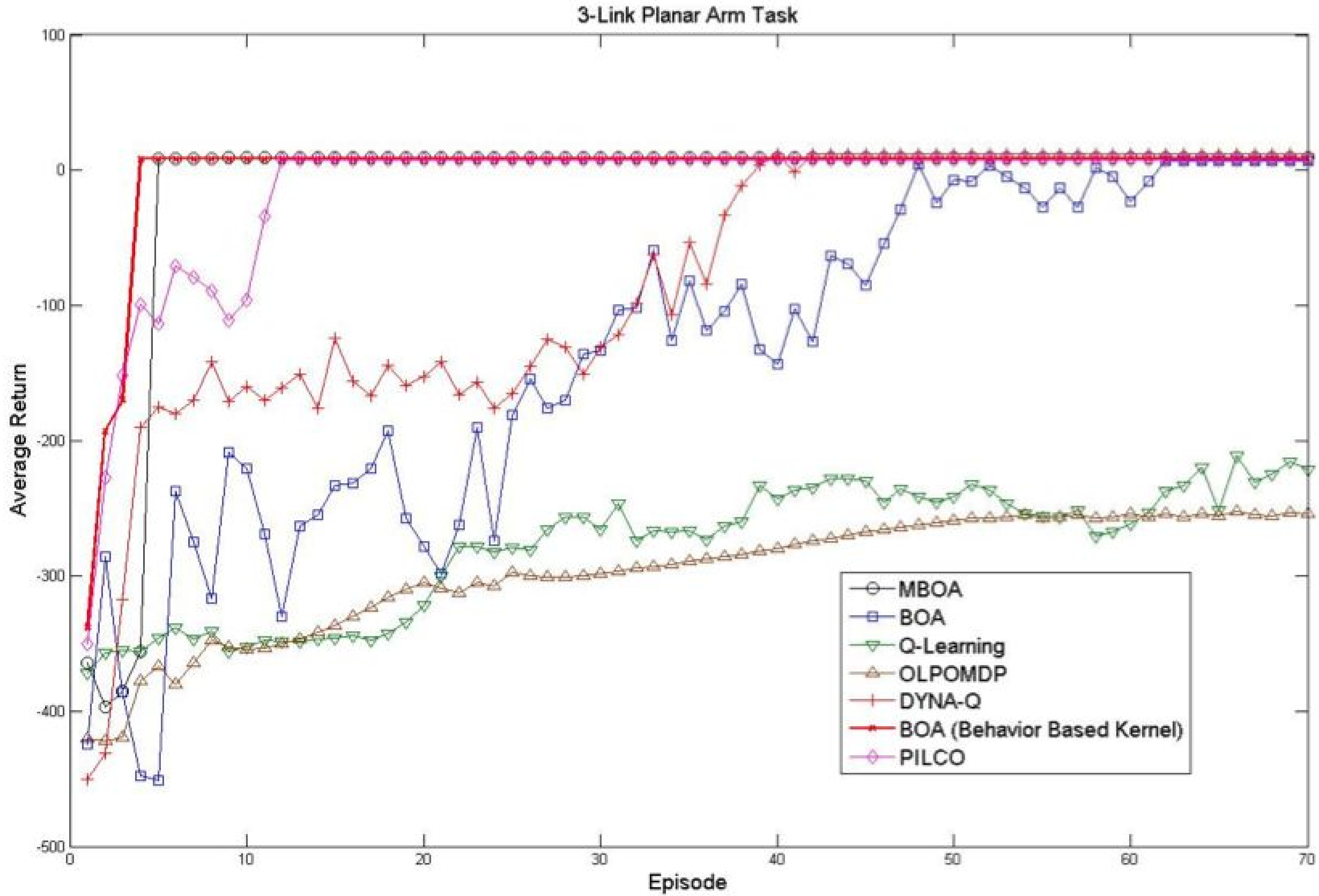


Figure from WIlson, Fern & Tadepalli JMLR 2014

Carnegie Mellon University

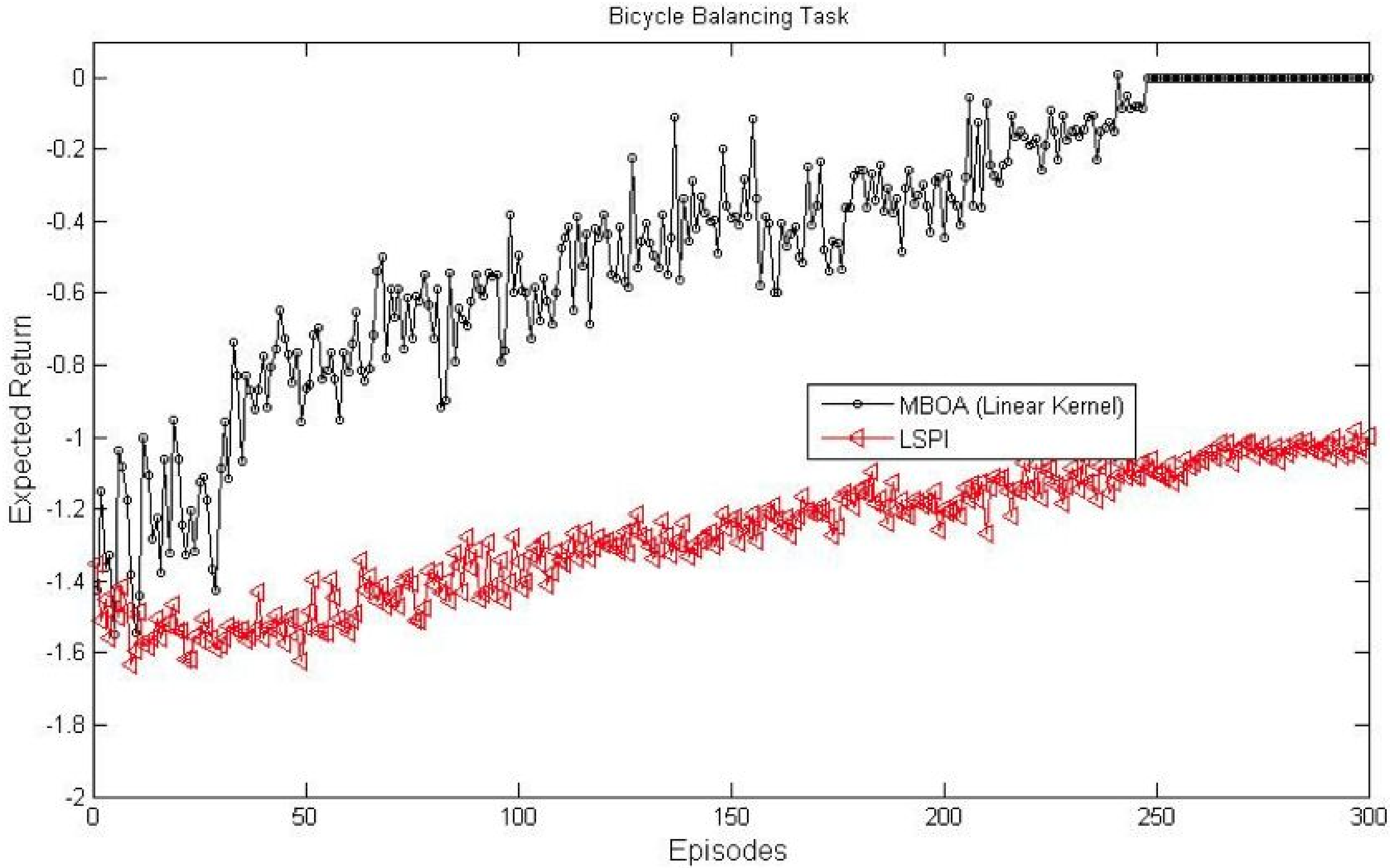
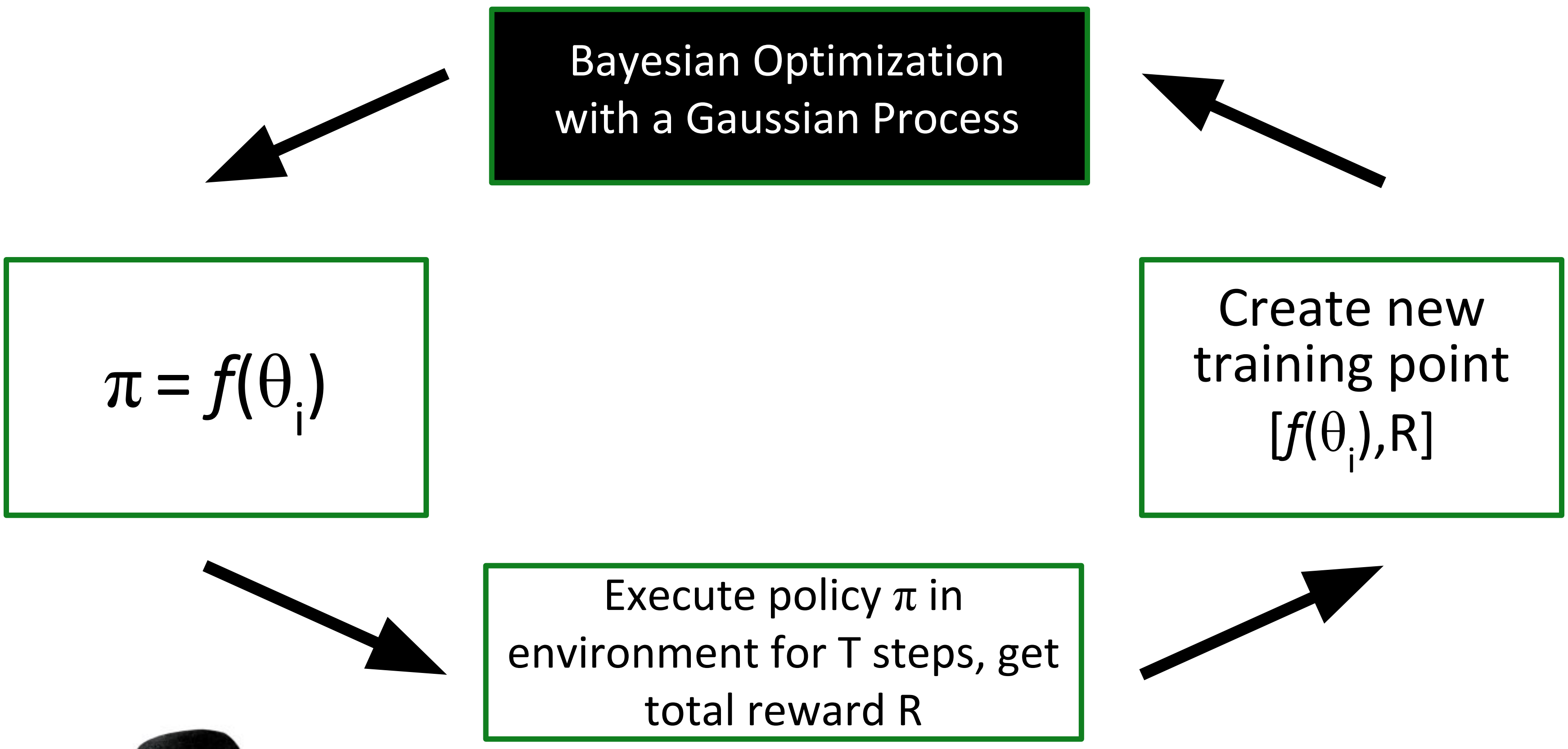


Figure from WIlson, Fern & Tadepalli
JMLR 2014

New Kernel vs Model Based Information?

- Using model information often greatly improves performance… if it’s a good model
- If it’s not good, learns to ignore
- New kernel to relate policies (BBK) much less of an impact

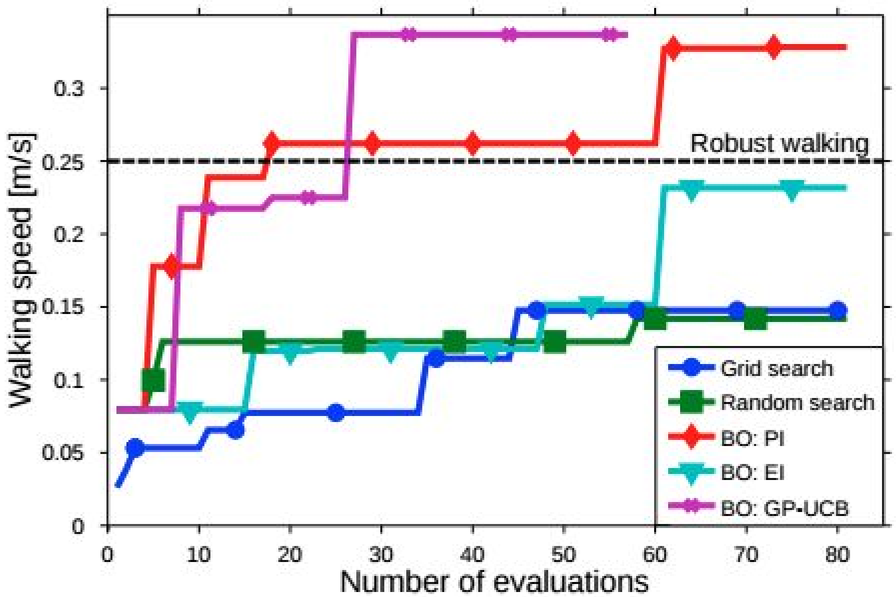
Other Ways to Go Beyond Black-Box Bayesian Optimization
Current work in my group (Rika, Christoph, Dexter Lee, Joe Runde)



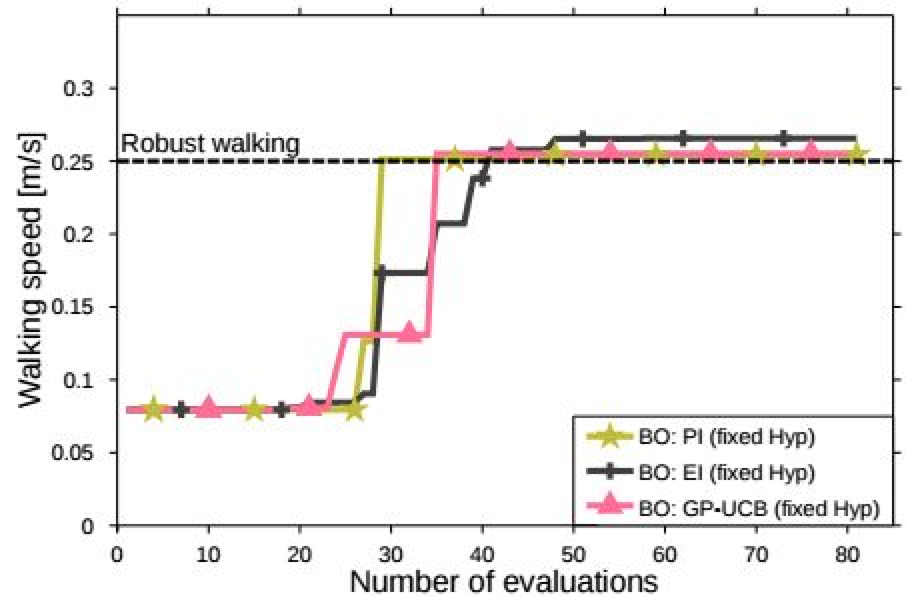
Subtleties of Bayesian Optimization

- Still have to choose policy class (this determines the input space)
- Have to choose kernel function
- Have to choose hyperparameters of kernel function (can optimize these)
- Have to choose acquisition function

Impact of Acquisition Function & Hyperparameters



(a) Different optimization methods



(b) BO with fixed hyperparameters

Manually fixed hyperparameters led to sub-optimal solutions for all the acquisition functions.

Summary: Bayesian Optimization for Efficient Policy Search

- Benefits
  - Direct policy search
  - Finds global optima
  - Uses sophisticated function class (GP) to model input policy param & output policy value
  - Use smart (but typically myopic) acquisition function to balance exploration/exploitation in searching policy space
  - Can be very sample efficient
- Not a silver bullet
  - Still have to decide on policy space
  - Choose kernel function (though squared expl often works well)
  - Should optimize hyperparameters

Stuff to Know: Bayesian Optimization for Efficient Policy Search

- Properties (global optima, no gradient information used)
- Define and know benefits/drawbacks of different acquisition functions
- Understand how to take a policy search problem and formulate as a black box Bayesian optimization problem
- Be able to list some things have to do in practice (optimize hyperparameters, etc)