

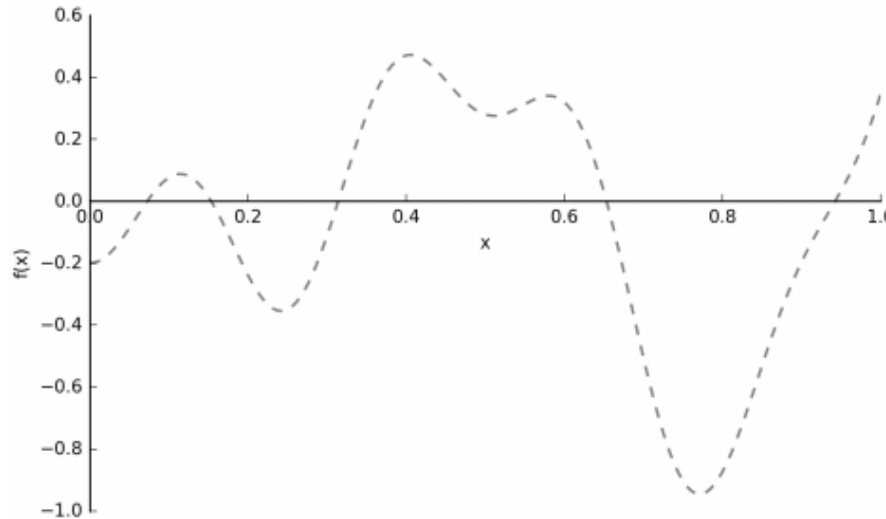
Bayesian Optimization

CS698X: Topics in Probabilistic Modeling and Inference

Piyush Rai

Bayesian Optimization: The Basic Formulation

- Consider finding the optima x_* (say minima) of a function $f(x)$



- Caveat: We don't know the form of the function; can't get its gradient, Hessian, etc
- Can only query the **function's values** at certain points (i.e., only “black-box” access)
 - The values may or may not be noisy (i.e., we may be given $f(x)$ or $f(x) + \epsilon$)



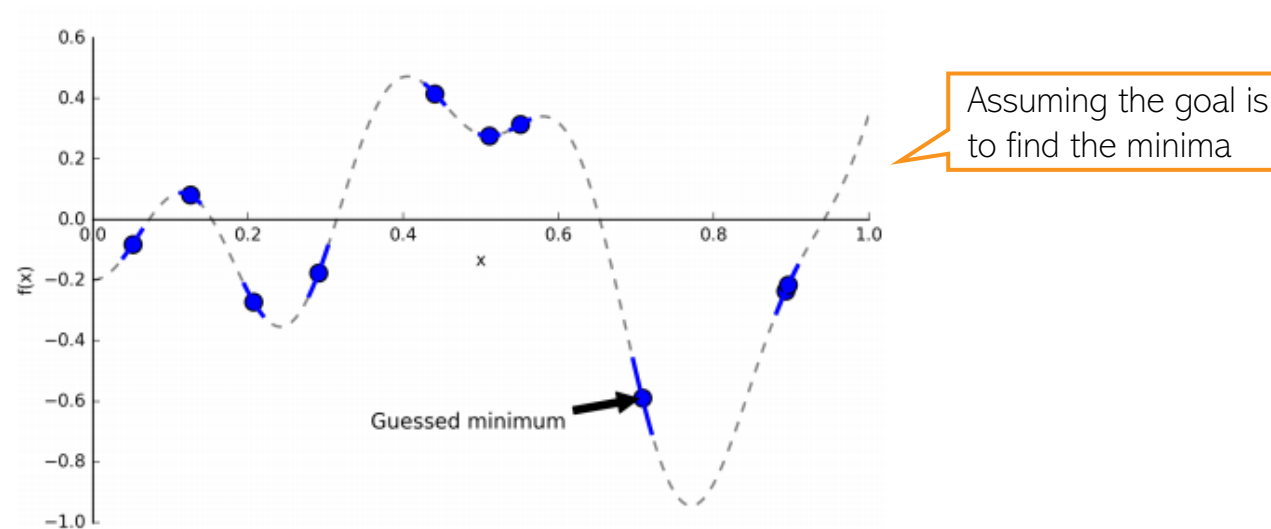
Bayesian Optimization: Some Applications

- Drug Design: Want to find the optimal chemical composition for a drug
 - Optimal composition will be the one that has the best efficacy
 - But we don't know the efficacy function
 - Can only know the efficacy via doing clinical trials
 - Each trial is expensive; can't do too many trials
- Hyperparameter Optimization: Want to find the optimal hyperparameters for a model
 - Optimal hyperparam values will be those that give the lowest test error
 - Don't know the true "test error" function
 - Need to train the model each time with different h.p. values and compute test error
 - Training every time will be expensive (e.g., for deep nets)
 - Note: Hyperparams here can even refer to the structure of a deep net (depth, width, etc)
- Many other applications: Website design via A/B testing, material design, optimizing physics based models (e.g., aircraft design), etc



Bayesian Optimization

- Can use BO to find maxima or minima
- Would like to locate the optima by querying the function's values (say, from an oracle)



- We would like to do so using as few queries as possible
- Reason: The function's evaluation may be time-consuming or costly



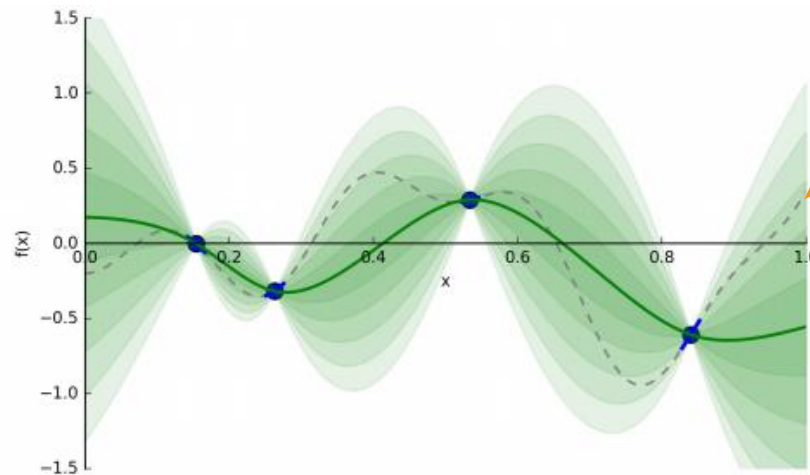
Bayesian Optimization

- Suppose we are allowed to make the queries sequentially
- This information will be available to us in form of query-function value pairs
- Queries so far can help us estimate the function

Note: Function values can be noisy too, e.g., $f(x_n) + \epsilon_n$

$\{(x_n, f(x_n))\}_{n=1}^N$

By solving a regression problem



Dotted curve: True function
Green curve: Current estimate ("surrogate") of the function
Shaded region: Uncertainty in the function's estimate

- BO uses **past queries** + **function's estimate+uncertainty** to decide **where to query next**
- Similar to Active Learning but the goal is to learn **f** as well as finds its optima

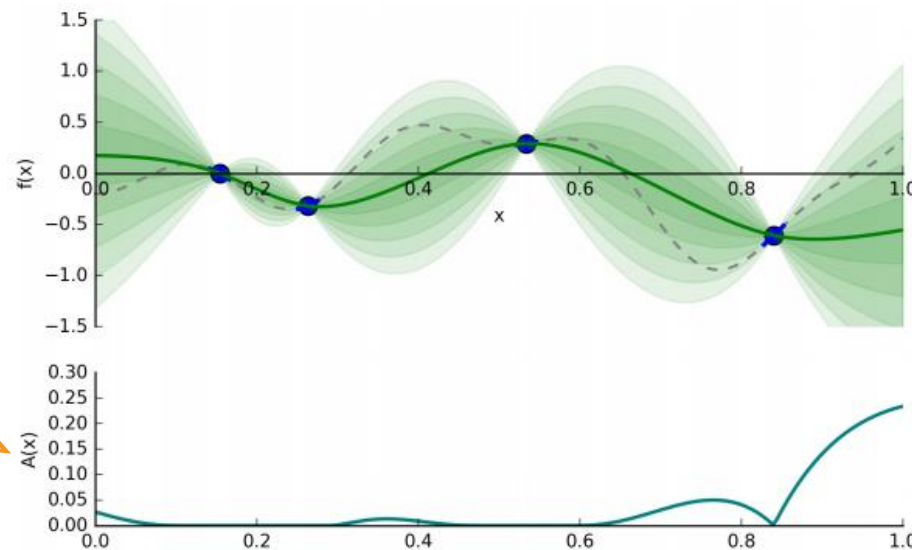


Bayesian Optimization

- BO requires two ingredients
 - A regression model to learn a surrogate of $f(x)$ given previous queries $\{(x_n, f(x_n))\}_{n=1}^N$
 - An acquisition function $A(x)$ to tell us where to query next

Assumption: $A(x)$ should be easier to optimize than $f(x)$

A typical example of what $A(x)$ might look like, assuming that the goal is to find the **maxima** of $f(x)$



Dotted curve: True function
Green curve: Current surrogate of the function
Shaded region: Uncertainty in the function's estimate

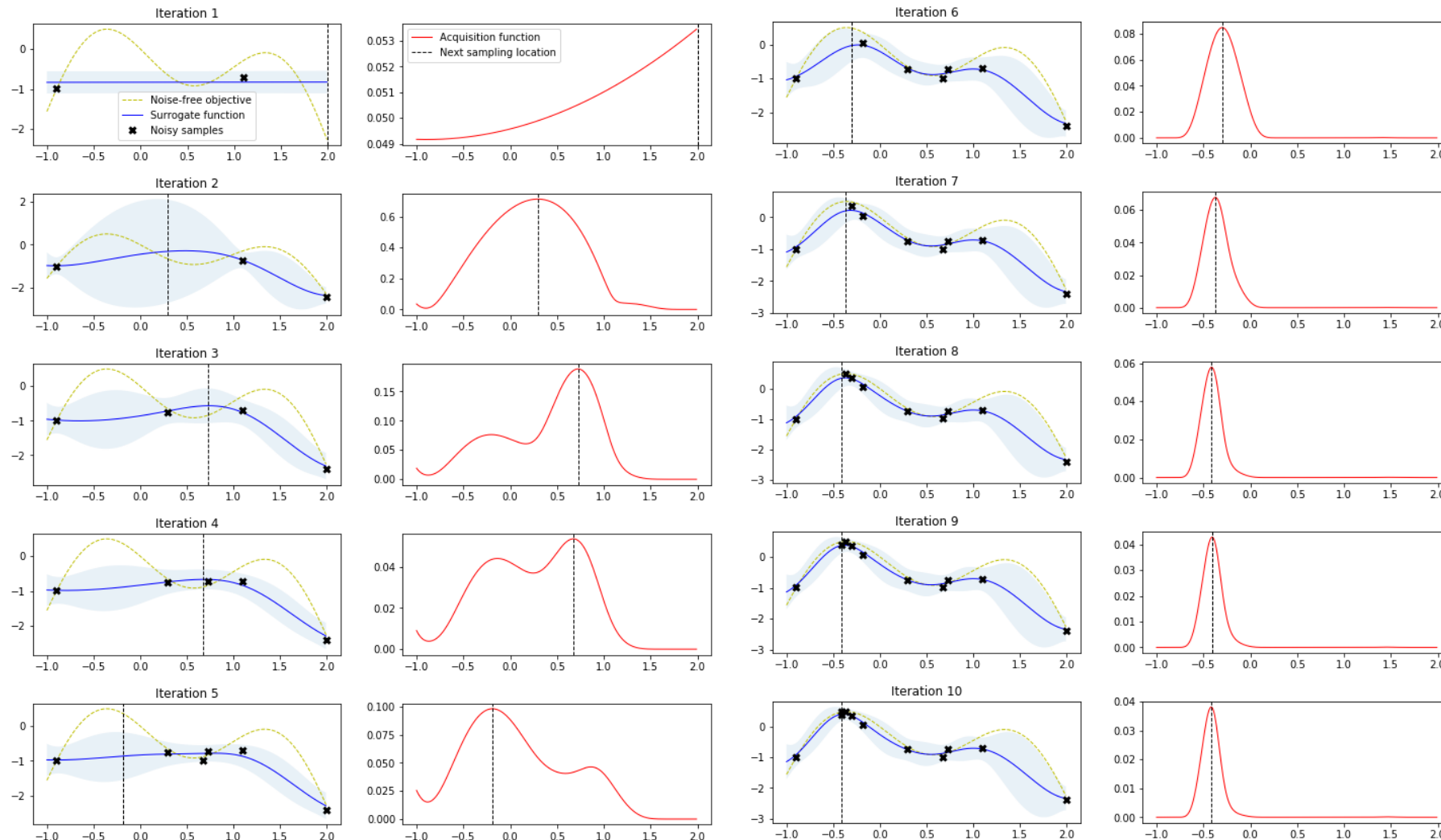
Note: Function values can be noisy too, e.g., $f(x_n) + \epsilon_n$

- Note: The regression model must also have estimate of function's uncertainty
 - Bayesian nonlinear regression, such as GP, Bayesian Neural network, etc would be ideal



Bayesian Optimization: An Illustration

- Suppose our goal is to find the maxima of $f(x)$ using BO



Acquisition Functions for BO

(assuming we are finding the minima)



Acquisition Functions: Probability of Improvement⁹

- Assume past queries $\mathcal{D}_N = (\mathbf{X}, \mathbf{f}) = (x_n, f(x_n))_{n=1}^N$ and suppose $f_{\min} = \min \mathbf{f}$
- Suppose f_{new} denotes the function's value at the next query point x_{new}
- We have an improvement if $f_{\text{new}} < f_{\min}$ (recall we are doing minimization)
- Assuming the function is real-valued, suppose the posterior predictive for x_{new} is

$$p(f_{\text{new}} | x_{\text{new}}, \mathcal{D}_N) = \mathcal{N}(f_{\text{new}} | \mu(x_{\text{new}}), \sigma^2(x_{\text{new}}))$$

- We can define a probability of improvement based acquisition function

$$A_{PI}(x_{\text{new}}) = p(f_{\text{new}} \leq f_{\min}) = \int_{-\infty}^{f_{\min}} \mathcal{N}(f_{\text{new}} | \mu(x_{\text{new}}), \sigma^2(x_{\text{new}})) df_{\text{new}} = \Phi\left(\frac{f_{\min} - \mu(x_{\text{new}})}{\sigma(x_{\text{new}})}\right)$$

- The optimal query point will be one that maximizes $A_{PI}(x_{\text{new}})$

$$x_* = \operatorname{argmax}_{x_{\text{new}}} A_{PI}(x_{\text{new}})$$

Exercise: Verify

$\Phi()$ denotes CDF of $\mathcal{N}(0,1)$

Acquisition Functions: Expected Improvement

- PI doesn't take into account the amount of improvement
- Expected Improvement (EI) takes this into account and is defined as

$$A_{EI}(x_{new}) = \mathbb{E}[f_{min} - f_{new}] = \int_{-\infty}^{f_{min}} (f_{min} - f_{new}) \mathcal{N}(f_{new} | \mu(x_{new}), \sigma^2(x_{new})) df_{new}$$

Exercise: Prove
this result

$$= (f_{min} - \mu(x_{new})) \Phi\left(\frac{f_{min} - \mu(x_{new})}{\sigma(x_{new})}\right) + \sigma(x_{new}) \mathcal{N}\left(\frac{f_{min} - \mu(x_{new})}{\sigma(x_{new})}; 0, 1\right)$$

- The optimal query point will be one that maximizes $A_{EI}(x_{new})$

$$x_* = \operatorname{argmax}_{x_{new}} A_{EI}(x_{new})$$

Focus on points where
the function has small
values (since we are
looking for its minima)

Focus on points where
the function has high
uncertainty (so that
including them
improves our estimate
of the function)

- Note that the above acquisition function trades off exploitation vs exploration
 - Will prefer points with small predictive mean $\mu(x_{new})$: Exploitation
 - Will prefer points with large predictive variance $\sigma(x_{new})$: Exploration



Acquisition Functions: Lower Confidence Bound

- Lower Confidence Bound (LCB) also takes into account exploitation vs exploration

- Used when the regression model is a Gaussian Process (GP)

- Assume the posterior predictive for a new point to be

$$p(f_{new}|x_{new}, \mathcal{D}_N) = \mathcal{N}(f_{new}|\mu(x_{new}), \sigma^2(x_{new}))$$

- The LCB based acquisition function is defined as

$$A_{LCB}(x_{new}) = \mu(x_{new}) - \kappa \sigma(x_{new})$$

- Point with the smallest LCB is selected as the next query point

$$x_* = \operatorname{argmin}_{x_{new}} A_{LCB}(x_{new})$$

- κ is a parameter to trade-off exploitation (low mean) and exploration (high variance)

- Under certain conditions, the iterative application of this acquisition function will converge to the true global optima of f (Srinivas et al. 2010)

When using BO for maximization, we use Upper Confidence Bound (UCB) defined as $A_{UCB}(x_{new}) = \mu(x_{new}) + \kappa \sigma(x_{new})$ and $x_* = \operatorname{argmax}_{x_{new}} A_{UCB}(x_{new})$



Acquisition Functions: Entropy Search

- Entropy Search (ES): Assuming \mathbf{x}_{opt} to be the optima, the ES acquisition function is

$$A_{ES}(\mathbf{x}_{new}) = \mathbb{H}[\mathbf{x}_{opt} | \mathcal{D}_N] - \mathbb{E}_{p(f_{new} | \mathbf{x}_{new})} \mathbb{H}[\mathbf{x}_{opt} | \mathcal{D}_N \cup (\mathbf{x}_{new}, f_{new})]$$

- ES directly looks at reduction in the uncertainty about the optima \mathbf{x}_{opt}
- ES requires $p(\mathbf{x}_{opt} | \mathcal{D}_N)$, computing which is hard
 - However, $p(\mathbf{x}_{opt} | \mathcal{D}_N)$ be approximated using f 's posterior $p(f | \mathcal{D}_N)$



Bayesian Optimization: The Overall Algo

- Initialize $\mathcal{D} = \{\}$
- For $n = 1, 2, \dots, N$ (or until the budget doesn't exhaust)
 - Select the next query point \mathbf{x}_n by optimizing the acquisition function

$$\mathbf{x}_n = \operatorname{argopt}_x A(x)$$

- Get function's value from the black-box oracle: $f_n = f(\mathbf{x}_n)$
- $\mathcal{D} = \{\mathcal{D} \cup (\mathbf{x}_n, f_n)\}$
- Update the regression model for f using data \mathcal{D}



BO: Some Challenges/Open Problems

- Learning the regression model for the function
 - GPs are flexible but can be expensive as N grows
 - Bayesian neural networks can be an more efficient alternative to GPs (Snoek et al, 2015)
 - Hyperparams of the regression model itself (e.g., GP cov. function, Bayesian NN hyperparam)
- High-dimensional Bayesian Optimization (optimizing functions of many variables)
 - Most existing methods work well only for a moderate-dimensional x
 - Number of function evaluations required would be quite large in high dimensions
 - Lot of recent work on this (e.g., based on dimensionality reduction)
- Multitask Bayesian Optimization (joint BO for several related functions)
 - Basic idea: If two functions are similar their optima would also be nearby



BO: Some Further Resources

- Some survey papers:
 - A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning (Brochu et al., 2010)
 - Taking the Human Out of the Loop: A Review of Bayesian Optimization (Shahriari et al., 2015)
- Some open source software libraries
 - BoTorch: Bayesian Optimization in PyTorch
 - GPflowOpt: Bayesian Optimization in Tensorflow (and using GP for modeling the function)
 - Also available in scikit-optimize



Coming Up

- More on approximate inference methods
 - Local Conjugacy
 - Variational inference
 - Sampling based inference

