Bayesian Optimization

Marc Deisenroth
Centre for Artificial Intelligence
Department of Computer Science
University College London

@mpd37
m.deisenroth@ucl.ac.uk
https://deisenroth.cc

Reading Material

- Brochu et al.: *A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning*, arXiv:1012.2599, 2012
- Shahriari et al.: *Taking the Human Out of the Loop: A Review of Bayesian Optimization*, Proceedings of the IEEE, 2016

Machine Learning Meta-Challenges

- Machine learning models are getting more and more complicated
  - Usually more parameters (e.g., deep neural networks)
- Non-convex and stochastic optimization methods have meta-parameters that are difficult to tune (learning rates, momentum parameters, ...)

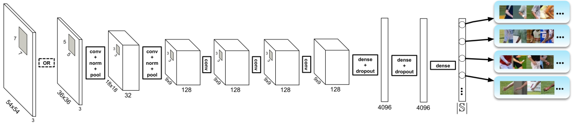
▸▸ Generally hard to apply modern techniques or reproduce results

Machine Learning Meta-Challenges

- Machine learning models are getting more and more complicated
  - ⏩ Usually more parameters (e.g., deep neural networks)
- Non-convex and stochastic optimization methods have meta-parameters that are difficult to tune (learning rates, momentum parameters, ...)

⏩ Generally hard to apply modern techniques or reproduce results

Goal: Automate the selection of critical meta-parameters
(see also: Automated Machine Learning (AutoML))



Huge interest in large neural networks

- When well-tuned, very successful for visual object identification, speech recognition, computational biology, ...
- Huge investments by Google, Facebook, Microsoft, etc.
- Many choices: number of layers, weight regularization, layer size, which nonlinearity, batch size, learning rate schedule, stopping conditions

Example: Online Latent Dirichlet Allocation

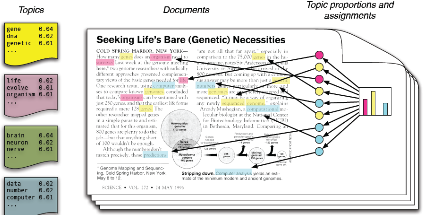
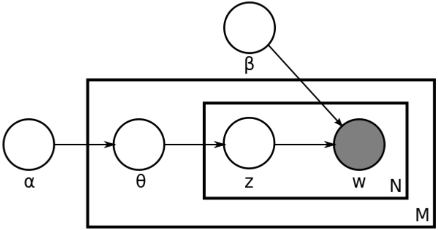
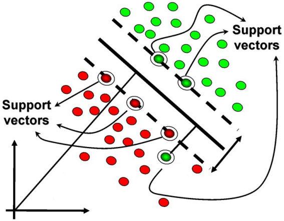


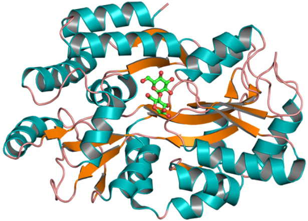
Figure by David Blei



- Hoffman et al. (2010): Approximate inference for large-scale text analysis (topic modeling) with Latent Dirichlet Allocation
- Good empirical results when well tuned
- Hyper-parameters tricky to set: Dirichlet parameters, number of topics, learning rate schedule, batch size, vocabulary size, ...

Example: Classification of DNA Sequences





- Objective: Predict which DNA sequences will bind with which proteins
- Miller et al. (2012): Latent Structural Support Vector Machine
- Hyper-parameters: margin/slack parameter, entropy parameter, convergence criterion

Search for Good Hyper-parameters

- Define an objective function to evaluate the quality of the hyper-parameters
  - Usually, we care about generalization performance
  - Cross validation to measure parameter quality

Search for Good Hyper-parameters

- Define an objective function to evaluate the quality of the hyper-parameters
  - Usually, we care about generalization performance
  - Cross validation to measure parameter quality
- Standard search procedures:
  - Manual tuning
  - Grid search
  - Random search (very simple, works surprisingly well)
  - Black magic

Search for Good Hyper-parameters

- Define an objective function to evaluate the quality of the hyper-parameters
  - Usually, we care about generalization performance
  - Cross validation to measure parameter quality
- Standard search procedures:
  - Manual tuning
  - Grid search
  - Random search (very simple, works surprisingly well)
  - Black magic
- Painful:
  - Evaluating the quality of the objective may be very expensive (e.g., time or money)
    - ▶▶ Imagine we would need to run a GPU/TPU cluster for 2 weeks
  - Many training cycles
  - Possibly noisy

Alternative Approach: Bayesian Optimization

Setting

Globally optimize a black-box objective that is expensive to evaluate (e.g., cross-validation error for a massive neural network)

- Build a probabilistic proxy model for the objective using outcomes of past experiments as training data

Alternative Approach: Bayesian Optimization

Setting

Globally optimize a black-box objective that is expensive to evaluate (e.g., cross-validation error for a massive neural network)

- Build a probabilistic proxy model for the objective using outcomes of past experiments as training data
- The proxy model is much cheaper to evaluate than the original objective

Alternative Approach: Bayesian Optimization

Setting

Globally optimize a black-box objective that is expensive to evaluate (e.g., cross-validation error for a massive neural network)

- Build a probabilistic proxy model for the objective using outcomes of past experiments as training data
- The proxy model is much cheaper to evaluate than the original objective
- Optimize cheap proxy function to determine where to evaluate the true objective next

Alternative Approach: Bayesian Optimization

Setting

Globally optimize a black-box objective that is expensive to evaluate (e.g., cross-validation error for a massive neural network)

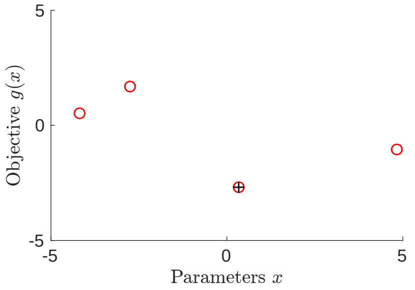
- Build a probabilistic proxy model for the objective using outcomes of past experiments as training data
- The proxy model is much cheaper to evaluate than the original objective
- Optimize cheap proxy function to determine where to evaluate the true objective next
- Standard proxy: Gaussian process

Setting (2)

- Objective: Find global minimum of objective function g:

$$\boldsymbol{x}_* = \arg\min_{\boldsymbol{x}} g(\boldsymbol{x})$$

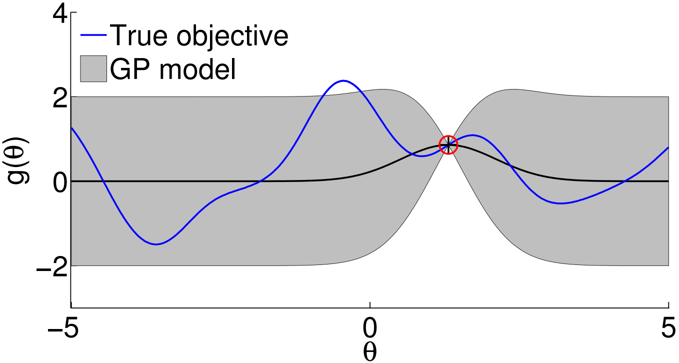
- We can evaluate the objective g pointwise, but do not have an easy functional form or gradients; observations may be noisy
- Evaluating g is costly (e.g., train a massive deep network)



Key Steps

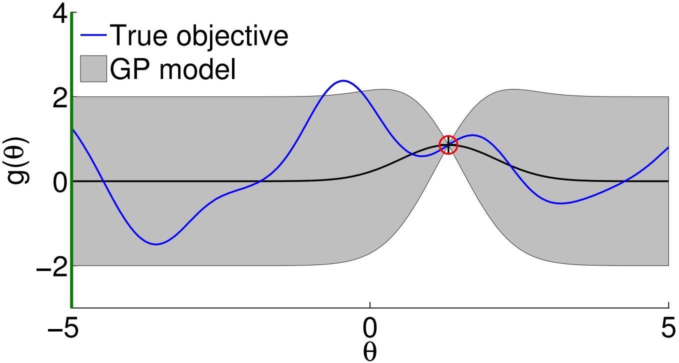
- To avoid evaluating g an excessive number of times, approximate it using a proxy function $\tilde{g}$ (which is cheap to evaluate)
- Find a global optimum $\tilde{g}(\boldsymbol{x}_*)$ of proxy function $\tilde{g}$
- Evaluate true objective g at $\boldsymbol{x}_*$
- Overall: Evaluate g only once

- To avoid evaluating g an excessive number of times, approximate it using a proxy function $\tilde{g}$ (which is cheap to evaluate)
- Find a global optimum $\tilde{g}(\boldsymbol{x}_*)$ of proxy function $\tilde{g}$
- Evaluate true objective g at $\boldsymbol{x}_*$
- Overall: Evaluate g only once
- Works well if $\tilde{g} \approx g$.
- Usually not the case ⏩ Repeat this cycle and keep updating $\tilde{g}$

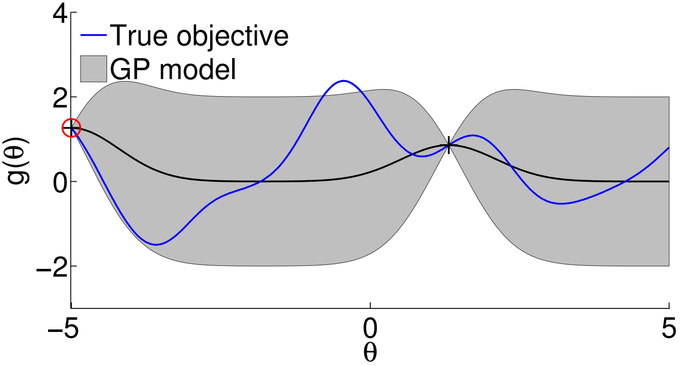


True objective
GP model
4
2
0
–2
g(θ)
–5
0
5
θ

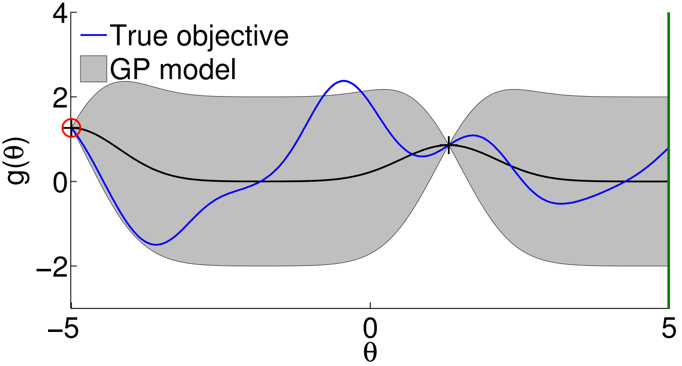
Bayesian Optimization: Illustration

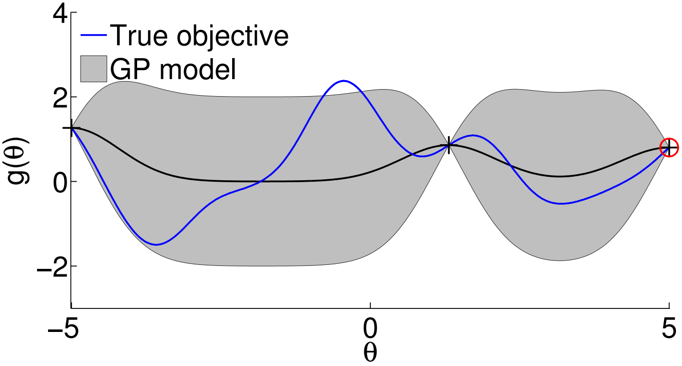


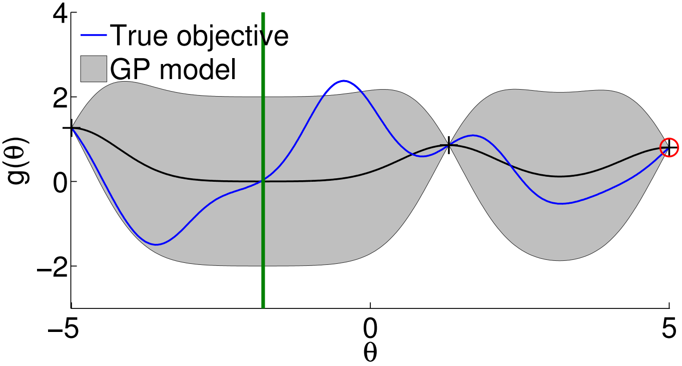
Bayesian Optimization: Illustration



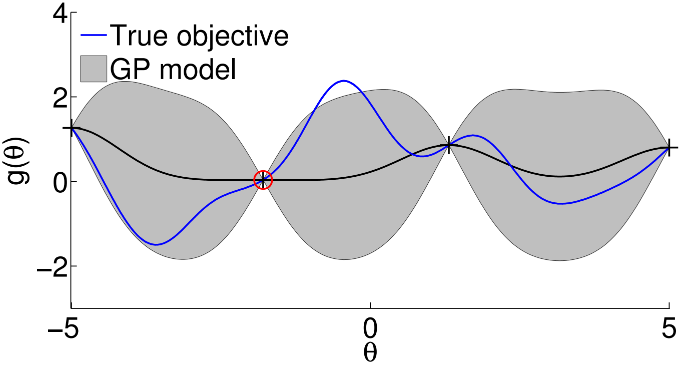
Bayesian Optimization: Illustration



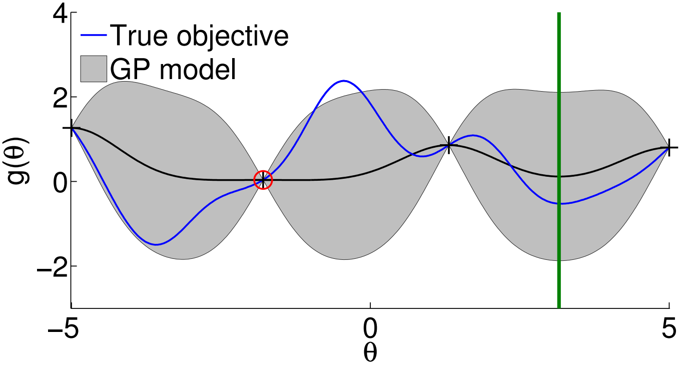

True objective
GP model
4
2
0
−2
g(θ)
−5
0
5
θ



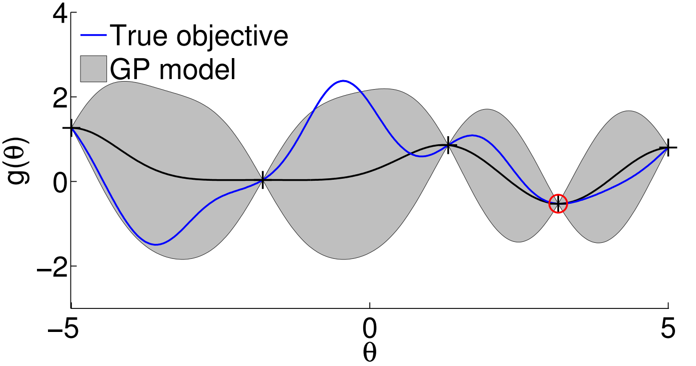
True objective
GP model
4
2
0
−2
g(θ)
−5
0
5
θ


True objective
GP model
4
2
0
−2
g(θ)
−5
0
5
θ

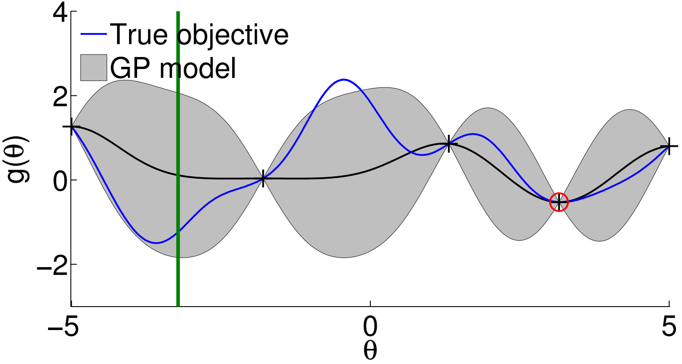
Bayesian Optimization: Illustration



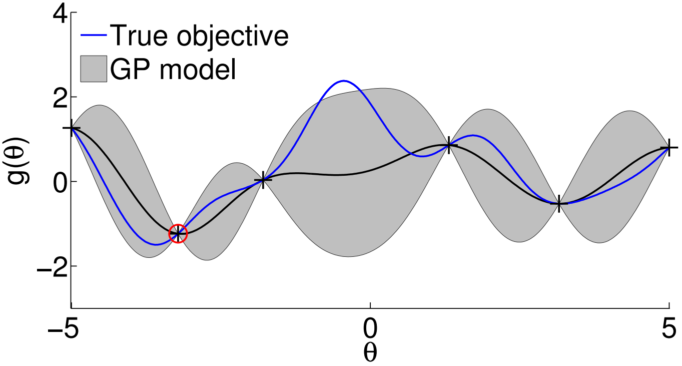
Bayesian Optimization: Illustration



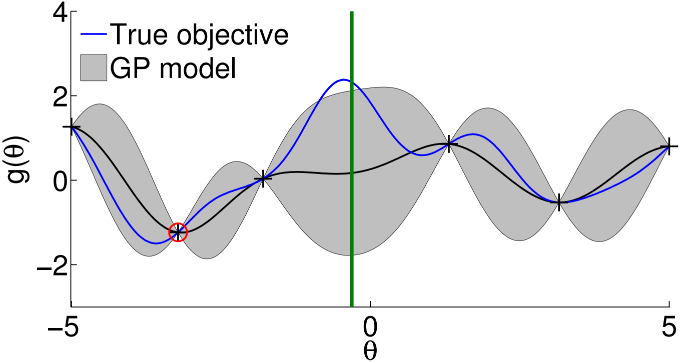
Bayesian Optimization: Illustration



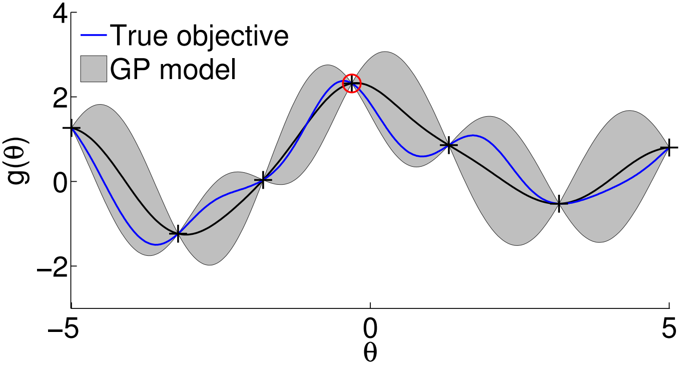
Bayesian Optimization: Illustration

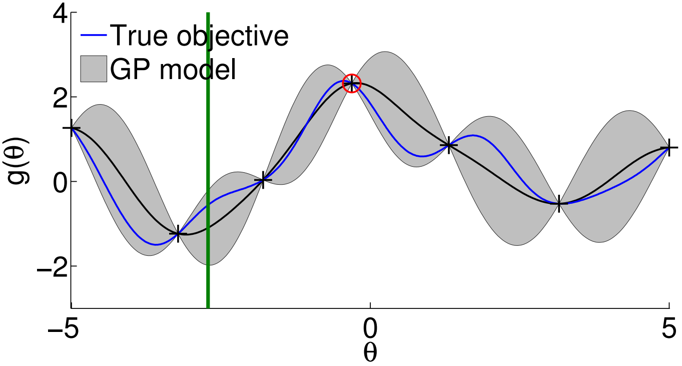


Bayesian Optimization: Illustration

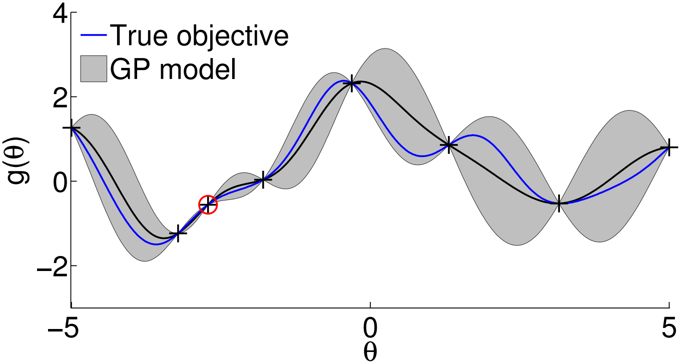


Bayesian Optimization: Illustration

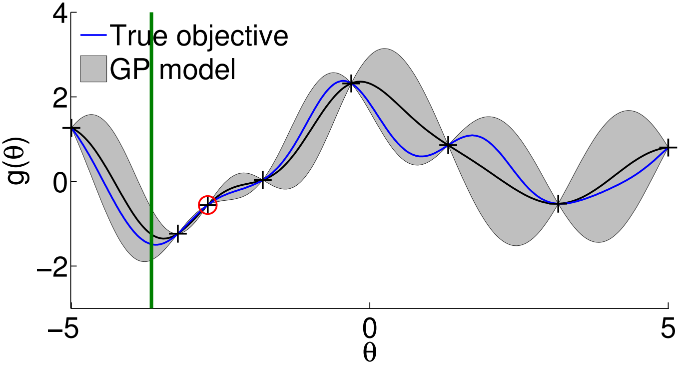



True objective
GP model
4
2
0
−2
g(θ)
−5
0
5
θ

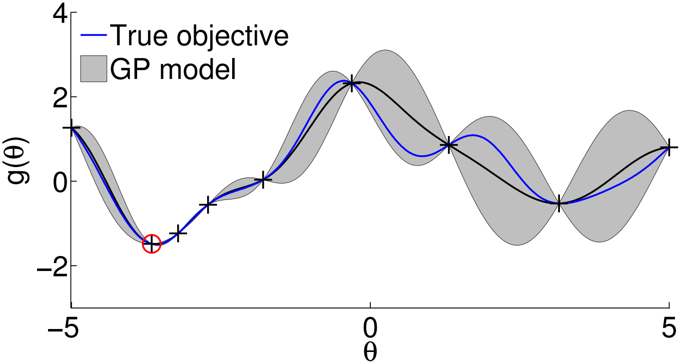
Bayesian Optimization: Illustration

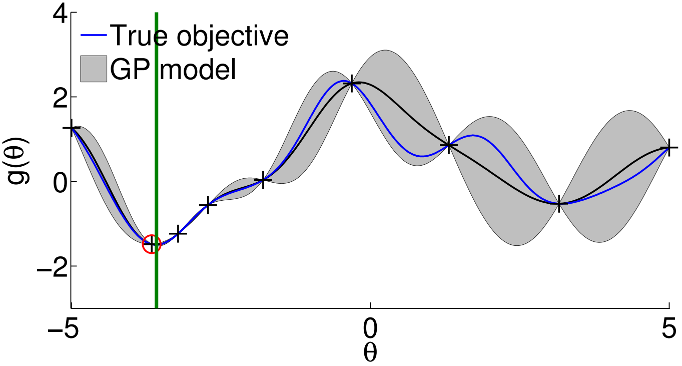


Bayesian Optimization: Illustration

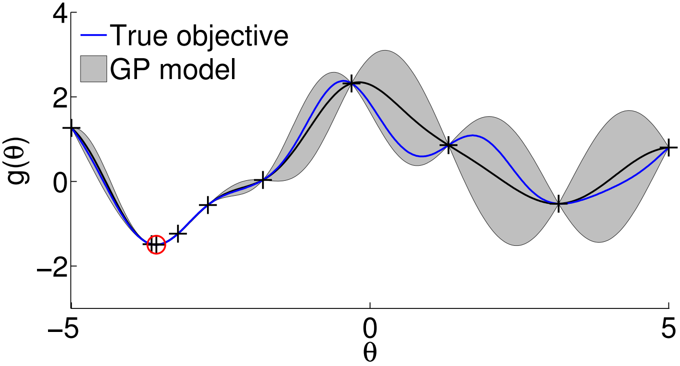


Bayesian Optimization: Illustration

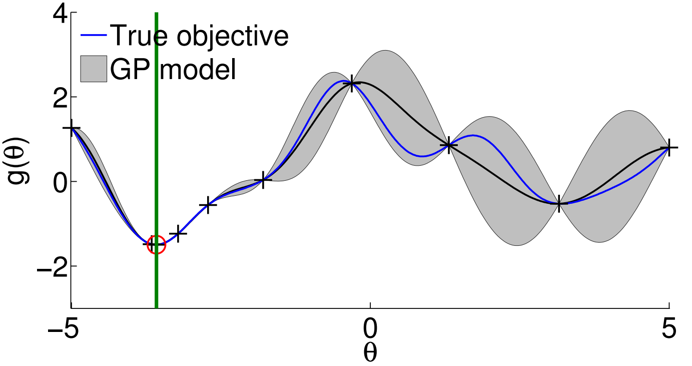



True objective
GP model
4
2
0
−2
g(θ)
−5
0
5
θ

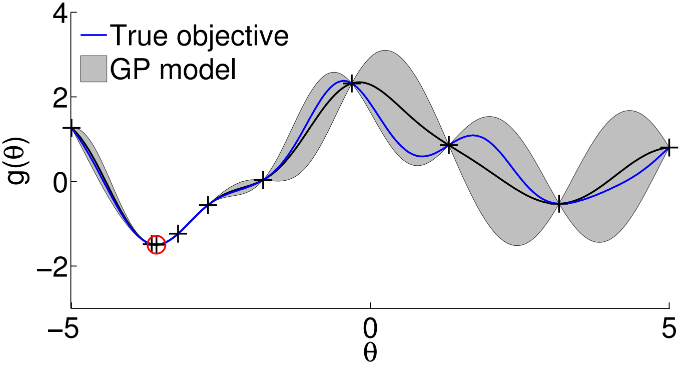
Bayesian Optimization: Illustration

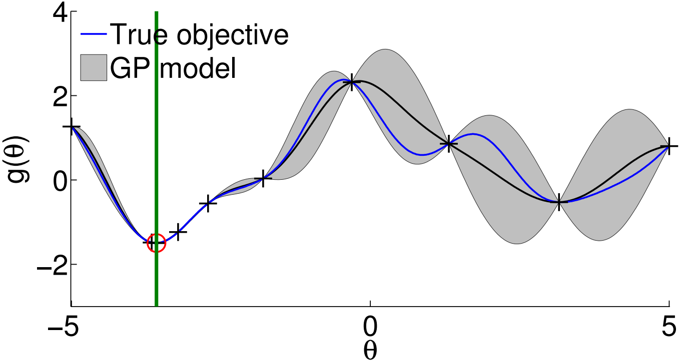


Bayesian Optimization: Illustration



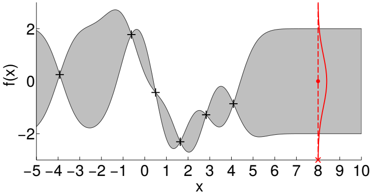
Bayesian Optimization: Illustration




True objective
GP model
4
2
0
−2
g(θ)
−5
0
5
θ

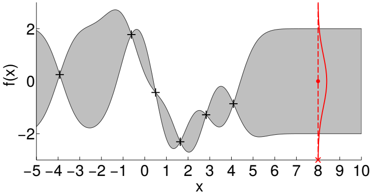
Choosing the Next Point to Evaluate the True Objective: Acquisition Functions

Using Uncertainty in Global Optimization



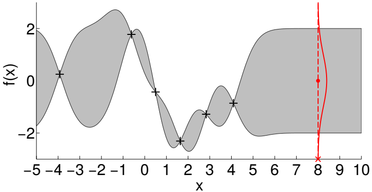
- Find a good (global) optimum
  - ⏩ Need to get out of local optima

Using Uncertainty in Global Optimization



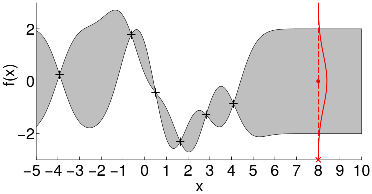
- Find a good (global) optimum
  - Need to get out of local optima
- Extrapolate from collected knowledge

Using Uncertainty in Global Optimization



- Find a good (global) optimum
  - Need to get out of local optima
- Extrapolate from collected knowledge
- GP gives us closed-form means and variances
  - Trade off exploration and exploitation
    - **Exploration:** Seek places with high variance/uncertainty
    - **Exploitation:** Seek places with low mean

Using Uncertainty in Global Optimization

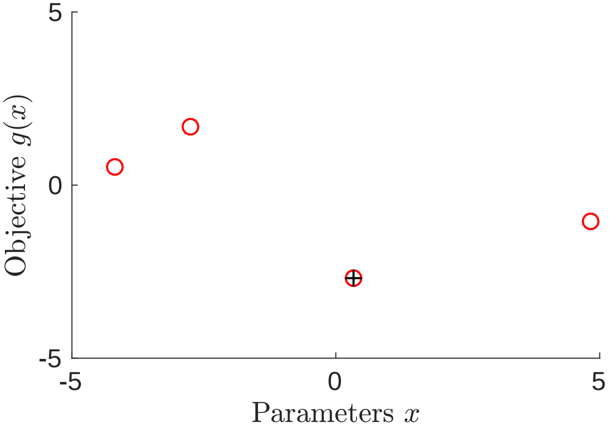


- Find a good (global) optimum
  - Need to get out of local optima
- Extrapolate from collected knowledge
- GP gives us closed-form means and variances
  - Trade off exploration and exploitation
    - **Exploration:** Seek places with high variance/uncertainty
    - **Exploitation:** Seek places with low mean
- Acquisition function α trades off exploration and exploitation for our proxy optimization

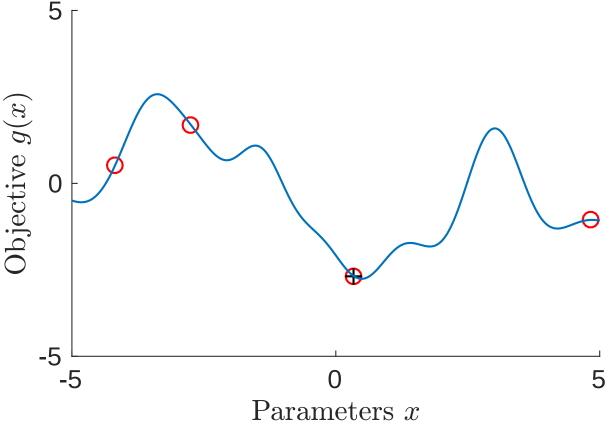
Key Steps (Pseudo-Code)

Init: Data set $\mathcal{D}_0 = \{\boldsymbol{X}_0, \boldsymbol{y}_0\}$
for iterations $t = 1, 2, ...$ **do**
    Update GP using data $\mathcal{D}_{t-1}$
    Select $\boldsymbol{x}_t = \arg\max_{\boldsymbol{x}} \alpha(\boldsymbol{x})$ by optimizing acquisition function
    Query true objective g at $\boldsymbol{x}_t$
    Augment data set $\mathcal{D}_t = \mathcal{D}_{t-1} \cup \{(\boldsymbol{x}_t, y_t)\}$
end for
Return best input in data set: $\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}} y(\boldsymbol{x})$

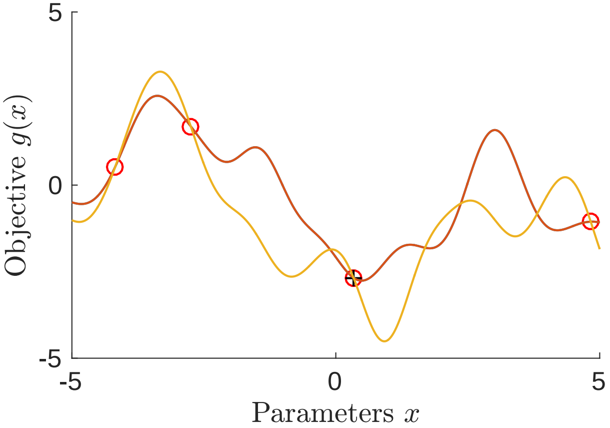
Where to Evaluate Next?



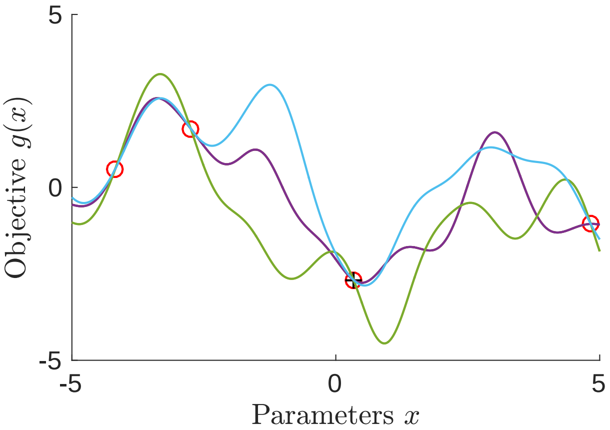
Where to Evaluate Next?



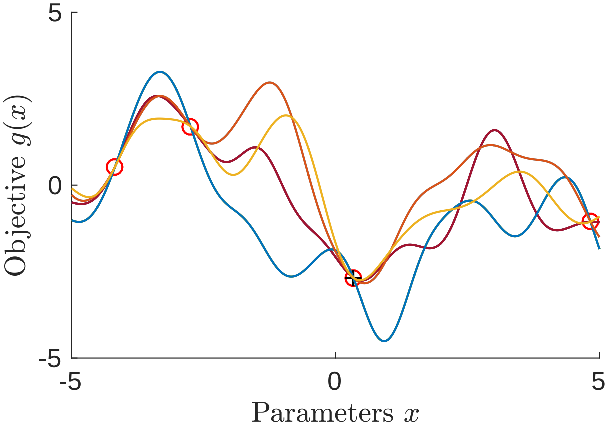
Where to Evaluate Next?

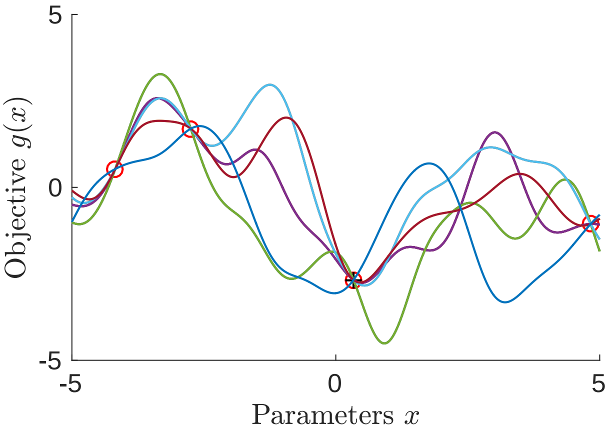


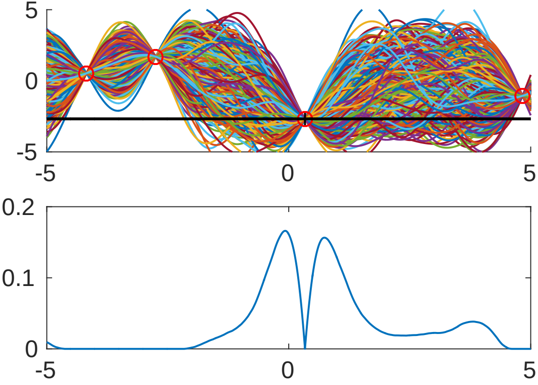
Where to Evaluate Next?



Where to Evaluate Next?

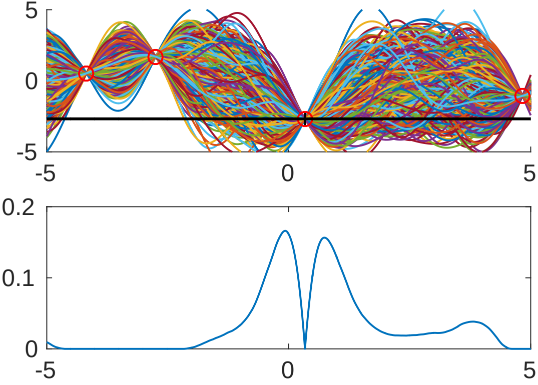



5
0
-5
-5
0
5
Objective $g(x)$
Parameters x



- Upper panel: Samples from a probabilistic proxy $\tilde{g}$

Where to Evaluate Next to Improve Most?



- Upper panel: Samples from a probabilistic proxy $\tilde{g}$
- Lower panel: Corresponding expected improvement over the best solution so far (black cross)
  - ▶▶ Evaluate g at the maximum of the expected improvement

Closed-Form Acquisition Functions

- For all $\boldsymbol{x} \in \mathbb{R}^D$ the GP posterior gives a predictive mean $\mu(\boldsymbol{x})$ variance $\sigma^2(\boldsymbol{x})$ of $g(\boldsymbol{x})$
- Define

$$\gamma(\boldsymbol{x}) = \frac{g(\boldsymbol{x}_{\text{best}}) - \mu(\boldsymbol{x})}{\sigma(\boldsymbol{x})}$$

Closed-Form Acquisition Functions

- For all $\boldsymbol{x} \in \mathbb{R}^D$ the GP posterior gives a predictive mean $\mu(\boldsymbol{x})$ variance $\sigma^2(\boldsymbol{x})$ of $g(\boldsymbol{x})$
- Define

$$\gamma(\boldsymbol{x}) = \frac{g(\boldsymbol{x}_{\text{best}}) - \mu(\boldsymbol{x})}{\sigma(\boldsymbol{x})}$$

- **Probability of Improvement (Kushner 1964):**

$$\alpha_{\text{PI}}(\boldsymbol{x}) = \Phi(\gamma(\boldsymbol{x}))$$

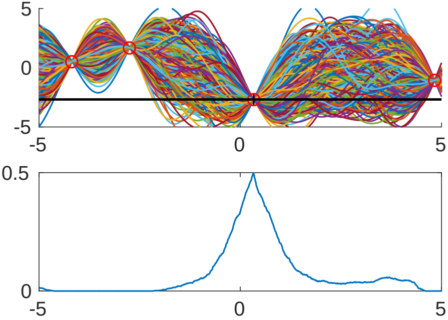
- **Expected Improvement (Mockus 1978):**

$$\alpha_{\text{EI}}(\boldsymbol{x}) = \sigma(\boldsymbol{x})\big(\gamma(\boldsymbol{x})\Phi(\gamma(\boldsymbol{x})) + \mathcal{N}\big(\gamma(\boldsymbol{x})\,|\,0,\,1\big)\big)$$

- **GP Lower Confidence Bound (Srinivas et al., 2010):**

$$\alpha_{\text{LCB}}(\boldsymbol{x}) = -(\mu(\boldsymbol{x}) - \kappa\sigma(\boldsymbol{x}))\,, \quad \kappa > 0$$

Probability of Improvement (1)

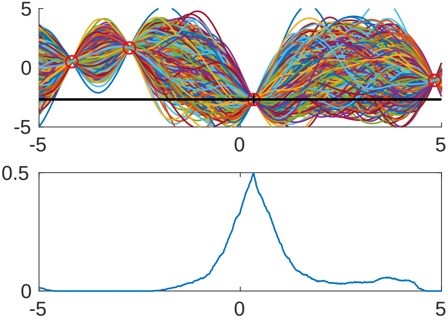


- Idea: Determine the probability that $\boldsymbol{x}_*$ leads to a better function value than the currently best one $g(\boldsymbol{x}_{\text{best}})$
- Sampling-based setting: Sample N functions g_i; at every input $\boldsymbol{x}$ compute a Monte-Carlo estimate

$$\alpha_{\text{PI}}(\boldsymbol{x}) = p(g(\boldsymbol{x}) < g(\boldsymbol{x}_{\text{best}})) \approx \frac{1}{N}\sum_{i=1}^{N} \delta\big(g_i(\boldsymbol{x}) < g(\boldsymbol{x}_{\text{best}})\big)$$

▶▶ Can lead to continued exploitation in an ϵ-region around $\boldsymbol{x}_{\text{best}}$.

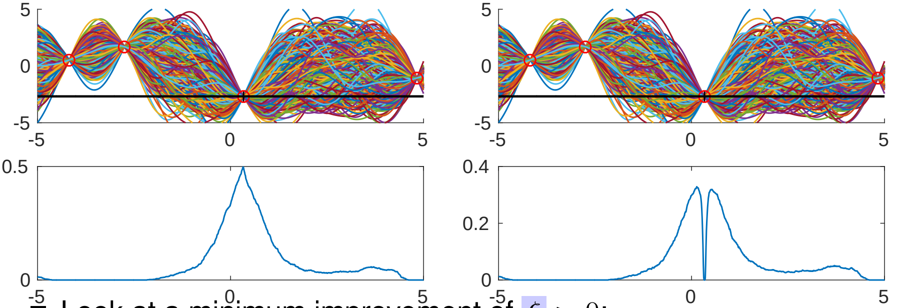
Probability of Improvement (1)



- Idea: Determine the probability that $\boldsymbol{x}_*$ leads to a better function value than the currently best one $g(\boldsymbol{x}_{\text{best}})$
- Sampling-based setting: Sample N functions g_i; at every input $\boldsymbol{x}$ compute a Monte-Carlo estimate

$$\alpha_{\text{PI}}(\boldsymbol{x}) = p(g(\boldsymbol{x}) < g(\boldsymbol{x}_{\text{best}})) \approx \frac{1}{N}\sum_{i=1}^{N} \delta\big(g_i(\boldsymbol{x}) < g(\boldsymbol{x}_{\text{best}})\big)$$

▸▸ Can lead to continued exploitation in an ϵ-region around $\boldsymbol{x}_{\text{best}}$.
▸▸ Introduce a "slack variable" ξ for more aggressive exploration



- Look at a minimum improvement of $\xi > 0$:

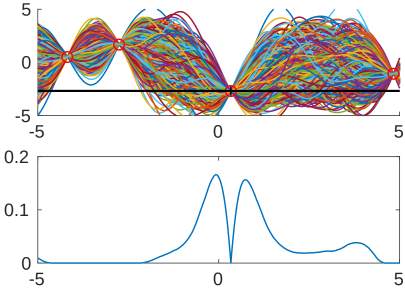
$$\alpha_{\text{PI}}(\boldsymbol{x}) = p(g(\boldsymbol{x}) < g(\boldsymbol{x}_{\text{best}}) - \xi) \approx \frac{1}{N}\sum_{i=1}^{N} \delta\big(g_i(\boldsymbol{x}) < g(\boldsymbol{x}_{\text{best}}) - \xi\big)$$

- If $f \sim GP$ and $p(g(\boldsymbol{x})) = \mathcal{N}\big(\mu(\boldsymbol{x}), \sigma(\boldsymbol{x})\big)$:

$$\alpha_{\text{PI}}(\boldsymbol{x}) = \Phi(\gamma(\boldsymbol{x}, \xi)), \qquad \gamma(\boldsymbol{x}, \xi) = \frac{g(\boldsymbol{x}_{\text{best}}) - \xi - \mu(\boldsymbol{x})}{\sigma(\boldsymbol{x})}$$

Expected Improvement

- Idea: Quantify the amount of improvement
- Sampling-based scenario, where $g_i \sim p(f)$:

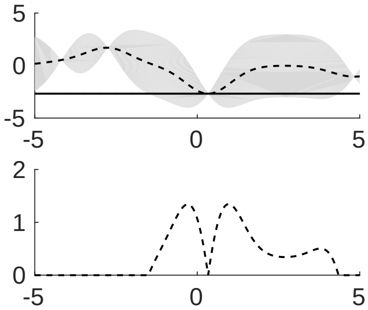


$$\alpha_{\text{EI}}(\boldsymbol{x}) = \mathbb{E}[\max\{0, g(\boldsymbol{x}_{\text{best}}) - g(\boldsymbol{x})\}]$$
$$\approx \frac{1}{N}\sum_{i=1}^{N} \max\{0, g(\boldsymbol{x}_{\text{best}}) - g_i(\boldsymbol{x})\}$$

- If $f \sim GP$, we have a closed-form expression:

$$\alpha_{\text{EI}}(\boldsymbol{x}) = \sigma(\boldsymbol{x})\big(\gamma(\boldsymbol{x})\Phi(\gamma(\boldsymbol{x})) + \mathcal{N}\big(\gamma(\boldsymbol{x})\,|\,0,\,1\big)\big)$$

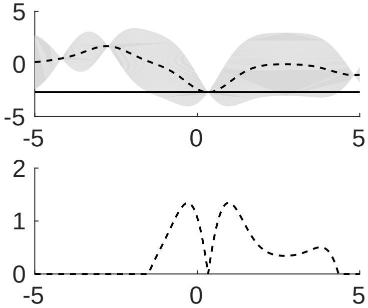
- Slack-variable approach also possible (similar to PI)



- Use the predictive mean $\mu(\boldsymbol{x})$ and variance $\sigma^2(\boldsymbol{x})$ of the GP prediction directly for targeted exploration by means of the acquisition function

$$\alpha_{\text{LCB}}(\boldsymbol{x}_t) = -\left(\mu(\boldsymbol{x}_t) - \sqrt{\kappa}\sigma(\boldsymbol{x}_t)\right)$$

GP-Lower Confidence Bound (2)



- More generally, we can get regret bounds for iteration-dependent κ (Srinivas et al., 2010)

$$\alpha_{\text{LCB}}(\boldsymbol{x}_t) = -\big(\mu(\boldsymbol{x}_t) - \sqrt{\kappa_t}\sigma(\boldsymbol{x}_t)\big)$$

where $\kappa_t \in \mathcal{O}(\log t)$ grows with the iteration t

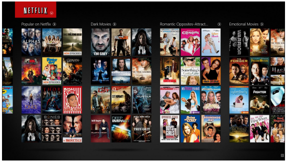
▶▶ Continue exploration

Optimizing the Acquisition Function

- Optimizing the acquisition function requires us to run a global optimizer inside Bayesian optimization
- What have we gained?

Optimizing the Acquisition Function

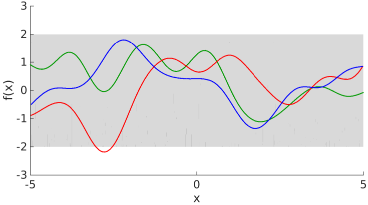
- Optimizing the acquisition function requires us to run a global optimizer inside Bayesian optimization
- What have we gained?
- Evaluating the acquisition function is cheap compared to evaluating the true objective
  - We can afford evaluating it many times

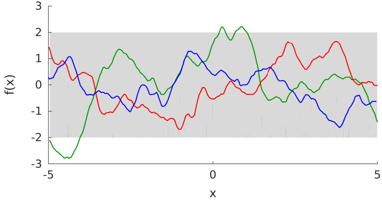


Limitations

- Getting the function model (e.g., covariance function) wrong can be catastrophic
- Limited scalability in the number of dimensions and/or evaluations of the true objective function
  Why?

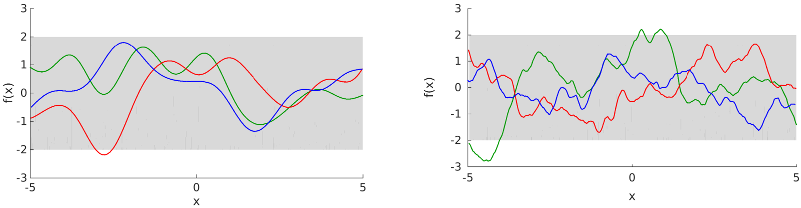
Poor Model Choice





- Covariance function selection is crucial for good performance
  - Choose a sufficiently flexible and adaptive kernel, e.g., Matérn (but not the squared exponential (Gaussian))

Poor Model Choice



- Covariance function selection is crucial for good performance
  ▶▶ Choose a sufficiently flexible and adaptive kernel, e.g., Matérn (but not the squared exponential (Gaussian))
- Nice side-effect of Matérn: Exploration is more encouraged than with the Gaussian kernel

Choosing Covariance Functions

- Structured SVM for Protein Motif Finding (Miller et al., 2012)
- Optimize hyper-parameters of SSVM using BO (Snoek et al., 2012)

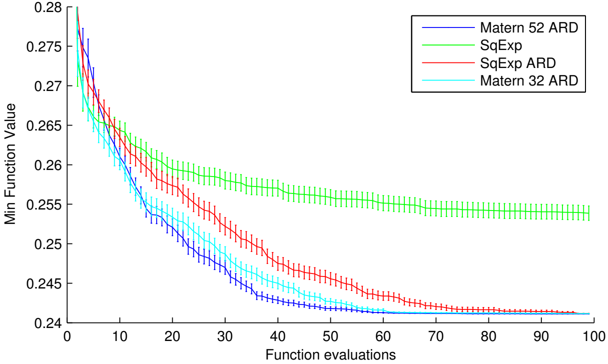


Figure: Figure from Snoek et al. (2012)

Gaussian Process Hyper-Parameters

- Empirical Bayes (maximize the marginal likelihood) can fail horribly, especially in the early stages of Bayesian optimization when we have only a few data points

Gaussian Process Hyper-Parameters

- Empirical Bayes (maximize the marginal likelihood) can fail horribly, especially in the early stages of Bayesian optimization when we have only a few data points
- Solution: Integrate out the GP hyper-parameters $\boldsymbol{\theta}$ by Markov Chain Monte Carlo (MCMC) sampling (e.g., slice sampling)

- Empirical Bayes (maximize the marginal likelihood) can fail horribly, especially in the early stages of Bayesian optimization when we have only a few data points
- Solution: Integrate out the GP hyper-parameters $\boldsymbol{\theta}$ by Markov Chain Monte Carlo (MCMC) sampling (e.g., slice sampling)
- Look at integrated acquisition function

$$\alpha(\boldsymbol{x}) = \mathbb{E}_{\boldsymbol{\theta}}[\alpha(\boldsymbol{x}, \boldsymbol{\theta})] = \int \alpha(\boldsymbol{x}, \boldsymbol{\theta}) p(\boldsymbol{\theta}) d\boldsymbol{\theta}$$

$$\approx \frac{1}{K} \sum_{k=1}^{K} \alpha(\boldsymbol{x}, \boldsymbol{\theta}^{(k)}) \,, \quad \boldsymbol{\theta}^{(k)} \sim \underbrace{p(\boldsymbol{\theta} | \boldsymbol{X}_n, \boldsymbol{y}_n)}_{\text{hyper-parameter posterior}}$$

Integrating out GP Hyper-parameters

- Online LDA (Hoffman et al., 2010) for topic modeling
- Two critical hyper-parameters that control the learning rate learned by BO (Snoek et al., 2012)

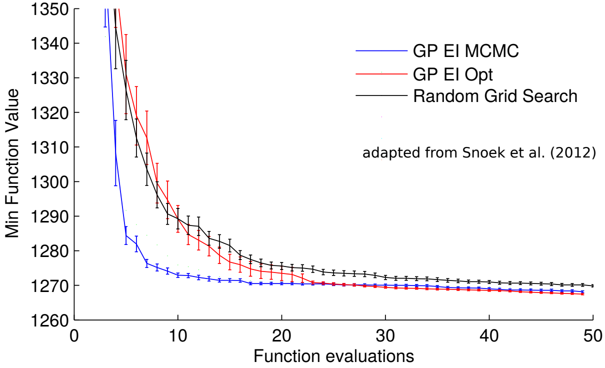
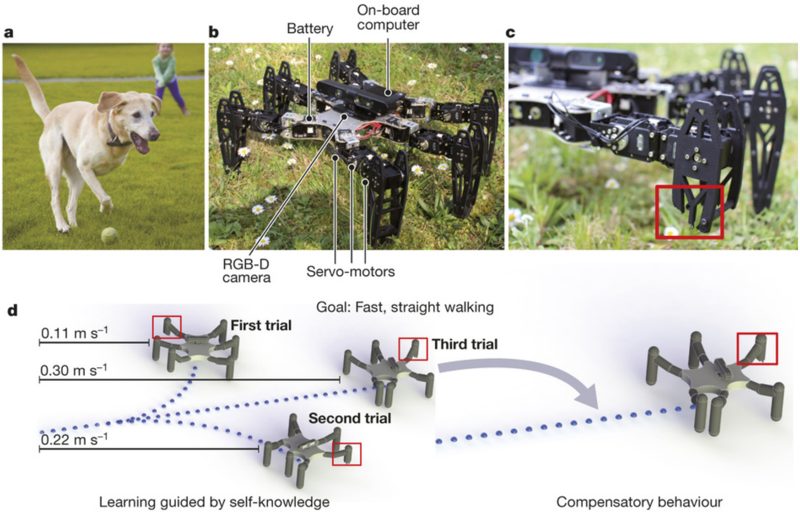


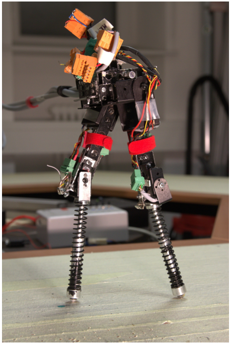
Figure: Figure from Snoek et al. (2012)

Robots That Learn to Recover from Damage



Cully et al. (2015)

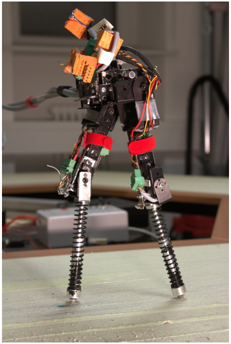
- Fragile bipedal robot
  - ⏩ Only few experiments feasible
- Maximize robustness and walking speed
- 4 motors:
  2 actuated hips + 2 actuated knees
- Controller implemented as a finite-state-machine (8 parameters)



Calandra et al. (2015)

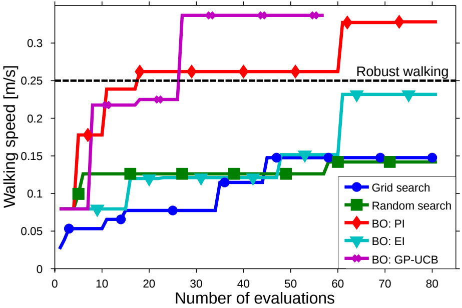
Application Example: Controller Learning in

- Fragile bipedal robot
  - ⏩ Only few experiments feasible
- Maximize robustness and walking speed
- 4 motors:
  2 actuated hips + 2 actuated knees
- Controller implemented as a finite-state-machine (8 parameters)
- Good parameters found after 80–100 experiments
- Substantial speed-up compared to manual parameter search



Calandra et al. (2015)

Comparison



- Squared exponential covariance function
- Learned GP hyper-parameters (no MCMC for integrating them out)

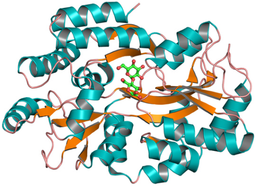
Further Topics in BO

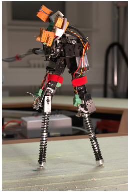
- Entropy-based acquisition functions: Directly describe the distribution over the best input location (Hennig & Schuler, 2012; Hernández-Lobato et al., 2014)
- Non-myopic Bayesian optimization (e.g., Osborne et al., 2009)
- High-dimensional optimization (e.g., Wang et al., 2016)
- Large-scale Bayesian optimization (Hutter et al., 2014)
- Efficient optimization of acquisition functions (Wilson et al., 2018)
- Non-GP Bayesian optimization (Hutter et al., 2014; Snoek et al., 2015)
- Constraints (e.g., Gelbart et al., 2014)
- Automated machine learning (e.g., Feurer et al., 2015)
- Multi-tasking, parallelizing, resource allocation, ... (e.g., Swersky et al., 2014; Snoek et al., 2012; Wilson et al., 2018)

Software

- BoTorch `https://github.com/pytorch/botorch` (Balandat et al., 2019)
- BayesOpt `https://bitbucket.org/rmcantin/bayesopt/` (Martinez-Cantin, 2014)
- Spearmint `https://github.com/HIPS/Spearmint`
- Pybo `https://github.com/mwhoffman/pybo` (Hoffman & Shariari)
- GPyOpt `https://github.com/SheffieldML/GPyOpt` (Gonzalez et al.)
- Matlab toolbox (bayesopt)

Summary





- Global optimization of black-box functions, which are expensive to evaluate ⏩ Meta-challenges in machine learning, Auto-ML
- Use a probabilistic proxy model that is cheap to evaluate and use this to suggest next experiments
- Acquisition function trades off exploration and exploitation

References I

[1] M. Balandat, B. Karrer, D. R. Jiang, S. Daulton, B. Letham, A. G. Wilson, and E. Bakshy. BoTorch: Programmable Bayesian Optimization in PyTorch. *arXiv: 1910.06403*, 2019.

[2] E. Brochu, V. M. Cora, and N. de Freitas. A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning. Technical Report TR-2009-023, Department of Computer Science, University of British Columbia, 2009.

[3] R. Calandra, A. Seyfarth, J. Peters, and M. P. Deisenroth. Bayesian Optimization for Learning Gaits under Uncertainty. *Annals in Mathematics and Artificial Intelligence*, pages 1–19, 2015.

[4] Y. Chen, A. Huang, Z. Wang, I. Antonoglou, J. Schrittwieser, D. Silver, and N. de Freitas. Bayesian Optimization in AlphaGo. *arXiv:1812.06855*, 2018.

[5] A. Cully, J. Clune, D. Tarapore, and J.-B. Mouret. Robots That Can Adapt Like Animals. *Nature*, 521:503–507, 2015.

[6] M. Feurer, A. Klein, K. Eggensperger, J. Springenberg, M. Blum, and F. Hutter. Efficient and Robust Automated Machine Learning. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, pages 2962–2970. Curran Associates, Inc., 2015.

[7] M. Gelbart, J. Snoek, and R. P. Adams. Bayesian Optimization with Unknown Constraints. In *International Conference on Uncertainty in Artificial Intelligence*, pages 1–14, 2014.

[8] P. Hennig and C. J. Schuler. Entropy Search for Information-Efficient Global Optimization. *Journal of Machine Learning Research*, 13:1809–1837, 2012.

[9] J. M. Hernández-Lobato, M. W. Hoffman, and Z. Ghahramani. Predictive Entropy Search for Efficient Global Optimization of Black-box Functions. In *Advances in Neural Information Processing Systems*, pages 1–9, 2014.

[10] M. D. Hoffman, D. M. Blei, and F. Bach. Online Learning for Latent Dirichlet Allocation. *Advances in Neural Information Processing Systems*, 23:1–9, 2010.

[11] F. Hutter, H. Hoos, and K. Leyton-Brown. An Efficient Approach for Assessing Hyperparameter Importance. In *Proceedings of International Conference on Machine Learning*, pages 754–762, June 2014.

References II

[12] D. R. Jones, M. Schonlau, and W. J. Welch. Efficient Global Optimization of Expensive Black-Box Functions. *Journal of Global Optimization*, 13(4):455–492, Dec. 1998.

[13] H. J. Kushner. A New Method of Locating the Maximum Point of an Arbitrary Multipeak Curve in the Presence of Noise. *Journal of Basic Engineering*, 86:97, 1964.

[14] D. Lizotte. *Practical Bayesian Optimization*. PhD thesis, University of Alberta, Edmonton, Alberta, 2008.

[15] R. Martinez-Cantin. BayesOpt: A Bayesian Optimization Library for Nonlinear Optimization, Experimental Design and Bandits. *Journal of Machine Learning Research*, 15:3915–3919, 2014.

[16] R. Martinez-Cantin, N. de Freitas, A. Doucet, and J. Castellanos. Active Policy Learning for Robot Planning and Exploration under Uncertainty. In *Proceedings of Robotics: Science and Systems III*, Atlanta, GA, USA, June 2007.

[17] K. Miller, M. P. Kumar, B. Packer, D. Goodman, and D. Koller. Max-Margin Min-Entropy Models. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, volume 22, pages 779–787, 2012.

[18] J. Mockus, V. Tiesis, and A. Zilinska. The Application of Bayesian Methods for Seeking the Extremum. *Towards Global Optimization*, 2:117–129, 1978.

[19] M. A. Osborne, R. Garnett, and S. J. Roberts. Gaussian Processes for Global Optimization. In *Proceedings of the International Conference on Learning and Intelligent Optimization*, 2009.

[20] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. De Freitas. Taking the Human out of the Loop: A Review of Bayesian Optimization. *Proceedings of the IEEE*, 104(1):148–175, 2016.

[21] J. Snoek, H. Larochelle, and R. P. Adams. Practical Bayesian Optimization of Machine Learning Algorithms. In *Advances in Neural Information Processing Systems*, 2012.

[22] J. Snoek, O. Rippel, K. Swersky, R. Kiros, N. Satish, N. Sundaram, M. M. A. Patwary, Prabhat, and R. P. Adams. Scalable Bayesian Optimization Using Deep Neural Networks. In *Proceedings of the International Conference on Machine Learning*, 2015.

[23] J. T. Springenberg, A. Klein, S. Falkner, and F. Hutter. Bayesian Optimization with Robust Bayesian Neural Networks. In *Advances in Neural Information Processing Systems 29*, December 2016.

[24] N. Srinivas, A. Krause, S. Kakade, and M. Seeger. Gaussian Process Optimization in the Bandit Setting: No Regret and Experimental Design. In *Proceedings of the International Conference on Machine Learning*, 2010.

[25] K. Swersky, J. Snoek, and R. P. Adams. Freeze-Thaw Bayesian Optimization. Technical report, 2014.

[26] D. Ulmasov, C. Baroukh, B. Chachuat, M. P. Deisenroth, and R. Misener. Bayesian Optimization with Dimension Scheduling: Application to Biological Systems. In *Proceedings of the European Symposium on Computer Aided Process Engineering*, 2016.

[27] Z. Wang, F. Hutter, M. Zoghi, D. Matheson, and N. De Freitas. Bayesian Optimization in a Billion Dimensions via Random Embeddings. In *Journal of Artificial Intelligence Research*, volume 55, pages 361–367, 2016.

[28] J. T. Wilson, F. Hutter, and M. P. Deisenroth. Maximizing Acquisition Functions for Bayesian Optimization. In *arXiv:1805.10196*, 2018.