Computing Policies with Gaussian Processes

Bayesian Optimization by Roman Garnett

Mengjia Zhu

University of Manchester
mengjia.zhu@manchester.ac.uk

Bayesian Optimization Book Club Chapter 8 - May 16, 2024

Overview

1. Recap
2. Problem formulation
3. Different Acquisition functions
4. Summary

---

input: initial dataset $\mathcal{D}$ ▸ can be empty
repeat
  $x \leftarrow \textsc{policy}(\mathcal{D})$ ▸ select the next observation location
  $y \leftarrow \textsc{observe}(x)$ ▸ observe at the chosen location
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(x, y)\}$ ▸ update dataset
until termination condition reached ▸ e.g., budget exhausted
return $\mathcal{D}$

Recap - Chapter 7

input: initial dataset $\mathcal{D}$ ▸ can be empty
repeat
  $x \leftarrow \textsc{policy}(\mathcal{D})$ ▸ select the next observation location
  $y \leftarrow \textsc{observe}(x)$ ▸ observe at the chosen location
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(x, y)\}$ ▸ update dataset
until termination condition reached ▸ e.g., budget exhausted
return $\mathcal{D}$

Common BO policies **w/o** considering noise and specific obj. fun. model (model-agnostic)

- One-step lookahead (see Table 7.1): Expected improvement, Knowledge gradient, Probability of improvement, Mutual information, Posterior mean
- Policies from multi-armed bandits: Upper confidence bound, Thompson sampling

Scope of Chapter 8

input: initial dataset $\mathcal{D}$	▸ can be empty
repeat	
$\quad x \leftarrow \textsc{policy}(\mathcal{D})$	▸ select the next observation location
$\quad y \leftarrow \textsc{observe}(x)$	▸ observe at the chosen location
$\quad \mathcal{D} \leftarrow \mathcal{D} \cup \{(x, y)\}$	▸ update dataset
until termination condition reached	▸ e.g., budget exhausted
return $\mathcal{D}$	

- **Chapter 7**: Common BO policies w/o noise and discussed under model-agnostic settings
- **Chapter 8**: How to compute policies, focus on
  - Model of the obj. fun.: Gaussian Processes (GPs)
  - Model of the noise observation: exact or additive Gaussian noise

Problem formulation

The University of Manchester

$$x \in \arg\max_{x' \in \mathcal{X}} \; \alpha(x'; \mathcal{D}) \tag{1}$$

Goal: compute/approximate the selected acq. fun. w.r.t the GP models and the selected noise models, which will then be optimized to identify x (acq. fun. defines the policy)

- Exact computation (when possible): Expected improvement, Probability of improvement, Knowledge gradient in **discrete** domains, Upper confidence bound
- Effective approximation schemes: Knowledge gradient in **continuous** domains, Mutual information, Thompson sampling

Whether can be computed exactly is model-dependent. Here, the discussion is based on GP only.

Problem formulation

$$x \in \arg\max_{x' \in \mathcal{X}} \; \alpha(x'; \mathcal{D}) \tag{1}$$

Goal: compute/approximate the selected acq. fun. w.r.t the GP models and the selected noise models, which will then be optimized to identify x (acq. fun. defines the policy)

- Exact computation (when possible): Expected improvement, Probability of improvement, Knowledge gradient in **discrete** domains, Upper confidence bound
- Effective approximation schemes: Knowledge gradient in **continuous** domains, Mutual information, Thompson sampling

Note: due to time constraints, only cover *Expected improvement* from exact computation and *Thompson sampling* from approximation schemes. The book provided very detailed and useful discussions (recommend reading thoroughly if interested!)

Whether can be computed exactly is model-dependent . Here, the discussion is based on GP only .

Notation

The GP belief for $x \in \mathcal{X}$:

$$p\ (f \mid \mathcal{D}) = \mathcal{GP}(f\ ; \mu_{\mathcal{D}}, K_{\mathcal{D}}) \tag{2}$$

The predictive distribution for $\phi = f(x)$ at a proposed location x (**obj. fun. evaluation**):

$$p\ (\phi \mid x, \mathcal{D}) = \mathcal{N}(\phi\ ; \mu, \sigma^2) \tag{3}$$

The predictive distribution for y measured at x (**noisy observation**):

$$\begin{aligned} \text{Indep. zero-mean additive Gaussian noise: } & p\ (y \mid \phi, \sigma_n) = \mathcal{N}(y\ ; \phi, \sigma_n^2) \\ \text{Gaussian noise depend on } x\text{: } & p\ (y \mid x, \mathcal{D}, \sigma_n) = \mathcal{N}(y\ ; \mu, \sigma^2 + \sigma_n^2) = \mathcal{N}(y\ ; \mu, s^2) \end{aligned} \tag{4}$$

- $f : \mathcal{X} \to \mathbb{R}$: obj. fun. with GP belief
- $\mathcal{D} = (\mathbf{x}, \mathbf{y})$: observations
- $\mu = \mu_{\mathcal{D}}(x)$: Predictive mean of ϕ
- $\sigma^2 = K_{\mathcal{D}}(x, x)$: Predictive variance of ϕ
- σ_n^2: variance of additive Gaussian noise
- s^2: Predictive variance of y
- Exact measurements: $y = \phi$, $s^2 = \sigma^2$

Notes

- Often, numerical solvers (*e.g.*, PSO, gradient-descent) will be used to optimize (1) to find x
  - Iterative solvers: at each step, a new candidate x is proposed, whose corresponding acq. fun. value will be evaluated. The solver terminates when the set tolerance is met or a maximum number of fun. eval. is reached
  - Why: (1) can be hard or impossible to solve analytically

Notes

- Often, numerical solvers (*e.g.*, PSO, gradient-descent) will be used to optimize (1) to find x
  - Iterative solvers: at each step, a new candidate x is proposed, whose corresponding acq. fun. value will be evaluated. The solver terminates when the set tolerance is met or a maximum number of fun. eval. is reached
  - Why: (1) can be hard or impossible to solve analytically
- *Goal of this chapter*: compute analytically or approximate the fun. form of the acq. fun so that candidate x during the optimization procedure can be evaluated
  - When evaluating the acq. fun., candidate x is treated as given and fixed (proposed by the solver)

Notes

- Often, numerical solvers (*e.g.*, PSO, gradient-descent) will be used to optimize (1) to find x
  - Iterative solvers: at each step, a new candidate x is proposed, whose corresponding acq. fun. value will be evaluated. The solver terminates when the set tolerance is met or a maximum number of fun. eval. is reached
  - Why: (1) can be hard or impossible to solve analytically
- *Goal of this chapter*: compute analytically or approximate the fun. form of the acq. fun so that candidate x during the optimization procedure can be evaluated
  - When evaluating the acq. fun., candidate x is treated as given and fixed (proposed by the solver)
- This presentation: provide a summary
- Mathematical derivations: book

One-step lookahead acquisition functions

Expected marginal gain to a utility fun.: $$\alpha\ (x\ ;\ \mathcal{D}) = \int [\ u\ (\mathcal{D}') - u\ (\mathcal{D})\ ]\ \mathcal{N}(y; \mu, s^2)\ dy \quad (5)$$

How to determine the computation methods for acq. fun.?

One-step lookahead acquisition functions

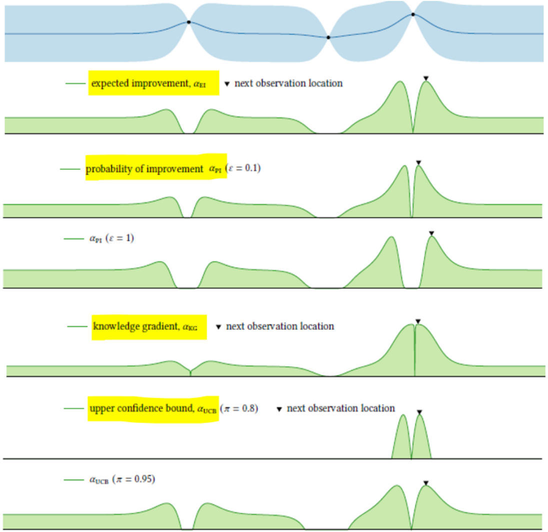
Expected marginal gain to a utility fun.: $$\alpha\ (x\ ;\ \mathcal{D}) = \int [\ u\ (\mathcal{D}') - u\ (\mathcal{D})\ ]\ \mathcal{N}(y; \mu, s^2)\ dy \quad (5)$$

How to determine the computation methods for acq. fun.?

- If the integral is tractable: can compute analytically
  can use approx. methods if suitable and computationally cheaper
- If the integral is intractable: have to use analytic approx. or numerical integration
  For GP, a common choice is Gauss-Hermite quadrature

Exact computation possible

- Expected improvement (EI)
  - $u(\mathcal{D})$: simple reward
- Probability of improvement (PI)
  - $u(\mathcal{D})$: improvement to simple reward
- Knowledge gradient (KG) in **discrete** domains
  - $u(\mathcal{D})$: global reward
- Upper confidence bound (UCB)
  - multi-armed bandits, analogy to PI



Expected Improvement

Expected marginal gain in simple reward:

$$\alpha_{EI}\ (x\ ;\ \mathcal{D}) = \mathbb{E}\left[\max\ \mu_{\mathcal{D}'}(\mathbf{x'}) | x,\ \mathcal{D}\right] - \max \mu_{\mathcal{D}}(\mathbf{x}) \tag{6}$$

This expectation can be computed **analytically** for GPs with exact and noisy observations.

$'$ (prime) is used to indicate post-observation quantities;
In the book, $\phi(\cdot)$ is used to indicate PDF, to avoid confusion with fun. eval. ϕ, here, we use $\varphi(\cdot)$ for PDF

Expected Improvement

Expected marginal gain in simple reward:

$$\alpha_{EI}\ (x\ ;\ \mathcal{D}) = \mathbb{E}\left[\max\ \mu_{\mathcal{D}'}(\mathbf{x'}) | x,\ \mathcal{D}\right] - \max \mu_{\mathcal{D}}(\mathbf{x}) \tag{6}$$

This expectation can be computed **analytically** for GPs with exact and noisy observations.

Noiseless case (Exact)

$$\alpha_{EI}\ (x\ ;\ \mathcal{D}) = (\mu - \phi^*)\ \underbrace{\Phi\left(\frac{\mu - \phi^*}{\sigma}\right)}_{\text{standard normal CDF}} + \sigma\ \underbrace{\varphi\left(\frac{\mu - \phi^*}{\sigma}\right)}_{\text{standard normal PDF}} \tag{7}$$

- First term: encourage exploitation, favor points with high expected value μ
- Second term: encourage exploration, favor points with high uncertainty σ
- Exploitation-exploration tradeoff is considered automatically

$'$ (prime) is used to indicate post-observation quantities;
In the book, $\phi(\cdot)$ is used to indicate PDF, to avoid confusion with fun. eval. ϕ, here, we use $\varphi(\cdot)$ for PDF

Expected Improvement

The University of Manchester

Noisy observations (Farzier *et al.* 2009)

$$\alpha_{EI}\ (x\ ;\ \mathcal{D}) = g(\mathbf{a}, \mathbf{b}) - \mu^* \tag{8}$$

- $g(\mathbf{a}, \mathbf{b}) = \int \max(\mathbf{a} + \mathbf{b}z)\ \varphi(z)\ dz$, z is a standard normal random variable
- $\mu^* = \max\ \mu_{\mathcal{D}}(\mathbf{x})$: simple reward of the current data
- $\mu_{\mathcal{D}'}(\mathbf{x'}) = \mathbf{a} + \mathbf{b}z$: posterior mean evaluated at $\mathbf{x'}$, linear
  obtained via linear transformation $y = \mu + sz$
- $\mathbf{a} = \mu_{\mathcal{D}}(\mathbf{x'})$, $\mathbf{b} = \frac{K_{\mathcal{D}}(\mathbf{x'}, x)}{s}$

Expected Improvement

The University of Manchester

Noisy observations (Farzier *et al.* 2009)

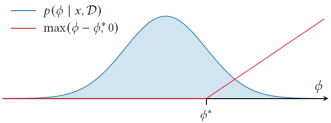
$$\alpha_{EI}\ (x\ ;\ \mathcal{D}) = g(\mathbf{a}, \mathbf{b}) - \mu^* \tag{8}$$

- $g(\mathbf{a}, \mathbf{b}) = \int \max(\mathbf{a} + \mathbf{b}z)\ \varphi(z)\ dz$, z is a standard normal random variable
- $\mu^* = \max\ \mu_{\mathcal{D}}(\mathbf{x})$: simple reward of the current data
- $\mu_{\mathcal{D}'}(\mathbf{x'}) = \mathbf{a} + \mathbf{b}z$: posterior mean evaluated at $\mathbf{x'}$, linear obtained via linear transformation $y = \mu + sz$
- $\mathbf{a} = \mu_{\mathcal{D}}(\mathbf{x'})$, $\mathbf{b} = \frac{K_{\mathcal{D}}(\mathbf{x'}, x)}{s}$

Updated simple reward can occur at

- Noisy observations: *any* previously visited points, including suboptimal ones, due to inherent uncertainty in the obj. fun.
- Exact observations: only possible at the newly observed point or the incumbent

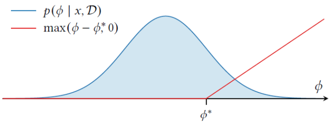
Expected Improvement: exact vs. noisy observations

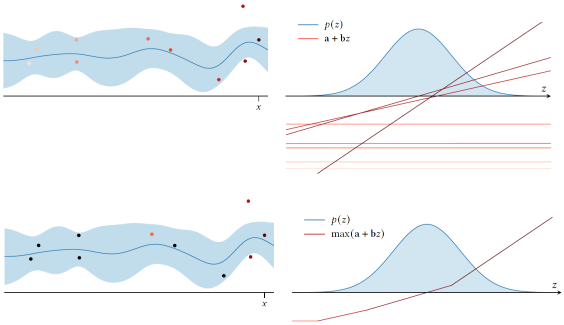


Exact observations

- Only needs to consider the incumbent for the improvement
- Reminder: candidate x is known when compute EI

Expected Improvement: exact vs. noisy observations





Exact observations

- Only needs to consider the incumbent for the improvement
- Reminder: candidate x is known when compute EI

Noisy observations

- Need to consider every visited point for the improvement
- Lead to an upper envelope
- The formation of the upper envelope is invariant to the order of the lines, and the deletion of non-dominate lines won't affect the results $\rightarrow$ effective preprocessing step proposed by Frazier *et al*, 2009.

Expected Improvement: alternative approximation formulations

Frazier *et al*, 2009:	Find analytical expression of EI for noisy case
Alternative:	Approximate EI for noisy case, used extensively in practice

Expected Improvement: alternative approximation formulations

Frazier *et al*, 2009: Find analytical expression of EI for noisy case
Alternative: Approximate EI for noisy case, used extensively in practice

- **Main idea**: Transfer to a noiseless (exact) case, *e.g.*,
  - Plug-in estimate:
    - Use the exact formulation (7), plug-in estimate for incumbent value ϕ^*, *e.g.*, $\phi^* \approx \max \boldsymbol{y}$
  - Re-interpolation:
    - Fit a noiseless GP to imputed values of the obj. fun. at the observed location $\boldsymbol{\phi} = f(\boldsymbol{x})$, *e.g.*, $\boldsymbol{\phi} \approx \mu_{\mathcal{D}}(\boldsymbol{x})$
- Follow the same procedures as for the exact case

Expected Improvement: alternative approximation formulations

Frazier *et al*, 2009: Find analytical expression of EI for noisy case
Alternative: Approximate EI for noisy case, used extensively in practice

- **Main idea**: Transfer to a noiseless (exact) case, *e.g.*,
  - Plug-in estimate:
    - Use the exact formulation (7), plug-in estimate for incumbent value ϕ^*, *e.g.*, $\phi^* \approx \max \boldsymbol{y}$
  - Re-interpolation:
    - Fit a noiseless GP to imputed values of the obj. fun. at the observed location $\boldsymbol{\phi} = f(\boldsymbol{x})$, *e.g.*, $\boldsymbol{\phi} \approx \mu_{\mathcal{D}}(\boldsymbol{x})$
- Follow the same procedures as for the exact case
- **Assumption**: the underlying noiseless EI assumes that our observation will reveal the exact obj. value (ignore entirely the obs. noise), b/c approx. is w.r.t *unobservable* ϕ rather than observed y

Plug-in estimate: two common ones

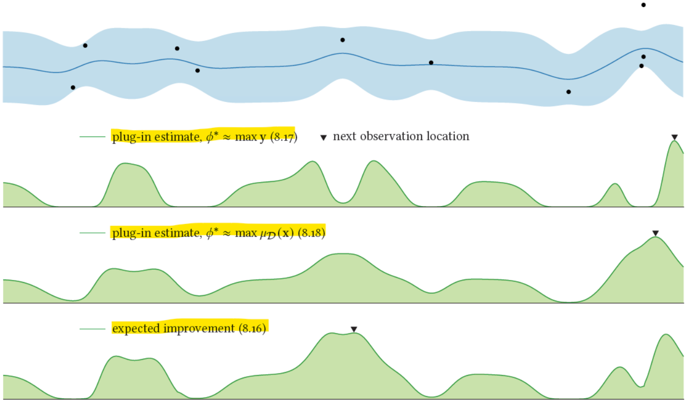
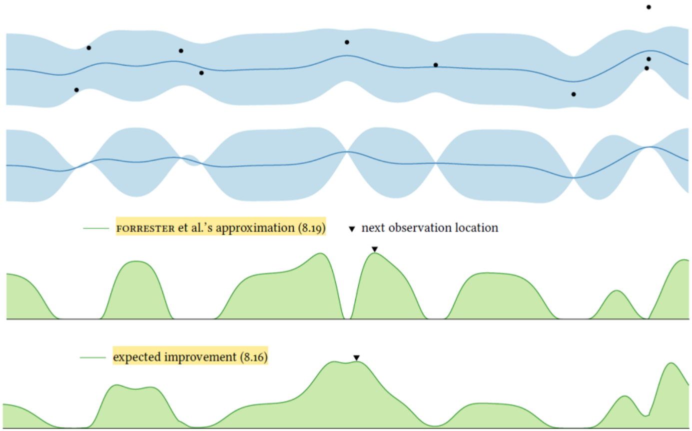


Figure 8.4: Expected improvement using different plug-in estimators (8.17–8.18) compared with the noisy expected improvement as the expected marginal gain in simple reward (8.7).

- Plug-in estimate for incumbent value ϕ^* –> (8.17), (8.18)
- Used empirically and showed various performance (case-dependent)

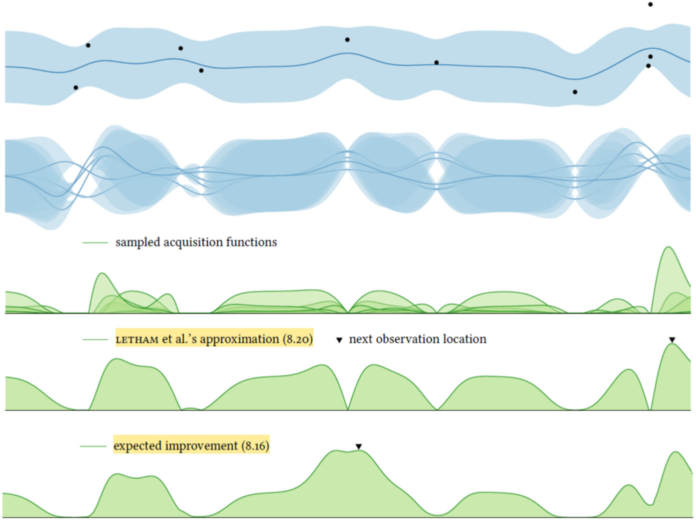
- Noisy observations
- (8.17) max. noisy obs.: for very noisy data –> excessively exploratory
- (8.18) simple reward of the data: less biased
- Why recommended next observation location is so different?
  - (8.17) and (8.18) only consider marginals
  - (8.16) consider the *joint* predictive distribution of **y'**

Re-interpolation: Forrester et al, 2006



- Noisy observations
- Re-interpolated noiseless GP using posterior mean $\boldsymbol{\phi} \approx \mu_{\mathcal{D}}(\boldsymbol{x})$
  acq. fun. approx.:
  $\alpha_{EI}(x; \mathcal{D}) \approx \alpha_{EI}(x; \boldsymbol{x}, \boldsymbol{\phi})$
- (8.19): resulting decision is very similar to (8.16) this time

Re-interpolation: Letham et al, 2019



- Noisy observations
- Re-interpolated by marginalizing rather than imputing the latent obj. fun. val.:
  $\alpha_{EI}(x; \mathcal{D}) \approx \int \alpha_{EI}(x; \boldsymbol{x}, \boldsymbol{\phi}) p(\boldsymbol{\phi}|\boldsymbol{x}, \mathcal{D}) d\boldsymbol{\phi}$
  - Integral cannot be computed exactly, use a quasi-Monte Carlo approx.
  - Take exp. of the exact EI for a noiseless GP fit to exact observations at the obs. loc.
- (8.20): the general shape of the approx. acq. fun. agrees with the (8.16), except near the chosen loc. of (8.16)
  - Inherent property of re-interpolation: acq. fun. vanishes at previously obs. loc. (property of exact EI, assumption of approx. methods)

Take away: EI

- EI acq. fun. (exact/noisy) can be analytically computed
- For noisy cases, different approx. formulations exists (plug-in, re-interpolation)

Take away: EI

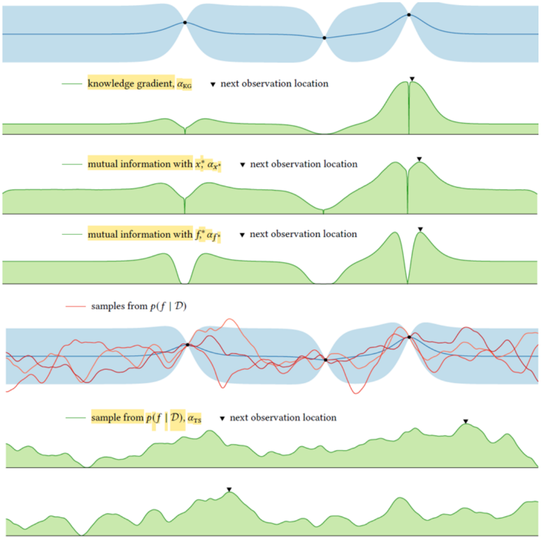
- EI acq. fun. (exact/noisy) can be analytically computed
- For noisy cases, different approx. formulations exists (plug-in, re-interpolation)
- Debate about the necessity of repeated measurements at the same location for noisy cases
  - Case-dependent
  - Necessary: reduce uncertainty
  - Not necessary: if desired, measurements in neighboring locations can be sampled
  - Remedy: *e.g.*, augmented EI to account for obs. noise (penalize loc. with low S/N)

Take away: EI

- EI acq. fun. (exact/noisy) can be analytically computed
- For noisy cases, different approx. formulations exists (plug-in, re-interpolation)
- Debate about the necessity of repeated measurements at the same location for noisy cases
  - Case-dependent
  - Necessary: reduce uncertainty
  - Not necessary: if desired, measurements in neighboring locations can be sampled
  - Remedy: *e.g.*, augmented EI to account for obs. noise (penalize loc. with low S/N)
- General practice:
  - Known low S/N (esp. with heteroskedatic noise): avoid using approx. scheme with exact EI
  - Otherwise, reasonable, b/c $y \approx \phi$, $s \approx \sigma$; computationally cheaper

Effective approximation schemes available

- Knowledge gradient in **continuous** domains
  - *e.g.*, KGCP (Scott *et al.*, 2011)
- Mutual information
  - w.r.t x^* or f^*
- Thompson sampling (TS)
  - exhaustive sampling
  - on-demand sampling



Thompson sampling

Recall:

$$\text{TS sample from the opt. belief}: \quad x \sim p\left(x^* \mid \mathcal{D}\right) \tag{9}$$

Thompson sampling

Recall:

$$\text{TS sample from the opt. belief :} \quad x \sim p\left(x^* \mid \mathcal{D}\right) \tag{9}$$

Computation method

- Reminder: We only consider GP in this chapter, all the discussion is model-dependent
- Compute analytically: special case with specific dist, *e.g.*, Wiener process (rare, can be exploited)
- Approximate: most of the time, b/c the opt. belief dist (9) is complicated

Thompson sampling

Example: most of the time $p\left(x^* \mid \mathcal{D}\right)$ can only be revealed via brute-force sample

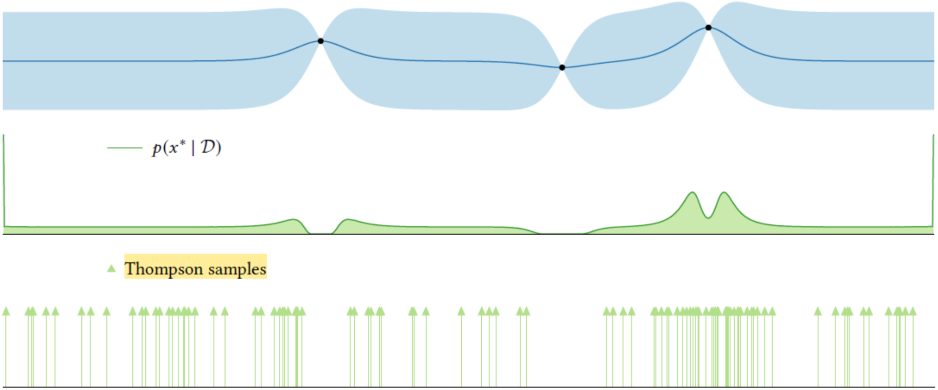


Figure 8.11: The distribution of the location of the global maximum, $p(x^* \mid \mathcal{D})$, for an example scenario, and 100 samples drawn from this distribution.

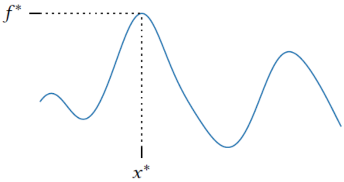
Thompson sampling

<u>Two-stage implementation</u>: max a draw (acq. fun.) from the *obj. fun.* posterior

1) Sample a rand. realization of the obj. fun. from its posterior : $\alpha_{TS}\,(x\,;\,\mathcal{D}) \sim p\,(f \mid \mathcal{D})$ (10)
2) Opt. to yield the desired sample : $x \in \arg\max\ \alpha_{TS}\,(x\,;\,\mathcal{D})$ (11)

<u>Note</u>: the global optimum of α_{TS} is a sample from

- Desired dist. $p\,(x^* \mid \mathcal{D})$, the opt. belief, and
- *Joint* dist. of the loc. and val. of the optimum, $p\,(x^*, f^* \mid \mathcal{D})$, because $f(x)$ is a sample from $p\,(f^* \mid x^*, \mathcal{D})$



Maximizing a draw from a Gaussian process naturally samples from $p(x^*, f^* \mid \mathcal{D})$

TS: approximate α_{TS} with exhaustive sampling

Suitable if the domain can be covered by a *sufficiently* small set of points $\boldsymbol{\xi}$, where $\boldsymbol{\phi_p} = f_p(\boldsymbol{\xi})$ whose dist. is *multivariate normal dist.* (easy to sample, note, *Not* taking the mean)

$$x = \arg\max\ \boldsymbol{\phi_p}; \qquad \boldsymbol{\phi_p} \sim p(\boldsymbol{\phi_p} \mid \boldsymbol{\xi},\ \mathcal{D})$$
$$p(\boldsymbol{\phi_p} \mid \boldsymbol{\xi},\ \mathcal{D}) = \mathcal{N}\ (\boldsymbol{\phi_p}\ ;\ \boldsymbol{\mu_p},\ \Sigma); \qquad \boldsymbol{\mu_p} = \mu_{\mathcal{D}}(\boldsymbol{\xi}); \qquad \Sigma = K_{\mathcal{D}}(\boldsymbol{\xi},\boldsymbol{\xi})$$

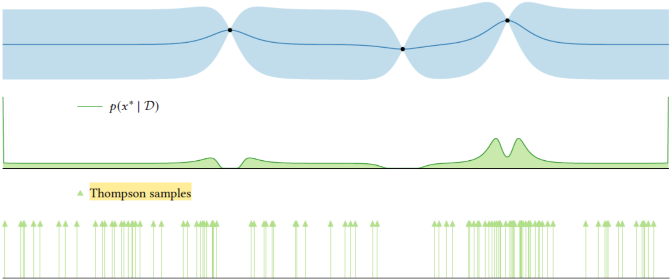
Subscript p is used (*e.g.*, ϕ_p and $f_p(\cdot)$) to denote the entities relevant/calculated based on the posterior of the obj. fun. to avoid confusion with the notations used in the previous chapters, where $f(\cdot)$ is used to refer to the true latent fun..

TS: approximate α_{TS} with exhaustive sampling

Suitable if the domain can be covered by a *sufficiently* small set of points $\boldsymbol{\xi}$, where $\boldsymbol{\phi_p} = f_p(\boldsymbol{\xi})$ whose dist. is *multivariate normal dist.* (easy to sample, note, *Not* taking the mean)

$$x = \arg\max \ \boldsymbol{\phi_p}; \qquad \boldsymbol{\phi_p} \sim p(\boldsymbol{\phi_p} \mid \boldsymbol{\xi}, \ \mathcal{D})$$

$$p(\boldsymbol{\phi_p} \mid \boldsymbol{\xi}, \ \mathcal{D}) = \mathcal{N}\left(\boldsymbol{\phi_p} \ ; \ \boldsymbol{\mu_p}, \ \Sigma\right); \qquad \boldsymbol{\mu_p} = \mu_{\mathcal{D}}(\boldsymbol{\xi}); \qquad \Sigma = K_{\mathcal{D}}(\boldsymbol{\xi}, \boldsymbol{\xi})$$



- Posterior of the obj. fun.
- Current optimal belief conditioned on $\mathcal{D}$ estimated from the approx. of α_{TS}
- based on 100 TS samples
  - achieved with $\boldsymbol{\xi}$: grid of 1000 pts

Note: an iterative solver is used (opt. within opt., and, to propose the next point for evaluation, the solver iteratively suggest candidate x, and select the "best" candidate, *i.e.*, the one maximize $\boldsymbol{\phi_p}$ as the proposed next point to observe

TS: approximate α_{TS} with on-demand sampling

Utilize the opt. routines of the iterative solver when opt. the acq. fun. to **max a draw** from the ***sudo*** obj. fun. posterior built progressively on demand

- augment our dataset $\mathcal{D}$ with the ***simulated*** observation $(x, \phi) \rightarrow \mathcal{D}_{TS}$
- sample $\phi \sim p(\phi \mid x \ , \ \mathcal{D}_{TS})$

$\mathcal{D}_{\text{TS}} \leftarrow \mathcal{D}$ ▸ initialize fictitious dataset with current data
repeat
  given request for observation at x:
    $\phi \leftarrow p(\phi \mid x, \mathcal{D}_{\text{TS}})$ ▸ sample value at x
    $\mathcal{D}_{\text{TS}} \leftarrow \mathcal{D}_{\text{TS}} \cup (x, \phi)$ ▸ update fictitious dataset
    yield ϕ
until external optimizer terminates

TS: approximate α_{TS} with on-demand sampling

MANCHESTER 1824 The University of Manchester

Utilize the opt. routines of the iterative solver when opt. the acq. fun. to **max a draw** from the ***sudo*** obj. fun. posterior built progressively on demand

- augment our dataset $\mathcal{D}$ with the ***simulated*** observation $(x, \phi) \to \mathcal{D}_{TS}$
- sample $\phi \sim p(\phi \mid x \ , \ \mathcal{D}_{TS})$

$\mathcal{D}_{\text{TS}} \leftarrow \mathcal{D}$ ▸ initialize fictitious dataset with current data
repeat
    given request for observation at x:
    $\phi \leftarrow p(\phi \mid x, \mathcal{D}_{\text{TS}})$ ▸ sample value at x
    $\mathcal{D}_{\text{TS}} \leftarrow \mathcal{D}_{\text{TS}} \cup (x, \phi)$ ▸ update fictitious dataset
    yield ϕ
until external optimizer terminates

Note: for *stationary* covariance fun., *i.e.*, $K(\mathbf{x}, \mathbf{x'}) = K(\mathbf{x} - \mathbf{x'})$, using *sparse spectrum approx.* to estimate the posterior GP can dramatically accelerate the optimization of acq. fun. with TS

Take away: TS

- Efficient approximation schemes available for TS under different assumptions
  - When the domain can be covered by a small set: exhaustive sampling
  - When not possible: on-demand sampling
    - Accelerating method (if stationary): use sparse spectrum approx, to estimate the GP posterior
- Not covered in this presentation, but in the book
  - TS is often integrated within the efficient approx. scheme for mutual information policies
- Other notes: nowadays, TS is getting more attention since it is easily parallelizable –> computational efficient for batch BO, parallel BO, ... for distributed/multi-agent learning

Summary

The University of Manchester

- Discussed computation methods (analytically/approximately), focusing on
  - Expected improvement (can be calculated both analytically and approximately)
  - Thompson sampling (most of the time only possible via approximation, special case exists)
- Important : whether the policy can be computed analytically or approximately depends on the obj. fun. and noise models selected
  - Discussion in this chapter: *GP models* as the obj. fun. model s.t *exact or additive Gaussian noise*
  - Depending on the model and the problem, one may prefer one policy to the other
- If analytical expression available (comparing to numerically approx. ones), policy may be optimized more efficiently using gradient-based methods
- Chapter 9: more implementation details

Summary

- Discussed computation methods (analytically/approximately), focusing on
  - Expected improvement (can be calculated both analytically and approximately)
  - Thompson sampling (most of the time only possible via approximation, special case exists)
- Important : whether the policy can be computed analytically or approximately depends on the obj. fun. and noise models selected
  - Discussion in this chapter: *GP models* as the obj. fun. model s.t *exact or additive Gaussian noise*
  - Depending on the model and the problem, one may prefer one policy to the other
- If analytical expression available (comparing to numerically approx. ones), policy may be optimized more efficiently using gradient-based methods
- Chapter 9: more implementation details
- Thank you 🙂 Questions?