Chapter 5

Experiments

In this chapter, we will apply the *Expected Improvement* (`EI`) as well as the *Gaussian Process Upper Confidence Bound* algorithm (`GP-UCB`) to determine the maxima of different one-dimensional objective functions of the form $f\colon \mathbb{R} \to \mathbb{R}$ on a compact subset $\mathbb{X} \subset \mathbb{R}$ using `MATLAB`®[1] and compare the results. Since we do not have any informations about the function f apart from the training values and previous test values in each iteration, we cannot apply the *Contextual Gaussian Process Upper Confidence Bound* algorithm (`CGP-UCB`).

As a short revision, both algorithms create a model of the function to be optimized using *Gaussian processes*, where f is generally assumed to be extremely costly. The model is then adapted in each iteration and used to determine the approximate optimum of the function. Both algorithms obtain a set of training values as parameters and choose new test points in every iteration. The `EI` algorithm chooses the next test point as

$$x_{t+1} := \arg\max_{x \in \mathbb{X}} \mathbb{E}\left[(\xi(x) - M_t)_+ \mid \mathcal{F}_t\right],$$

where M_t is the biggest function value so far, $\mathcal{F}_t$ denotes the σ-algebra generated by all previous training values and $\xi(x)$ is the Gaussian process evaluated at x. The `GP-UCB` algorithm chooses

$$x_{t+1} := \arg\max_{x \in \mathbb{X}} \left(\widehat{\xi}_t(x) + \beta_t^{1/2} \sigma_t(x)\right),$$

where $\widehat{\xi}_t(x)$ denotes the expected value of the Gaussian process at x given all previous results and $\sigma_t^2(x)$ its variance. $\beta_t^{1/2}$ can be considered as a weight which determines how important it is for the algorithm to explore new areas of the function compared to the importance of exploring areas where the maximum is assumed to be.

For both algorithms, we will use the squared exponential kernel function (cf. Section 2.4) where its parameters will be estimated using the Bayesian model described in Section 2.6 including empirical Bayes. Since the function evaluations are exact, we assume the noise ε_t to be zero for all t. The scripts support noisy optimization problems as well.

[1] `MATLAB`® Version 2014a (8.3.0.532), The MathWorks® Inc., 2014. To run the scripts, both the *Optimization Toolbox* and the *Statistics Toolbox* have to be installed.

Legend for `EI` plots: In all of the plots displayed in this chapter, we will use the following legend:

i) ***For* `EI` *plots:*** In every plot, the objective function is drawn as a *blue line*. The *red line* shows the model of the objective function generated using Gaussian processes, which is updated in every step. *Red crosses* indicate the positions of training- and previous test values. For visibility, only in the bigger plots, the next test value is indicated by an *orange cross*. The *green line* shows the expected improvement at every point $x \in \mathbb{X}$, which is rescaled to increase visibility. Additionally, a *black asterisk* indicates the actual global maximum of the objective function. A cyan cross shows $\max\{f(x_i), i \leq n\}$, where $\{x_1, \ldots, x_n\}$ is the set of training and test values, i.e. the current maximum found by the algorithm. Note that a plot of iteration t displays the model and expected improvement used to choose the t^{th} test value.

ii) ***For* `GP-UCB` *plots:*** The only thing that is different from the `EI` plots is that the *green line* displays the UCB value instead of the expected improvement. This value is not rescaled.

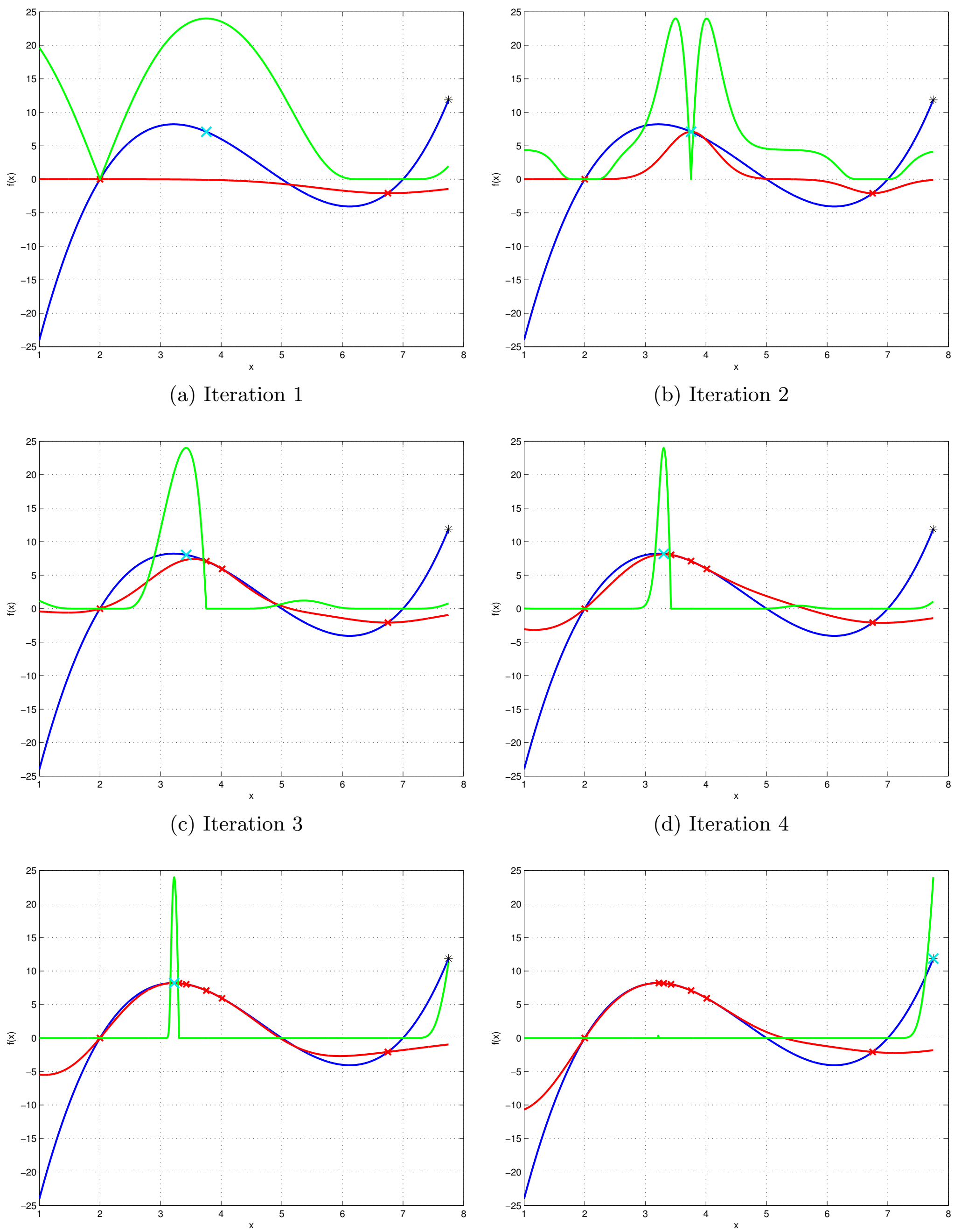
The script applying the `EI` algorithm implements an additional canonical stopping criterion. The idea is to stop the algorithm as soon as $\sup\limits_{x \in \mathbb{X}} \mathtt{EI}(x) \leq \alpha$, where $\alpha > 0$ is a value close to zero, for example $\alpha = 10^{-5}$. Note that this criterion should not be used if there are not enough training values available at the beginning: If the model is similar to a linear or constant model, it is very likely that the algorithm chooses a new training value close to the current maximum M_n. This leads to a small expected improvement since it depends on $\widehat{\xi}_t(x) - M_n$, i.e. the difference between the model prediction and the currently highest result. An example can be seen in Figure 5.3. To prevent this from happening, one can force the algorithm to perform a minimum amount of iterations before checking the stopping criterion.

We will start with two sample applications of both algorithms to a function with two local maxima with detailed plots of every iteration. Subsequently, we will compare the results of both algorithms applied to functions with multiple local maxima and different initial training values. In the simulations, we will mostly use

$$\beta_t = 2\log(t^2 2\pi^2/(3\delta)) + 2\log(t^2 r\sqrt{\log(4/\delta)},$$

where r denotes the length of the maximization interval. This definition of β_t can be found in Theorem 2 of Srinivas et al. [2009] and we will use it even if the assumptions might not be fulfilled.

Figures 5.1 and 5.2 show 6 iterations of the `EI` algorithm and 2 of the `GP-UCB` algorithm, respectively. Both were applied to the objective function $f(x) = (x-2)(x-5)(x-7)$, where $\mathbb{X} = [1, 7.75]$ and the initial training points are given by $\mathbf{x} = \{2, 6.75\}$. Note how the `EI` algorithm first concentrates on the exploration of a local maximum and finds the global maximum as soon as the area around the local maximum is sufficiently explored, since the expected improvement at the boundary increases. After 6 steps of the expected improvement algorithm, the cumulative regret is given by $R_6^{\mathtt{EI}} = 44.01$, where the cumulative regret of the `GP-UCB` algorithm after two steps is given by $R_2^{\mathtt{UCB}} = 25.79$ and $R_6^{\mathtt{UCB}} = 54.45$. Although `GP-UCB` finds the global maximum after 2 steps, this value is way higher than $R_2^{\mathtt{UCB}}$ since the algorithm continues to explore different regions after finding the maximum.



(a) Iteration 1

(b) Iteration 2

(c) Iteration 3

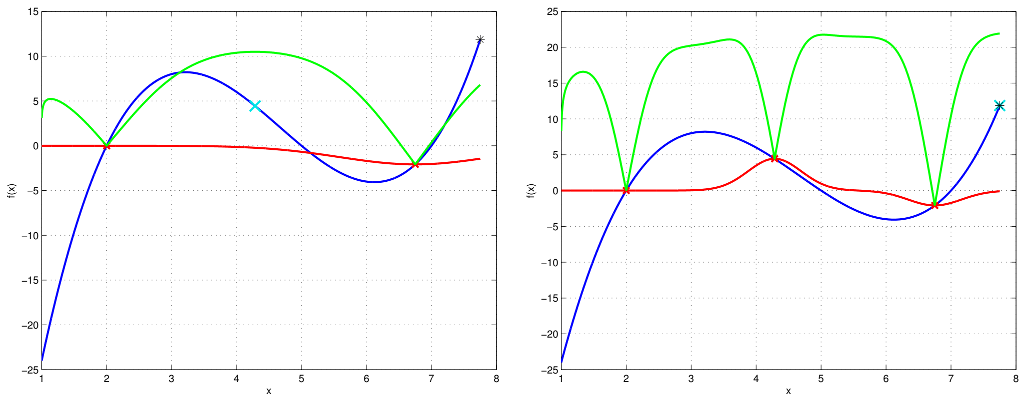
(d) Iteration 4

(e) Iteration 5

(f) Iteration 6

Figure 5.1: Six steps of the EI algorithm applied to $f_1(x) = (x-2)(x-5)(x-7)$, where $\mathbb{X} = [1, 7.75]$ and the initial training points are $\mathbf{x} = \{2, 6.75\}$. A description of the legend can be found in the main text above.

For simplicity, define $f_1(x) := (x-2)(x-5)(x-7)$, the function which we have used in



(a) Iteration 1 (b) Iteration 2

Figure 5.2: The first two steps of the `GP-UCB` algorithm applied to $f_1(x) = (x-2)(x-5)(x-7)$, where $\mathbb{X} = [1, 7.75]$ and the initial training points are $\mathbf{x} = \{2, 6.75\}$. A description of the legend can be found in the main text above.

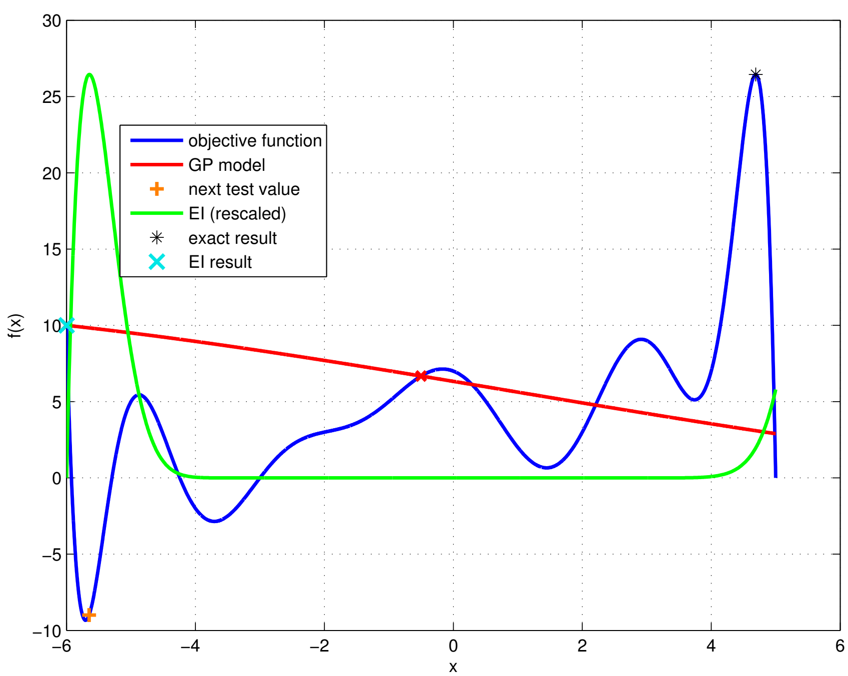


Figure 5.3: An example where the stopping criterion for the `EI` algorithm fails if α is not chosen small enough: The algorithm would terminate after two iterations, because it chooses two values which are very close to each other. The legend is the same one used in the previous figures.

the previous plots. Additionally, consider $f_2(x)$, the polynomial of degree 11 interpolating the points

$$\{(-6,10),(-5,5),(-4,-2),(-3,0),(-2,3),\\(-1,5),(0,7),(1,2),(2,3),(3,9),(4,7),(5,0)\}$$

and $f_3(x) := \frac{x}{5}\sin(x^3)$. Table 5.1 shows the results of a few experiments performed on those functions. Overall, the `GP-UCB` algorithm seems to perform better concerning convergence to the correct result. The main problem of the `EI` algorithm is that it tends to get stuck at a local optimum, i.e. it finds a peak in the objective function and then continuously chooses test values close to this local maximum. A possible way to fix this would be to detect if the algorithm gets stuck and then either choose a random value in $\mathbb{X}$ as new test value or the value

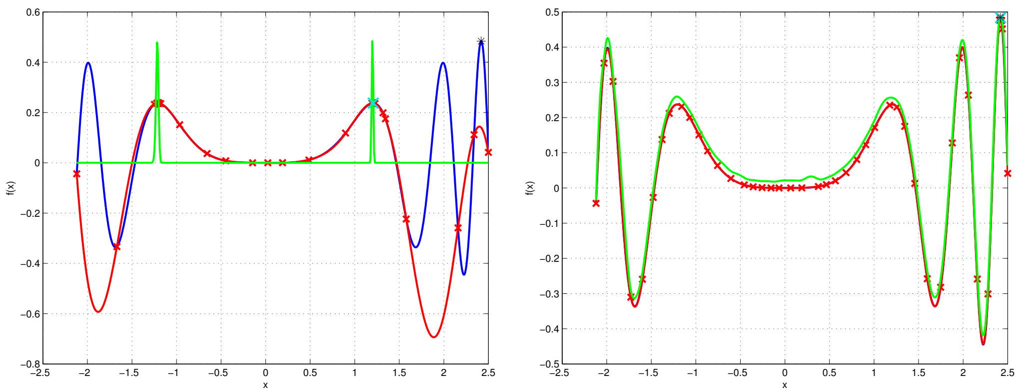
$$x^* = \arg\max_{x\in\mathbb{X}} \left(\min_{i\leq n} \|x - x_i\|_{\mathbb{X}}\right),$$

i.e. the point with the highest distance from all previous training values.

The cumulative regrets of both algorithms are, though, usually close to each other. Suprisingly, in the experiments, the choice of δ has a minor influence on the rate of convergence and cumulative regret. The influence of the choice of β_t, however, increases if it is chosen extremely high or low. Figure 5.4 displays plots of two interesting cases.

Fnct.	Alg.	Area	Tr. Values	Fnd. Max.	C. Regret	δ
f_1	`EI`	$[1, 7.75]$	$\{2, 6.75\}$	6	48.54	/
f_1	`GP-UCB`	$[1, 7.75]$	$\{2, 6.75\}$	2	70.49	0.1
f_1	`GP-UCB`	$[1, 7.75]$	$\{2, 6.75\}$	2	70.48	0.25
f_1	`GP-UCB`	$[1, 7.75]$	$\{2, 6.75\}$	2	102.76	0.75
f_1	`EI`	$[1, 7.5]$	5.5	56 $(*)$	111.26	/
f_1	`GP-UCB`	$[1, 7.5]$	5.5	9	54.66	0.1
f_1	`GP-UCB`	$[1, 7.5]$	5.5	9	53.43	0.25
f_1	`GP-UCB`	$[1, 7.5]$	5.5	9	53.95	0.75
f_2	`EI`	$[-6, 5]$	grid, 3	22	569.17	/
f_2	`GP-UCB`	$[-6, 5]$	grid, 3	40	652.58	0.1
f_2	`GP-UCB`	$[-6, 5]$	grid, 3	40	654.03	0.25
f_2	`GP-UCB`	$[-6, 5]$	grid, 3	40	654.92	0.75
f_3	`EI`	$[-2.1, 2.5]$	grid, 5	> 100 $(*)$	11.20	/
f_3	`GP-UCB`	$[-2.1, 2.5]$	grid, 5	23	12.49	0.1
f_3	`GP-UCB`	$[-2.1, 2.5]$	grid, 5	25	12.23	0.25
f_3	`GP-UCB`	$[-2.1, 2.5]$	grid, 5	25	12.27	0.75

Table 5.1: A table the results of various experiments involving both algorithms. *Tr. Values* are the initial training values used, where *grid, n* means that n equidistant initial training values have been used. *Fnd. Max.* shows the amount of iterations needed for the algorithm to find the correct maximum and *C. Regret* refers to the cumulative regret after 30 iterations. δ refers to the value of δ chosen for β_t, where we use the definition of β_t of Theorem 2 by Srinivas et al. In the experiments marked with $(*)$, the algorithm got stuck at a local maximum.



(a) `EI` results (b) `GP-UCB` results

Figure 5.4: Application of both algorithms to $f_3(x)$ with 100 iterations and 5 equidistant initial training values. The `EI` algorithm gets stuck and does not find the global maximum, while the result of `GP-UCB` is correct. The cumulative regrets are 28.47 (`EI`), resp. 17.44 (`GP-UCB`).

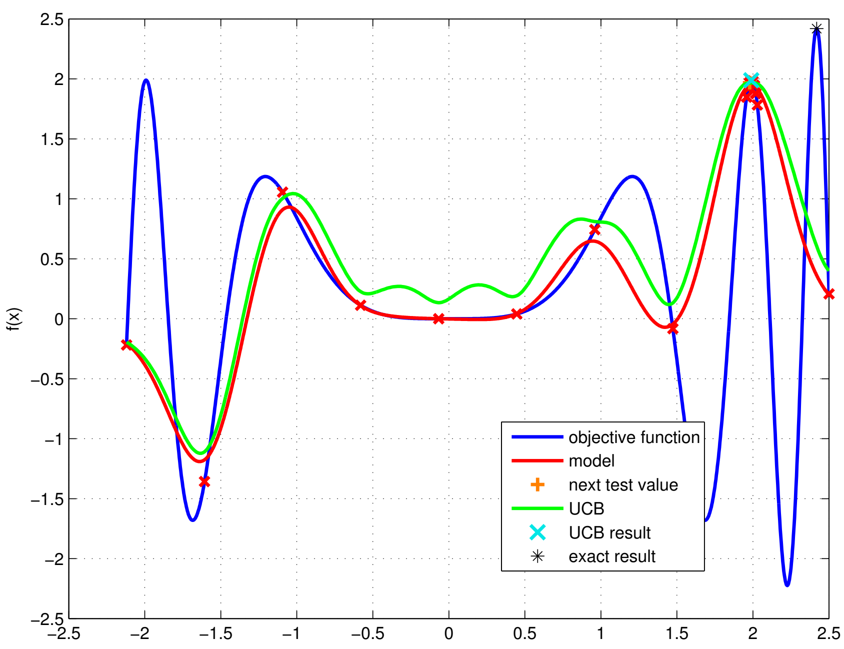
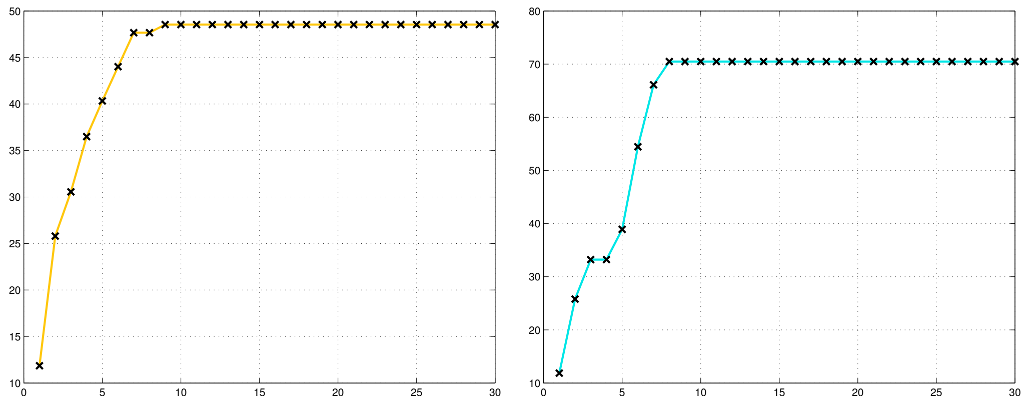


Figure 5.5: An example where the `GP-UCB` algorithm gets stuck due to a bad choice of β_t. The objective function is a rescaled version of f_3, $x\sin(x^3)$, and β_t is chosen to be $t/1000$. Furthermore, we assume the function to be noisy evaluated with $\varepsilon_t \overset{iid}{\sim} \mathcal{N}(0, 0.1)$. One can see that due to the bad choice of β_t, the UCB value at 2 stays very high although the area is already well explored, which is why the algorithm continues to explore this region.



(a) `EI` (b) `GP-UCB`

Figure 5.6: The plots display the evolution of the cumulative regret using `EI` and `GP-UCB`. The algorithms were applied to f_1 with $\mathbb{X} = [1, 7.75]$ and initial training points $\mathbf{x} = \{2, 6.75\}$. As expected, the cumulative regret flattens out since both algorithms evaluate the function near the optimum as soon as it has been explored well enough.

5.1 Conclusions

In the experiments, we have seen that the `GP-UCB` algorithm usually outperforms the `EI` algorithm. The main problem is that the `EI` algorithm tends to get stuck at local optima. The implementation of an additional test, which detects if the algorithm gets stuck, could, however, lead to significant performance improvements.

On the other hand, neither the paper by Srinivas et al. [2009], nor the one by Krause and Ong [2011] describes which choice of β_t leads to good results. Although in the experiment choosing β_t similar to its definition in the Theorems lead to good results, there could be real-life applications where the algorithm fails due to a bad choice of β_t. One example can be seen in Figure 5.5. Concerning this, the `EI` algorithm is more advanced as it does not need any additional constants, which cannot be chosen by the algorithm itself. Additionally, contrary to the theoretical results for the `EI` algorithm, both the paper by Srinivas et al. [2009] and Krause and Ong [2011] did not include convergence results for arbitrary continuous objective functions.

5.2 Simulation Files

The simulation files can be downloaded from `https://github.com/cglanzer/bayesian-global-optimization`. Within this repository, you can also find a detailed explanation on how to run the algorithms. All the scripts are well-commented and each function has a header explaining all of its arguments. Note that in order to run the scripts, both the *Optimization Toolbox* and the *Statistics Toolbox* have to be installed.

Bibliography

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory 2nd Edition (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, 2006. ISBN 0471241954.

Marco Cuturi. Positive definite kernels in machine learning. November 2009. URL http://arxiv.org/pdf/0911.5367.pdf.

Marcus Frean and Phillip Boyle. Using Gaussian processes to optimize expensive functions. *Lecture Notes in Computer Science*, pages 258–267, 2008. ISSN 1611-3349. doi: 10.1007/978-3-540-89378-3_25. URL http://dx.doi.org/10.1007/978-3-540-89378-3_25.

Andreas Krause and Cheng Soon Ong. Contextual gaussian process bandit optim. *Neural Information Processing Systems (NIPS)*, pages 2447–2455, 2011. URL http://las.ethz.ch/files/krause11contextual-long.pdf. Long Version.

Vern I. Paulsen. An introduction to the theory of reproducing kernel hilbert spaces. 2013. URL http://www.iam.uni-bonn.de/~maltsev/course_files/paulsen.pdf. (Retrieved: 16/06/2014).

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning series)*. The MIT Press, 2005. ISBN 026218253X.

Matthias Schonlau and William J. Welch. Global optimization with nonparametric function fitting. *Proceedings of the Section on Physical and Engineering Sciences, American Statistical Association*, pages 183–186, 1996. URL http://www.schonlau.net/publication/asa97.pdf.

Niranjan Srinivas, Andreas Krause, Sham M. Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. December 2009. URL http://dx.doi.org/10.1109/TIT.2011.2182033.

Alain-Sol Sznitman and Pierre Nolin. *Probability Theory, Lecture Notes*. 2013. URL http://www.math.ethz.ch/education/bachelor/lectures/hs2013/math/wkeitsth.

Aimo Törn and Antanas Zilinskas. *Global Optimization (Lecture Notes in Computer Science)*. Springer, 1989. ISBN 3540508716.

Emmanuel Vazquez and Julien Bect. Convergence properties of the expected improvement algorithm with fixed mean and covariance functions. *Journal of Statistical Planning and Inference*, 140(11):3088–3095, Nov 2010. ISSN 0378-3758. doi: 10.1016/j.jspi.2010.04.018. URL http://dx.doi.org/10.1016/j.jspi.2010.04.018.

Joannès Vermorel and Mehryar Mohri. Multi-armed bandit algorithms and empirical evaluation. In *European Conference on Machine Learning*, pages 437–448. Springer, 2005.

Ruye Wang. Computer image processing and analysis (lecture notes). http://fourier.eng.hmc.edu/e161/. (Retrieved: 16/04/2014).

Dirk Werner. *Funktionalanalysis (Springer-Lehrbuch) (German Edition)*. Springer, 2011. ISBN 3642210163.

Fuzhen Zhang. *The Schur Complement and Its Applications (Numerical Methods and Algorithms)*. Springer, 2005. ISBN 0387242716.