Parallel Gaussian Bandits

Johannes Verwijnen

(jverwijn@cs.helsinki.fi)

16.4.2014

Bandits

- A multi-armed bandit captures the tradeoff between exploration and exploitation
- How do we maximize the total expected reward?
- Used for dynamic effort allocation

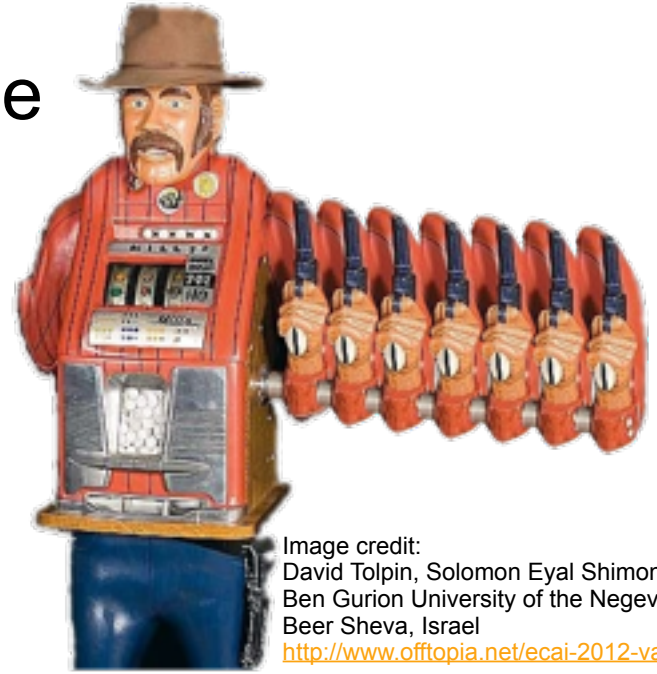


Image credit:
David Tolpin, Solomon Eyal Shimony
Ben Gurion University of the Negev,
Beer Sheva, Israel
http://www.offtopia.net/ecai-2012-vago-poster/

Different types of bandits

- Deterministic bandits
- Stochastic bandits
- Restless bandits
- Adversial bandits
- Arm-acquiring
- Switching penalty
- Correlated bandits

Image credit:
Buffalo Games, Jorge Baeza
http://calhoonplay.com/?q=node/18

Independent bandits

- For results in solving the basic case of independent bandits, one should look at Gittins indices [1] and Lai and Robbins [2].
- Auer et al [3] introduced the simple and efficient UCB algorithms
- Regret bounds are the usual way to evaluate the policies:

$$R_T = \sum_t f\left(\mathbf{x}^*\right) - f\left(\mathbf{x}_t\right)$$

Image credit: Time Bandits (1981), HandMade Films
http://calhoonplay.com/?q=node/18

Dependent arm bandits

- Arms are not always independent
  - Amazon buy-next display or other internet advertising
  - Finding the sensors to activate for local measurement
- How do we model this prior knowledge?

Image credit: Unknown
http://karlalant.com/2013/01/26/individuality-interdependence-and-motivation/

Gaussian Processes

- A generalization of the Gaussian probability distribution
- A collection of independent, normally distributed random variables
- Any finite subset of this collection has a multivariate normal distribution.
- If mean is zero, defined only by covariance kernel.

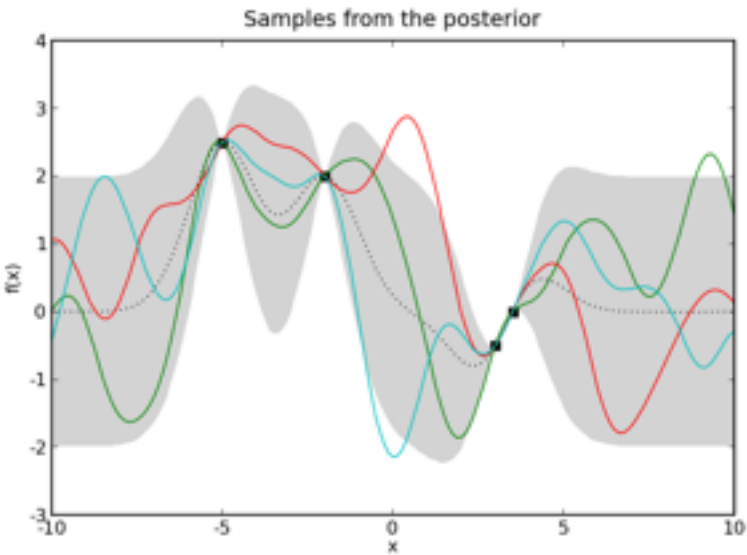


Image credit:
John Reid
http://sysbio.mrc-bsu.cam.ac.uk/johns/infpy/docs/build/html/gps.html

Why Gaussian Processes?

- Simple!
- Noisy sample of rewards $\mathbf{y}_{1:t-1} = [y, \ldots, y_{t-1}]$
  at points $X = \{x_1, \ldots, x_{t-1}\}$
  with Gaussian noise $\epsilon_t \sim N\left(0, \sigma^2\right)$

  the posterior over f is a Gaussian

$$f\left(\mathbf{x}\right) \mid \mathbf{y}_{1:t-1}\right) \sim N\left(\mu_{t-1}\left(\mathbf{x}\right), \sigma_{t-1}^2\left(\mathbf{x}\right)\right)$$

  where

$$\mu_{t-1}\left(\mathbf{x}\right) = \mathbf{k}\left[\mathbf{K} + \sigma^2\mathbf{I}\right]^{-1}\mathbf{y}_{1:t-1}$$
$$\sigma_{t-1}^2\left(\mathbf{x}\right) = k\left(\mathbf{x}, \mathbf{x}\right) - \mathbf{k}\left[\mathbf{K} + \sigma^2\mathbf{I}\right]^{-1}\mathbf{k}^T$$

GP-UCB

- The GP-UCB selection rule [4]

$$\mathbf{x}_t = \arg\max_{\mathbf{x}\in\mathcal{X}} \left(\mu_{t-1}(\mathbf{x}) + \beta_t^{\frac{1}{2}}\sigma_{t-1}(\mathbf{x})\right)$$

- consists of the exploitation part (left) and the exploration part (right)
- Beta is the balancing factor and its optimum shown to depend on the kernel function, t and |X|

GP-UCB algorithm

- The algorithm itself is quite simple:

for t = 1,2,… do

Choose $\mathbf{x}_t = \arg\max_{\mathbf{x}\in\mathcal{X}} \left(\mu_{t-1}(\mathbf{x}) + \beta_t^{\frac{1}{2}}\sigma_{t-1}(\mathbf{x})\right)$

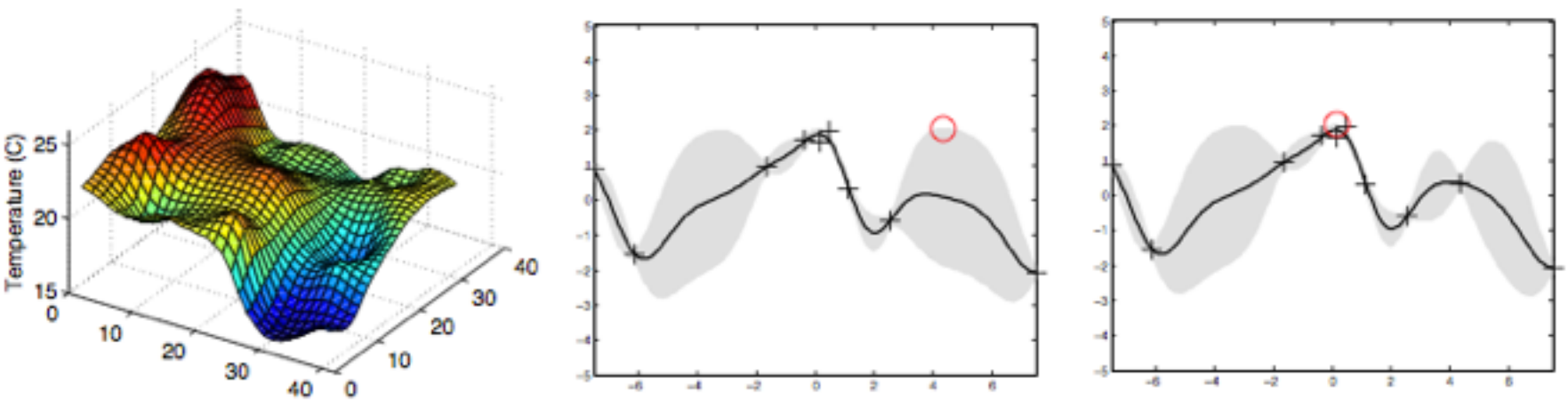
Sample $y_t = f\left(\mathbf{x}_t\right) + \epsilon_t$

Perform Bayesian update to obtain new mu and sigma

end for

GP-UCB results

Kernel	Linear	RBF	Matérn
Regret R_T	$d\sqrt{T}$	$\sqrt{T(\log T)^{d+1}}$	$T^{\frac{\nu+d(d+1)}{2\nu+d(d+1)}}$



(a) *Temperature data* (b) *Iteration t* (c) *Iteration t + 1*

Figure 2. (a) Example of temperature data collected by a network of 46 sensors at Intel Research Berkeley. (b,c) Two iterations of the GP-UCB algorithm. It samples points that are either uncertain (b) or have high posterior mean (c).

Parallel

- Finding the reward may take time
- Let's sample many decisions
- How do we choose the batch?

Image credit:
Argonne National Laboratory
http://en.wikipedia.org/wiki/File:IBM_Blue_Gene_P_supercomputer.jpg

Batch selection

- We only have feedback on the previous decisions. $\mathrm{fb}[t] \leq t-1$
- We need to find a batch of size B

- GP-BUCB uses GP-UCB on limited feedback information [5]
  - also introduces trick for variance calculation
- GP-UCB-PE is a more explorative version [6]
  - introduces a relevant region for explicit exploration

GP-BUCB algorithm

- The algorithm itself is quite simple:

for t = 1,2,… do

Choose $\mathbf{x}_t = \arg\max_{\mathbf{x}\in\mathcal{X}} \left(\mu_{t-1}(\mathbf{x}) + \beta_t^{\frac{1}{2}}\sigma_{t-1}(\mathbf{x})\right)$

Compute sigma

if t = fb(t+1) then

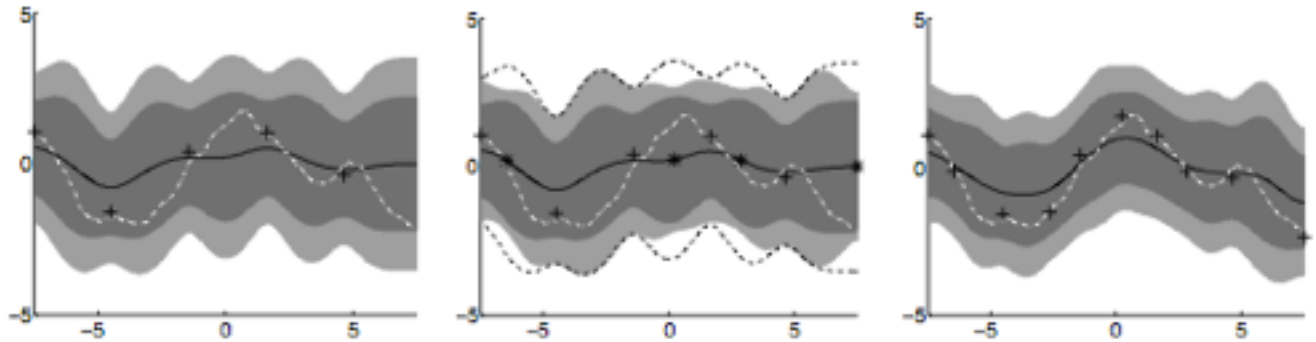
Obtain $y_{t'} = f\left(\mathbf{x}_{t'}\right) + \epsilon_{t'}$ $\quad t' \in \{\mathrm{fb}[t], \ldots, t\}$

Perform Bayesian update to obtain new mu

- end if

end for

GP-BUCB hallucination effect

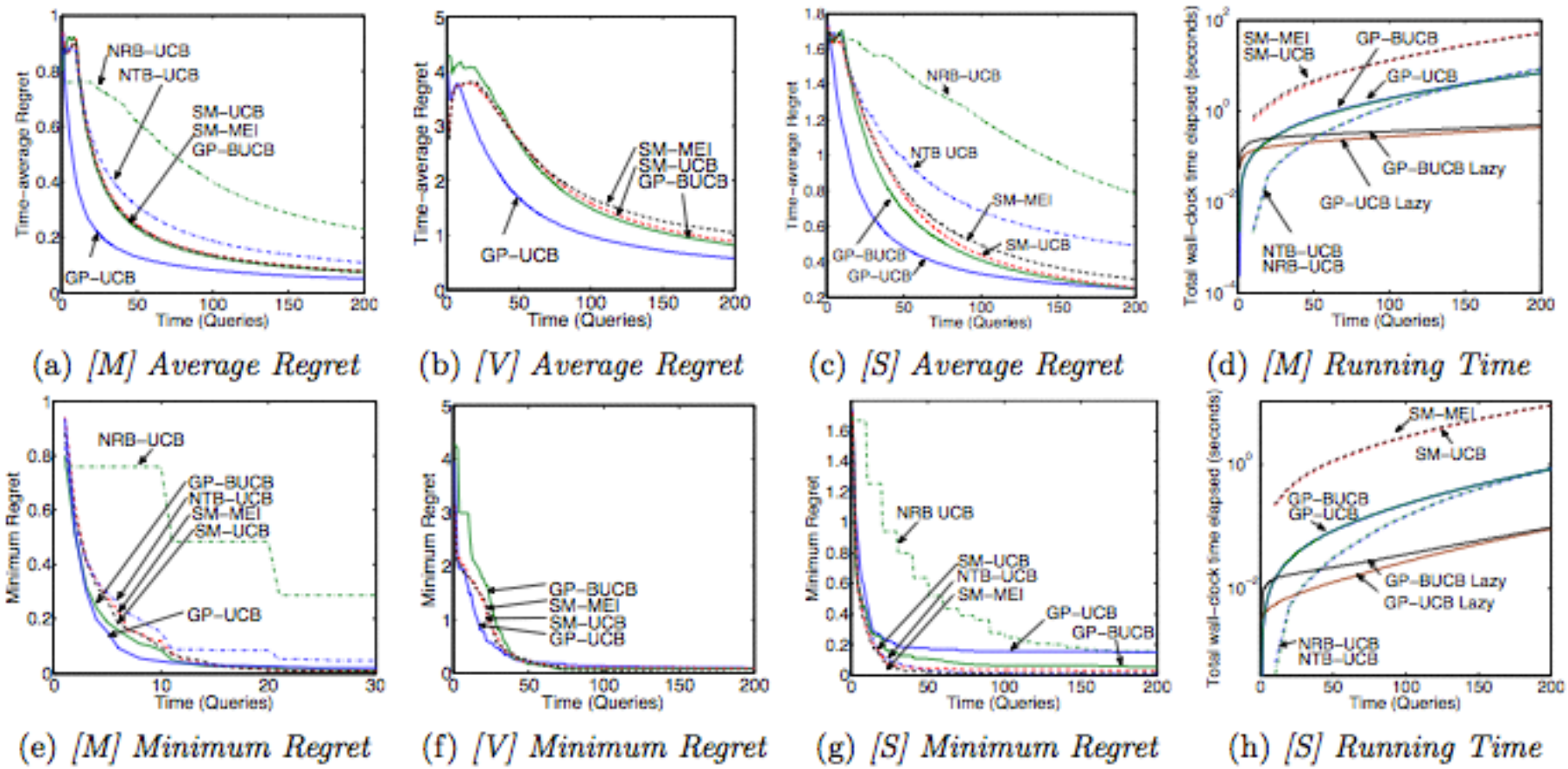


(a) *Beginning of batch 2* (b) *Last decision of batch 2* (c) *Beginning of batch 3*

Figure 1. (a): The confidence intervals $C^{\text{seq}}_{\text{fb}[t]}(\boldsymbol{x})$ (dark), computed from previous noisy observations (crosses), are centered around the posterior mean (solid black) and contain $f(\boldsymbol{x})$ (white dashed) w.h.p. To avoid overconfidence, GP-BUCB chooses $C^{\text{batch}}_{\text{fb}[t]}(\boldsymbol{x})$ (light gray) such that even in the worst case, $C^{\text{batch}}_{t}(\boldsymbol{x})$ will contain $C^{\text{seq}}_{\text{fb}[t]}(\boldsymbol{x})$. (b): Due to the observations that GP-BUCB "hallucinates" (stars), the outer posterior confidence intervals $C^{\text{batch}}_{t}(\boldsymbol{x})$ shrink from their values at the start of the batch (black dashed), but still contain $C^{\text{seq}}_{\text{fb}[t]}(\boldsymbol{x})$, as desired. (c): Upon selection of the last decision of the batch, the feedback for all decisions is obtained, and new confidence intervals $C^{\text{seq}}_{\text{fb}[t']}(\boldsymbol{x})$ and corresponding $C^{\text{batch}}_{\text{fb}[t']}(\boldsymbol{x})$ are computed.

GP-BUCB results

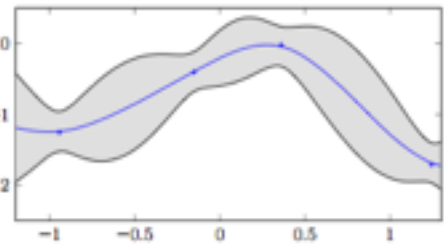
- Regret bounds independent of batch size (see paper for details)



(a) *[M] Average Regret* (b) *[V] Average Regret* (c) *[S] Average Regret* (d) *[M] Running Time*

(e) *[M] Minimum Regret* (f) *[V] Minimum Regret* (g) *[S] Minimum Regret* (h) *[S] Running Time*

GP-UCB-PE



- Here we find a confidence region (from GP-UCB) that f is included with high probability:

$$\hat{f}^{+}_{t-1}(\mathbf{x}) = \left(\mu_{t-1}(\mathbf{x}) + \beta_t^{\frac{1}{2}}\sigma_{t-1}(\mathbf{x})\right)$$
$$\hat{f}^{-}_{t-1}(\mathbf{x}) = \left(\mu_{t-1}(\mathbf{x}) - \beta_t^{\frac{1}{2}}\sigma_{t-1}(\mathbf{x})\right)$$

- Then we define a relevant region, which contains the argmax with high probability with lower bound:

$$y_t^{\bullet} = \hat{f}_t^{-}\left(x_t^{\bullet}\right), \text{ where } x_t^{\bullet} = \arg\max_{\mathbf{x}\in\mathcal{X}} \hat{f}_t^{-}\left(\mathbf{x}\right)$$

giving

$$\mathfrak{R}^{+}_{t-1} = \left\{\mathbf{x} \in \mathcal{X} \mid \mu_{t-1}\left(\mathbf{x}\right) + 2\sqrt{\beta_t}\sigma_{t-1}\left(\mathbf{x}\right) \geq y_{t-1}^{\bullet}\right\}$$

Intuition

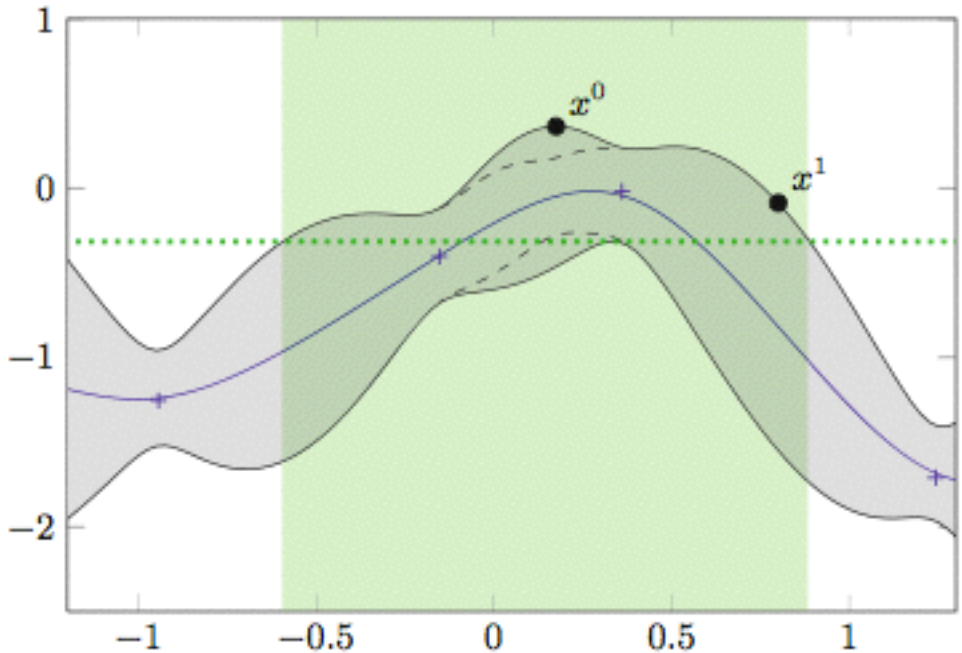


Fig. 2. Two queries of GP-UCB-PE on the previous example. The lower confidence bound on the maximum is represented by the horizontal dotted green line at $y_t^\bullet$. The relevant region $\mathfrak{R}$ is shown in light green (without edges). The first query x^0 is the maximizer of $\widehat{f}^+$. We show in dashed line the upper and lower bounds with the update of $\widehat{\sigma}$ after having selected x^0. The second query x^1 is the one maximizing the uncertainty inside $\mathfrak{R}^+$, an extension of $\mathfrak{R}$ which is not illustrated here.

GP-UCB-PE algorithm

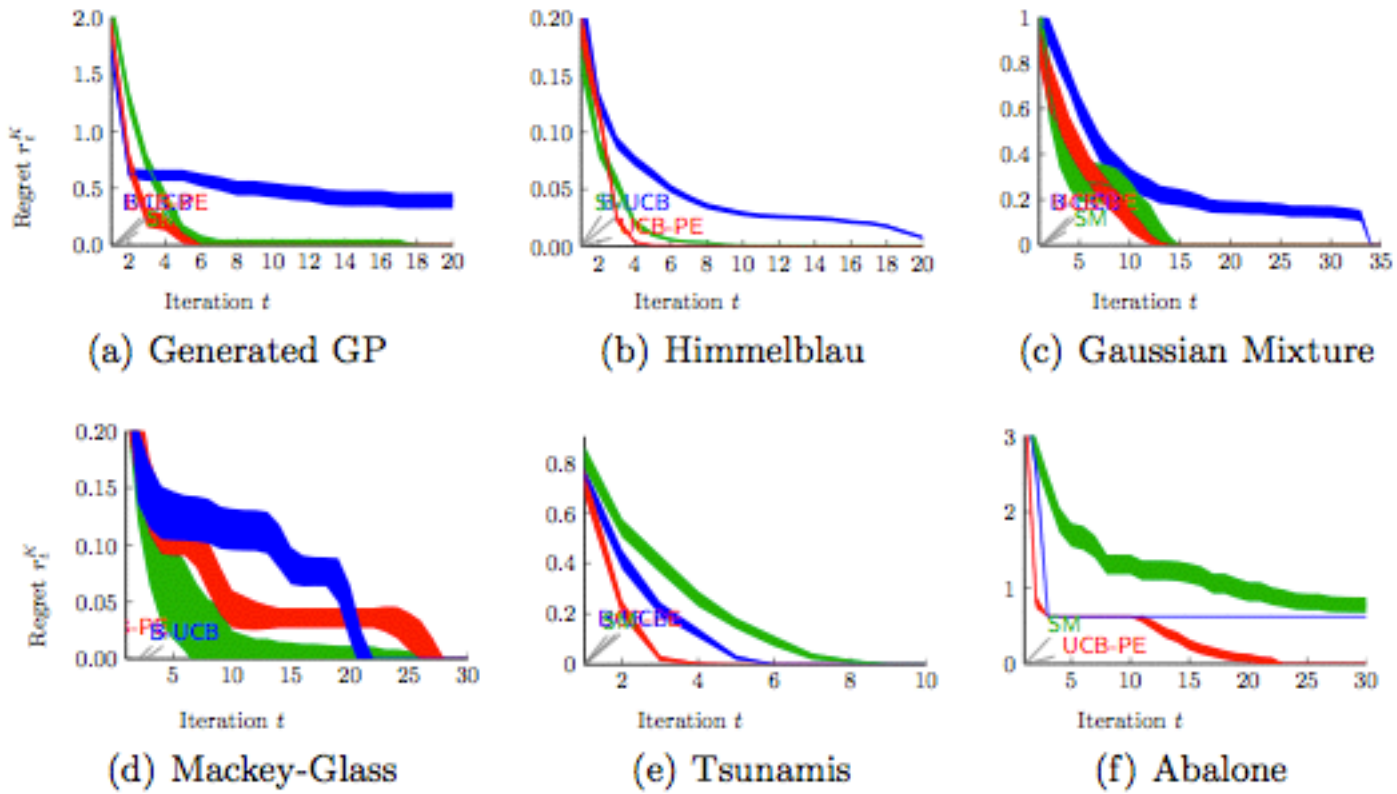
- For t = 0…T do

  Compute mu and sigma using Bayesian inference
  Choose first location according to GP-UCB

  for k = 1, …, B-1 do
    Compute sigma using Bayesian inference
    Choose argmax of sigma from the relevant region
  end for

  Query rewards
  end for

Results



(a) Generated GP (b) Himmelblau (c) Gaussian Mixture

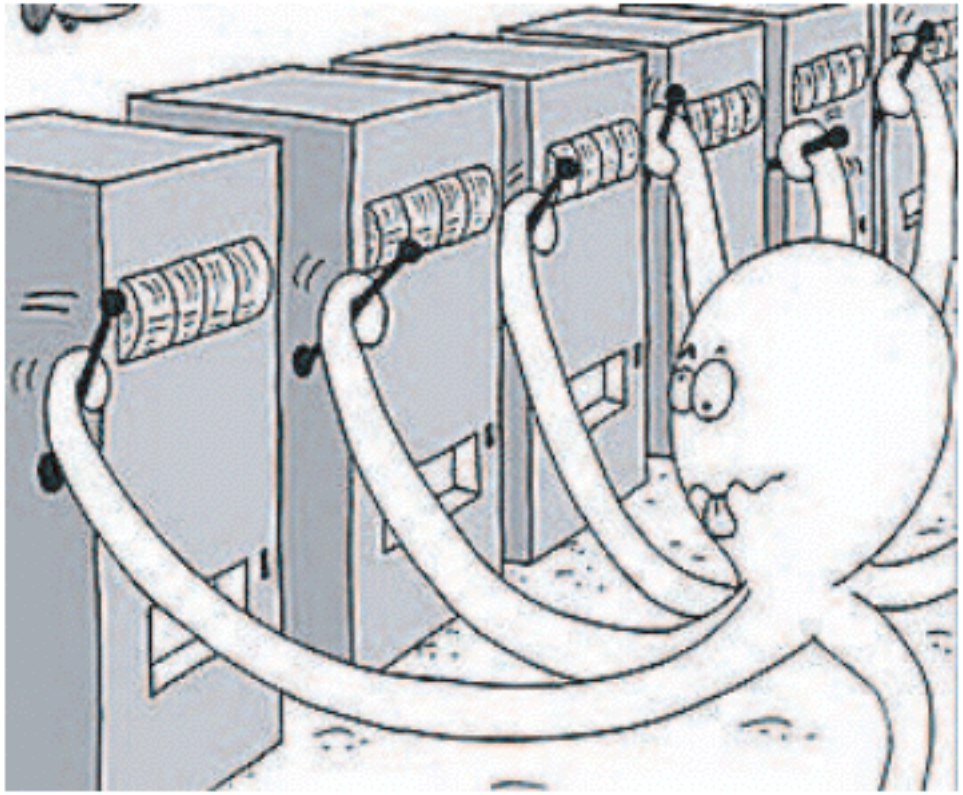
(d) Mackey-Glass (e) Tsunamis (f) Abalone

Fig. 4. Experiments on several real and synthetics tasks. The curves show the decay of the mean of the simple regret r_t^K with respect to the iteration t, over 64 experiments. We show with the translucent area the confidence intervals.

Empirical results

- JAVA and Matlab implementation
- using standard data until now
- Also thinking on including Monte Carlo - based SM-UCB and SM-MEI

Discussion



References

- [1] J. C. Gittins: Bandit Processes and Dynamic Allocation Indices *Journal of the Royal Statistical Society. Series B (Methodological)* Vol. 41, No. 2 (1979), pp. 148-177 http://www.jstor.org/stable/2985029
- [2] T. L. Lai and Herbert Robbins: Asymptotically Efficient Adaptive Allocation Rules *Advances in applied mathematics* Vol. 6, No. 1 (1985), pp. 4-22 http://dx.doi.org/10.1016%2F0196-8858%2885%2990002-8
- [3] Peter Auer, Nicolò Cesa-Bianchi and Paul Fischer: Finite-time Analysis of the Multiarmed Bandit Problem *Machine Learning* Vol. 47, No. 2-3 (2002), pp. 235-256 http://dx.doi.org/10.1023/A:1013689704352“

References

- [4] Niranjan Srinivas, Andreas Krause, Sham Kakade, and Matthias Seeger: Gaussian process optimization in the bandit setting: No regret and experimental design. *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pp. 1015–1022 http://las.ethz.ch/files/srinivas10gaussian.pdf
- [5] Thomas Desautels, Andreas Krause and Joel Burdick: Parallelizing Exploration-Exploitation Tradeoffs with Gaussian Process Bandit Optimization *Proceedings of the 29th International Conference on Machine Learning (ICML-12)* http://las.ethz.ch/files/desautels12parallelizing.pdf
- [6] Emile Contal, David Buffoni, Alexandre Robicquet and Nicolas Vayatis: Parallel Gaussian Process Optimization with Upper Confidence Bound and Pure Exploration *Machine Learning and Knowledge Discovery in Databases (Springer)* http://arxiv.org/abs/1304.5350