Knowing The What But Not The Where in Bayesian Optimization

Vu Nguyen, Michael A Osborne

Digital Twin | DRC | 07/12/2022

彭家祐 (Jack) Intern

Introduction

- **Motivation**：In some settings, the **optimum output $f^* = f(x^*)$ is known** (e.g. optimum accuracy is 100 in tuning classification algorithm)
- **Goal**：exploit the knowledge about f^* to search for the input x^* efficiently (how to efficiently utilize such prior knowledge to find the optimal inputs using the fewest number of queries)
- **Method**
  - use the knowledge of f^* to build a **transformed GP** surrogate model
  - proposed two novel **acquisition functions** (confidence bound minimization & expected regret minimization)
- **Contribution**
  - a first study of Bayesian optimization for exploiting the known optimum output f^*
  - a transformed Gaussian process surrogate using the knowledge of f^*
  - two novel acquisition functions to efficiently select the optimum location given f^*

Available acquisition functions for the known f*

- **Expected improvement with known incumbent f*** (Wang & de Freitas, 2014; Berk et al., 2018)
  - Typical choice of the incumbent is the best observed value so far in the observation set

$$\mathbb{E}\left[I_t\left(\mathbf{x}\right)\right] = \mathbb{E}\left[\max\left\{0, f\left(\mathbf{x}\right) - \xi\right\}\right] \qquad \xi = \max_{y_i \in \mathscr{D}_{t-1}} y_i \text{ where } \mathscr{D}_{t-1}$$

  - Given the known optimum output f*, one can readily use it as the incumbent

$$\alpha^{\mathrm{EI}^*}\left(\mathbf{x}\right) = \sigma\left(\mathbf{x}\right)\phi\left(z\right) + \left[\mu\left(\mathbf{x}\right) - f^*\right]\Phi\left(z\right) \qquad z = \frac{\mu(\mathbf{x}) - f^*}{\sigma(\mathbf{x})}$$

- **Output entropy search with known f*** (Wang & Jegelka,2017)
  - Given the known f* value, MES (Max-value Entropy Search) approximates I(x,y; f*) using a truncated Gaussian distribution such that the distribution of y needs to satisfy y < f*

$$I(\mathbf{x}, y; f^*) = H\left[p(y|D_t, \mathbf{x})\right] - \mathbb{E}[H\left(p(y|D_t, \mathbf{x}, f^*)\right)]p(f^*|D_t)$$

Let $\gamma(\mathbf{x}, f^*) = \frac{f^* - \mu(\mathbf{x})}{\sigma(\mathbf{x})}$, we have the MES* as

$$\alpha^{\mathrm{MES}^*}(\mathbf{x} \mid f^*) = \frac{\gamma(\mathbf{x}, f^*)\phi\left[\gamma(\mathbf{x}, f^*)\right]}{2\Phi\left[\gamma(\mathbf{x}, f^*)\right]} - \log\Phi\left[\gamma(\mathbf{x}, f^*)\right].$$

Gaussian process transformation for $f \leq f^*$

- Transformation with sigmoid or tanh …
  - problem
    - need to know lower and upper bound of f(x), but we do not know the lower bound in the setting
    - Under these transformations will become the Gaussian process classification problem

- Transform the output of a GP using warping
  - Problem
    - Less efficient
    - Requires more data points

- **Linearization trick**
  - ensures that we arrive at another GP after transformation given our existing GP
  - In this paper, we shall follow this linearization trick to transform the surrogate model given f*

$$L(x) = f(a) + f'(a)(x-a)$$

Gaussian process transformation for $f \leq f^*$

- Transformation with sigmoid or tanh …
  - problem
    - need to know lower and upper bound of f(x), but we do not know the lower bound in the setting
    - Under these transformations will become the Gaussian process classification problem

- Transform the output of a GP using warping
  - Problem
    - Less efficient
    - Requires more data points

- **Linearization trick**
  - ensures that we arrive at another GP after transformation given our existing GP
  - In this paper, we shall follow this linearization trick to transform the surrogate model given f*

$$L(x) = f(a) + f'(a)(x-a)$$

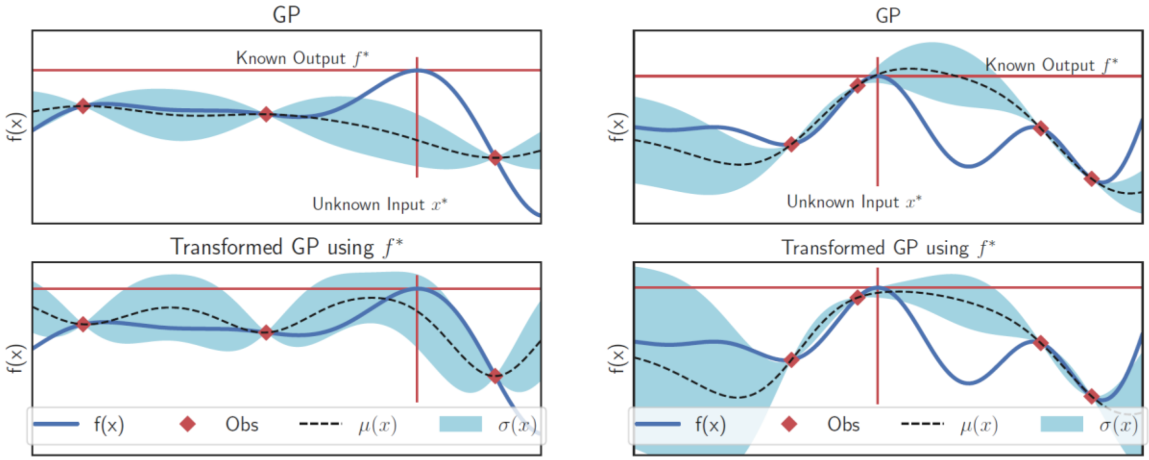
Proposed Method - Transformed GP (2/2)

- **Transformed Gaussian process**
  - Linear transformation of a GP remains GP, the predictive posterior distribution for f has a closed form :

$$p(f \mid .) = \mathcal{N}(f \mid \mu, \sigma) \qquad \mu(\mathbf{x}) = f^* - \frac{1}{2}\mu_g^2(\mathbf{x}),$$
$$\sigma(\mathbf{x}) = \mu_g(\mathbf{x})\sigma_g(\mathbf{x})\mu_g(\mathbf{x})$$

  - Effect of the transformation : predictive uncertainty $\sigma(\mathbf{x})$ becomes larger than normal GP where $\mu(\mathbf{x})$ is low → let some acquisition functions (e.g., UCB, EI) explore more aggressively

- **Transformed GP compared with normal GP**
  - transforming the GP using f*, we encode the knowledge about f* into the surrogate model
  - Transformed GP Surrogate model gets close to optimum output f* but never above f*



Proposed Method - Confidence bound minimization

- GP surrogate at any location x with high probability : $\mu(\mathbf{x}) - \sqrt{\beta_t}\sigma(\mathbf{x}) \leq f(\mathbf{x}) \leq \mu(\mathbf{x}) + \sqrt{\beta_t}\sigma(\mathbf{x})$
- β_t is a hyperparameter. Given the knowledge of f *, we can express this property at the optimum location x* where $f^* = f(\mathbf{x}^*)$ to have w.h.p

$$\mu(\mathbf{x}^*) - \sqrt{\beta_t}\sigma(\mathbf{x}^*) \leq f^* \leq \mu(\mathbf{x}^*) + \sqrt{\beta_t}\sigma(\mathbf{x}^*) \Longleftrightarrow |\mu(\mathbf{x}^*) - f^*| \leq \sqrt{\beta_t}\sigma(\mathbf{x}^*).$$

- Selecting next point by taking

$$\mathbf{x}_{t+1} = \operatorname*{argmin}_{\mathbf{x}\in\mathcal{X}} \alpha_t^{\text{CBM}}(\mathbf{x}). \qquad \alpha_t^{\text{CBM}}(\mathbf{x}) = |\mu(\mathbf{x}) - f^*| + \sqrt{\beta_t}\sigma(\mathbf{x})$$

- Take the minimum value at ideal location where $\boxed{\mu(\mathbf{x}_t) = f^*, \sigma(\mathbf{x}_t) = 0}$
- Has hyperparameter β_t , to which performance can be sensitive → propose another acquisition function incorporating the knowledge of f* using no hyperparameter

Proposed Method - Expected regret minimization

- Start with the regret function $r(\mathbf{x}) = f^* - f(\mathbf{x})$
- The probability of regret r(x) on a normal posterior distribution : $p(r) = \frac{1}{\sqrt{2\pi}\sigma(\mathbf{x})} \exp\left(-\frac{1}{2}\frac{[f^* - \mu(\mathbf{x}) - r(\mathbf{x})]^2}{\sigma^2(\mathbf{x})}\right)$
- acquisition function to minimize this expected regret as $\alpha^{\mathrm{ERM}}(\mathbf{x}) = \mathbb{E}[r(\mathbf{x})]$

$$\mathbb{E}[r(\mathbf{x})] = \int \frac{r}{\sqrt{2\pi}\sigma(\mathbf{x})} \exp\left(-\frac{1}{2}\frac{[f^* - \mu(\mathbf{x}) - r(\mathbf{x})]^2}{\sigma^2(\mathbf{x})}\right) dr.$$

- Let z = $\frac{f^* - \mu(\mathbf{x})}{\sigma(\mathbf{x})}$, we obtain the closed-form computation as $\alpha^{\mathrm{ERM}}(\mathbf{x}) = \sigma(\mathbf{x})\phi(z) + [f^* - \mu(\mathbf{x})]\Phi(z)$
- select the next point, we minimize this acquisition function which is equivalent to minimizing the expected regret $\mathbf{x}_{t+1} = \arg\min_{\mathbf{x}\in\mathscr{X}} \alpha^{\mathrm{ERM}}(\mathbf{x}) = \arg\min_{\mathbf{x}\in\mathscr{X}} \mathbb{E}[r(\mathbf{x})]$
- Take the minimum value at ideal location where $\boxed{\mu(\mathbf{x}_t) = f^*,\ \sigma(\mathbf{x}_t) = 0}$

- Original EI prefer high GP mean and variance (is to balance exploitation and exploration)
- ERM selects the point to minimize expected regret $\mathbb{E}[f^* - f(\mathbf{x})]$ with $\mu(\mathbf{x}) \rightarrow$ f* with low variance

Algorithm – BO with known optimum output

- Given original observation $\{\mathbf{x}_i, y_i\}_{i=1}^{N}$ and f*, compute $g_i = \sqrt{2(f^* - y_i)}$ to build a transformed GP
- Using transformed GP, predict the mean and variance at any location x
- compute the CBM and ERM acquisition functions to select next point

Algorithm 1 BO with known optimum output.

Input: #iter T, optimum value $f^* = \max_{\mathbf{x} \in \mathscr{X}} f(\mathbf{x})$

1: **while** $t \leq T$ and $f^* > \max_{\forall y_i \in D_t} y_i$ **do**

2: Construct a transformed Gaussian process surrogate model from $\mathscr{D}_t$ and f^*.

3: Estimating μ and σ from Eqs. (2) and (3).

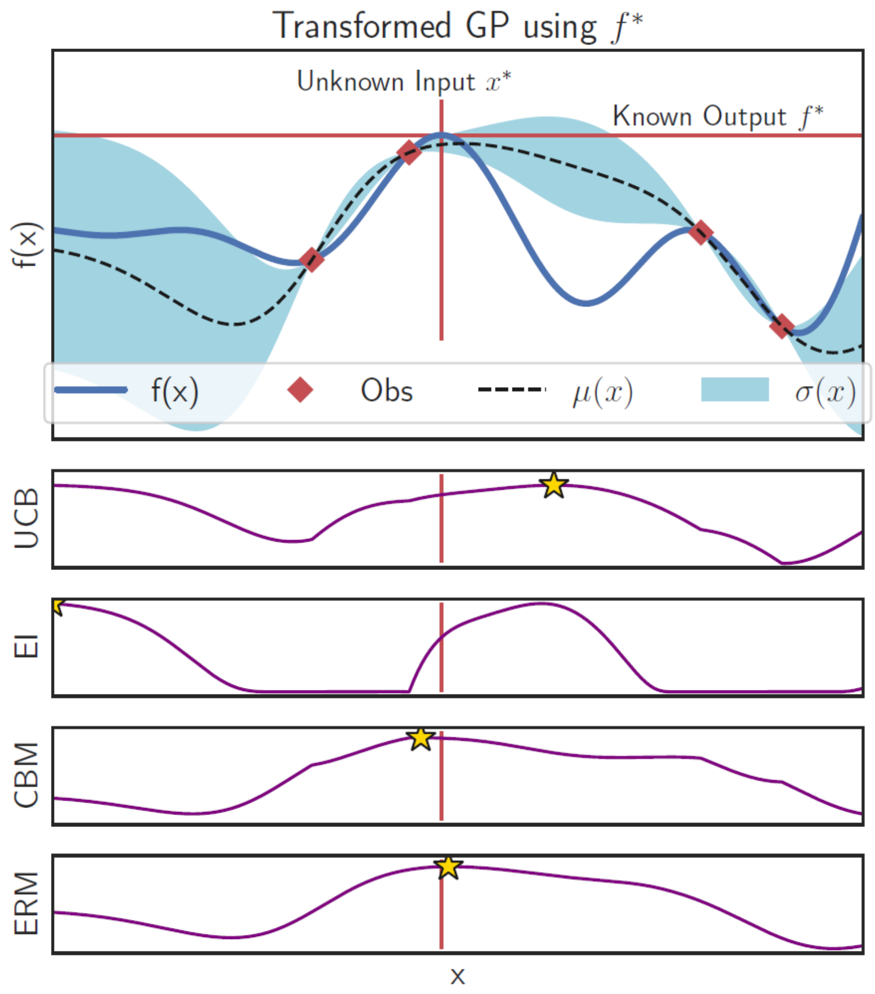
4: Select $\mathbf{x}_t = \arg\min_{\mathbf{x} \in \mathscr{X}} \alpha_t^{\mathrm{ERM}}(\mathbf{x})$, or $\alpha_t^{\mathrm{CBM}}(\mathbf{x})$, using the above transformed GP model.

5: Evaluate $y_t = f(\mathbf{x}_t)$, set $g_t = \sqrt{2(f^* - y_t)}$ and augment $\mathscr{D}_t = \mathscr{D}_{t-1} \cup (\mathbf{x}_t, y_t, g_t)$.

6: **end while**

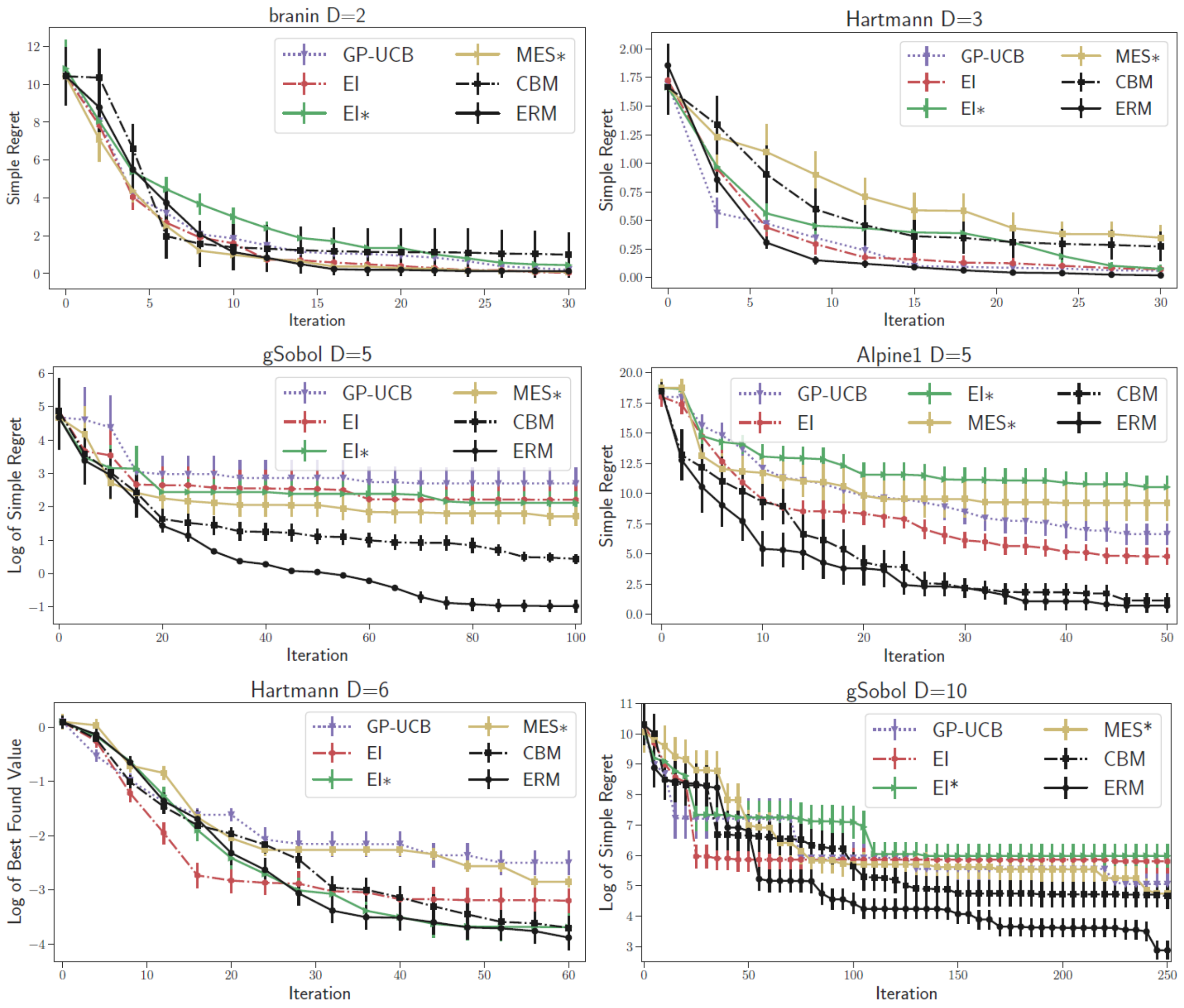
Experiment – Comparison of Acquisition function

- CBM and ERM will select the location where the GP mean m(x) is close to the optimal value f* and we are highly certain about it (low $\boldsymbol{\sigma}(\mathbf{x})$)
- UCB and EI will always keep exploring as the principle of explore-exploit without using the knowledge of f*
- UCB and EI can not identify the unknown location x efficiently as opposed to proposed acquisition functions



Experiment – 6 Benchmark functions given f *

- Proposed framework has utilized the additional knowledge of f* to build an informed surrogate model and decision functions
- ERM outperforms all methods by a wide margin.
- CBM can be sensitive to the hyperparameter, ERM has no parameter and is thus more robust
- approaches with f* perform significantly better than the baselines in gSobol and Alpine1 functions.
- The results indicate that the knowledge of f* is particularly useful for high dimensional functions.

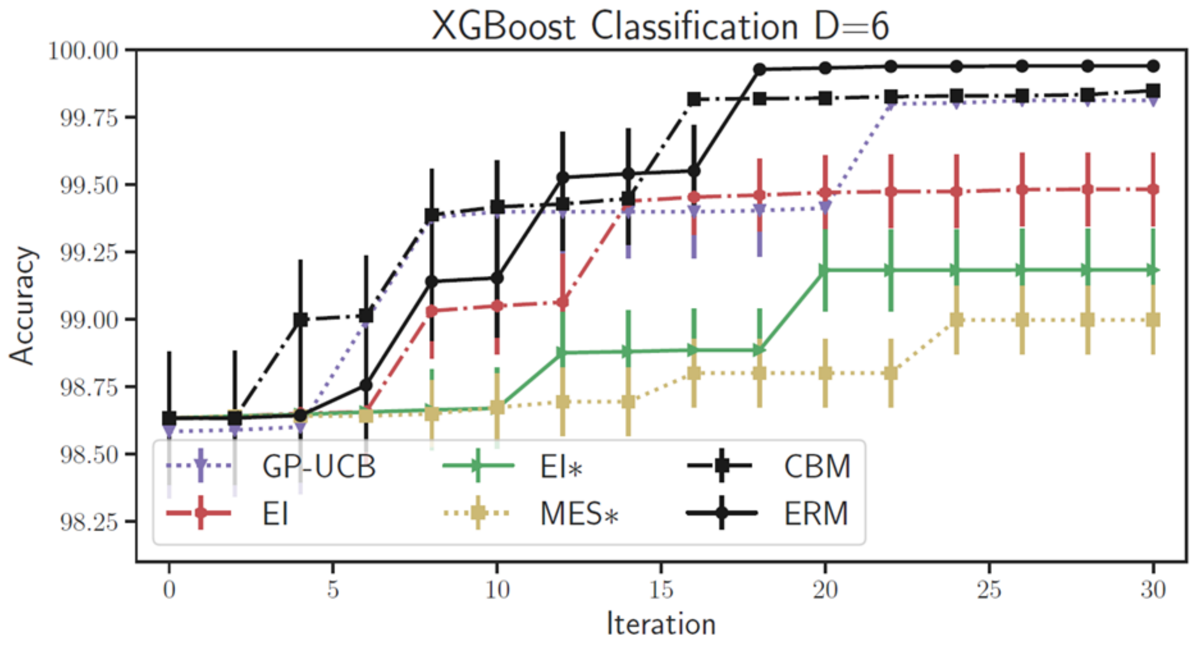


Experiment – Tuning machine learning algorithms with f *

- XGBoost classification
  - Skin Segmentation dataset / best accuracy is f* = 100 / 6 hyperparameters
  - To optimize the integer(ordinal) variables, round the scalars to the nearest values in the continuous space
  - proposed ERM is the best approach, outperforming all the baselines by a wide margin. This demonstrates the benefit of exploiting the optimum value f* in BO

Table 1. Hyperparameters for XGBoost.

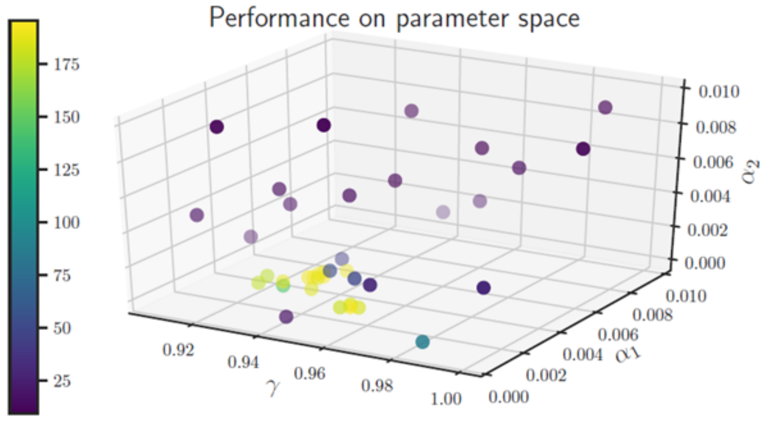
Known $f^* = 100$ (Accuracy)			
Variables	Min	Max	Found $\mathbf{x}^*$
min child weight	1	20	4.66
colsample bytree	0.1	1	0.99
max depth	5	15	9.71
subsample	0.5	1	0.77
alpha	0	10	0.82
gamma	0	10	0.51

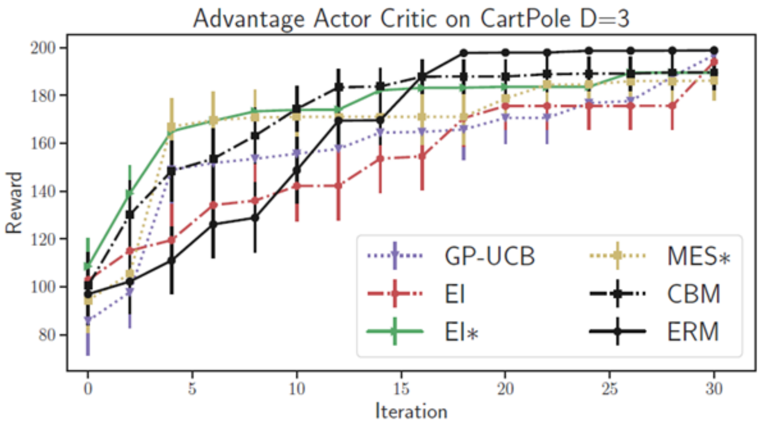


Experiment – Tuning Deep reinforcement learning

- CartPole Problem
  - The goal is to keep the cartpole balanced by controlling a pivot point
  - maximum reward is known from the literature as f* = 200
  - Tuning 3 hyperparameters : discount factor, learning rate for actor model, learning rate for critic model

  - ERM reaches the optimal performance after 20 iterations outperforming all other baselines
  - In Fig. 6 Left, we visualize the selected point $\{\mathbf{x}_t\}_{t=1}^{T}$ by our ERM acquisition function.
  - Our ERM initially explores at several places and then exploits in the high value region (yellow dots)





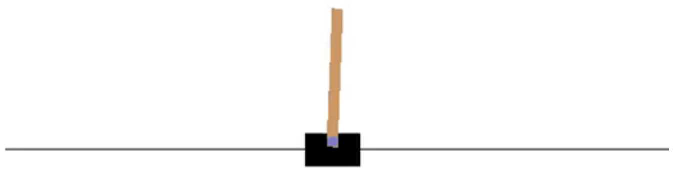
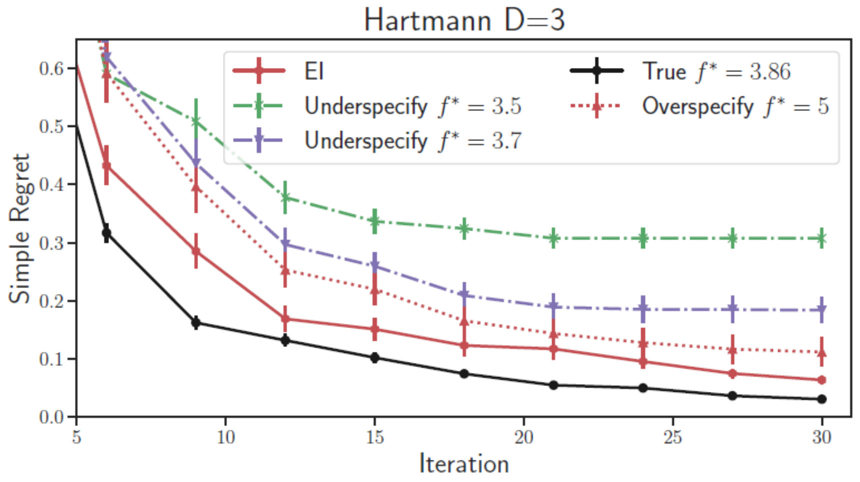
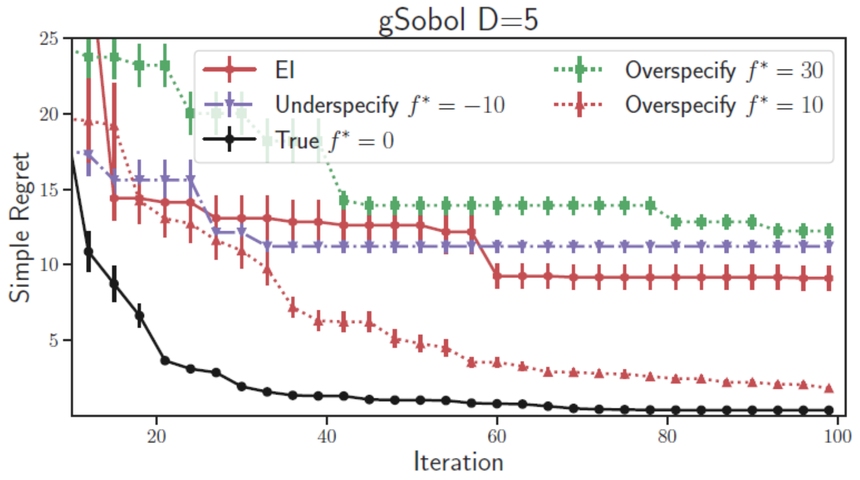


Figure 6. Hyperparameter tuning for a deep reinforcement learning algorithm. The optimum value is available $f^* = 200$. Left: Selected points by our algorithm on tuning DRL. Color indicates the reward $f(x)$ value. Right: Performance comparison with the baselines.

Experiment – misspecify the optimum value

- setting the f* to a value which is not the true optimum of the black-box function
- under-specifying case will result in worse performance than over-specifying because our acquisition function will get stuck at the area once being found wrongly as the optimal
- over-specify f*,our algorithm continues exploring to find the optimum because it can not find the point where both conditions are met $\sigma(\mathbf{x}_t) = 0$ and $f^* = \mu(\mathbf{x}_t)$



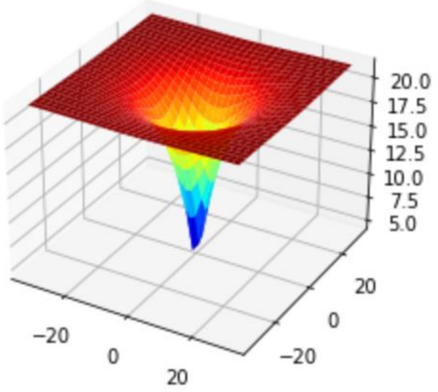


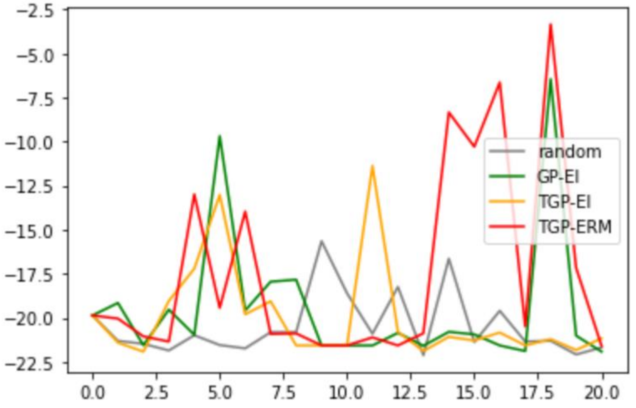
Conclusion

- Considered a new setting in Bayesian optimization with **known optimum output**
- Proposed **transformed Gaussian process** to model the objective function better by **exploiting the knowledge of f***
- **Proposed two decision strategies** which can **exploit the function optimum value** to make informed decisions
- Know the true value f*, **ERM can converge quickly to the optimum** in benchmark functions and real-world applications
- Do not know the exact f* value, the performance of our approach is degraded, should use the existing BO approaches (such as EI) for the best performance
- Expand proposed algorithm to handle batch setting for parallel evaluations
- Extend this work to other classes of surrogate functions such as Bayesian neural networks and deep GP

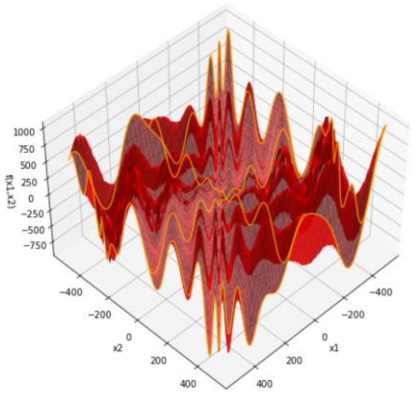
Demo - Other Benchmark Functions (2-D)

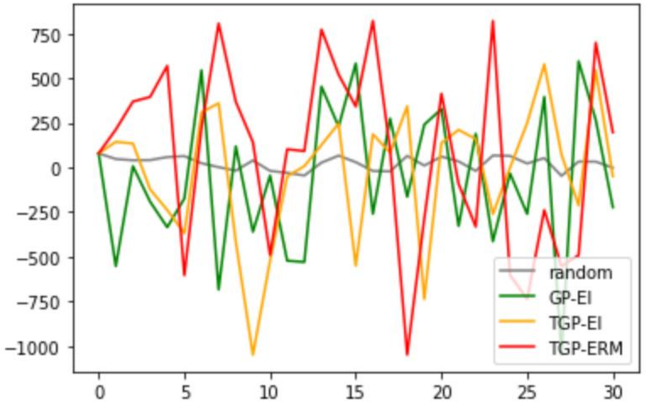
- Ackley (minimization)
  - f* = 0
  - x* = (0,0)
  - 20 iterations





- Eggholder (minimization)
  - f* = -959.6407
  - x* = (512, 404.2319)
  - 30 iterations





- Drop Wave (minimization)
  - f* = -1
  - x* = (0, 0)
  - 20 iterations