Bayesian Optimization with Known Optimum Demo：Benchmark Functions & Tuning XGBoost Hyperparameters

Ref.：Knowing The What But Not The Where in Bayesian Optimization

Digital Twin │ DRC │ 07/20/2022
彭家祐 (Jack) Intern

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 $f^* = \max_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x})$

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 $\mathcal{D}_t$ and f^*.
3: Estimating μ and σ from Eqs. (2) and (3).
4: Select $\mathbf{x}_t = \arg\min_{\mathbf{x} \in \mathcal{X}} \alpha_t^{\text{ERM}}(\mathbf{x})$, or $\alpha_t^{\text{CBM}}(\mathbf{x})$, using the above transformed GP model.
5: Evaluate $y_t = f(\mathbf{x}_t)$, set $g_t = \sqrt{2(f^* - y_t)}$ and augment $\mathcal{D}_t = \mathcal{D}_{t-1} \cup (\mathbf{x}_t, y_t, g_t)$.
6: **end while**

Proposed Acquisition Functions :

Confidence Bound Minimization

$$\alpha_t^{\text{CBM}}(\mathbf{x}) = |\mu(\mathbf{x}) - f^*| + \sqrt{\beta_t}\sigma(\mathbf{x})$$

Expected Regret Minimization

$$\alpha^{\text{ERM}}(\mathbf{x}) = \sigma(\mathbf{x})\phi(z) + [f^* - \mu(\mathbf{x})]\Phi(z)$$

Take the minimum value
at ideal location where $\mu(\mathbf{x}_t) = f^*$, $\sigma(\mathbf{x}_t) = 0$

The setting of Beta for UCB & CBM

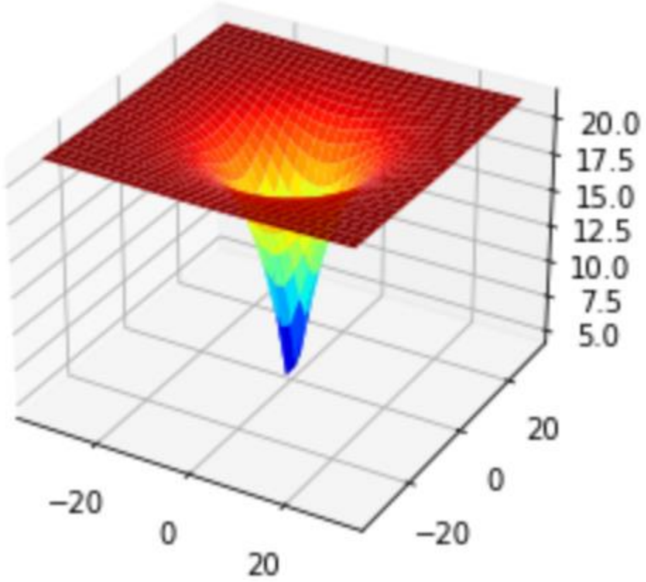
- $\beta = \log\ (\#\ of\ GP\ fitting\ points)$
- more iteration → more data → β be larger

```python
def _gp_ucb(gp,xTest,fstar_scale=0):
    #dim=gp.dim
    #xTest=np.reshape(xTest,(-1,dim))
    mean, var= gp.predict(xTest)
    var.flags['WRITEABLE']=True
    #var=var.copy()
    var[var<1e-10]=0
    #mean=np.atleast_2d(mean).T
    #var=np.atleast_2d(var).T

    # Linear in D, log in t https://github.com/kirth
    #beta_t = gp.X.shape[1] * np.log(len(gp.Y))
    beta_t = np.log(len(gp.Y))

    #beta=300*0.1*np.log(5*len(gp.Y))# delta=0.2, ga
    temp=mean + np.sqrt(beta_t) * np.sqrt(var)
    #print("input",xTest.shape,"output",temp.shape)
    return  temp
```

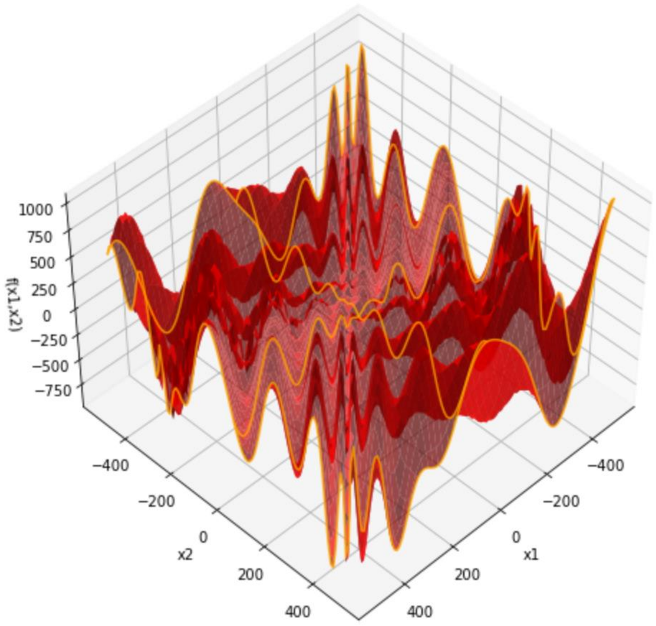
```python
def _cbm(x, gp, target): # confidence bound minimization
    mean, var = gp.predict(x)
    var.flags['WRITEABLE']=True
    var[var<1e-10]=0
     mean=np.atleast_2d(mean).T
     var=np.atleast_2d(var).T

    # Linear in D, log in t https://github.com/kirthevasank/add-gp-banc
    #beta_t = gp.X.shape[1] * np.log(len(gp.Y))
    beta_t = np.log(len(gp.Y))

    #beta=300*0.1*np.log(5*len(gp.Y))# delta=0.2, gamma_t=0.1
    return -np.abs(mean-target) - np.sqrt(beta_t) * np.sqrt(var)
```

Demo - 2D - Other Benchmark Functions

- Ackley (minimization)
  - f* = 0
  - x* = (0,0)
  - 較平滑、椎狀



- Eggholder (minimization)
  - f* = -959.6407
  - x* = (512, 404.2319)
  - 上下起伏、震盪



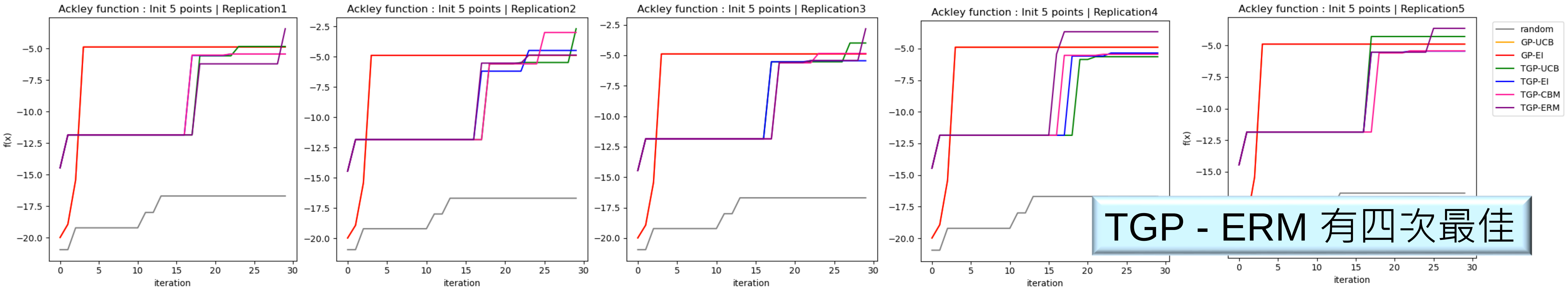
times -1 → maximization

Methods

Random
GP-UCB
GP-EI
TGP-UCB
TGP-EI
TGP-CBM
TGP-ERM

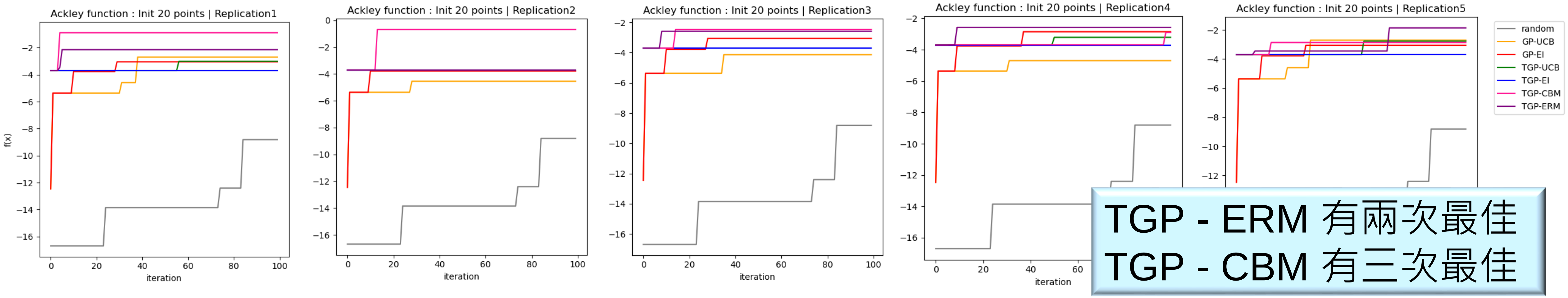
Demo - 2D - Ackley Function | f* = 0

Init 5 points | 30 iterations



TGP - ERM 有四次最佳

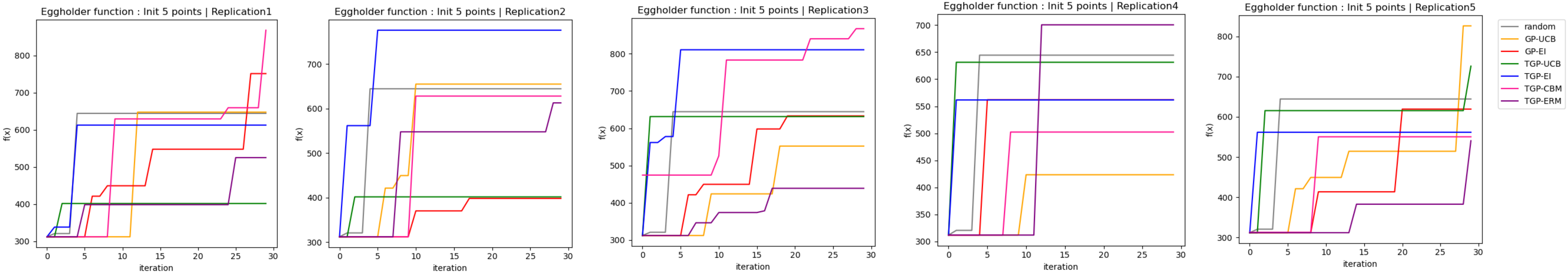
Init 20 points | 100 iterations



TGP - ERM 有兩次最佳
TGP - CBM 有三次最佳

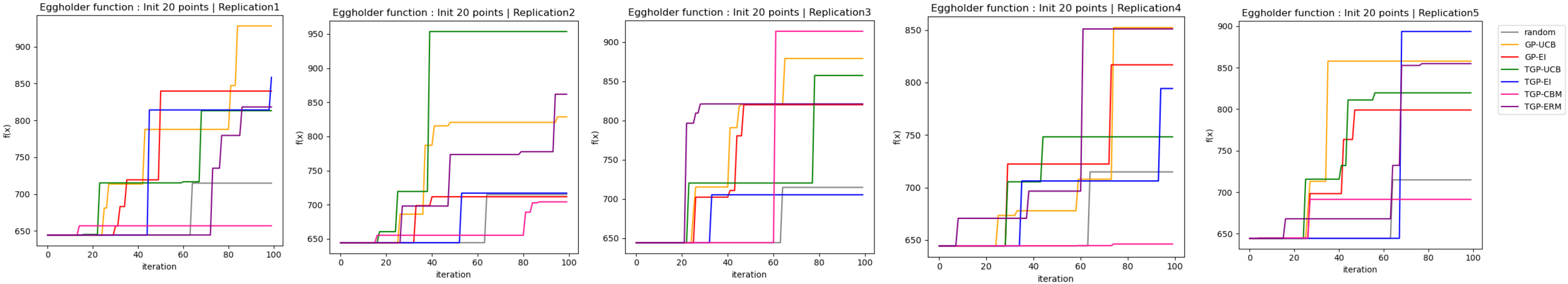
Demo - 2D - Eggholder Function | f* = -959.6407 * -1

Init 5 points | 30 iterations



Init 20 points | 100 iterations

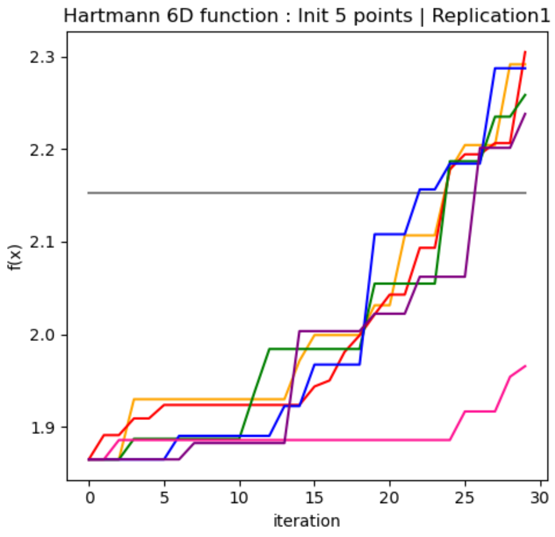
相較於Ackley Function的實驗，proposed acquisition function表現無較好

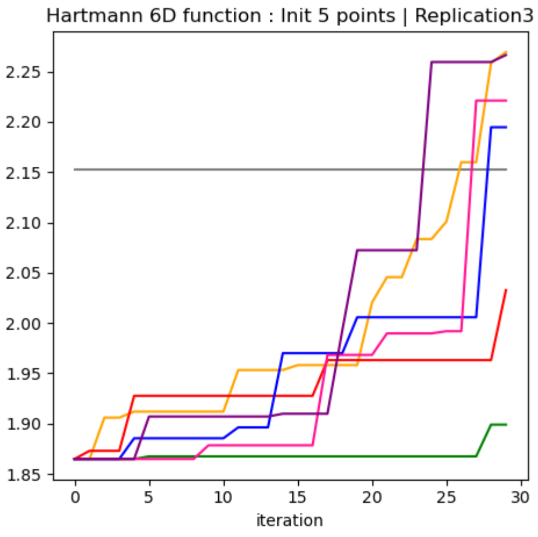


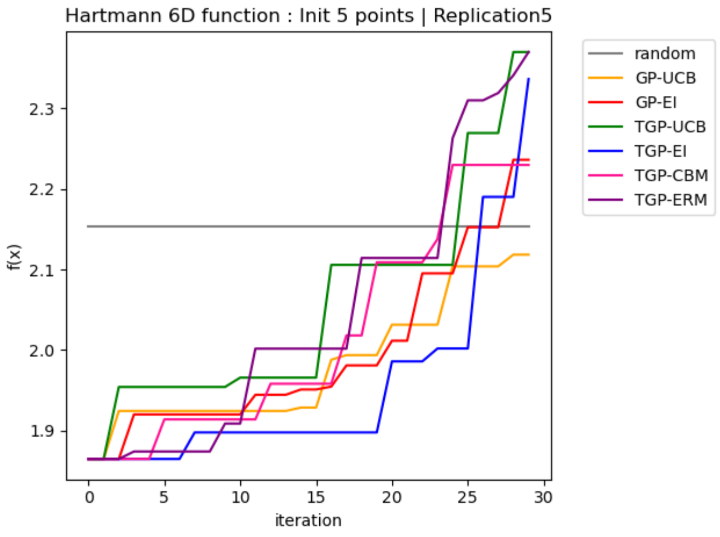
black box function很震盪→ UCB、EI偏好high variance，找到最佳組合的機率會較高?
(但現實面無法得知Black Box Function 長得如何)

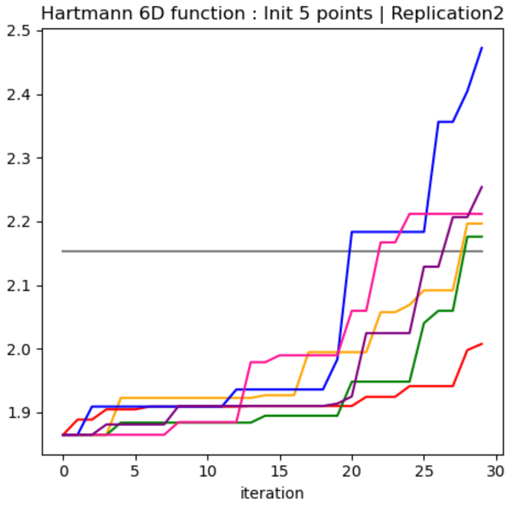
Demo - 6D – Hartmann Function | f* = -3.32237* -1

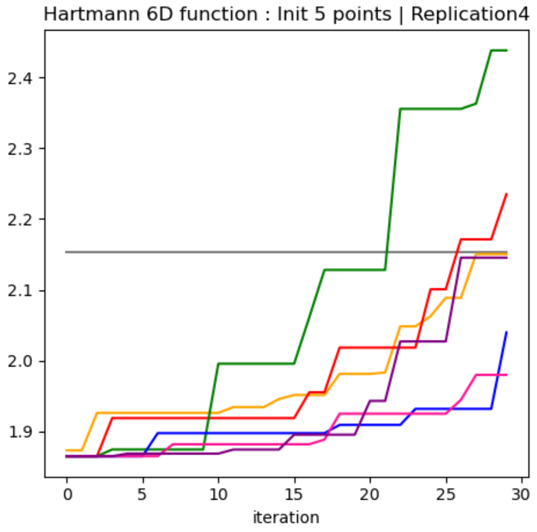
Init 5 points | 30 iterations











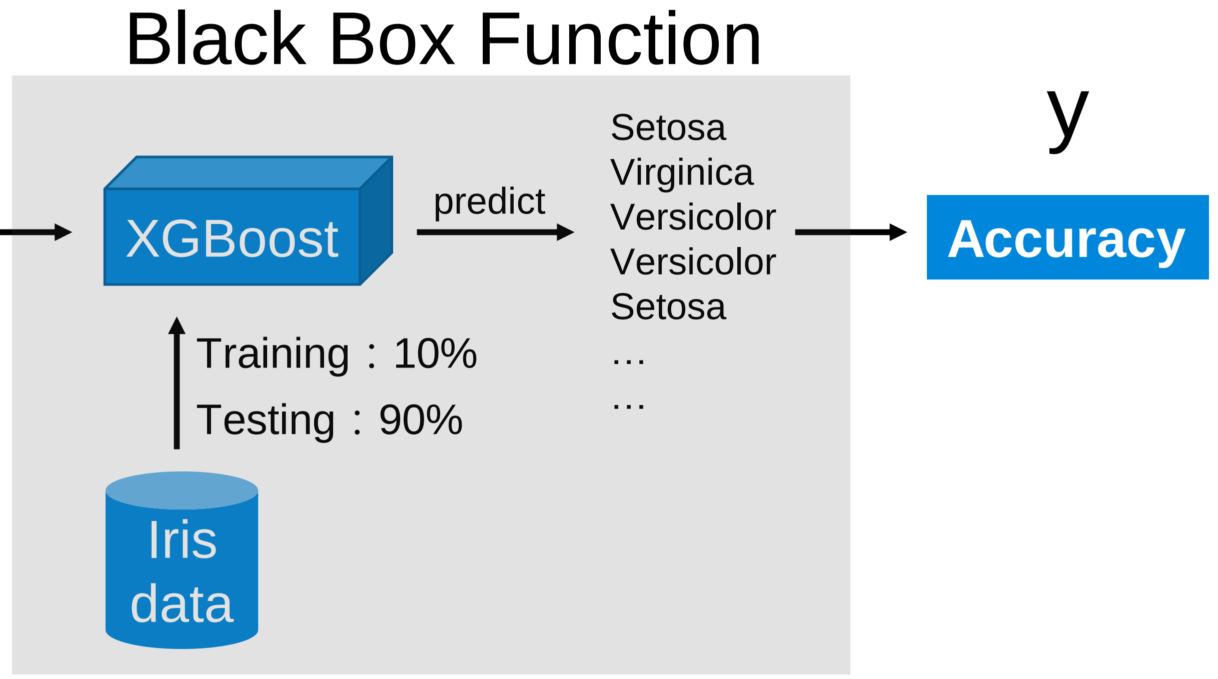
TGP - ERM 有兩次最佳

Demo – Tuning Hyperparameters of XGBoost

- Tuning 4 hyperparameters of XGBoost
- Data : iris dataset
- Metric : Accuracy → Maximization
- Goal : raise Accuracy

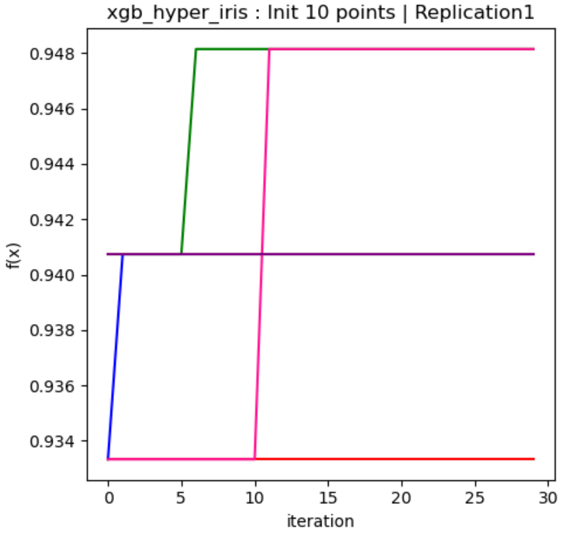
x

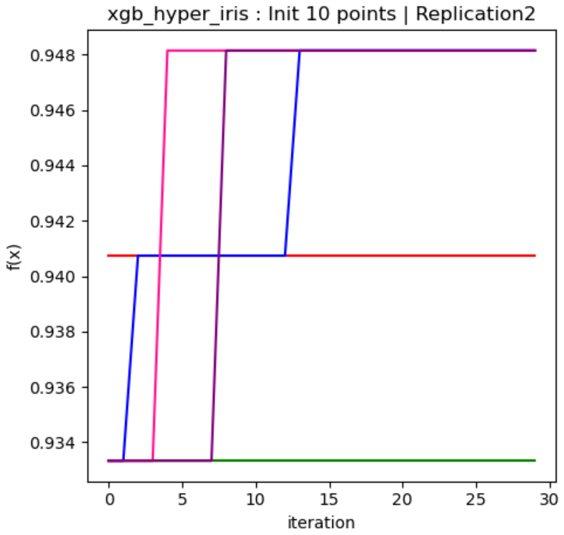
hyperparameter	boundary
Max_depth	(3, 9)
eta	(0.1, 0.5)
min_child_weight	(1, 20)
colsample_bytree	(0.1, 1)

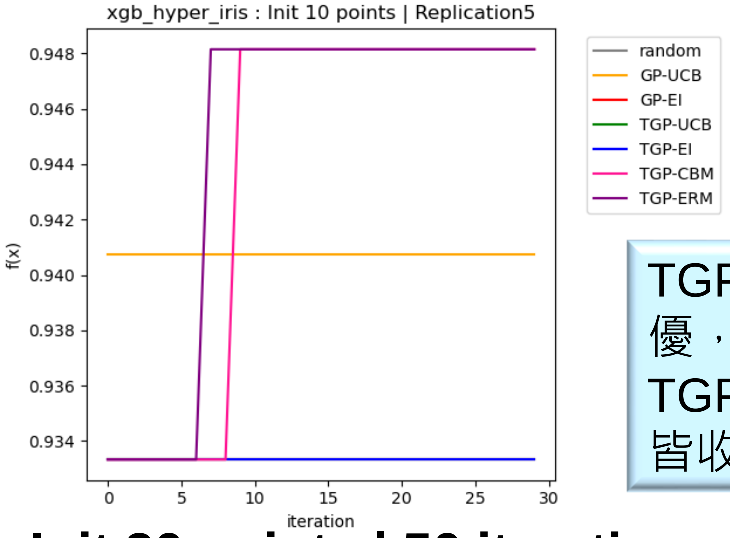


Demo – Tuning Hyperparameters of XGBoost

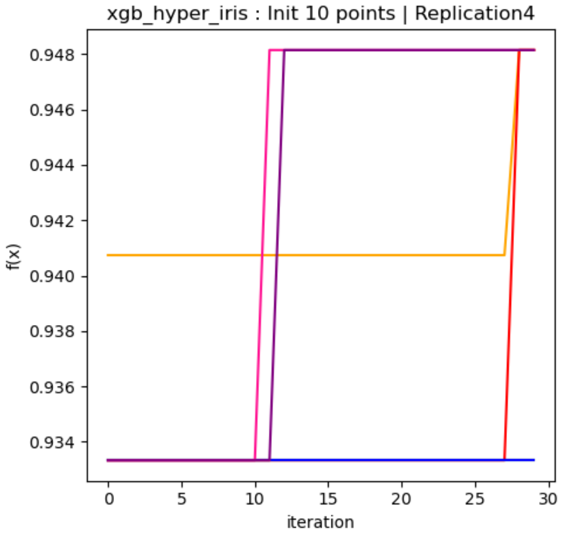
Init 10 points | 30 iterations

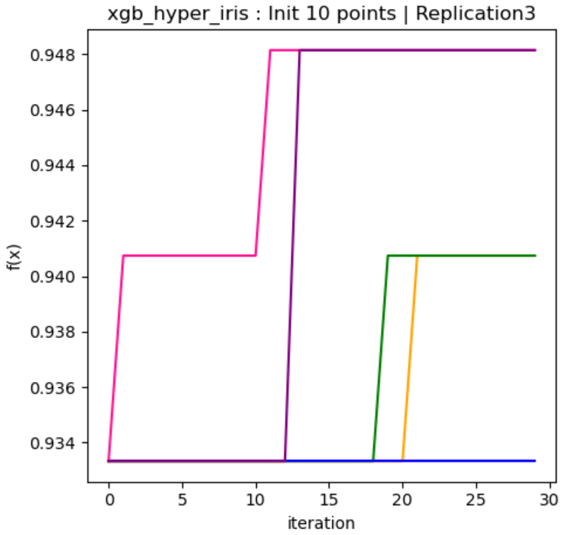




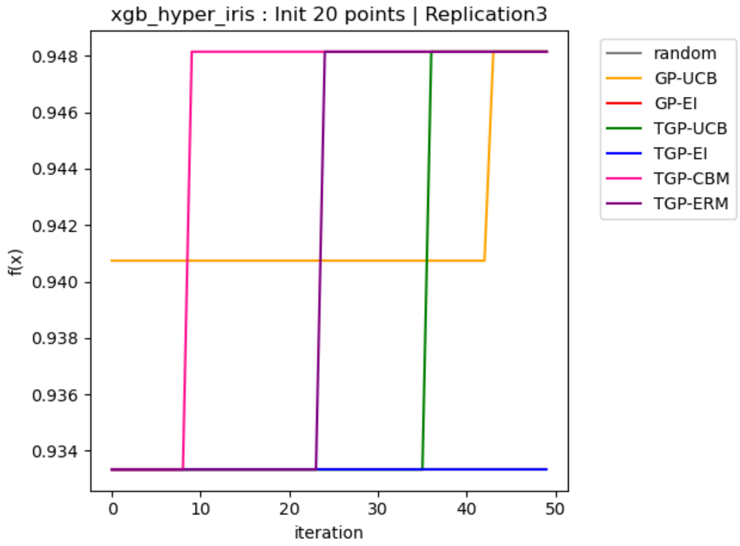


TGP – CBM每次皆最優，且四次為最快收斂
TGP – ERM幾乎每次皆收斂，且收斂快速





Init 20 points | 50 iterations



小結

- 實驗方式
  - 上述實驗皆以設立特定迭代次數
  - 與目標值差值的停止迭代機制之驗證(不同方法收斂所需的迭代次數)方式難以進行、耗時

- Experiments of benchmark functions
  - 平滑的black box function：acquisition function使用ERM或CBM的表現較佳
  - 震盪的black box function：使用EI、UCB較佳

- Tuning 4 hyperparameters of XGBoost with Iris Dataset
  - acquisition function使用CBM的表現最佳，ERM也表現不錯

- Transformed GP 若搭配EI、UCB，實驗無明顯效果