

# ベイズ最適化とは？

理化学研究所 仁科加速器科学研究センター  
森田泰之

# 自己紹介



仁科加速器研究センター イオン源開発チーム  
基礎科学特別研究員  
森田泰之

## 主な研究

- ・ ECRイオン源の自動制御 (博士論文)
- ・ ビーム診断機
- ・ ビーム輸送計算

あくまで**‘機械学習のユーザー’**であり、**‘機械学習の研究者’**ではありません。  
機械学習の数学に関しても勉強しましたが、完ぺきではありませんのでご容赦ください。

# 本講演の目的

- ベイズ最適化を使用する上で必要な理論と知識を

既存のライブラリを動作させられれば実用可能

➤ 0から自作できるほど完全に理論を理解することが目的ではない

1. 国内外での活用例      活用可能な状況・逆に活用が困難な状況の参考に
2. 理論的なお話
  - ガウス過程回帰      確率的な予測モデル
  - 最適化      測定点の決め方
3. 実践      GPyOptというライブラリを使ったチュートリアル

全体的に文字が多いですがご容赦ください。

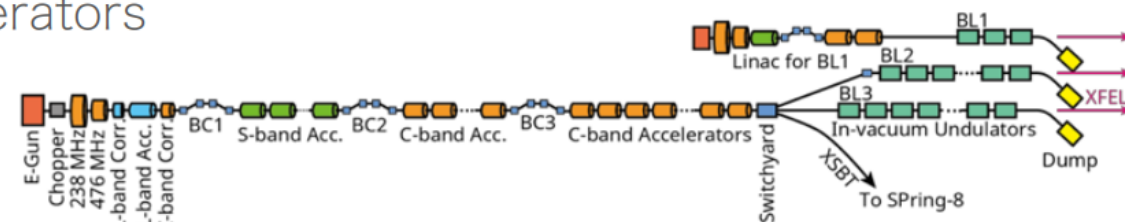
# Machine Learning @ SPring-8

► Gaussian Process based beam optimizer was developed and introduced at SACLA/XFEL

✓ Commonly used for actual accelerator tuning by operators

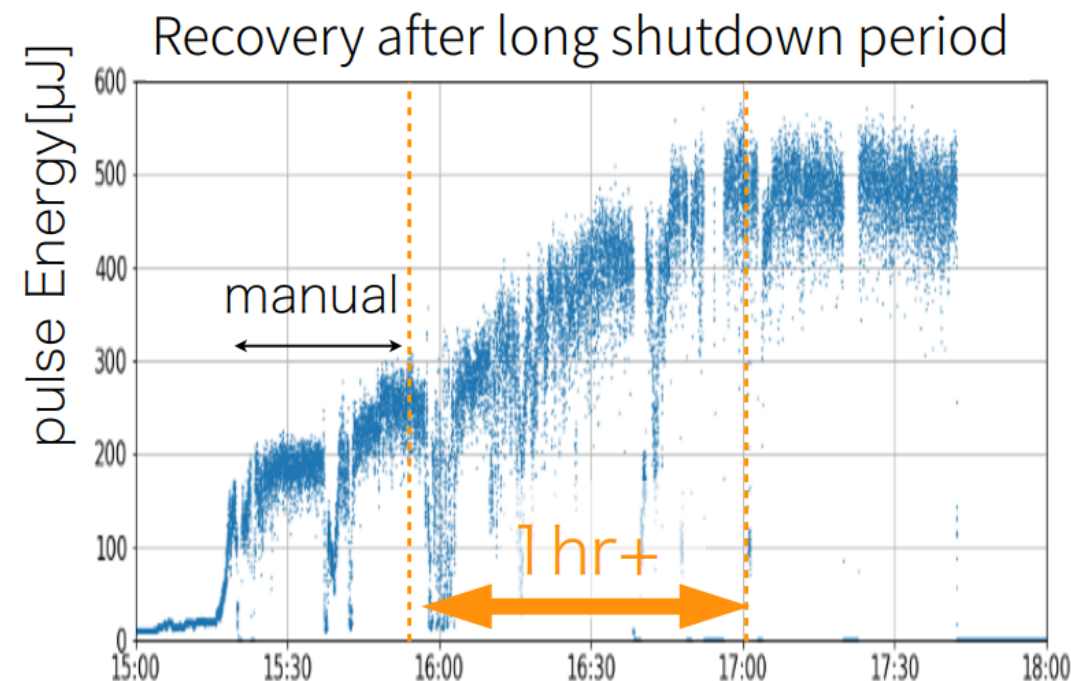
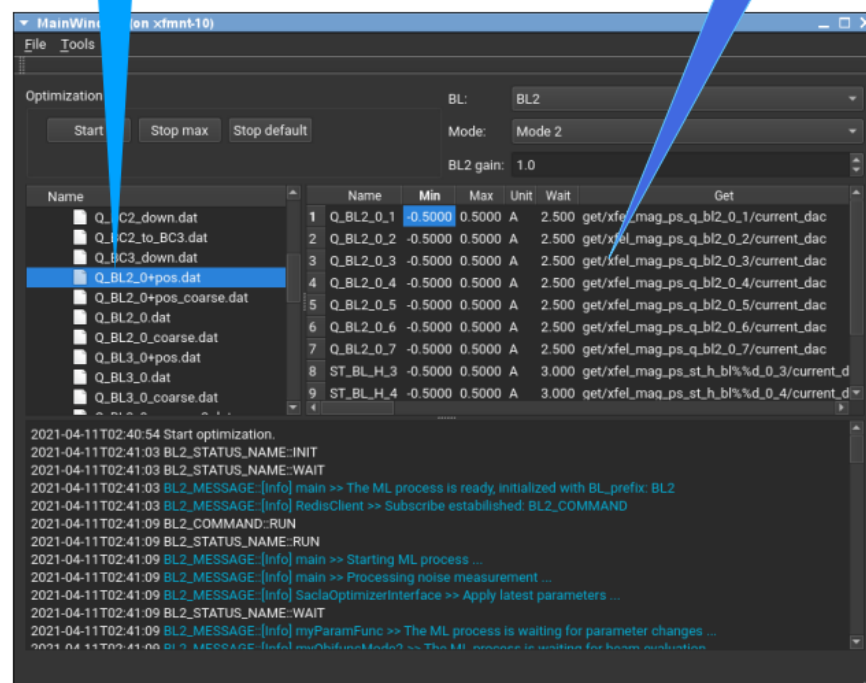
→ efficient tuning

✓ Contribution to achieve 1mJ@10keV (all-time high)



Templates of control parameters for purposes and sections

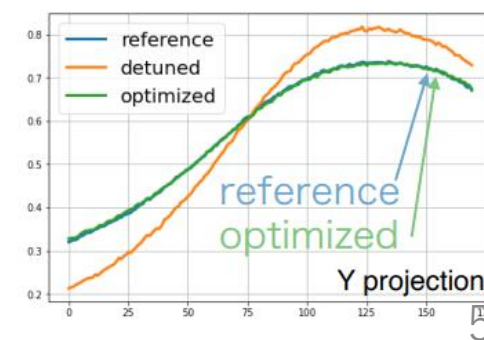
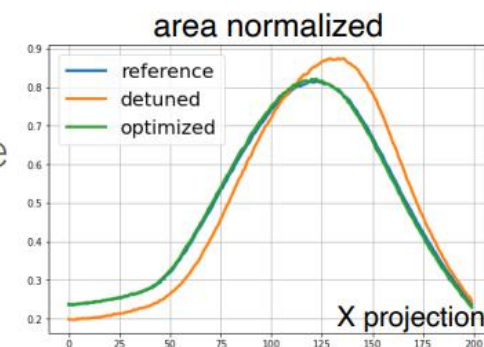
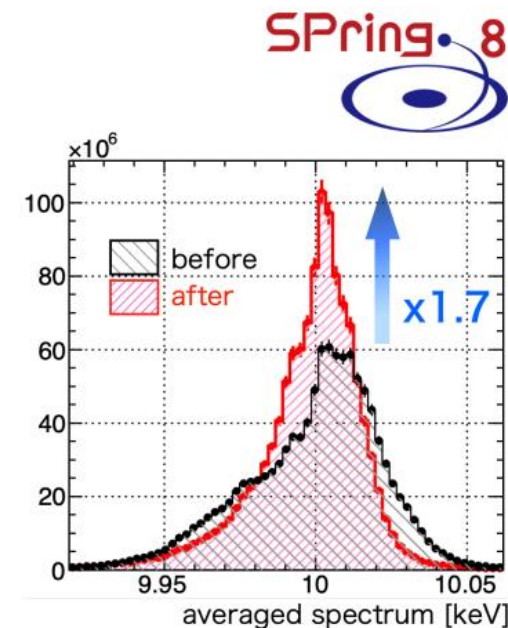
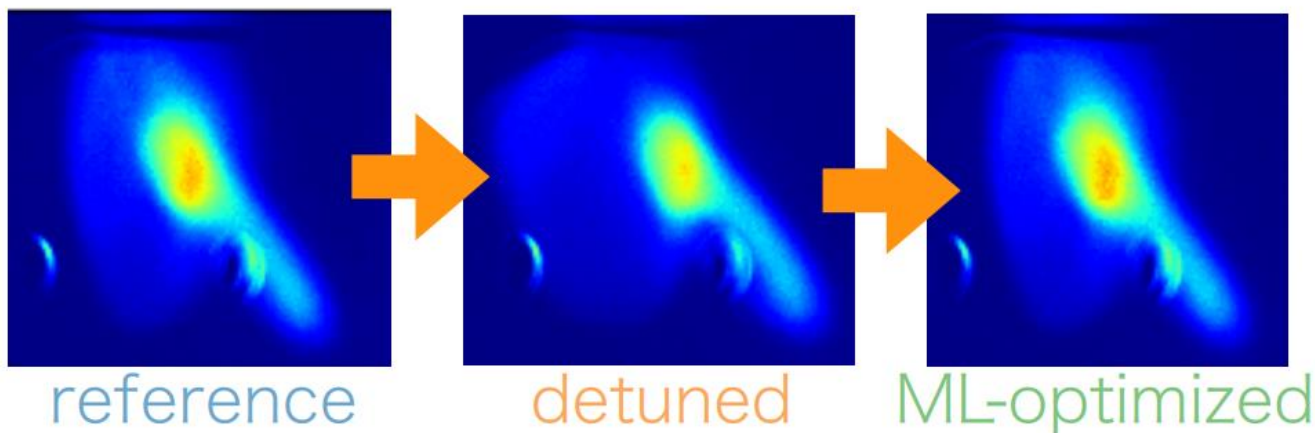
List of range, wait-time and control message for each device





# Related projects

- ▶ Spectral brightness optimization
  - Monitor pulse-by-pulse spectral width in addition to center wavelength with a newly developed inline spectrometer (energy resolution: a few eV)
  - This scheme was tested, and the spectrometer is now installed permanently  
→ can be used for daily tuning
- ▶ Optimize/Reproduce spatial profiles on YAG/SM at the upstream section (for beam tuning after thermal gun cathode replacement)
  - Tested with a single screen
  - Goal is to evaluate multiple screen images (+  $\alpha$ ) at a time to reproduce 6D phase space
  - Update control system to enable to switch screens many times



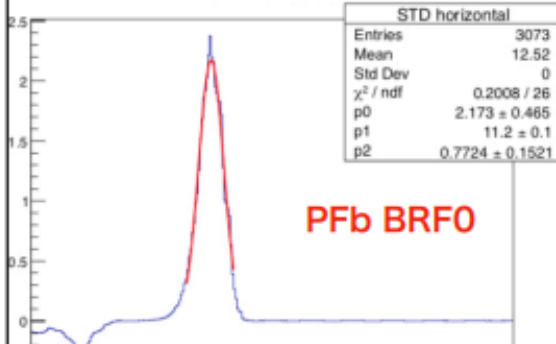
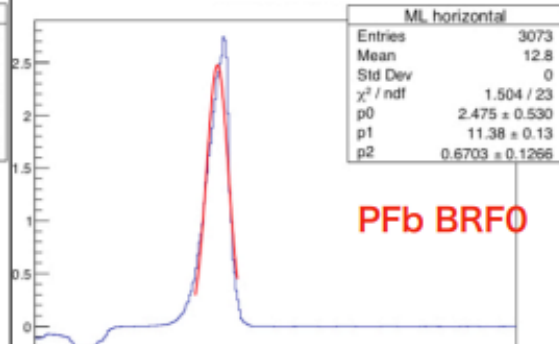




# 1st Exp. : Auto Tuning with Low Intensity Beam @RIKEN

Demonstration test at 2020 Oct. 21:00 ~ 9:00

Compare the result of manual optimization / manual + ML optimization  
using high intensity beams and wire scanner / Faraday cups

	Manual Optimization	Manual + ML Optimization
FC <sub>up</sub> [e $\mu$ A]	7.20	7.8
Beam Dump [e $\mu$ A]	7.25	8.0
Ratio (BD/FC <sub>up</sub> )※	1.01	1.03
Wire Scanner の像		

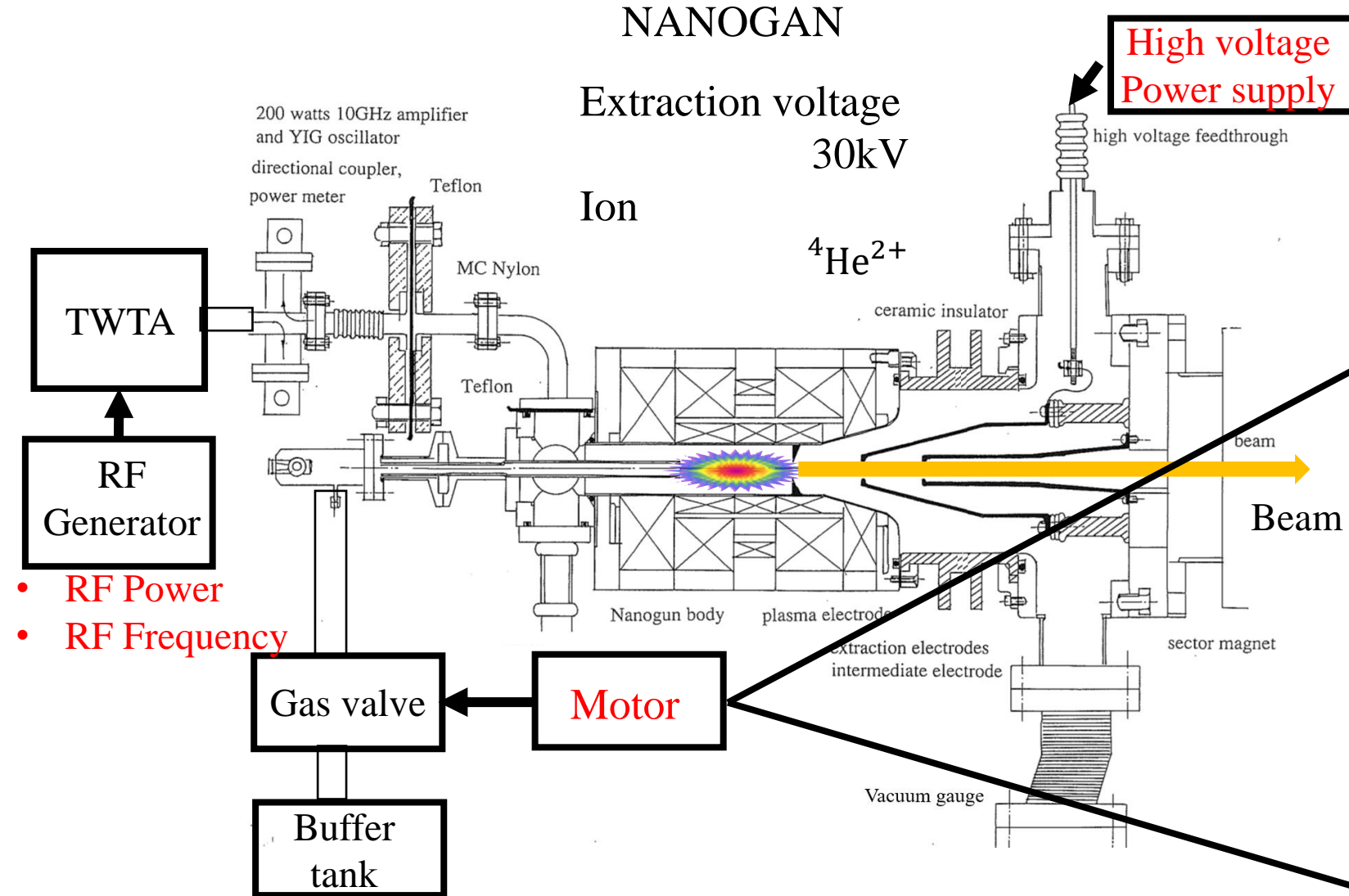
- Transmission ↗ 2%
- Beam width ↘ 13%

Significant improvements  
in the 1st test

※ Beam dump does not have  
good suppressor

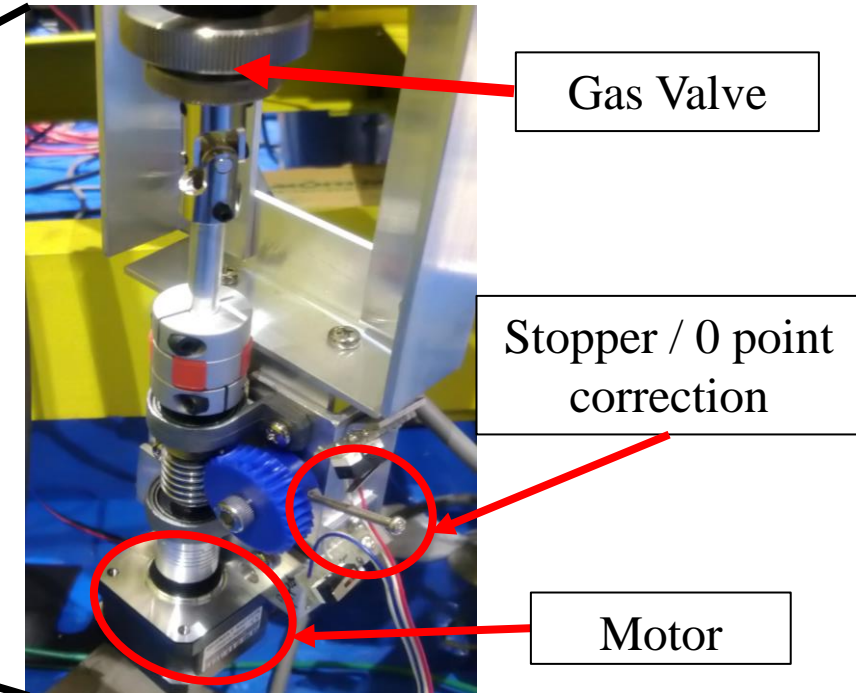
- Algorithm works as expected
- Optics is improved with low intensity beam (limit by fluorescent viewer: 0.001 enA)
- △ Effective algorithm

# 10 GHz ECR Ion Source 'NANOCHAN'



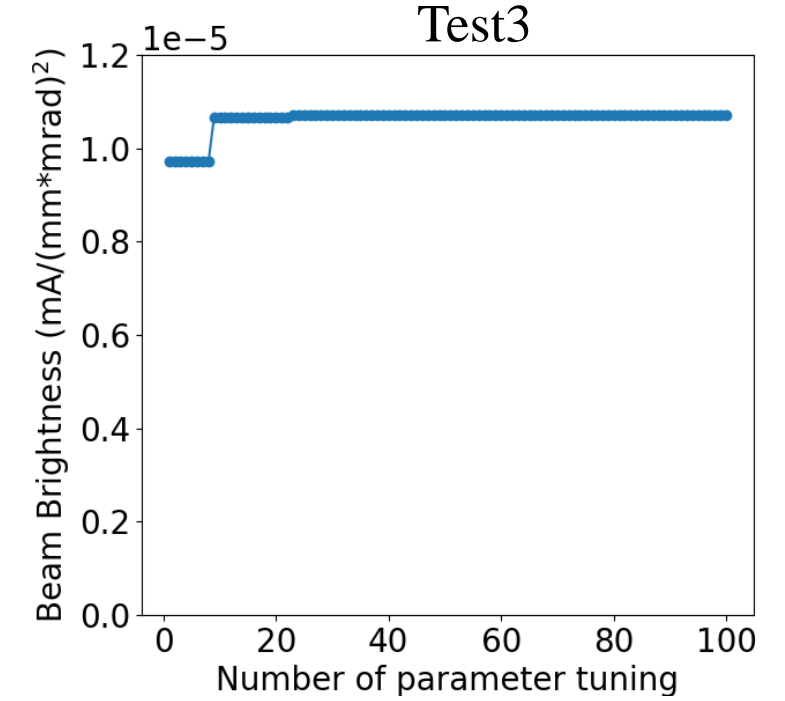
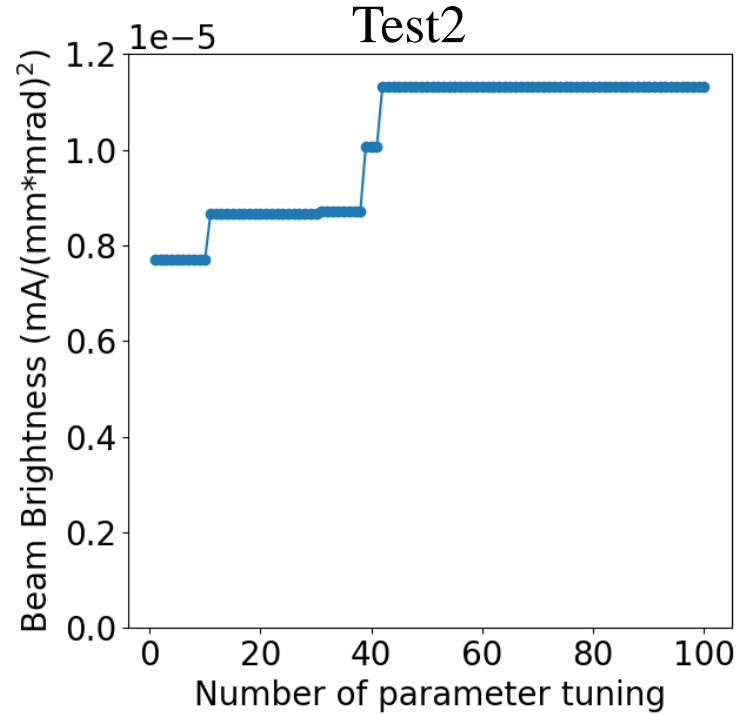
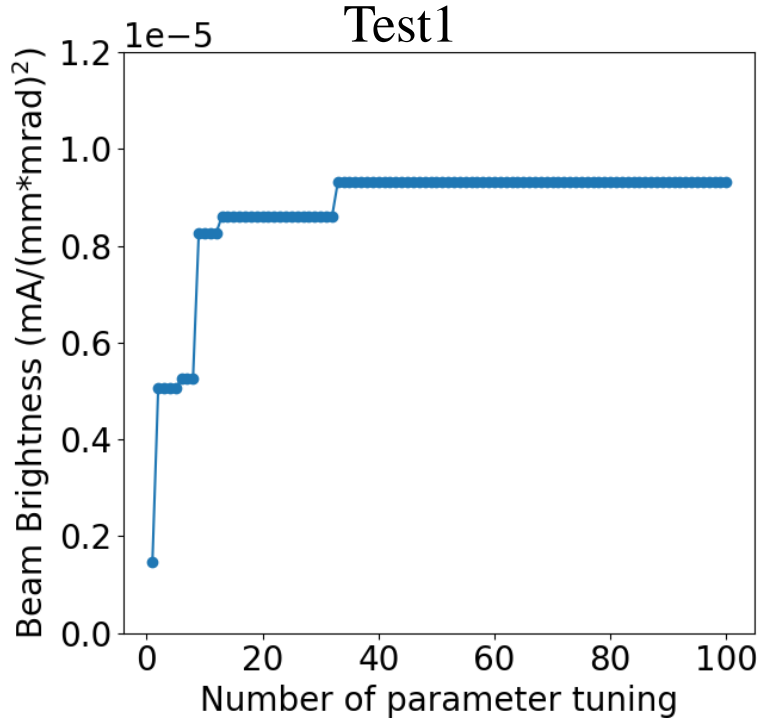
How to tune gas valves

1. Needle valve knob is connected to motor.
2. Tune the gas valve by controlling the amount of rotation of the motor.





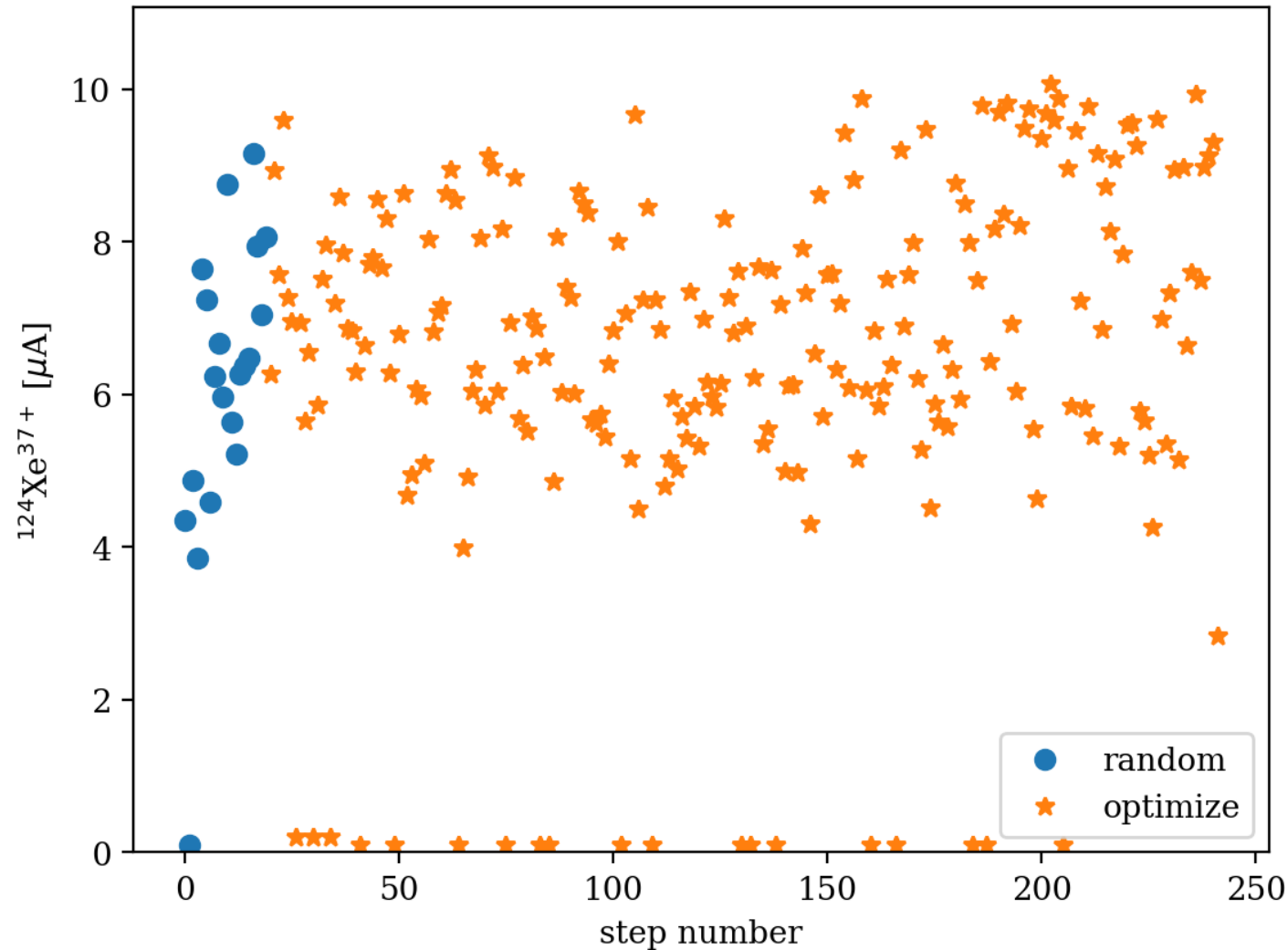
# Result : Ion Source tuning



	RF Power [dBm]	RF Frequency [GHz]	Gas valve [steps]	Intermediate electrode [kV]	Beam intensity [mA]	x-x' plane emittance [ $\pi$ mm mrad]	y-y' plane emittance [ $\pi$ mm mrad]	Beam Brightness [mA/(mm mrad) <sup>2</sup> ]
Test1	-13.4	10.03	11000	12.67	0.400	188	229	$9.32 \times 10^{-6}$
Test2	-14.2	9.95	11000	13.68	0.459	179	226	$1.13 \times 10^{-5}$
Test3	-13.8	9.98	11700	13.27	0.521	206	236	$1.07 \times 10^{-5}$

Good tuning in a short time

# Optimizing a little more like a human



Parameter	Min	Max
Bias voltage [V]	25	65
Oxygen valve	11.5	12.0
Xenon valve	9.0	12.0
18 GHz [kW]	1.20	1.80

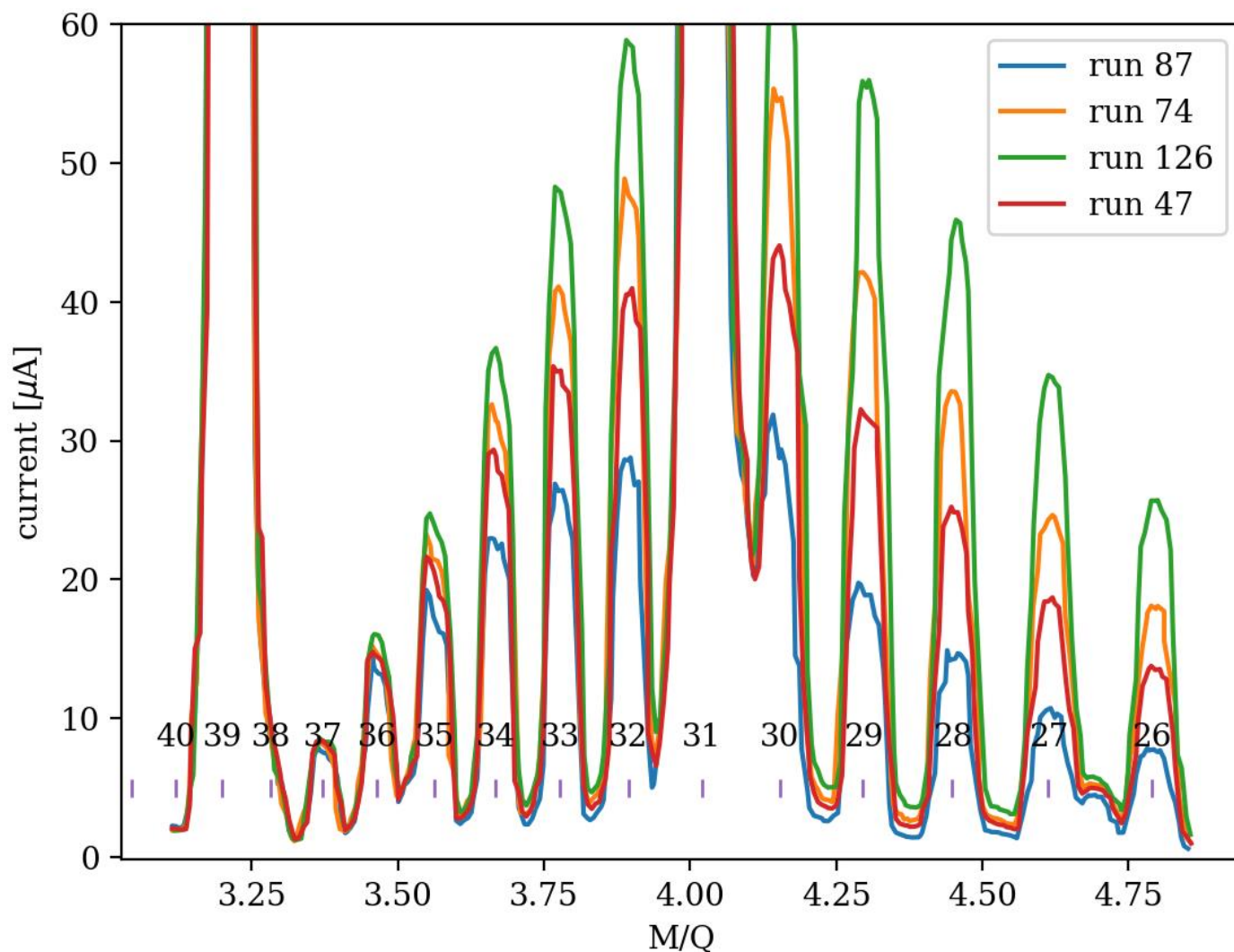
**Many human records are achieved by using a cost-function-like approach:**

- Coils are slow, so find a pretty good solution and work from there

**Note:**

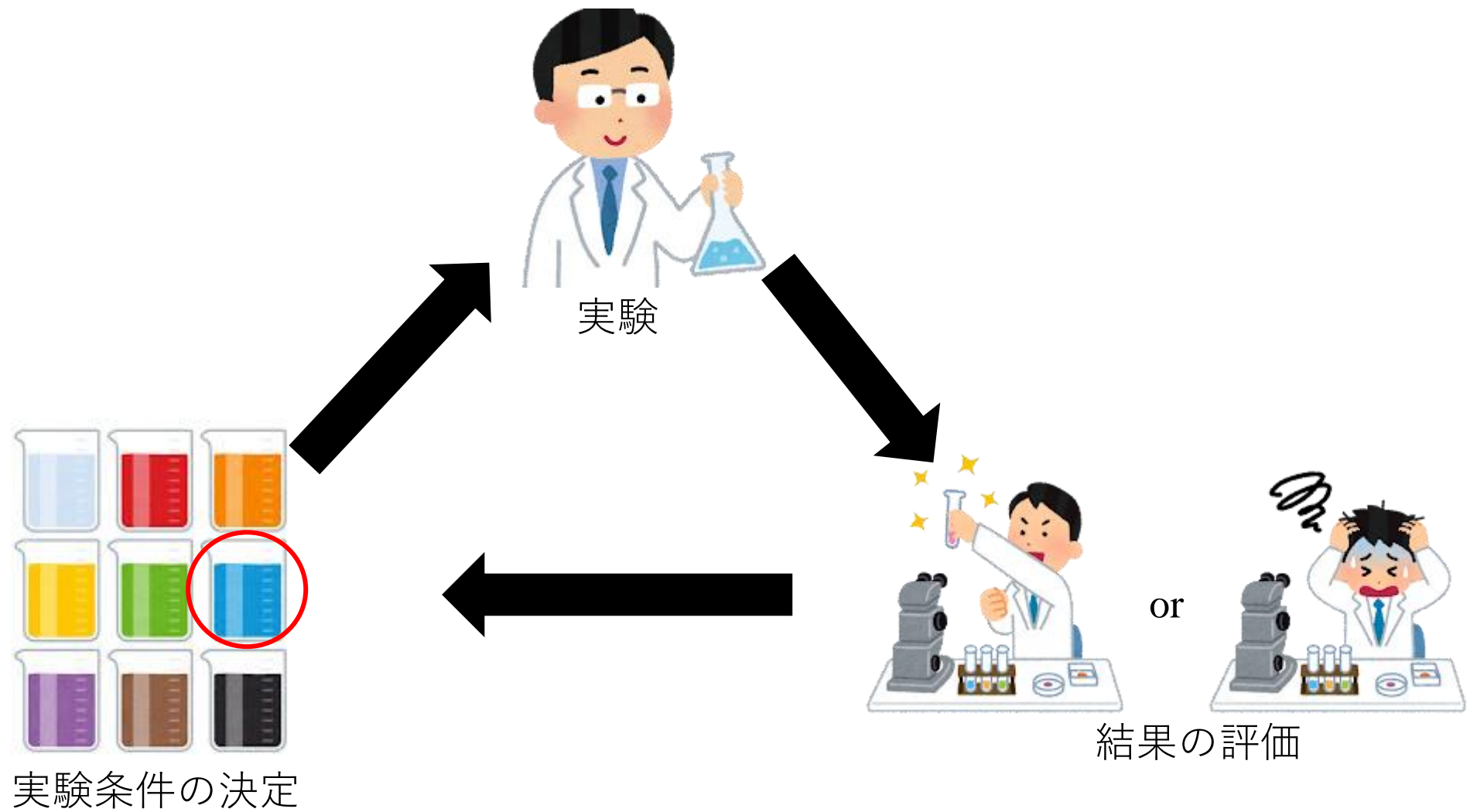
When instabilities or too low pressures encountered, low current is recorded

# Exploiting patience of computer



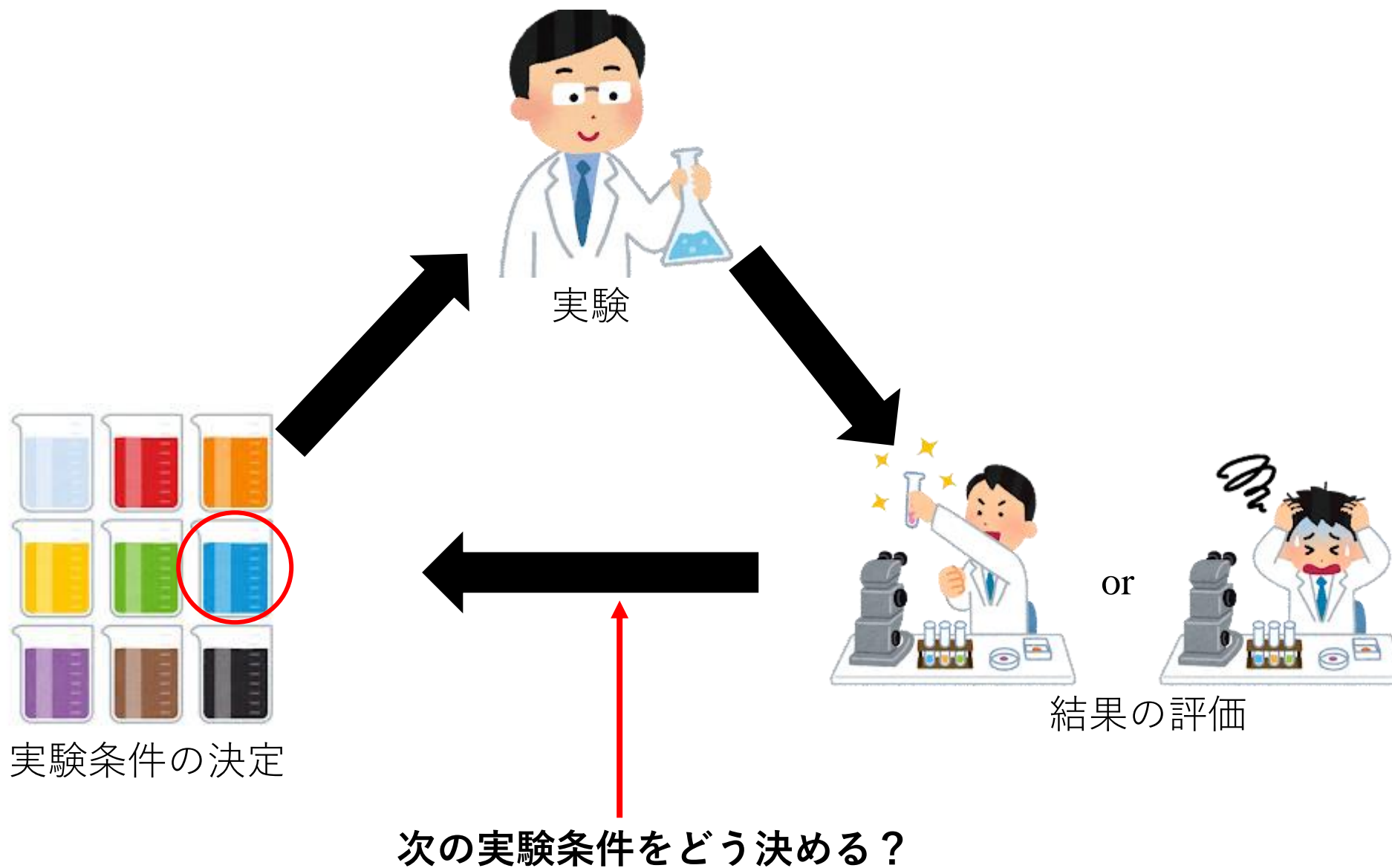
- ~5 minutes between optimization as computer performed charge state distribution (CSD) measurement at each step (~1-2 minutes)
- Most human optimizations are missing CSD information
- These four runs had almost identical  $^{124}\text{Xe}^{37+}$  currents but their CSDs were dramatically different
- This information will be fed into neural network efforts for general plasma understanding

# ベイズ最適化のイメージ





# ベイズ最適化のイメージ



# ベイズ最適化

Q. ブラックボックスに対して、最適なパラメーターを探すにはどうすればよいのだろうか。

- 案1. 今ある中でbestなパラメーター周辺を探す
- 案2. パラメーターを片っ端からしらみつぶしに探す
- 案3. 分布を予測し、予測に従って探す
- 案4. 経験則に基づいた勘

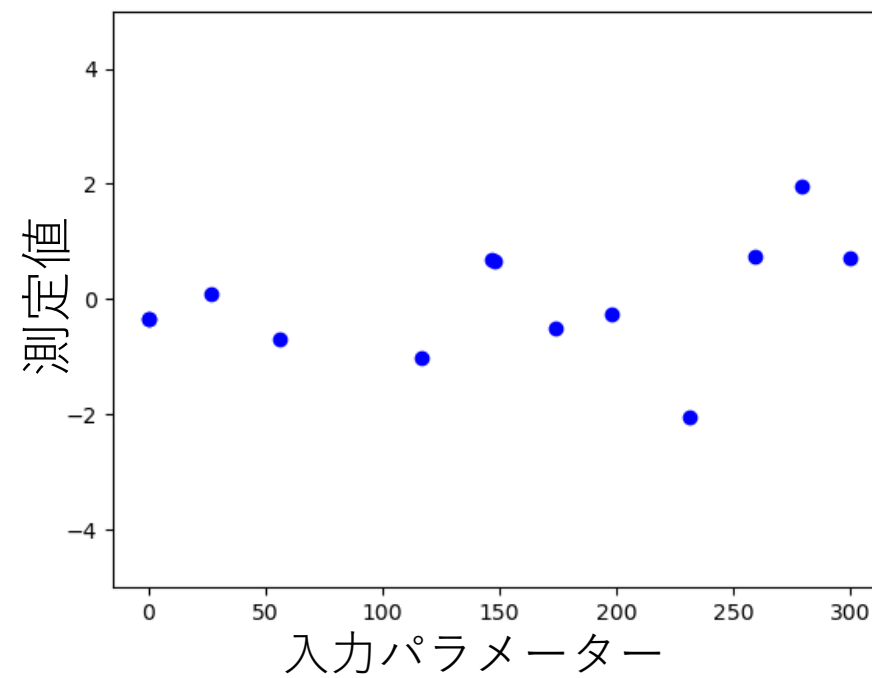
- 普通の人なら案1か案4 ?  
パラメーター数が少ないなら案2も不可能ではない
- 機械に'勘'は存在しないので、案1か案3がほとんど

ベイズ最適化は

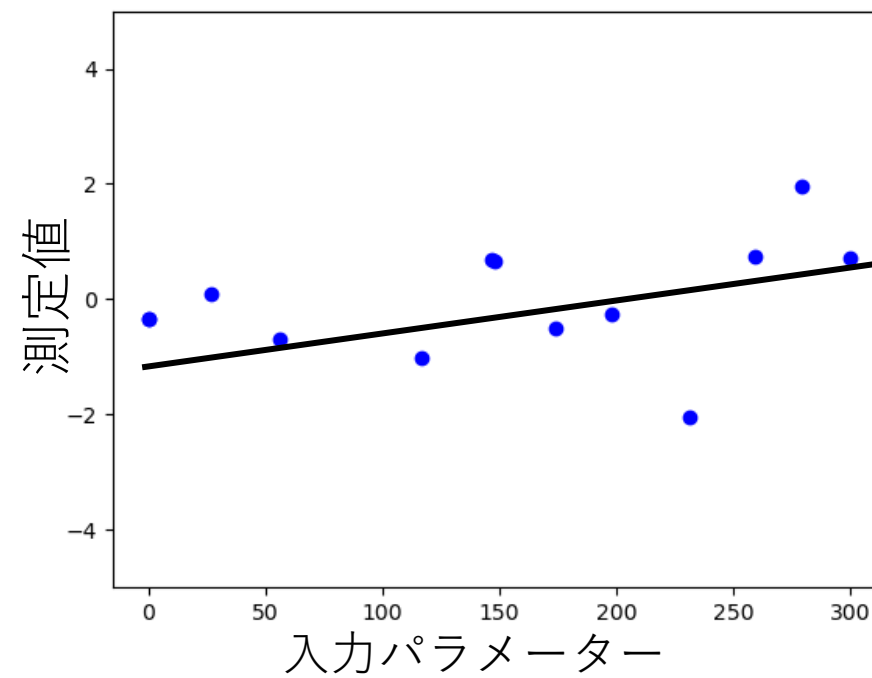
**データに合う関数を予測することで、最適解を効率的に探す手法(案3)**

- ・ 関数の予測
- ・ 予測した関数が最小(最大)となるパラメーターの予測

# 関数予測

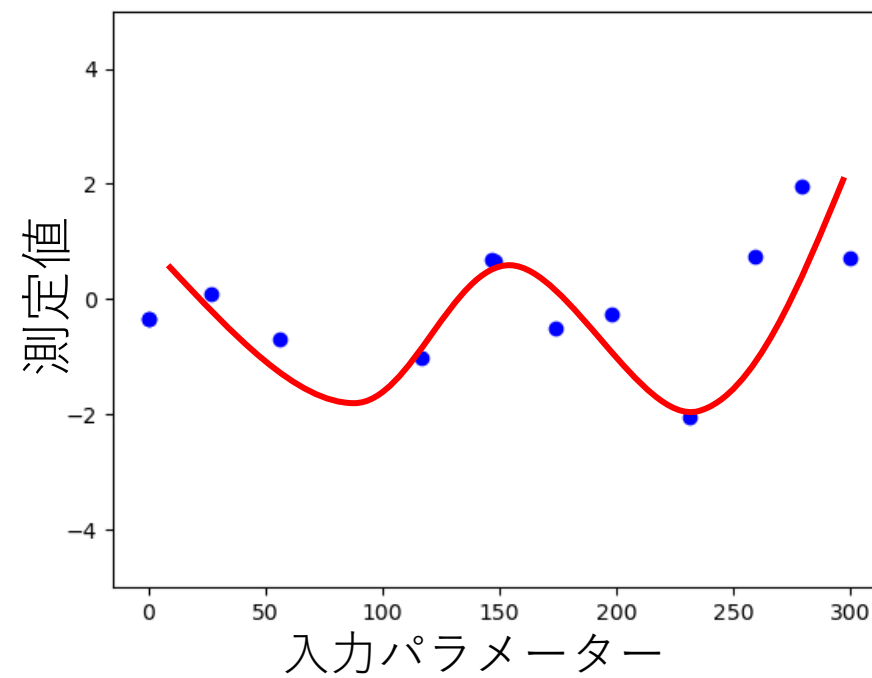


# 関数予測

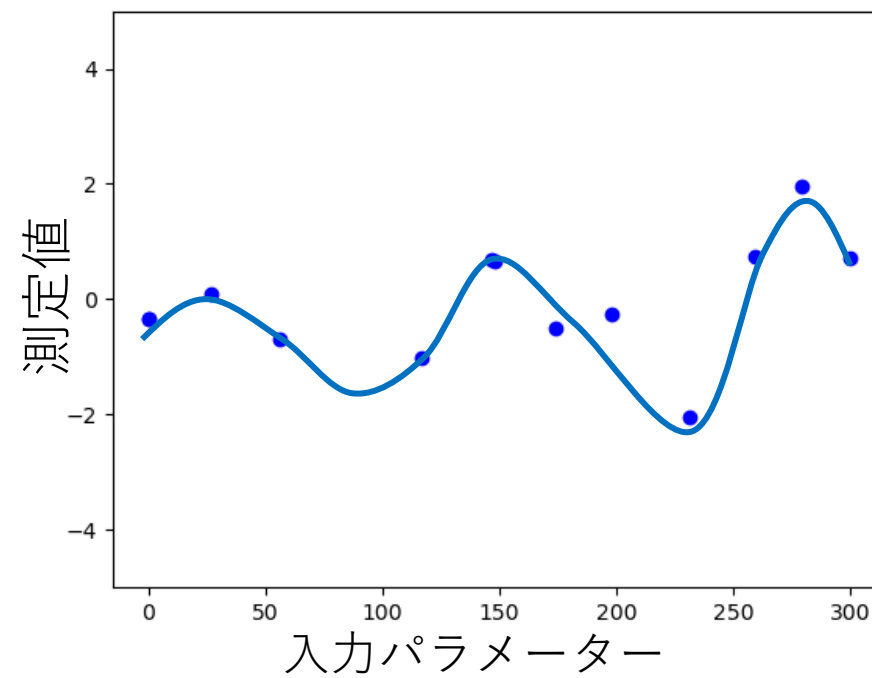




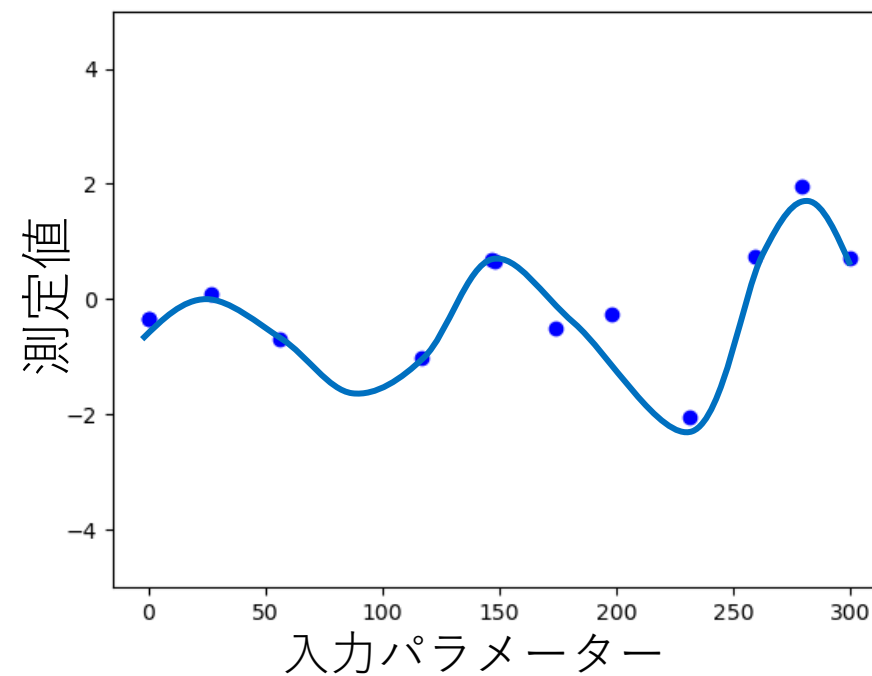
# 関数予測



# 関数予測



# 関数予測



ベイズ最適化では'ガウス過程回帰'を使って関数を予測する

# 理論：ガウス過程回帰

数学的な話に関しては**プログラムを書く上で必要最低限**な部分を説明します。  
(プログラムを書く際に触れないものに関しては省略or概念的な話のみ)

より深く勉強したい方には講談社より出版されている  
『**ガウス過程と機械学習 (機械学習プロフェッショナルシリーズ)**』  
をお勧めします。(¥3,300)

## Amazonリンク

[https://www.amazon.co.jp/%E3%82%AC%E3%82%A6%E3%82%B9%E9%81%8E%E7%A8%8B%E3%81%A8%E6%A9%9F%E6%A2%B0%E5%AD%A6%E7%BF%92%E6%A9%9F%E6%A2%B0%E5%AD%A6%E7%BF%92%E3%83%97%E3%83%AD%E3%83%95%E3%82%A7%E3%83%83%E3%82%B7%E3%83%A7%E3%83%8A%E3%83%AB%E3%82%B7%E3%83%AA%E3%83%BC%E3%82%BA%E6%8C%81%E6%A9%8B%E5%A4%A7%E5%9C%B0ebook/dp/B07QMMJJV8/ref=tmm\\_kin\\_swatch\\_0?encoding=UTF8&qid=1700124131&sr=8-1](https://www.amazon.co.jp/%E3%82%AC%E3%82%A6%E3%82%B9%E9%81%8E%E7%A8%8B%E3%81%A8%E6%A9%9F%E6%A2%B0%E5%AD%A6%E7%BF%92%E6%A9%9F%E6%A2%B0%E5%AD%A6%E7%BF%92%E3%83%97%E3%83%AD%E3%83%95%E3%82%A7%E3%83%83%E3%82%B7%E3%83%A7%E3%83%8A%E3%83%AB%E3%82%B7%E3%83%AA%E3%83%BC%E3%82%BA%E6%8C%81%E6%A9%8B%E5%A4%A7%E5%9C%B0ebook/dp/B07QMMJJV8/ref=tmm_kin_swatch_0?encoding=UTF8&qid=1700124131&sr=8-1)

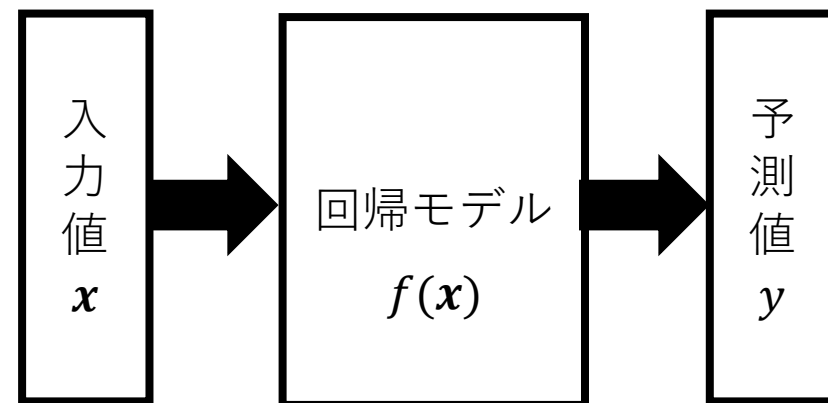




## 理論：回帰分析

### Q. 回帰とは？

A. 連続した入力値に対し、対応するデータを予測すること



**$f(x)$  がどんな関数かわからない**

→表現力の高い関数を使って、何かしら上手くいけばいい

例

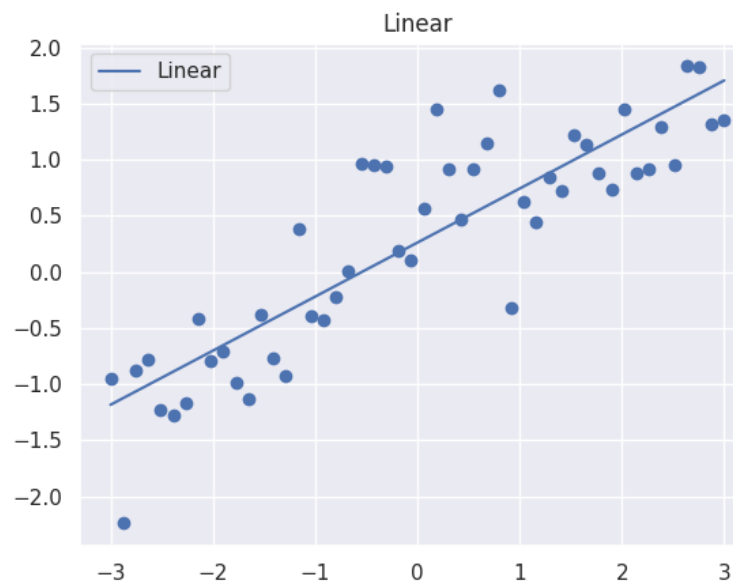
- 線形回帰
- リッジ回帰
- ガウス過程回帰

⋮

回帰分析の手法はたくさんある

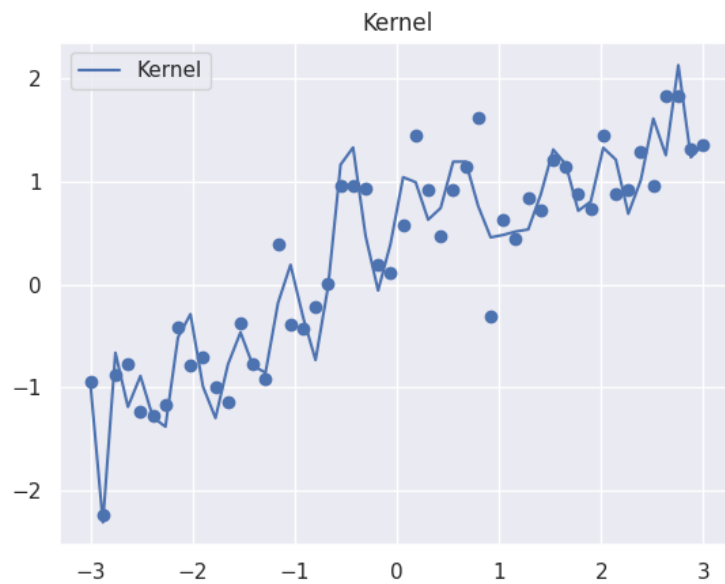
# 理論：回帰分析

- 線形回帰分析の例



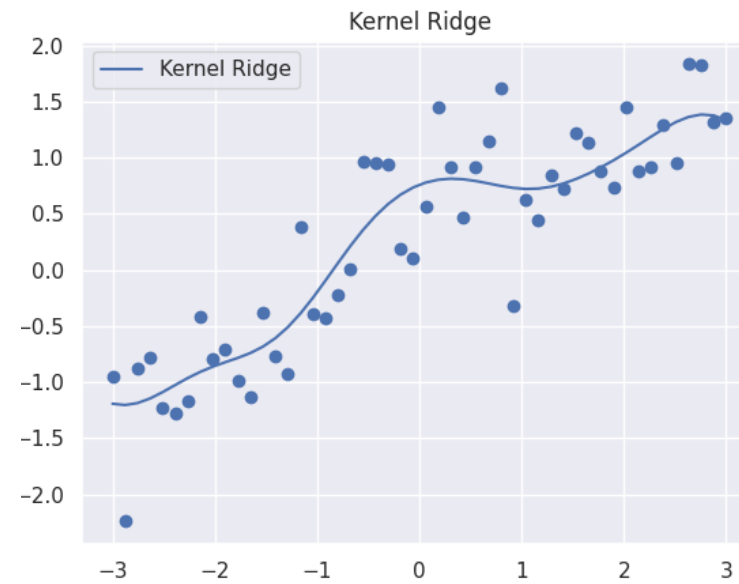
線形回帰

- 表現力は低い
- 非常にシンプル



カーネル回帰

- 表現力が過剰
- 過学習になりやすい



カーネルリッジ回帰

- 表現力は高い
- 表現力に制限をかけ、過学習を防ぐ工夫

# 理論：ガウス過程回帰

## 回帰分析はあくまで予測

近しい値になることが多いが、完全に一致はしないし、外れることもある

➤ 予測値だけでなく、予測の確からしさも知りたい！

そこでガウス過程回帰で回帰分析を行う

- 似た入力値がある ➔ 似た値を予測し、誤差は小さい
  - 似た入力値がない ➔ 予測の精度が下がる分、誤差は大きくなる
  - ガウス過程回帰
    1. ベイズ推定を用いて '**確率分布**'を予測するための推定手法
    2. 確率分布が正規分布になるという仮定と既知のデータから全体の確率分布を推定する
- 予測の確からしさを確率分布として表現可能

実装の課題は

非線形な事象を表現できる '高い表現力を持つ関数'を '少ない計算コスト'で実現すること

# 理論：カーネル法

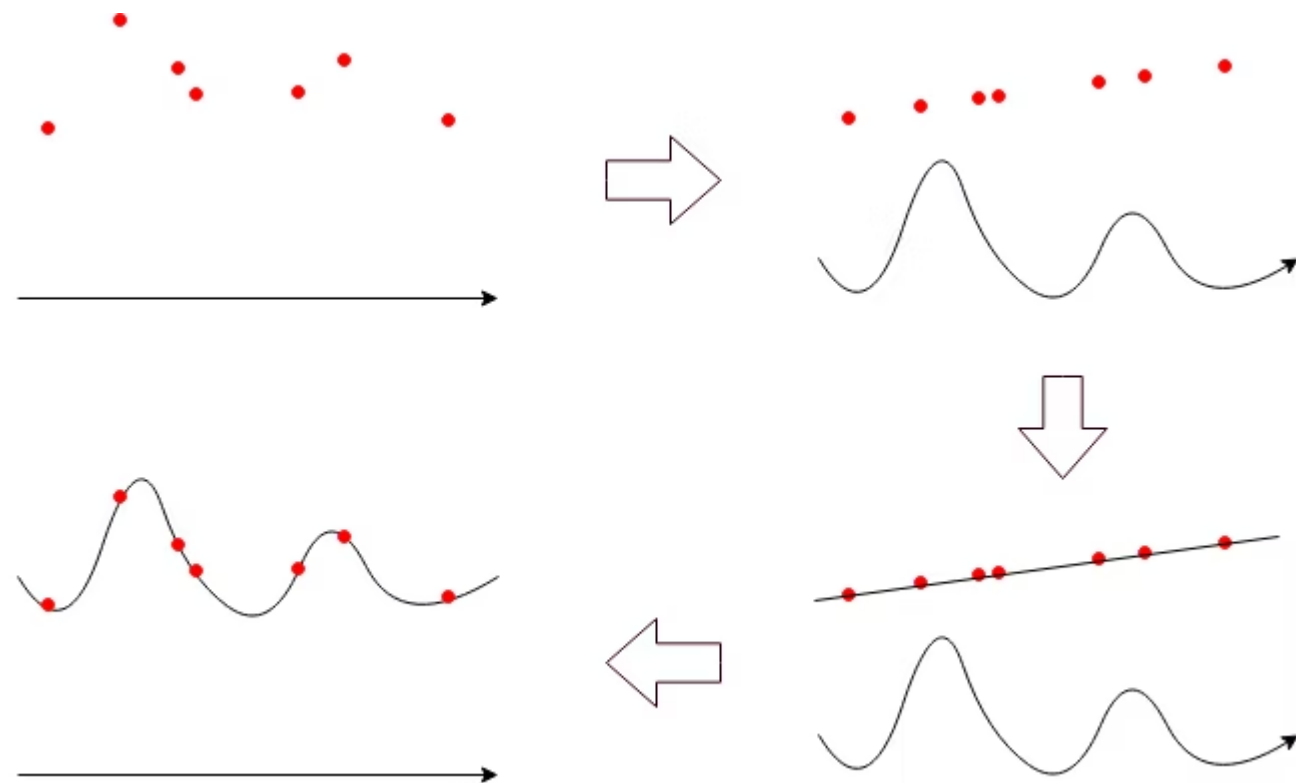
線形回帰では非線形なデータは推定できない

- 軸を曲げれば楽なのは？  
(特徴空間での回帰)

ただし特徴空間はベクトルの内積計算で算出

$$(f_1(\mathbf{x}), f_2(\mathbf{x}) \cdots, f_n(\mathbf{x})) (f_1(\mathbf{x}'), f_2(\mathbf{x}') \cdots, f_n(\mathbf{x}'))^T$$

- 表現力を高める = 計算コスト大



## カーネル法

内積を関数で表現することで表現力と計算コスト低減を両立する手法

- カーネル関数例

- 線形カーネル

$$k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T \mathbf{x}'$$

- ガウスカーネル

$$k(\mathbf{x}, \mathbf{x}') = \theta_1 \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{\theta_2}\right)$$

- 周期カーネル

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(\theta_1 \cos\left(\frac{\|\mathbf{x} - \mathbf{x}'\|}{\theta_2}\right)\right)$$

等

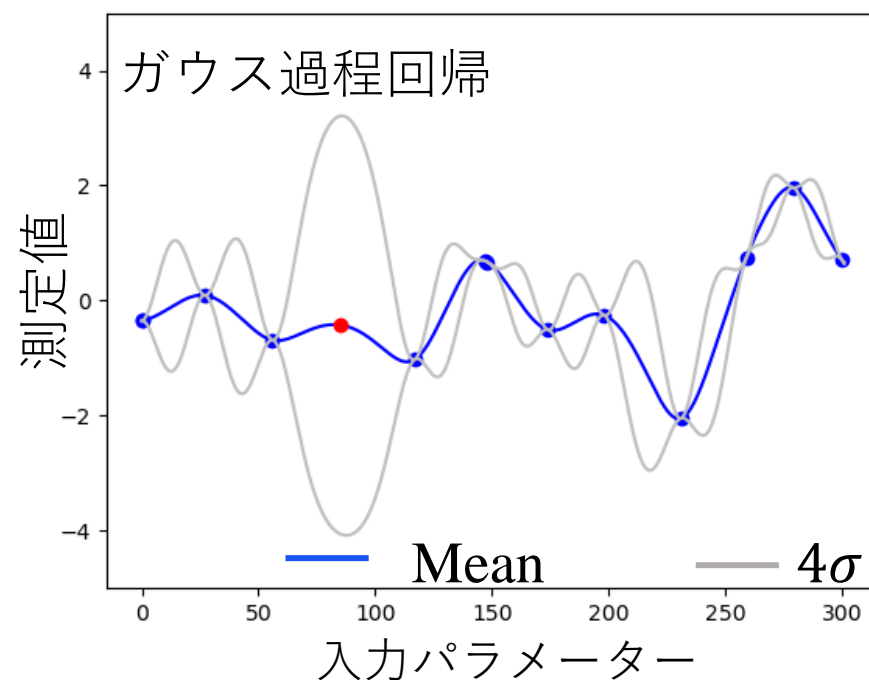
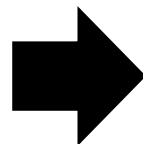
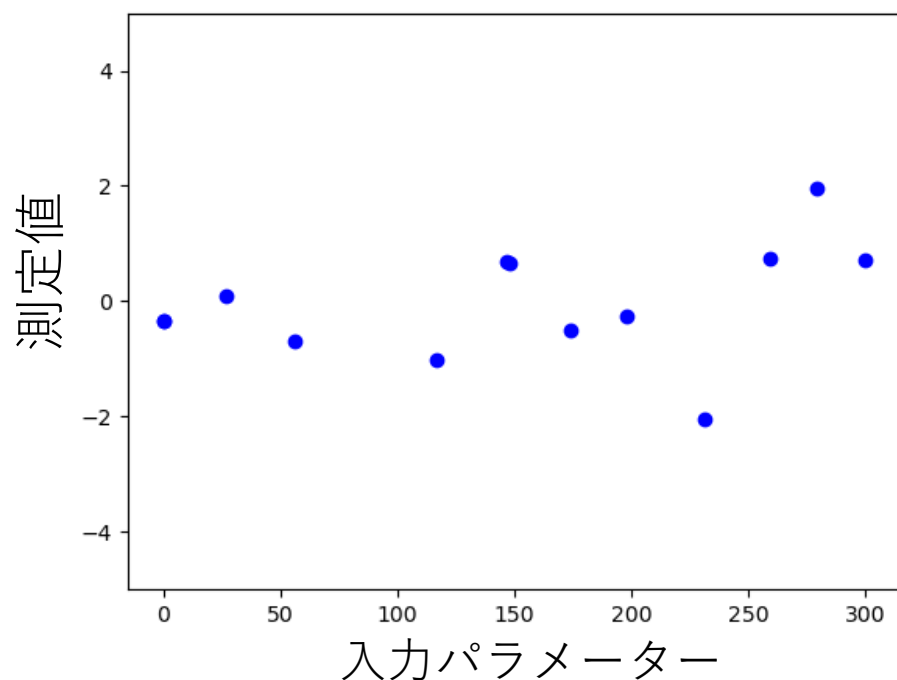


# 理論：ガウス過程回帰

ベイズの定理による確率分布予測 + カーネル法による計算コスト削減 → 効果的な回帰手法

- ガウス過程回帰は、**'確率分布が正規分布になる'**という仮定をでベイズの定理を活用

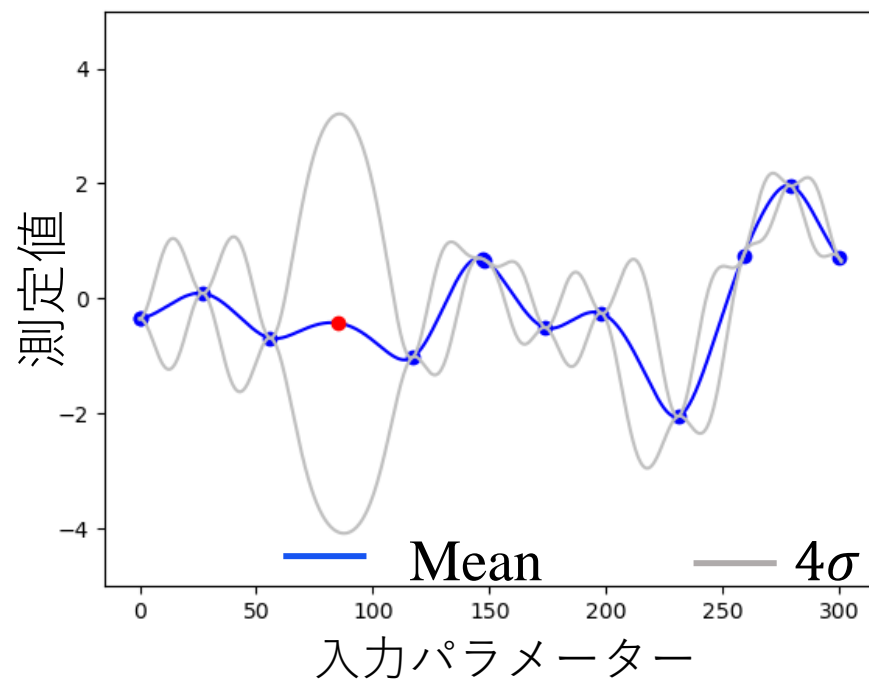
ガウスカーネルを使うからガウス過程回帰ではない



# 理論：ベイズ最適化

ガウス過程回帰によって'データがどんな関数に従っているのか'の予測ができる

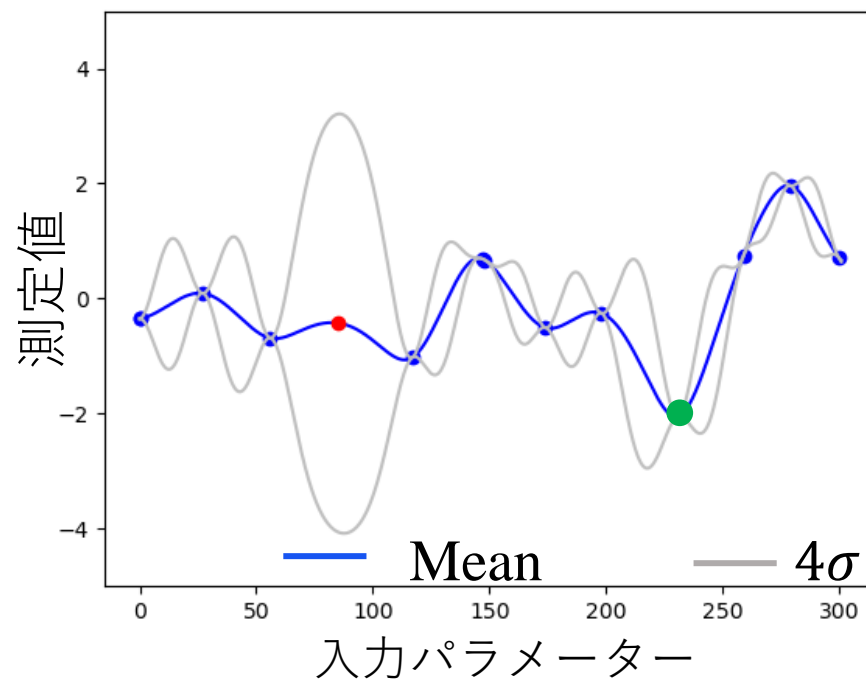
➤ '次にどこを探すのか'を決める手法が重要



# 理論：ベイズ最適化

ガウス過程回帰によって'データがどんな関数に従っているのか'の予測ができる

➤ '次にどこを探すのか'を決める手法が重要

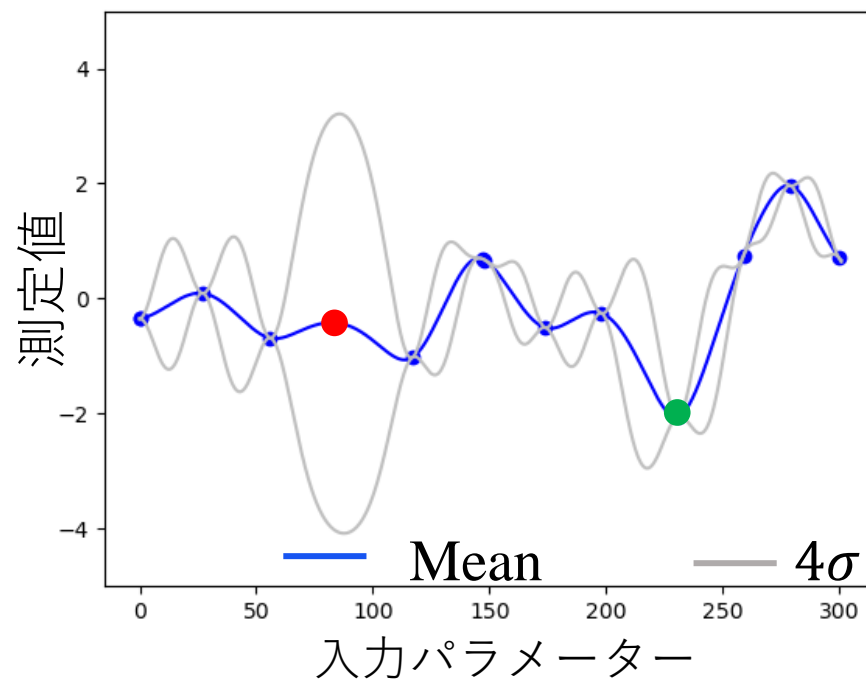


予測値を信じる(活用重視)なら ●

# 理論：ベイズ最適化

ガウス過程回帰によって'データがどんな関数に従っているのか'の予測ができる

➤ '次にどこを探すのか'を決める手法が重要

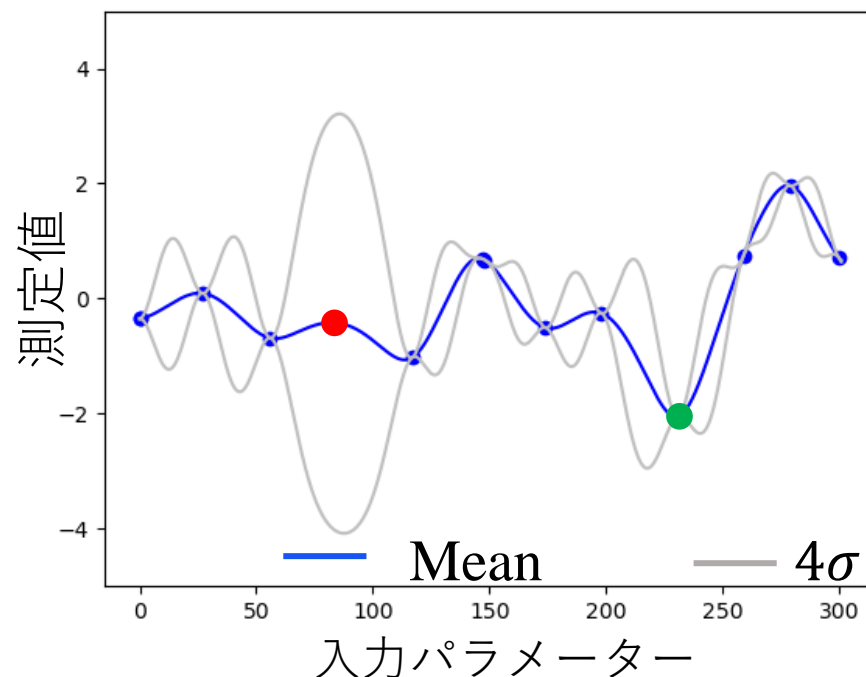


予測値を信じる(活用重視)なら ●  
可能性を信じる(探索重視)なら ●

# 理論：ベイズ最適化

ガウス過程回帰によって'データがどんな関数に従っているのか'の予測ができる

➤ '次にどこを探すのか'を決める手法が重要



予測値を信じる(活用重視)なら ●  
可能性を信じる(探索重視)なら ●

次にどこを探すかを決定する関数を**獲得関数(Acquisition function)**と呼ぶ

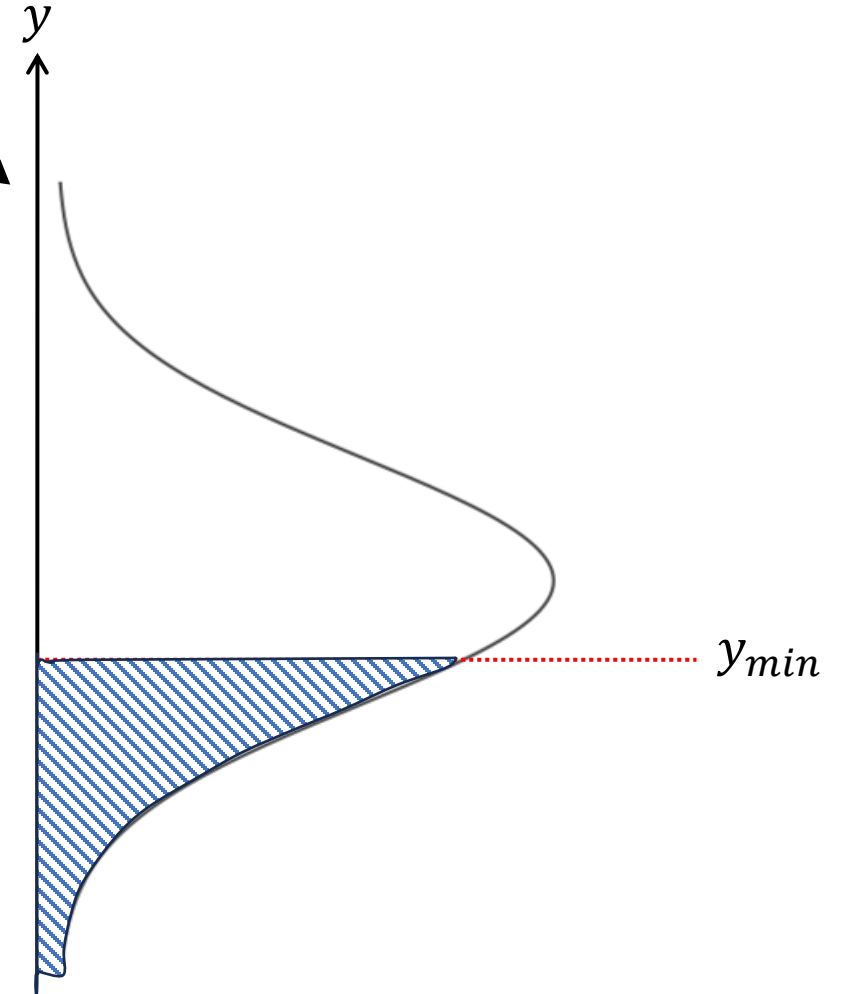
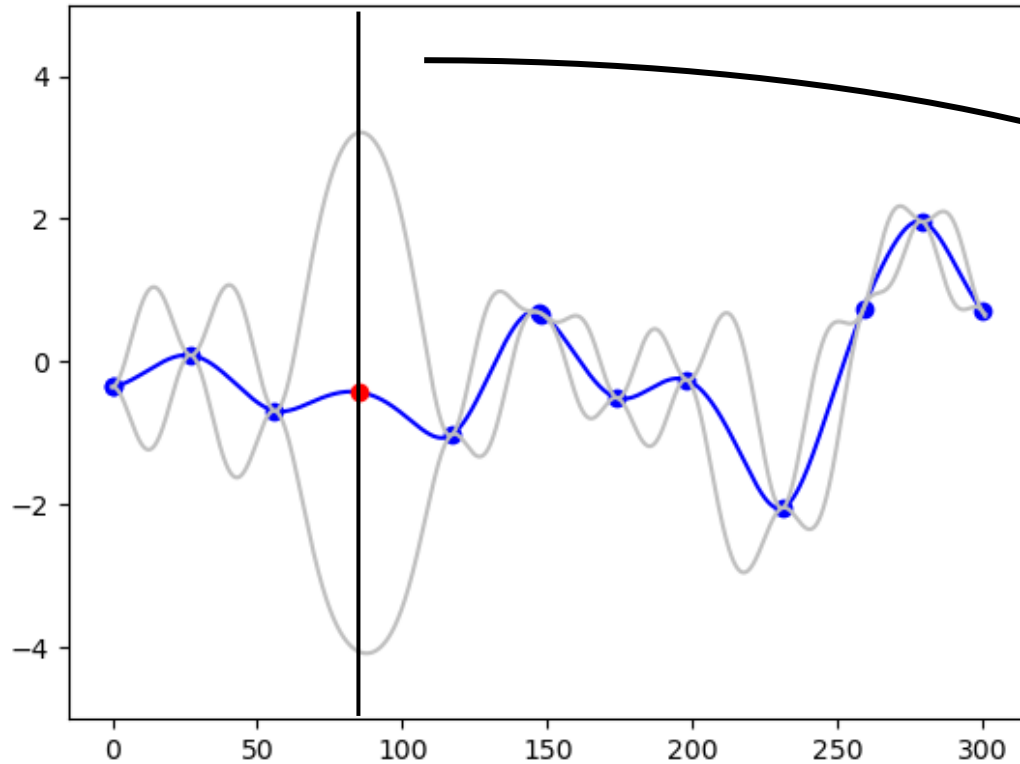
# 理論：ベイズ最適化

## 獲得関数一例

- **Probability of Improvement (PI)**
  - ‘改善する確率’が最も高いパラメーターを次の実験として選ぶ
  - 活用重視の傾向が強い
- **Expected Improvement (EI)**
  - ‘改善の期待値’が最も高いパラメーターを次の実験として選ぶ
  - PIと比べ、探索を行う傾向が強い(それでもまだ活用重視という意見もある)
- **Lower Confidence Bound (LCB)                      or                      Upper Confidence Bound (UCB)**
  - ‘分散を含めた値’を判断基準として次の実験として選ぶ
  - この三つの中では最も探索を重視する
  - ハイパーパラメーター(人が決めるパラメーター)によって探索と活用のバランスを変えられる



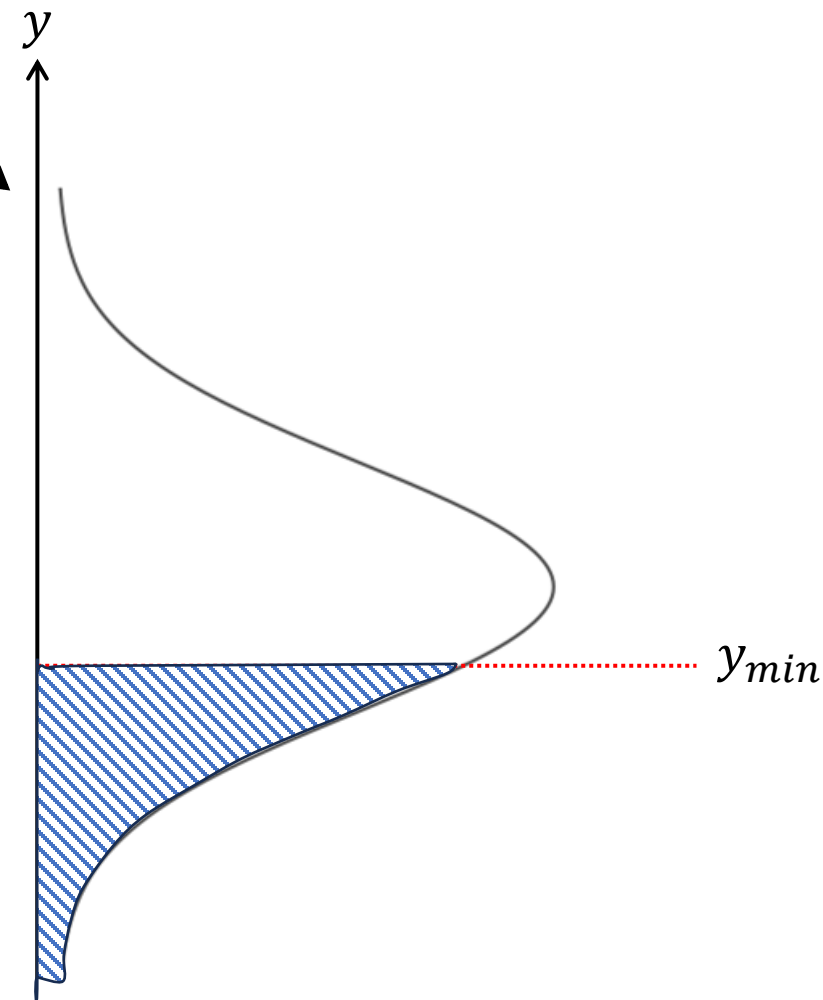
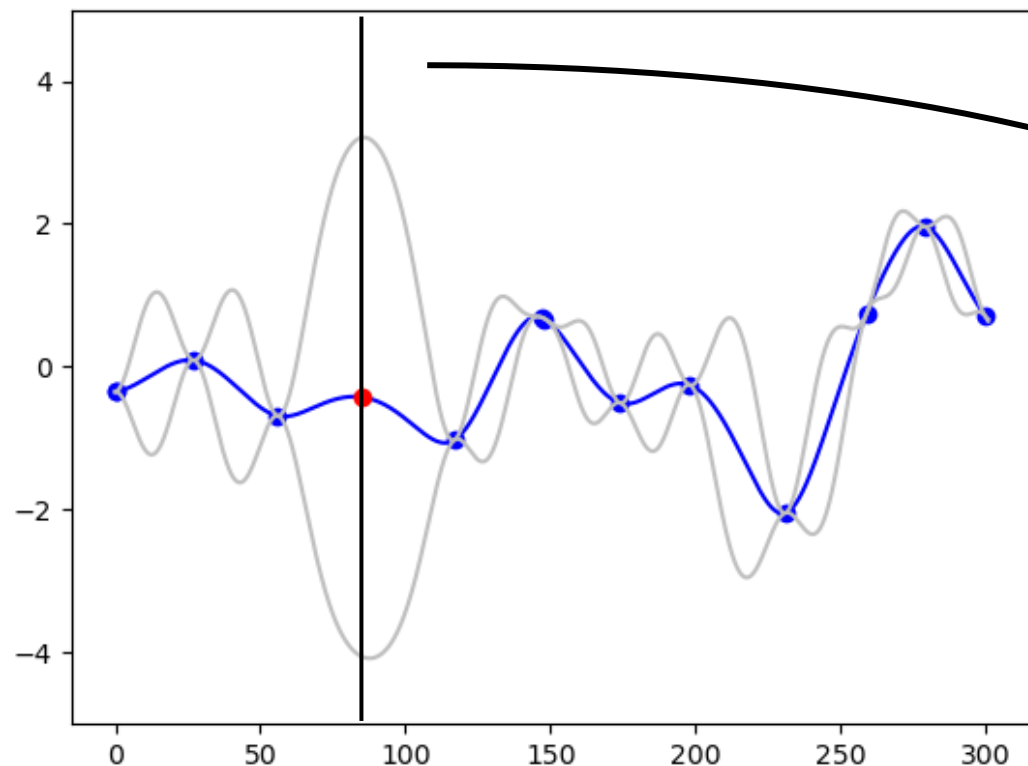
# Probability of Improvement (PI)



‘改善する確率’を指標に方針を決定する

$$\alpha_{PI} = \mathbb{P}[y < y_{min}]$$
$$y_{min} = \min(y_1, y_2, \dots, y_n)$$

# Expected Improvement (EI)



‘改善の期待値’を基準に方針を決める

$$\alpha_{PI} = \mathbb{E}[(y_{min} - y) \times \mathbb{I}(y < y_{min})]$$

$$y_{min} = \min(y_1, y_2, \dots, y_n)$$

$$\mathbb{I}(y < y_{min}) = \begin{cases} 0 & (y \geq y_{min}) \\ 1 & (y < y_{min}) \end{cases}$$

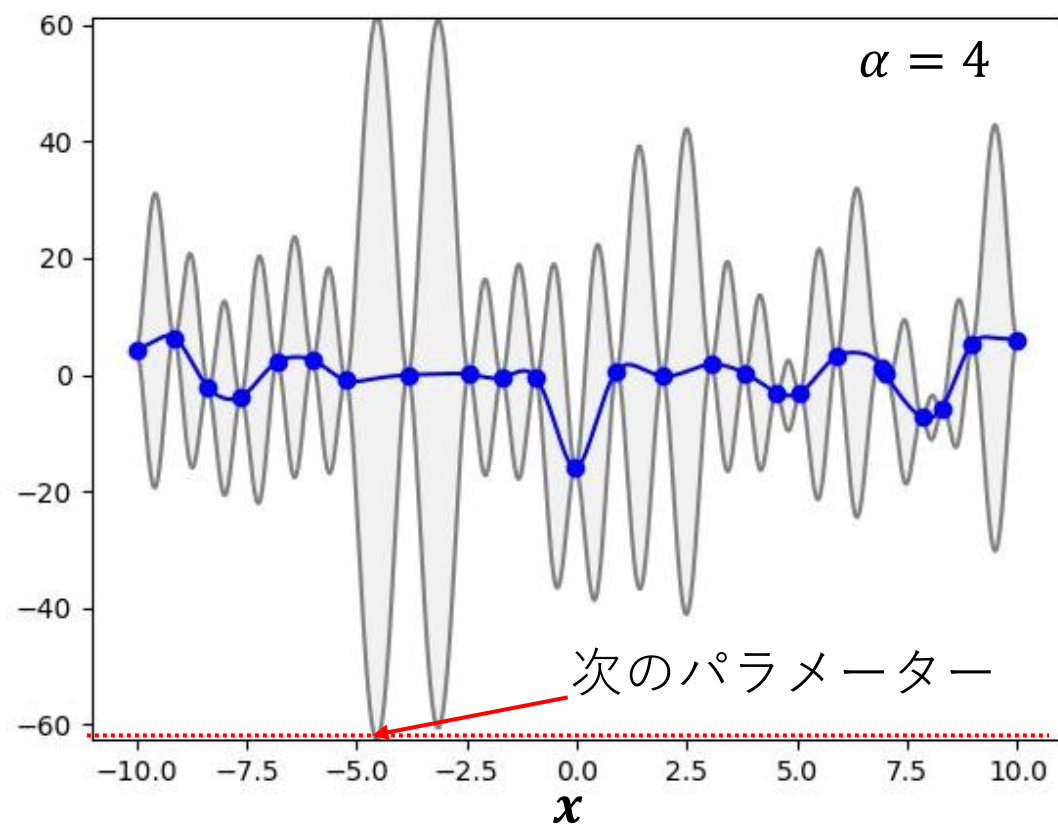
# Lower Confidence Bound (LCB)

‘分散を含めた値’を指標に方針を決める

$$\alpha_{LCB} = \underbrace{\mu(\mathbf{x})}_{\text{予測値}} - \underbrace{\alpha \times \sigma(\mathbf{x})}_{\text{分散}}$$

$\alpha$ 大: 探索重視

$\alpha$ 小: 活用重視



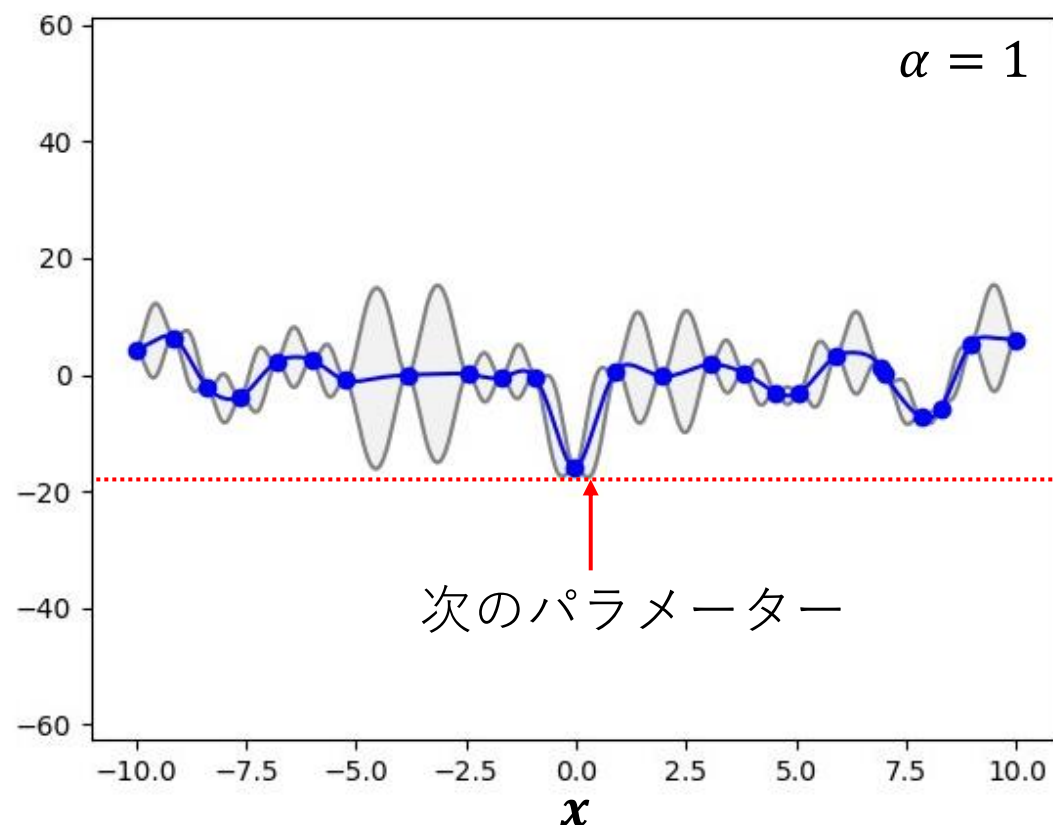
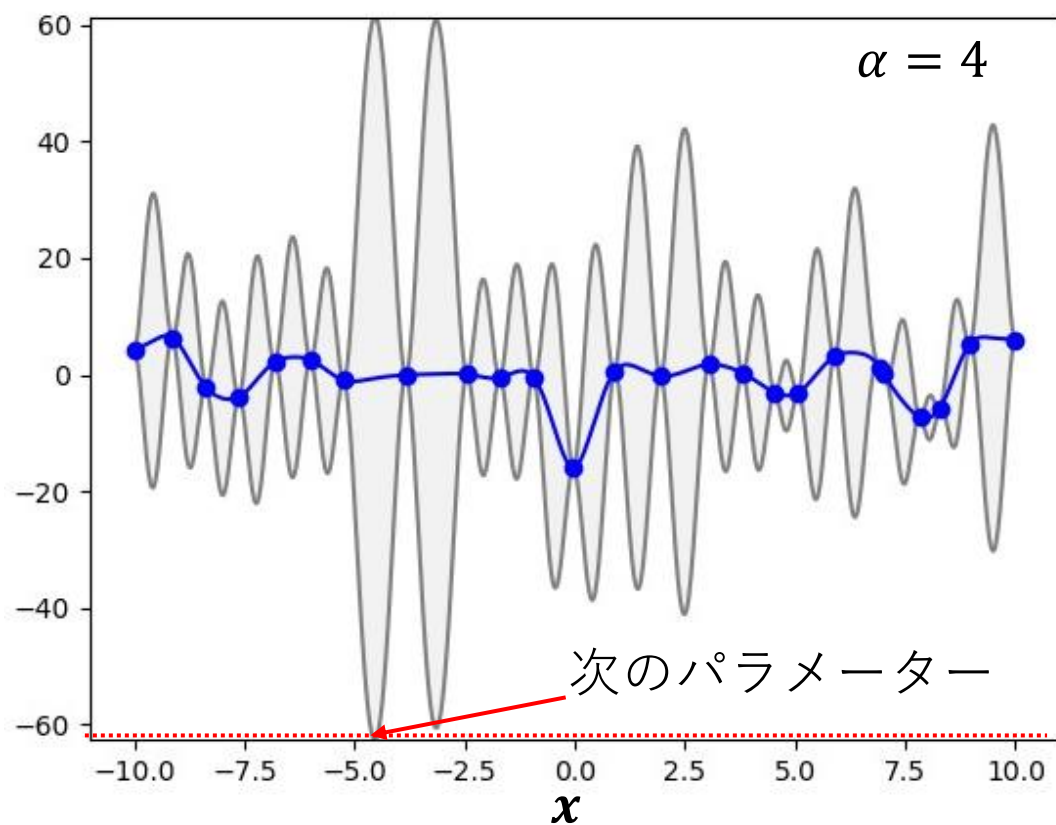
# Lower Confidence Bound (LCB)

‘分散を含めた値’を指標に方針を決める

$$\alpha_{LCB} = \underbrace{\mu(\mathbf{x})}_{\text{予測値}} - \underbrace{\alpha}_{\text{定数}} \times \underbrace{\sigma(\mathbf{x})}_{\text{分散}}$$

$\alpha$ 大: 探索重視

$\alpha$ 小: 活用重視



# 活用重視と探索重視の違い

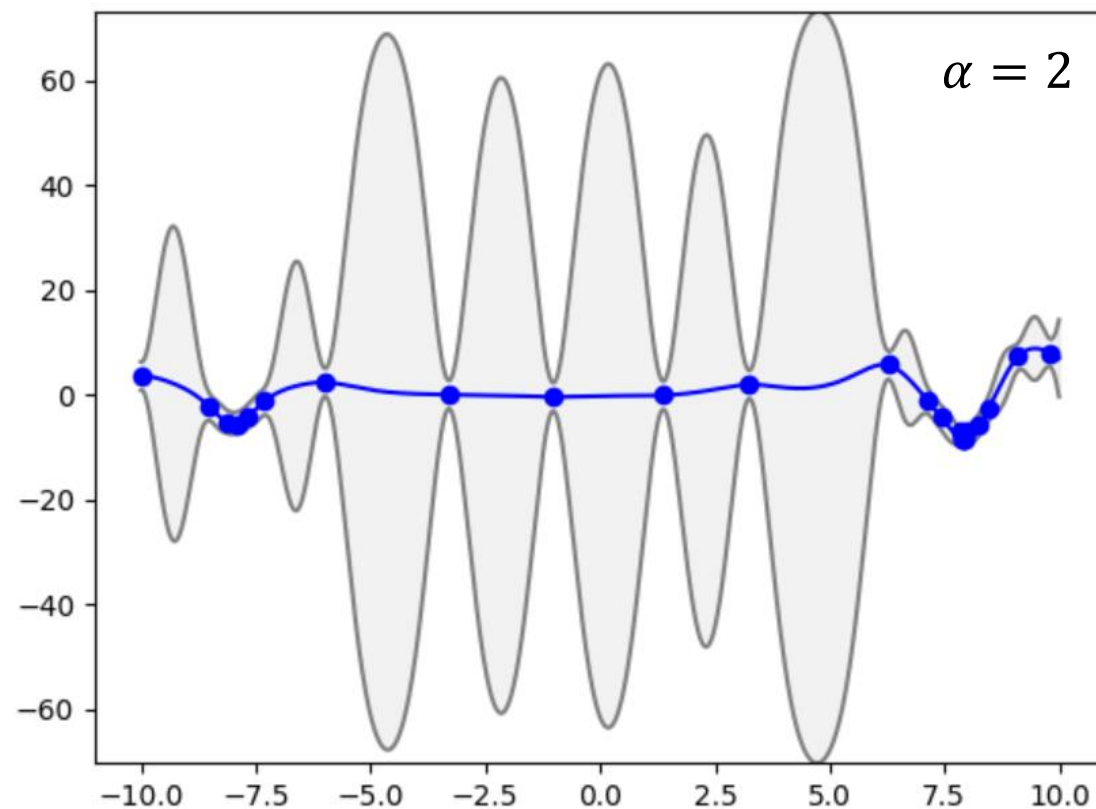
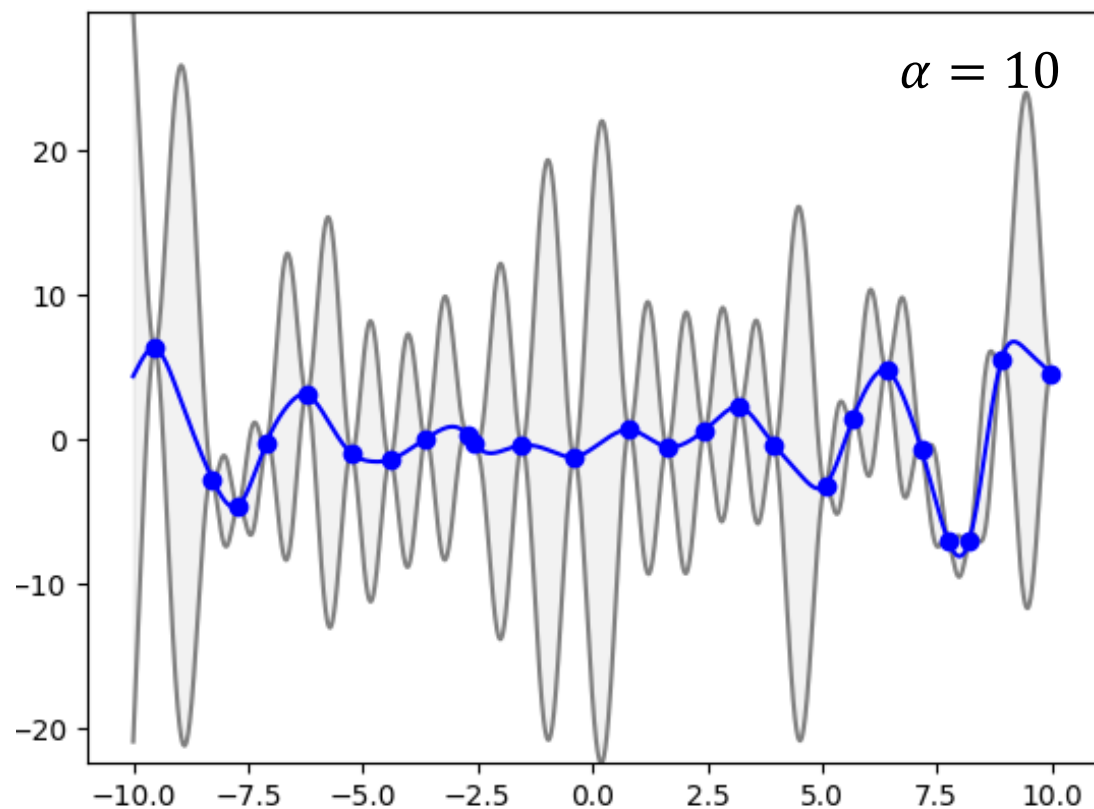
‘分散を含めた値’を指標に方針を決める

$$\alpha_{LCB} = \underbrace{\mu(\mathbf{x})}_{\text{予測値}} - \underbrace{\alpha}_{\text{定数}} \times \underbrace{\sigma(\mathbf{x})}_{\text{分散}}$$

$\alpha$ 大: 探索重視

$\alpha$ 小: 活用重視

$\alpha$ の値によってデータ取得は大きく異なる



## まとめ

- ベイズ最適化は未知の関数を近似し、最適解を効率的に探す手法
- 未知の関数は'**ガウス過程回帰**'によって'**確率分布**'として推測
- 予測した確率分布関数に対して'**獲得関数**'を用いて次の実験点を決める
  - 獲得関数には
    - **Probability of Improvement (PI)**
    - **Expected Improvement (EI)**
    - **Lower Confidence Bound (LCB)**
  - 等がある。
- 獲得関数によって活用重視(既知の領域を重視)や探索重視(未知の領域を重視)が異なる
  - PI, EI, LCBでは、PIが最も活用重視でLCBが最も探索重視になる。

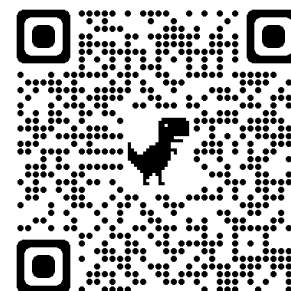
この後は実際にサンプルプログラムを使って実践してみましょう



# 各種ファイル関連

実行環境

Google Colaboratory  
<https://colab.google/>



今回使用するファイル

<https://github.com/Morita1116/Machine-Learning-Workshop-on-Accelerator-and-Beam-Physics-Tutorial-Materials>

Indico上にこのページをアップロードしております。

- |                     |                                     |
|---------------------|-------------------------------------|
| • Setup.txt         | Google Colaboratory上に環境を構築するためのコマンド |
| • BO_2parameters.py | プログラム例(2パラメーターの最適化)                 |
| • BO_4parameters.py | プログラム例(4パラメーターの最適化)                 |