ベイズ的最適化入門

佐藤　一誠*

Introduction of Bayesian Optimization

Issei Sato*

Key words: Bayesian Optimization, Experimental Design, Gaussian Process

1. はじめに

工学的な問題の多くは，何らかの目的関数の最適化を行う必要がある．ここでは，$\mathbb{R}$を実数として，コンパクト集合 $\mathcal{X} \subset \mathbb{R}^d$ 上の関数 $f(\boldsymbol{x})$ の最適化：

$$\boldsymbol{x}^* = \underset{\boldsymbol{x} \in \mathcal{X}}{\operatorname{argmax}}\, f(\boldsymbol{x}) \qquad (1)$$

を考えることにする．$f(\boldsymbol{x})$ か微分可能で更に上に凸な関数の場合は，任意の初期値 $\boldsymbol{x}^{(0)}$ から始めて勾配法などを用いれば，次に探索すべき $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots$ を次々に生成し最適な $\boldsymbol{x}^*$ を見つけることは容易である．

しかし，入力 $\boldsymbol{x}_t$ を与えることで出力 $y_t = f(\boldsymbol{x}_t)$ が得られるオラクルのみが存在し，$f(\boldsymbol{x})$ は非線形で多峰である可能性があり，更に出力を得るのが高コスト(時間がかかる．費用がかかるなど)である場合には，勾配法のような通常の最適化法を使うことができない．このような場合の最適化技術としてベイズ的最適化と呼ばれる方法が注目を集めている．ベイズ的最適化の歴史は古くは1960年代までさかのぼる[7,9]．しかし，近年この技術は様々な分野で活用されるようになってきた．

そもそもブラックボックス関数は，科学的な問題設定では常に存在する．例えば，化学実験において3つの試薬の分量 (x_1, x_2, x_3) に対して何らかの化学反応の強さを数値化したものを y とすれば，目的とする y を得るための分量 (x_1, x_2, x_3) を模索する作業はブラックボックス関数の最適化である．

ここでは，ブラックボックス関数を最適化する際の指針を，**図1**を使って説明する．今，**図1**の左の図のように2点だけ観測が得られているとする．次に観測する x の候補としてA-Gがあるときにどの点を選ぶと良いだろうか？というのが1次元でのブラックボックス関数の最適化である．

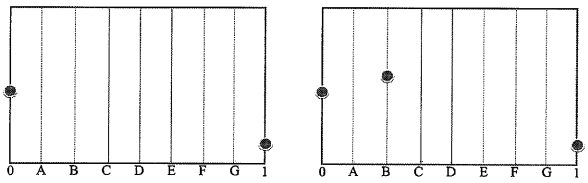

図1　ブラックボックス関数の最適化の例

A-Gを選ぶ指針は複数あると考えられるが，例えば，単純に真ん中のDを選ぶという人もいるだろうし，0のほうが1よりも高いので，真ん中よりは左のA-Cを選ぶという人も居るだろう．ここでは，次にBを選んだと考える．さて，観測データとして3点あると，次に選ぶ戦略もまた様々に考えられる．この配置を見ると，AかCを選ぶと高い点が得られる可能性が高そうであると考える人も居るだろう．もしくは，右側の領域をほとんど調べていないので，次は少し冒険してE-G辺りを調べるという人も居るだろう．

このように，現在観測されているデータから，次にどの点を選ぶかを考える際には，既に得られている観測データの中から高いデータ点の周りを選ぶ戦略(これを「**活用**」呼ぶ)とまだ観測されていないような点を選ぶ戦略(これを「**探索**」と呼ぶ)が考えられる．この「探索と活用」のトレードオフを取ることが重要である．ベイズ的最適化では，「探索と活用」のトレードオフを獲得関数の設計によって数理的に扱うことでブラックボックス関数の最適化を行う．本稿では，このようなベイズ的最適化の近年の話題を，機械学習分野での発展を中心に紹介する．

* 東京大学
University of Tokyo

2. 実験計画法

実験計画法は，効率的に実験を行うための応用統計学の一分野で，自然科学，社会科学，エンジニアリングなど様々な分野で用いられている．実験計画法の歴史は，20世紀前半．ロナルド・A・フィッシャー(Ronald A. Fisher)が勤務していたイギリスのロザムステッド農業試験場において，少ない投資で最大の効果を得るために統計的な手法を用いることから始まる．ここでは，もっとも基本的な方法であるラテン方格を用いた方法について説明する．これらの方法は，ベイズ的最適化でも要素技術として用いられる重要な技術である．

2.1 ラテン方格

ラテン方格は，少ない実験回数で複数の要因を分析する際に用いられる基本的な道具である．ラテン方格は，m 個のシンボルで埋められた $m \times m$ の表で，それぞれの行と列に現れるシンボルに重複を許さない配置で構成される．

例えば，三つの試薬 A, B, C をテストする際の順番の効果を分析することを考えよう．すべての順列をテストできれば $3! = 6$ 回実験を行えば良い．しかし，一回の実験でかかる費用が大きく，結果が出るのに一週間かかるとするとできるだけ少ない実験回数で効果を分析したくなる．ここで，3-ラテン方格を用いると，行と列に重複を許さずシンボルを配置するので

$$\begin{bmatrix} A & B & C \\ C & A & B \\ B & C & A \end{bmatrix}$$

，となるが，これを

Order	1	2	3
Week 1	A	B	C
Week 2	C	A	B
Week 3	B	C	A

のように，実験を行う順番だと思えば，三週間で全ての順番(列)での試薬の効果を確認できる．この情報を元に分散分析など統計的な道具を用いて分析することで，より少ない実験回数でそれぞれの効果を分析することができる可能性がある．

2.2 ラテンハイパーキューブサンプリング

ラテンハイパーキューブサンプリングは，ラテン方格から派生したサンプリング手法である[14]．ラテン方格では，行と列でシンボルに重複を許さない配置を構成するが，この方法ではシンボルデータの実験計画しか構成できない．ラテンハイパーキューブサンプリングでは，超立方体上で各点ができるだけ均等に分布するような点の配置を生成する．これは，単純な一様分布からのサンプリングで良いように思えるが，一様分布からのサンプリングでは偏った点の配置も生成されてしまう可能性がある．以下，ラテンハイパーキューブサンプリングについて説明する．

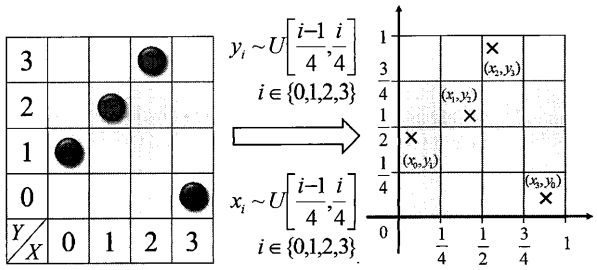

図2 ラテンハイパーキューブサンプリングの例

サンプル数 n，d 次元の独立な $[0,1]^d$ 上の点のサンプリングを考える．**図2**を用いて，ラテンハイパーキューブサンプリングの動作例を説明する．ここでは，サンプル数 $n = 4$，次元数 $d = 2$ のサンプルを生成することを考える．

アルゴリズムは以下の通りである．

1. $[0,1]$ を $n(=4)$ 個の領域に分け，$n(=4)$ 個の非負整数 $(0,1,2,3)$ を用意する．
2. $(0,1,2,3)$ に対して $d(=2)$ 個の順列を用意する．例えば，$(1,3,2,0)$，$(2,0,3,1)$ とする．
3. 行に $(0,1,2,3)$ の順列を $d(=2)$ 個ならべ，それぞれの列から $n(=4)$ 個の $d(=2)$ 次元のベクトルを生成する．例えば，

$$\begin{bmatrix} 1 & 3 & 2 & 0 \\ 2 & 0 & 3 & 1 \end{bmatrix} \Rightarrow \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 3 \\ 0 \end{bmatrix}, \begin{bmatrix} 2 \\ 3 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \quad (2)$$

> 生成された $n(=4)$ 個の $d(=2)$ 次元のベクトルの点を配置すると，$n(=4)$ 個の領域に分けた空間を表と見ると，どの行もどの列にも1つの点のみ配置されることがわかる(**図2**の左参照)．したがって，$n(=4) \times n(=4)$ に区切られた各々の領域内で一様にサンプリングを行えば，空間上に一様に配置された点配置を得ることができる(**図2**の右参照)．

4. $d = 2$ 次元のベクトル $(X, Y) = (x_i, y_i)$ $i = 0, 1, \ldots, n-1$ を以下によって生成する．

$$x_i \sim U\left[\frac{i}{n}, \frac{i+1}{n}\right], \; y_i \sim U\left[\frac{i}{n}, \frac{i+1}{n}\right]. \quad (3)$$

ここで，$U[\cdot]$ は一様分布である．

このようなアルゴリズムによって得られた点配置は，ラテン方格に類似した性質(各ベクトルのもつ要素の値にバラつきがある)を持つことから，数値データの実験計画に用いることができる．

ラテン方格やラテンハイパーキューブサンプリングを用いることで，少ない実験回数の結果から情報を引き出すことができる可能性があるが，これらの方法は予め決められた実験計画を構成するもので，観測され

た情報を利用して実験設定を更新するものではない．次に説明するベイズ的最適化を用いることで，観測された情報を用いて次の実験設定を決める実験計画を行うことができる．

3. ベイズ的最適化(Bayesian Optimization)

ここでは一般的によく研究されているブラックボックス関数の事前分布としてガウス過程を仮定するベイズ的最適化手法について説明する．観測が得られる毎に，ブラックボックス関数の事後分布を更新し，事後分布によって周辺化されたベイズ予測分布の情報を元に次に観測すべき点を決定する．まず，ガウス過程について簡単に説明し，次に，ガウス過程の事後分布を次の観測点の選択に活かすためのアルゴリズムについて説明する．

3.1 ガウス過程(Gaussian Process)

任意の n で，任意の部分集合 $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\} \subset \mathcal{X}$ に対して，$(f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), \ldots, f(\boldsymbol{x}_n))^\top \in \mathbb{R}^n$（$\top$：転置）が，

$$\text{平均：}\ \boldsymbol{m}(\boldsymbol{x}_{1:n}) = (m(\boldsymbol{x}_1), m(\boldsymbol{x}_2), \ldots, m(\boldsymbol{x}_n))^\top, \quad (4)$$

共分散行列：

$$K_\theta(\boldsymbol{x}_{1:n}) = \begin{bmatrix} k_\theta(\boldsymbol{x}_1, \boldsymbol{x}_1) & k_\theta(\boldsymbol{x}_1, \boldsymbol{x}_2) & \ldots & k_\theta(\boldsymbol{x}_1, \boldsymbol{x}_n) \\ k_\theta(\boldsymbol{x}_2, \boldsymbol{x}_1) & k_\theta(\boldsymbol{x}_2, \boldsymbol{x}_2) & \ldots & k_\theta(\boldsymbol{x}_2, \boldsymbol{x}_n) \\ \vdots & \vdots & \ddots & \vdots \\ k_\theta(\boldsymbol{x}_n, \boldsymbol{x}_1) & k_\theta(\boldsymbol{x}_n, \boldsymbol{x}_2) & \ldots & k_\theta(\boldsymbol{x}_n, \boldsymbol{x}_n) \end{bmatrix}$$

の n 次元ガウス分布に従うとき，f はガウス過程に従うと定義し，$f \sim \mathcal{GP}(f|m, k_\theta)$ と記す．ここで，$m : \mathcal{X} \to \mathbb{R}$ は平均関数と呼ばれ，$k_\theta : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$ をカーネル関数，$\boldsymbol{\theta} = (\theta_0, \theta_1, ..., \theta_D)$ をカーネル関数のパラメータとする．ガウス過程は平均関数とカーネル関数によって決定される．

観測データを $\mathcal{D}_{1:n} = \{\boldsymbol{x}_i, y_i\}_{i=1}^n$ とする．また，y_i と $\boldsymbol{x}_i$ に対して以下の関係を仮定する．

$$y_i = f(\boldsymbol{x}_i) + \epsilon_i,\ \epsilon \sim \mathcal{N}(0, \rho^2), \quad (5)$$

ここで，$\mathcal{N}(0, \rho^2)$ は，平均 0，分散 ρ^2 の正規分布である．

関数 $f : \mathcal{X} \to \mathbb{R}$ の事前分布としてガウス過程を仮定する．

$$f \sim \mathcal{GP}(f|m, k_\theta). \quad (6)$$

ガウス過程による関数のモデル化は，直感的には，$\boldsymbol{x}_i$ と $\boldsymbol{x}_j$ が“近い”場合 $f(\boldsymbol{x}_i)$ と $f(\boldsymbol{x}_j)$ も近い値になるようなモデリングになっており，この“近さ”を表現するのがカーネル関数 $k_\theta(\boldsymbol{x}_i, \boldsymbol{x}_j)$ である．また，関数の滑らかさを制御するのもカーネル関数である．

ガウス過程の数理的に有用な性質の 1 つにベイズ予測分布が解析的に計算できることが挙げられる．式(4)及び式(5)によって仮定されたモデルにおいて，$\mathcal{D}_{1:n}$ が与えられた下で，未知の入力 $\boldsymbol{x}$ に対する y のベイズ予測分布は，

$$p(y|\boldsymbol{x}, \mathcal{D}_{1:n}, \boldsymbol{\theta}) = \mathcal{N}(y|\mu(\boldsymbol{x}; \mathcal{D}_{1:n}, \boldsymbol{\theta}), \sigma^2(\boldsymbol{x}; \mathcal{D}_{1:n}, \boldsymbol{\theta})) \quad (7)$$

と求めることができる[4]．ここで，

$$\mu(\boldsymbol{x}; \mathcal{D}_{1:n}, \boldsymbol{\theta}) = \boldsymbol{k}_\theta(\boldsymbol{x})^\top (K_\theta(\boldsymbol{x}_{1:n}) + \rho^2 \boldsymbol{I})^{-1} (\boldsymbol{y}_{1:n} - m(\boldsymbol{x}_{1:n})), \quad (8)$$

$$\sigma^2(\boldsymbol{x}; \mathcal{D}_{1:n}, \boldsymbol{\theta} = k_\theta(\boldsymbol{x}, \boldsymbol{x}) - \boldsymbol{k}_\theta(\boldsymbol{x})^\top (K_\theta(\boldsymbol{x}_{1:n}) + \rho^2 \boldsymbol{I})^{-1} \boldsymbol{k}_\theta(\boldsymbol{x}), \quad (9)$$

$$\boldsymbol{k}_\theta(\boldsymbol{x}) = (k_\theta(\boldsymbol{x}, \boldsymbol{x}_1), k_\theta(\boldsymbol{x}, \boldsymbol{x}_2), \ldots, k_\theta(\boldsymbol{x}, \boldsymbol{x}_n))^\top \quad (10)$$

である．この予測分布の情報を用いて $f(\boldsymbol{x})$ の最適化を行うのが GP によるベイズ的最適化である．

3.2 カーネル関数

ガウス過程をベイズ的最適化で用いる場合に重要なのはカーネル関数の設計である(平均関数は 0 または適当な定数(入力の平均値)を置くことが多い)．カーネル法で有名なカーネル関数に，二乗指数カーネル(squared exponential kernel)がある．特に，入力の各次元にパラメータを仮定した場合，関連度自動決定二乗指数カーネル(automatic relevance determination(ARD) squared exponential kernel)：

$$k_\theta^{\mathrm{SE}}(\boldsymbol{x}, \boldsymbol{x}') = \theta_0 \exp\left\{-\frac{1}{2} r_\theta^2(\boldsymbol{x}, \boldsymbol{x}')\right\}, \quad (11)$$

$$r_\theta^2(\boldsymbol{x}, \boldsymbol{x}') = \sum_{d=1}^{D} \frac{(x_d - x_d')^2}{\theta_d^2} \quad (12)$$

と呼ばれている．

二乗指数カーネルは，ガウス過程を用いた回帰や識別問題などではよく用いられるが，二乗指数カーネルを用いたガウス過程から生成される関数は，滑らかな関数である傾向が強いため，ブラックボックス関数を表現するには適切ではない可能性がある．そこで，ベイズ的最適化ではマターンカーネル関数がよく用いられる．特にマターン 5/2 カーネル関数：

$$k_\theta^{\mathrm{M5/2}}(\boldsymbol{x}, \boldsymbol{x}') = \theta_0 \left(1 + \sqrt{5 r_\theta^2(\boldsymbol{x}, \boldsymbol{x}')} + \frac{5}{3} r^2(\boldsymbol{x}, \boldsymbol{x}')\right) \times \exp\left\{-\sqrt{5 r_\theta^2(\boldsymbol{x}, \boldsymbol{x}')}\right\}. \quad (13)$$

がよく用いられる．マターンカーネルはベッセル関数によって定義される関数で 5/2 はベッセル関数のパラメータの値である．

3.3 獲得関数と「探索と活用」のトレードオフ

$f(\boldsymbol{x})$ が，微分可能で上に凸な関数であれば，最適値 $f(\boldsymbol{x}^*)$ を得るための探索 $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots \in \mathcal{X}$ は，勾配法

などによって簡単にアルゴリズムを構成することができる．しかし，$f(\boldsymbol{x})$がブラックボックス関数で，非線形性・多峰である可能性がある場合，従来の最適化法を使うことは難しい．ベイズ的最適化では，$f(\boldsymbol{x})$の最適化問題を，獲得関数$a(\boldsymbol{x})$の最適化問題へと置き換えることで，アルゴリズムを構成する．獲得関数は，ガウス過程のベイズ予測分布を用いて構成するため，カーネル関数κ_θおよびそれまでの観測データ$\mathcal{D}_{1:n}$に依存する関数である．したがって，ベイズ的最適化による$f(\boldsymbol{x})$の最適化アルゴリズムは，

$$\boldsymbol{x}_{t+1} = \operatorname*{argmax}_{\boldsymbol{x}} a(\boldsymbol{x}; \mathcal{D}_{1:t}, \boldsymbol{\theta}) \tag{14}$$

によって，$f(\boldsymbol{x}^*)$を得るための点列$\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots \in \mathcal{X}$を生成することによって構成される．

獲得関数の定式化によってベイズ的最適化のアルゴリズムは異なる．獲得関数を構成する際の基本的な考え方として，「探索と活用」のトレードオフが重要となる．例えば，既に観測した点の中で最も値の高い点の周辺を観測すれば，現状よりも高い点が見つかる可能性は高い．これを知識の「活用」と言う．しかし，ブラックボックス関数に多峰性がある場合，「活用」ばかりしていては局所的な解に留まってしまう．そこで，これまで観測してきたものと異なる領域に目を向けるのが「探索」である．この2つのトレードオフを考慮してアルゴリズムを構成する必要がある．ここでは，紙面の関係上重要な3つの獲得関数について紹介する．

3.3.1 GP-UCB (Upper Confidence Bound)

直感的に最もわかりやすい獲得関数が信頼区間を用いたGP-UCBである[13]．GP-UCBの獲得関数は

$$a^{\mathrm{GP\text{-}UCB}}(\boldsymbol{x}; \mathcal{D}_{1:t}, \boldsymbol{\theta}) = \mu(\boldsymbol{x}; \mathcal{D}_{1:t}, \boldsymbol{\theta}) + \beta\sigma(\boldsymbol{x}; \mathcal{D}_{1:t}, \boldsymbol{\theta}), \tag{15}$$

と定義される．ここでβは探索と活用をトレードオフするパラメータである．βが大きくなると$\sigma(\boldsymbol{x}; \mathcal{D}_{1:t}, \boldsymbol{\theta})$を重要視するようになる．$\sigma(\boldsymbol{x}; \mathcal{D}_{1:t}, \boldsymbol{\theta})$はこれまで観測していない不確実な領域程高い値を取るので「探索」を重要視していることになる．GP-UCBでは，このβによってアルゴリズムの挙動が変わる．Regretによる理論解析によりβを決めることはできるものの実際にはあまりうまく機能しないことが多い．

3.3.2 GP-EI (Expected Improvement)

GP-UCBの遥か昔，古よりベイズ的最適化で用いられていた獲得関数はGP-EIである[9]．GP-EIの獲得関数は

$$a^{\mathrm{GP\text{-}EI}}(\boldsymbol{x}; \mathcal{D}_{1:t}, \boldsymbol{\theta}) = \mathbb{E}_{\mathcal{GP}(f|\mathcal{D}_{1:t}, m, k_\theta)}[f(\boldsymbol{x}) - \max f(\boldsymbol{x}_{1:t})], \tag{16}$$

と定義される．ここで，$\max f(\boldsymbol{x}_{1:t}) = \max_i f(\boldsymbol{x}_i)$で，$\mathbb{E}_{\mathcal{GP}(f|\mathcal{D}_{1:t}, m, k_\theta)}[\cdot]$はガウス過程の事後分布$\mathcal{GP}(f|\mathcal{D}_{1:t}, m, k_\theta)$による平均である．この期待値計算は，解析的に解けることが知られている．

3.3.3 GP-MI (Mutual Infomation)

現在．おそらくもっとも性能の良いであろう獲得関数は，相互情報量に基づくGP-MIである[3]．GP-MIの獲得関数は，

$$a^{\mathrm{GP\text{-}MI}}(\boldsymbol{x}; \mathcal{D}_{1:t}, \boldsymbol{\theta}) = \mu(\boldsymbol{x}; \mathcal{D}_{1:t}, \boldsymbol{\theta}) + \beta\phi_t(\boldsymbol{x}), \tag{17}$$

$$\phi_t(\boldsymbol{x}) = \left(\sqrt{\sigma^2(\boldsymbol{x}; \mathcal{D}_{1:t}, \boldsymbol{\theta}) + \sum_{i=1}^{t-1} \sigma^2(\boldsymbol{x}; \mathcal{D}_{1:i}, \boldsymbol{\theta})} - \sqrt{\sum_{i=1}^{t-1} \sigma^2(\boldsymbol{x}; \mathcal{D}_{1:i}, \boldsymbol{\theta})} \right) \tag{18}$$

と定義される．

GP-MIは，一見すると複雑な獲得関数に見えるが，$\phi_t(\boldsymbol{x})$から$\sum_{i=1}^{t-1} \sigma^2(\boldsymbol{x}; \mathcal{D}_{1:i}, \boldsymbol{\theta})$を除いて考えるとGP-UCBの獲得関数となる．GP-MIは，Regretによる理論解析の結果としてもGP-UCBの性能を改善していることが知られている．

MIの名前の由来は，以下の関係から来ている．

$$\sum_{t=1}^{T} \sigma^2(\boldsymbol{x}_t; \mathcal{D}_{1:t}, \boldsymbol{\theta}) \leq \frac{2}{\log(1+\rho^{-2})} \max_{S_T \subset \mathcal{X}: |S_T| = T} \mathrm{MI}(\mathcal{S}_T). \tag{19}$$

ここで，$\mathcal{S}_T = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T\}$，$\mathrm{MI}(\mathcal{S})$は$\mathcal{S}$の点を用いて計算されるガウス過程の相互情報量である．したがって，$\sum_{i=1}^{t} \sigma^2(\boldsymbol{x}; \mathcal{D}_{1:i}, \boldsymbol{\theta})$は，観測が$t$個得られた時の相互情報量の下限を表している．ガウス過程の相互情報量は，センサーのは位置問題などに使われており，相互情報量の差が大きくなるような点を選ぶことで，それまでの点では被覆できなかった領域を選ぶことができる[6]．したがって，$\phi_t(\boldsymbol{x})$が大きくなるような点を選ぶことで探索の効果が得られる．

3.4 GP-UCBの動作例

「はじめに」で触れた1次元のブラックボックス関数の最適化をベイズ的最適化によって解いた場合の動作例を示す．ここでは，獲得関数として解釈の容易なGP-UCBを用いて説明する．**図3**は，GP-UCBで$\beta = 1$とした場合の動作例である．点線で表現された曲線が真の関数である．緑の曲線が獲得関数の値である．青い曲線が$\mu(\boldsymbol{x})$で，その周りの影で分散$\sigma^2(\boldsymbol{x})$(95%信頼区間)を表現する．

図1の場合と同様に，観測として2点与えられた状態(**図3(1)**)から見ていこう．まず，2点与えられた状態では，$\boldsymbol{\mu}(\boldsymbol{x})$は直線として推定されており．そ

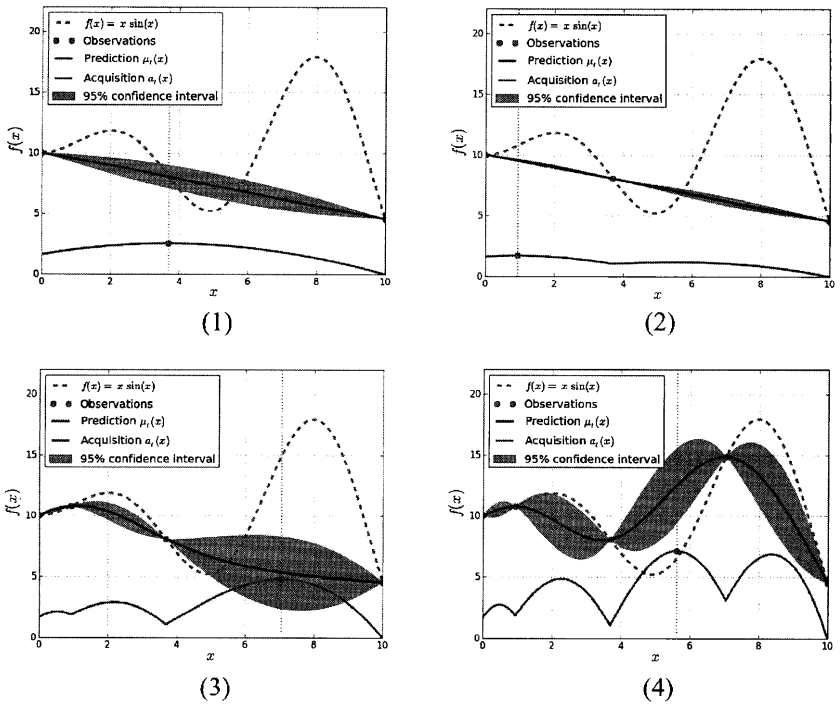

図3　GP-UCB の動作例

の周りの分散(すなわち不確実性)を考慮して，真ん中より少し左側の点が選ばれる(図3(2))．次に，獲得関数によって選ばれる点は，$x=0$ に近い点のようだ(図3(3))．4点選ばれた配置を見ると $x=0$ 付近に少し観測が集中しているため，右側領域の $\sigma(\boldsymbol{x})$ が大きくなっていることがわかる(図3(3))．これにより，獲得関数も右側領域が大きくなるので，右側領域の点を観測している．結果としては，左側領域で探索するよりも高い点が得られた(図3(4))．これらを繰り返すことにより局所解に陥ることなく最適解を見つけられそうである．

3.5　獲得関数の最適化

ベイズ的最適化では，ブラックボックス関数 $f(\boldsymbol{x})$ の最適化の代わりに，獲得関数 $a(\boldsymbol{x})$ の最適化を行う．しかし，図3(4)の獲得関数を見てもらえば分かる通り，GP-UCB のように平均と分散を足し合わせただけの単純な獲得関数でも，多峰の複雑な関数になってしまい，獲得関数そのものの最適化は実は難しい問題である．ただし，候補となる $\{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_T\}$ が予め決められていれば，その中から選べば良いのでそれほど問題にはならない．

獲得関数の最適化は，非線形最適化なので様々な方法があるが，簡単な方法として実験計画法で紹介したラテンハイパーキューブサンプリングを用いる方法がある．ラテンハイパーキューブサンプリングによって，重なりの少ない候補点を大量にサンプリングし，その中で獲得関数の高いものを選べばよい．また，ラテンハイパーキューブサンプリングをより精密にしたものとして，ソボル列も用いられている[5]．

3.6　カーネル関数のパラメータの扱い

カーネル関数と獲得関数を決めると，ベイズ的最適化の振る舞いはカーネル関数のパラメータに依存する．ガウス過程では，周辺尤度を解析的に求めることができるため，経験ベイズ法によりカーネル関数のパラメータ推定を行うことができる．このような経験ベイズ法によるアプローチは，観測点が多い場合には有効である．しかし，ベイズ的最適化では，観測点が徐々に増えていく設定であるため，経験ベイズ法を行うと初期の観測点に引きづられて好ましくない値に収束してしまう可能性がある．

そこで，カーネル関数のパラメータに(ほとんど無情報な)事前分布 $p(\boldsymbol{\theta})$ を仮定し，カーネル関数のパラメータ上の事後分布 $p(\boldsymbol{\theta}|\mathcal{D}_{1:n})$ で平均を取った獲得関数

$$\widetilde{a}(\boldsymbol{x};\mathcal{D}_{1:n}) = \int a(\boldsymbol{x};\mathcal{D}_{1:n},\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathcal{D}_{1:n})d\boldsymbol{\theta}. \qquad (20)$$

が提案されている[12]．これをベイズ獲得関数と呼ぶことにする．

実際には，ベイズ獲得関数における期待値計算は解析的に求めることができないため，事後分布 $p(\boldsymbol{\theta}|\mathcal{D}_{1:n})$ からのサンプル点を用いた

$$\widetilde{a}(\boldsymbol{x};\mathcal{D}_{1:n}) = \frac{1}{S}\sum_{s=1}^{S} a(\boldsymbol{x};\mathcal{D}_{1:n},\boldsymbol{\theta}^{(s)}),\ \boldsymbol{\theta}^{(s)} \sim p(\boldsymbol{\theta}|\mathcal{D}_{1:n}) \qquad (21)$$

が用いられる．この事後分布 $p(\theta|\mathcal{D}_{1:n})$ からのサンプリングにはスライスサンプラーを用いることができる[10]．

4. おわりに：機械学習分野におけるベイズ的最適化の最近の話題

近年，機械学習では，モデルの複雑化に伴い，チューニングすべきハイパーパラメータが増えている．

このようなハイパーパラメータは，通常，交差検定を用いて決定することが多いが，ハイパーパラメータの数が多くなるとグリッドサーチによって適切なハイパーパラメータを設定することが難しくなる．このような場合にベイズ的最適化が効果的であることが確認されている[12]．また，すでにあるデータセットにおけるベイズ的最適化で得られたハイパーパラメータの情報を，別のデータセットにおけるベイズ的最適化に用いることで，ベイズ的最適化に必要な観測コストを抑える方法が提案されている[15]．制約付きのベイズ的最適化も近年提案されてる[1,4]．人によるブラックボックス関数の最適化とベイズ的最適化の実験的な比較もされている[2]．

謝辞

本稿で使用した図の一部は東京大学中川研究室卒業生の梁 曾漢さんに作成していただいた．

参 考 文 献

1) Bayesian optimization with inequality constraints. In *Proceedings of the 31st International Conference on Machine Learning*, pages 937-945, 2014.

2) Ali Borji and Laurent Itti. Bayesian optimization explains human active search. In *Advances in Neural Information Processing Systems 26*, pages 55-63. 2013.

3) Emile Contal, Vianney Perchet, and Nicolas Vayatis. Gaussian process optimization with mutual information. In *Proceedings of the 31th International Conference on Machine Learning*, pages 253-261, 2014.

4) Michael A. Gelbart, Jasper Snoek, and Ryan P. Adams. Bayesian optimization with unknown constraints. In *Proceedings of the 30st Uncertainty in Artificial Intelligence*, 2014.

5) M. W. Hoffman and B. Shahriari. Modular mechanisms for bayesian optimization. In *NIPS Workshop on Bayesian optimization*, 2014.

6) Andreas Krause, Ajit Singh, and Carlos Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9: 235-284, 2008.

7) Harold J Kushner. A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. *Journal of Fluids Engineering*, 86(1): 97-106, 1964.

8) Bertil Matérn. *Spatial Variation*. Springer, 1960.

9) J. Močkus, V. Tiesis, and A. Zilinskas. The application of bayesian methods for seeking the extremum. *Towards Global Optimization*, 2: 117-129, 1978.

10) Iain Murray and Ryan P Adams. Slice sampling covariance hyperparameters of latent gaussian models. In *Advances in Neural Information Processing Systems*, pages 1732-1740. 2010.

11) Carl Edward Rasmussen. *Gaussian processes for machine learning*. The MIT Press, 2006.

12) Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In *Neural Information Processing Systems*, pages 2951-2959, 2012.

13) Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. pages 1015-1022, 2010.

14) M.L. Stein. Large sample properties of simulations using latin hypercube sampling. pages 143-151, 1987.

15) Kevin Swersky, Jasper Snoek, and Ryan P Adams. Multi-task bayesian optimization. In *Advances in Neural Information Processing Systems 26*, pages 2004-2012. 2013.