4.1 導入

一般的な非線形最適化問題は次の 3 つに分類される

1. 一次元無制約問題
2. 多次元無制約問題
3. 多次元制約問題

一つ目の問題は最も簡単に解け、三つ目の問題が一番難しい。実際には多次元制約問題は通常多次元無制約問題に変形され、次に一次元無制約問題へと変形される。事実上多くの利用可能な非線形計画アルゴリズムは無制約の 1 変数関数の最小化に基づいている。それゆえ、効率的な多次元無制約最適化アルゴリズムと多次元制約最適化アルゴリズムを構築する際には、効率的な一次元最適化アルゴリズムが要求されることになっている。

$minimizeF = f(x)$ (f は一変数関数)

この問題は $f(x)$ がある範囲で単峰性関数である場合に解を持つ。つまり、$f(x)$ がある範囲 $[x_L \leq x \leq x_U]$ (x_L と x_U はそれぞれ最小点 $x*$ の下限と上限) で $f(x)$ が唯一の最小点を持つ場合、解を持つ。

一次元最適化法には一般的に二つの方法がある、即ち、探索法と近似法である。

探索法では x を含む $[x_L, x_U]$ の範囲を関数の評価にしたがって縮小していき $[x_{\{}L, k\}, x_{\{}U, k\}]$ なる充分に小さい範囲を得るまで縮小を繰り返す。最小点は $[x_{\{}L, k\}, x_{\{}U, k\}]$ の範囲の中間点に存在すると予想できる。これらの方法はあらゆる関数に適用でき、$f(x)$ の微分可能性は必要でない。

近似法では、低次の多項式の形の関数の近似が想定される。通常は 2 次や 3 次の多項式である。これは、初等微積分及び推定される近似値 x^* を用いる解析の時である。 $[x_L, x_U]$ の範囲はこのとき変形され、その作業は充分に正確な x^* の値を得るまで幾度も繰り返される、

これらの方法では、$f(x)$ は連続的微分可能である必要がある、つまり、$f(x) \in C^1$ ということである。

この章では次の 1 から 8 のいくつかの 1 次元の最適化のアプローチが考察されている。

1. 二分探索法
2. フィボナッチ探索
3. 黄金分割探索
4. 二次補間法
5. 三次補間法
6. Davies-Swann-Campey 法

1 から 3 は探索法であり、4,5 番目は近似法である、そして 6 番目は実用的かつ有用な、探索法と近似法を組み合わせた方法である。この章はさらに、Fletcher による非厳密直線探索と呼ばれる方法を扱う。この方法はいくつかの最適化法における計算量を減らすようなある重要な利点をもたらしている。

4.2 二分探索

$[x_L, x_U]$ の区間に最小値が存在すると考えられる単峰性関数について考える。この区間は不確実の区間と呼ばれる。$f(x)$ の最小点 x^* は, 十分に小さい区間が得られるまで漸次不確実の区間に縮小していくことにより見つけられる。探索法では, これは複数の $f(x)$ の適切な点での値を用いて達成される。

もし,$f(x)$ の値が $x_a \in (x_L, x_U)$ の単一点のみで分かる場合, 図 4.1(a) で描かれているように x^* の点は x_L から x_a または,x_a から x_U の範囲の中に一致しそうである。従って, 利用可能な情報は不確実な区間を縮小するのに十分ではない。しかしながら, もし $f(x)$ の値が 2 点分かれば, たとえば x_a と x_b での値が分かれば、すぐに縮小は可能になる。それには 3 つの可能性があるだろう、いわゆる

- (a) $f(x_a)<f(x_b)$
- (b) $f(x_a)>f(x_b)$
- (c) $f(x_a) = f(x_b)$

(a) のケースでは x^* は $x_L<x^*<x_a$ または $x_a<x^*<x_b$ の範囲にあるだろう、 すなわち、図 4.1a で示されるように、$x_L<x^*<x_b$ である。 $x_b<x^*<x_U$ の可能性は明確に除外する、なぜならこの範囲であれば、$f(x)$ が x_b の左右に一つずつ最小値を持つことを暗示するからである。。似たように、(b) のケースでは、図 4.1b で示されるように x^* は $x_a<x^*<x_U$ の範囲にあるはずだ。 (c) のケースでは、x^* は $x_a<x^*<x_b$ にあるはずだ、 すなわち図 4.1c で示されるように $x_L<x^*<x_b$ と $x_a<x^*<x_U$ の両方の不等式はともに除外するのに十分だからだ。

不確実な範囲を縮小していく基本的な戦略はいわゆる二分探索である。この方法では,$f(x)$ は十分小さな正の ϵ の下、$x_a = x_1 - \epsilon/2$ と $x_b = x_1 + \epsilon/2$ の 2 点で計算される。 このとき、$f(x_a)<f(x_b)$ か $f(x_a)>f(x_b)$ かによって x_L から $x_1 + \epsilon/2$ の範囲か、$x_1 - \epsilon/2$ から x_L の範囲かを選んで縮小できる。、 そして $f(x_a) = f(x_b)$ の時、範囲はどちらをとっても良いだろう。 もし $x_1 - x_L = x_U - x_1$ つまり、$x_1 = (x_L + x_U)/2$ と仮定すると、不確実な範囲は即座に半分に縮小できる。同じ手順を縮小された範囲に繰り返すことができる、つまり、x_2 が縮小された範囲の中心の点に位置する場合など、$f(x)$ は $x_2 - \epsilon/2$ と $x_2 + \epsilon/2$ で計算することができる。例えば、もし二分法が図 4.2 の関数に適用されるとき、不確実な範囲は 4 回の反復で $0<x^*<1$ から $9/16+9\epsilon/16<x^*<5/8-3\epsilon/8$ に縮小される。それぞれの反復は不確実な範囲を半分の範囲に縮小し、また、従って k 回の反復で不確実な範囲の広さは $I_0 = x_U - x_L$ としたとき $I_k = (\frac{1}{2})^k\{I_0 - \epsilon\} + \epsilon$ に縮小される。例えば、7 回の反復で不確実な範囲は 1%未満の大きさの範囲に縮小される。計算量の対応は各反復に 2 回の関数計算が必要になるとして、14 回の関数計算になるだろう。

4.3 Fibonacci 探索

図 4.3 に示すように不確実な範囲 $I_k = [x_{L,k}, x_{U,k}]$ 及び I_k に位置する $x_{a,k}, x_{b,k}$ について考える。 4.2 節のように、$f(x_{a,k}), f(x_{b,k})$ で書かれる、$x_{a,k}, x_{b,k}$ における $f(x)$ の値は、 もし,$f(x_{a,k})$<$f(x_b, k)$ であれば、左側の範囲を $I^L_{k+1} = [x_{L,k}, x_{b,k}]$ として選択するため、

もし,$f(x_{a,k})$>$f(x_b, k)$ であれば、右側の範囲を $I^R_{k+1} = [x_{a,k}, x_{U,k}]$ として選択するため、

さもなくば、$f(x_a, k) = f(x_b, k)$ であれば、I^R_{k+1} と I^L_{k+1} のどちらかを選択するために使われる。

もし、右の区間 I^R_{k+1} が選択された場合、その範囲は最小点を含み、それに加え、$f(x)$ の値は I^R_{k+1} の内点の一点でわかっている、すなわち $x_{b,k}$ の点である。また $f(x)$ が I^R_{k+1} 内点の一つ以上の点、つまり $x_{b,k+1}$ の点で評価されるとき、さらに不確実な区間を縮小するための十分な情報が得られ、そして、上記の処理サイクルは繰り返される。図 4.3 で示されるような I^L_{k+2} と I^R_{k+2} の二つの新しい部分区間のうち一つを、前述のように選択することが可能であり、これが繰り返される。この方法では、一回の繰り返しに対し関数評価が一つだけしか要求されず、計算量は二分探索法で要求される計算量と比較して削減されるだろう。

図 4.3 では

$$I_k = I^L_{k+1} + I^R_{k+2} \quad (4.1)$$

としていて、そして便宜上、同一の区間である場合、

$$I^L_{k+1} = I^R_{k+1} = I_{k+1}$$
$$I^L_{k+2} = I^R_{k+2} = I_{k+2}$$

式 4.1 は

$$I_k = I_{k+1} + I_{k+2} \quad (4.2)$$

という漸化式を与えている。

上記の手順が何回も繰り返されるとき、区間列 $\{I_1, I_2, \cdots, I_n\}$ は次のようにして得られる

$$\begin{aligned} I_1 &= I_2 + I_3 \\ I_2 &= I_3 + I_4 \\ &\vdots \\ I_n &= I_{n+1} + I_{n+2} \end{aligned}$$

上記の一連の n 個の方程式では、$n+2$ 個の変数があり、そして I_1 が初期区間として与えられた時 $n+1$ 個の変数が残る。それゆえに、無限数列は何らかの追加のルールを特定することで作れる。特に興味深い 2 つの特殊な順

序は Fibonacci 数列と黄金分割数列である。Fibonacci 数列はこの章で、黄金分割数列は 4.4 章で考える。

Fibonacci 数列は $n+2$ 回目の繰り返しで零になる範囲、つまり $I_{n+2}=0$ を仮定することで得られる。 式 4.2 で $k=n$ と置けば、次のように書ける。

$$
\begin{aligned}
I_{n+1} &= I_n - I_{n+2} = I_n \equiv F_0 I_n \\
I_n &= I_{n+1} + I_{n+2} = I_n \equiv F_1 I_n \\
I_{n-1} &= I_n + I_{n+1} = 2I_n \equiv F_2 I_n \\
I_{n-2} &= I_{n-1} + I_n = 3I_n \equiv F_3 I_n \\
I_{n-3} &= I_{n-2} + I_{n-1} = 5I_n \equiv F_4 I_n \\
I_{n-4} &= I_{n-3} + I_{n-2} = 8I_n \equiv F_5 I_n \\
&\vdots \\
I_k &= I_{k+1} + I_{k+2} = F_{n-k+1} I_n \ \cdots \ (4.3a) \\
&\vdots \\
I_1 &= I_2 + I_3 = F_n I_n \ \cdots \ (4.3b)
\end{aligned}
$$

得られた数列、すなわち $\{1,1,2,3,5,8,13,\cdots\}=\{F_0,F_1,F_2,F_3,F_4,F_5,F_6,\cdots\}$ の数列は、数学の様々な分野に存在する Fibonacci 数列として有名である。

それは $F_0=F_1=1$ の下での再帰的な関係によって生まれ、

$$F_k = F_{k-1} + F_{k-2} \text{ for } k \geq 2 \ (4.4)$$

である。

一次元最適化におけるこれの応用は Fibonacci 探索法である。 この方法の、$n=6, I_1=100$ で始まり、常に左側の範囲が選択された場合、つまり最小値が $x=0$ の近傍で存在する場合が、図 4.4 で描かれている。

もし、繰り返しの回数が n になると仮定すると、式 (4.3b) から、Fibonacci 探索は不確実な範囲を

$$I_n = \frac{I_1}{F_n} \ (4.5)$$

に縮小する。

例えば、もし $n=11$ なら、$F_n=144$ なので、I_n は I_1 の値の 1% 未満の値に縮小される。これは 11 回の反復を伴い、各反復に一回の関数の評価が必要になるので、同じだけの正確さを得るために 14 回の関数評価が必要な二分探索法に対して、合計 11 回の関数の評価が必要になる。事実上 Fibonacci 探索は二分探索に比べて効率が良い。それにまた、他の探索法に比べて最も大きな範囲の削減を達成していることも示されている、そしてしたがって、必要計算量の面においても最も効率的である。

区間の Fibonacci 数列は n が明らかになっている場合のみ得られる。もし、最適化の目的が規定された許容誤差の範囲で x^* を見つけることの場合、要求される n は式 (4.5) を用いることで用意に推測できる。しかしながら、もし目的が $f(x)$ の最小値を規定された許容誤差の範囲で見つけることの場合、問題を解かずに要求される n を決めるのは難しくなるだろう。唯一得られる情報

は、もし $f(x)$ の最小値が浅く、$f(x)$ が解の近傍で急速に変化する場合に、n が小さくなるだろうということである。

上の原理は Fibonacci 探索を実装するために使われる。初期の最小化値の境界を仮定しよう、いわゆる $x_{L,1}$ と $x_{U,1}$ である、そして n の値が与えられ、$f(x)$ の数学的表現が存在する。この実装には連続した範囲を計算すること、$f(x)$ を評価すること、そして適切な範囲を選択することからなっている。

k 回目の繰り返しでは、$x_{L,k}, x_{a,k}, x_{b,k}, x_{U,k}, I_{k+1}$ の量と

$f_{a,k} = f(x_{a,k}), f_{b,k} = f(x_{b,k})$

が分かる、そして、 $x_{L,k+1}, x_{a,k+1}, x_{b,k+1}, x_{U,k+1}, I_{k+2}, f_{a,k+1}, f_{b,k+1}$ の量が必要となる。I_{k+2} の範囲は

$$I_{k+2} = \frac{F_{n-k-1}}{F_{n-k}} I_{k+1} \quad (4.6)$$

のようにして式 $(4.3a)$ から得られる。

残った量は以下の様にして計算される。

$f_{a,k} > f_{b,K}$ ならば、x^* は $[x_{a,k}, x_{U,k}]$ の範囲の中にあり、従って x^* の境界は

$$x_{L,k+1} = x_{a,k} \quad (4.7)$$

$$x_{b,k+1} = x_{L,k+1} + I_{k+2} \quad (4.8)$$

の様に更新される。

似たように、新しい範囲の二つの内点、いわゆる $x_{a,k+1}$ と $x_{b,k+1}$ はそれぞれ $x_{b,k}$ と $x_{L,k+1} + I_{k+2}$ になるだろう。このように図 4.5 に示されるように $x_{a,k+1} = x_{a,k}$ (4.9) $x_{b,k+1} = x_{L,k+1} + I_{k+2}$ (4.10) の様に割り当てられる。

$f_{b,k}$ の値は $x_{a,k+1}$ における $f(x)$ の値として保持され、$x_{b,k+1}$ での $f(x)$ の値は計算される、つまり $f_{a,k+1} = f_{b,k}$ (4.11) $f_{b,k+1} = f(x_{b,k+1})$ (4.12) である。

一方で、$f_{a,k} < f_{b,k}$ の場合、x^* は $[x_{L,k}, x_{b,k}]$ の範囲の中にある。この場合では、図 4.6 で示されるように

$$x_{L,k+1} = x_{L,k} \quad (4.13)$$

$$x_{U,k+1} = x_{b,k} \quad (4.14)$$

$$x_{a,k+1} = x_{U,k+1} - I_{k+2} \quad (4.15)$$

$$x_{b,k+1} = x_{a,k} \quad (4.16)$$

$$f_{b,k+1} = f_{a,k} \quad (4.17)$$

のように割り当て、

$$f_{a,k+1} = f(x_{a,k+1}) \quad (4.18)$$

のように計算する。

運悪く $f_{a,k} = f_{b,k}$ の場合、x^* が $[x_{L,k}, x_{b,k}]$ と $[x_{a,k}, x_{U,k}]$ のどちらの範囲にも含まれるので上のどちらかの割り当てが使われる。

上記の手順は図 4.7 に示されるように、

$$I_{k+2} = I_n$$

かつ

$$x^* = x_{a,k+1} = x_{b,k+1}$$

の場合、$k = n - 2$ になるまで繰り返される。明らかに, 最小値はある許容誤差 $\pm 1/F_n$ の間の値に決まるだろう。

x^* の誤差は、二分探索法の 1 段階の適用時の二分の一になるだろう。これは、ϵ<$1/F_n$ の下で、

$$x^* = \begin{cases} x_{a,k+1} + \frac{1}{2F_n} \& \text{if} \& f(x_{a,k+1} + \epsilon) < f(x_{a,k+1}) \\ x_{a,k+1} + \frac{\epsilon}{2} \& \text{if} \& f(x_{a,k+1} + \epsilon) = f(x_{a,k+1}) \\ x_{a,k+1} - \frac{1}{2F_n} \& \text{if} \& f(x_{a,k+1} + \epsilon) > f(x_{a,k+1}) \end{cases}$$

と $x*$ を定めたときに、点 $x = x_{a,k+a} + \epsilon$ で $f(x)$ を評価することで達成される。

もし n がとても大きい場合、$x_{a,k}$ と $x_{b,k}$ の差はとても小さくなる、そして $x_{a,k}$ は丸め誤差のために、$x_{b.k}$ を超える可能性がある。もしこれが起きたとき、信頼できない結果が得られることになる。そのような適用の場合、もし丸め誤差が発生すれば、問題を取り除くために、アルゴリズムの中に誤差の検査が組み込まれることになる。おそらく、$x_{a,k} \approx x_{b,K}$ であれば、十分な精度が達成されるため、一つの可能性 ($x_{a,k} \approx x_{b,K}$ の場合) がアルゴリズムを終了させるでしょう。

上記の原理は次のアルゴリズムによって構築される。

アルゴリズム 4.1 Fibonacci 探索

Step1

$x_{L,1}, x_{U,1}$ と n を入力する

Step2

式 $(4,4)$ を用いて、$F_1, F_2, \ldots, F_n$ を計算する

Step3

$I_1 = x_{U,1} - x_{L,1}$ を割り当て、
$I_2 = \frac{F_{n-1}}{F_n} I_1$ $(seeEq.(4.6))$
$x_{a,1} = x_{U,1} - I_2,\ x_{b,1} = x_{L,1} + I_2$
$f_{a,1} = f(x_{a,1}),\ f_{b,1} = f(x_{b,1})$
を計算し、$k = 1$ と置く

Step4

I_{k+2} を式 (4.6) を用いて計算する。
もし $f_{a.k} \geq f_{b,k}$ であるとき、$x_{L,k+1}, x_{U,k+1}, x_{b,k+1}, f_{a,k+1}, f_{b,k+1}$ を式 (4.7) から (4.12) を使って更新する。
さもなくば、もし $f_{a,k}<f_{b.k}$ であれば式 (4.13) から (4.18) を使って情報を更新する。

Step5

もし $k = n - 2$ または $x_{a,k+1}>x_{b,k+1}$ であれば、 $x^* = x_{a,k+1}$ と $f^* = f(x*)$ を出力し、終了する。
さもなくば、$k = k + 1$ と置き、Step4 からの手順を繰り返す。
$x_{a,k+1}>x_{b,k+1}$ の条件は、コンピュータを利用した精度での $x_{a,k+1} \approx x_{b,k+1}$ に、早くなったこと、または、このアルゴリズムに何らかの誤りがあることを暗示している。このように、この条件は代替終了基準として用いられる。

4.4 黄金分割探索

主なFibonacci探索の不利益は反復の回数を入力として与えなければならないことである。目的関数の極小または最小値が望ましい正確さを達成するまで反復を続けるような探索法がいわゆる黄金分割探索である。このアプローチでは、図4.8で示されるようにFibonacci探索のように区間列$\{I_1, I_2, I_3, \cdots\}$を、(4.2)の再帰的関数を用いて生成する。連続した区間を生み出す規則はどの隣接した二つの区間の比も一定になるものである、すなわち

$$\frac{I_k}{I_{k+1}} = \frac{I_{k+1}}{I_{k+2}} = \frac{I_{k+2}}{I_{k+3}} = \cdots = K \tag{4.19}$$

つまり、

$$\frac{I_k}{I_{k+2}} = K^2 \tag{4.20}$$

$$\frac{I_k}{I_{k+3}} = K^3$$

などが成り立つ。
式(4.2)をI_{k+2}で割るにあたり、

$$\frac{I_k}{I_{k+2}} = \frac{I_{k+1}}{I_{k+2}} + 1 \tag{4.21}$$

得る、そして式(4.19)から式(4.20)までを用いると

$$K^2 = K + 1 \tag{4.22}$$

今,Kを解くと

$$K = \frac{1 \pm \sqrt{5}}{2} \tag{4.23}$$

を得る。

Kの負の値は不適切なので、$K = 1.618034$となる。この定数は黄金比として有名である。この用語は古代ギリシャで辺の比が$1 : K$となる四角形が最も美しい四角形であると考えられていた事実に由来していて、そしてその後黄金比として知られるようになったものである。同様に、$\{I_1, I_1/K, I_1/K^2, \ldots, I_1/K^{n-1}\}$の数列は黄金分割数列として知られるようになった

左の区間が常に選択された条件下での黄金分割探索が図 4.8 に示されている。図のように、この探索法は多くの点で Fibonacci 探索に似ている。 2 つの例外は

1. 連続した区間は n に依存しない。従って、反復は不確実な区間か目的関数の値の変化がある許容誤差 ϵ より低くなるまで繰り返される。
2. 連続した区間同士の比は、つまり F_{n-k-1}/F_{n-k} は、式 (4.22)-(4.23) により
$$\frac{1}{K} = K - 1 = 0.618034$$
になるような、$1/K$ に置き換えられる。

である。

黄金分割探索の効率は簡単に Fibonacci 探索の効率と比較できる。既知の、大きな値の n を適用する F_n と K の間の関係は

$$F_n \approx \frac{K^{n+1}}{\sqrt{5}} \quad (4.24)$$

である (e.g. $n = 11, F_n = 1.44$ の場合 $K^{n+1}/\sqrt{5} \approx 144.001$) 従って、式 (4.5) と式 (4.24) は Fibonacci 探索に対して、

$$\Lambda_F = I_n = \frac{I_1}{F_n} \approx \frac{\sqrt{5}}{K^{n+1}} I_1$$

のような不確実な範囲を与える。

似たように、黄金分割探索に対しては,

$$\Lambda_{GS} = I_n = \frac{I_1}{K^{n-1}}$$

それゆえに

$$\frac{\Lambda_{GS}}{\lambda_F} = \frac{K^2}{\sqrt{5}} \approx 1.17$$

が成り立つ。

従って、もし反復の回数が二つの方法で同じであれば、黄金分割探索法の不確実な範囲は、Fibonacci 探索の不確実な範囲と比べて 17% 大きな範囲となる。あるいは、黄金分割探索は Fibonacci 探索と同じだけの正確さを達成するためにより多くの反復を必要とするだろう。しかしながら、この不利益は合計の反復回数を最適化の開始時に与える必要がない事実によって相殺される。

黄金分割探索の実装は以下のものである

アルゴリズム 4.2 黄金分割探索

Step1

$x_{L,1}, x_{U,1}, \epsilon$ を入力する

Step2

$I_1 = x_{U,1} - x_{L,1}, K = 1.618034$ と割り当て、

$$I_2 = i_1/K$$
$$x_{a,1} = x_{U,1} - I_2, \qquad x_{b,1} = x_{L,1} + I_2$$
$$f_{a,1} = f(x_{a,1}), \& f_{b,1} = f(x_{b,1})$$

を計算し、 $k = 1$ とおく。

Step3

$$I_{k+2} = I_{k+1}/K$$

を計算する。

$f_{a,k} \geq fb, k$ の場合、

$x_{L,k+1}, x_{U,k+1}, x_{a,k+1}, x_{b,k+1}, f_{a,k+1}, f_{b,k+1}$

を式 (4.7) から (4.12) を用いて更新する。

さもなければ、つまり $f_{a,k}$<$f_{b,k}$ の場合、式 (4.13) から (4.18) を用いて、情報を更新する。

Step4

I_k<ϵ または $x_{a,k+1}$>$x_{b,k+1}$ の場合は、以下を実行する

- $f_{a,k+1}$>$f_{b,k+1}$ の場合
  - $x^* = \frac{1}{2}(x_{b,k+1} + x_{u,k+1)}$ を計算する
- $f_{a,k+1} = f_{b,k+1}$ の場合
  - $x^* = \frac{1}{2}(x_{a,k+1} + x_{a,k+1)}$ を計算する
- $f_{a,k+1}$<$f_{b,k+1}$ の場合
  - $x^* = \frac{1}{2}(x_{L,k+1} + x_{a,k+1)}$ を計算する

$f^* = f(x^*)$ を計算する x^*, f^* を出力し、終了

Step5

$k = k + 1$ と置き、Step3 から繰り返す。