

収束速度 $f(x) = 0$ の真の解を $x = \xi$ とし、ニュートン法の第 n ステップでの誤差を e_n とすると、

$$e_n = x_n - \xi \quad (2.6)$$

$$e_{n+1} = x_{n+1} - \xi \quad (2.7)$$

e_n を用いて $f(x)$ を $x = \xi$ の周りでテイラー展開すると、次のようになる。

$$\begin{aligned} f(x_n) &= f(\xi + e_n) \\ &= f(\xi) + f'(\xi)e_n + \frac{1}{2}f''(\xi)e_n^2 + O(e_n^3) \\ &\simeq f'(\xi)e_n + \frac{1}{2}f''(\xi)e_n^2 \end{aligned} \quad (2.8)$$

ここで、 $f(\xi) = 0$ を用いた。同様にして $f'(x)$ もテイラー展開すると、

$$\begin{aligned} f'(x_n) &= f'(\xi + e_n) \\ &= f'(\xi) + f''(\xi)e_n + \frac{1}{2}f'''(\xi)e_n^2 + O(e_n^3) \\ &\simeq f'(\xi) + f''(\xi)e_n \end{aligned} \quad (2.9)$$

ここで、ニュートン法の反復式

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \quad (2.10)$$

の両辺から ξ を引くと、(2.6)、(2.7) 式より

$$e_{n+1} = e_n - \frac{f(x_n)}{f'(x_n)} \quad (2.11)$$

となり、これに (2.8)、(2.9) 式を代入すると、

$$e_{n+1} = e_n - \frac{f'(\xi)e_n + \frac{1}{2}f''(\xi)e_n^2}{f'(\xi) + f''(\xi)e_n} \quad (2.12)$$

となる。いま、真の解 ξ の十分近くでは e_n は小さいので、分母の第 2 項を 0 として計算すると、この式は

$$\begin{aligned} e_{n+1} &= \frac{e_n f'(\xi) - f'(\xi)e_n + \frac{1}{2}f''(\xi)e_n^2}{f'(\xi)} \\ &= \frac{1}{2} \frac{f''(\xi)}{f'(\xi)} e_n^2 \end{aligned} \quad (2.13)$$

とすることができる。

一般に、正の数 $D > 0$ 、 $p > 1$ が存在して、十分大きな n について

$$|x_{n+1} - \xi| < D|x_n - \xi|^p \quad (2.14)$$

のとき、数列 $\{x_n\}$ は ξ に p 次収束するという。式 (2.13) より、ニュートン法は 2 次収束する解法であることがわかる。

表 2.2: ニュートン法による $\sqrt{2}$ の計算

n	x_n	$f(x_n)$
0	2	2
1	1.5	0.25
2	1.416666667	6.94×10^{-3}
3	1.414215686	6.01×10^{-6}
4	1.414213562	-1.06×10^{-9}
5	1.414213562	-1.06×10^{-9}

誤差 2次収束する解法では、ある反復での誤差が 10^{-m} のオーダーだった場合、それが反復毎に 10^{-2m} , 10^{-4m} , 10^{-8m} , ... と減少する。したがって、解の有効桁数は、毎回ほぼ2倍に増加する。

数値例 $f(x) = x^2 - 2$, $x_0 = 2$ としてニュートン法により $\sqrt{2}$ を計算した例を表 2.2 に示す（森下浩二君のレポートより）。 x_n の値を見ると、正しい値 $1.414213562 \dots$ と一致する桁数が毎回2倍程度に増えていることがわかり、2次収束していることが確認できる。また、表 2.1 の二分法では10回反復しても4桁しか正しい値が得られなかったが、ニュートン法では4回の反復で既に10桁も正しい値が得られており、収束が極めて速いことがわかる。

使用の際の注意事項 1 二分法では、 $f(a)f(b) \leq 0$ を満たすような区間から出発すれば解への収束が保証されたが、ニュートン法ではそのような大域的収束性は一般に保証できない。実際、図 2.1 に示すように、解から遠い初期値から出発した場合には、 x_n の値が振動し、一定値に収束しない場合もある。正しい解を得るには、解に十分近い初期値から出発する必要がある。

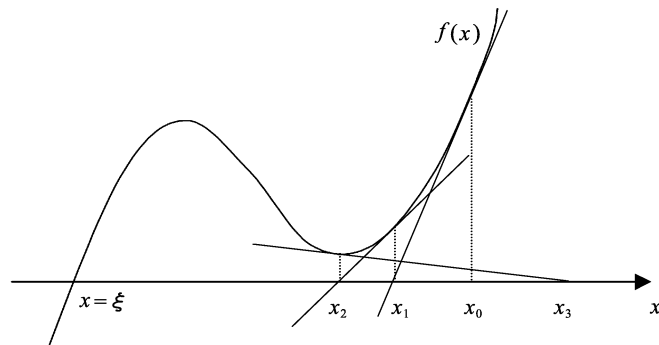


図 2.1: 解から遠い初期値から出発したためにニュートン法が収束しない例

使用の際の注意事項 2 2次収束性の証明では、解 ξ において $f'(\xi)$ が 0 でないことを暗黙のうちに用いた。しかし、重根の場合には $f'(\xi) = 0$ となり、(2.13) 式は成り立たなくなる。このため、2次収束性が崩れてしまう。この場合の収束速度については次節で述べる。

2.3 ニュートン法（重根の場合）

2重根の場合 方程式 $f(x) = 0$ が $x = \xi$ で2重根を持つ場合には、 $f(\xi) = f'(\xi) = 0$ となる。このときは、(2.12) 式で分母の第1項が 0 となるので、第2項を無視することができず、次のようになる。

$$\begin{aligned} e_{n+1} - e_n &= -\frac{f'(\xi)e_n + \frac{1}{2}f''(\xi)e_n^2}{f'(\xi) + f''(\xi)e_n} \\ &= -\frac{1}{2}e_n \quad (f(\xi) = 0 \text{ より}) \end{aligned} \quad (2.15)$$

したがって、

$$e_{n+1} = \frac{1}{2}e_n \quad (2.16)$$

$$|e_n| \leq \frac{1}{2}|e_{n-1}| \leq \cdots \leq \left(\frac{1}{2}\right)^n e_0 \quad (2.17)$$

よって、収束率 $\frac{1}{2}$ の1次収束となり、これは二分法の収束率と同じである。

n 重根の場合 このときは

$$f(\xi) = f'(\xi) = f''(\xi) = \cdots = f^{(n-1)}(\xi) = 0 \quad (2.18)$$

となる。そこで、 $f(x)$ と $f'(x)$ のテイラー展開をする際、次のように高次の項を（0でない最初の項が出てくるまで）取ることで、2重根のときと同様に収束率を求めることができる。

$$f(x_n) = f(\xi) + f'(\xi)e_n + \frac{1}{2}f''(\xi)e_n^2 + \frac{1}{3!}f'''(\xi)e_n^3 + \cdots \quad (2.19)$$

$$f'(x_n) = f'(\xi) + f''(\xi)e_n + \frac{1}{2}f'''(\xi)e_n^2 + \frac{1}{3!}f^{(4)}(\xi)e_n^3 + \cdots \quad (2.20)$$

2.4 ニュートン法（連立方程式の場合）

ニュートン法は n 元連立非線形方程式

$$\mathbf{f}(\mathbf{x}) = \mathbf{0} \quad (2.21)$$

の解法に拡張できる。(2.21) 式を成分で表すと,

$$\begin{cases} f_1(x_1, \dots, x_n) = 0 \\ f_2(x_1, \dots, x_n) = 0 \\ \vdots \\ f_n(x_1, \dots, x_n) = 0 \end{cases} \quad (2.22)$$

となる。いま, 初期値を $\mathbf{x}^{(0)} = (x_1^{(0)}, \dots, x_n^{(0)})^t$ (ただし上付きの t はベクトルの転置を表す) とし, 方程式の各成分を $\mathbf{x}^{(0)}$ の周りで1次の項までテイラー展開すると,

$$\begin{cases} f_1(x_1, \dots, x_n) \simeq f_1(x_1^{(0)}, \dots, x_n^{(0)}) + \frac{\partial f_1}{\partial x_1}(x_1 - x_1^{(0)}) + \dots + \frac{\partial f_1}{\partial x_n}(x_n - x_n^{(0)}) = 0 \\ f_2(x_1, \dots, x_n) \simeq f_2(x_1^{(0)}, \dots, x_n^{(0)}) + \frac{\partial f_2}{\partial x_1}(x_1 - x_1^{(0)}) + \dots + \frac{\partial f_2}{\partial x_n}(x_n - x_n^{(0)}) = 0 \\ \vdots \\ f_n(x_1, \dots, x_n) \simeq f_n(x_1^{(0)}, \dots, x_n^{(0)}) + \frac{\partial f_n}{\partial x_1}(x_1 - x_1^{(0)}) + \dots + \frac{\partial f_n}{\partial x_n}(x_n - x_n^{(0)}) = 0 \end{cases} \quad (2.23)$$

となる。ただし, 微係数 $\frac{\partial f_i}{\partial x_j}$ ($i, j = 1, 2, \dots, n$) はすべて $\mathbf{x} = \mathbf{x}^{(0)}$ での値とする。(2.23) 式をまとめると,

$$\begin{aligned} \mathbf{f}(\mathbf{x}) &= \mathbf{f}(\mathbf{x}^{(0)}) + \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \dots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \dots & \frac{\partial f_n}{\partial x_n} \end{pmatrix}_{\mathbf{x}=\mathbf{x}^{(0)}} (\mathbf{x} - \mathbf{x}^{(0)}) \\ &= \mathbf{f}(\mathbf{x}^{(0)}) + \left. \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right|_{\mathbf{x}=\mathbf{x}^{(0)}} (\mathbf{x} - \mathbf{x}^{(0)}) \end{aligned} \quad (2.24)$$

となる。ここで,

$$\frac{\partial \mathbf{f}}{\partial \mathbf{x}} = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \dots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \dots & \frac{\partial f_n}{\partial x_n} \end{pmatrix}_{\mathbf{x}=\mathbf{x}^{(0)}} : \text{ヤコビアン行列} \quad (2.25)$$

$$\mathbf{x} - \mathbf{x}^{(0)} = \begin{pmatrix} x_1 - x_1^{(0)} \\ \vdots \\ x_n - x_n^{(0)} \end{pmatrix} \quad (2.26)$$

である。

計算手順は1変数の場合とほとんど同じで, $f'(x)$ による割り算が $\partial \mathbf{f} / \partial \mathbf{x}$ の逆行列を掛ける操作に置き換わる。第 n ステップの近似解から第 $n+1$ ステップの近似解を求める反復計算式は,

$$\mathbf{x}_{n+1} = \mathbf{x}_n - \left(\left. \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right|_{\mathbf{x}=\mathbf{x}^{(0)}} \right)^{-1} \mathbf{f}(\mathbf{x}) \quad (2.27)$$

となる。

第3章 関数の補間

物理の実験で、入力パラメータ x を変化させて物理量 y を測定することを考える。このとき、一般に測定は飛び飛びの点 $x_0, x_1, \dots, x_n$ でのみ行う。従って、実験を行わなかった点での物理量の値を後から推測したい場合には、 $x_0, x_1, \dots, x_n$ での測定値 $y_0, y_1, \dots, y_n$ を用いて入力と出力の関係 $y = f(x)$ を推定する必要がある。このようなときに使われるのが補間である。補間では、 $x_0, x_1, \dots, x_n$ における関数値がそれぞれ $y_0, y_1, \dots, y_n$ に一致するような関数を求める。

補間は、上記のような関数推定に使われるほか、関数の近似計算にも使われる。また、次章で述べる数値積分法の基礎となる。本章では、1次元のデータに対する補間法として、多項式近似に基礎を置いたラグランジュ補間、ニュートン補間、スプライン補間を学ぶ。

3.1 ラグランジュ補間

ラグランジュ補間とは、既知の $n+1$ 個のデータ点 $(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$ をすべて通る n 次関数により補間を行う方法である。 $x_i \neq x_j$ ($i \neq j$) のときこのような n 次関数がただ1つ存在することは、代数学の基本定理により保証される。この関数は次のようにして簡単に構成できる。

いま、

$$\begin{aligned}\varphi_i(x) &= \frac{(x-x_0)(x-x_1)\cdots(x-x_{i-1})(x-x_{i+1})\cdots(x-x_n)}{(x_i-x_0)(x_i-x_1)\cdots(x_i-x_{i-1})(x_i-x_{i+1})\cdots(x_i-x_n)} \\ &= \prod_{\substack{j=0 \\ j \neq i}}^n \frac{x-x_j}{x_i-x_j}\end{aligned}\quad (3.1)$$

という関数を考える。 $\varphi_i(x)$ は n 次関数であり、 x_j を代入すると

$$\varphi_i(x_j) = \begin{cases} 0 & (j \neq i) \\ 1 & (j = i) \end{cases}\quad (3.2)$$

となって、 $j = i$ の時のみ値 1 を取り、 $j \neq i$ の時は 0 を取ることがわかる。次に

$$f_n(x) = \sum_{i=0}^n y_i \varphi_i(x_i)\quad (3.3)$$

という関数を考えると、 $f_n(x)$ は n 次関数で、

$$f_n(x_j) = y_j \quad (j = 0, 1, \dots, n) \quad (3.4)$$

より、 $(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$ をすべて通る。

こうして得られる補間関数 $f_n(x)$ をラグランジュの補間多項式と呼ぶ。ラグランジュ補間多項式は、考え方は簡単であるが、任意の点 x での関数値を求めるのに $O(n^2)$ の計算量が必要である。また、新しいデータ点を加えた場合に、式を最初から作り直さなければならないという欠点を持つ。

3.2 ニュートン補間

新しいデータ点を加えた時に、式を最初から作り直さなければならないというラグランジュ補間の欠点を補うのがニュートン補間である。ニュートン補間もラグランジュ補間と同様、 $(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$ をすべて通る n 次関数による補間であり、多項式としてはラグランジュ補間と同じものであるが、補間を計算するアルゴリズムが異なり、データ点を付け加えた場合の補間多項式の再計算が容易になっている。

ニュートン補間では、点 (x_0, y_0) を通る 0 次多項式 $f_0(x) = a_0$ から始め、多項式の次数を 1 ずつ上げていくことにより、通る点を $(x_1, y_1), (x_2, y_2), \dots$ と 1 個ずつ増やしていく。補間多項式の具体的な定め方は次のようになる。

$$n = 1 : f_0(x) = a_0$$

$$\text{ただし } a_0 = y_0$$

$$n = 2 : f_1(x) = f_0(x) + a_1(x - x_0)$$

$$\text{ただし } a_1 = \frac{y_1 - f_0(x_1)}{x_1 - x_0}$$

$$n = 3 : f_2(x) = f_1(x) + a_2(x - x_0)(x - x_1)$$

$$\text{ただし } a_2 = \frac{y_2 - f_1(x_2)}{(x_2 - x_0)(x_2 - x_1)}$$

⋮

$$n = i + 1 : f_i(x) = f_{i-1}(x) + a_i(x - x_0)(x - x_1) \cdots (x - x_{i-1}) \quad (3.5)$$

$$\text{ただし } a_i = \frac{y_i - f_{i-1}(x_i)}{(x_i - x_0)(x_i - x_1) \cdots (x_i - x_{i-1})} \quad (3.6)$$

いま、 f_{i-1} が点 $x_0, x_1, \dots, x_{i-1}$ における $i-1$ 次の多項式補間になっていると仮定すると、 $i = 1$ のときこの仮定は正しい。第 $i+1$ 番目の点 x_i を加えるときを考えると、 $f_{i-1}(x)$ に足す項は $x = x_0, x_1, \dots, x_{i-1}$ で 0 になるよう選んでいるので、 $f_i(x)$ はこれらの点で $f_{i-1}(x)$ と等しい値を持つ。更に、 a_i は $f_i(x_i) = y_i$ となるよう選んである（確認せよ）。したがって、 $f_i(x)$ は点

$x_0, x_1, \dots, x_i$ における i 次の多項式補間になっている。こうして上記のアルゴリズムの正当性が示された。

上記のアルゴリズムでは、各ステップで $f_{i-1}(x_i)$ の計算が必要である。また、求める補間式ができた場合、その任意の点での値を求めるにも $f_n(x)$ の計算が必要である。そこで、 $f_n(x)$ の効率的な計算法について考える。アルゴリズムより、 $f_n(x)$ は明らかに次の形に書ける。

$$\begin{aligned} f_n(x) = & a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) \\ & + \cdots + a_n(x - x_0) \cdots (x - x_{n-1}) \end{aligned} \quad (3.7)$$

この式に直接 x の値を代入して計算すると、計算量は $O(n^2)$ となる。ところが、次の補助変数 z_i を導入すると、この計算量を削減できる。

$$\begin{aligned} z_i = & a_i + a_{i+1}(x - x_i) + a_{i+2}(x - x_i)(x - x_{i+1}) \\ & + \cdots + a_n(x - x_i) \cdots (x - x_{n-1}) \end{aligned} \quad (3.8)$$

z_i の定義式より、

$$z_0 = f_n(x) \quad (3.9)$$

であるから、この z_0 が計算したい量ということになる。また、 z_i を変形すると

$$\begin{aligned} z_i &= a_i + (x - x_i) \{a_{i+1} + a_{i+2}(x - x_{i+1}) \\ &\quad + \cdots + a_n(x - x_{i+1}) \cdots (x - x_{n-1})\} \\ &= a_i + (x - x_i) \cdot z_{i+1} \end{aligned} \quad (3.10)$$

となる。 $z_n = a_n$ の値は既知であるので、 z_0 は後ろ向きの漸化式

$$\begin{cases} z_n = a_n \\ z_i = a_i + (x - x_i) \cdot z_{i+1} \end{cases} \quad (3.11)$$

により $O(n)$ の計算量で求めることができる。以上より、ニュートン補間はラグランジュ補間と比べ、データ点の追加が容易であり、かつ任意の点 x で補間多項式の値を求めるための計算量が少ない補間法だと言える。

3.2.1 ラグランジュ補間の誤差

(第4回の授業へ)

3.2.2 スプライン補間

(第4回の授業へ)