第3章 補間と補外
(interpolation and extrapolation)

いなか*

11月11日=ポッキー or プリッツの日

0 はじめに

補間・補外とは、有限個のデータ:$(x_1, f(x_1)), (x_2, f(x_2)), \ldots, (x_N, f(x_N))$ $(x_1 < x_2 < \cdots < x_N)$ だけが与えられたとき、適当な関数形でモデル化をし、任意の点 x おける $f(x)$ の値を知ることである。

- 補間と補外の違いは？
  - 補間：x がデータの範囲内 $(x_1 < x < x_N)$
  - 補外：x がデータの範囲外 $(x < x_1 || x_N < x)$
- 適当な関数形って？
  - 多項式
  - 有理式
  - 三角関数 etc
- いつでも上手くいくものなの？
  - 上手くいかない場合もある (図1)
    → 誤差の評価が必要
    → 確実な方法はない
- 補間の精度と標本点の数の関係は？
  - 標本点の個数から1引いたものを補間の次数という
  - 次数を上げても正確になるとは限らない
    → 特に多項式補間の場合
- 一般的に補間よりも補外の方が危険度が非常に高い

*inaka@am.ics.keio.ac.jp

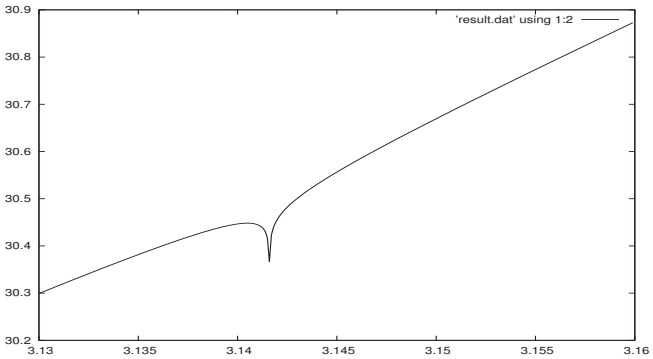

図 1: $f(x) = 3x^2 + \frac{1}{\pi^4}\ln[(\pi - x)^2] + 1$

1 多項式による補間と補外

ここでは、x の多項式を用いることで補間を行う方法を紹介する。

- 任意の N 個の点を通る $N-1$ 次関数はただ一つ存在する
- ↑の場合の補間多項式は Lagrange の公式で与えられる

Lagrange の公式

$$P(x) = \frac{(x-x_2)(x-x_3)\dots(x-x_N)}{(x_1-x_2)(x_1-x_3)\dots(x_1-x_N)}y_1 + \frac{(x-x_1)(x-x_3)\dots(x-x_N)}{(x_2-x_1)(x_2-x_3)\dots(x_2-x_N)}y_2 + \dots + \frac{(x-x_1)(x-x_2)\dots(x-x_{N-1})}{(x_N-x_1)(x_N-x_2)\dots(x_N-x_{N-1})}y_N \quad (1)$$

- Lagrange の公式は、プログラム的にも誤差の評価ができない点でも実用的でない
- 数値的には、Neville のアルゴリズムや Aitken のアルゴリズムが向いている[1]
- Neville のアルゴリズム

  1: 点 (x_1, y_1) を通る 0 次の多項式を P_1 とする $(P_1 = y_1)$
  2: 同様に $P_2, P_3, \dots, P_N$ を求める
  3: 次に点 $(x_1, y_1), (x_2, y_2)$ を通る 1 次の多項式を P_{12} とする
  4: 同様に $P_{23}, P_{34}, \dots, P_{(N-1)N}$ を求める
  5: 以下同様に繰り返して、$P_{123\dots N}$ を求める

- Neville のアルゴリズムは図 2 にあるとおり親と子の関係で表せる

Neville のアルゴリズム

$$P_{i(i+1)\dots(i+m)} = \frac{(x - x_{i+m})P_{i(i+1)\dots(i+m-1)} + (x_i - x)P_{(i+1)(i+2)\dots(i+m)}}{x_i - x_{i+m}} \quad (2)$$

[1] Aitken のアルゴリズムは現在使用されていない

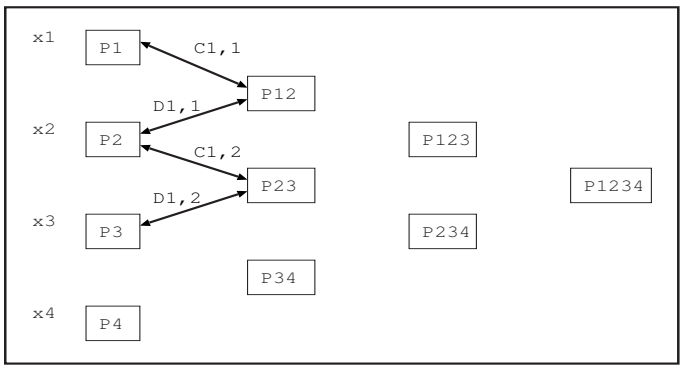

図 2: Neville のアルゴリズム

ここで図 2 中の親子の差に注目して、誤差を以下のように定義する。

$$C_{m,i} \equiv P_{i\ldots(i+m)} - P_{i\ldots(i+m-1)} \tag{3}$$
$$D_{m,i} \equiv P_{i\ldots(i+m)} - P_{(i+1)\ldots(i+m)} \tag{4}$$

このとき、差 C、D は漸化式で以下のように表すことができる。

差の漸化式

$$C_{m+1,i} \equiv \frac{(x_i - x)(C_{m,i+1} - D_{m,i})}{x_i - x_{i+m+1}} \tag{5}$$
$$D_{m+1,i} \equiv \frac{(x_{i+m+1} - x)(C_{m,i+1} - D_{m,i})}{x_i - x_{i+m+1}} \tag{6}$$

- 任意の y_i に差 C、D を順次加えていくことで最終的に $P_{1\ldots N}$ を得る
- 最後に加えた $C(D)$ は誤差の指標とできる

2 有理関数による補間と補外

多項式の近似が上手くいかなくても有理関数で上手く近似できることがある。例えば点 $(x_i, y_i)\ldots(x_{i+m}, y_{i+m})$ を補間する有理関数は、

$$R_{i(i+1)\ldots(i+m)} = \frac{P_\mu(x)}{Q_\nu(x)} = \frac{p_0 + p_1 x + \cdots + p_\mu x^\mu}{q_0 + q_1 x + \cdots + q_\nu x^\nu} \tag{7}$$
$$m + 1 = \mu + \nu + 1 \tag{8}$$

と表せる。

- 補間したい関数が極を持つとき
  → 有理関数による補間が多項式の補間より優れている
- 有理関数の近似を行う場合、パデ近似を用いる

定義 ある分母が m 次、分子が n 次の有理関数の級数展開が、近似すべき関数の級数展開と $n+m$ 次まで一致する場合、$m+n$ 次のパデ近似と呼ぶ。

1: $m+n$ 次の近似を行う場合、分母の次数 m、分子の次数 n の有理関数 $r(x)$ を用意する

2: 近似する関数 $f(x)$ を級数展開する

3: $k=0,\ldots n+m$ として、以下が成り立つように有理数の係数を求める
$f^{(k)}(0)=r^{(k)}(0)$

- Bulirsch-Store アルゴリズムは、Neville アルゴリズム同様に漸化式の形で有理関数を補間できる

Bulirsch-Store アルゴリズム

$$R_{i(i+1)\ldots(i+m)}=R_{(i+1)\ldots(i+m)}+\frac{R_{(i+1)\ldots(i+m)}-R_{i\ldots(i+m-1)}}{(\frac{x-x_i}{x-x_{i+m}})(1-\frac{R_{(i+1)\ldots(i+m)}-R_{i\ldots(i+m-1)}}{R_{(i+1)\ldots(i+m)}-R_{(i+1)\ldots(i+m-1)}})-1} \tag{9}$$

- Bulirsch-Store アルゴリズムは、$m+1$ 個の点を通る関数を m、$m-1$ 個の点を通るものから生成する
- 出発点は $R_i=y_i$、$R\equiv 0$ とする

Neville のアルゴリズム同様に親子の差を、

$$C_{m,i} \equiv R_{i\ldots(i+m)}-R_{i\ldots(i+m-1)} \tag{10}$$
$$D_{m,i} \equiv R_{i\ldots(i+m)}-R_{(i+1)\ldots(i+m)} \tag{11}$$

とすると、$C_{m+1,i}-D_{m+1,i}=C_{m,i+1}-D_{m,i}$ となる (らしい) ので、

差の漸化式

$$C_{m+1,i} = \frac{(\frac{x-x_i}{x-x_{i+m+1}})D_{m,i}(C_{m,i+1}-D_{m,i})}{(\frac{x-x_i}{x-x_{i+m+1}})D_{m,i}-C_{m,i+1}} \tag{12}$$
$$D_{m+1,i} = \frac{C_{m,i+1}(C_{m,i+1}-D_{m,i})}{(\frac{x-x_i}{x-x_{i+m+1}})D_{m,i}-C_{m,i+1}} \tag{13}$$

- 多項式補間と同様に $P_{1\ldots N}$ を得ることができる

3　3 次スプライン補間

スプライン補間とは補間する領域をデータ間隔に区切り、各領域に対して低次の多項式で補間する手法。ここでは、よく使用される (らしい)3 次多項式による 3 次スプライン補間を説明…したいのですが、式変形が結構省かれているので
http://www.akita-nct.ac.jp/yamamoto/lecture/2004/5E/interpolation /text/html/node3.html
が参考になると思います。(というか参考にしました)

- N 個の点 $(x_i, y_i) \quad (i = 1, \ldots N)$ が与えられているとする
- このとき補間すべき領域 (データ間) は $N - 1$ 個となる
- 全ての領域を 3 次多項式で補間すると未知数は $4 \times (N - 1)$ 個となる
- **条件 1：全てのデータを通る=領域両端の値が決っている**
  $\rightarrow 2 \times (N - 1)$ 個の方程式ができる
- **条件 2：領域境界の 1 階導関数は連続である**
  $\rightarrow N - 2$ 個の方程式ができる
- **条件 3：領域境界の 2 階導関数は連続である**
  $\rightarrow N - 2$ 個の方程式ができる
- $4(N - 1) - 2(N - 1) + (N - 2) + (N - 2) = 2$ 個方程式が足りない
- 通常 x_1, x_N での境界条件を用いた以下のどちらかの方法をとる
  1. **点 x_0, x_N における 2 階導関数の値の片方または両方を 0 とする**
  2. **点 x_0, x_N における 1 階導関数の値の片方または両方を 2 階導関数から指定する**
- 上記 1 の方法は、自然 3 次スプライン補間と呼ばれる
- 以上から補間に必要な係数がすべて分かるので、任意の値を補間できる

実際にプログラミングをするために、式を当てはめていきます。

3 次スプライン補間ができるまで

前提 $y_i = f(x_i)$ として $i = 1, \ldots N$ までのデータが与えられている
また、$p_i = g(x_i)$ として、同様に $i = 1, \ldots N$ までのデータが与えられている

1： 区間 $[x_j, x_{j+1}]$ を線形に補間したとすると、

$$A \equiv \frac{x_{j+1} - x}{x_{j+1} - x_j} \qquad B \equiv \frac{x_{j+1} - x}{x - x_j} \tag{14}$$

を用いて表すと

$$y = Ay_j + By_{j+1} \tag{15}$$

2： 3 次補間をしたいので、p_j, p_{j+1} を係数とする線形独立な 3 次多項式 C, D を用いて拡張すると、

$$y = Ay_j + By_{j+1} + Cp_i + Dp_{j+1} \tag{16}$$

3： 点 x_j, x_{j+1} での式 16 の値が p_j, p_{j+1} によらず y_j, y_{j+1} になるように少し考えると下の様になるそうです (少しじゃないと思いたい)

$$C \equiv \frac{1}{6}(A^3 - A)(x_{j+1} - x_j)^2 \qquad D \equiv \frac{1}{6}(B^3 - B)(x_{j+1} - x_j)^2 \tag{17}$$

4： A, B, C, D の定義を用いると $y = f(x)$ の 1 階導関数と 2 階導関数は

$$\frac{dy}{dx} = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{3A^2 - 1}{6}(x_{j+1} - x_j)p_j + \frac{3B^2 - 1}{6}(x_{j+1} - x_j)p_{j+1} \tag{18}$$

$$\frac{d^2y}{dx^2} = Ap_j + Bp_{j+1} \tag{19}$$

5： 式 19 から x_j, x_{j+1} のときを考えると、$p_j = y_j'', p_{j+1} = y_{j+1}''$ であることがわかるので、

$$\frac{d^2y}{dx^2} = Ay_j'' + By_{j+1}'' \tag{20}$$

6： 以降はこの導関数を条件に合うように解けば、補間多項式の係数が得られる
(ex)1 次導関数の方程式

$$(x_j - x_{j-1})y_{j-1}'' + 2(x_{j+1} - x_{j-1})y_j'' + (x_{j+1} - x_j)y_{j+1}'' = 6\left(\frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{y_j - y_{j-1}}{x_j - x_{j-1}}\right) \tag{21}$$

7： ここで、以下のようにおく

$$h_j = x_{j+1} - xj \quad v_j = 6\left(\frac{y_{j+1} - y_j}{h_j} - \frac{y_j - y_{j-1}}{h_{j-1}}\right) \tag{22}$$

8： 式 21 は 3 重対角行列の問題となる

$$\begin{pmatrix} 2(h_1+h_2) & h_2 & & & & & 0 \\ h_2 & 2(h_2+h_3) & h_3 & & & & \\ & h_3 & 2(h_3+h_4) & h_4 & & & \\ & & & \ddots & & & \\ & & & h_{j-1} & 2(h_{j-1}+h_j) & h_j & \\ & & & & & \ddots & \\ 0 & & & & & h_{N-1} & 2(h_{N-1}+h_N) \end{pmatrix} \begin{pmatrix} y_1'' \\ y_2'' \\ y_3'' \\ \vdots \\ y_j'' \\ \vdots \\ y_{N-1}'' \end{pmatrix} = (v_1, v_2, v_3, \ldots, v_j, \ldots, v_{N-1}) \tag{23}$$

4 整列した表の探索法

- ある点 x における値を知りたい
  $\rightarrow$ その x がデータのどの区間にあるかが重要
- 与えられた x_i が昇順もしくは降順に整列していると仮定すると以下のような探索法がある
  - 二分法：計算量は $\log_2 n$
    上限もしくは下限を更新しながら、2 等分していく
  - ハントによる探索：計算量は最良で二分法の $\log_2 n$ 速い、最悪で 2 倍遅い
    探索距離を増分していき、目的の値を越えたら二分法にシフト
- ハントによる探索は、x_i の値の変化が少ないときに有効

5 補間多項式の係数

- 少数個の点を通る補間多項式については、以下のような場合、係数を知りたいことがある。
  - 関数の補間値とそのいくつかの導関数を求めたいとき
  - 関数同士の畳込みを求めたいときで、一方の関数は表の形で与えられており、もう一方はモーメントが解析的にわかっているとき
- 補間多項式の係数は補間値よりも一般的に精度が落ちる
- 補間多項式と最良近似多項式は異なる
- 補間多項式
  - 標本点と係数が同数である
  - 補間多項式は全ての標本点を通る
  - 標本点に誤差が含まれていると、誤差が拡大され標本点間で振動してしまう
- 最良近似多項式
  - 近似は平滑化の過程
  - 係数の個数は標本点より少ない
  - 誤差に強く安定して係数を当てはめることができる
- 補間多項式：$y = c_0 + c_1 x + c_2 x^2 + \cdots + c_N x^N$ の係数を求めるには？
  - Vandermonde の連立方程式：$O(N^2)$

$$\begin{bmatrix} 1 & x_0 & x_0^2 & \dots & x_0^N \\ 1 & x_1 & x_1^2 & \dots & x_1^N \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_N & x_N^2 & \dots & x_N^N \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_N \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_N \end{bmatrix} \tag{24}$$

- 補間値から帰納的に求める：$O(N^3)$

- Vandermonde の方法では不安定である
- また、どちらも N が大きくなると上手く働かない

問題点 1：$x = 0$ が標本点の範囲 (またはその近く) にないとき
→ 一般的に係数は非常に大きくなる
→ 誤差が増えると、有効桁数が減ってしまう

問題点 2：滑らかな関数を高次の多項式で補間するとき
→ データに無理やり合致させる
→ 拘束点の間で振動する傾向が現れる

- 著者によると係数を求めることは勧められないらしい

6　2 次元以上の補間

n 次元の関数の補間を考える。$\overrightarrow{x} = \{x_1, x_2, \ldots, x_N\}$ としたとき $y = f(\overrightarrow{x})$ となるので、$(\overrightarrow{x}, y)$ の組が与えられれば良い。今回は 2 次元の場合について考える。
関数値の配列 $ya[j][k] \quad 1 \le j \le m, 1 \le k \le n$ と長さ m の配列 $x1a$ 長さ n の配列 $x2a$ が与えられたとする。このとき、$ya[j][k] = y(x1a[j], x2a[k])$ の関係がある。
双 1 次補間 (bilinear interpolation)

- 最も簡単な補間
- 境界上での補間関数の勾配は不連続
- 補間する点 (x_1, x_2) の周囲 4 点を考える

$$x1a[j] \le x_1 \le x1a[j+1]$$
$$x2a[k] \le x_2 \le x2a[k+1]$$

と、j, k を定め、

$$\begin{aligned} y_1 &\equiv ya[j][k] \\ y_2 &\equiv ya[j+1][k] \\ y_3 &\equiv ya[j+1][k+1] \\ y_4 &\equiv ya[j][k+1] \end{aligned}$$

とする。このとき双 1 次補間の公式は、

双 1 次補間

$$t \equiv (x_1 - x1a[j])/(x1a[j+1] - x1a[j]) \tag{25}$$
$$u \equiv (x_2 - x2a[k])/(x2a[k+1] - x2a[k]) \tag{26}$$

とおいて、

$$y(x_1, x_2) = (1-t)(1-u)y_1 + t(1-u)y_2 + tuy_3 + (1-t)uy_4 \tag{27}$$

精度を上げる高次補間

1. x_2 方向へ補間をし、各 j について $f(x1a[j], x_2)$ の値を求める
2. もとめた値を使って x_1 方向へ補間をし、(x_1, x_2) の値を補間する
3. 誤差は x_1 方向のときのものを使用する

滑らかさを上げる高次補間：双 3 次補間 (bicubic interpolation)

- 双 1 次補間と同様に、$\overrightarrow{x}$ と $f(\overrightarrow{x})$ の組を与える
- 式 25, 26 から t, u を定義する
- さらに点 $\overrightarrow{x}$ における関数値以外に以下の導関数も与える
  - $\frac{\partial y}{\partial x_1} \equiv y_{,1}$
  - $\frac{\partial y}{\partial x_2} \equiv y_{,2}$
  - $\frac{\partial^2 y}{\partial x_1 \partial x_2} \equiv y_{,12}$
- これらの導関数は正確である**必要はない**
  → 方法自体に滑らかさが組み込まれている
  → もちろん正確な方が補間値も正確になる
- 与えられた関数値、導関数と**謎の表**から 16 個の c_{ij} を求める
- 求まった c_{ij} および t, u から各関数値と導関数を求める

双 3 次公式

$$y(x_1, x_2) = \sum_{i=1}^{4}\sum_{j=1}^{4} c_{ij} t^{i-1} u^{j-1} \tag{28}$$

$$y_{,1}(x_1, x_2) = \sum_{i=1}^{4}\sum_{j=1}^{4} (i-1) c_{ij} t^{i-2} u^{j-1} \tag{29}$$

$$y_{,2}(x_1, x_2) = \sum_{i=1}^{4}\sum_{j=1}^{4} (j-1) c_{ij} t^{i-1} u^{j-2} \tag{30}$$

$$y_{,12}(x_1, x_2) = \sum_{i=1}^{4}\sum_{j=1}^{4} (i-1)(j-1) c_{ij} t^{i-2} u^{j-2} \tag{31}$$

滑らかさを上げる高次補間：双 3 次スプライン (bicubic spline)

- 双 3 次補間の特別な場合
- 補間関数の形は上記双 3 次公式同様
- しかし、通常関数値を求めるには"精度を上げる場合"と同様に求める
- どこまで前計算しておくかは個人の自由