数値計算 Class-7 演習

21T2166D 渡辺大樹

2023/07/23

1　演習内容

Class-7 ではいくつかの xy 平面上のデータからそのデータをすべて通るなめらかな曲線を描くための手法、スプライン関数についてコードを実装し、実際に動かしていく。

スプライン関数は n 個あるデータの隣り合うデータからデータの区間にそれぞれ同じ次数の多項式を定義しつなぎ合わせた、区分的多項式である。区間の多項式の次数が n であるとき、n 次のスプライン関数という。この関数はその定義にあるようになめらかな曲線である必要がある。

関数がなめらかであるというのはその区間の両端で関数が連続かつ微分可能である必要がある。n 次のスプライン関数では $n-1$ 次の微分係数が区間の両端で隣接する区間と一致する必要がある。

以下では 3 次のスプライン関数 $S(x)$ を考えていく。

xy 平面上に $n+1$ 個のデータ $(x_0, y_0), \cdots, (x_n, y_n)$(ただし $x_{i-1} < x_i, \quad (i = 1, 2, \cdots, n)$) が与えられてる。この隣り合うデータがつくる n 個の小区間 $l_i = [x_{i-1}, x_i]$ 上の 3 次関数 $S_i(x)$ を次の二つの条件を満たすように設定する。

1. $S_i(x)$ は区間の両端で連続、すなわちデータ点を通る。

$$S_i(x_{i-1}) = y_{i-1}$$
$$S_i(x_i) = y_i$$

2. 隣り合う関数がその節点となるデータ点で 2 階微分までの係数が一致する。

$$S_i'(x_i) = S_{i+1}'(x_i)$$
$$S_i''(x_i) = S_{i+1}''(x_i)$$

この $S_i(x)$ をつないで区間 $[x_0, x_n]$ の関数にすることで、3 次のスプライン関数 $S(x)$ が得られる。

スプライン関数を求めるアルゴリズムは以下のソースコード 1 で実装される。

ソースコード 1　spline.c

```
1 /*************************************************************************/
2 /* スプライン関数の決定(1次関数法)spline.c */
```

```
/* スプライン関数を求める。さらに等分刻みのx の値に */
/* 対する関数値を求めて結果を数表の形で出力する。*/
/*************************************************************************/
#include <stdio.h>
#include <math.h>
#define N 10
    int
    main(void)
{
    int i, j, k, m, n, kz, z, lp;
    double A[N], B[N], C[N + 2], D[N], h[N], x[N];
    double y[N], u[N], p[N + 2][N + 3];
    double g, q, xw, s, f;
    char qq, zz;
    /* h[]:hj u[]:uj A[],B[],C[],D[]:Aj,Bj,Cj,Dj */
    /* p[][]: 連立方程式の係数配例 */
    /* x,y, に関するデータの入力 */
    while (1)
    {
        printf("スプライン関数の決定(1次関数法) \n\n");
        printf("データの個数は何個ですか？ (2<m<10) m = ");
        scanf("%d%c", &m, &zz);
        if ((m <= 2) || (10 <= m))
            continue;
        n = m - 1;
        for (i = 0; i <= n; i++)
        {
            printf("x(%d)= ", i);
            scanf("%lf%c", &x[i], &zz);
            printf("y(%d)= ", i);
            scanf("%lf%c", &y[i], &zz);
        }
        printf("全区間の左端点における 1次微分係数は？ ");
        scanf("%lf%c", &C[1], &zz);
        printf("全区間の右端点における 1次微分係数は？ ");
        scanf("%lf%c", &C[n + 1], &zz);
        printf("\n 正しく入力ましたか？ (y/n) ");
        scanf("%c%c", &qq, &zz);
        if (qq == 'y')
            break;
    }
    for (i = 1; i <= n; i++)
```

```
45      {
46          D[i] = y[i - 1];
47          h[i] = x[i] - x[i - 1];
48          u[i] = (y[i] - y[i - 1]) / h[i];
49      }
50      for (i = 1; i <= n + 1; i++)
51      {
52          for (j = 1; j <= n + 2; j++)
53          {
54              p[i][j] = 0.0;
55          }
56      }
57      /* 漸化式から得られる連立方程式の係数を*/
58      /* 配列p に入れる */
59      p[1][1] = 1.0;
60      p[n + 1][n + 1] = 1.0;
61      p[1][n + 2] = C[1];
62      p[n + 1][n + 2] = C[n + 1];
63      for (j = 2; j <= n; j++)
64      {
65          p[j][j - 1] = h[j];
66          p[j][j] = 2 * (h[j - 1] + h[j]);
67          p[j][j + 1] = h[j - 1];
68          p[j][n + 2] = 3 * (h[j] * u[j - 1] + h[j - 1] * u[j]);
69      }
70      /* ガウス･ジョルダン（掃き出し）法により*/
71      /* 連立方程式を解く */
72      for (i = 1; i <= n + 1; i++)
73      {
74          q = p[i][i];
75          for (j = 1; j <= n + 2; j++)
76          {
77              p[i][j] = p[i][j] / q;
78          }
79          for (k = 1; k <= n + 1; k++)
80          {
81              g = p[k][i];
82              if (k != i)
83              {
84                  for (j = 1; j <= n + 2; j++)
85                  {
86                      p[k][j] = p[k][j] - g * p[i][j];
```

```
87                  }
88              }
89          }
90      }
91      /*上で得られた解を配列c に入れる*/
92      for (j = 1; j <= n; j++)
93      {
94          C[j] = p[j][n + 2];
95      }
96      /*B1,B2,...,Bn を求める*/
97      for (j = 1; j <= n - 1; j++)
98      {
99          B[j] = (3 * u[j] - 2 * C[j] - C[j + 1]) / h[j];
100     }
101     B[n] = -(3 * u[n - 1] - C[n - 1] - 2 * C[n]) / h[n - 1];
102     /*A1,A2,...,An を求める*/
103     for (j = 1; j <= n - 1; j++)
104     {
105         A[j] = (B[j + 1] - B[j]) / (3 * h[j]);
106     }
107     A[n] = (u[n] - h[n] * B[n] - C[n]) / (h[n] * h[n]);
108
109     /*結果を出力する*/
110     printf("名区間のスプライン関数の係数を出力します\n");
111     printf("(X-Xj)の降べきの順 (係数Aj,Bj,Cj,Dj の値) \n");
112     for (i = 1; i <= n; i++)
113     {
114         printf("S%d(x)=", i);
115         printf("(%6.3lf)(x-%6.3lf)^3+", A[i], x[i - 1]);
116         printf("(%6.3lf)(x-%6.3lf)^2+", B[i], x[i - 1]);
117         printf("(%6.3lf)(x-%6.3lf)+", C[i], x[i - 1]);
118         printf("(%6.3lf)\n", D[i]);
119     }
120     /*求めたスプライン関数を使って補間値を求める*/
121     printf("x 座標の範囲を何等分して補間値を求めますか？ \n");
122     scanf("%d", &kz);
123     xw = (x[n] - x[0]) / kz;
124     for (z = 1, s = x[0], lp = 0; lp <= kz; lp++)
125     {
126         if (z >= n)
127             z--;
128         f = A[z] * pow((s - x[z - 1]), 3.0) + B[z] * pow((s - x[z - 1]),
```

```
            2.0) + C[z] * (s - x[z - 1]) + D[z];
129         printf("%10.6lf %10.6lf\n", s, f);
130         s = s + xw;
131         if (s > x[z])
132             z++;
133     }
134     return 0;
135 }
```