Вычислительная математика
Лабораторная работа 1

Ярослав Пылаев, гр. 3530904/80001

March 5, 2020

1 Задание

Вариант 14. Для $1 \leq x \leq 4$ с $h = 0.375$ вычислить значения

$$f(x) = \int_0^{20} \frac{dz}{e^z(z+x)},$$

используя для вычисления интеграла программу `QUANC8`. По полученным точкам построить сплайн-функцию и полином Лагранжа 8-й степени. Сравнить значения обеих аппроксимаций в точках

$$x_k = 1.1875 + 0.375k, \quad (k = 0, 1, \ldots, 7).$$

1.1 Цель работы

Вычислить значение заданного интеграла и исследовать точность аппроксимирующих функций, полученных

- сплайн-интерполяцией,
- построением интерполяционного полинома Лагранжа.

1.2 Задачи

1. Используя программу `QUANC8` определить значение интеграла.
2. При помощи программ `SPLINE` и `SEVAL` построить сплайн-функцию.
3. Построить интерполяционный полином Лагранжа.
4. Сравнить полученные значения в точках x_k, дополнительно для определения точности аппроксимаций для этих же точек применив программу `QUANC8`.

2 Реализация

2.1 QUANC8

Listing 1: QUANC8

```
const int X_LIMIT_1 = 1;
const int X_LIMIT_2 = 4;
const double H = 0.375;

const double A = 0;
const double B = 20;

int main()
{
//other INPUT parameters
  const double epsabs = 0.0;
  const double epsrel = 1.0e-10;

//OUTPUT parameters
  double result = 0.0;
  double errest = 0.0;
  int nofun = 0;
  double posn = 0.0;
  int flagQuanc = 0;

  const int n = 1 + static_cast<int>(abs(X_LIMIT_1 -
        - X_LIMIT_2) / H);

  double arrQuancRes[n] = { 0 };

  for (int i = 0; i < n; ++i) {
    quanc8(fun, A, B, epsabs, epsrel, &result,
          &errest, &nofun, &posn, &flagQuanc); //QUANC8
    arrQuancRes[i] = result;
  }
  ......................
```

Listing 2: Подынтегральная функция

```
double fun(double z)
{
  double x = X_LIMIT_1;
  static int i = -1;
  if (z == A) {
    x = X_LIMIT_1 + (++i * H); //compute x
  }
  else {
    x = X_LIMIT_1 + (i * H);
  }
  return { 1 / (exp(z) * (z + x)) };
}
```

2.1.1 Результат

x	1	1.375	1.75	2.125	2.5	2.875	3.25	3.625	4
$f(x)$	0.5963	0.4774	0.3998	0.3448	0.3035	0.2713	0.2454	0.2241	0.2063

`FLAG QUANC8` $= 0$

2.1.2 Комментарии

Так как подынтегральная функция зависит от двух переменных, то программа `QUANC8` работает относительно переменной z, а изменение x учитывается в самой функции (используется статическая переменная для расчета).

Результаты программы являются разумными, потому что при увеличении переменной x, находещейся в знаменателе, значение функции уменьшается.

2.2 Сплайн-интерполяция

Listing 3: SPLINE-SEVAL

```
1   ......................
2   const int k_min = 0;
3   const int k_max = 7;
4   const int nK = k_max - k_min + 1;
5
6   double arrSplineRes[nK] = { 0 };
7
8   //compute X_k values
9   double arrXKPoints[nK];
10  for (int i = 0; i < nK; ++i) {
11    arrXKPoints[i] = 1.1875 + 0.375 * i;
12  }
13
14  double arrB[nK] = { 0 };
15  double arrC[nK] = { 0 };
16  double arrD[nK] = { 0 };
17
18  int flagSpline = 0;
19  int last = 0;
20
21  spline(n, 1, 1, 1, 1, arrXPoints, arrQuancRes,
22        arrB, arrC, arrD, &flagSpline); //SPLINE
23
24  for (int i = k_min; i < k_max + 1; ++i) {
25    arrSplineRes[i] = seval(n, arrXKPoints[i], arrXPoints,
26          arrQuancRes, arrB, arrC, arrD, &last); //SEVAL
27  }
28  ......................
```

2.2.1 Результат

x_k	1.1875	1.5625	1.9375	2.3125	2.6875	3.0625	3.4375	3.8125
$f(x)$	0.5963	0.4193	0.3744	0.3213	0.2879	0.2531	0.2509	0.2268

FLAG SPLINE $= 0$

2.3 Интерполяционный полином Лагранжа

Listing 4: Построение полинома Лагранжа

```
1  ......................
2  double arrLagrRes[nK] = { 0 };
3
4  for (int i = k_min; i < k_max + 1; ++i) {
5    arrLagrRes[i] = lagrange(n, arrXPoints, arrQuancRes,
6          nK - 1, arrXKPoints[i]); //LAGRANGE
7  }
8  ......................
```

Listing 5: lagrange.cpp

```
1  double lagrange(int n, double arrPoints[],
2        double arrFunctions[], int m, double x)
3  {
4    if (n < m) {
5      throw std::invalid_argument("There are not enough points");
6    }
7    if (n <= 0) {
8      throw std::invalid_argument("Tables are empty");
9    }
10   //find nodes satisfying the condition of interpolation
11   int h = 0;
12   for (int i = 0; i < n; i++) {
13     if ((arrPoints[i] > x) && (i >= m)) {
14         h = i + 1 - m;
15         break;
16     }
17   }
18   //compute
19   double q = 0.0;
20   for (int i = h; i < h + m; i++) {
21     double p = 1.0;
22     for (int j = h; j < h + m; j++) {
23       if (j != i) {
24         p *= x - arrPoints[j];
25         p /= arrPoints[i] - arrPoints[j];
26       }
27     }
28     p *= arrFunctions[i];
```

```
29      q += p;
30    }
31    return q;
32  }
```

2.3.1 Результат

x_k	1.1875	1.5625	1.9375	2.3125	2.6875	3.0625	3.4375	3.8125
$f(x)$	0.5293	0.4350	0.3702	0.3228	0.2864	0.2577	0.2342	0.2148

2.3.2 Комментарии

Для построения интерполяционного полинома Лагранжа была использована программа `lagrange.cpp`, написанная еще в 3-м семестре в рамках курса по вычислительной математике, которая самостоятельно способна выбирать наиболее подходящие узлы интерполирования. Даже несмотря на то, что в решаемой задаче выбор узлов не предоставляется (9 точек, требуемая степень - 8), упомянуть о свойстве программы имеет смысл.

2.4 Дополнительные расчеты

Listing 6: QUANC8 по точкам x_k

```
1   ..........................
2   for (int i = 0; i < nK; ++i) {
3     quanc8(checkFun, A, B, epsabs, epsrel, &result,
4           &errest, &nofun, &posn, &flagQuanc);
5     arrQuancRes[i] = result;
6   }
7
8   return 0;
9 }
```

Listing 7: Функция с расчетом x_k

```
1 double checkFun(double z)
2 {
3   double x = 1.1875;
4   static int i = -1;
5   if (z == A) {
6     x = 1.1875 + 0.375 * (++i);
7   }
8   else {
9     x = 1.1875 + 0.375 * i;
10  }
11
```

```
12    return { 1 / (exp(z) * (z + x)) };
13  }
```

2.4.1 Результат

x_k	1.1875	1.5625	1.9375	2.3125	2.6875	3.0625	3.4375	3.8125
$f(x)$	0.5298	0.4350	0.3702	0.3228	0.2864	0.2577	0.2343	0.2148

2.5 Сравнение результатов

x_k	$\triangle f(x)_S$	$\triangle f(x)_L$
1.1875	-0.0665	0.0004
1.5625	0.0156	$-2.8 \cdot 10^{-5}$
1.9375	-0.0042	$6.4 \cdot 10^{-6}$
2.3125	0.0014	$-3.0 \cdot 10^{-6}$
2.6875	-0.0014	$2.6 \cdot 10^{-6}$
3.0625	0.0045	$-4.2 \cdot 10^{-6}$
3.4375	-0.0166	$1.4 \cdot 10^{-5}$
3.8125	-0.0119	$-4.8 \cdot 10^{-6}$

$\triangle f(x)_{S|L} = f(x)_{Q8} - f(x)_{S|L}$

3 Вывод

Значение заданного интеграла было успешно вычисленно с помощью программы `QUANC8`, а в результате сплайн-интерполяции и построения полинома Лагранжа 8-й степени были получены аппроксимации исходной функции, что позволяет исследовать ее поведение. Проанализировав таблицу п. 2.5 "Сравнение результатов", можно сделать вывод о точности методов, заключачающийся в том, что значения аппроксимации, полученной построением полинома Лагранжа, более точные, чем значения, полученные сплайн-интерполяцией.