Санкт-Петербургский Политехнический Университет
Институт компьютерных наук и технологий
Высшая школа программной инженерии

Отчёт по лабораторной работе №1

По дисциплине «Вычислительная математика»

Студент: Бабинцева К.А.
Группа: 23534/2
Преподаватель: Леонтьева Т.В.

Санкт-Петербург
2019

Постановка задачи:

ВАРИАНТ N 10

Для $1.5 \le x \le 2.7$ с шагом h=0.2 вычислить значения функции f(x) с использованием программы **QUANC8**, где $f(x) = \int_0^x \sin t^2 dt$. По полученным точкам построить сплайн-функцию и полином Лагранжа 6-й степени. В точках $x_k = 1.6 + 0.2k$ для k=0,1,…5 сравнить значения сплайн-функции и полинома с точным значением f(x) (вычислить интеграл по **QUANC8** с высокой точностью)

Ход работы:

1) Вычислены значения функции $f(x) = \int_0^x \sin(t^2)\, dt$, где x ϵ [1.5, 2.7] с шагом h=0.2 с использованием программы QUANC8.
2) С использованием данных точек (значениями функции) построены полином Лагранжа 6-й степени и сплайн-функция.
3) В точках xk = 1.6 + 0.2k, где k=0, 1,…5 подсчитаны значения сплайн-функция и полином Лагранжа.
4) Сравнены значения функции, подсчитанные с помощью программы Quanc8 и значения, полученные при подсчете полинома Лагранжа, а также значения сплайн-функции, подсчитанные с помощью подпрограмм Spline(для нахождения коэффициентов b, c, d) и Seval(для непосредственного нахождения значений самого сплайна).

Результат работы программы:

Values of integral at first:

x	f(x)
1.5	0.778238
1.7	0.885702
1.9	0.865837
2.1	0.718067
2.3	0.526114
2.5	0.430518
2.7	0.518771

x	f(x)	Lagrange	Spline	|f(x)-L(x)|	|f(x)-S(x)|
1.6	0.845269	0.845496	0.847266	0.000226837	0.00199673
1.8	0.893481	0.893456	0.892526	2.4628e-005	0.000955093
2	0.804776	0.804763	0.805344	1.35429e-005	0.000567546
2.2	0.619228	0.619276	0.616297	4.72787e-005	0.0029318
2.4	0.457864	0.457709	0.468532	0.000155268	0.0106677
2.6	0.452119	0.452961	0.414496	0.000842221	0.0376227

Рис.1: Точные значения функции, полинома Лагранжа и сплайн-функции

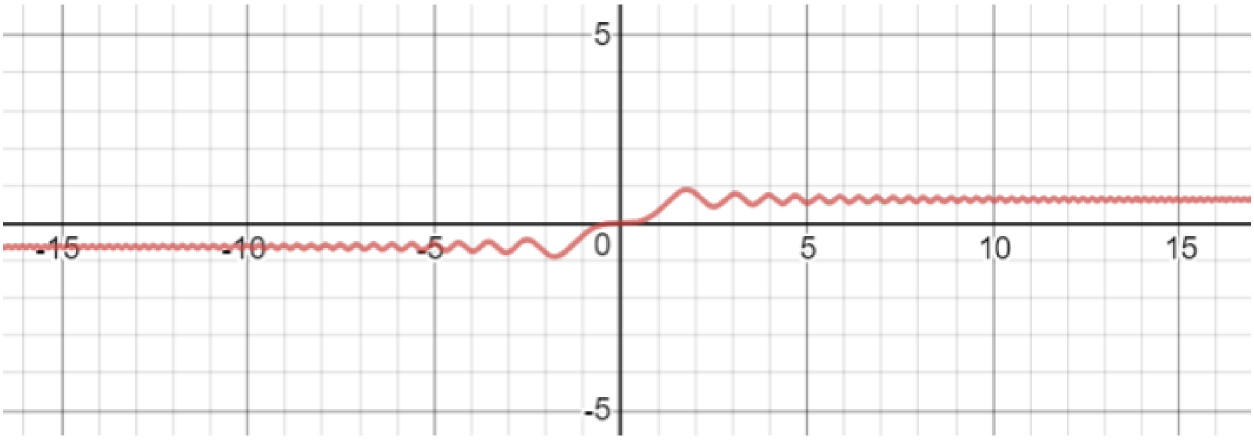
Рис.2: Разница между точными значениями функции и сплайн-функции/полиномом Лагранжа в зависимости от x*

Вывод:

По полученным результатам значений функции $f(x) = \int_0^x \sin(t^2)\,dt$, где x ϵ [1.5, 2.7] с шагом h=0.2 можно сделать вывод, что для ее аппроксимации лучше использовать интерполяционный полином Лагранжа, так как в данном случае самая большая погрешность равна ε = 10^-3, наименьшая – 10^-4. У сплайн-функции наибольшая погрешность составляет 10^-1, а наименьшая – 10^-3.

Приложение

Рис. 3: График функции $f(x) = \int_0^x \sin(t^2)\,dt$



Листинг 1. main.cpp

```
#include <cmath>
#include <iostream>
#include <iomanip>
#include "lagrange.hpp"
#include "quanc8.hpp"
#include "spline.hpp"

double function(double x)
{
  return sin(x*x);
}

int main()
{
  const int N = 7;
  double nodes[N] = {};
```

```
  double node = 1.5;

  for (int i = 0; i < N; i++)
  {
    nodes[i] = node;
    node += 0.2;
  }

  double values[N] = {};

  double a = 0.0;
  double b;
  double abserr = 1e-10;
  double relerr = 0;
  double errest;
  int nofun;
  double flag;
  double result;

  for (int i = 0; i < N; i++) // подсчет интеграла с пределами
интегрирования от 0 до 1.5, 1.7 и т.д.
  {
    b = nodes[i];
    quanc8(function, a, b, abserr, relerr, &result, &errest, &nofun,
&flag);
    values[i] = result;
  }

  std::cout << "Values of integral at first: " << "\n";

  std::cout << std::setw(12) << "x"
           << std::setw(12) << "f(x)\n";

  for (int i = 0; i < N; i++)
  {
    std::cout << std::setw(12) << nodes[i]
             << std::setw(12) << values[i] << "\n";
  }
```

```
  std::cout <<
"______________________________________________________________________
________\n";

  double x_1[N-1] = {};  // массив точек, в которых нужно считать
функцию, полином Лагранжа и сплайн-функцию
  node = 1.6;

  for (int i = 0; i < N-1; i++)
  {
    x_1[i] = node;
    node += 0.2;
  }

  std::cout << std::setw(6) << std::left << "x"
            << std::setw(12) << "f(x)"
            << std::setw(12) << "Lagrange"
            << std::setw(12) << "Spline"
            << std::setw(14) << "|f(x)-L(x)|"
            << std::setw(14) << "|f(x)-S(x)|"
            << std::endl;

  double x_spline[N+1] = {}; // для сплайн-функции
  double f_spline[N+1] = {};
  double b_k[N] = {};
  double c_k[N] = {};
  double d_k[N] = {};

  for (int i = 0; i < N; i++)
  {
    x_spline[i+1] = nodes[i];
    f_spline[i+1] = values[i];
  }

  a = 0.0;
  b = 1.6;

  for (int i = 0; i < N-1; i++)
  {
```

```
    std::cout << std::setw(6) << x_1[i];
    quanc8(function, a, b, abserr, relerr, &result, &errest, &nofun,
&flag);
    std::cout << std::setw(12) << result
              << std::setw(12) << calculateLagrangePolynom(N, x_1[i],
nodes, values);
    spline(N, x_spline, f_spline, b_k, c_k, d_k);
    std::cout << std::setw(12) << seval(N, x_1[i], x_spline, f_spline,
b_k, c_k, d_k)
              << std::setw(14) << fabs(result -
calculateLagrangePolynom(N, x_1[i], nodes, values))
              << std::setw(14) << fabs(result - seval(N, x_1[i], x_spline,
f_spline, b_k, c_k, d_k)) << "\n";

    b += 0.2;
  }

  return 0;
}
```