

**МИНОБРНАУКИ РОССИИ**

**федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Московский государственный технологический университет «СТАНКИН»**

**(ФГБОУ ВО «МГТУ «СТАНКИН»)**

|  |  |
| --- | --- |
| **Институт  информационных технологий** | **Кафедра прикладной математики** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Основная образовательная программа 09.03.02 «Информационные системы и технологии»**  **Отчет по дисциплине «Вычислительная математика»** | | | |
| **по лабораторной работе №2** | | | |
| **Тема: «Интерполирование кубическим сплайном дефекта 1»** | | |
|  | | |
| **Проверил**  **преподаватель** |  | **Красикова Е.М.** | |
|  | подпись |  | |
| **Выполнил**  **студент группы ИДБ-22-06** |  | **Мустафаева П.М.** | |
|  | подпись |  | |

**ЦЕЛЬ РАБОТЫ**

Изучить метод интерполяции кубическим сплайном дефекта 1 и применить его на практике для получения сплайна функции f(x) на отрезке [a b].

**ТАБЛИЧНОЕ ЗАДАНИЕ ФУНКЦИИ**

f(x) = sin(1,5x) + cos (1,5x)

Возьмем отрезок [-2;2] разбив его на 5 локальных отрезков: [-2;-1,2], [-1,2;-0,4], [-0,4;0,4], [0,4;1,2], [1,2;2].

Тогда значения функции:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| x | -2 | -1.2 | -0.4 | 0.4 | 1.2 | 2 |
| f(x) | -1,13111 | -1,20105 | 0,260693 | 1,38998 | 0,746646 | -0,848872 |

Запишем полиномы Si(x) и их производные в виде:

Si(x) = ai + bi(x-xi) + ci/2(x-xi)2 + di/2(x-xi)3 (1)

x∈[xi-1, xi], i = 0, 1, 2, 3, 4, 5

Si’(x) = bi+ ci(x-xi) + di/2(x-xi)2 (2)

Si’’(x) = ci + di(x-xi) (3)

Подставим xi в (1), (2), (3), получим Si(xi) = ai, Si’(xi) = bi, Si’’(xi) = ci.

**Исходная система 15 х 15 и её решение:**

­–1,20105 + b1(–2 + 1,2) + c1/2(–2 + 1,2)2 + d1/6(–2 + 1,2)3 = –1,13111

0,260693 + b2(–1,2 + 0,4) + c2/2(–1,2 + 0,4)2 + d2/6(–1,2 + 0,4)3 = –1,20105

1,38998 + b3(–0,4 – 0,4) + c3/2(–0,4 – 0,4)2 + d3/6(–0,4 – 0,4)3 = 0,260693

0,746646 + b4(0,4 – 1,2) + c4/2(0,4 – 1,2)2 + d4/6(0,4 – 1,2)3 = 1,38998

–0,848872 + b5(1,2 – 2) + c5/2(1,2 – 2)2 + d5/6(1,2 – 2)3 = 0,746646

c2\*0,8 – d2/2\*0,64 = b2 – b1

c3\*0,8 – d3/2\*0,64 = b3 – b2

c4\*0,8 – d4/2\*0,64 = b4 – b3

c5\*0,8 – d5/2\*0,64 = b5 – b4

0,8\*d2 = c2 – c1

0,8\*d3 = c3 – c2

0,8\*d4 = c4 – c3

0,8\*d5 = c5 – c4

c1 – 0,8\*d1 = 0

c5 = 0

**КОД ПРОГРАММЫ**

#include <iostream>

#include <cmath>

const int n = 5;

const float a = -2;

const float b = 2;

const float h = (b - a) / n;

using namespace std;

void fillFunction(float\*& X, float\*& Y) {

for (size\_t i = 0; i < n + 1; i++) {

X[i] = a + i \* h;

Y[i] = sin(15 \* X[i] / 10) + cos(15 \* X[i] / 10);

}

}

void fillMatrix(float\*\*& matrix, unsigned int& N, float\* Y) {

N = n - 1;

const unsigned int \_N = n - 1;

matrix = new float\* [N]; for (size\_t i = 0; i < N; i++)

matrix[i] = new float[N + 1];

for (size\_t i = 0; i < N; i++) for (size\_t j = 0; j < N; j++)

matrix[i][j] = 0;

for (size\_t i = 0; i < N; i++) {

if (i == 0) {

matrix[i][i] = 4;

matrix[i][i + 1] = 1;

}

else if (i == N - 1) {

matrix[i][i - 1] = 1;

matrix[i][i] = 4;

}

else {

matrix[i][i - 1] = 1; matrix[i][i] = 4; matrix[i][i + 1] = 1;

}

}

for (size\_t i = 0; i < N; i++) {

if (i == 0)

matrix[i][N] = 3.0 / (h \* h) \* (Y[i + 2] - 2 \* Y[i + 1]);

else if (i == N - 1)

matrix[i][N] = 3.0 / (h \* h) \* (Y[i] - 2 \* Y[i + 1]); else

matrix[i][N] = 3.0 / (h \* h) \* (Y[i + 2] + Y[i] - 2 \* Y[i + 1]);

}

}

void print(float\*\* matrix, unsigned int N) {

cout << "--------------" << endl; for (size\_t i = 0; i < N; i++)

{

for (size\_t j = 0; j < N + 1; j++) cout << matrix[i][j] << ' ';

cout << endl;

}

cout << "--------------" << endl;

}

float\* Solve(float\*\* matrix, unsigned int N) {

float\* P = new float[N]; float\* Q = new float[N]; float\* X = new float[N + 1];

float a, b, c, d;

for (size\_t i = 0; i < N; i++) {

a = matrix[i][i - 1];

b = matrix[i][i];

c = matrix[i][i + 1];

d = matrix[i][N];

if (i == 0) {

P[i] = -c / b;

Q[i] = d / b;

}

else if (i == N - 1) {

P[i] = 0;

Q[i] = (d - a \* Q[i - 1]) / (b + a \* P[i - 1]);

}

else {

P[i] = -c / (b + a \* P[i - 1]);

Q[i] = (d - a \* Q[i - 1]) / (b + a \* P[i - 1]);

}

}

X[N - 1] = Q[N - 1]; for (int i = N - 2; i > -1; i--)

X[i] = P[i] \* X[i + 1] + Q[i];

for (int i = N; i > 0; i--) X[i] = X[i - 1];

X[0] = 0;

return X;

}

void fillCoefficients(float\* X, float\* Y, float\*& A, float\*& B, float\*& C, float\*& D)

{

for (size\_t i = 0; i < n - 1; i++) {

A[i] = Y[i];

D[i] = (C[i + 1] - C[i]) / (3 \* h);

B[i] = (Y[i + 1] - Y[i]) / h - (C[i + 1] + 2 \* C[i]) \* h / 3;

}

A[n - 1] = Y[n - 1];

B[n - 1] = (Y[n] - Y[n - 1]) / h - 2 \* C[n - 1] \* h / 3; D[n - 1] = -C[n - 1] / (3 \*

h);

}

int main() {

float\*\* Matrix;

float\* X = new float[n]; float\* Y = new float[n]; float\* A = new float[n]; float\*

B = new float[n]; float\* C = new float[n]; float\* D = new float[n]; unsigned int N;

fillFunction(X, Y); fillMatrix(Matrix, N, Y); print(Matrix, N);

C = Solve(Matrix, N); fillCoefficients(X, Y, A, B, C, D); cout << endl;

for (size\_t i = 0; i < n; i++)

cout << "i = " << i << " A[i] = " << A[i] << " B[i] = " << B[i] << " C[i] = "

<< C[i] << " D[i] = " << D[i] << endl; return 0;

}

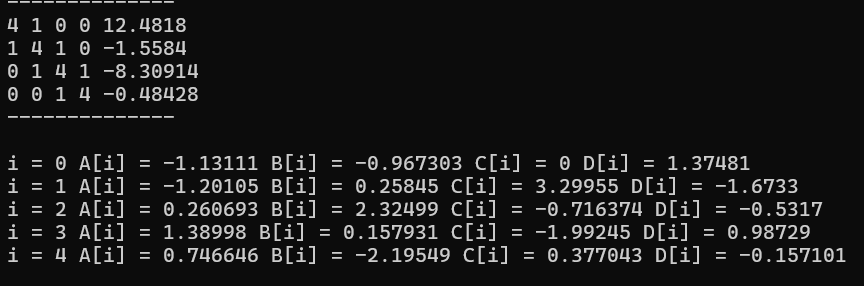
****

Рис. 1 Коэффициенты исходной матрицы и коэффициенты для полиномов сплайнов

Функции полиномов для каждого участка сплайна:

Si(x) = –1.13111 – 0.967303\*(x + 2) + 0\*(x + 2)2 + 1.37481(x + 2)3

x ∈ [–2; –1,2]

Si(x) = –1.20105 + 0.25845\*(x + 1.2) + 3.29955\*(x + 1.2)2 – 1.6733\*(x + 1.2)3

x ∈ [–1,2; –0,4]

Si(x) = 0.260693 + 2.32499\*(x + 0.4) – 0.716374\*(x + 0.4)2 – 0.5317\*(x + 0.4)3

x ∈ [–0,4;0,4]

Si(x) = 1.38998 + 0.157931\*(x – 0.4) – 1.99245\*(x – 0.4)2 + 0.98729\*(x – 0.4)3

x ∈ [0,4;1,2]

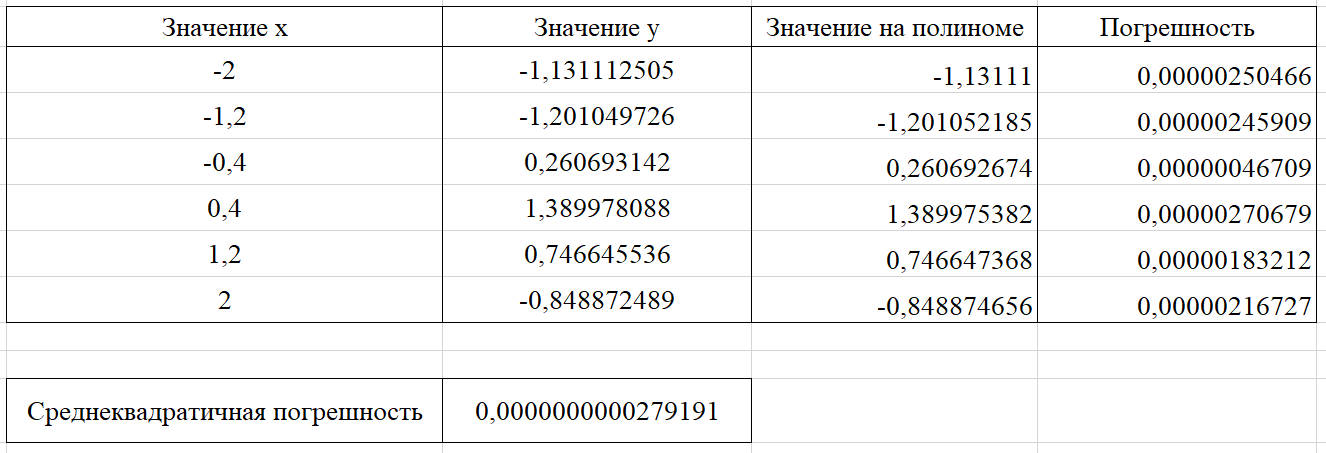
Si(x) = 0.746646 – 2.19549\*(x – 1.2) + 0.377043\*(x – 1.2)2 – 0.157101\*(x – 1.2)3

x ∈ [1,2;2]

**ГРАФИК И ПОГРЕШНОСТЬ СПЛАЙН – ИНТЕРПОЛЯЦИИ**

Погрешность интерполяции построенного сплайна:

δi = |f(xi) – Si(x)|



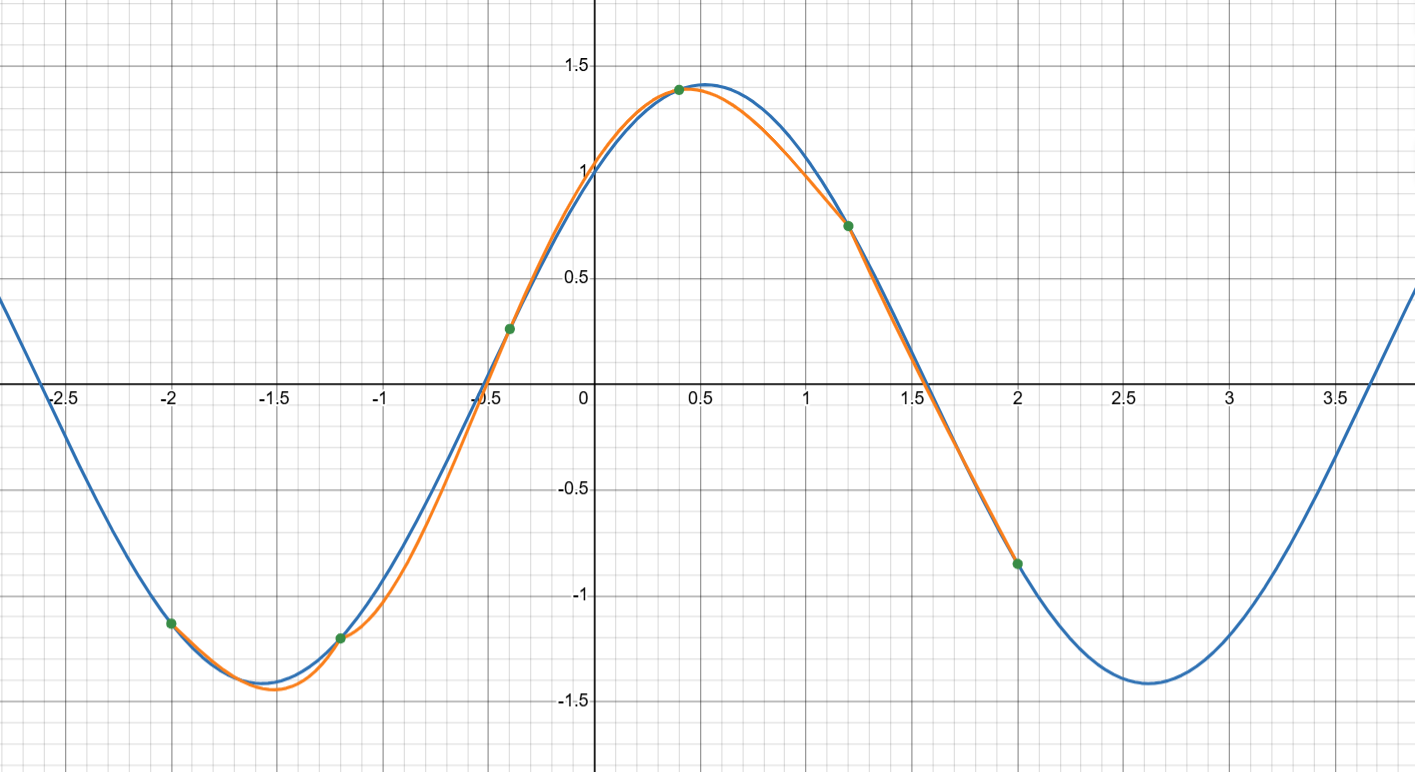


Рис. 2 График функции и сплайна в desmos

**ВЫВОД**

В ходе лабораторной работы был изучен метод интерполирования кубическим сплайном дефекта 1, метод решения СЛАУ матричной прогонкой, написаны программы, задающие таблично исходную функцию, решающие получившуюся СЛАУ методом матричной прогонки, определяющие коэффициенты полиномов сплайна и строящие графики исходной функции и получившегося сплайна.