**ДОМАШНЯЯ РАБОТА №6**

«Приближение кривых. Среднеквадратичное приближение функции»»

**Выполнил:**

студент 3 курса 13 группы кафедры ТП.

Петров Андрей Александрович

**Вариант 8-З).**

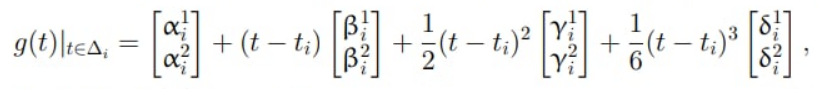
**Задание №1:** Для точек [0; 0], [8; 0], [8; 1], [4; 2] построить интерполирующий алгебраический многочлен и кубический сплайн;

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| x | 0 | 8 | 8 | 4 |
| y | 0 | 0 | 1 | 2 |
| ti | 1 | 2 | 3 | 4 |

**Ход работы:**

**КУБИЧЕСКИЙ СПЛАЙН**

Построим **кубический сплайн** для параметрической функции



Напишем функцию нахождения Si(t):

let splineCubic = function (i, t, alpha, beta, gamma, delta, ti) {  
 return alpha[i] +  
 beta[i] \* (t - ti[i]) +  
 (1 / 2) \* gamma[i] \* ***Math***.pow(t - ti[i], 2) +  
 (1 / 6) \* delta[i] \* ***Math***.pow(t - ti[i], 3);  
}

Изображение выглядит как текст

Автоматически созданное описание

Найдем значения ci, ei и bi:

for (let i = 2; i < n; i++) {  
 c.push(h / (ti[i + 1] - ti[i - 2]));  
 e.push(h / (ti[i + 1] - ti[i - 2]));  
}  
  
for (let i = 1; i <= n - 1; i++) {  
 b.push(6 \* ((alpha[i + 1] - alpha[i]) / h - (alpha[i] - alpha[i - 1]) / h)

/ (ti[i + 1] - ti[i - 1]));  
}

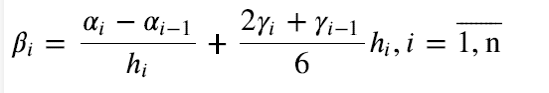
Используем метод прогонки (Tridiagonal matrix algorithm) для нахождения значения γ:

let tmpGamma = tridiagonalMatrixAlgorithm(1, n - 1, c, e, [2, 2], b);  
const gamma = [0, ...tmpGamma, 0];

Найдем значения β, δ и S(x) и построим график:

Изображение выглядит как текст

Автоматически созданное описание



for (let i = 1; i <= n; i++) {  
 delta[i] = (gamma[i] - gamma[i - 1]) / h;  
 beta[i] = (alpha[i] - alpha[i - 1]) / h + h \* (2 \* gamma[i] + gamma[i - 1])  
 / 6;  
}

Произведем вычисления

Изображение выглядит как текст, устройство, датчик

Автоматически созданное описание

Результат нахождения x(ti):

Изображение выглядит как текст

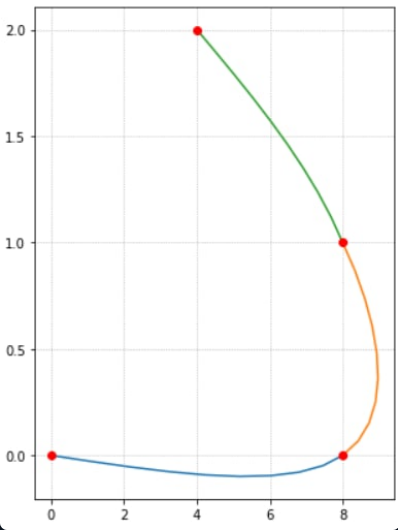
Автоматически созданное описание

Результат нахождения y(ti):

Изображение выглядит как текст

Автоматически созданное описание

График:



**ИНТЕРПОЛИРУЮЩИЙ АЛГЕБРАИЧЕСКИЙ МНОГОЧЛЕН**

Построим интерполирующий алгебраический многочлен

Изображение выглядит как текст, часы

Автоматически созданное описаниеИзображение выглядит как текст, часы, датчик

Автоматически созданное описание

Реализуем функцию g(t):

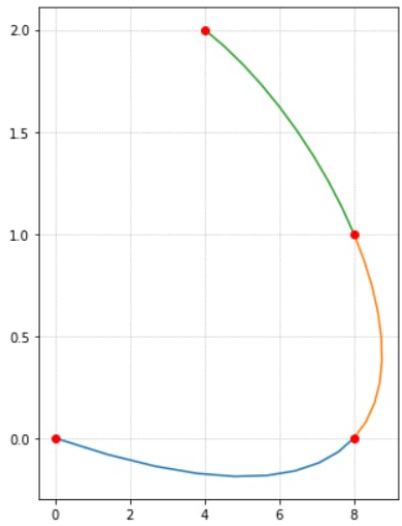
function funcG(points, hop, t){  
 let sum = 0;  
 points.forEach((point, i) => {  
 let lambda = 1;  
 t.forEach((item, j) => {  
 if(i !== j) {  
 lambda \*= (hop - t[j]) / (t[i] - t[j]);  
 }  
 });  
 sum += lambda \* point;  
 })  
 return sum;  
}

Произведем расчеты:



+ + – +

График:



**Задание №2:**

Построить среднеквадратичное приближение параболой к выбранной функции. Выбрать 5 равномерно расположенных на отрезке узлов. Построить среднеквадратичное приближение параболой по выбранным узлам.

**Ход работы:**

Построим среднеквадратичное приближение параболой к выбранной функции.

Построим матрицу Грамма:

=

Среднеквадратичное приближение:

Выберем 5 равномерно расположенных на отрезке узлов и построим среднеквадратичное приближение параболой по выбранным узлам.

Nodes: [1.0, 1.25, 1.5, 1.75, 2.0]

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **φ(x) \ x** | **1** | **1.25** | **1.5** | **1.75** | **2.0** | **sum** |
| **φ0(x)** | 1 | 1 | 1 | 1 | 1 | 5 |
| **φ1(x)** | 1 | 1.25 | 1.5 | 1.75 | 2 | 7.5 |
| **φ2(x)** | 1 | 1.5625 | 2.25 | 3.0625 | 4 | 11.875 |
| **φ3(x)** | 1 | 1.953125 | 3.375 | 5.359375 | 8 | 19.6875 |
| **φ4(x)** | 1 | 2.44140625 | 5.0625 | 9.37890625 | 16 | 33.8828125 |

Изображение выглядит как текст

Автоматически созданное описание

=

Среднеквадратичное приближение на 5 узлах: