МФТИ

Лабораторная работа №2

Вариант 9: “Вычисление с использованием интерполяции”

Выполнил
Студент Б03-907
Алиев Артем Эльдарович

Долгопрудный, 2021

Задача

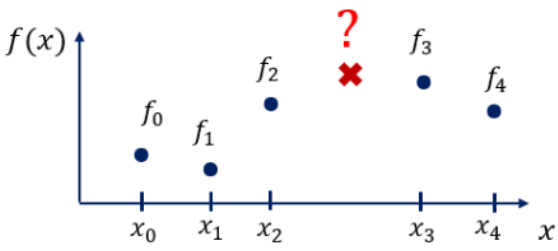
Функция $y = f(x)$ задана таблицей

x	0.2050	0.2052	0.2065	0.2069	0.2075
y	0.20792	0.20813	0.20896	0.20990	0.21053

Вычислить $f(0.2062)$, пользуясь линейной, квадратичной и кубической интерполяциями. В какой форме – Лагранжа или Ньютона – удобнее записывать интерполяционные многочлены?

Составить представление о погрешности, используя остаточный член интерполяции.

Интерполяция



Интерполяция – это способ нахождения промежуточных значений величины по имеющемуся дискретному набору известных значений.

x_i - узлы интерполяции

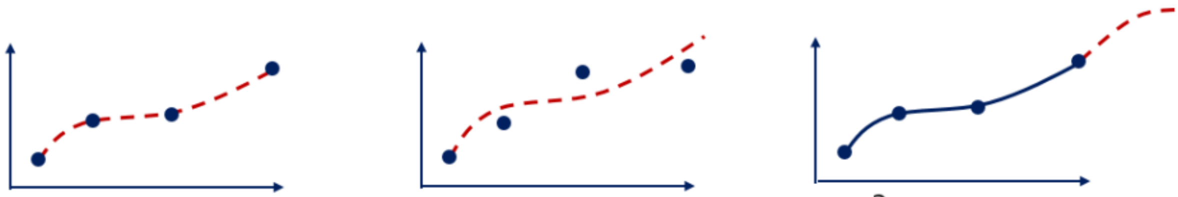
Алгебраическая интерполяция

$$\{f_i\}_{i=0}^{N} \xrightarrow{I} P_N(x)$$

Сеточная функция → Интерполяционный полином

Основное условие интерполяции: равенство функции и интерполяционного полинома в узлах интерполяции

$$\boxed{P_N(x_i) \equiv f_i}$$



Интерполяция алгебраическими полиномами

Сеточная функция

x	x_1	x_2	x_3	x_4	x_5	…	x_N
$f(x)$	f_1	f_2	f_3	f_4	f_5	…	f_N

Строим интерполянт в виде полинома: $$P_N(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_N x^N$$

неизвестны

Требуем выполнения основного условия интерполяции $P_N(x_i) = f_i$ и находим a_i из решения СЛАУ:

$$\begin{cases} a_0 + a_1 x_0 + \cdots + a_N x_0^N = f_0 \\ a_0 + a_1 x_1 + \cdots + a_N x_1^N = f_1 \\ \cdots \\ a_0 + a_1 x_N + \cdots + a_N x_N^N = f_N \end{cases}$$

Определитель матрицы – детерминант Вандермонда. В случае различия всех узлов сетки он отличен от нуля, и, значит, существует единственное решение системы – набор коэффициентов.

$$\begin{vmatrix} 1 & x_0 & \dots & x_0^N \\ 1 & x_1 & \dots & x_1^N \\ \dots & \dots & \dots & \dots \\ 1 & x_N & \dots & x_N^N \end{vmatrix} = \prod_{0 \le j \le i \le N} (x_i - x_j) = 0$$

$$\Updownarrow$$

Существует пара (x_i, x_j): $x_i = x_j, \quad i \neq j$

Утверждение Если заданы $N+1$ узлов x_0, …, x_N среди которых нет совпадающих, и значения функции в этих узлах $f(x_0)$, …, $f(x_N)$, то существует один и только один многочлен степени не выше N, принимающий в узлах x_i заданные значения $f(x_i)$.

Метод Лагранжа

Строим интерполяционный полином в виде: $L_N(x) = \sum_{k=0}^{N} \varphi_k(x) f_k$

Из основного условия интерполяции получаем $L_N(x_i) = f_i \quad \forall i \quad \Leftrightarrow \quad \sum_{k=0}^{N} \varphi_k(x_i) f_k = f_i$

Соответственно $\varphi_k(x_i) = \begin{cases} 0, i \neq k \\ 1, i = k \end{cases} \qquad i = 0, \dots, N$

Каждая из функций $\varphi_k(x)$ имеет не менее N нулей на $[a, b]$.	$\Rightarrow$	Ищем $\varphi_k(x)$ в виде полинома степени N

$$\varphi_k(x) = \alpha_k (x - x_0)(x - x_1) \dots (x - x_{k-1})(x - x_{k+1}) \dots (x - x_N)$$

Из условия $\varphi_k(x_k) = 1$ находим α_k

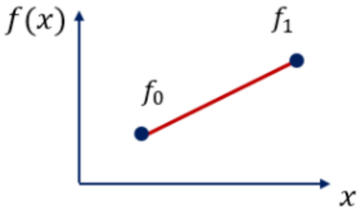
$$\alpha_k = \frac{1}{(x_k - x_0)(x_k - x_1) \dots (x_k - x_{k-1})(x_k - x_{k+1}) \dots (x_k - x_N)}$$

$$\boxed{\varphi_k(x) = \prod_{\substack{i=0 \\ i \neq k}}^{N} \frac{x - x_i}{x_k - x_j}}$$

Примеры построенных методом Лагранжа полиномов

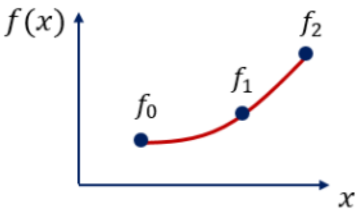
$$L_N(x) = \sum_{k=0}^{N} \varphi_k(x) f_k \qquad \varphi_k(x) = \prod_{\substack{i=0 \\ i \neq k}}^{N} \frac{x - x_i}{x_k - x_j}$$

Линейная интерполяция:



$$L_1(x) = \sum_{k=0}^{1} \varphi_k(x) f_k = f_0 \frac{x - x_1}{x_0 - x_1} + f_1 \frac{x - x_0}{x_1 - x_0}$$

Квадратичная интерполяция:



$$L_2(x) = \sum_{k=0}^{2} \varphi_k(x) f_k = f_0 \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} +$$

$$+ f_1 \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} + f_2 \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)}$$

Метод Ньютона

Интерполяционный полином в форме Ньютона – разностный аналог формулы Тейлора

$$f(x) = f(x_0) + (x - x_0)f'(x_0) + \frac{(x - x_0)^2 f''(x_0)}{2!} + \cdots$$

Разделенная разность первого порядка

$$f_{ij} = f\left(x_i, x_j\right) = \frac{f_i - f_j}{x_i - x_j}, \qquad i, j = 0, \dots, N \quad i \neq j$$

Разделенная разность второго порядка

$$f_{j\ j+1\ j+2} = f\left(x_j, x_{j+1}, x_{j+2}\right) = \frac{f_{j\,j+1} - f_{j+1\ j+2}}{x_j - x_{j+2}} = \frac{\dfrac{f_j - f_{j+1}}{x_j - x_{j+1}} + \dfrac{f_{j+1} - f_{j+2}}{x_{j+1} - x_{j+2}}}{x_j - x_{j+2}}$$

Разделенная разность k-го порядка

$$f_{j\ j+1\ j+2\ \dots j+k} = f\left(x_j, x_{j+1}, \dots, x_{j+k}\right) = \frac{f_{j+1 \dots j+k} - f_{j \dots j+k-1}}{x_{j+k} - x_j}$$

Интерполяционный полином в форме Ньютона

$$P_N = f_0 + (x - x_0)f_{01} + (x - x_0)(x - x_1)f_{012} + \cdots (x - x_0) \dots (x - x_{N-1})f_{012\dots N}$$

Метод Ньютона

Разделенная разность k-го порядка

$$f_{j\ j+1\ j+2\ \dots j+k} = f\left(x_j, x_{j+1}, \dots, x_{j+k}\right) = \frac{f_{j+1 \dots j+k} - f_{j \dots j+k-1}}{x_{j+k} - x_j}$$

x_0	f_0				
		f_{01}			
x_1	f_1		f_{012}		
		f_{12}	.		
x_2	f_2	.	.	...	$f_{0\dots N}$
.	.	.	.		
.	.	.	$f_{N\text{-}2\ N\text{-}1\ N}$		
.	.	$f_{N\text{-}1\ N}$			
x_N	f_N				

Интерполяционный полином в форме Ньютона

$$P_N = f_0 + (x - x_0)f_{01} + (x - x_0)(x - x_1)f_{012} + \cdots (x - x_0) \dots (x - x_{N-1})f_{012\dots N}$$

Погрешность интерполяции

Опр: Разница между функцией и интерполяционным полиномом N-ой степени в точке x называется остаточным членом интерполяции: $R_N(x) = f(x) - L_N(x)$

Утверждение Пусть на отрезке $[a,b]$ функция $u(x)$ $(N+1)$ раз непрерывно дифференцируема. Тогда:

$$\boxed{R_N(x) = \frac{u^{(N+1)}(\xi)}{(N+1)!}\prod_{j=0}^{N}(x - x_j), \qquad \xi \in [a,b]}$$

Доказательство. Если $x = x_j$, то утверждение верно. Иначе введем в рассмотрение функцию:

$$g(t) = f(t) - L_N(t) - [f(x) - L_N(x)]\prod_{j=0}^{N}\frac{(t - x_j)}{(x - x_j)}$$

Функция $g(t)$, имеет $N + 2$ нуля на $[a, b]$:

$$g(x_i) = f(x_i) - L_N(x_i) - [f(x) - L_N(x)]\prod_{j=0}^{N}\frac{(x_i - x_j)}{(x - x_j)} = 0, \qquad i = 0, \dots N$$

$$g(x) = f(x) - L_N(x) - [f(x) - L_N(x)]\prod_{j=0}^{N}\frac{(x - x_j)}{(x - x_j)} = 0.$$

По обобщенной теореме Ролля: $\exists \xi \in [a,b]\colon\ g^{(N+1)}(\xi) = 0$

$$g^{(N+1)}(\xi) = f^{(N+1)}(\xi) - 0 - [f(x) - L_N(x)]\frac{(N+1)!}{\prod_{j=0}^{N}(x - x_j)} = 0$$

$$f(x) - L_N(x) = \frac{f^{(N+1)}(\xi)}{(N+1)!}\prod_{j=0}^{N}(x - x_j) \Rightarrow \max|f(x) - L_N(x)| = \frac{\max\limits_{\xi\in[a,b]}\left|f^{(N+1)}(\xi)\right|}{(N+1)!}\left|\prod_{j=0}^{N}(x - x_j)\right|$$

Погрешность интерполяции на равномерной сетке

Утверждение Для случая равномерной сетки на отрезке $[a,b]$

$$\{x_i\}_{i=0}^{N}, \qquad x_i = a + ih, \qquad h = (b - a)/N$$

для любого x на отрезке $[a, b]$

$$\boxed{|f(x) - L_N(x)| \le \frac{h^{N+1}}{N+1}\max_{\xi\in[a,b]}\left|f^{(N+1)}(\xi)\right|.}$$

Доказательство. Положим $x = x_k + \alpha h, \qquad \alpha \in (0,1), \qquad k = 0, \dots N - 1$

Тогда $x - x_j = kh + \alpha h - jh = h(k + \alpha - j)$

$$\prod_{j=0}^{N}(x - x_j) = h^{N+1}\prod_{j=0}^{N}(k + \alpha - j) \le h^{N+1}N!$$

$$|f(x) - L_N(x)| = \frac{\max\limits_{\xi\in[a,b]}\left|f^{(N+1)}(\xi)\right|}{(N+1)!}\left|\prod_{j=0}^{N}(x - x_j)\right|$$

$$\Rightarrow |f(x) - L_N(x)| \le \frac{h^{N+1}}{N+1}\max_{\xi\in[a,b]}\left|f^{(N+1)}(\xi)\right|.$$

j	0	…	$k-2$	$k-1$	k	$k+1$	$k+2$	…	N
$\lvert k+\alpha-j\rvert$	$k+\alpha$	…	$2+\alpha$	$2+\alpha$	α	$1-\alpha$	$2-\alpha$	…	$N-k-\alpha$
Маж. мн. в $N!$	$k+1$	…	3	2	1	1	$2+k$	…	N

Линейная интерполяция в форме Ньютона

Код:

```python
import numpy as np


def linear_interpolation(massx, massy, x):
    for i in range(1, len(massx)):
        if massx[i - 1] <= x <= massx[i]:
            a = (massy[i] - massy[i - 1]) / (massx[i] - massx[i - 1])
            b = massy[i - 1] - a * massx[i - 1]
            return a * x + b


X = [0.205, 0.2052, 0.2065, 0.2069, 0.2075]
F = [0.20792, 0.20813, 0.20896, 0.20990, 0.21053]

print(linear_interpolation(X, F, 0.2062))
```

Результат: 0.20876846153846154

Квадратичная интерполяция в форме Ньютона:

Код:

```python
from math import *


def find_determinant(a, rev):...


def solve(a, b, n):...


def quadratic_interpolation(massx, massy, x):
    min = abs(massx[0] - x) + abs(massx[1] - x) + abs(massx[2] - x)
    min_i = 1
    for i in range(2, len(massx) - 1):
        if abs(massx[i - 1] - x) + abs(massx[i] - x) + abs(massx[i + 1] - x) < min:
            min = abs(massx[i - 1] - x) + \
                      abs(massx[i] - x) + abs(massx[i + 1] - x)
            min_i = i
    a, b, c = solve([[massx[min_i - 1] ** 2, massx[min_i - 1], 1], [massx[min_i] ** 2, massx[min_i], 1],
                      [massx[min_i + 1] ** 2, massx[min_i + 1], 1]],
                     [massy[min_i - 1], massy[min_i], massy[min_i + 1]], 3)
    return a * x ** 2 + b * x + c


X = [0.205, 0.2052, 0.2065, 0.2069, 0.2075]
F = [0.20792, 0.20813, 0.20896, 0.20990, 0.21053]

print(quadratic_interpolation(X, F, 0.2062))
```

Результат: 0.2084664253393811

Кубическая интерполяция в форме Ньютона

$P_3(x) = P_2(x) + b_3(x - 0.2050)(x - 0.2052)(x - 0.2065)$

$P_3(x) = P_2(x) + 673268.42(x - 0.2050)(x - 0.2052)(x - 0.2065)$

$P_3(0.2062) = 0.20861$

Представление о погрешности, используя остаточный член интерполяции:

$P_4(x) = P_3(x) + b_4(x - 0.2050)(x - 0.2052)(x - 0.2065)(x-0.2069)$

$P_4''''(x) = 24b_4 = -1.6*10^{10}$

$|R_3(x)| \le | -1.6*10^{10} (x - 0.2050)(x - 0.2052)(x - 0.2065)(x-0.2069)/(4!) |$

$R_3(0.2062)| \le 16.8*10^{-5}$

Кубическая интерполяция в форме Лагранжа

$P_3(x) = 0.20792*(x - 0.2052)(x - 0.2065)(x - 0.2069) / ((0.2050 - 0.2052)(0.2050 - 0.2065)(0.2050 - 0.2069)) +$

$+ 0.20813*(x - 0.2050)(x - 0.2065)(x - 0.2069) / ((0.2052 - 0.2050)(0.2052 - 0.2065)(0.2052 - 0.2069)) +$

$+ 0.20896*(x - 0.2050)(x - 0.2052)(x - 0.2069) / ((0.2065 - 0.2050)(0.2065 - 0.2052)(0.2065 - 0.2069)) +$

$+ 0.20990*(x - 0.2050)(x - 0.2052)(x - 0.2065) / ((0.2069 - 0.2050)(0.2069 - 0.2052)(0.2069 - 0.2065)) =$

$= -364771929.8*(x - 0.2052)(x - 0.2065)(x - 0.2069) + 470882352.9*(x - 0.2050)(x - 0.2065)(x - 0.2069) +$

$- 267897435.9*(x - 0.2050)(x - 0.2052)(x - 0.2069) + 162461300.3*(x - 0.2050)(x - 0.2052)(x - 0.2065)$

$P_3(0.2062) = 0.20860$

Представление о погрешности, используя остаточный член интерполяции:

$P_4''''(x) = -1.61*10^{10}$

$|R_3(x)| \le | -1.61*10^{10}(x - 0.2050)(x - 0.2052)(x - 0.2065)(x-0.2069)/(4!) |$

$|R_3(0.2062)| \le 16.9*10^{-5}$

Вывод

Форму Лагранжа удобно использовать, когда интерполируется несколько функций, а узлы фиксированы, а Ньютона наоборот, когда число узлов постепенно увеличивается, а функция та же. Поэтому *удобнее Ньютон*, тем более для программной реализации.

[Ссылка](#) на GitHub с полными кодами