Выполнил: Сокольвяк Сергей, 344 группа

**Лабораторная работа 2:**

**Полиномы наилучшего приближения в пространстве C**

**Вариант 4**

Задача:

1. Построить полином наилучшего приближения в для заданной функции . Найти наилучшее приближение.

1.1. Построить на одном рисунке графики и . Пунктиром представить хорду и касательную. Показать местоположение точек альтернанса.

2. Найти полином наилучшего приближения для .

2.1. Построить на одном рисунке графики , и полинома Чебышева , сосчитав предварительно значения корней и точек экстремума. Указать на графике местоположение точек альтернанса.

Теория:

1)

*Определение.* Точки альтернанса ­– это точки t, в которых модуль максимален.

*Теорема Чебышёва об альтернансе.* Пусть многочлен степени и нашлись точки такие, что и знаки чередуются. Тогда полином наилучшего приближения в .

Заметим, что линейная функция, равная среднему арифметическому двух линейных функций, задающих хорду, соединяющую точки и касательную к , параллельную хорде, удовлетворяет условию теоремы, а следовательно является полиномом наилучшего приближения функции .

2)

*Замечание.* Для того чтобы найти полином наилучшего приближения для функции , можно воспользоваться нормированным полиномом Чебышёва. Тогда полином Чебышёва, .

*Замечание.* Полином Чебышёва можно вычислить по рекуррентным формулам:

*Замечание.* Корни и точки экстремума полинома Чебышёва можно найти, воспользовавшись формулами:

Расчеты:

Вычислим многочлен Чебышёва 6-ой степени:

Программа:

% функция поиска наилучшего приближения

% произвольной функции f

% принимает функцию f, для которой требуется найти

% полином наилучшего приближения p = ax + b,

% интервал, на котором заданна фукнция,

% возвращает полином p = ax + b,

% реализующий наилучшее приближение,

% массив значений исходной функции,

% массив значений хорды,

% массив значений касательной

% наилучшее приближение

function [p\_arr, f\_arr, h\_arr, k\_arr, bst] = BestEstimatePolinom(f, x)

% размер исходного интервала

n = size(x, 2);

% коэффициенты уравнения хорды

h1 = (f(x(n)) - f(x(1))) / (x(n) - x(1));

h2 = (x(1) \* f(x(n)) - x(n) \* f(x(1))) / (x(1) - x(n));

% значения исходной функции на заданном интервале

f\_arr = zeros(1, n);

% значения хорды на заданном интервале

h\_arr = zeros(1, n);

% значения касательной на заданном интервале

k\_arr = zeros(1, n);

% заполнение массива значений исходной функции

% и значений хорды на исходном интервале

for i = 1:n

f\_arr(i) = f(x(i));

h\_arr(i) = h1 \* x(i) + h2;

end

% поиск точек, через которые можно провести касательную

res = GetAlternancePoints(f\_arr, h\_arr, n);

% заполнение массива значений касательной

for i = 1:n

if (f\_arr(res(3)) > h\_arr(res(3)))

k\_arr(i) = h\_arr(i) + res(1);

else

k\_arr(i) = h\_arr(i) - res(1);

end

end

% вычисление двух точек для нахождения полинома

if (f\_arr(res(3)) > h\_arr(res(3)))

y1\_p = h\_arr(1) + res(1) / 2;

y2\_p = h\_arr(res(3)) + res(1) / 2;

else

y1\_p = h\_arr(1) - res(1) / 2;

y2\_p = h\_arr(res(3)) - res(1) / 2;

end

% массив коэффициентов уравнения полинома,

% реализующего наилучшее приближение

p\_arr = zeros(1, 2);

% подсчет коэффициентов уравнения полинома

p\_arr(1) = (y2\_p - y1\_p) / (x(res(3)) - x(1));

p\_arr(2) = (x(1) \* y2\_p - x(res(3)) \* y1\_p) / (x(1) - x(res(3)));

% наилучшее приближение

bst = res(1);

% функция поиска точек альтернанса

% принимает массивы значений двух функций,

% для которых ведется поиск точек альтернанса

% возвращает массив с информацией

% о точках альтернанса

function res = GetAlternancePoints(y1, y2, n)

% массив значений модуля разности точек фукнций y1 и y2

alter = zeros(1, n);

% массив индексов точек альтернанса

res = zeros(1, n + 2);

% заполнение массива значениями разностей точек функций y1 и y2

for i = 1:n

alter(i) = abs(y1(i) - y2(i));

end

% максимальное значение разности

res(1) = max(alter);

% счетчик для записи индексов точек альтернанса в res

k = 3;

% погрешность сравнения

eps = 0.00000000001;

% запись индексов всех точек альтернанса в массив res

for i = 1:n

if (abs(alter(i) - res(1)) <= eps)

res(k) = i;

k = k + 1;

end

end

% запись кол-ва точек альтернанса

res(2) = k - 3;

function PlotsGraphAndChord()

% заданный интервал

x = 0:0.01:1;

% исходная функция

f = @(t)(1 / sqrt(1 + t));

% функция нахождения информации о полиноме,

% реализующим наилучшее приближение

% о хорде и о касательной

[p, f\_arr, h\_arr, k\_arr] = BestEstimatePolinom(f, x);

% массив значений полинома на отрезке [0,1]

p\_arr = zeros(1, 101);

% заполнение массива значений полинома

for i = 1:101

p\_arr(i) = p(1) \* x(i) + p(2);

end

% заполнение массива с информацией о точках альтернанса

res = GetAlternancePoints(f\_arr, p\_arr, 101);

% массивы первой и второй координат точек альтернанса

x1 = zeros(1, res(2));

f1 = zeros(1, res(2));

% заполнение массивов координат точками альтернанса

for i = 1:res(2)

x1(i) = x(res(2 + i));

f1(i) = f\_arr(res(2 + i));

end

% построение графиков

plot(x,f\_arr,'-b',x,p\_arr,'-r',x,h\_arr,'--g')

hold on

plot(x,k\_arr,'--k',x1, f1, '\*')

hold off

legend('Исходная функция','Полином','Хорда',...

'Касательная','Точки альтернанса')

xlabel('ось абcцисс')

ylabel('ось ординат')

grid on

% функция поиска полинома наилучшего приближения

% функции f = x^n с помощью полинома Чебышёва

% возвращает полином Pn-1,

% реализующий наилучшее приближение,

% массив значений исходной функции,

% массив значений полинома Чебышёва

function [p\_arr, f\_arr, T\_arr] = BestEstimatePolinomByCheb()

% исходный интервал

x = -1:0.01:1;

% массив значений исходной функции

% на интервале [-1, 1]

f\_arr = zeros(1, 201);

% массив значений полинома Чебышёва

% на исходном интервале

T\_arr = zeros(1, 201);

% массив значений полинома приближения

p\_arr = zeros(1, 201);

% полином Чебышёва необходимой степени

T = @(x)(32\*x^6 - 48\*x^4 + 18\*x^2 - 1);

% вычисление значений массива f\_arr,

% массива значений полинома Чебышёва,

% массива значений полинома приближения

for i = 1:201

f\_arr(i) = x(i) ^ 6;

T\_arr(i) = T(x(i));

p\_arr(i) = f\_arr(i) - (T\_arr(i) / (2^(5)));

end

% функция поиска корней и

% точек экстремума полинома Чебышёва

% возвращает массивы координат корней и

% точек экстремума полинома Чебышёва

function [xr, yr, xe, ye] = GetRootsAndExtremum()

% полином Чебышёва

T = @(x)(32\*x^6 - 48\*x^4 + 18\*x^2 - 1);

% массив координат корней

xr = zeros(1, 6);

yr = zeros(1, 6);

% массив координат экстремумов

xe = zeros(1, 5);

ye = zeros(1, 5);

% заполнение массивов координат корней

for i = 1:6

xr(i) = cos(pi\*(2\*(i-1)+1)/(2\*6));

yr(i) = T(xr(i));

end

% заполнение массивов координат экстремумов

for i = 1:5

xe(i) = cos(pi\*i/6);

ye(i) = T(xe(i));

end

% функция поиска полинома наилучшего приближения

% функции f = x^n с помощью полинома Чебышёва

% возвращает полином Pn-1,

% реализующий наилучшее приближение,

% массив значений исходной функции,

% массив значений полинома Чебышёва

function PlotsGraphAndCheb()

% заданный интервал

x = -1:0.01:1;

% функция нахождения информации о полиноме,

% реализующим наилучшее приближение,

% исходном полиноме и

% о полиноме Чебышёва

[p\_arr, f\_arr, T\_arr] = BestEstimatePolinomByCheb();

% заполнение массива с информацией о точках альтернанса

res = GetAlternancePoints(f\_arr, p\_arr, 201);

% массивы первой и второй координат точек альтернанса

x1 = zeros(1, res(2));

f1 = zeros(1, res(2));

% заполнение массивов координат точками альтернанса

for i = 1:res(2)

x1(i) = x(res(2 + i));

f1(i) = f\_arr(res(2 + i));

end

% массивы координат корней и

% экстремумов полинома Чебышёва

[xr, yr, xe, ye] = GetRootsAndExtremum();

% построение графиков

plot(x, f\_arr, '-b', x, p\_arr, '-r', x, T\_arr, '--g')

hold on

plot (x1, f1, '\*', xr, yr, 'xk', xe, ye, 'ok')

hold off;

legend('Исходная функция','Полином приближения','Многочлен Чебышёва', ...

'Точки альтернанса','Корни полинома Чебышёва','Точки экстремума')

ylabel('ось ординат')

grid on

Графики:



