|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Laboratorium Metod Numerycznych | | | | | | |
| Rok akademicki | Termin | Rodzaj studiów | Kierunek | Prowadzący | Grupa | Sekcja |
| 2017/2018 | Czwartek | Dzienne | INF | mgr inż. Wojciech Łabaj | 6 | 11 |
| 16:45 - 19:45 |

Sprawozdanie z ćwiczenia numer 1

Data wykonania ćwiczenia: 2018-04-26

Temat ćwiczenia:

Interpolacja

|  |
| --- |
| Skład podsekcji:  Michał Miciak  Bartłomiej Krasoń |

## Treść zadania:

Napisać program wyznaczający wartość wielomianu interpolacyjnego Lagrange’a w punktach leżących w przedziale <a,b> dla funkcji interpolowanej f(x).

Funkcja interpolowana: f(x)= |cos(x)\*x|

Przyjąć(n+1) węzłów – dla n=7, 8, 15, 16

1. Równoodległych)
2. Dobranych optymalnie

Przedziały

Punkty w których należy wyznaczyć wartości wielomianu Lagrange’a są również równoodległe – dla np = 150

)

Program oprócz funkcji głównej (main) ma zawierać dodatkowo następujące funkcje pomocnicze:

1. Funkcja wprowadzająca dane(n, np, a, b),
2. Funkcja tablicująca węzły i wartości funkcji interpolowanej f(x) w tych węzłach,
3. Funkcja wyznaczająca wartości wielomianu Lagrange’a w dowolnym punkcie
4. Funkcja wyznaczająca tablice: punktów xpj, wartości funkcji interpolowanej f(xpj) i wielomianu Lagrange’a w tych punktach Ln(xpj) i zapisująca te tablice do pliku.

W oparciu o te tablice zrealizować wykres(np. za pomocą Excela) zawierający:

- wykres funkcji interpolowanej,

- wykres funkcji interpolującej,

- błąd

- węzły

# Rozwiązanie

Wzór Lagrange’a:

Wzór na błąd:

# Wykresy

|  |
| --- |
| Dla punktów równoodległych: |
| n=7 a=-3 b=3 row.jpg |
| n=7 a=-6 b=6 row.jpg |
| n=8 a=-3 b=3 row.jpg |
| n=8 a=-6 b=6 row.jpg |
| n=15 a=-3 b=3 row.jpg |
| n=15 a=-6 b=6 row.jpg |
| n=16 a=-3 b=3 row.jpg |
| n=16 a=-6 b=6 row.jpg |

|  |
| --- |
| Dla punktów dobranych optymalnie: |
| n=7 a=-3 b=3 opt.jpg |
| n=7 a=-6 b=6 opt.jpg |
| n=8 a=-3 b=3 opt.jpg |
| n=8 a=-6 b=6 opt.jpg |
| n=15 a=-3 b=3 opt.jpg |
| n=15 a=-6 b=6 opt.jpg |
| n=16 a=-3 b=3 opt.jpg |
| n=16 a=-6 b=6 opt.jpg |

# Wnioski

1. Błąd interpolacji jest znacznie mniejszy w przypadku, gdy węzły interpolacji są dobrane optymalnie.
2. Dla większej ilości węzłów, błąd interpolacji jest mniejszy.
3. W przedziale <-3,3> wartości wielomianu Lagrange’a różnią się mniej od wartości funkcji interpolowanej, niż w przedziale <-6,6>.

# Kod

#include <iostream>

#include <math.h>

#include <vector>

#include <fstream>

#include <iomanip>

#include <string>

using namespace std;

#define M\_PI 3.141592653589

// Funkcja pobierajaca dane od uzytkownika

void wejscie(int &n, int &np, int &a, int &b, bool &optymalnie)

{

cout << "Podaj n: ";

cin >> n;

cout << "Podaj np: ";

cin >> np;

cout << "Podaj a: ";

cin >> a;

cout << "Podaj b: ";

cin >> b;

cout << "Jak wyznaczyc wezly:\n 1 - optymalnie\n 0 - rownoodlegle\n";

cin >> optymalnie;

}

// Funkcja zwraca wartosc funkcji interpolowanej dla danego argumentu x

double funkcjaInterpolowana(double x)

{

double result = cos(x)\*x;

if (result < 0)

{

result = result \* (-1.0);

}

return result;

}

// Funkcja zwraca wektor zawierajacy wezly

vector<double> wyznaczWezly(int n, int a, int b, bool optymalne)

{

vector<double> vectorDoZwrotu;

if (optymalne)

{

const double pi = M\_PI;

double p1 = (double)(a + b) / 2;

double p2 = (double)(b - a) / 2;

for (int i = 0; i < (n + 1); i++)

{

double pom1 = cos(2 \* i + 1);

double pom2 = (2 \* n + 2);

double x = (p1 + p2 \* cos(((2 \* i + 1) / (double)(2 \* n + 2))\*pi));

vectorDoZwrotu.push\_back(x);

}

}

else

{

double p = (double)(b - a) / n;

for (int i = 0; i < (n + 1); i++)

{

double x = a + (i\*p);

vectorDoZwrotu.push\_back(x);

}

}

return vectorDoZwrotu;

}

// Funckja zwraca wektor wartosci funkcji interpolowanej dla wezlow

vector<double> wyznaczWartosciFunkcjiInterpolowanejDlaWezlow(vector<double> &wezly)

{

vector<double> vectorDoZwrotu;

for (int i = 0; i < wezly.size(); i++)

{

vectorDoZwrotu.push\_back(funkcjaInterpolowana(wezly[i]));

}

return vectorDoZwrotu;

}

// Funkcja zwraca wektor wartosci bledow interpolacji

vector<double> wyznaczBladInterpolacji(vector<double> funkcjaInterpolowana, vector<double> wartosciWielomianuLagrange)

{

vector<double> vectorDoZwrotu;

for (int i = 0; i < funkcjaInterpolowana.size(); i++)

{

vectorDoZwrotu.push\_back(abs(funkcjaInterpolowana[i] - wartosciWielomianuLagrange[i]));

}

return vectorDoZwrotu;

}

// Funkcja zwraca wektor wartosci Wielomianu Lagrange'a dla danych x

vector<double> wyznaczWartosciWielomanuLagrange(vector<double> punkty, int n, vector<double> wartosciFunkcjiInterpolowanej, vector<double> wezly)

{

vector<double> vectorDoZwrotu;

for (int k = 0; k < punkty.size(); k++)

{

double wynik = 0;

for (int i = 0; i < (n + 1); i++)

{

double wynikIloczynu = 1;

for (int j = 0; j < (n + 1); j++)

{

if (i != j)

{

wynikIloczynu \*= (punkty[k] - wezly[j]) / (wezly[i] - wezly[j]);

}

}

wynik += wartosciFunkcjiInterpolowanej[i] \* wynikIloczynu;

}

vectorDoZwrotu.push\_back(wynik);

}

return vectorDoZwrotu;

}

// Funkcja wykonujaca obliczenia oraz generujaca raport

void wyjscie(int n, int np, int a, int b, bool optymalnie)

{

string nazwa = "raport\_n=";

nazwa += to\_string(n);

nazwa += "\_ab=";

nazwa += to\_string(a);

nazwa += to\_string(b);

if (optymalnie)

{

nazwa += "\_opt";

}

else

{

nazwa += "\_row";

}

fstream plik;

plik.open(nazwa + ".csv", ios\_base::out);

vector<double> punktyX = wyznaczWezly(np, a, b, false);

vector<double> wartosciFunkcjiInterpolowanej = wyznaczWartosciFunkcjiInterpolowanejDlaWezlow(punktyX);

vector<double> wezly = wyznaczWezly(n, a, b, optymalnie);

vector<double> wartosciFunkcjiInterpolowanejDlaWezlow = wyznaczWartosciFunkcjiInterpolowanejDlaWezlow(wezly);

vector<double> wartosciWielomianuLagrange = wyznaczWartosciWielomanuLagrange(punktyX, n, wartosciFunkcjiInterpolowanejDlaWezlow, wezly);

vector<double> blad = wyznaczBladInterpolacji(wartosciFunkcjiInterpolowanej, wartosciWielomianuLagrange);

plik << "x,y,L,b,w,fw\n";

for (int i = 0; i < np + 1; i++)

{

plik << punktyX[i] << "," << wartosciFunkcjiInterpolowanej[i] << "," << wartosciWielomianuLagrange[i] << "," << blad[i];

if (i < wezly.size()) plik << "," << wezly[i] << "," << wartosciFunkcjiInterpolowanejDlaWezlow[i];

plik << endl;

}

plik.close();

}

int main()

{

int n;

int np;

int a;

int b;

bool optymalnie;

wejscie(n, np, a, b, optymalnie);

wyjscie(n, np, a, b, optymalnie);

}