LABORATORIUM METOD NUMERYCZNYCH

Aproksymacja

prowadzący: dr inż. Jerzy Respondek

Judyta Żurek
sekcja 12

Celem ćwiczenia była aproksymacja za pomocą wielomianów ortogonalnych Grama.

Badana funkcja: $y = \cos x \cdot \sin 2x^2$
Liczba punktów: $n = 50$
Przedział: $x \in \langle -2, 2 \rangle$

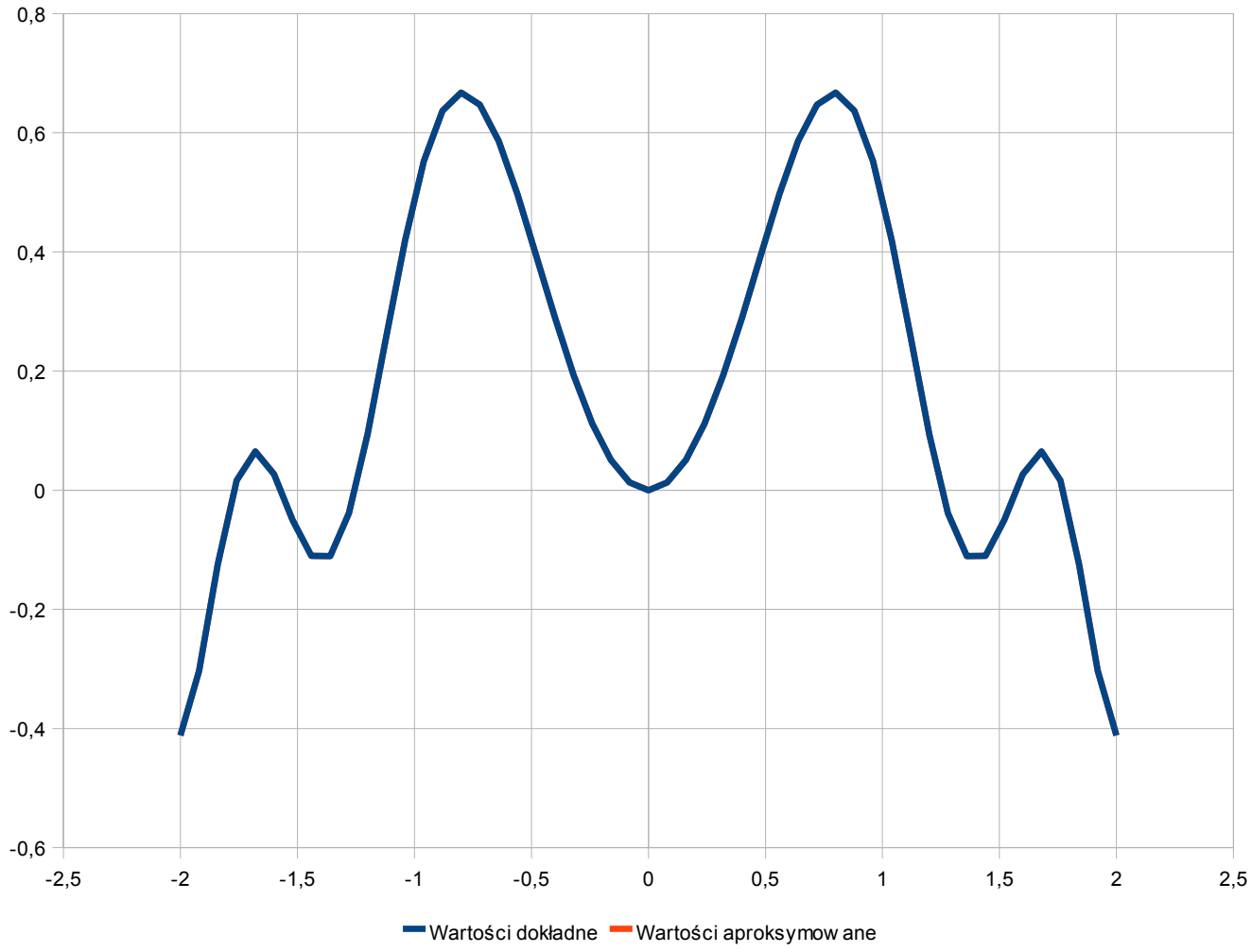
Stopień wielomianu	Wartość błędu aproksymacji
1	0.298454
2	0.221642
3	0.221642
4	0.207997
5	0.207997
6	0.183706
7	0.183706
8	0.0780541
9	0.0780541
10	0.0465286
11	0.0465286
12	0.016193
13	0.016193
14	0.00430233
15	0.00430233
16	0.00130273
17	0.00130273
18	0.000200744
19	0.000200748
20	5.15647e-005

Na podstawie powyższej tabeli można wywnioskować, że w przedziale <-2, 2> najlepszym przybliżeniem zadanej funkcji jest wielomian 20 stopnia, gdyż dla tego wielomianu błąd aproksymacji ma najmniejszą wartość. Możliwe, że jeszcze mniejszy błąd uzyskalibyśmy dla wielomianu wyższego stopnia, jednak w ćwiczeniu badamy wielomiany do stopnia 20 włącznie.

Wartości współczynników dla wielomianu 20 stopnia:

$a_0 = 0.17134$
$a_1 = -6.27954e-018$
$a_2 = -0.421391$
$a_3 = -8.42889e-017$
$a_4 = -0.188752$
$a_5 = 2.1318e-015$
$a_6 = 0.232734$
$a_7 = 3.40126e-014$
$a_8 = -0.337384$
$a_9 = 8.39768e-013$
$a_{10} = 0.0969293$
$a_{11} = -2.38063e-011$
$a_{12} = 0.0465196$
$a_{13} = -2.20596e-010$
$a_{14} = -0.0104245$
$a_{15} = -1.65644e-009$
$a_{16} = -0.00155912$
$a_{17} = 1.94923e-008$
$a_{18} = 0.00025328$
$a_{19} = -1.74479e-007$
$a_{20} = 1.79617e-005$

Wykresy funkcji aproksymowanej i aproksymującej:



Zmiana przedziału i ilości węzłów:

Dla wielomianu stopnia 20 i liczby węzłów n = 50 zmieniamy przedział (szerokość, początek i koniec):

Początek przedziału	Koniec przedziału	Szerokość	Wartość błędu
-5	5	10	0.416389
-1	1	2	3.91104e-006
0	1	1	3.95104e-006
4	6	2	0.0828249
0	2	2	2.18236e-006
1	3	2	7.93631e-006
2	4	2	0.000593475
-2	0	2	8.69424e-007
-3	3	6	0.0329386
-2,5	2,5	5	0.00124329
-2,25	2,25	4,5	0.000544236
5	7	2	0.485605
-7	7	14	0.281824
-10	10	20	0.309633

-7	2	9	0.357011
9	10	1	0.0502278
5	10	5	0.494945
3	6	3	0.283432

Dla przedziału <-5, 5> i stopnia 20 zmieniamy ilość węzłów:

Ilość węzłów	Wartość błędu
25	0.081738
30	0.310436
35	0.343909
40	0.323449
45	0.444183
55	0.387714
60	0.39576
65	0.399738
70	0.398162
100	0.400326
120	0.400825
150	0.401189
200	0.401489

Wygładzanie funkcji:
Dla przedziału <-5, 5> stopnia 20 wartości funkcji zostały zaburzone losowymi wartościami z przedziału <-0.1, 0.1>. Następnie zastosowano algorytm wygładzania SE13:

Ilość wygładzeń	Błąd aproksymacji
1	0.418004
2	0.419673
3	0.421345
4	0.423397
5	0.425292
6	0.42708
7	0.428672
8	0.430082
9	0.431314
10	0.432385
15	0.435971
20	0.437863

Wnioski:
Największy wpływ na błąd aproksymacji ma stopień wielomianu aproksymującego. Im wyższy stopień, tym błąd mniejszy, ale tylko do pewnego momentu. W tym przypadku dla wielomianów o stopniu wyższym od 14 wraz ze wzrostem stopnia wielomianu wzrasta również błąd aproksymacji (nieznacznie, ale jednak).
Szerokość przedziału oraz rozmieszczenie węzłów również mają znaczenie. Dla badanej funkcji najmniejszy błąd powstaje przy aproksymacji w przedziale o szerokości 2 lub mniejszej, o węzłach rozmieszczonych w przedziale <-2, 2>. Im przedział szerszy lub węzły bardziej oddalone od zera (wtedy szerokość przedziału przestaje mieć znaczenie), tym błąd aproksymacji większy. Wynikać to może ze specyfiki aproksymowanej funkcji.
Zwiększanie ilości węzłów również powoduje wzrost błędu aproksymacji. W przypadku badanej funkcji po przekroczeniu 100 węzłów błąd wzrasta nieznacznie. Zatem używanie bardzo dużej ilości węzłów mija się z celem – niepotrzebnie tylko wydłuża czas obliczeń.
Wielokrotne wygładzanie funkcji nieznacznie wpływa na wzrost wartości błędu aproksymacji.
Niewątpliwą zaletą aproksymacji w stosunku do interpolacji jest to, iż wielomian aproksymujący nie musi być bardzo wysokiego stopnia. W naszym przypadku do bardzo dobrego przybliżenia skomplikowanej funkcji wystarczy wielomian stopnia tylko 20.
Zastosowanie aproksymacji:

- określanie przybliżonego wzoru funkcji dla stablicowanych wartości
- przedstawianie funkcji o skomplikowanej postaci analitycznej w bardziej znośnej formie
- tworzenie grafiki w grach (aproksymacja mapy)
- usuwanie szumów przy obróbce dźwięku
- kompresja obrazów
- przewidywanie wartości akcji giełdowych

Listing programu:

```
#include <iostream>
#include <math.h>
#include <fstream>

using namespace std;


double P_pocz = -5;                    //--- początek przedziału
double P_kon = 5;                      //--- koniec przedziału
const int N = 50;                      //--- ilość węzłów
const int M = 20;                      //--- stopień wielomianu
const double H = (P_kon-P_pocz)/N;     //--- odległość między węzłami

double X[N + 1];            //--- węzły
double Y[N + 1];            //--- wartości funkcji
double YD[N + 1];           //--- dokładne wartości funkcji
double F[N + 1];            //--- wartości funkcji aproksymującej
double Z[N + 1];            //--- wartości wygładzone
double A[M + 1];            //--- wartości współczynników funkcji aproksymującej

double potega(double a, int n)
{
        double p = a;

        if (n == 0)
                return 1;

        for (int i = 1; i < n; ++i)
                p *= a;

        return p;
}
```

```
double silnia(int n)
{
	double s = 1;

	for (int i = 1; i < n; ++i)
		s = s * (i + 1);

	return s;
}

double symbol_newtona(int n, int k)
{
	return (silnia(n)) / (silnia(k) * silnia(n - k));
}

void wartosci_dokladne()
{
	for (int i = 0; i <= N; ++i)
	{
		X[i] = P_pocz + H * i;
		Y[i] = YD[i] = cos(X[i]) * sin(2 * (X[i] * X[i]));
	}
}

double wielomian_czynnikowy(double r, int s)
{
	double wynik = r;

	if (s == 0)
		return 1;

	for (int i = 1; i < s; ++i)
		wynik *= (r - i);

	return wynik;
}

double wielomian_grama(int k, double q)
{
	double suma = 0;

	for (int s = 0; s <= k; ++s)
		suma += potega(-1, s) * symbol_newtona(k, s)*symbol_newtona(k+s, s) *
		wielomian_czynnikowy(q, s)/wielomian_czynnikowy(N, s);

	return suma;
}

double funkcja_aproksymujaca(double x)
{
	double wynik, suma_C, suma_S;

	wynik = 0;

	for (int j = 0; j <= M; ++j)
	{
		suma_C = suma_S = 0;
		for (int q = 0; q <= N; ++q)
		{
			suma_S += wielomian_grama(j, q) * wielomian_grama(j, q);
```

```
                suma_C += wielomian_grama(j,q) * Y[q];
            }

            A[j] = suma_C / suma_S;
            wynik += A[j] * wielomian_grama(j, (x - P_pocz) / H);
        }

        return wynik;
}

double blad_aproksymacji()
{
        double suma = 0;

        for (int i = 0; i <= N; ++i)
            suma += potega(F[i]-YD[i], 2);

        return sqrt(suma/(N+1)) ;
}

void zaburzanie()
{
        for (int i = 0; i <= N; ++i)
            Y[i] = Y[i] + (rand()%20-10) / 100.0;

        return;
}

void wygladzanie_se13()
{
        Z[0] = (5*Y[0]+2*Y[1]-Y[2])/6;

        Z[N] = (5*Y[N]+2*Y[N-1]-Y[N-2])/6;

        for (int i = 1; i <= N; ++i)
            Z[i]=(Y[i+1] + Y[i] + Y[i-1])/3;

        for (int i = 0; i <= N; ++i)
            Y[i] = Z[i];

        return;
}

void zapis_do_pliku()
{
        ofstream plik;

        plik.open("raport.txt");
        plik << "Blad wynosi: " << blad_aproksymacji() << "\n\nWspolczynniki a[j]:\n\n";

        for (int i = 0; i <= M; ++i)
            plik<< i << "\t" << A[i] << endl;

        plik.close();

        return;
}

int main()
{
```

```
	wartosci_dokladne();

	zaburzanie();

	for (int i = 0; i < 20; ++i)
		wygladzanie_se13();

	for (int i = 0; i <= N; ++i)
		F[i] = funkcja_aproksymujaca(X[i]);

	zapis_do_pliku();

	return 0;
}
```