Marcin Rociek
Informatyka, II rok

Metody Obliczeniowe w Nauce i Technice

laboratorium

zestaw 1: interpolacja

Zadanie 1: Znaleźć wzór interpolacyjny Lagrange'a mając tablicę wartości:

x	0	2	3	5	6
y	1	3	2	5	6

Do rozwiązania tego zadania wykorzystałem funkcję **polcoe** z biblioteki *Numerical Recipes*. Funkcja ta wyznacza metodą Lagrange'a współczynniki wielomianu przechodzącego przez zadane punkty. Na wejściu otrzymuje tablicę ze współrzędnymi *x* i *y* zadanych punktów, ich ilość, oraz tablicę do której wpisywane są wartości współczynników wielomianu.

Treść programu:

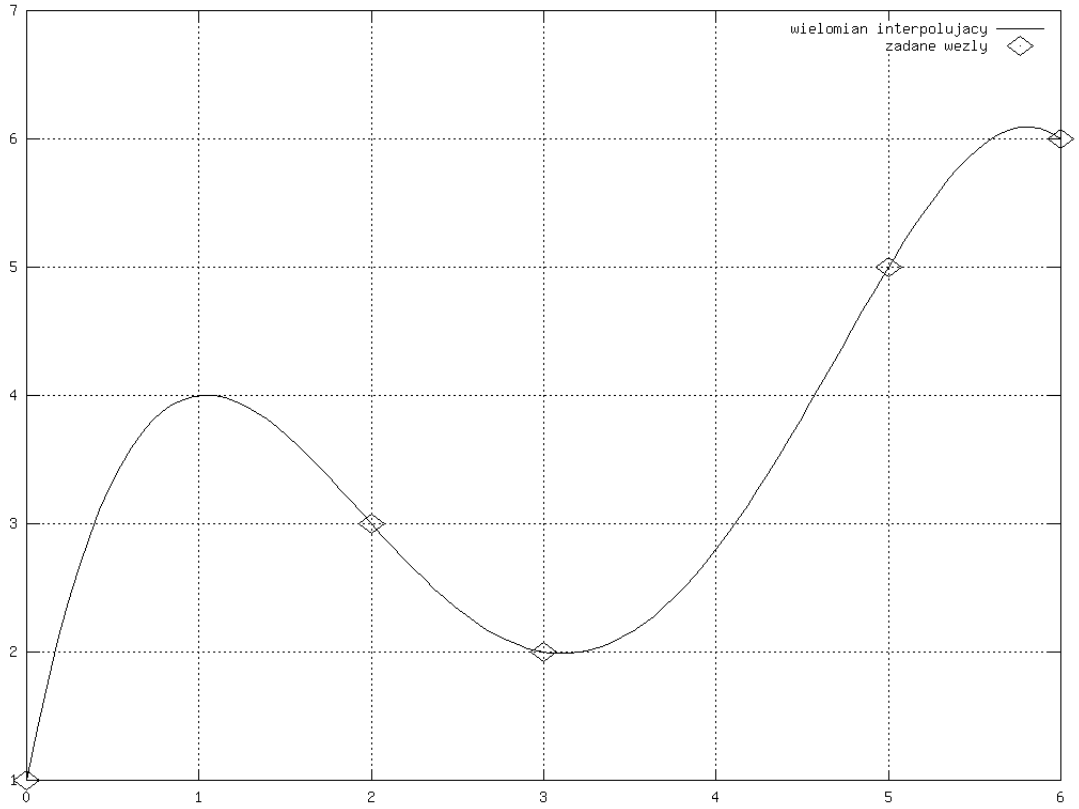
```
#include "nr.h"

void main(void)
{
  float x[] = { 0, 2, 3, 5, 6};
  float y[] = { 1, 3, 2, 5, 6};
  float coef[5];
  int i;

  polcoe(x, y, 4, coef);

  printf("znaleziony wielomian:\n");
  for(i=0; i<5; i++)
    printf("%f*x^%d\n", coef[i], i);
}
```

Wynik działania programu:

```
znaleziony wielomian:
1.000000*x^0
6.883333*x^1
-5.008333*x^2
1.216667*x^3
-0.091667*x^4
```

Wielomianem przechodzącym przez zadane punkty jest wielomian zadany wzorem:
$y=-0.091667x^4+1.216667x^3-5.008333x^2+6.883333x+1$

Poniżej znajduje się wykres funkcji tego wielomianu, wraz z naniesionymi węzłami interpolacji:



Zadanie 2: Znaleźć wielomian interpolacyjny co najwyżej 3-go stopnia z oszacowaniem błędu dla $y = \sqrt{x}$ na przedziale <1; 9/4>. Znane są węzły: 1, 25/16, 16/9, 9/4. Korzystając z zer wielomianu Czebyszewa znaleźć optymalny wielomian dla tej funkcji. Oszacować wielomian wg. $|f(x) - W_n(x)| \le \frac{M_{n+1}}{2^n(n+1)!}$

Do rozwiązania tego zadania posłużyłem się, podobnie jak w poprzednim zadaniu funkcją polcoe z bibloteki *Numerical Recipes*. Najpierw wyznaczane są współczynniki wielomianu na podstawie zadanych węzłów, a następnie za pomocą zer wielomianu Czebyszewa. Optymalnymi węzłami dla przedziału <*a*,*b*> są bowiem punkty:

$$x_m = \frac{1}{2}\left[(b-a)\cos\frac{2m+1}{2n}p + (b+a)\right]$$

Treść programu:

```
#include "../numrec/recipes/nr.h"
#include <math.h>

void main(void)
{
  float x[] = { 1, 25.0f/16.0f, 16.0f/9.0f, 9.0f/4.0f};
  float y[] = { 1,  5.0f/4.0f,   4.0f/3.0f, 3.0f/2.0f};
  float xc[4];
  float yc[4];
  float coef[4];
  int i, m;

  polcoe(x, y, 3, coef);
```

```
  printf("znaleziony wielomian:\n");
  for(i=0; i<4; i++)
    printf("%f*x^%d\n", coef[i], i);

  printf("\nwezly czebyszewa:\n");
  for(m=0; m<4; m++)
  {
    xc[m] = 0.5*((x[3]-x[0])*cos(PI*((2.0f*m+1.0f)/8.0f))+(x[3]+x[0]));
    yc[m] = sqrt(xc[m]);
    printf("xc=%f\tyc=%f\n", xc[m], yc[m]);
  }

  polcoe(xc, yc, 3, coef);

  printf("\noptymalny wielomian:\n");
  for(i=0; i<4; i++)
    printf("%f*x^%d\n", coef[i], i);
}
```

Wynik działania programu:

```
znaleziony wielomian:
0.387211*x^0
0.750417*x^1
-0.156742*x^2
0.019128*x^3

wezly czebyszewa:
xc=2.202425     yc=1.484057
xc=1.864177     yc=1.365349
xc=1.385823     yc=1.177210
xc=1.047575     yc=1.023511

optymalny wielomian:
0.385169*x^0
0.755223*x^1
-0.160022*x^2
0.019821*x^3
```

Wielomianem interpolującym dla zadanych węzłów jest wielomian zadany wzorem:
$y=0.019128x^3-0.156742x^2+0.750417x+0.387211$

Oszacowanie błędu:

$$f(x)=\sqrt{x}$$

$$f^{(4)}(x)=\frac{15}{16}x^{-\frac{7}{2}}$$

$$M_4=\sup_{x\in<1;9/4>}\left|f^{(4)}(x)\right|=\frac{15}{16}$$

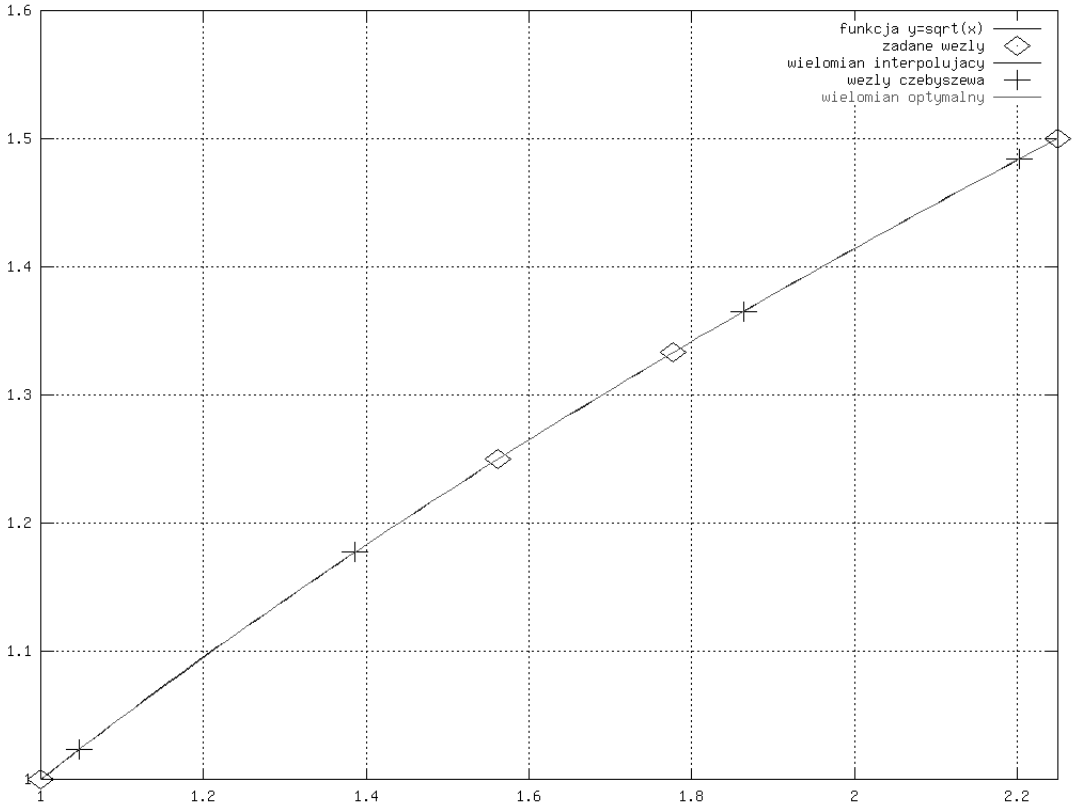
$$\left|f(x)-W(x)\right|\le\frac{15}{384}(x-1)\left(x-\frac{25}{16}\right)\left(x-\frac{16}{9}\right)\left(x-\frac{9}{4}\right)$$

Wielomianem optymalnym wykorzystującym jako węzły zera wielomianu Czebyszewa jest wielomian:
$y=0.019821x^3-0.160022x^2+0.755223x+0.385169$

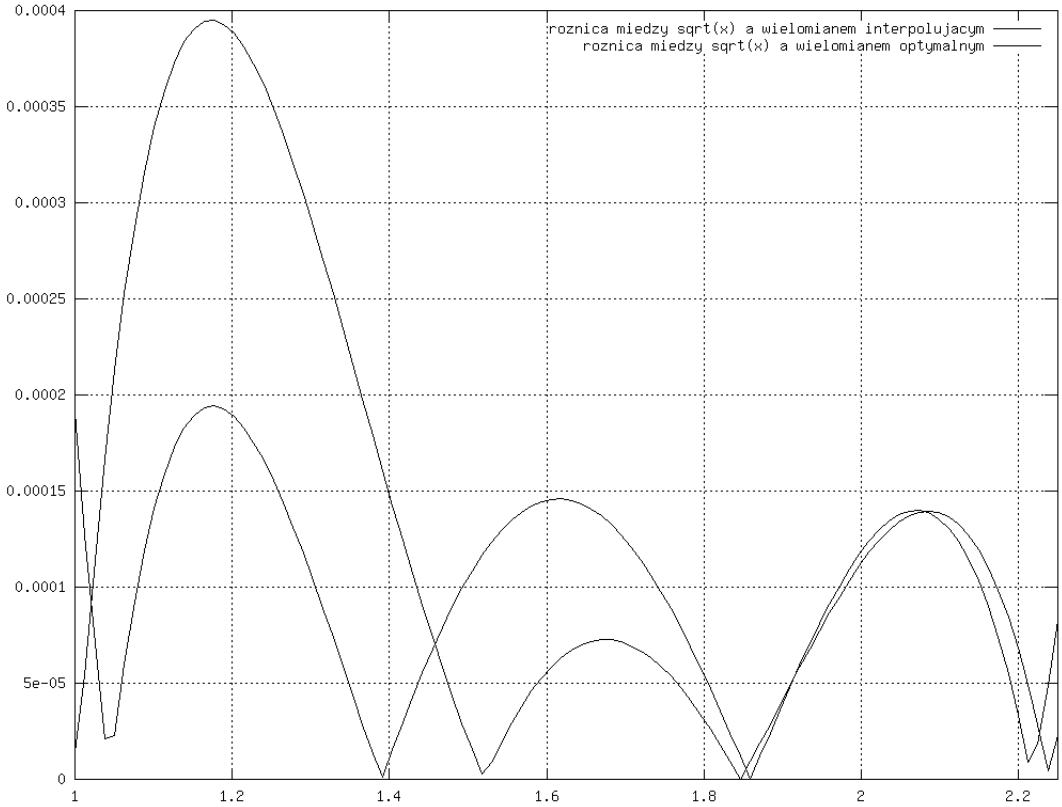
Oszacowanie błędu zgodnie z wzorem $\left|f(x)-W_n(x)\right|\le\frac{M_{n+1}}{2^n(n+1)!}$:

$$\left|f(x)-W(x)\right|\le\frac{M_4}{2^3 4!}\approx 0.0048828$$

Wykresy funkcji obu wielomianów niemal pokrywają się z wykresem funkcji interpolowanej:



Różnicę między wielomianem a intrepolowaną funkcją widać wyraźnie na wykresie funkcji $e(x) = \left| f(x) - W(x) \right|$, na którym również widać wpływ zastosowania węzłów Czebyszewa:



Zadanie 3: Obliczyć wartość wielomianu Newtona dla zad.1

Do rozwiązania tego zadania napisałem własny program wyznaczający współczynniki wielomianu interpolacyjnego metodą Newtona.
Wielomian Newtona ma postać:

$$W_n(x) = f(x_0) + f(x_0;x_1)w_0(x) + f(x_0;x_1;x_2)v_1(x) + \ldots + f(x_0;x_1;\ldots;x_n)v_{n-1}(x)$$

Gdzie:

$$v_n(x) = (x-x_0)(x-x_1)\mathbf{K}(x-x_n)$$

$$f(x_0;x_1) = \frac{f(x_1)-f(x_0)}{x_1-x_0},\ f(x_0;x_1;x_2) = \frac{f(x_1;x_2)-f(x_0;x_1)}{x_2-x_0}, \ldots.$$

Najpierw obliczane są potrzebne ilorazy różnicowe: wartości funkcji w kolejnych punktach zawarte w tablicy **y** zastępowane są przez kolejne ilorazy różnicowe, tzn: y_0, $f(x_0;x_1)$, $f(x_0;x_1;x_2)$, itd.
Następnie wyznaczane są współczynniki wielomianu interpolacyjnego: w kolejnych iteracjach następuje dodanie do wielomianu W wyrażenia $w \cdot f(x_0;\ldots;x_i)$, a następnie wymnożenie wielomianu w przez dwumian $(x-x_i)$.

Treść programu:

```
#define MAX 5

void main(void)
{
  float x[MAX] = { 0, 2, 3, 5, 6 };  /* dane wejsciowe */
  float y[MAX] = { 1, 3, 2, 5, 6 };

  float omega[MAX]  = { 1, 0, 0, 0, 0 }; /* (x-x0)*(x-x1)*...*(x-xi) */
  float w[MAX]      = { 0, 0, 0, 0, 0 }; /* szukany wielomian */

  int i, j, k;

  /* obliczenie potrzebnych ilorazow roznicowych */
  for(j=0; j<MAX-1; j++)
    for(i=0; i<MAX-j-1; i++)
      y[MAX-i-1] = ((y[MAX-i-1]-y[MAX-i-2])/(x[MAX-i-1]-x[MAX-i-j-2]));

  /* obliczenie wspolczynnikow wielomianu Newtona */
  for(i=0; i<MAX; i++)
  {
    for(k=0; k<i+1; k++)
      w[k] += omega[k]*y[i];
    for(k=MAX-1; k>0; k--)
      omega[k] = omega[k-1]-x[i]*omega[k];
    omega[0] = -x[i]*omega[0];
  }

  printf("znaleziony wielomian:\n");
  for(i=0; i<MAX; i++)
    printf("%f*x^%d\n", w[i], i);
}
```

Wynik działania programu:

```
znaleziony wielomian:
1.000000*x^0
6.883334*x^1
-5.008334*x^2
1.216667*x^3
-0.091667*x^4
```

Wielomianem interpolacyjnym jest więc wielomian
$y=-0.091667x^4+1.216667x^3-5.008333x^2+6.883333x+1$
który jest identyczny z wielomianem wyznaczonym metodą Lagrange'a (a więc został wyznaczony poprawnie).

Zadanie 4: Funkcja $f(x)=e^x/2$ dana jest na przedziale <3.5; 3.8> za pomocą tablicy w odstępach 0,1. Obliczyć wartości funkcji w przedziale z odstępem 0,01.

Wielomian interpolacyjny Lagrange'a dla równoodległych wartości argumentu ma postać:

$$W_n(x) = y_0 + \frac{\Delta y_0}{h} w_0(x) + \frac{\Delta^2 y_0}{2!h^2} v_1(x) + \ldots + \frac{\Delta^n y_0}{n!h^n} v_{n-1}(x)$$

gdzie:

$$v_n(x) = (x - x_0)(x - x_1)\mathbf{K}(x - x_n)$$

$$\Delta y_0 = y_1 - y_0,\ \Delta^2 y_0 = \Delta y_1 - \Delta y_0, \ldots$$

Do rozwiązania tego zadania wykorzystałem zmodyfikowaną nieco wersję programu z zadania 3. Różnica polega na tym, że zamiast ilorazów różnicowych wyliczane są różnice progresywne, a w pętli wyznaczającej współczynniki wielomianu dochodzi jeszcze wyznaczanie współczynnika $1/(h^2 i!)$.

Treść programu:

```
#define MAX 4

void main(void)
{
  float x[MAX] = {      3.5,      3.6,      3.7,      3.8 };
  float y[MAX] = { 16.557726, 18.299117, 20.223652, 22.350592 };
  float newy[41];

  float omega[MAX]  = { 1, 0, 0, 0, }; /* (x-x0)*(x-x1)*...*(x-xi) */
  float w[MAX]      = { 0, 0, 0, 0, }; /* szukany wielomian */
  float f = 1;
  float h = x[1]-x[0];

  int i, j, k;

  /* obliczenie potrzebnych roznic */
  for(j=0; j<MAX-1; j++)
    for(i=0; i<MAX-j-1; i++)
      y[MAX-i-1] = y[MAX-i-1]-y[MAX-i-2];

  /* obliczenie wspolczynnikow wielomianu Newtona */
  for(i=0; i<MAX; i++)
  {
    for(k=0; k<i+1; k++)
      w[k] += omega[k]*y[i]*f;
    for(k=MAX-1; k>0; k--)
      omega[k] = omega[k-1]-x[i]*omega[k];
    omega[0] = -x[i]*omega[0];
    f *= 1/(h*(float)(i+1));
  }

  printf("znaleziony wielomian:\n");
  for(i=0; i<MAX; i++)
    printf("%f*x^%d\n", w[i], i);

  printf("\nwartosci wielomianu dla x od 3.5 do 3.8 co 0.01:\n");
  printf("x\ty\n");
  for(f=3.5; f<=3.8; f+=0.01)
  {
    float y = 0;
    float xt = 1;
    for(i=0; i<MAX; i++)
    {
      y += xt*w[i];
      xt *= f;
    }
    printf("%.2f\t%f\n", f, y);
  }
}
```

Wynik działania programu:

znaleziony wielomian:
-78.648056*x^0
77.160469*x^1
-25.508104*x^2
3.209759*x^3

wartosci wielomianu dla x od 3.5 do 3.8 co 0.01:

x	y
3.50	16.557726
3.51	16.724178
3.52	16.892273
3.53	17.062056
3.54	17.233522
3.55	17.406723
3.56	17.581661
3.57	17.758341
3.58	17.936792
3.59	18.117041
3.60	18.299114
3.61	18.483019
3.62	18.668766
3.63	18.856384
3.64	19.045851
3.65	19.237284
3.66	19.430635
3.67	19.625914
3.68	19.823172
3.69	20.022411
3.70	20.223639
3.71	20.426912
3.72	20.632225
3.73	20.839602
3.74	21.049057
3.75	21.260614
3.76	21.474302
3.77	21.690094
3.78	21.908085
3.79	22.128248
3.80	22.350599

Wielomianem interpolującym jest wielomian
$y=3.209759x^3-25.508104x^2+77.160469x-78.648056$
Za jego pomocą wyznaczone zostały wartości funkcji w przedziale z krokiem 0,01.