Wojciech Ładyga - zadanie 10

Język technologia: c++, GSL

Aby wyliczyć jawnie współczynniki wielomianu interpolacyjnego na podstawie danych wejściowych zawartych w tablicach xTab[N] i yTab[N] można wprost stworzyć układ równań i go rozwiązać.

Ja zastosowałem dekompozycję LU do rozwiązania takiego układu i wyniki zapisałem do wektora a przechowującego współczynniki wielomianu.
Do wypisania wartości zastosowałem precyzje 4 `setprecision(4)`.

Aby narysować wykres zastosowałem polecenia

```
./a.out > interp.dat
graph -T ps < interp.dat > interp.ps
```

aby wypisać punkty które były wykorzystane do konstrukcji zastosowałem metodę:

```
printf("#m=0,S=2\n");

double xTab[] = {0.062500, 0.187500, 0.312500, 0.437500, 0.562500, 0.687500,
0.812500, 0.935700};
double yTab[] = {0.687959, 0.073443, -0.517558, -1.077264, -1.600455, -2.080815,
-2.507266, -2.860307};

for (int j = 0; j < N; j++)
{
    printf("%g %g\n", xTab[j], yTab[j]);
}
printf("#m=1,S=0\n");
```

a resztę punktów tworzących cały wykres stworzyłem w pętli od -1 do 1 przemieszczając się o 0.0001 i następnie wyliczając ten wielomian na podstawie wszystkich dostarczonych punktów.

Generowanie wykresu oparłem o przykład ze strony https://www.gnu.org/software/gsl/doc/html/interp.html. Wykorzystywany jest tam gnu plotunit.

Kod programu:

```
/*
* @Author: Wojciech Ladyga
* @Date: 2018-12-24
* @Description: Zad 10
*/
#include <iostream>
#include <iomanip>
```

```
#include <gsl/gsl_errno.h>
#include <gsl/gsl_linalg.h>
#include <gsl/gsl_eigen.h>
using namespace std;

//wyklad
//https://pl.wikipedia.org/wiki/Interpolacja_wielomianowa
//http://www.algorytm.org/procedury-numeryczne/interpolacja-wielomianowa.html
//+dokumentacja GSL 2.5

const int N = 8;

//funkcja przelicza wektor zawierający współczynniki
gsl_vector *lu(gsl_vector *x, gsl_vector *y, gsl_vector *a, gsl_permutation
*permutation, gsl_matrix *countMat)
{
    for (int i = 0; i < N; i++)
    {
        for (int j = 0; j < N; j++)
        {
            gsl_matrix_set(countMat, i, j, pow(gsl_vector_get(x, i), N - 1 - j));
        }
    }

    //rozwiązujemy macierz stosując dekompozycje LU
    int tmp;
    gsl_linalg_LU_decomp(countMat, permutation, &tmp);
    gsl_linalg_LU_solve(countMat, permutation, y, a);

    return a;
}

//funkcja przyjmuje dane i wyswietla wynik operacji
void wspolczynniki(double xTab[], double yTab[])
{
    //ogolnie a(n)x^(n)+a(n-1)x^(n-1)+... = y

    gsl_vector *a = gsl_vector_alloc(N);                    //vector na
współczynniki
    gsl_permutation *permutation = gsl_permutation_alloc(N); //wektor permutacji
do dekompozycji LU
    //macierz pozwalająca umiescic odpowiednie wartosci typu ax^n
    gsl_matrix *countMat = gsl_matrix_alloc(N, N);

    //kopiuje zawartosc tablic do wektorow
    gsl_vector *x = gsl_vector_alloc(N);
    gsl_vector *y = gsl_vector_alloc(N);
    gsl_vector_view tmp_x = gsl_vector_view_array(xTab, N);
    gsl_vector_view tmp_y = gsl_vector_view_array(yTab, N);
    gsl_vector_memcpy(x, &tmp_x.vector);
    gsl_vector_memcpy(y, &tmp_y.vector);
```

```
    //wypisz wartosci
    for (int i = N - 1; i >= 0; i--)
    {
        cout << "a" << N - 1 - i << " = " << setprecision(4) << fixed <<
gsl_vector_get(lu(x, y, a, permutation, countMat), i) << endl;
    }

    double yy;
    for (double xx = -1.0; xx <= 1.0; xx += 0.0001)
    {
        yy = 0;
        for (int i = 0; i < N; i++)
        {
            yy += gsl_vector_get(lu(x, y, a, permutation, countMat), i) * pow(xx,
N - 1 - i);
        }
        printf("%g %g\n", xx, yy);
    }

    //zwalnianie pamięci
    gsl_vector_free(x);
    gsl_vector_free(y);
    gsl_matrix_free(countMat);
    gsl_vector_free(a);
}

int main()
{
    printf("#m=0,S=2\n");
    //dane wejsciowe
    double xTab[] = {0.062500, 0.187500, 0.312500, 0.437500, 0.562500, 0.687500,
0.812500, 0.935700};
    double yTab[] = {0.687959, 0.073443, -0.517558, -1.077264, -1.600455,
-2.080815, -2.507266, -2.860307};

    for (int j = 0; j < N; j++)
    {
        printf("%g %g\n", xTab[j], yTab[j]);
    }
    printf("#m=1,S=0\n");

    wspolczynniki(xTab, yTab);
    return 0;
}
```

Wyniki działania programu to:

```
a0 = 1.0010
a1 = -5.0304
a2 = 0.3262
a3 = 0.3130
a4 = 2.7206
a5 = -6.3099
a6 = 5.8857
a7 = -1.9207
```

oraz wykres