Metody numeryczne

Wojciech Chrobak

12 grudnia 2017

Zadanie 4 - obowiązkowe

GSL

W bibliotece GSL mamy gotowe funkcje do konstruowania naturalnego splajnu kubicznego.

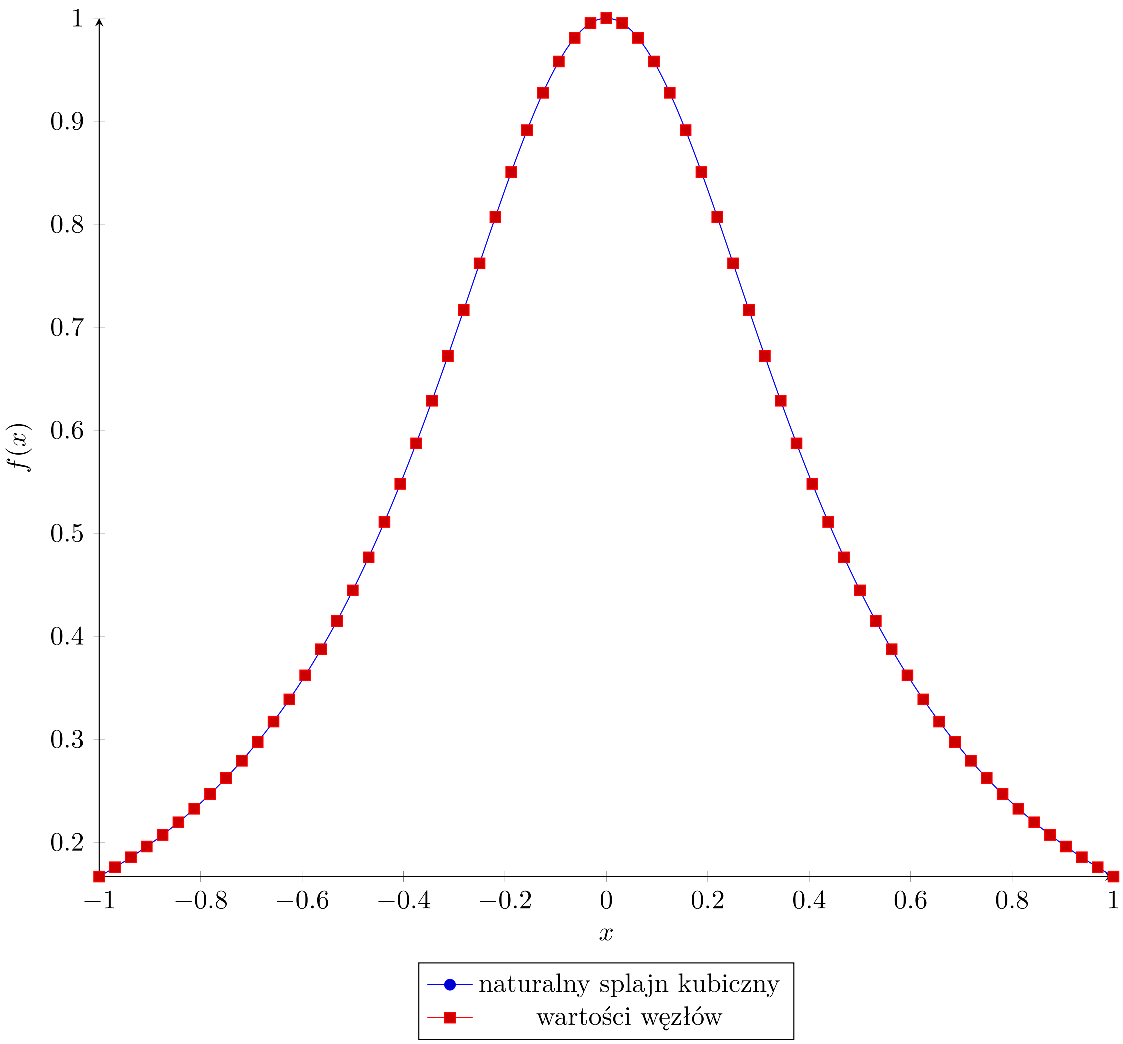
$gsl\_interp\_accel\ *acc = gsl\_interp\_accel\_alloc()$; - obszar do interpolacji
$gsl\_spline\ *spline = gsl\_spline\_alloc(gsl\_interp\_cspline,\ 65)$; - splajn typu cspline czyli naturalny splajn kubiczny
$gsl\_spline\_init(spline,\ x,\ y,\ 65)$; - inicjalizuje splajn dla 65 węzłów
$gsl\_spline\_eval(spline,\ j,\ acc)$; - wartosc splajnu dla punktu j

Kod

```
#include <iostream>
#include <gsl/gsl_errno.h>
#include <gsl/gsl_spline.h>
#include <fstream>

using namespace std;

int main() {
    ofstream file, file2;
    file2.open("input.dat"); // plik do wezlow i jego wartosci
    file.open("splines.dat"); // plik do wartosci splajnu
    int i;
    double xi, yi, x[65], y[65];

    // wypelnianie tablic wezlow i jego wartosci
    for (i = 0; i < 65; i++) {
        x[i] = -1 + i / 32.0;
        y[i] = 1 / (1 + 5 * x[i] * x[i]);
        file2 << x[i] << " " << y[i] << endl;
    }

    cout << endl << endl;
    gsl_interp_accel *acc = gsl_interp_accel_alloc();
    gsl_spline *spline = gsl_spline_alloc(gsl_interp_cspline, 65);

    gsl_spline_init(spline, x, y, 65);

    // obliczanie wartosci splajnu
    for (double j = -1.0; j <= 1.0; j = j + 0.01) {
        yi = gsl_spline_eval(spline, j, acc);
        file << j << " " << yi << endl;
```

```
    }

    gsl_spline_free(spline);
    gsl_interp_accel_free(acc);
    file.close();
    file2.close();
    return 0;
}
```

Wykres



Standardowa metoda

Aby skonstruować naturalny splajn kubiczny dla równoodległych węzłów należy rozwiązać układ:

$$\begin{bmatrix} 4 & 1 & & & & \\ 1 & 4 & 1 & & & \\ & 1 & 4 & 1 & & \\ & & ... & ... & ... & \\ & & & 1 & 4 & 1 \\ & & & & 1 & 4 \end{bmatrix} \begin{bmatrix} \xi_2 \\ \xi_3 \\ \xi_4 \\ ... \\ \xi_{n-2} \\ \xi_{n-1} \end{bmatrix} = \frac{6}{h^2} \begin{bmatrix} f_1 - 2f_2 + f_3 \\ f_2 - 2f_3 + f_4 \\ f_3 - 2f_4 + f_5 \\ ... \\ f_{n-3} - 2f_{n-2} + f_{n-1} \\ f_{n-2} - 2f_{n-1} + f_n \end{bmatrix}$$

$$h = x_{j+1} - x_j$$

Następnie w każdym przedziale $[x_j, x_{j+1}]$, $j = 1, 2, ..., n-1$, używamy wielomianu:

$$y_j(x) = Af_j + Bf_{j+1} + C\xi_j + D\xi_{j+1}$$

gdzie:

$$A = \frac{x_{j+1} - x}{x_{j+1} - x_j}$$

$$B = \frac{x - x_j}{x_{j+1} - x_j}$$

$$C = \frac{1}{6}(A^3 - A)(x_{j+1} - x_j)^2$$

$$D = \frac{1}{6}(B^3 - B)(x_{j+1} - x_j)^2$$