

Sprawozdanie  
Metody numeryczne 2  
**Temat 4, Zadanie nr 12**

Mateusz Śliwakowski, F4

4/12/2018

## 1 Treść zadania

Wzory empiryczne. Baza:  $1, x, y, x^2y^2$ . Tablicowanie błędów w punktach pomiarowych oraz obliczanie błędu średniokwadratowego. Punkty pomiarowe wybieramy z prostokąta  $[a, b] \times [c, d]$ .

## 2 Opis metody

Dana jest baza funkcji:  $1, x, y, x^2y^2$ , nazwijmy je  $g_1, g_2, g_3, g_4$ . Za ich pomocą chcemy aproksymować daną funkcję  $f$  funkcją  $f^* = \sum_{j=1}^4 a_j g_j$  w taki sposób, aby  $H = \sum_{i=1}^N (f(x_i) - f^*(x_i))^2 \rightarrow \min$ . Zadanie sprowadza się do znalezienia współczynników  $a_1, a_2, a_3, a_4 \in \mathbb{R}$ .  $N$  to liczba punktów pomiarowych. Najczęściej powyższej metody używa się, gdy do danych zebranych doświadczalnie chcemy jak najlepiej dopasować krzywą.

Chcemy znaleźć minimum lokalne funkcji  $H$ :

$$\begin{aligned} \frac{\partial H}{\partial a_k} (a_1, \dots, a_4) \sum_{i=1}^N 2(f(x_i) - \sum_{j=1}^4 a_j g_j(x_i))(0 - g_k(x_i)) &= 0 \\ - \sum_{i=1}^N f(x_i) g_k(x_i) + \sum_{i=1}^N \sum_{j=1}^4 a_j g_j(x_i) g_k(x_i) &= 0 \\ \sum_{j=1}^4 a_j \sum_{i=1}^N g_j(x_i) g_k(x_i) = \sum_{i=1}^N f(x_i) g_k(x_i), &\text{ gdzie } k = 1, \dots, 4 \end{aligned}$$

Otrzymaliśmy układ równań normalnych:

$$\sum_{j=1}^4 a_j \langle g_j, g_k \rangle = \langle f, g_k \rangle, \text{ gdzie } k = 1, \dots, 4$$

Układ ten można zapisać jako  $Ga = d$ , gdzie:

- $G$  jest macierzą Grama ( $G_{jk} = \langle g_j, g_k \rangle, j, k = 1, \dots, n$ )

- $a = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix}$

- $d = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \end{bmatrix}, d_k = \langle f, g_k \rangle, k = 1, \dots, 4$

Macierz Grama można zapisać jako  $G = M^T M$ , gdzie

$$M = \begin{bmatrix} g_1(x_1) & g_2(x_1) & g_3(x_1) & g_4(x_1) \\ g_1(x_2) & g_2(x_2) & g_3(x_2) & g_4(x_2) \\ \vdots & \vdots & \vdots & \vdots \\ g_1(x_n) & g_2(x_n) & g_3(x_n) & g_4(x_n) \end{bmatrix}$$

A wektor  $d = M^T F$ , gdzie  $F = \begin{bmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_n) \end{bmatrix}$ .

### 3 Warunki i założenia

1. Punkty pomiarowe są generowane losowo z prostokąta  $[a, b] \times [c, d]$ .
2. Dane  $a, b, c, d, n$  są podane prawidłowo.
3.  $f$  jest ograniczona na danym obszarze.

### 4 Implementacja

Funkcja realizująca założenia zadania to *lsfApproximation*.

```
function [fApprox, tab, err] = lsfApproximation(f,n,a,  
b,c,d)
```

Parametry wejściowe:

- $f$  - uchwyt do aproksymowanej funkcji dwóch zmiennych,
- $n$  - ilość punktów pomiarowych,
- $a, b, c, d$  - liczby rzeczywiste, definiujące prostokąt  $[a, b] \times [c, d]$ .

Parametry wyjściowe:

- $fApprox$  - uchwyt wynikowej funkcji ( $f^*$ ),
- $tab$  - tablica wynikowa zawierająca w kolumnach kolejno: współrzędne  $x$  oraz  $y$  punktu, wartość funkcji aproksymowanej w tym punkcie, wartość funkcji wynikowej, błąd,
- $err$  - wartość błędu średniokwadratowego.

W implementacji postępujemy zgodnie z algorytmem przedstawionym w punkcie *Opis metody*. Najpierw definiujemy funkcje z bazy i losujemy punkty pomiarowe. Potem przechodzimy do wyznaczenia macierzy kolejno:  $M$ ,  $G$ ,  $F$ ,  $d$ ,  $a$  korzystając z operatorów Matlaba. Na koniec wszystkie wyniki tablicujemy oraz obliczamy błąd średniokwadratowy.

## 5 Przykłady i wnioski

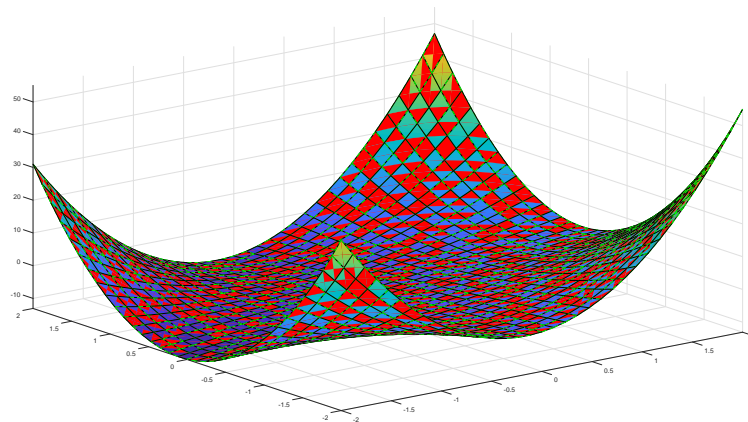
Na potrzeby prezentacji przykładów posłużymy się pomocniczą funkcją *presentResult*, która wyświetla funkcję aproksymowaną i wynikową na jednym wykresie oraz konstruuje tablicę wynikową w przystępnym formacie.

### 5.1 Funkcja z bazy

Rozpocznijmy od sprawdzenia aproksymacji dla funkcji z bazy.

```
f = @(x,y) -5 + 4.*x - 2.*y + 3.*y.*y.*y.*y.*x.*x.*x.*  
x;  
presentResult(f,10,-2,2,-2,2);
```

Otrzymujemy wynik z dokładnością maszynową. Na wykresie funkcja aproksymowana i wynikowa pokrywają się.



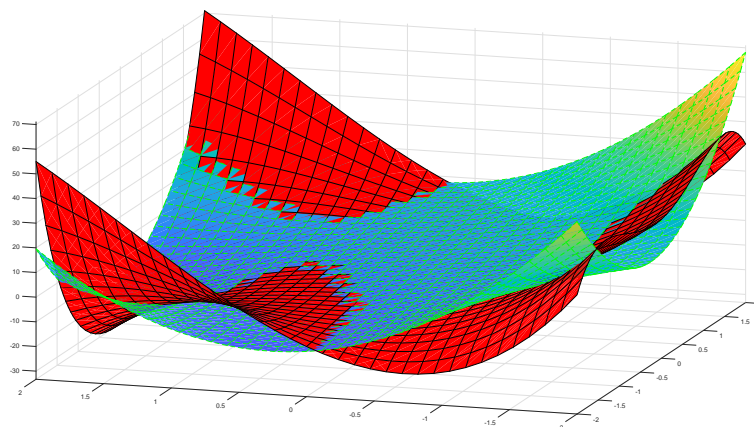
Wartość błędu średniokwadratowego wynosi  $2.0583e - 15$ . Porównywalne rezultaty otrzymujemy dla innych funkcji z bazy.

### 5.2 Funkcja wielomianowa wyższego stopnia

Następnie sprawdzimy działanie dla funkcji wielomianowej wyższego stopnia.

```
f = @(x,y) -5 + 4.*x - 2.*y + 3.*y.*y.*x.*x + 1.5.*x.*  
x.*x.*x.*y - 3.*y.*y.*y;  
presentResult(f,10,-2,2,-2,2);
```

Kolorem czerwonym zaznaczona jest funkcja aproksymowana, zielonym - wynikowa.

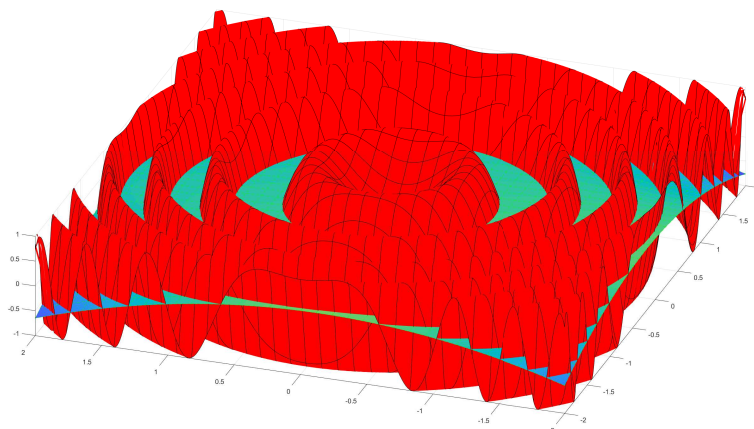


Błąd średniokwadratowy wynosi 1.5181 - funkcje z bazy nie pozwalają oddać kształtu danej funkcji, zwłaszcza gdy jej pochodna osiąga duże wartości.

### 5.3 Funkcja trygonometryczna, silnie oscylująca

```
f = @(x,y) sin(5.*(x.*x+y.*y));  
presentResult(f,40,-2,2,-2,2);
```

Za pomocą danej bazy nie jest możliwe dokładnie odwzorować kształtu tego typu funkcji. Funkcja przyjmuje intuicyjnie oczekiwany kształt - przechodzi w przybliżeniu przez średnią wartość funkcji.



Błąd średniokwadratowy w tym przypadku wynosi 0.76338.

## 5.4 Badanie wpływu ilości punktów pomiarowych na błąd średniokwadratowy

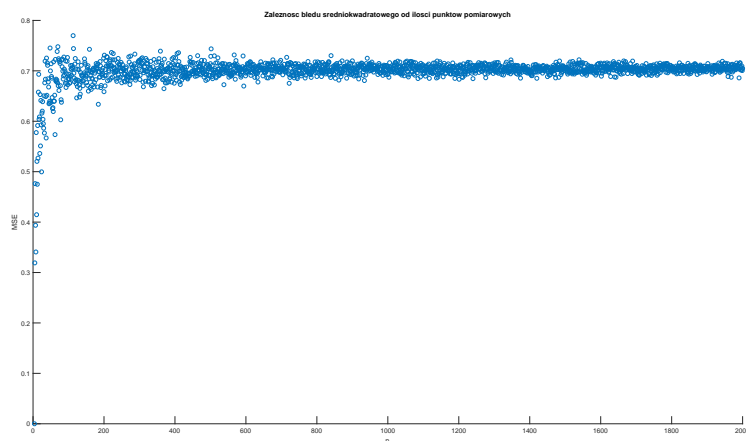
Weźmy funkcję  $f$ :

```
f = @(x,y) sin(5.*(x.*x+y.*y));
```

Wykonamy poniższy skrypt i na podstawie wynikowej tabeli utworzymy wykres.

```
errors = zeros(1,1996);  
for i = 4:1:2000  
    [~, ~, errors(i-3)] = lsfitApproximation(f,i,-2,2,-2,2);  
end
```

Wykres wygląda następująco:



Można zauważyć, że wartości błędu kwadratowego zbiegają do pewnej wartości - jest to wartość błędu dla aproksymacji średniokwadratowej ciągłej. Dla  $n = 4$  wartość błędu średniokwadratowego wynosi 0 - mamy wtedy faktycznie doczynienia z zadaniem interpolacji, czyli funkcja wynikowa w punktach pomiarowych przyjmuje wartości dokładne.

## 5.5 Wnioski końcowe

- Dokładność maszynową osiągamy tylko dla funkcji z bazy.
- Dla złożonych funkcji nie jesteśmy w stanie oddać dokładnego kształtu za pomocą danej bazy.
- Funkcja wynikowa przyjmuje intuicyjnie oczekiwany kształt.

- Wraz ze wzrostem ilości punktów pomiarowych wartość błędu średniokwadratowego zbliża się do wartości błędu dla aproksymacji średniokwadratowej ciągłej.