

## Optymalizacja funkcji wielu zmiennych metodami bezgradientowymi

### 1. Cel ćwiczenia.

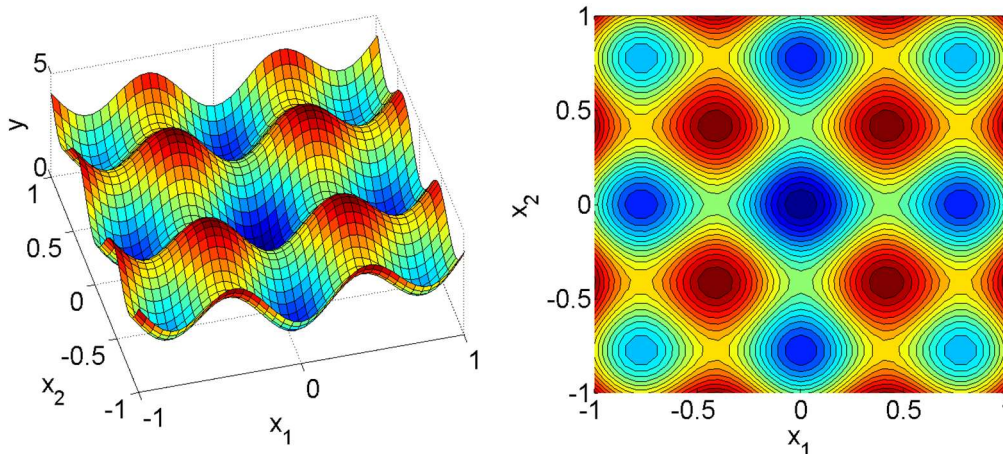
Celem ćwiczenia jest zapoznanie się z metodami bezgradientowymi poprzez ich implementację oraz wykorzystanie do rozwiązania problemu optymalizacji.

### 2. Testowa funkcja celu.

Funkcja celu dana jest wzorem:

$$f(x_1, x_2) = x_1^2 + x_2^2 - \cos(2,5\pi x_1) - \cos(2,5\pi x_2) + 2$$

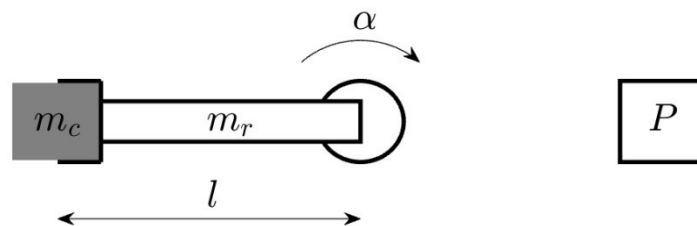
Jej wykres przedstawiony jest poniżej.



Punkt startowy powinien należeć do przedziału  $x_1^{(0)} \in [-1, 1]$ ,  $x_2^{(0)} \in [-1, 1]$ .

### 3. Problem rzeczywisty.

Ramie robota o długości  $l = 1 \text{ m}$  oraz masie  $m_r = 1 \text{ kg}$  ma za zadanie umieścić ciężarek o masie  $m_c = 5 \text{ kg}$  na platformie  $P$ . W tym celu ramie musi wykonać obrót o kąt  $\pi \text{ rad}$  i zatrzymać się.



Ruch ramienia opisany jest równaniem:

$$I \frac{d^2 \alpha}{dt^2} + b \frac{d\alpha}{dt} = M(t),$$

gdzie:  $b = 0.5 \text{ Nms}$  jest współczynnikiem tarcia, a moment bezwładności ramienia z ciężarkiem  $I$  wynosi:

$$I = \frac{1}{3} m_r l^2 + m_c l^2.$$

Moment siły przykładany do ramienia wyznaczany jest ze wzoru:

$$M(t) = k_1 (\alpha_{ref} - \alpha(t)) + k_2 (\omega_{ref} - \omega(t)),$$

gdzie:  $\omega = \frac{d\alpha}{dt}$ ,  $\alpha_{ref} = \pi \text{ rad}$ ,  $\omega_{ref} = 0 \text{ rad/s}$ ,  $k_1$  oraz  $k_2$  współczynniki wzmocnienia regulatora.

Celem optymalizacji jest znalezienie takich wartości współczynników wzmocnienia  $k_1$  oraz  $k_2$ , dla których funkcjonal jakości:

$$Q(k_1, k_2) = \int_0^{t_{end}} \left( 10 (\alpha_{ref} - \alpha(t))^2 + (\omega_{ref} - \omega(t))^2 + (M(t))^2 \right) dt$$

przyjmuje najmniejszą wartość. Początkowe wartości współczynników wzmocnienia powinny należeć do przedziału:  $k_1^{(0)} \in [0, 10] \text{ Nm}$ ,  $k_2^{(0)} \in [0, 10] \text{ Nms}$ . Symulacje należy przeprowadzać dla czasu od  $t_0 = 0$  do  $t_{end} = 100 \text{ s}$  z krokiem  $dt = 0.1 \text{ s}$ .

W celu sprawdzenia poprawności implementacji modelu oraz obliczenia funkcji celu, można obliczyć wartość funkcji celu dla  $k_1 = 5 \text{ Nm}$  oraz  $k_2 = 5 \text{ Nms}$ . Zakładając, że całka liczona jest metodą prostokątów, wartość funkcji celu powinna wynosić  $Q(k_1, k_2) \approx 244.784$ .

#### 4. Algorytmy optymalizacji.

Optymalizację należy przeprowadzić metodą Hooke'a-Jeevesa oraz metodą Rosenbrocka.

#### 5. Zadanie do samodzielnego wykonania.

##### a. Testowa funkcja celu.

Zadanie polega na wykonaniu 100 optymalizacji dla trzech różnych długości kroku startując z losowego punktu (jeżeli w dwóch sprawozdaniach pojawią się identyczne punkty startowe będą one ocenione na 0 punktów). Wyniki należy zestawić pliku xlsx w tabeli 1. Wartości średnie (tylko dla optymalizacji zakończonych znalezieniem minimum globalnego) należy przedstawić w tabeli 2. Dodatkowo, dla jednego wybranego przypadku należy na wykres poziomicy funkcji celu nanieść rozwiązania optymalne uzyskane po każdej iteracji (rozwiązania bazowe dla metody Hooke'a-Jeevesa).

##### b. Problem rzeczywisty.

Zadanie polega na przeprowadzeniu optymalizacji dla jednej długości kroku. Wyniki należy zestawić w tabeli 3. Dla znalezionych, optymalnych wartości współczynników  $k_1$  oraz  $k_2$  należy przeprowadzić symulację, a jej wyniki wstawić do arkusza Symulacja. Na ich podstawie należy narysować wykresy przedstawiające położenie oraz prędkość ramienia.

## 6. Sprawozdanie.

Sprawozdanie powinno zostać przygotowane w formacie docx (lub doc) albo pdf i powinno zawierać parametry poszczególnych algorytmów, dyskusję wyników oraz wnioski. Dodatkowo, w sprawozdaniu należy umieścić kod zaimplementowanych metod, funkcję lab2 oraz funkcje wykorzystane do obliczenia funkcji celu i pochodnych podczas rozwiązywania równań różniczkowych. Wyniki optymalizacji oraz wykresy należy przygotować w formacie xlsx (lub xls).

### Pseudokod metody Hooke'a-Jeevesa.

**Dane wejściowe:** punkt startowy  $x$ , długość kroku  $s$ , współczynnik zmniejszania długości kroku  $\theta < \alpha < 1$ , dokładność  $\epsilon > 0$ , maksymalna liczba wywołań funkcji celu  $N_{\max}$

```
1:  repeat
2:       $x^B = x$ 
3:       $x = \text{PRÓBUJ}(x^B, s)$ 
4:      if  $f(x) < f(x^B)$  then
5:          repeat
6:               $\underline{x}^B = x^B$ 
7:               $x^B = x$ 
8:               $x = 2x^B - \underline{x}^B$ 
9:               $x = \text{PRÓBUJ}(x, s)$ 
10:             if  $f_{\text{calls}} > N_{\max}$  then
11:                 return error
12:             end if
13:         until  $f(x) \geq f(x^B)$ 
14:          $x = x^B$ 
15:     else
16:          $s = \alpha \cdot s$ 
17:     end if
18:     if  $f_{\text{calls}} > N_{\max}$  then
19:         return error
20:     end if
21: until  $s < \epsilon$ 
22: return  $x^* = x^B$ 

1:  procedure PRÓBUJ( $x, s$ )
2:      for  $j = 1$  to  $n$  do
3:          if  $f(x + s \cdot e^j) < f(x)$  then
4:               $x = x + s \cdot e^j$ 
5:          else
6:              if  $f(x - s \cdot e^j) < f(x)$  then
7:                   $x = x - s \cdot e^j$ 
8:              end if
9:          end if
10:      end for
11:      return  $x$ 
12: end procedure
```

### Pseudokod metody Rosenbrocka.

**Dane wejściowe:** punkt startowy  $x^{(0)}$ , wektor długość kroków  $s^{(0)}$ , współczynnik ekspansji  $\alpha > 1$ , współczynnik kontrakcji  $0 < \beta < 1$ , dokładność  $\varepsilon > 0$ , maksymalna liczba wywołań funkcji celu  $N_{\max}$

```
1:  i = 0
2:  dj(0) = ej, j = 1, 2, ..., n
3:  λj(0) = 0, j = 1, 2, ..., n
4:  pj(0) = 0, j = 1, 2, ..., n
5:  xB = x(0)
6:  repeat
7:      for j = 1 to n do
8:          if f(xB + sj(i) · dj(i)) < f(xB) then
9:              xB = xB + sj(i) · dj(i)
10:             λj(i+1) = λj(i) + sj(i)
11:             sj(i+1) = α · sj(i)
12:          else
13:              sj(i+1) = -β · sj(i)
14:              pj(i+1) = pj(i) + 1
15:          end if
16:      end for
17:      i = i + 1
18:      x(i) = xB
19:      if λj(i) ≠ 0 and pj(i) ≠ 0, j = 1, 2, ..., n then
20:          zmiana bazy kierunków dj(i)
21:          λj(i) = 0, j = 1, 2, ..., n
22:          pj(i) = 0, j = 1, 2, ..., n
23:          sj(i) = sj(0), j = 1, 2, ..., n
24:      end if
25:      if fcalls > Nmax then
26:          return error
27:      end if
28:  until maxj=1,...,n(|sj(i)|) < ε
29:  return x* = x(i)
```

$$d_j^{(i+1)} = \frac{v_j^{(i+1)}}{\|v_j^{(i+1)}\|_2}$$

gdzie:

$$v_1^{(i+1)} = Q_{*1}^{(i)}$$

$$v_j^{(i+1)} = Q_{*j}^{(i)} - \sum_{k=1}^{j-1} \left( (Q_{*j}^{(i)})^T d_k^{(i+1)} \right) d_k^{(i+1)}$$

$$Q^{(i)} = D^{(i)} \begin{bmatrix} \lambda_1^{(i+1)} & 0 & \dots & 0 \\ \lambda_2^{(i+1)} & \lambda_2^{(i+1)} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ \lambda_n^{(i+1)} & \lambda_n^{(i+1)} & \lambda_n^{(i+1)} & \lambda_n^{(i+1)} \end{bmatrix}$$

$$D^{(i)} = [d_1^{(i)} \quad d_2^{(i)} \quad \dots \quad d_n^{(i)}]$$