AKADEMIA GÓRNICZO-HUTNICZA
KATEDRA INFORMATYKI STOSOWANEJ I MODELOWANIA

METODY OPTYMALIZACJI

Optymalizacja funkcji wielu zmiennych metodami bezgradientowymi

1. Cel ćwiczenia.

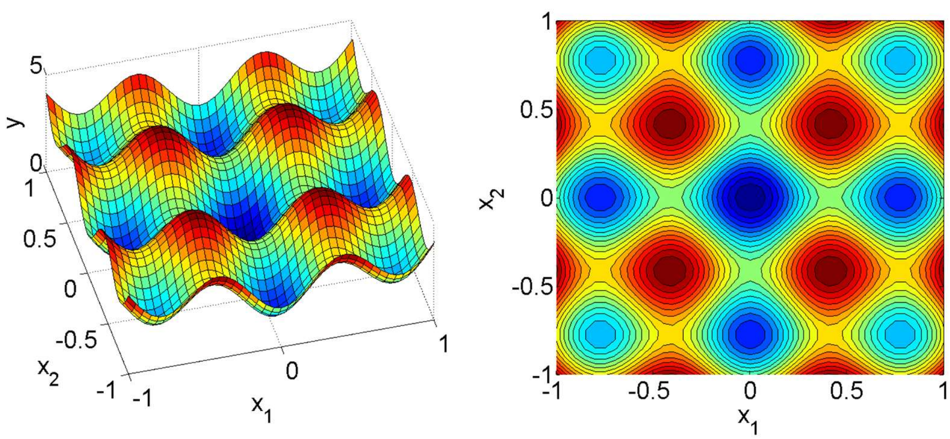
Celem ćwiczenia jest zapoznanie się z metodami bezgradientowymi poprzez ich implementację oraz wykorzystanie do rozwiązania problemu optymalizacji.

2. Testowa funkcja celu.

Funkcja celu dana jest wzorem:

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

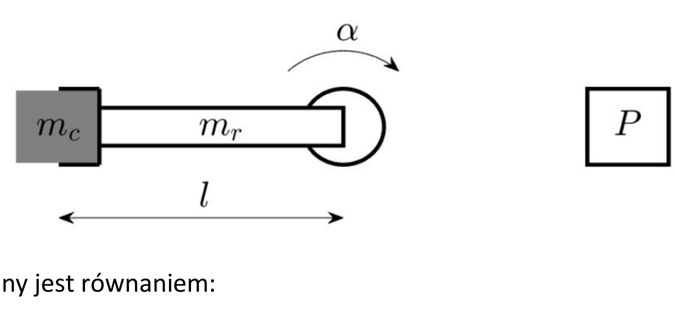
Jej wykres przedstawiony jest poniżej.



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 $l = 2\ m$ oraz masie $m_r = 1\ kg$ ma za zadanie umieścić ciężarek o masie $m_c = 5\ kg$ na platformie P. W tym celu ramie musi wykonać obrót o kąt $\pi\ rad$ i zatrzymać się.



Ruch ramienia opisany jest równaniem:

$I \frac{d^2\alpha}{dt^2} + b\frac{d\alpha}{dt} = M(t)$, gdzie: $b = 0.25\ Nms$ 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:

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

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

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

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

przyjmuje najmniejszą wartość. Początkowe wartości współczynników wzmocnienia powinny należeć do przedziału: $k_1^{(0)} \in [0,20]\ Nm$, $k_2^{(0)} \in [0,20]\ Nms$. Symulacje należy przeprowadzać dla czasu od $t_0 = 0$ do $t_{end} = 100s$ z krokiem $dt = 0.1s$.

W celu sprawdzenia poprawności implementacji modelu oraz obliczenia funkcji celu, można obliczyć wartość funkcji celu dla $k_1 = 5Nm$ oraz $k_2 = 5Nms$. Zakładając, że całka liczona jest metodą prostokątów, wartość funkcji celu powinna wynosić $Q(k_1, k_2) \approx 775.229$.

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 poziomic 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 $0 < \alpha < 1$, dokładność $\varepsilon > 0$, maksymalna liczba wywołań funkcji celu N_{max}

```
repeat
    x^B = x
    x = PRÓBUJ(x^B, s)
    if f(x) < f(x^B) then
        repeat
            x̲^B = x^B
            x^B = x
            x = 2x^B - x̲^B
            x = PRÓBUJ(x, s)
            if f_calls > N_max then
                return error
            end if
        until f(x) ≥ f(x^B)
        x = x^B
    else
        s = α·s
    end if
    if f_calls > N_max then
        return error
    end if
until s < ε
return x* = x^B
```

```
procedure PRÓBUJ(x, s)
    for j = 1 to n do
        if f(x + s·e^j) < f(x) then
            x = x + s·e^j
        else
            if f(x - s·e^j) < f(x) then
                x = x - s·e^j
            end if
        end if
    end for
    return x
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}

```
i = 0
d_j^(0) = e^j, j = 1, 2, …, n
λ_j^(0) = 0, j = 1, 2, …, n
p_j^(0) = 0, j = 1, 2, …, n
x^B = x^(0)
repeat
     for j = 1 to n do
          if f(x^B + s_j^(i)·d_j^(i)) < f(x^B) then
               x^B = x^B + s_j^(i)·d_j^(i)
               λ_j^(i+1) = λ_j^(i) + s_j^(i)
               s_j^(i+1) = α·s_j^(i)
          else
               s_j^(i+1) = -β ·s_j^(i)
               p_j^(i+1) = p_j^(i) + 1
          end if
     end for
     i = i + 1
     x^(i) = x^B
     if λ_j^(i) ≠ 0 and p_j^(i) ≠ 0, j = 1, 2, …, n then
          zmiana bazy kierunków d_j^(i)
          λ_j^(i) = 0, j = 1, 2, …, n
          p_j^(i) = 0, j = 1, 2, …, n
          s_j^(i) = s_j^(0), j = 1, 2, …, n
     end if
     if f_calls > N_max then
          return error
     end if
until max_{j=1,…,n}(|s_j^(i)|) < ε
return x* = x^(i)
```

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

gdzie:

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

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

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

$$D^{(i)} = \begin{bmatrix} d_1^{(i)} & d_2^{(i)} & \dots & d_n^{(i)} \end{bmatrix}$$