Nieliniowe zadanie optymalizacji bez ograniczeń – numeryczne metody iteracyjne optymalizacji

$$\min_{\mathbf{x}\in R^n} f(\mathbf{x}) = f\left(\hat{\mathbf{x}}\right)$$

Algorytmy poszukiwania minimum lokalnego zadania programowania nieliniowego:

- Bez ograniczeń
- Z ograniczeniami

Algorytmy zbieżne do minimum lokalnego x*, jeżeli taki punkt istnieje.

I. Techniki optymalizacji lokalnej

Ad.I Iteracyjne algorytmy optymalizacji

- **Algorytmy optymalizacji w kierunku**
- **Algorytmy optymalizacji bez ograniczeń**
- **Algorytmy optymalizacji z ograniczeniami**

Algorytmy zbieżne do minimum lokalnego x*, jeżeli taki punkt istnieje.

Algorytm optymalizacji lokalnej - przemierzanie obszaru rozwiązań dopuszczalnych w poszukiwaniu ekstremum funkcji celu według iteracyjnego schematu.

Schemat algorytmu optymalizacji lokalnej bez ograniczeń

(1) Wybierz punkt startowy **xo=xk**.
(2) Oblicz wartość funkcji ***f(x^k)*** oraz jeżeli jest to wymagane to jej gradient **∇ *f(x^k)***

(3) Zbadaj przyjęte kryterium zbieżności.
•Jeśli kryterium jest spełnione to koniec algorytmu – uzyskano rozwiązanie optymalne ***x^k*** i optymalną wartość funkcji celu ***f(x^k)***
•Jeżeli nie, to przejdź do (4)

(4) Wyznacz ustalony kierunek poszukiwań : d^k

(5) Wykonaj minimalizację kierunkową wybraną metodą:

$$x^{k+1} \in T(x^k, d^k)$$

(6) Podstaw $x^k \Leftarrow x^{k+1} \quad oraz \quad k \Leftarrow k+1$

i przejdź do (2)

Do minimalizacji w kierunku można użyć kilku algorytmów takich jak np.:

Algorytmy bez-gradientowe:
- złotego podziału,
- aproksymacji kwadratowej,

Algorytmy gradientowe:
- ekspansji i kontrakcji geometrycznej z testem jednoskośnym,
- logarytmiczny złoty podział odcinka ze wstępną ekspansją i kontrakcją geometryczną,
- aproksymacji parabolicznej z testem jednoskośnym,
- bisekcji z testem dwuskośnym Goldstein'a,

Iteracja metody poszukiwania minimum w kierunku

Przebieg typowej k-tej iteracji dowolnej metody realizującej ideę poszukiwania wzdłuż kierunku:

1. Określ kierunek poszukiwań d^k.

2. Znajdź α **minimalizujące** $\tilde{f}(\alpha) = f(x^k + \alpha d^k)$ **ze względu na** α.

3. Podstaw $x^{k+1} = x^k + \alpha\, d^k$.

Zbieżność ciągu punktów

Definicja. Mówimy, że ciąg punktów $\{x^k\}_{k=1}^{\infty}$ **jest zbieżny do punktu x jeżeli ciąg różnic k-tych przybliżeń i punktu optymalnego (punktu minimum)** $h^k = x^k - x$ **zbiega do zera, co w przestrzeni** R^n **oznacza, że**

$$\left\| h^k \right\| \to 0$$

Kryteria zbieżności:

1. Test teoretyczny

$$f^k - \hat{f} \le \varepsilon_1, \left\| x^k - \hat{x} \right\| \le \varepsilon_2$$

2. Przybliżona stacjonarność rozwiązania

$$\nabla f(x^k) = g^k \quad \Rightarrow \left\| g^k \right\| \le \varepsilon$$

3. Testy praktyczne:

$$\left\| x_i^k - x_i^{k+1} \right\| \le \varepsilon_i, \quad \forall i = 1, \dots, n$$

lub

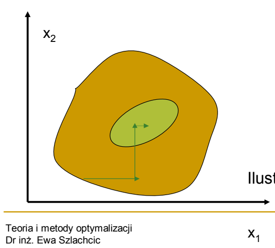
$$\left| f^k - f^{k+1} \right| \le \varepsilon_1$$

Algorytmy optymalizacji lokalnej

- **Algorytmy bezgradientowe**
  - **Algorytm Hooke'a-Jeeves'a**
  - **Algorytm Nelder'-Meade'a**
  - **Algorytm Gauss'a-Seidla**
  - **Algorytm Powella**
- **Algorytmy gradientowe**
  - **Algorytm największego spadku**
  - **Zmodyfikowany algorytm Newtona**
  - **Algorytm Zangwilla**
  - **Algorytm Fletchera-Reeves'a**
  - **Algorytm Polaka-Ribiery**
  - **Algorytm Fletchera-Powella-Davidona**

Metody podstawowe kierunków poprawy

1. Metoda Gaussa-Seidla (bezgradientowa).
2. Metoda największego spadku (gradientowa).
3. Metoda Newtona (gradient i hesjan).



Ilustracja metody Gaussa-Seidla

$$\mathbf{d}^{(k)} = \mathbf{e}_{i(k)}$$

Metoda Gaussa-Seidla (bardzo wolna zbieżność liniowa)

$$\mathbf{d}^{(k)} = -\nabla f\left(\mathbf{x}^{(k)}\right)$$

Metoda największego spadku (zbieżność liniowa)

$$\mathbf{d}^{(k)} = -\mathbf{H}\left(\mathbf{x}^{(k)}\right)^{-1} \nabla f\left(\mathbf{x}^{(k)}\right)$$

$$\mathbf{H} = \left\{h_{ij}\right\} = \left\{ \frac{\partial^2 f}{\partial x_i \partial x_j} \right\}_{i,j \in \{1,2,\dots,n\}}$$

Metoda Newtona (zbieżna kwadratowo ale kosztowna i nie zawsze stabilna)

Najefektywniejsze są tzw. metody quasi-newtonowskie, w których w kolejnych iteracjach konstruuje się przybliżenie odwrotności hesjanu.

Algorytm Gauss'a-Seidela

Istotą metody jest minimalizacja funkcji f(x) wzdłuż kolejnych kierunków ortogonalnej bazy , która utworzona jest z wersorów układu współrzędnych kartezjańskich.

Algorytm Gaussa-Seidela polega na cyklicznym stosowaniu odwzorowania T do kierunków . Wykonanie jednego takiego cyklu nazywa się k-tą iterację.

Odwzorowanie T:

$$T(x, d^i) = \left\{ x_i^{k+1} : f(x_i^{k+1}) = \min_{\tau \in \sigma} f(x_i^k + \tau_i d^i), \quad x_i^{k+1} = x_i^k + \tau_i d^i \right\}$$

Algorytm obliczeń – metoda Gauss'a-Seidla

(1) Wybierz punkt startowy **x^o=x^k**. Oblicz wartość funkcji ***f(x^k)***

(2) Zbadaj kryterium zbieżności:

$$\left\| x_i^k - x_i^{k+1} \right\| \le \varepsilon_i, \quad \forall i = 1, \dots, n \qquad \text{oraz} \qquad \left| f^k - f^{k+1} \right| \le \varepsilon_1$$

$$gdzie \quad \varepsilon \in [0, \delta] \quad np.: \varepsilon = 10^{-6}$$

Jeśli tak, to koniec, jeśli nie, to przejdź do (3)

(3) Wyznacz kierunek poszukiwań : są to kolejne kierunki ortogonalnej bazy

$d^k = e^k$ Np. $e^1 = [1,0,...,0]$

(4) Wykonaj minimalizację kierunkową wybraną metodą:

$$x^{k+1} \in T(x^k, d^k)$$

(5) Podstaw $x^k \Leftarrow x^{k+1}$ $oraz$ $k \Leftarrow k+1$ i $powtórz$ (1)

Metoda największego spadku NS

jest to metoda gradientowa, która pozwala szukać minimum różniczkowalnej funkcji nieliniowej f(x).

Koncepcja metody wynika z lematu, w którym wykazano, że jeśli istnieje kierunek d w przestrzeni R^n taki, że

$$\langle \nabla f(x), d \rangle < 0$$

to

$$f(x + \tau d) < f(x)$$

Algorytm obliczeń – metoda NS

(1) Wybierz punkt startowy **x^o=x^k**. Oblicz wartość funkcji ***f(x^k)*** oraz jej gradient **∇ *f(x^k)***

(2) Zbadaj kryterium zbieżności:

$$\langle \nabla f(x^k), \nabla f(x^k) \rangle = 0 \quad czyli \quad \langle \nabla f(x^k), \nabla f(x^k) \rangle \le \varepsilon$$

$$gdzie \quad \varepsilon \in [0, \delta] \quad np.: \varepsilon = 10^{-6}$$

Jeśli tak, to koniec, jeśli nie, to przejdź do (3)

(3) Wyznacz kierunek poszukiwań :

$$d^k = -\nabla f(x^k)$$

(4) Wykonaj minimalizację kierunkową wybraną metodą:

$$x^{k+1} \in T(x^k, d^k)$$

(5) Podstaw $x^k \Leftarrow x^{k+1}$ $oraz$ $k \Leftarrow k+1$ i $powtórz$ (1)

Do minimalizacji w kierunku zastosowano gradientowy algorytm bisekcji z testem dwuskośnym Goldstein'a :

Algorytm bisekcji z testem dwuskośnym Golstein'a – algorytm gradientowy

Praktycznie do wyszukania punktów spełniających test dwuskośny Goldsteina stosuje się następujący algorytm bisekcji:

(1) Oblicz pochodną w kierunku $p = \nabla f(x^o)^T d$ oraz współczynnik kroku $\tau_R > 0$ $taki$, $że$ $f(x^0 + \tau_R d) < f(x^0)$

(2) Wyznacz $\tau = \frac{1}{2}(\tau_L + \tau_R)$. $Oblicz$ $f(x^0 + \tau d)$.

(3) Jeśli $f(x^0 + \tau d) < f(x^0) + (1-\beta) p \tau$ to podstaw $\tau_L \Rightarrow \tau$ i przejdź do kroku (2), w przeciwnym razie przejdź do kroku (4)

(4) Jeśli $f(x^0 + \tau d) > f(x^0) + \beta p \tau$ to podstaw $\tau_R \Rightarrow \tau$ i przejdź do kroku (2), w przeciwnym przypadku koniec.

Działanie algorytmu bisekcji z testem dwuskośnym Goldstein'a dla funkcji:

$$f(x_1, x_2) = (x_1)^2 + 2(x_2)^2 - 6x_1 + x_1x_2$$

- punkt początkowy $x^0 = [0, 0]^T$
- kierunek $d = [1, 0]^T$
- współczynnik testu $\beta = \frac{2}{5}$
- początkowa wartość współczynnika kroku $\tau_R = 9$
- dokładność dla testu $\varepsilon' = 10^{-5}$

Pochodna w kierunku $p = \nabla f(x^o)^T d$
zatem mamy:

$$\nabla f(x) = \left[\frac{\partial f(x)}{\partial x_1}, \frac{\partial f(x)}{\partial x_2}\right] = [2x_1 - 6 + x_2, 4x_2 + x_1]$$

dla $x = x^0 = [0, 0]^T$

$$\nabla f(x^0) = [-6, 0]$$

Otrzymujemy wartość pochodnej p:

$$p = \nabla f(x^o)^T d = [-6 \quad 0] \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix} = -6$$

(2) Obliczamy $\tau = \frac{1}{2}(\tau_L + \tau_R)$ oraz $f(x^0 + \tau d)$.

$$\tau = \frac{1}{2}(\tau_R) = \frac{1}{2}(9) = 4{,}5$$

$$f(x^0 + \tau d) = f\left[(0{,}0) + (4{,}5 \quad ,0)\right] = 20{,}25 - 27 = -6{,}75$$

Przechodzimy do kroku (3)

(3) Jeżeli $f(x^0 + \tau d) < f(x^0) + (1 - \beta) p \tau$
to podstaw $\tau_L \Rightarrow \tau$
i przejdź do kroku (2). W przeciwnym wypadku przejdź do kroku (4)

sprawdzamy: -6,75 <? $0 + (-6) \cdot (0{,}6) \cdot (4{,}5) = -16{,}2$

NIE

Przechodzimy do kroku (4)

(4) Jeżeli $f(x^\circ + \tau d) > f(x^\circ) + \beta p \tau$
to podstaw $\tau_R \Rightarrow \tau$
i przejdź do kroku (2). W przeciwnym wypadku KONIEC

sprawdzamy: -6,75 >? $0 + (-6) \cdot (0{,}4) \cdot (4{,}5) = -10{,}8$

TAK

i przechodzimy do kroku (2)

DRUGA ITERACJA

(...)

Po trzeciej iteracji otrzymujemy wynik

$$\tau = 3{,}375$$

Działanie algorytmu najszybszego spadku dla funkcji:
$f(x_1, x_2) = 2(x_1)^2 + (x_2)^2 - 2\, x_1x_2$

- punkt początkowy $x^0 = [2, 3]^T$
- współczynnik testu $\beta = \frac{1}{4}$
- początkowa wartość współczynnika kroku $\tau_R = 1$

- Obliczamy $d^0 = -\nabla f(x^0) = [-2, -2]^T$
- Ponieważ pierwsza stosowana wartość współczynnika kroku $\tau_R = 1$ spełnia test dwuskośny Goldsteina, więc:
- $x^1 = x^0 + \tau_0 d^0 = [0\ 1]^T$ $\quad d^1 = -\nabla f(x^1) = [2\ \ -2]^T$
- W drugiej iteracji mamy:

$$f(x^1 + \tau d^1) = 20\tau^2 - 8\tau + 1$$

- Otrzymujemy:

$$p = \nabla f(x^1)^T d^1 = \begin{bmatrix} -2 \\ 2 \end{bmatrix} \cdot [2\ \ -2] = -8$$

- Zatem test dwuskośny ma postać
- $-6 \le 20\tau^2 - 8\tau \le -2$
- Za pomocą algorytmu bisekcji (test dwuskośny Goldsteina) w trzeciej próbie znajdujemy wartość współczynnika $\tau_1 = 0{,}25$

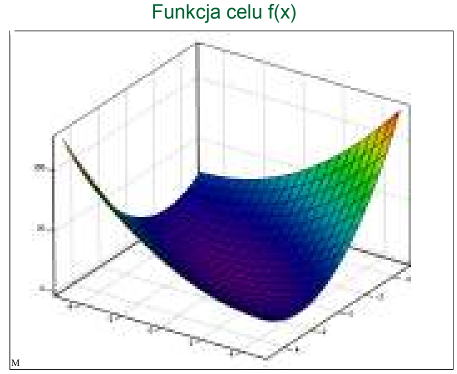
Stąd
- $x^2 = x^1 + \tau_1 d^1 = [\frac{1}{2}\ \ \frac{1}{2}]^T$
- Postępując zgodnie z algorytmem otrzymujemy kolejne wartości punktów optymalizowanej funkcji.

- Kolejno podane są punkty wyznaczone za pomocą algorytmu najszybszego spadku dla funkcji:
- $f(x_1, x_2) = 2(x_1)^2 + (x_2)^2 - 2\, x_1 x_2$
- $x^0 = [2\ \ 3]$
- $x^1 = [0\ \ 1]$
- $x^2 = [\frac{1}{2}\ \ \frac{1}{2}]$
- $x^3 = [\frac{1}{4}\ \ \frac{1}{2}]$
- $x^4 = [\frac{1}{4}\ \ \frac{1}{4}]$ itd....
- I tak kolejno, aż do momentu gdy zostanie spełniony warunek

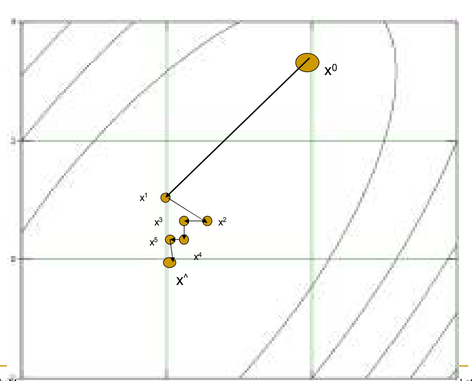
$$\left\langle \nabla f(\hat{x}), \nabla f(\hat{x}) \right\rangle < \varepsilon = 10^{-3}$$

Tak uzyskano rozwiązanie optymalne *x^k=[0,0]* i *$f(x^k)$=0.*

Funkcja celu f(x)



Kolejne iteracje metody największego spadku NS



Algorytmy optymalizacji z ograniczeniami

W celu uwzględnienia ograniczeń można postąpić w poniższy sposób:
- dokonać transformacji zmiennych decyzyjnych
- dokonać transformacji funkcji celu wprowadzając funkcje kary.

Przykłady transformacji zmiennych dla typowych ograniczeń:

1. $x_i \ge 0$ $\qquad x_i = u_i^2$, $\quad x_i = \exp(u_i)$, $\quad x_i = |u_i|$
2. $0 \le x_i \le 1$ $\qquad x_i = \sin^2 u_i$, $\quad x_i = \dfrac{\exp(u_i)}{\exp(u_i) + \exp(-u_i)}$
3. $a_i \le x_i \le b_i$ $\qquad x_i = a_i + (b_i - a_i)\sin^2(u_i)$

Algorytmy optymalizacji z ograniczeniami cd.

Transformacja funkcji kryterialnej:

$$P(x,\sigma,\theta) = f(x) + \sum_{i=1}^{m} \sigma_i \varphi(g_i(x)+\theta_i)\, H(g_i(x)+\theta_i)$$

Funkcja kary charakteryzuje się tym, że w zbiorze rozwiązań dopuszczalnych ***X*** przyjmuje wartość równą zeru lub bliską zeru, a poza tym obszarem przyjmuje bardzo duże wartości.

Gdzie: $\sigma_i > 0,\ \sigma = [\sigma_1, \sigma_2, \ldots, \sigma_m]$ jest wektorem współczynników kary

$\theta_i > 0,\ \theta = [\theta_1, \theta_2, \ldots, \theta_m]$ jest wektorem przesunięć kary

φ( ) funkcja kary

$$\varphi(g_i(x)+\theta_i):\quad np.\,(g_i(x)+\theta_i)^2 \quad \text{lub}\ (-g_i^{-1}(x)+\theta_i)$$

Algorytmy optymalizacji z ograniczeniami cd.

Funkcja H ma poniższą własność:

$$H(g_i(x)+\theta_i) = \begin{cases} 1 & dla\ g_i(x)+\theta_i > 0 \\ 0 & dla\ g_i(x)+\theta_i \le 0 \end{cases}$$

1. **Metody zewnętrznej funkcji kary (metoda Couranta, metoda Schmita i Foxa)**
2. **Metody wewnętrznej funkcji kary (metoda Rosenbrocka, metoda Carolla)**
3. **Metody przesuwanej funkcji kary (metoda Powella).**