



# Algoritmi numerici pentru optimizare. Algoritmi determiniști de ordin zero.

Prof.dr.ing. Gabriela Ciuprina

Universitatea "Politehnica" București, Facultatea de Inginerie Electrică,  
Departamentul de Electrotehnică

Suport didactic pentru disciplina *Algoritmi Numerici*, 2015

# Cuprins

## Introducere

### Optimizare unidimensională

- Metoda căutării simultane
- Metoda căutării dihotomice
- Metoda Fibonacci
- Metoda secțiunii de aur

### Optimizare multidimensională

- Metoda simplexului descendent (Nelder-Mead)
- Metoda Powell



## Formularea problemei

Să se găsească  $n$  parametri independenți, notați  $x_1^*, x_2^*, \dots, x_n^*$ , pentru care expresia  $E$  este minimă, unde

$$E = f(x_1, x_2, \dots, x_n), \quad (1)$$

și  $f : \Omega \rightarrow \mathbb{R}$ ,  $\Omega \subset \mathbb{R}^n$ , este dată.

Pe scurt:

$$(x_1^*, x_2^*, \dots, x_n^*) = \arg \min f(x_1, x_2, \dots, x_n). \quad (2)$$

Notății

$$\mathbf{x} = [x_1, x_2, \dots, x_n]^T \in \Omega. \quad (3)$$

$$\mathbf{x}_{\min} = [x_1^*, x_2^*, \dots, x_n^*]^T \in \Omega \quad (4)$$

$$\mathbf{x}_{\min} = \arg \min f(\mathbf{x}), \quad \mathbf{x} \in \Omega$$

$$E_{\min} = f(\mathbf{x}_{\min}).$$



## Formularea problemei

$$\mathbf{x}_{\min} = \arg \min f(\mathbf{x}), \quad \mathbf{x} \in \Omega \quad (5)$$

$$E_{\min} = f(\mathbf{x}_{\min}). \quad (6)$$

Observații:

1. Min / Max - limitare ?

$$\max \{ f(\mathbf{x}) \mid \mathbf{x} \in \Omega \} = - \min \{ -f(\mathbf{x}) \mid \mathbf{x} \in \Omega \}$$

2. Optimizarea "scalară" - un singur număr înglobează criterii

- de proiectare ( $\|$  performanța cerută – cea obținută  $\|$ );
- de economie (prețul).

$\Rightarrow f$  este numită *funcție obiectiv*, *funcție de cost*, *funcție de merit*, *criteriu de performanță*.



## Formularea problemei

### Minime globale/locale

$$\mathbf{x}_{\min} = \arg \min f(\mathbf{x}), \quad \mathbf{x} \in \Omega \quad (7)$$

$$E_{\min} = \min \{ f(\mathbf{x}) \mid \mathbf{x} \in \Omega \} \quad (8)$$

$\mathbf{x}_{\min}$  este *minim global* dacă

$$E_{\min} \leq f(\mathbf{x}), \quad \forall \mathbf{x} \in \Omega \quad (9)$$

- dacă  $E_{\min} \leq f(\mathbf{x})$  doar într-o vecinătate a lui  $\mathbf{x}_{\min}$  atunci minimul este *local*.
- în practică este dificil de stabilit dacă un minim găsit este local sau global;
- minimul global s-ar putea să nu fie unic.



## Metode de optimizare

**I. Deterministe** - conduc la aceeași soluție pentru rulări diferite ale programului, dacă pornesc din aceeași condiții inițiale și au aceeași parametri.

- Dezavantaj: găsesc întotdeauna un minim local, dependent de inițializare;
- Avantaj: efort de calcul mic.

În problemele de optimizare din efortul de calcul se exprimă în număr de evaluări de funcții obiectiv.

Pot fi

1. **de ordin zero** - necesită doar evaluări de funcții obiectiv;

Ex: metoda căutării simultane; metoda căutării dihotomice; metoda Fibonacci; metoda secțiunii de aur; metoda simplexului descendent (Nelder-Mead); metoda Powell, etc.

2. **de ordin superior** (1,2) - necesită și evaluări ale derivatelor funcției obiectiv.

## II. Stocastiche



## Metode de optimizare - 1D

$$(x_1^*, x_2^*, \dots, x_n^*) = \arg \min f(x_1, x_2, \dots, x_n). \quad (10)$$

"Optimizare 1D":  $n = 1$

→ În contextul optimizării 1D:  $f$  se numește **unimodală** atunci când are un singur minim în domeniul ei de definiție.

"Optimizare nD":  $n > 1$

Metode

- **de căutare** = intervalul care conține minimul este micșorat prin evaluarea lui  $f$  în anumite puncte;
- **de aproximare** = funcția de optimizat este aproximată printr-o funcție cunoscută care poate fi analizată ușor.



## Metoda căutării simultane

### Ideea:

- se împarte  $[a, b]$  în  $n$  subintervale  $x_i = a + ih, i = 0, \dots, n$ ,  
 $h = (b - a)/n$ .
- se selectează valoarea minimă dintre  $f(x_i)$ .

Obs:  $h$  - suficient de mic

```
funcție met_căutării_simultane(real a, b, funcție f, întreg n)
h = (b - a)/n
minim = f(b)
pentru i = 0 : n - 1
    x = a + ih
    val = f(x)
    dacă (val < minim)
        minim = val;
întoarce minim
```

**Complexitate**<sup>1</sup>  $T = O(n)$

---

<sup>1</sup>Operația de referință este evaluarea funcției  $f$ .





## Metoda căutării simultane

**Acuratețe** (pp  $f$  unimodală)

Dacă  $x_k$  este punctul de minim obținut  $\Rightarrow x_{\min} \in [x_{k-1}, x_{k+1}]$ .

Obs:

- $[x_{k-1}, x_{k+1}]$  - *interval de incertitudine*  
 $x_{\min} \in [x_k - h, x_k + h] \Leftrightarrow "x_{\min} = x_k \pm h"$ .

- Eroarea absolută

$$|x_{\min} - x_k| \leq h$$

- Eroarea relativă

$$\left| \frac{x_{\min} - x_k}{x_{\min}} \right| \leq \frac{h}{\min(|x_{k-1}|, |x_{k+1}|)}$$

$\rightarrow$  dificultate dacă  $x_{k-1} = 0$  sau  $x_{k+1} = 0$



## Metoda căutării simultane

**Acuratețe** (pp  $f$  unimodală)

Se preferă

**Estimator al erorii relative** = raportul dintre intervalul de incertitudine final și cel inițial

$$\text{er}_{\text{rel}} = \frac{2h}{b-a} = \frac{2}{n}$$

Pentru ca  $\text{er}_{\text{rel}} \leq \varepsilon \Rightarrow \frac{2}{n} \leq \varepsilon \Rightarrow n = \left\lceil \frac{2}{\varepsilon} \right\rceil + 1$  evaluări.

Dacă  $\varepsilon = 10^{-m} \Rightarrow n = 2 \cdot 10^m + 1$  (ex:  $\varepsilon = 1\% \Rightarrow n = 201$ ).

## Generalizare

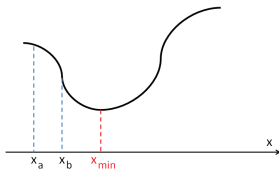
Ideea metodei s-ar putea generaliza în spațiul  $n$ -dimensional, dar efortul de calcul ar crește enorm de mult.

## Metoda căutării dihotomice

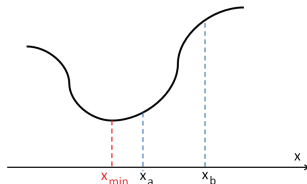
Pp.  $f$  unimodală.

### Ideea căutării dihotomice<sup>2</sup>

- dacă  $x_a$  și  $x_b$  sunt două puncte situate de aceeași parte a punctului de minim  $x_{\min}$  atunci cel care se găsește mai aproape de  $x_{\min}$  furnizează o aproximație mai bună.



$$x_a < x_b < x_{\min} \Rightarrow \\ f(x_b) < f(x_a)$$



$$x_{\min} < x_a < x_b \Rightarrow \\ f(x_a) < f(x_b)$$

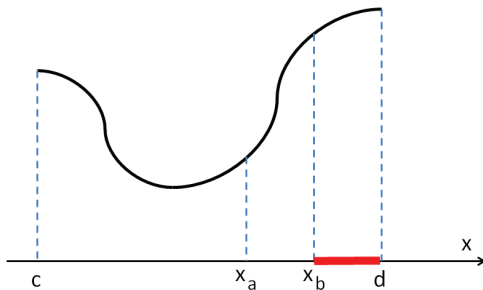
<sup>2</sup>Dihotomie = diviziune în două părți; bifurcare; împărțirea unei noțiuni în alte două noțiuni care epuizează întreaga sferă a noțiunii împărțite.



## Metoda căutării dihotomice

### Ideea algoritmului

Fie  $I = (c, d)$  intervalul de incertitudine și  $x_a < x_b$  două puncte în acest interval.



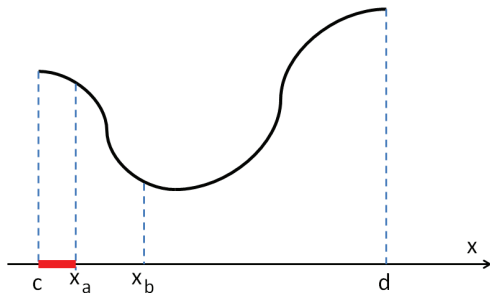
dacă  $f(x_a) < f(x_b)$   
atunci  $I_{\text{nou}} = (c, x_b)$

 se poate elimina

## Metoda căutării dihotomice

### Ideea algoritmului

Fie  $I = (c, d)$  intervalul de incertitudine și  $x_a < x_b$  două puncte în acest interval.



dacă  $f(x_a) > f(x_b)$   
atunci  $I_{\text{nou}} = (x_a, d)$

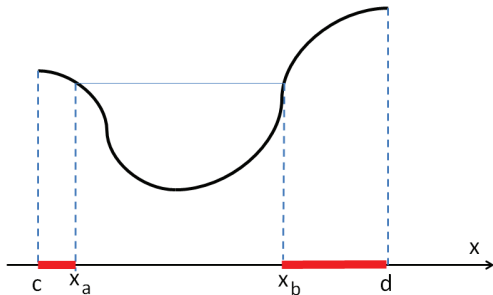
■ se poate elimina



## Metoda căutării dihotomice

### Ideea algoritmului

Fie  $I = (c, d)$  intervalul de incertitudine și  $x_a < x_b$  două puncte în acest interval.



dacă  $f(x_a) = f(x_b)$   
atunci  $I_{\text{nou}} = (x_a, x_b)$

 se pot elimina

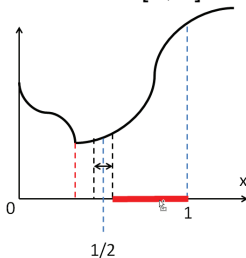


## Metoda căutării dihotomice

Ipoteza de unimodalitate permite micșorarea succesivă a intervalului de incertitudine.

**Algorithm:** se înjumătățește intervalul de incertitudine plasând perechi de puncte test foarte aproape una de cealaltă ( $\Delta x$  - distanța dintre ele) în interiorul fiecărui interval.

Exemplu - dacă  $I = [0, 1]$



După primul pas

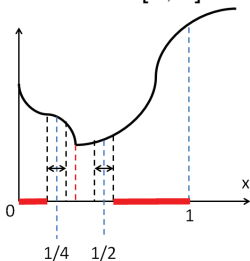
$$I_{\text{nou}} = \left[0, \frac{1}{2} + \frac{\Delta x}{2}\right)$$



Ipoteza de unimodalitate permite micșorarea succesivă a intervalului de incertitudine.

**Algoritm:** se înjumătățește intervalul de incertitudine plasând perechi de puncte test foarte aproape una de cealaltă ( $\Delta x$  - distanța dintre ele) în interiorul fiecărui interval.

Exemplu - dacă  $I = [0, 1]$



## După al doilea pas

$$I_{\text{nou}} = [\frac{1}{4} - \frac{\Delta x}{2}, \frac{1}{2} + \frac{\Delta x}{2})$$





## Metoda căutării dihotomice

### Acuratețe

- $m$  pași:  $2m$  evaluări și un interval de incertitudine de  $l/2^m$ .
- După  $n$  evaluări intervalul de incertitudine este  $l/2^{n/2}$ .
- Eroarea relativă

$$er_{rel} = \frac{1/2^{n/2}}{1} = \frac{1}{2^{n/2}}$$

Pentru ca  $er_{rel} \leq \varepsilon \Rightarrow \frac{1}{2^{n/2}} \leq \varepsilon \Rightarrow n = 2 \log_2 \left( \frac{1}{\varepsilon} \right) + 1$  evaluări.

Dacă  $\varepsilon = 1\% \Rightarrow n = 14$ , mai eficient decât la căutarea simultană.

### Dezavantaje

- $\Delta x$  nu poate fi mai mic decât zeroul mașinii
- Este ineficientă evaluarea funcției pentru valori atât de apropiate. Erorile de rotunjire ar putea duce la decizii greșite.
- Metoda nu se poate generaliza în cazul n-dimensional.



## Metoda Fibonacci

Ideea: eliminarea intervalelor similar ca la metoda căutării dihotomice, dar punctele intermediare  $x_a$  și  $x_b$  nu sunt apropiate.

Reamintire: șirul lui Fibonacci

$$a_0 = a_1 = 1$$

$$a_n = a_{n-1} + a_{n-2}, \quad n = 2, 3, \dots \quad (11)$$

1, 1, 2, 3, 5, 8, 13, 21, ...

Obs:

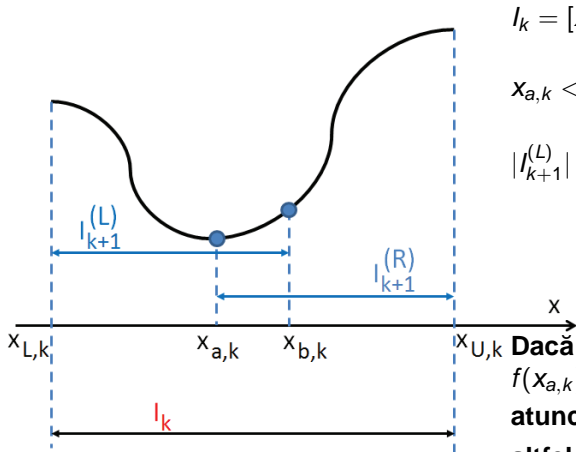
$$\lim_{n \rightarrow \infty} \frac{a_{n-1}}{a_n} = \frac{-1 + \sqrt{5}}{2} \approx 0.615 \quad (12)$$

Dem:

$$a_n/a_{n-1} = 1 + a_{n-2}/a_{n-1} \Rightarrow 1/x = 1 + x \Rightarrow x^2 + x - 1 = 0, \text{ etc.}$$



## Metoda Fibonacci



$$I_k = [x_{L,k}, x_{U,k}]$$

$$x_{a,k} < x_{b,k}$$

$$|I_{k+1}^{(L)}| = |I_{k+1}^{(R)}|$$

**Dacă**

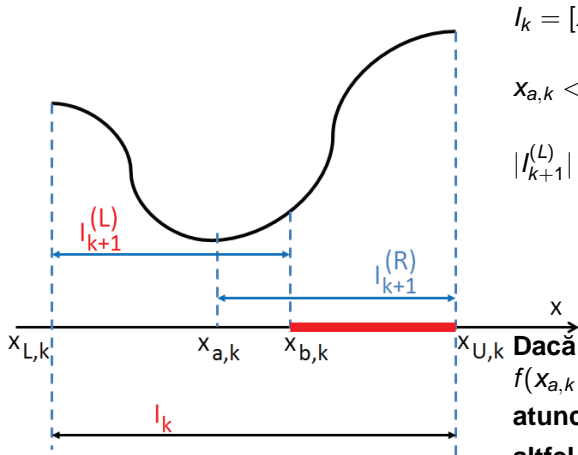
$$f(x_{a,k}) < f(x_{b,k})$$

**atunci**  $I_{k+1} = I_{k+1}^{(L)}$

**altfel**  $I_{k+1} = I_{k+1}^{(R)}$



## Metoda Fibonacci



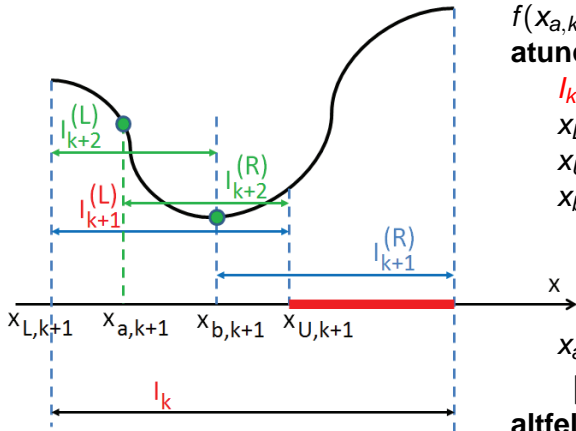
$$I_k = [x_{L,k}, x_{U,k}]$$

$$x_{a,k} < x_{b,k}$$

$$|I_{k+1}^{(L)}| = |I_{k+1}^{(R)}|$$

**Dacă**  
 $f(x_{a,k}) < f(x_{b,k})$   
**atunci**  $I_{k+1} = I_{k+1}^{(L)}$   
**altfel**  $I_{k+1} = I_{k+1}^{(R)}$

## Metoda Fibonacci



**Dacă**

$$f(x_{a,k}) < f(x_{b,k})$$

**atunci**

$$I_{k+1} = I_{k+1}^{(L)}$$

$$x_{L,k+1} = x_{L,k}$$

$$x_{U,k+1} = x_{b,k}$$

$$x_{b,k+1} = x_{a,k}$$

$x_{a,k+1}$  ales a.î

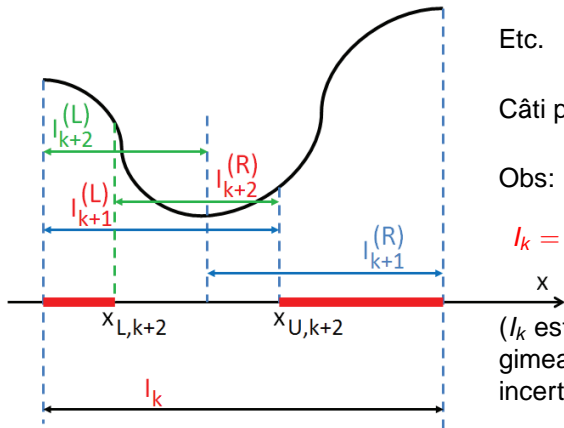
$$|I_{k+2}^{(L)}| = |I_{k+2}^{(R)}|$$

**altfel**

...



## Metoda Fibonacci



Etc.

Câți pași ?

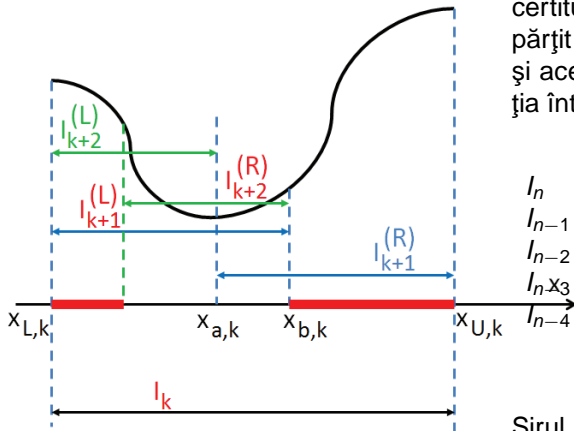
Obs:

$$I_k = I_{k+1} + I_{k+2} \quad (13)$$

( $I_k$  este de acum lungimea intervalului de incertitudine).



## Metoda Fibonacci



Ultimul interval de incertitudine va fi împărțit în 2 și nu în 3 și aceasta va fi soluția întoarsă.

$$\begin{aligned}
 I_n &= I_{n+1} = 1I_n \\
 I_{n-1} &= I_n + I_{n+1} = 2I_n \\
 I_{n-2} &= I_{n-1} + I_n = 3I_n \\
 I_{n-3} &= I_{n-2} + I_{n-1} = 5I_n \\
 I_{n-4} &= I_{n-3} + I_{n-2} = 8I_n \\
 &\vdots
 \end{aligned}
 \tag{14}$$

Șirul Fibonacci:  
1, 1, 2, 3, 5, 8, ...



## Metoda Fibonacci

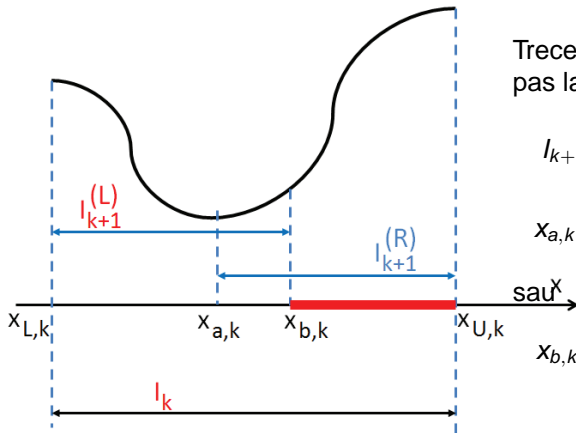
$$\begin{aligned}
 l_{n-1} &= a_2 l_n \\
 l_{n-2} &= a_3 l_n \\
 l_{n-3} &= a_4 l_n \\
 l_{n-4} &= a_5 l_n \\
 &\vdots \\
 l_k &= a_{n-k+1} l_n \\
 &\vdots \\
 l_1 &= a_n l_n
 \end{aligned} \tag{15}$$

*n* se stabilește de la început!





## Metoda Fibonacci



Trecerea de la un pas la altul:

$$l_{k+1} = \frac{a_{n-k}}{a_{n-k+1}} l_k \quad (16)$$

$$x_{a,k} = x_{U,k} - l_{k+1} \quad (17)$$

$$x_{b,k} = x_{L,k} + l_{k+1} \quad (18)$$



# Metoda Fibonacci

**funcție** [real  $x\_min$ , tol\_x,  $f\_min$ ] = met\_fibonacci(real  $la$ ,  $lb$ , funcție  $f$ , întreg  $n$ )

$a$  = numere\_fibonacci( $n$ ) ; calculează numerele Fibonacci între 1 și  $n$

$xL_1 = la$  ; capătul din stânga al intervalului inițial

$xU_1 = lb$  ; capătul din dreapta al intervalului inițial

$l_1 = lb - la$  ; lungimea intervalului inițial

$l_2 = (a_{n-1} / a_n) l_1$

$xa_1 = xU_1 - l_2$

$xb_1 = xL_1 + l_2$

$fa = f(xa_1)$

$fb = f(xb_1)$

**pentru**  $k = 2, n - 1$

$l_{k+1} = (a_{n-k} / a_{n-k+1}) l_k$

**dacă** ( $fa \leq fb$ ) **atunci**

$xL_k = xL_{k-1}$

$xU_k = xb_{k-1}$

$xa_k = xU_k - l_{k+1}$

$xb_k = xa_{k-1}$

$fb = fa$

$fa = f(xa_k)$

**altfel**

$xL_k = xa_{k-1}$

$xU_k = xU_{k-1}$

$xb_k = xL_k + l_{k+1}$

$xa_k = xb_{k-1}$

$fa = fb$

$fb = f(xb_k)$

$x\_min = (xL_k + xU_k) / 2$

tol\_x =  $l_{n-1}$

$f\_min = f(x\_min)$

**retur**



## Metoda Fibonacci

### Acuratețe și efort de calcul

#### Eroarea relativă

$$\frac{l_n}{l_1} = \frac{1}{a_n} \quad (19)$$

se obține cu un efort de  $n + 1$  evaluări de funcții.

$a_n$ : 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, ...

$\frac{l_{11}}{l_1} = \frac{1}{144}$  după 12 evaluări

Căutare simultană: 21 de evaluări pentru a scădea intervalul de incertitudine de 10 ori.

Fibonacci:  $\frac{l_6}{l_1} = \frac{1}{13} \Rightarrow 7$  evaluări.



## Metoda Fibonacci

Numărul de evaluări de funcții trebuie impus ca parametru de intrare.

- Dacă se cere ca valoarea funcției să scadă de un anumit număr de ori față de valoarea inițială, atunci numărul de evaluări nu poate fi estimat.
- Dacă se cere ca intervalul de incertitudine să scadă de un anumit număr de ori față de valoarea inițială, atunci se poate determina cea mai mică valoare  $N$  pentru care  $1/a_N < \varepsilon$ , și folosit acest  $N$  ca parametru pentru procedura Fibonacci.
- Se poate demonstra că dintre metodele care cer un număr fixat de evaluări de funcții, metoda Fibonacci realizează cea mai mare micșorare a intervalului de incertitudine.

Metoda nu se poate generaliza în cazul multidimensional



## Metoda secțiunii de aur

Fibonacci: primele două puncte  $x_a$  și  $x_b$  depind de numărul de evaluări de funcții. Odată specificat acest număr și pusă condiția ca la sfârșit  $x_a = x_b$  rezultă lungimea intervalului.

Metoda secțiunii de aur - presupune tot că

$$l_k = l_{k+1} + l_{k+2}, \quad (20)$$

dar, impune ca

$$\frac{l_k}{l_{k+1}} = \frac{l_{k+1}}{l_{k+2}} = \varphi \quad (21)$$



## Metoda secțiunii de aur

$$l_k = l_{k+1} + l_{k+2}$$

$$\frac{l_k}{l_{k+2}} = \frac{l_{k+1}}{l_{k+2}} + 1,$$

$$\varphi^2 = \varphi + 1,$$

$$\varphi = \frac{1 + \sqrt{5}}{2} \approx 1.618$$

Secțiunea de aur = împărțirea unui interval în două subintervale a.î. raportul dintre intervalul întreg și subintervalul mai mare este egal cu raportul dintre subintervalul mai mare și cel mai mic.

Vedeți și [https://en.wikipedia.org/wiki/Golden\\_ratio](https://en.wikipedia.org/wiki/Golden_ratio)



## Metoda secțiunii de aur

Algoritm - seamană cu cel al metodei Fibonacci, se fac următoarele modificări:

- $a_{n-1}/a_n$  se înlocuiește cu  $\varphi$
- $a_{n-k}/a_{n-k+1}$  se înlocuiește cu  $\varphi$
- în loc de ciclu cu contor se folosește ciclu cu test
- criteriul de oprire este mai natural - bazat pe reducerea lungimii intervalului de incertitudine de un număr impus de ori.

Temă - scrieți pseudocodul algoritmului, implementați-l, testați-l și comparați rezultatele cu cele obținute cu metoda Fibonacci.



## Metoda secțiunii de aur

### Comparație cu metoda Fibonacci

Se poate arăta că

$$a_n \approx \frac{\varphi^{n+1}}{\sqrt{5}} \quad (22)$$

Fibonacci:

$$\varepsilon_F = \frac{l_n}{l_1} = \frac{1}{a_n} = \frac{\sqrt{5}}{\varphi^{n+1}} \quad (23)$$

Secțiunea de aur (pentru același număr de evaluări):

$$\varepsilon_g = \frac{l_n}{l_1} = \frac{1}{\varphi^{n-1}} \quad (24)$$





## Metoda secțiunii de aur

### Comparație cu metoda Fibonacci

Rezultă că

$$\frac{\varepsilon_g}{\varepsilon_F} = \frac{\varphi^2}{\sqrt{5}} \approx 1.17 \quad (25)$$

Pentru același număr de evaluări de funcții, reducerea intervalului la Fibonacci este mai mic cu 17% decât la secțiunea de aur.

Acest avantaj se obține însă pe baza specificării în avans a numărului de evaluări de funcții.



## Metoda simplexului descendent (Nelder-Mead)

### Formularea problemei:

$$(\mathbf{x}_1^*, \mathbf{x}_2^*, \dots, \mathbf{x}_n^*) = \arg \min f(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n). \quad (26)$$

 $n > 1$ 

Nelder și Mead (1965) - metoda simplexului descendent:

- E simplă conceptual și are o natură geometrică;
- Nu are nevoie de un algoritm de minimizare 1D;
- Nu este foarte eficientă dpdv al efortului de calcul;
- E frecvent utilizată dacă efortul necesar evaluării funcției este mic;
- Nu trebuie confundată cu metoda simplex a programării liniare;



## Metoda simplexului descendent (Nelder-Mead)

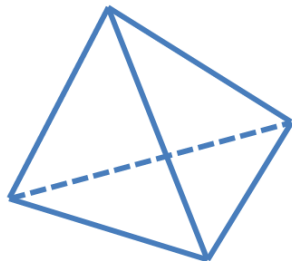
**Simplex** = poliedru cu  $n + 1$  vârfuri



$n = 1$



$n = 2$



$n = 3$

- nu este neaparat regulat;
- ne interesează cele nedegenerate (cu măsura nenulă).

## Metoda simplexului descendent (Nelder-Mead)

**Idea:** simplexul se "mișcă" în spațiul  $n$ -dimensional, până când se întâlnește un minim.

**Inițializarea:** este nevoie de  $n + 1$  puncte de start.

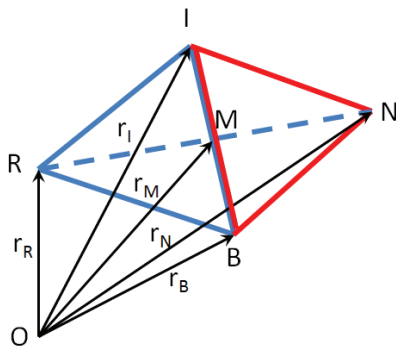
### Mișcările:

- reflectare
- expansiune
- contracție parțială
- contracție totală

mută (în general) vârful căreia îi corespunde cea mai proastă valoare a funcției obiectiv.

## Metoda simplexului descendent (Nelder-Mead)

**Reflectarea** - mișcarea punctului cel mai prost se face prin simetrie față de centrul feței opuse a.î măsura simplexului să se conserve.



Simplex IBR, unde  
 $f(B) < f(I) < f(R)$

(B = bun, I = intermediar, R = rău)

M - mijlocul segmentului IB:

$$\mathbf{r}_M = (\mathbf{r}_I + \mathbf{r}_B)/2$$

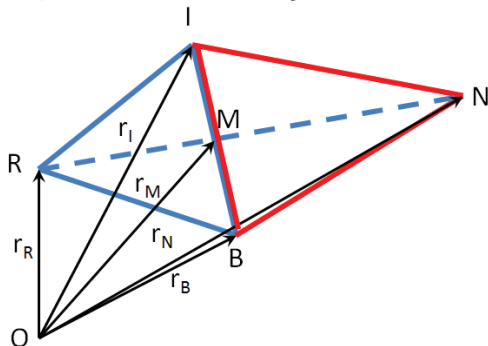
$$\|\mathbf{MN}\| = \|\mathbf{RM}\|$$

$$\mathbf{r}_N = \mathbf{r}_M + \mathbf{MN} = \mathbf{r}_M + \|\mathbf{MN}\| \frac{\mathbf{RM}}{\|\mathbf{RM}\|} = \mathbf{r}_M + \mathbf{r}_M - \mathbf{r}_R = 2\mathbf{r}_M - \mathbf{r}_R. \quad (27)$$

**Dacă  $f(N) < f(R)$  atunci reflectarea este reușită, noul simplex IBN**

## Metoda simplexului descendent (Nelder-Mead)

**Expansiunea** - e o mișcare făcută pentru a accelera căutarea.



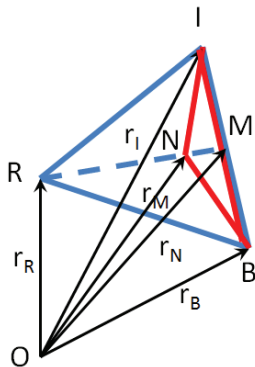
Simplex IBR, unde  
 $f(B) < f(I) < f(R)$   
 $\mathbf{r}_M = (\mathbf{r}_I + \mathbf{r}_B)/2$   
 $\|MN\| > \|RM\|$

$$\mathbf{r}_N = \mathbf{r}_M + \mathbf{MN} = \mathbf{r}_M + \|MN\| \frac{\mathbf{RM}}{\|RM\|} = \mathbf{r}_M + g(\mathbf{r}_M - \mathbf{r}_R). \quad (28)$$

$g > 1$ . Uzual  $g = 2$ . După expansiune măsura simplexului crește.  
 "Expansiunea" cu  $g < 1$  se numește tot reflectare.

## Metoda simplexului descendent (Nelder-Mead)

**Contractarea parțială** - se face atunci când simplexul ajunge într-o vale și el încearcă să pătrundă mai adânc în vale.



Simplex IBR  
 $f(B) < f(I) < f(R)$   
 $\mathbf{r}_M = (\mathbf{r}_I + \mathbf{r}_B)/2$

$$\mathbf{r}_N = \mathbf{r}_M + \mathbf{MN} = \mathbf{r}_M + b\mathbf{MR} = \mathbf{r}_M + b(\mathbf{r}_R - \mathbf{r}_M). \quad (29)$$

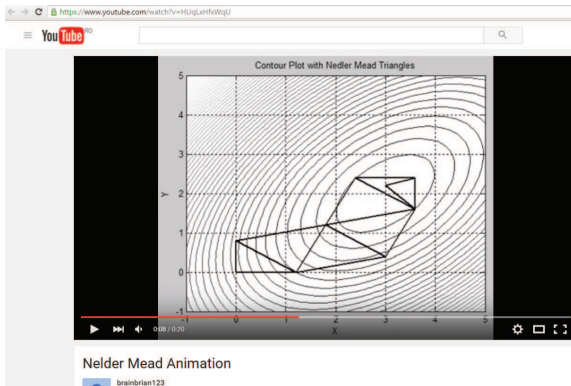
$b \in (0, 1)$ , în mod uzual  $b = 0.5$







## Metoda simplexului descendent (Nelder-Mead)



<https://www.youtube.com/watch?v=HUqLxHfxWqU>



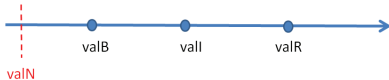
## Metoda simplexului descendent (Nelder-Mead)

**Algorithm** - se bazează pe reflectare și în funcție de rezultat se încearcă alte mișcări

```
; inițializare simplex  
P1 = ...  
P2 = ...  
P3 = ...  
g = 2 ; factor de expansiune  
b = 0.5; factor de contractare parțială  
; sortare [B, I, R, valB, valI, valR] = sortează(P1,P2,P3)  
repetă  
    ; încearcă reflectare  
    M = (B + I)/2  
    N = 2 M - R  
    valN = f(N)
```



# Metoda simplexului descendent (Nelder-Mead)

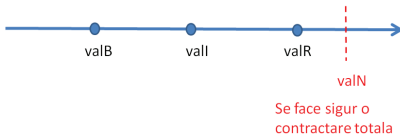


Încerc expansiune

```
; cazul 1
dacă valN < valB ; sunt în direcția bună, încerc accelerare
    ; încerc expansiune
    Nexp = M + g(M-R)
    valNexp = f(Nexp)
    dacă valNexp < valN
        ; expansiune reușită
        R = I
        I = B
        B = Nexp
        valR = valI
        valI = valB
        valB = valNexp
    altfel
        ; reflectare reușită
        R = I
        I = B
        B = N
        valR = valI
        valI = valB
        valB = valN
```



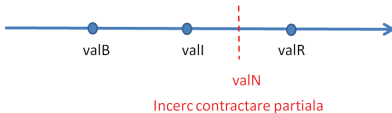
## Metoda simplexului descendent (Nelder-Mead)



; cazul 2

**dacă**  $valN > valR$  ; sunt într-o direcție total greșită  
; se face contracție totală păstrându-l pe cel mai bun  
 $N = (B+R)/2$   
 $[B, I, R, valB, vall, valR] = \text{sortează}(B, M, N)$

# Metoda simplexului descendent (Nelder-Mead)



; cazul 3

**dacă**  $valN \in (valR, valI)$  ;

; încerc contractare parțială

$Nc = M + b(R-M)$

$valNc = f(Nc)$

**dacă**  $valNc < valN$

; contractare parțială reușită

$[B, I, R, valB, valI, valR] = \text{sortează}(B, I, Nc)$

**altfel**

; reflectare reușită, dar același B

$R = N$

$valR = valN$

## Metoda simplexului descendent (Nelder-Mead)



Ramane doar reflectarea

; cazul 4

**dacă**  $valN \in (valI, valB)$  ;

    ; reflectare reușită, același B

$R = I$

$I = N$

$valR = valI$

$valI = valN$

**până când** (criteriu)



## Metoda simplexului descendent (Nelder-Mead)

### Criteriul de oprire

- Este delicat în orice optimizare multidimensională deoarece nu există posibilitatea de a aplica tehnici de încadrare, deci nu se poate cere o toleranță pentru fiecare variabilă independentă;
- Exemple de criterii de oprire:

1.

$$\|valB_{nou} - valB\| \leq \varepsilon \|valB\| + eps \quad (30)$$

2.

$$\|B_{nou} - B\| \leq \varepsilon \|B\| + eps \quad (31)$$

3. Numărul de evaluări de funcții  $\geq$  o valoare impusă.

- Oricare din criterii poate conduce la o soluție proastă, de aceea se recomandă restartarea algoritmilor din punctul în care se pretinde că s-a găsit minimul.



# Metoda Powell

Formularea problemei:

$$(x_1^*, x_2^*, \dots, x_n^*) = \arg \min f(x_1, x_2, \dots, x_n). \quad (32)$$

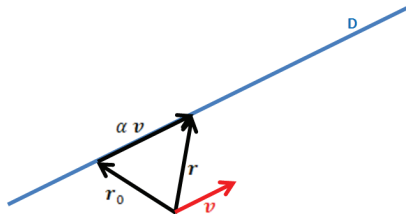
$n > 1$

- Metoda Powell are nevoie de un algoritm de minimizare 1D ca parte a strategiei de calcul.



## Metoda Powell

Reamintire: ecuația vectorială a unei drepte în spațiul nD.



Dreapta care conține  
punctul  $\mathbf{r}_0$  și are direcția  
 $\mathbf{v}$

$$\mathbf{r} = \mathbf{r}_0 + \alpha \mathbf{v} \quad (33)$$

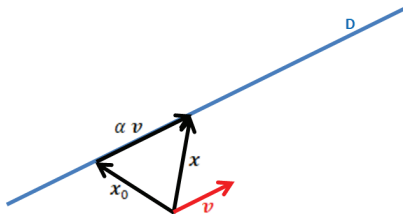
- $\mathbf{r}$  - vector de poziție;
- $\mathbf{r}_0$  - vector de poziție al unui punct fix pe dreaptă;
- $\alpha$  - coordonata de-a lungul drepte;
- $\mathbf{v}$  - vector (fix) ce orientează dreapta.



## Metoda Powell

### Minimizarea după o direcție a unei funcții de mai multe variabile

$$f : \mathbb{R}^n \rightarrow \mathbb{R} \quad f(\mathbf{x}) = f(x_1, x_2, \dots, x_n)$$



$$f(\mathbf{x})|_{\mathbf{x} \in D} = f(\mathbf{x}_0 + \alpha \mathbf{v})$$

Se definește  $t : \mathbb{R} \rightarrow \mathbb{R}$

$$t(\alpha) = f(\mathbf{x}_0 + \alpha \mathbf{v})$$

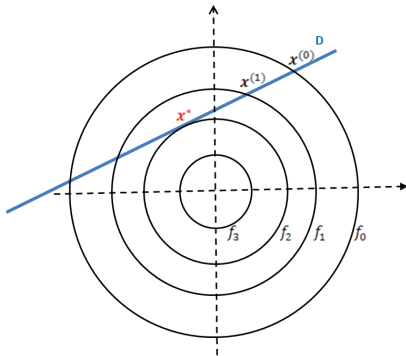
unde  $\mathbf{x}_0$  și  $\mathbf{v}$  sunt date.

Problema minimizării nD funcției  $f$  după direcția  $\mathbf{v}$  se reduce la problema minimizării 1D a funcției  $t$ .

## Metoda Powell

## Minimizarea după o direcție a unei funcții de mai multe variabile

### Semnificație geometrică:



```

procedură [x,fmin] = linmin(x,v)
    [α,tol,fmin] = secțiunea_aur(...,f1D,...)
    x = x + αv
retur

```

$$f : \mathbb{R}^2 \rightarrow \mathbb{R}$$

$$f(x_1, x_2) = x_1^2 + x_2^2$$

$$f_3 < f_2 < f_1 < f_0$$

- Minimul după direcția D se află pe cercul cu centrul în origine, tangent la D.
- Minimizarea nu se face exact.

**funcție**  $[t] = f_1 D(\alpha)$   
 $t = f(\mathbf{x} + \alpha \mathbf{v})$   
**întoarce**  $t$



## Metoda Powell

**Ideea:** căutări succesive pe  $n$  direcții liniar independente  $\mathbf{v}^{(i)}$ ,  $i = 1, n$ .

- Inițializare  $\mathbf{x}^{(0)}$
- Direcția  $\mathbf{v}^{(1)} \Rightarrow D_1: \mathbf{x} = \mathbf{x}^{(0)} + \alpha \mathbf{v}^{(1)} \Rightarrow \alpha_1$  prin minimizare;  
Minimul după această direcție:  $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha_1 \mathbf{v}^{(1)}$
- Direcția  $\mathbf{v}^{(2)} \Rightarrow D_2: \mathbf{x} = \mathbf{x}^{(1)} + \alpha \mathbf{v}^{(2)} \Rightarrow \alpha_2$   
Minimul:  $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha_2 \mathbf{v}^{(2)}$
- În general  $\mathbf{x}^{(i)} = \mathbf{x}^{(i-1)} + \alpha_i \mathbf{v}^{(i)}$  unde  $\alpha_i$  se determină prin minimizare după direcția  $\mathbf{v}^{(i)}$ ,  $i = 1, n$ .

Dacă  $|f(\mathbf{x}_n) - f(\mathbf{x}_0)|$  e mare, atunci se reia căutarea cu o nouă inițializare, e.g.  $\mathbf{x}_{\text{nou}}^{(0)} = \mathbf{x}^{(n)}$ .

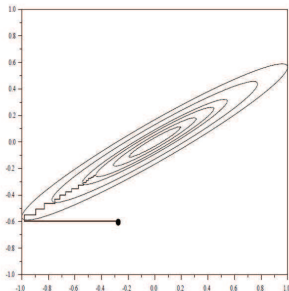
**"Iterație" în metoda Powell** = calculul unui minim aproximativ pornind dintr-o inițializare și făcând  $n$  minimizări consecutive după direcțiile  $\mathbf{v}^{(i)}$ .



## Metoda Powell

**Ideea:** căutări succesive pe  $n$  direcții .

Căutarea după direcțiile axelor s-ar putea să ducă la o convergență foarte lentă.



Ex - vale îngustă, iar axa văii nu coincide cu nicio direcție de căutare.

⇒

Trebuie ca setul de direcții de căutare să fie adaptat funcției, a.î. avansul către minim să fie cât mai rapid.



## Metoda Powell

Setul de direcții se ajustează după fiecare iterație Powell, pe baza informațiilor obținute la acea iterație.

Ideea:

1. se elimină o direcție din cele  $n$ ;
2. se adaugă direcția  $\mathbf{v}^{(m)}$  a deplasării medii

$$\mathbf{v}^{(m)} = \mathbf{x}^{(n)} - \mathbf{x}^{(0)} \quad (34)$$

3. noua inițializare se determină făcând încă o minimizare după direcția  $\mathbf{v}^{(m)}$

$$\mathbf{x}_{\text{nou}}^{(0)} = \mathbf{x}^{(n)} + \alpha_{n+1} \mathbf{v}^{(m)} \quad (35)$$

Elementul cheie este alegerea direcției eliminate.

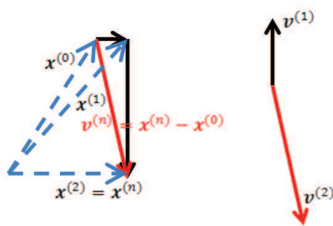
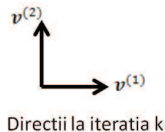
Există două tehnici posibile.



## Metoda Powell

### Metoda Powell de bază:

- se elimină întotdeauna  $\mathbf{v}^{(1)}$ ;
- se renumerează direcțiile ( $\mathbf{v}^{(2)}$  devine  $\mathbf{v}_{\text{nou}}^{(1)}$ , etc.);
- se adaugă  $\mathbf{v}^{(m)}$  ca ultimă direcție.



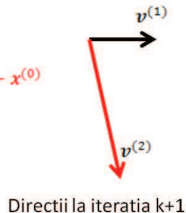
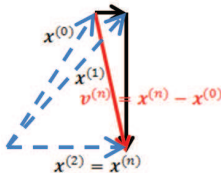
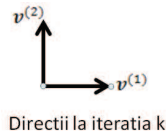
Poate da rezultate proaste dacă direcțiile adăugate tind să devină aproape coliniare cu unele din direcțiile existente.



## Metoda Powell

### Metoda Powell modificată:

- se elimină direcția cea mai bună  $\mathbf{v}^{(j)}$  (după care funcția a avut cea mai mare scădere la iterația curentă);
- se pune  $\mathbf{v}^{(n)}$  în locul lui  $\mathbf{v}^{(j)}$ ;
- se adăuga  $\mathbf{v}^{(m)}$  ca ultimă direcție.







## Metoda Powell modificată

**Date:** inițializarea  $\mathbf{x}_0$ , direcțiile inițiale  $\mathbf{v}_i$ ,  $i = 1, \dots, n$

$f_0 = f(\mathbf{x}_0)$

**repetă**

dfmax = 0

; cea mai mare scădere a funcției

j = 0

; direcția cu cea mai mare scădere

xv = x0, fv = f0

**pentru** i = 1, n

; parcurge direcțiile

[xn, fn] = linmin(xv, v<sub>i</sub>)

; minimizează pe direcția v<sub>i</sub>

**dacă** |fn - fv| > dfmax **atunci**

; am găsit o scădere mai mare

dfmax = |fn - fv|

j = i

xv = xn

; actualizează x și f(x)

fv = fn

vm = xn - x0

; direcția medie

deltaf = |fn - f0|

; variația funcției la iterația curentă

v<sub>j</sub> = v<sub>n</sub>

; pune direcția v<sub>n</sub> pe poziția j

v<sub>n</sub> = vm

; pune direcția v<sub>m</sub> pe ultima poziție

[x0, f0] = linmin(xn, vm)

; determină noua inițializare

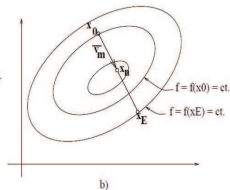
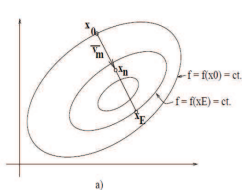
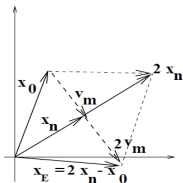
**până când** deltax < eps

Condiția de oprire ar putea fi de tip relativ:

$$|f(\mathbf{x}^{(n)}) - f(\mathbf{x}^{(0)})| \leq \text{ftol} \cdot ((f(\mathbf{x}^{(n)}) + f(\mathbf{x}^{(0)}))/2 + \text{eps})$$

## Metoda Powell

**Îmbunătățiri** - uneori este mai bine să nu se modifice direcțiile de căutare.



$$\begin{aligned} \mathbf{x}_E &= \mathbf{x}_n + \mathbf{v}_m = \\ &= 2\mathbf{x}_n - \mathbf{x}_0 \end{aligned}$$

b):  $f(\mathbf{x}_E) \geq f(\mathbf{x}_0) \Rightarrow$  nu se modifică direcțiile.



## Metoda Powell

**Îmbunătățiri** - uneori este mai bine să nu se modifice direcțiile de căutare.

Dacă la iterația curentă scăderea nu se datorează cu precădere unei anumite direcții, adică dacă

$$f_0 - f_n \gg \Delta f_{\max}$$

unde  $\Delta f_{\max}$  este scăderea cea mai mare (după direcția  $j$ )  
 $\Rightarrow$  nu se modifică direcțiile.



## Metoda Powell

### Ordin de complexitate

- Se dem. că metoda Powell de bază aplicată unei funcții pătratice de  $n$  variabile conduce la minim după  $n$  iterații  $\Rightarrow n(n+1)$  minimizări 1D  $\Rightarrow O(n^2)$  dacă se consideră ca operație elementară minimizarea unei funcții 1D
- Dacă funcția nu e pătratică sunt necesare mai multe iterații.
- În metoda modificată se pierde ordinul de complexitate pătratic, dar algoritmul devine mai robust.