

**Praca zbiorowa. Powodzenia na egzaminie!!!!**

**Macierz dodatnio określona** - wszystkie wartości własne dodatnie.

**Macierz nieosobliwa** -  $\det(A) \neq 0$

**Macierz odwrotna**  $A^{-1}$  istnieje gdy  $\det(A) \neq 0$

**Układ równań ma jednoznaczne rozwiązanie** gdy  $\det(A) \neq 0$

**Macierz symetryczna** ma rzeczywiste wartości własne ( był do tego jakiś dowód ale walić )

**Typy macierzy:**

▣ **Macierz symetryczna:**

$$a_{ij} = a_{ji} \\ A^T = A$$

Każdą macierz symetryczną można zdiagonalizować macierzą ortogonalną  
Ślad ( wartości własne macierzy ) jest niezmiennikiem podczas ortogonalizacji

▣ **Macierz hermitowska** - macierz równa swojemu sprzężeniu hermitowskiemu

$$a_{ij} = \overline{a_{ji}} \\ A^* = A$$

▣ **Macierz ortogonalna** - macierz kwadratowa spełniająca równość:

$$A^T A = A A^T = I$$

$I$  = macierz jednostkowa

Macierz jest ortogonalna, jeśli jej macierzą odwrotną jest macierz do niej transponowana.

▣ **Macierz unitarna:**

$$U^\dagger U = U U^\dagger = I$$

$I$  - macierz jednostkowa

$$U^\dagger = U^{-1}$$

$U^\dagger$  jest sprzężeniem hermitowskim macierzy (zestawienie operacji transpozycji i sprzężenia zespolonego)

### Równanie własne macierzy:

$$Ax = \lambda x$$

**Normy wektorów** - nie chce mi się pisać wzorów ale sobie ogarnijcie xd

- taksówkowa
- Euklidesowa
- maximum

**Promień spektralny macierzy** - maksymalna wartość modułu wartości własnej  $\lambda$  macierzy (największa na moduł wartość własna)

**Norma macierzy i współczynnik uwarunkowania:**

[https://www.youtube.com/watch?v=c-LJa0Pd\\_fm](https://www.youtube.com/watch?v=c-LJa0Pd_fm)

WSPÓŁCZYNNIK UWARUNKOWANIA MACIERZY

$$\text{cond}(A) = \|A\| * \|A^{-1}\|$$
$$\frac{1}{\det A} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$
$$A = \begin{bmatrix} 3 & 2 \\ 2 & 0 \end{bmatrix} \quad A^{-1} = -\frac{1}{4} \begin{bmatrix} 0 & -2 \\ -2 & 3 \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{2} \\ \frac{1}{2} & -\frac{3}{4} \end{bmatrix}$$
$$\det A = 3 \cdot 0 - (2 \cdot 2) = -4$$
$$\|A\| = \max(|3| + |2|, |2| + |0|) = \max(5, 2) = 5$$
$$\|A^{-1}\| = \max(|0| + |\frac{1}{2}|, |\frac{1}{2}| + |-\frac{3}{4}|) = \max(\frac{1}{2}, \frac{5}{4}) = \frac{5}{4}$$
$$\text{cond}(A) = 5 \cdot \frac{5}{4} = \frac{25}{4}$$

**Współczynnik uwarunkowania macierzy:**

$K(A) = \text{Norm}(A) * \text{Norm}(A^{-1})$ , gdzie  $\text{Norm}(A)$  to norma macierzy (zapisuje się jak podwójny moduł (2 pionowe kreski) ale jestem zbyt leniwy, żeby szukać tych symboli xd)

albo

$$K(A) = \frac{\lambda_{\max}}{\lambda_{\min}}$$

albo zamiast  $\lambda$  dajemy sigma (czyli wartości szczególne rozkładu SVD)

**Eliminacja Gaussa** - koszt  $O(N^3)$

Sprowadzanie do postaci macierzy trójkątnej górnej, gdzie zmienna  $x_1$  występuje tylko w pierwszym wierszu,  $x_2$  w pierwszym i drugim i tak dalej.

Usunięcie zmiennej  $x_1$  z jednego wiersza -  $O(N)$  operacji

A usuwamy z  $N-1$  wierszy więc  $O(N^2)$

Oraz powtarzamy dla  $x_2, x_3$  i tak dalej więc ostatecznie  $O(N^3)$

**Backsubstitution** -  $O(N^2)$  - podstawianie/zamiana zmiennej  $x_i$   $O(N)$  a musimy usunąć/podstawić  $N$  zmiennych

Zawodzi przy dzieleniu przez ZERO.

np.  $a_{11} = 0$

Potrzebna znajomość kolumny wyrazów wolnych, gdyż też są permutowane.

**Wybór elementu podstawowego** - element kolumny największy na moduł powinien się znaleźć na diagonalu.

**Koszt pivotu** (wyboru elementu podstawowego) wynosi  $O(N)$  dla kroku, w sumie  $O(N^2)$

Powinno się robić częściowy pivoting (permutacja wierszy)

**Pełny pivoting** (permutacja wierszy i kolumn) -  $O(N^3)$  duży koszt (taki jak samej Eliminacji, nie opłaca się raczej)

**Rozkład LU** - koszt  $O(N^3)$

Obliczanie nieznanego elementu  $L$  lub  $U$  -  $O(N)$

A musimy obliczyć  $N^2$  elementów więc ostatecznie  $O(N^3)$

Gdy już mamy rozkład to układ równań rozwiązujemy ze wzorów:

$$Ax = LUx = b$$

$$Ly = b$$

$$Ux = y$$

**Algorytmy LU: Doolittle'a albo Crouta** (jedyne na diagonalu albo w L albo w U)

*Dodatkowo możliwy i wskazany częściowy pivoting (pełny niemożliwy przez symetrię)*

**Faktoryzacja Cholesky'ego** - koszt  $O(N^3)$

$$A = CC^T, \quad C - \text{macierz trójkątna dolna} - \text{tak zwany czynnik Cholesky'ego}$$

A - symetryczna i dodatnio określona !!

- 2x szybsze od LU
- niemożliwy wybór elementu podstawowego (nawet częściowy ;(( chuj i tak jest szybsze)
- wymaga pierwiastkowania (stosunkowo kosztowne)

**Faktoryzacja LDL** - koszt  $O(N^3)$

$$A = LDL^T, \quad L - \text{trójkątna dolna}$$

A - symetryczna i dodatnio określona !!

D - macierz diagonalna o dodatnich elementach

**Macierze rzadkie:**

Są epickie bo:

- w ciul zer, lubimy zera
- Dla trójdzielnej macierzy rozkład LU w czasie  $O(N)$
- Jeśli możliwa jest faktoryzacja Cholesky'ego macierzy M-dielnej, także jej czynnik Cholesky'ego będzie M-dielny ale może dość do wypełnienia - będzie mniej zer

**Minimum Degree Algorithm**

zamiast  $Ax=b$

rozwiązujemy równanie  $(PAP^T)(Px)=Pb$ , P - ortogonalna macierz Permutacji

Macierz permutacji ciężko jest znaleźć, jest to problem NP zupełny, czyli po polskiemu: nie da się tego obliczyć od tak i do tego stosowane są te algorytmy Minimum Degree Algorithm.

**Transformacja Householdera** - koszt transformacji na jednym wektorze  $O(N)$

Efektem działania macierzy Householdera na wskazany wektor jest wyzerowanie wszystkich jego składowych poza pierwszą i "przelanie" całej jego długości na pierwszą składową.

H - symetryczna i ortogonalna macierz Householdera jest to macierz przekształcenia wektora, która odbija go względem płaszczyzny

$$H = I - 2xx^T, \quad x - \text{wektor znormalizowany}$$

**Faktoryzacja QR** - koszt  $O(N^3)$

$P_1$  - ortogonalna macierz transformacji Householdera

Budujemy transformację Householdera na kolejnych kolumnach macierzy A, by wyzerować elementy poniżej diagonalu i sfaktoryzować (rozłożyć) macierz na macierz ortogonalną Q i trójkątną dolną R

$$P_{n-1} P_{n-2} \dots P_1 A = R$$

$P_{n-1} P_{n-2} \dots P_1$  to jest tak naprawdę macierz  $Q^T$  bo jest to zestawienie macierzy ortogonalnych P które w wyniku dają macierz ortogonalną  $Q^T$ .

$$A=QR$$

teraz rozwiązanie układu poprzez:

$$Ax=b$$

$$QRx=b$$

$$Rx=Q^T b$$

**Obroty Givensa** - koszt  $O(N)$

$$G(i, j) = \begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & 1 & & & & \\ & & & c & & s & \\ & & & & \ddots & & \\ & & & & & 1 & \\ & & & & & & \ddots \\ & & -s & & c & & \\ & & & & & 1 & \\ & & & & & & \ddots \\ & & & & & & & 1 \end{bmatrix}$$

Macierz Givensa  $G(i, j)$  - ortogonalna macierz obrotu w płaszczyźnie  $(x_i, x_j)$  o kąt  $\theta$  przeciwnie do ruchu wskazówek zegara.

Stosuje się ją do selektywnego zerowania poszczególnych elementów macierzy,

Dla macierzy trójdzielnej obrotu Givensa to  $O(N)$  - stały koszt kroku razy  $N$  kroków.

**Rozkład SVD** - rozkład na macierze:

$$A = U \Sigma V^T$$

$U$  - macierz kolumnowo ortogonalna

$\Sigma$  - macierz Pseudodiagonalna z wartościami szczególnymi (pierwiastek z  $\lambda$ ) na diagonalu

$V$  - macierz ortogonalna

Algorytm:

- 1) Znajdujemy wartości własne  $\lambda_i$  macierzy  $AA^T$  albo  $A^T A$  zależnie co daje w wyniku mniejszą macierz (Tutaj  $A$  nie musi być kwadratowa)
- 2) Określamy liczbę  $r$  niezerowych wartości własnych macierzy  $AA^T$  albo  $A^T A$
- 3) Znajdujemy ortonormalne wektory własne macierzy  $AA^T$  albo  $A^T A$  odpowiadające wartościom własnym. Tworzymy macierz ortogonalną  $V$  której

kolejne kolumny tworzą wektory własne macierzy  $AA^T$  uporządkowane w malejącym porządku co do odpowiadających im wartości własnych.

- 4) Tworzymy pseudodiagonalną macierz  $\Sigma$  umieszczając na diagonalu pierwiastki kwadratowe z wartości własnych  $\lambda_i$  w porządku malejącym.
- 5) Znajdujemy pierwszych  $r$  wektorów kolumnowych macierzy  $U$  z równań  $u = \frac{1}{\sqrt{\lambda}} Av_j$  dla  $j=1,2,\dots, r$ .
- 6) Dodajemy do macierzy  $U$  pozostałe  $m-r$  wektorów wykorzystując proces ortogonalizacji Grama-Schmidta (Good luck, krzyż na drogę †)

## \\ / ۞ ۞ \ / METODY ITERACYJNE \\ / ۞ ۞ \ /

### Metoda Jacobiego:

- Zbieżna, jeśli macierz  $A$  jest silnie diagonalnie dominująca

**Macierz silnie diagonalnie dominującą nazywamy** taką macierz  $A$  o stopniu  $n$ , w której występuje nierówność ostra (mocna, czyli  $>$ , bez  $=$ ) dla poniższego warunku:

$$|a_{ii}| \geq \sum_{\substack{k=1 \\ k \neq i}}^n |a_{ik}|$$

czyli po polsku: moduły elementów na diagonalu są większe od sumy elementów macierzy stojącej w danym wierszu bez elementu na diagonalu.

- Zbieżna dla dowolnego przybliżenia początkowego  $x_0$  jeśli promień spektralny  $-D^{-1}(L+U)$  jest mniejszy od 1.

(Promień spektralny - największa na moduł wartość własna  $\max |\lambda_i|$ )

Algorytm:

- zapisujemy układ w postaci  $Ax=b$
- dzielimy macierz A na sumę macierzy  $L + D + U$
- Obliczamy macierz  $N = D^{-1}$  poprzez podniesienie wartości na diagonalu macierzy D do potęgi -1
- Obliczamy  $M = -D^{-1}(L+U) = -N(L + U)$
- rozpoczynamy od wektora  $x_0$  ( np same zera w wektorze)
- kolejne iteracje wzorem:  
$$x^{n+1} = Mx + Nb$$
 b - wektor wyrazów wolnych

**EPICKIE WYTŁUMACZENIE NA PRZYKŁADZIE W LINKU PONIŻEJ:**

<http://www.algorytm.org/procedury-numeryczne/metoda-jacobiego.html>

**Metoda Gaussa-Seidela:**

- Zbieżna, gdy macierz A jest symetryczna i dodatnio określona

$$Mx^{k+1} = Nx^k + b$$

$$M = D + L$$

$$N = -U$$

Algorytm:

- Zapisujemy układ równań w postaci  $Ax=b$
- Rozpisujemy A na sumę  $L + D + U$
- Obliczamy macierz  $N = D^{-1}$  poprzez podniesienie wartości na diagonalu macierzy D do potęgi -1
- Liczymy kolejno:  $D^{-1}b$ ,  $D^{-1}L$ ,  $D^{-1}U$
- rozpoczynamy od wektora  $x_0$  (np wektor z samymi zerami)
- liczymy kolejne iteracje wzorem:  
$$x^{n+1} = D^{-1}b - D^{-1}Lx^{n+1} - D^{-1}Ux^n$$

Przykład obliczania Kapska-Sznycela poniżej:

<http://www.algorytm.org/procedury-numeryczne/metoda-gaussa-seidela.html>



## SOR (succesive over relaxation)

Dla Gaussa-Seidela	Dla Jacobiego
$M = D + L$ $N = -U$	$M = D$ $N = -(L + U)$

Jeśli  $\rho(M^{-1}N)$  (promień spektralny) w METODZIE GAUSSA SEIDELA jest bliskie jedności, zbieżność metody jest bardzo wolna. Można ją poprawić dodając do wzoru parametr relaksacji

## Metoda gradientów sprzężonych

Dla macierzy symetrycznej i dodatnio określonej

$\nabla f$  - gradient - pole wektorowe o składowych będących pochodnymi cząstkowymi  $f$ .

Zbiega się po  $N$  krokach.

Dla macierzy pełnej - koszt  $O(N^3)$

Dla pasmowej o szerokości pasma  $M$  - koszt  $O(MN^2)$

Zamiast liczyć  $Ax=b$

Poszukujemy minimum dodatnio określonej formy kwadratowej:

$$f(x) = \frac{1}{2}x^T Ax - b^T x + c$$

Minimalizacja polega na poszukiwaniu kierunku najszybszego spadku i poruszaniu się w tym kierunku. Każdy krok to poszukiwanie takiej wartości alfa, która minimalizuje:

$$f(x^{i+1}) \quad x^{i+1} = x^i + \alpha p^i$$

Kierunek poruszania zmianami metodą gradientów sprzężonych lub metodą najszybszego spadku (ta druga jest modyfikacją metody gradientu prostego).

### Metoda potęgowa

Szukanie największej wartości własnej.

- nie działa dla macierzy niesymetrycznych
- powolna gdy wartości własne są do siebie zbliżone co do modułu

Algorytm:

- przyjmujemy wektor początkowy np  $x_0 = [1, 1, \dots, 1]$  (oczywiście jest to wektor stojący ale zapisałem go na leżąco)
- Mnożymy macierz  $A$  przez wektor  $x_i$ ,  $Ax_i$
- Normalizujemy wektor (dzielimy każdą wartość wektora przez największą wartość co do modułu i wyciągamy tą liczbę przed wektor)
- liczba ta jest kolejnym przybliżeniem wartości własnej  $\lambda$  (dominującej)
- Obliczamy aż różnica  $\lambda$  przy kolejnych iteracjach jest bliska 0. ( $= 0$  ?)

## Interpolacja

### Interpolacja odcinkami liniowa

- prowadzimy łamaną pomiędzy węzłami interpolacji (ogólnie istnieje nieskończenie wiele funkcji ciągłych, które są sobie równe w skończonej liczbie węzłów)
- Brzydki sposób, śmierdzi strasznie kapskiem, nie róbmy tak.

### Interpolacja Wielomianowa - $O(N^2)$

Do wielomianu za  $x$  kolejno wstawiamy  $x_1, x_2, \dots, x_n$  oraz wartości wielomianu w tych punktach  $f_1, f_2, \dots, f_n$

Wyznacznik macierzy Vandermonde'a jest różny od zera jeśli żadne punkty  $x_1, x_2, \dots, x_n$  nie pokrywają się - ma jednoznaczne rozwiązanie

Oscylacje Rungego - duże wahania przeważnie na krańcach interpolacji wynikające np z nieciągłości funkcji i sztywności wielomianów wysokiego stopnia.

### **Interpolacja Lagrange'a**

Jest to interpolacja za pomocą wielomianów ale zamiast rozwiązywać układ równań w celu znalezienia współczynników korzystamy ze wzoru interpolacyjnego.

Wartość funkcji w punkcie x:

$$L(x) = \sum_{i=1}^n y_i l_i(x)$$

x - argument, dla którego chcemy znaleźć wartość funkcji

y<sub>i</sub> - wartość funkcji odpowiadająca argumentowi x<sub>i</sub>

### **Interpolacja Hermite'a**

Użyteczna jeśli znamy nie tylko wartości funkcji interpolowanej w węzłach ale i wartości pochodnej w węźle.

niewielkie zastosowanie praktyczne, duże teoretyczne

### **Interpolacja za pomocą funkcji sklejanych ( splajnów )**

Splajn rzędu k to funkcja która:

- lokalnie jest wielomianem rzędu k.
- jest ( k - 1 ) krotnie różniczkowalna w węzłach ( jej pochodne rzędu k-2 i niższych są ciągłe )

Najczęściej używa się funkcji sklejanych trzeciego rzędu czyli splajnów kubicznych.

### **Splajn kubiczny**

Zakładamy, że oprócz wartości funkcji w węzłach znamy także drugie pochodne funkcji w węzłach ( Jest to tylko założenie robocze )

W każdym przedziale konstruujemy wielomian trzeciego stopnia, przedziały [ x<sub>j</sub>, x<sub>j+1</sub> ] dla j = 1, 2, ..., n - 1

W rzeczywistości nie znamy wartości drugiej pochodnej f''<sub>j</sub>. Korzystamy jednak z wymogu ciągłości pierwszej pochodnej w węzłach. Żądamy by pochodna y<sub>j</sub>(x) w prawym krańcu przedziału równała się pochodnej y<sub>j+1</sub>(x) w lewym krańcu.

Otrzymujemy przejebane równanie z poprzedniego przejebanego równania którego nie będę rozpisywał i pewnie znać nie trzeba.

Ważne jest, że to równanie daje nam trójdzielny układ równań na nieznane wartości  $\{f_j\}$

Dla równoodległych węzłów macierz posiada łatwy do znalezienia rozkład Cholesky'ego

- rozwiązujemy układ równań ( tak ten przejebany)  $O(N)$
- w celu znalezienia wartości między węzłami wykonujemy inne równanie tyle razy ile wartości chcemy znaleźć
- Wychodzi na to że złożoność jest  $O(N) \cdot \text{ilość powtórzeń}$  (pewności nie mam)

Przykład splajnu:

<https://www.youtube.com/watch?v=b4Ro7i9c2QE>

### **Splajn bikubiczny ( na płaszczyźnie )**

Mamy funkcję dwu zmiennych  $f(x,y)$  stabelaryzowaną w węzłach dwuwymiarowej siatki.

Wiersze siatki odpowiadają ustalonym wartościom zmiennej  $y$ .

Kolumny natomiast wartości  $x$ .

- Przeprowadzamy splajn wzdłuż każdego wiersza
- obliczamy wartość każdego z powyższych splajnów w punkcie  $x$
- przez powyższe punkty przeprowadzamy splajn w kierunku  $y$

## Całkowanie

Całka - proces odwrotny do pochodnej funkcji

Całka oznaczona - geometryczną interpretacją jest pole powierzchni między wykresem funkcji a osią odciętych w pewnym przedziale  $[a,b]$

Wzory na całkowanie przybliżone tak zwane kwadratury uzyskuje się przez całkowanie odpowiednich wielomianów interpolacyjnych

### **Kwadratura Newtona - Cotesa**

Opiera się na interpolacji wielomianami niskiego stopnia.

### **Kwadratury:**

#### ■ Metoda trapezów:

$$\text{całka}(x) = \frac{b-a}{2}(f_0 + f_1)$$

#### ■ Metoda Simpsona:

$$\text{całka}(x) = \frac{b-a}{6}(f_0 + 4f_1 + f_2)$$

#### ■ Metoda $\frac{3}{8}$

$$\text{całka}(x) = \frac{b-a}{8}(f_0 + 3f_1 + 3f_2 + f_3)$$

#### ■ Metoda Milne'a:

$$\text{całka}(x) = \frac{b-a}{90}(7f_0 + 32f_1 + 12f_2 + 32f_3 + 7f_4)$$

### **Ekstrapolacja Richardchona (trochę mało, trzeba dodać coś więcej)**

Stosuje się ją do przyspieszenia zbieżności kwadratur (np. Newtona - Cotesa).

### **Metoda Romberga (trochę mało, trzeba dodać coś więcej)**

Wielokrotne zastosowanie ekstrapolacji w całkowaniu numerycznym, które prowadzi do utworzenia trójkątnej tablicy nazywane jest metodą Romberga.

### **Całkowanie po przedziałach nieskończonych**

Przy obliczaniu całek typu:  $\int_0^{\infty} f(x) dx$ , należy szczególnie uważać, aby numerycznie nie

“obliczyć” całki, która jest rozbieżna. Kwadratury służą do znajdowania przybliżonych wartości całek, o których wiemy, że istnieją. Funkcja podcałkowa musi w nieskończoności zmierzać dostatecznie szybko do zera, żeby całka istniała.

Całkę należy rozłożyć na sumę dwóch całek (ta sama funkcja, tylko granica w pierwszej od 0 do A, a w drugiej od A do  $\infty$ , gdzie A jest dostatecznie dużą stałą dodatnią by wartości funkcji  $x > A$  dostatecznie szybko zmierzały do zera, a całkę z tej funkcji łatwo dało się obliczyć **analitycznie**).

### **Kwadratury adaptacyjne**

Algorytm, który lokalnie sam dopiera krok całkowania, dostosowując go do charakteru zmienności funkcji. W tym celu algorytm musi mieć dwa niezależne oszacowania całki po danym przedziale - ich różnica jest miarą popełnianego błędu.

### **Całki wielowymiarowe**

Jeżeli wymiar całki jest  $\geq 3$  należy ją obliczać metodami Monte Carlo. Dla wymiaru = 2 metody Monte Carlo nie są konkurencyjne wobec podejścia tradycyjnego.

### **Krotność miejsca zerowego**

Mówimy, że  $x_0$  jest miejscem zerowym funkcji  $f(x)$  o krotności  $k$ , jeżeli w tym punkcie zeruje się funkcja wraz ze swoimi pochodnymi rzędu:  $k-1$ :  $f(x_0) = f'(x_0) = \dots = f^{(k-1)}(x_0) = 0$ .

Np. wielomian:  $P(x) = x^4 - x^3 - x^2 + x$  ma jednokrotne miejsce zerowe w  $x = -1$ ,  $x = 0$  i dwukrotne miejsce zerowe w  $x = 1$ .

Funkcja zmienia znak w otoczeniu miejsca o krotności nieparzystej i nie zmienia znaku w otoczeniu miejsca o krotności parzystej

## **Rozwiązywanie Równań Algebraicznych**

### **Metoda Bisekcji:**

Jeżeli funkcja jest ciągła i mamy 2 punkty w których znak funkcji jest przeciwny:

$$F(x_1) * F(x_2) < 0$$

to jako przybliżenie bierzemy środkowy punkt przedziału  $x_3 = \frac{1}{2} * (x_1 + x_2)$

Ustalamy w którym przedziale funkcja zmienia znak i powtarzamy.

Zbieżność jest liniowa.

Metoda działa dla miejsc zerowych o nieparzystej krotności.

### **Metoda regula falsi - metoda fałszywego położenia**

Funkcja  $f(x)$  jest ciągła.

Znamy dwa punkty w których znak jest przeciwny:

$$F(x_1) * F(x_2) < 0$$

Jako przybliżenie bierzemy punkt przecięcia siecznej przechodzącej przez punkty  $(x_1, f(x_1))$  oraz  $(x_2, f(x_2))$  z osią OX.

$$x_3 = \frac{f(x_1)x_2 - f(x_2)x_1}{f(x_1) - f(x_2)}$$

### Metoda siecznych:

Dowolne dwa punkty  $f(x_1) \neq f(x_2)$

Prowadzimy sieczną niezależnie od znaków. ( nie obowiązuje nas zasada  $F(x_1) * F(x_2) < 0$  z poprzednich metod i tym się różni też to od regula falsi (często mylone))

$x_3$  to miejsce przecięcia tej cieczonej z OX.

Bierzemy dwa ostatnie punkty i powtarzamy, aż do znalezienia miejsca zerowego.

### Metoda Newtona ( metoda stycznych )

założenia:

- W przedziale  $[a,b]$  znajduje się dokładnie jeden pierwiastek
- funkcja ma różne znaki na końcach przedziału
- pierwsza i druga pochodna mają stały znak w tym przedziale




Algorytm geometryczny:

- Wybieramy punkt startowy z którego prowadzona jest styczna w  $f(x_1)$
- Odcięta punktu przecięcia stycznej z osią OX jest pierwszym przybliżeniem  $x_2$
- Powtarzamy

Kolejne przybliżenia dane są wzorem

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

w mianowniku jest pochodna

(ja jebie ale maciupkie te wzory   

Może być rozbieżna i prowadzić do cykli

- ☑ Jest zbieżna kwadratowo dla jednokrotnych miejsc zerowych
- ☑ Metoda jest tym szybciej zbieżna, im bliżej poszukiwanego miejsca leży przybliżenie początkowe
- ☑ Zbieżna liniowo do wielokrotnych miejsc zerowych

### Metody wykrozystujące drugą pochodną

Metoda Newtona opiera się na rozwinięciu Taylora ( $f(x_0 + \delta) \cong f(x_0) + \delta * f'(x_0)$ ) do pierwszego rzędu. Można to uogólnić na rozwinięcie do rzędu drugiego [...]. Jak poprzednio, żądamy, aby lewa strona znikała, co prowadzi do kroku [...], a dalej po prostych przekształceniach do iteracji.

### Metoda Halleya

Inną metodę daje zastosowanie metody Newtona do równania:

$$g(x) = f(x) / \text{pierwiastek}(|f'(x)|) = 0$$

Każdy pierwiastek  $f(x)$ , który nie jest miejscem zerowym pochodnej, jest pierwiastkiem  $g(x)$ ; każdy pierwiastek  $g(x)$  jest pierwiastkiem  $f(x)$  (rozwiązaniem równania  $f(x) = 0$ ).

### Metoda Newtona

$$g: \mathbb{R}^N \rightarrow \mathbb{R}^N$$

- 1) rozwijając funkcję  $g$  w szereg Taylora do pierwszego rzędu otrzymujemy:  
 $g(x + \delta x) \cong g(x) + J \delta x$ , gdzie  $J$  - jacobian funkcji  $g$
- 2) Żądamy aby  $g(x + \delta x) = 0$ , skąd otrzymujemy  $\delta x = -J^{-1}g(x)$ .
- 3) prowadzi to do iteracji:  $x_{k+1} = x_k - J^{-1}(x_k) g(x_k)$

W tej metodzie trzeba obliczać w każdym kroku jacobia. Oznacza to, że w każdym kroku trzeba rozwiązywać inny układ równań liniowych, co czyni metodę dość kosztowną, zwłaszcza jeśli  $N$  (wymiar problemu) jest znaczne. Często dla przyspieszenia obliczeń macierz  $J$  zmieniamy nie co krok, ale co kilka kroków - pozwala to użyć tej samej faktoryzacji  $J$  do rozwiązania kilku kolejnych równań  $Jz = g(x_k)$ . Jest to dodatkowe uproszczenie, ale jest ono bardzo wydajne, przy  $N \gg 1$ .

Oczywiście zapis  $z = J^{-1}g$  należy rozumieć w ten sposób, że spełnia równanie  $Jz = g$ . Nie należy konstruować jawnej odwrotności jacobianu.



\\fz\ PANDA 3 \\fz\

(၃၅)

\r\n\r\n33333333333333333333 \r\n\r\n\r\n)

BOOM HEADSHOT