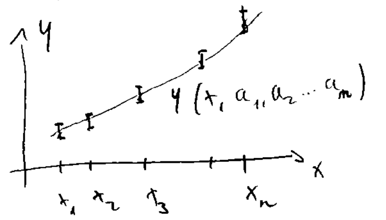
Fizmum 2.

Centrális határeloszlás tétel: ha van sok „tetszőleges" eloszlású valószínűségi változó összege, nagyjából Gauss eloszlású lesz.

Legkisebb négyzetek módszere



x mérése hibája kicsi

y-ban van a hiba

$$\chi^2 = \sum_{i=1}^{N} \left(y_i - y(x_i, a_1, a_2 \dots a_m)\right)^2$$ kvázi χ^2 :)

mennyi χ^2 minimuma? Feltevések: - a hibák Gauss eloszlásúak

- a hibák függetlenek
- minden mérésnél azonos a szórás
- csak y-ban van hiba

Maximum likelihood:

maximális valószínűségű paraméter becslés

$$P(y_i) = \exp\left[-\frac{1}{2}\left(\frac{y_i - y(x_i, a_1, a_2 \dots a_m)}{\sigma}\right)^2\right]$$

mennyi a valószínűsége, ha $y(x_i)$ az egzakt eredmény, mi mégis y_i-t mérünk

$$P(y_1, y_2, \dots) = \prod_{i=1}^{n} \exp\left[-\frac{1}{2}\left(\frac{y_i - y(x_i, a_1, a_2 \dots a_m)}{\sigma}\right)^2\right] \Delta y$$

Mennyi max P? Inkább keresem $\min(-\log P)$ értékét.

$$-\log \prod_{i=1}^{n} \left(\exp[\dots] \Delta y\right) = -\sum_{i=1}^{n} \log\left(\exp[\dots] \Delta y\right) = -\sum_{i=1}^{n} [\dots] + N \log \Delta y$$

$$\min(-\log P) = \min\left(-\sum_{i=1}^{N} \frac{1}{2}\left(\frac{y_i - y(x_i, a_1 \dots a_m)}{\sigma}\right)^2\right)$$

Így a legvalószínűbb érték $\min \sum_{i=1}^{N} \left(y_i - y(x, a_1 \dots a_m)\right)^2$

Ha nem egyforma a Gauss eloszlás szélessége

$$\chi^2 = \sum_{i=1}^{N} \frac{\left(y_i - y(x_i, a_1 \dots a_m)\right)^2}{\sigma_i^2}$$ igazi χ^2

min χ^2-hez tartozik $a_1^*, a_2^* \dots a_m^*$ értékek, ebből egy χ^2_{min} értéket kapunk, ez utal a becslés pontosságára. A legjobb (*) paraméterekhez is van hibája. A χ^2_{min} mindig túl nagy, az jelentheti a rossz hibamodellt, esetleg a rossz modellt.

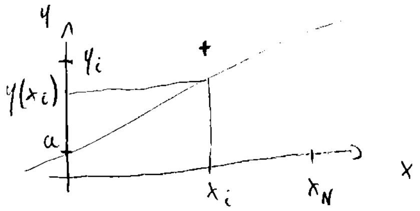
Minimum feltétele:

$$\frac{\partial \chi^2}{\partial a_j} = 0 \quad \forall j = 1 \dots M \qquad \frac{\partial \chi^2}{\partial a_j} = \sum_{i=1}^{N} \frac{2}{\sigma_i^2}\left(y_i - y(x_i, a_1 \dots a_m)\right) \frac{\partial y}{\partial a_j}(-1)$$

Ez általában M egyenletből álló, nemlineáris egyenletrendszer

Egyenes illesztése

$y = a + bx$



$$\chi^2(a,b) = \sum_{i=1}^{N}\left(\frac{y_i - y(x_i, a, b)}{\sigma_i}\right)^2 = \sum_{i=1}^{N}\left(\frac{y_i - a - bx_i}{\sigma_i}\right)^2$$

$$\text{I} \quad 0 = \frac{\partial \chi^2}{\partial a} = -2\sum_{i=1}^{N}\left(\frac{y_i - a - bx_i}{\sigma_i^2}\right)$$

$$\text{II} \quad 0 = \frac{\partial \chi^2}{\partial b} = -2\sum_{i=1}^{N}\left(\frac{y_i - a - bx_i}{\sigma_i^2}\right) x_i$$

$$S := \sum_{i=1}^{N}\frac{1}{\sigma_i^2} \quad S_x := \sum_{i=1}^{N}\frac{x_i}{\sigma_i^2} \quad S_y := \sum_{i=1}^{N}\frac{y_i}{\sigma_i^2} \quad S_{xx} := \sum_{i=1}^{N}\frac{x_i^2}{\sigma_i^2} \quad S_{xy} := \sum_{i=1}^{N}\frac{x_i y_i}{\sigma_i^2}$$

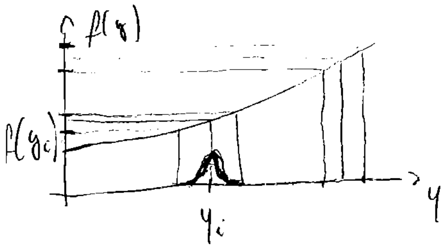
I: $aS + bS_x = S_y$
II: $aS_x + bS_{xx} = S_{xy}$

kétismeretlenes lineáris egyenletrendszer

$$a = \frac{S_{xx}S_y - S_x S_{xy}}{\underbrace{SS_{xx} - S_x^2}_{\Delta}} \qquad b = \frac{SS_{xy} - S_x S_y}{\underbrace{SS_{xx} - S_x^2}_{\Delta \text{ Determináns}}}$$

Hibaterjedés törvénye

ha y_i hibaeloszlása ismert (σ_i ismert), akkor mennyi $f(y_i)$ hibája, vagyis hibaeloszlása?



$$\sigma_f^2 = \sigma^2\left(\frac{\partial f}{\partial y}\right)^2 = \sum_{i=1}^{N}\sigma_i^2\left(\frac{\partial f}{\partial y_i}\right)^2$$

$\vdots$

$$\sigma_a^2 = \frac{S_{xx}}{\Delta} \qquad \sigma_b^2 = \frac{S}{\Delta}$$

Ha x_i-nek is hibája van

$$\chi^2(a,b) = \sum_{i=1}^{N} \frac{(y_i - a - b x_i)^2}{\sigma_{y_i}^2 + b^2 \sigma_{x_i}^2} \longrightarrow \text{nemlineáris egyenletrendszerre vezet } \frac{\partial \chi^2}{\partial a} = \frac{\partial \chi^2}{\partial b} = 0$$

Általános lineáris legkisebb négyzetek.

Mi van, ha parabolát szeretnénk illeszteni? $\chi^2 \sim \sum \left(y_i - a_1 - a_2 x_i - a_3 x_i^2\right)^2$

Ha ennek vizsgáljuk a minimumát a_1, a_2, a_3-ban, mindig lineáris egyenletrendszerre jutok, hiába nemlineáris a próbafüggvény. Sőt, x helyett $\sin x$ is lehetne, x^2 helyett $\ln$, vagy bármi más.

Legyen $X_\ell(x)$ az x-nek tetszőleges (akár nemlineáris) függvénye

Általános alak: $y(x) = \sum_{\ell=1}^{M} a_\ell X_\ell(x)$, ekkor $\chi^2 = \sum_{i=1}^{N} \left(\frac{y_i - \sum_{\ell=1}^{M} a_\ell X_\ell(x)}{\sigma_i^2} \right)^2$ ennek kell

minimumnak lennie $a_1 \dots a_N$-ben, vagyis $\frac{\partial \chi^2}{\partial a_\ell} = 0 \quad \forall \ell \in \left([1, N] \cap \mathbb{N} \right) \Rightarrow$

M ismeretlen

N egyenlet (lineáris)

$$0 = \frac{\partial \chi^2}{\partial a_\ell} = \sum_{i=1}^{N} 2 \cdot \frac{1}{\sigma_i^2} \left[y_i - \sum_{j=1}^{M} a_j X_j(x_i) \right] X_\ell(x_i)$$

$$\alpha_{\ell j} := \sum_{i=1}^{N} \frac{X_j(x_i) \, X_\ell(x_i)}{\sigma_i^2} \qquad \beta_\ell := \sum_{i=1}^{N} \frac{y_i X_\ell(x_i)}{\sigma_i^2}$$

$$0 = \sum_{i=1}^{N} \frac{1}{\sigma_i^2} y_i X_\ell(x_i) - \sum_{j=1}^{M} \sum_{i=1}^{N} \frac{1}{\sigma_i^2} a_j X_j(x_i) X_\ell(x_i) \Rightarrow \sum_{j=1}^{N} \alpha_{\ell j} a_j = \beta_\ell$$

$\underline{\underline{\alpha}} \cdot \underline{a} = \underline{\beta}$ Ez számán megoldható

$$a_j = \sum_{\ell=1}^{N} \alpha^{-1}_{j\ell} \beta_\ell = \sum_{\ell=1}^{M} C_{j\ell} \left[\sum_{i=1}^{N} \frac{y_i X_\ell(x_i)}{\sigma_i^2} \right]$$

$\underline{a} = \underline{\underline{\alpha}}^{-1} \underline{\beta} \qquad C_{j\ell} := \alpha^{-1}_{j\ell}$

$$\sigma^2(a_j) = \sum_{i=1}^{N} \sigma_i^2 \left(\frac{\partial a_j}{\partial y_i} \right)^2 \qquad \frac{\partial a_j}{\partial y_i} = \sum_{\ell=1}^{M} C_{j\ell} \frac{X_\ell(x_i)}{\sigma_i^2}$$

$$\boxed{\sigma^2(a_j)} = \sum_{\ell=1}^{M} \sum_{\ell'=1}^{M} C_{j\ell} C_{j\ell'} \underbrace{\left[\sum_{i=1}^{N} \frac{X_\ell(x_i) \, X_{\ell'}(x_i)}{\sigma_i^2} \right]}_{\alpha_{\ell\ell'} = C_{\ell\ell'}^{-1}} = \boxed{C_{jj}}$$

<u>Megj</u>: lineáris egyenletrendszert kell megoldani, erre jó a Gauss elimináció, ami az inverzet is megadja (a hibához). Ha nagyon sok a paraméter, más algoritmusok is előfordulnak: SVD (szinguláris dekompozíció).

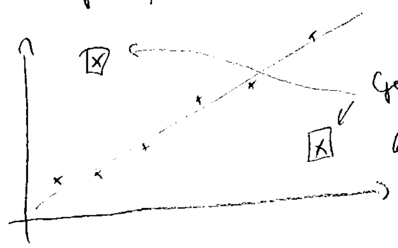
Olyan nemlineárisnak tűnő problémák valójában lineárisak. pl: $y(t) = a e^{-bx}$ az olyan, mintha nemlineáris volna. $z := \log(y(t))$, $c := \log a \Rightarrow z(t) = c - bx$, egyenesillesztési probléma lett. A hibák eloszlása már más lesz! (Gauss → nem Gauss)

Több dimenzióra is megy: $\chi^2 = \sum_{i=1}^{N} \left(\frac{\overset{\text{mérési pontok}}{y_i} - \sum_{k=1} a_k X_k(\underline{x}_i)}{\sigma_i} \right)^2$. Ha a paraméterek ténylegesen nemlineárisan szerepelnek:

χ^2 =
- nemlineáris optimalizáció (minimumkeresés) Levenberg-Marquardt, ez a gnuplotban benne van
- $\frac{\partial}{\partial a_j}$ → nemlineáris egyenletrendszer

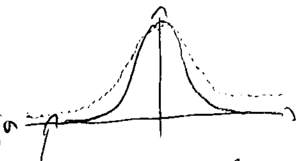
Robusztus becslés

Kiugró pontok



Gauss-i hibamodell problémája

ki kell dobni

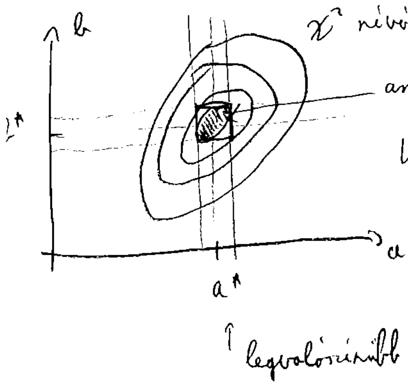


polinomiális lecsengés

$$P \sim \prod_{i=1}^{N} \exp\left(-\rho\left(\frac{y_i, y(t_i, a_1 \ldots a_M)}{\sigma_i} \right) \right) \Rightarrow \sum_{i=1}^{N} \rho\left(\frac{y_i - y(t_i)}{\sigma_i} \right)$$

pl: $\sum_{i=1}^{N} \left| y_i - y(t_i) \right| \Leftrightarrow \chi^2$ Ebbe nem szólnak bele annyira az outlierek

konvergencia határok, kontúrvonalak



χ^2 nívók

annyit mondhatunk, hogy pl 68,2% a valószínűsége, hogy ide esik a 2 változó. Azért nem a téglalap, mert a és b hibái korreláltak. De mi is kijelölhetünk területet, és arra valószínűséget mondhatunk.

Programozás

Felhasználó: B = inv(A), gépi: ADD 1,3
JUMP 137
a kettő között a programozó munkája van,

a magasszintű programozási nyelvek:

elég elterjedt: C, C++, Fortran ← matematikai célokra

JAVA, PASCAL, BASIC ← alapszintű

C#, PYTHON

PERL ← script nyelv, oprendszereket lehet bántani ezzel

webalkalmazásokra remek

C++-ban írják: OS, irodai szoftver, játékok. Objektum orientált (az mi?)

standard input output

```
# include <stdio.h>

main() {   ← OS itt kezdi futtatni
           ← ezt hajtja végre
  fprint(stdout, "Hello World!\n");

}
```

float v[3] lefoglal 3 memóriaegységet, floatnál 4 byte

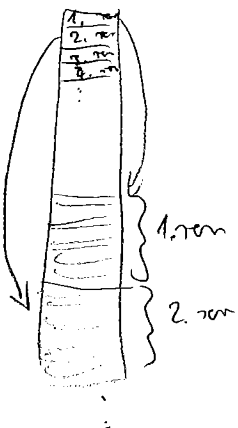
De itt meg kell mondani, mekkora lesz

float *v ezzel v létezik, de nincs mérete, értéke

```
v = (float *)calloc(3, sizeof(float));
```

táblázatot definiálni: v[1][2] ⟺ v[1·3+2]

másik lehetőség



MSYS
CYGWIN } LINUXOT emulálnak

VIRTUÁLIS GÉP
↳ VIRTUAL PC
↳ VMWARE

BEJELENTKEZÉS
↳ SSH → PUTTY
↳ WIN→WIN REMOTE DESKTOP
↳ VNC LIN→WIN
NX, NOMACHINE

```
# include <stdlib.h>
```

fscanf (stdin, "format", &(beolvasandó változók címe))

fileból olvas formázottan — file* mutató — pl %g

%: formázó karakter

g: general

c: character

d: decimal (egész)

f: float

lf: double

miért kell & ?

int i; változó

i = 3

float *v   v egy mutató, pointer

float x   x egy érték (float típusú)

$v = \boxed{\&x}$

pointer — érték

*v = 3.6 egy jó

03.08.

Lineáris egyenletrendszer

$$a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n = b_1$$
$$a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n = b_2$$
$$\vdots$$
$$a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n = b_m$$

$$\underline{\underline{A}}\ \underline{x} = \underline{b}$$

M=N esetén jó esély az egyenletek megoldására

Problémák: • a mátrix szinguláris, nincs megoldás, ennyi
- • létezik megoldás, de numerikusan közel szinguláris
  - mert szinguláris lesz, pl 0-val osztana
  - vagy van megoldás, de hibás, ekkor ellenőrizni kell behelyettesítéssel

Olyan algoritmus kell, ami gyors és kicsi. Hol jelentenek gondot?

N ~ 10 semmi

N ~ 100 double kell (ma már olcsó)

N ~ 1000 időbaj, könnyen numerikus szinguláris lesz

Programozó magok 03.08

adott $\underline{\underline{A}}$-ra sok $\underline{x}$ és $\underline{b}$ kell $\underline{\underline{A}}\,\underline{x}_1 = \underline{b}_1$ ... gyorsabb algoritmus is van, mint a szimmel végigmenni

$\underline{\underline{A}}^{-1}$ meghatározására elégséges, de nem szükséges

$\det \underline{\underline{A}}$

Ha $M<N$ vagy degenerált, megoldás alteret kereshetünk (SVD szinguláris dekompozíció)

$M>N$, nincs megoldás, de kereshetünk olyan pontot, mely jól kielégíti az egyenletet

Spec. mátrixra ügyesebb algoritmusok van

LAPACK ⇐ LINPACK ⇐ BLAS (fortran, C++)

IMSL, NAG

SLATEC, GSL
↑
Gnu sci lib

GAMS

Gauss-Jordan elimináció

- lin. egyenlet megoldása
- mátrixinvertálás

A megoldás nem változik, ha
- két sort felcserélek
- két oszlopot felcserélek
- lineáris kombinációt alkalmazok

$$\underline{\underline{A}}\,\underline{x} = \underline{b} \longrightarrow \underline{\underline{A}}'\,\underline{x} = \underline{b}' \xrightarrow{\dots} \underline{\underline{A}}''\,\underline{x} = \underline{b}'' \qquad \underline{\underline{A}}'' = \underline{\underline{I}}$$

03.15

Lépések:

1: 1. sor / a_{11}

2: 2-től N-edik sorig i. sor $- a_{i1}$ 1. sor

Műveletigény $\sim N^3$, $= cN^3$

Vízszintes helyettesítéses Gauss-elimináció

a/ mátrix átalakítása

b/ megoldás (visszahelyettesítés)

a/: mátrix háromszögesítése $\begin{bmatrix} \diagdown & \\ 0 & \diagdown \end{bmatrix}$ csak a pivot sor alatt nullázzuk ki az oszlopot

I' $r_{il} := a_{il}^{(l)} / a_{ll}^{(l)}$

II' $\begin{cases} a_{ij}^{(k+1)} := a_{ij}^{(l)} - r_{il}\, a_{lj}^{(l)} & \forall j = l+1 \dots N \\ b_i^{(k+1)} := b_i^{(l)} - r_{il}\, b_l^{(l)} & \text{Műveletigény kb } \frac{1}{3} c N^3 \end{cases}$

b: $a_{11}' x_1 + a_{12}' x_2 + \dots + a_{1N}' x_n = b_1'$

$0 + a_{22}' x_2 + \dots + a_{2N}' x_n = b_2'$

$\vdots$

$a_{N-1,N-1}' x_{n-1,n-1} + a_{N-1,N} x_N = b_{n-1}'$

$a_{NN}' x_n = b_n' \Rightarrow x_n = b_n'/a_{nn}'$

$x_{n-1} = \frac{1}{a_{N-1,N-1}} \left(b_{N-1}' - x_N a_{N-1,N}' \right)$

Műveletigény dN^2

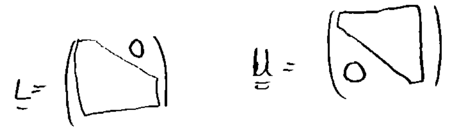
összes műveletigény $\frac{1}{3} c N^3 + dN^2 \sim \frac{c}{3} N^3$

Előfordulhat, hogy a pivot elem 0

- sor vagy oszlopcserével megoldható a probléma
- érdemes olyan sort betenni, melynek a négyzetösszege minél nagyobb
- stabilabb, pontosabb eredmény

Optimálisabb megoldás akkor, ha $\underline{\underline{A}}\,\underline{x}^{(l)} = \underline{b}^{(l)}$, $l \in I$ több feladatunk van $\underline{y}$

Mat. tétel: $\forall$ mátrix felírható úgy: $\underline{\underline{A}} = \underline{\underline{L}} \cdot \underline{\underline{U}}$, akkor $\underline{\underline{A}}\,\underline{x} = \underline{\underline{L}}\left(\underline{\underline{U}}(\underline{x})\right) = \underline{b}$



$$y_1 = \frac{b_1}{l_{11}} \qquad y_i = \frac{1}{l_{ii}}\left[b_i - \sum_{j=1}^{i-1} l_{ij} y_j\right] \leftarrow \underline{\underline{L}}\,\underline{y} = \underline{b}$$

$$\underline{\underline{U}}(\underline{x}) = \underline{y}$$

2 db lineáris egyenletrendszer

$$x_N = \frac{y_N}{u_{NN}}$$

$$x_i = \frac{1}{u_{ii}}\left[y_i - \sum_{j=1}^{N} u_{ij} x_j\right]$$

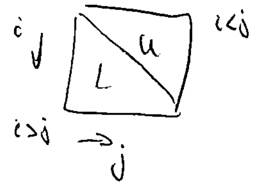
Már csak a mátrix felbontás kell

$$\sum_k l_{ik} u_{kj} = a_{ij}$$

$i<j$ $\quad l_{i1}u_{1j} + l_{i2}u_{2j} + \dots + l_{ii}u_{ij} = a_{ij}$

$i>j$ $\quad l_{i1}u_{1j} + l_{i2}u_{2j} + \dots + l_{ij}u_{jj} = a_{ij}$

$i=j$ $\quad l_{i1}u_{1j} + l_{i2}u_{2j} + \dots + l_{ii}u_{jj} = a_{ij}$



Crout algoritmus

Ugyanannyi művelet mint a hagyományos módszernél a Δ alakúra hozás

Ha $\underline{\underline{A}}$ szimmetrikus, $\underline{\underline{U}} = \underline{\underline{L}}^T$ és fele annyi művelet kell ab,

ha sávmátrix $M^2 N$ ahol M a sávszélesség

WANDERMONDE $\begin{pmatrix} 1 & \alpha_1 & \alpha_1^2 \dots & \alpha_1^{N-1} \\ 1 & \alpha_2 & \alpha_2^2 \dots & \alpha_2^{N-1} \\ \vdots & \vdots & & \\ 1 & \alpha_N & \alpha_N^2 \dots & \alpha_N^{N-1} \end{pmatrix}$ megoldásra $\sim N^2$

```
j = 1,2,...,N {
    i = 1,2,...,j {
        u_ij = a_ij - Σ_{k=1}^{i-1} l_ik u_kj
    }
    i = 1,2,...,N {
        l_ij = (1/u_jj) a_ij - Σ_{k=1}^{j-1} l_ik u_kj
    }
}
```

$$\det \underline{\underline{A}} = \prod_{j=1}^{N} a'_{jj} = \prod_{j=1}^{N} a_{jj}$$

03. 22.

$$a_{jj} = \varepsilon_j = 10^{\alpha_j}$$

$$\prod_{j=1}^{N} \varepsilon_j \cdot 10^{\left(\sum_{j=1}^{N} \alpha_j\right)} = \prod_{j=1}^{N} a_{jj}$$

Speciális módszerek

ritka mátrixok - leghíresebb részgyakorlat iterációk (ritka mátrix: nem 0 elemek száma $\sim n$)

$\underline{\underline{A}}\,\underline{x} = \underline{b}$ helyett $\min\limits_{\underline{x}} \left| \underline{\underline{A}}\,\underline{x} - \underline{b} \right|^2$

lineáris egyenletrendszer helyett numerikus optimalizálás

pl. konjugált gradiens módszer

$\underline{\underline{A}}\,\underline{x}$ ritka mátrixok esetén nem n^2 műveletek kell, inverz mátrix meghatározható

szinguláris dekompozíció

SVD: singular value decomposition

$$\underset{M \times N}{\underline{\underline{A}}} = \underset{M \times N}{\underline{\underline{U}}} \cdot \begin{pmatrix} w_1 & & 0 \\ & w_2 & \\ 0 & & \ddots \\ & & & w_n \end{pmatrix} \underline{\underline{V}}^T$$

$\underline{\underline{U}}$: oszlop ortogonális

$\underline{\underline{V}}^T$: $N \times N$ ortogonális

áll: minden mátrix (szinguláris is) felbontható ilyenek szorzatára

NUM. REC. tárgyalja, LAPACK-ban van 2 módszer is rá.

$$\underline{\underline{A}}^{-1} = \underline{\underline{V}} \begin{pmatrix} 1/w_1 & & \\ & 1/w_2 & \\ & & \ddots \\ & & & 1/w_n \end{pmatrix} \underline{\underline{U}}^T$$

probléma ott jöhet be, hogy $1/w_i \to \infty$ mert $w_i \approx 0$. A módszer kibírja, ha helyére $\frac{1}{w_i} = 0$-t írunk. (adat törlése)

$\frac{w_i}{w_{max}} \approx$ gépi pontosság, akkor törölhető

Iteratív módszerek

$\underline{\underline{A}}\,\underline{x} = \underline{b}$

$\Downarrow$

$(\underline{\underline{I}} - \underline{\underline{A}})\underline{x} = \underline{x} - \underline{b}$

$\underline{x} = (\underline{\underline{I}} - \underline{\underline{A}})\underline{x} + \underline{b}$

$\underline{x}_{k+1} = (\underline{\underline{I}} - \underline{\underline{A}})\underline{x}_k + \underline{b}$ $\quad x_k$ - k sorozata remélhetőleg konvergens, legfőképp ha $\|\underline{\underline{I}} - \underline{\underline{A}}\| < 1$

iteráció + dekompozíció : Gauss - Seidel

Gyökök pontosítása iterációval

$\underline{\underline{A}}(\underline{x} + \delta\underline{x}) = \underline{b} + \delta'\underline{b}$

$\Downarrow$

$\boxed{\underline{\underline{A}}\,\delta\underline{x}} = \delta'\underline{b} = \boxed{\underline{\underline{A}}\underbrace{(\underline{x} + \delta\underline{x})}_{x_1} - \underline{b}}$ $\quad$ egy újabb egyenletrendszer

$\underline{x}_2 = \underline{x}_1 - \delta\underline{x}_1$

LU dekompozíció segítségével a javító lépések N^2 műveletet igényelnek

① $\underline{\underline{A}}\,\underline{x}$ elvégzése

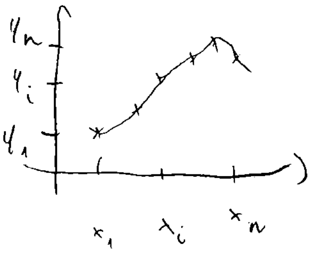
② Gauss-elimináció sor és oszlopcserével

③ N változós polinom illesztés (általános lineáris legkisebb négyzetek módszere)

$f(x) = \sum_i a_i x^{i-1}$ $\qquad y(x_1, x_2, x_3) = a_0 + a_1x_1 + a_2x_2 + a_3x_3 + a_4x_1x_2 + a_5x_2x_3 + a_6x_1x_3 +$

$+ a_7x_1^2 + a_8x_2^2 + a_9x_3^2$

interpoláció és extrapoláció 1D-ben



(fv illesztés : modell, paraméterek)

interpoláció : $\forall$ ponton megy át az interpolált fv.

nem fontos a pontos függvényalak.

Arra jó ez, h. köztes pontokat számítunk ki, „átmintavételezés".

pl: galaxis spektrumok, diszkrét hullámhosszakra tudjuk a fényességet. Ha egy másik galaxissal össze akarom vetni, baj lehet, mert vissza-vöröseltolni kell, így nem ugyanott lettek mintavételezve.

pl: fényképek átméretezése

interpoláció, ha $x_1 < x < x_n$, extrapoláció ha $x < x_1$ vagy $x_n < x$

vannak jól és rosszul interpolálható függvények, n pontnál általában n szabad paramétere van, vagyis n-1-edfokú polinom illeszthető

kétfajta interpoláció van : globális és lokális ↖ csak néhány pontra

↑ egész pontsorozatra adunk interpolációs függvényt

kétmódszer van: - kiszámoljuk a polinom együtthatóit, majd x_i-ket behelyettesítjük

- y_i-kből határozzuk y-t (gyorsabb, becslést ad a hibára)

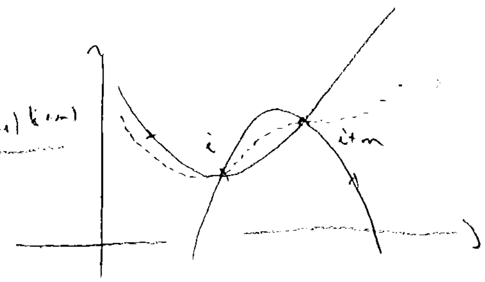
$$y = P(x) = \frac{(x-x_2)(x-x_3)\dots(x-x_n)}{(x_1-x_2)(x_1-x_3)\dots(x_1-x_n)} y_1 + \frac{(x-x_1)(x-x_3)(x-x_4)\dots(x-x_n)}{(x_2-x_1)(x_2-x_3)\dots(x_2-x_n)} y_2 + \dots + \frac{(x-x_1)(x-x_2)\dots(x-x_{N-1})}{(x_N-x_1)(x_N-x_2)\dots(x_N-x_{N-1})} y_N$$

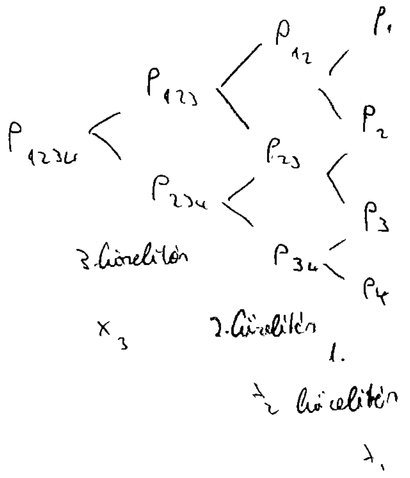
Neville algoritmus

rekurzió a Lagrange-függvény előállítására

$$P_{i(i+1)\dots(i+m)} = \frac{(x-x_{i+m})\, P_{i(i+1)\dots(i+m-1)} + (x_i - x)\, P_{(i+1)\dots(i+m)}}{x_i - x_{i+m}}$$

↑
polinom, mely átmegy i, (i+1), (i+2), ... pontokon





$$C_{mi} := P_{i(i+1)\dots(i+m)} - P_{i(i+1)\dots(i+m-1)}$$

$$D_{mi} := P_{i(i+1)\dots(i+m)} - P_{(i+1)(i+2)\dots(i+m)}$$

$$C_{m+1,i} = \frac{(x_i - x)(C_{m,i+1} - D_{m,i})}{x_i - x_{i+m+1}}$$

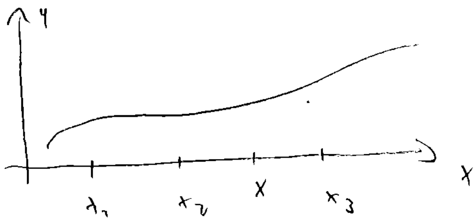
$$D_{m+1,i} = \frac{(x_{i+m+1} - x)(C_{m,i+1} - D_{m,i})}{x_i - x_{i+m+1}}$$

x_k konvergál.

03.30.

<u>polinom interpoláció</u>

$$y_i = c_0 + c_1 x_i + c_2 x_i^2 + c_3 x_i^3 + \dots + c_n x_i^n$$



A/ Lagrange-formula

$$y_i(x_i = 0) = c_0$$

vonjuk ki c_0-t mindkét oldalból, osszuk x_i-vel

B/ lineáris egyenletrendszer megoldás

$$\underline{\underline{A}}\,\underline{c} = \underline{y} \qquad \underline{\underline{A}} = \begin{pmatrix} 1 & x_1 & x_1^2 & \dots & x_1^N \\ 1 & x_2 & x_2^2 & \dots & x_2^N \\ \vdots & & & & \\ 1 & x_N & x_N^2 & \dots & x_N^N \end{pmatrix} \qquad T \sim N^2$$

VANDERMONDE

Polinommal nem jól interpolálható függvények

ekkor alkalmazhatunk: interpolációs racionális tö. függvényekkel

$$R_m(x) = \frac{P_n(x)}{Q_\nu(x)} = \frac{p_0 + p_1 x + \dots p_n x^n}{q_0 + q_1 x + \dots q_\nu x^\nu}$$

$m = n + \nu$

. $n + \nu + 1$ szabad paraméter

- nem jó ha divergál (pl $1/x$ a 0-ban)
- nem divergál, de komplex síkra kiterjesztve pólusa van
- egyéb

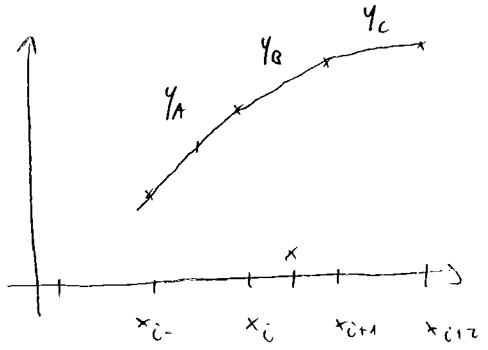
Eddig volt globális interpoláció
Most lesz lokális interpoláció (szakaszonkénti)

lineáris interpoláció $\underset{\overset{\shortparallel}{x}}{\xi} \in (x_i, x_{i+1})$, $f(\xi) = A y_i + B y_{i+1}$

$$A = \frac{x_{j+1} - \xi}{x_{j+1} - x_j}$$

$$B = 1 - A$$

Köbös interpoláció



$$\left.\begin{aligned} y_A(x_i) &= y_i \\ y_B(x_i) &= y_A x_i \end{aligned}\right\} \quad \mathrm{I}$$

$$\frac{\partial y_A}{\partial x}(x_i) = \frac{\partial y_B}{\partial x}(x_i) \quad \mathrm{II}$$

$$\frac{\partial^2 y_A}{\partial x^2}(x_i) = \frac{\partial^2 y_B}{\partial x^2}(x_i) \quad \mathrm{III}$$

$$y = A y_i + B y_{i+1} + C y_j'' + D y_{j+1}'' \Rightarrow \text{egyenletrendszer}$$

Ha adott lenne ∀ pontban y_j''

$\frac{0\,330}{3}$

$$A = \frac{x_{j+1} - x}{x_{j+1} - x_j} \qquad B = \frac{x - x_j}{x_{j+1} - x_j}$$

$$\left.\begin{aligned} C &= \frac{1}{6}(A^3 - A)(x_{j+1} - x_j)^2 \\ D &= \frac{1}{6}(B^3 - B)(x_{j+1} - x_j)^2 \end{aligned}\right\} \text{ekkor teljesül I, IV.}$$

II. egyenletrendszer:
$$\frac{x_j - x_{j-1}}{6} y_{j-1}'' + \frac{x_{j+1} - x_{j-1}}{3} y_i'' + \frac{x_{j+1} - x_j}{6} y_{j+1}'' = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{y_j - y_{j-1}}{x_j - x_{j-1}}$$

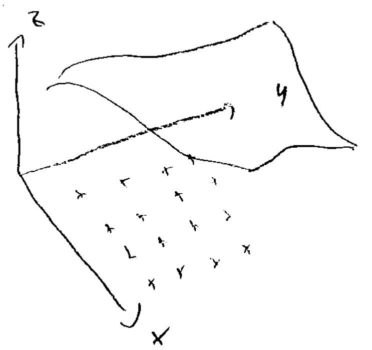
lin. egyenletrendszer y_i''-re

N-2 db egyenlet, 2 érték szabadon választható,

pl $y_1'' = 0$, $y_N'' = 0$ vagy $y_1' = \alpha$, $y_N' = \beta$

$$\begin{pmatrix} \ddots & & 0 \\ & \ddots & \\ 0 & & \ddots \end{pmatrix} \begin{pmatrix} y'' \end{pmatrix}$$

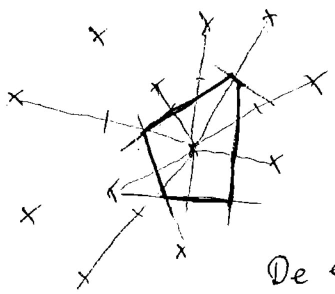
$T \sim 5N$



Több dimenzióra is van általánosítás

Egyik eset, ha rácspontban vannak a pontok

Másik, ha nem. Ekkor nem triviális a szomszédság



Cellázás

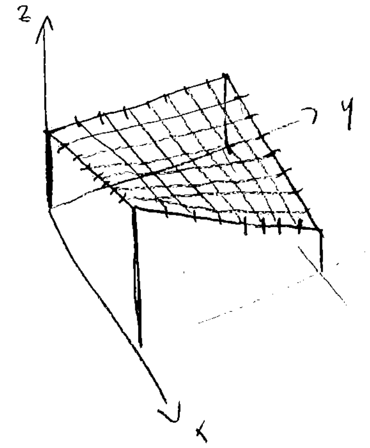
két elem szomszédos,

ha van közös oldaluk

De ez túl bonyolult, vissza a rácspontokra

Bilineáris interpoláció

Baj: 4 pontra sík nem illeszthető



$$z = (1-t)(1-u)\, z_{i,j} + t(1-u)\, z_{i+1,j} +$$
$$+ (1-t)\, u\, z_{i,j+1} + t u\, z_{i+1,j+1}$$

Közönséges differenciálegyenletek numerikus megoldása

04.06

$$\frac{dy}{dx} = f(x,y) \longleftarrow \lim_{\Delta x \to 0} \frac{\Delta y(x)}{\Delta x}$$

függvény egyenlet,

melyet $y(x)$ teljesíti,

aminek a deriváltja $f(x, y(x))$

$$\frac{y_{x+1} - y_x}{\Delta x} = f(x,y)$$

$$y_{x+1} = y_x + \Delta x\, f(x,y)$$

később időpont — megkapható a korábbiból

Vannak magasabbrendűek is: $\frac{d^2y}{dx^2} + q(x)\frac{dy}{dx} = r(x)$ Minden magasabbrendű visszavezethető csatolt elsőrendűekre

$$\frac{dy}{dx} := z(x)$$

tiszta algebrai (deriv. nélküli) kifejezések

nem mindig a deriváltat jó elnevezni z-nek

$$\frac{dz}{dx} = r(x) - q(x) z(x)$$

Általában olyan köz. diff. egy. rendszere

$$\frac{dy_l(x)}{dx} = f_l(x, y_1, y_2 \dots y_k) \quad l \in [1, k] \cap \mathbb{N}$$

ezt kell együttesen megoldani, numerikusan léptetni

azért nem egzaktul oldjuk meg, mert az nem mindig állítható meg

A megoldás nem egyértelmű, peremfeltételek kellenek (kezdeti feltételek), ha az nem kezdeti, akkor iterációs megoldás szükséges.

Léptetéses módszer, Euler. módszer 1 DB egyenlet, elsőrendű, kezdetiérték. probléma

$$y_{n+1} = y_n + h f(x_n, y_n)$$

h: lépés, kicsi, de nem 0

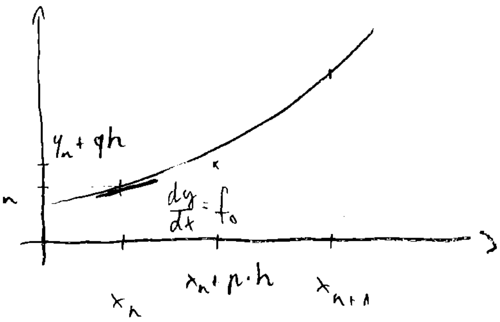
$$x_{n+1} \equiv x_n + h$$
$$y_{n+1} \equiv y(x_n + h)$$
$$y_n \equiv y(x_n)$$

$$y(x+h) = y(x) + \frac{\frac{dy}{dx}}{1!} h + \frac{\frac{d^2y}{dx^2}}{2!} h^2 + \dots$$

$$\frac{dy}{dx} = f(x, y)$$

$$y_{n+1} = y_n + h f(x, y) + O(h^2)$$

halmozódnak a hibák, túl gyorsan egyszerű, de túl sok hiba



Runge-Kutta módszerek

$O(h^3)$ vagy $O(h^4)$ hibát szeretnék

$$y(x+h) = y(x) + A \cdot h f_0 + B h \cdot f_1$$

$$f_0 = f(x_n, y_n)$$
$$f_1 = f(x_n + ph, y_n + qh)$$

x_n, y_n helyen; $f(x_n, y_n) = f_0$

$$f_1 = f(x_n, y_n) + \frac{\partial f}{\partial x}(x_n, y_n) p \cdot h + \frac{\partial f}{\partial y} \frac{dy}{dx} q \cdot h + O(h^2)$$

$$y(x_n + h) = y(x_n) + (A+B) h f(x_n, y_n) + B h^2 p \frac{\partial f}{\partial x}(x_n, y_n) + B h^2 q \frac{\partial f}{\partial y}(x_n, y_n) f(x_n, y_n) + O(h^3)$$

$$y(x_n+h) = y(x_n) + h\, \underset{\substack{\parallel \\ \frac{dy}{dx}}}{f}(x_n, y_n) + \frac{h^2}{2} y''(x) + O(h^3)$$

$$y''(x) = \frac{d^2y}{dx^2} = \left(\frac{dy}{dx}\right)' = f'(x,y) = \frac{\partial f}{\partial x}(x,y) + \frac{\partial f(x,y)}{\partial y}\frac{dy}{dx}$$

$$y(x_n+h) = y(x_n) + h\, f(x_n, y_n) + \frac{h^2}{2}\left(\frac{\partial f}{\partial x}(x_n, y_n) + \frac{\partial f}{\partial y}(x_n, y_n)\, f(x_n, y_n)\right) + O(h^3)$$

$\Downarrow$ $\quad$ $\Downarrow$ $\quad$ $\Downarrow$

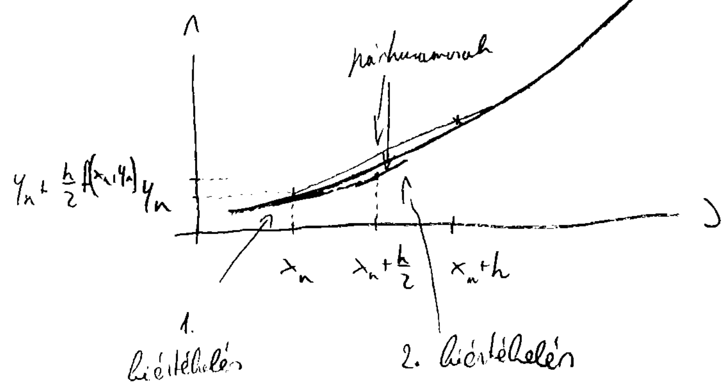
$A+B=1$ $\quad$ $B\cdot P = \frac{1}{2}$ $\quad$ $B\cdot Q = \frac{1}{2}$ $\quad$ A, B, P, Q 4 változó, 3 egyenlet

Pl: 2-odrendű R-K

$A=0 \quad B=1$

$P=Q=1/2$

$$y(x_n+h) = y(x_n) + h\, \overset{②}{f}\Big(x+\frac{h}{2},\ y + \overbrace{\frac{h}{2} f(x_n, y_n)}^{①}\Big)$$



Másik fajta $\quad y(x_n+h) = y(x_n) + \frac{h}{2}\, f(x,y) + f(x+h,\ y+h\, f(x,y)))$

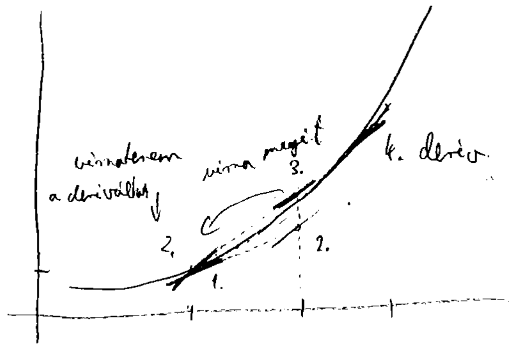
4. odrendű R-K

$k_1 = h\, f(x_n, y_n)$ $\qquad$ 1.

$k_2 = h\, f(x_n + 0{,}5h,\ y_n + 0{,}5 k_1)$ $\qquad$ 2.

$k_3 = h\, f(x_n + 0{,}5h,\ y_n + 0{,}5 k_2)$ $\qquad$ 3.

$k_4 = h\, f(x_n + h,\ y_n + k_3)$ $\qquad$ 4. függvény kiértékelés



$$y_{n+1} = y_n + \frac{k_1}{6} + \frac{k_2}{3} + \frac{k_3}{3} + \frac{k_4}{6} + O(h^5)$$

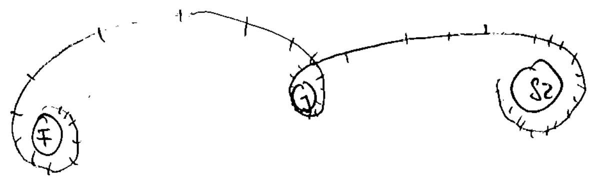
Lehetne tovább növelni, de nem biztos, hogy jó, lehet hogy kisebb lépés célravezető

Adaptív lépésköz választás

$y_n \xrightarrow{\text{léptetés}} y_{n+1}$ egzakt

$y_n \longrightarrow y_{n+1}^{*} + \Delta$

számított — egzakttól való eltérés



Szeretnénk, ha épp csak annyit számolnánk, hogy Δ_0 állandó legyen. Ha van 1 nagy hibánk, nem lesz kevesebb, ha későbbi Δ-k kicsik.

Δ „mérése"

$\Delta_1 \sim |y_1 - y_2|$ ← 1 db h-val lépve

mért hiba — két db $\frac{h}{2}$-vel lépten

$$\left|\frac{h_0}{h}\right|^5 = \left|\frac{\Delta_0}{\Delta_1}\right|$$

elvárt hibához szükséges lépés $\quad h_0 = h\left(\frac{\Delta_0}{\Delta_1}\right)^{1/5}$

lépés, két kis lépés $\Rightarrow \Delta_1$, Δ_0 adott eleve is $\Rightarrow h_0$

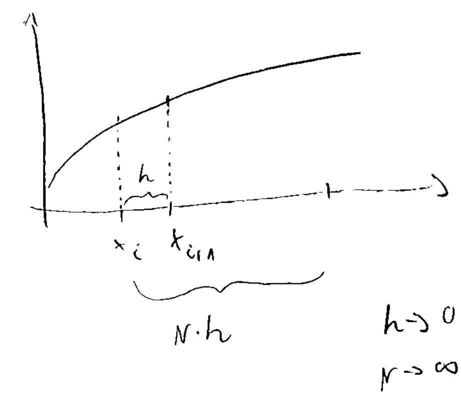
ha $h_0 > h$, léphetünk nagyobbat, h_0-lal a következőt

ha $h_0 < h$, stornó, újra lépünk h_0-lal

11 kr. kiértékelés 8 helyett

04. 20

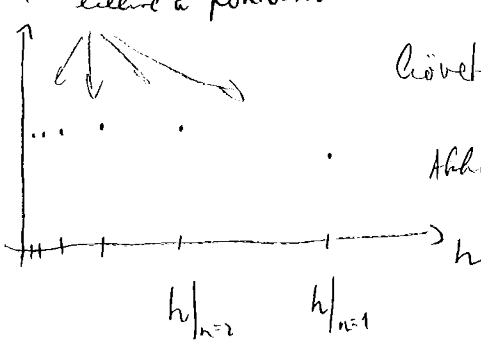
Richardson extrapoláció



$h \to 0$

$N \to \infty$

y_i ezekre a pontokra illesztünk görbét, véges sok h értékből



következtetünk a nagyon kicsire

Akkor jó, ha az integrálandó dif. egyenlet „simán, jól" viselkedik (azaz ha az extrapolált függvény hibája kicsi

Bulirsch-Stoer módszer

(Richardson extrap + mod. középponti módszer)

$z_0 = y(x)$

$z_1 = z_0 + h\, f(x, z_0)$

$\vdots$

$z_{m+1} = z_{m-1} + 2h\, f(x + mh, z_m)$

csak 1 fv. kiértékelés kell hozzá

$y(x+H) = y_m = \frac{1}{2}\left(z_n + z_{n-1} + h f\left(x+H, z_n\right)\right)$

hibája 3 odrendű, mint a R-K és csak páros hibákat tartalmaz

$y(x+H) = y_n$ ← az „eredeti"

$y(x+H) = \frac{4 y_n - y_{n/2}}{3}$

ezzel kiesik a 3. odrendű hiba, így 4-edrendig pontos, 5. ödrendű hibákat tartalmaz

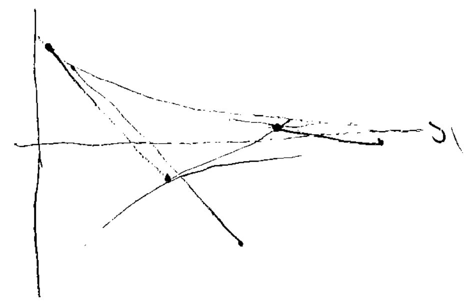
Más fajta módszer, az implicit módszerek

$y' = -cy \Rightarrow y = e^{-cx}$

Mit csinálna a gép Euler-módszerrel: $y_{n+1} = y_n + h y_n' = (1 - ch)\, y_n$

ekkor ugyan konvergál $\frac{2}{c} > h > \frac{1}{c}$

nem konvergál $h \geq \frac{2}{c}$



példa az elcsúszásra

$u' = 998u + 1998v$

$v' = -999u - 1999v$

helyettesítés: $u = 2y - z$

$v = -y + z$

$2y' - z' = u' = 1996y - 998z - 1998y + 1998z = 1000z - 2y$

$-y' + z' = v' = -1998y + 999z + 1999y - 1999z = y - 1000z$

$$\left.\begin{array}{ll} y' = -y & y = e^{-x} \\ z' = -1000z & z = e^{-1000x} \end{array}\right\} \quad \begin{array}{l} u = 2e^{-x} - e^{-1000x} \\ v = -e^{-x} + e^{-1000x} \end{array}$$

kiesik

hiába elhanyagolható $e^{-1000x} \ll e^{-x}$, mégis a lépésközben figyelembe kell venni

Tehát az Euler módszer explicit volt, $y_{n+1} = y_n + h y_n'$

Az implicit módszer más: veszi a lépéseket „visszafelé": $y_{n+1} = y_n + h y_{n+1}'$

ha $y' = -cy$, akkor most $y_{n+1} = y_n + h(-c) y_{n+1}$

akkor egy → $y_{n+1} = \dfrac{y_n}{1+ch}$ ez minden h-ra stabil

lineáris egyenletrendszert

kell megoldani

Ha a dif. egyenletrendszer nem lineáris, akkor nemlineáris egyenletrendszert kapunk, ezt csak iterációval lehet akkor.

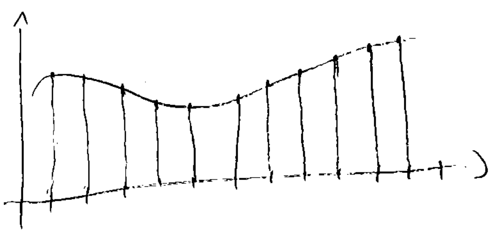
Ez eddig az Euler-módszer implicit továbbfejlesztése volt. ∃ a R-K-nak is implicit változata

Numerikus integrálás

$$I = \int_a^b f(x)\,dx$$

A baj: deriválni könnyű, integrálni nehéz. Itt nincsen sincs meg kompozíciófüggvény integrálási szabálya.

Módszerek 1D-ra



1. függvényértékek súlyozott összege
2. interpoláció a pontok között
3. Függvénybázis szerinti kifejtés (az integrálás lineáris), a báziselemek legyenek integrálhatók

$$\int_a^b f(x)\,dx = \int_a^b \sum_{i=0}^N c_i \varphi_i(x)\,dx = \sum_{i=0}^N \int_a^b c_i \varphi_i(x)\,dx = \sum_{i=0}^N c_i \Big[F_i(x)\Big]_a^b$$

ügyes gondolat, hogy olyan bázist válasszunk, hogy

F_i = függvénye $(F_{i-1}, F_{i-2} \dots F_1)$

vannak zárt és nyitott formulák (integrálható, nem integrálható)

04.20
4

zárt Newton-Cotes formulák

trapéz szabály $\int_{x_1}^{x_2} f(x)dx = h\left[\frac{1}{2} f_1 + \frac{1}{2} f_2\right] + O(h^3 f'')$ ez egy lineáris illesztés

elsőrendű polinomokig egzakt

megoldást nyújt ($|x|$ nem elsőrendű)

lehetőség van 2-odrendű görbeillesztésre

ha a derivált értéke nagy, a hiba is nagy lehet, hiába kicsi h

$$\int_{x_1}^{x_2} f(x)\,dx = h\left[\frac{1}{3} f_1 + \frac{4}{3} f_2 + \frac{1}{3} f_3\right] + O(h^5 f^{(4)})$$

de jó! 5-ödrendű, nem pedig $4 = 3+1$-edrendű

Együtthatók indoklása

trapéz módszernél: $\int_{x_1}^{x_2} f(x)dx = h(\alpha f_1 + \beta f_2)$

tudjuk, hogy elsőrendűre egzakt: $\int_{x_1}^{x_2}(ax+b)\,dx = a\int_{x_1}^{x_2} x\,dx + b\int_{x_1}^{x_2} 1\cdot dx$

$f(x) = 1$

$f(x) = x$

$x_2 - x_1 = h(\alpha + \beta) = \beta = 1 - \alpha$ $\quad f(x) = 1$

$\frac{x_2^2}{2} - \frac{x_1^2}{2} = h(\alpha x_1 + \beta x_2) \Rightarrow x_2 - x_1 = h$

$x_2^2 - x_1^2 = (x_2 - x_1)(x_2 + x_1) \Rightarrow \frac{x_2}{2} + \frac{x_1}{2} = \alpha x_1 + (1-\alpha) x_2$

$\alpha = \frac{1}{2} \quad \beta = \frac{1}{2}$

2-odrendűnél (Simpson formula) hasonlóképp

felösszegzési szabály: $\int_{x_0}^{x_{N-1}} f(x)\,dx = h\left(\frac{1}{2} f_0 + f_1 + f_2 + \ldots + f_{N-2} + f_{N-1}\right) + O\frac{(b-a)^3 f^{(2)}}{N^2}$

integrál közelítései

04.20
5

$$\left.\begin{aligned} I_1 = s_1 &= (b-a)\left(\frac{1}{2} f(a) + \frac{1}{2} f(b)\right) \\ s_2 &= \frac{b-a}{2} f\left(\frac{a+b}{2}\right) \end{aligned}\right\} \quad I_2 = \frac{1}{2} I_1 + s_2$$

$I_n - I_{n-1}$ hova tart ?

Hibatagok kiejtése felezéses során : $I = \frac{4}{3} I_{2N} - \frac{1}{3} I_N$ hiba $\frac{1}{N^4}$ -en

u.a mint a felezett Simpson

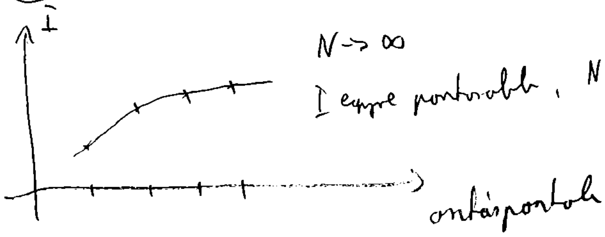
04.27.

Zárt Newton-Cotes formulák

→ felezett trapéz szabály

$$I = \frac{4}{3} I_{2N} - \frac{1}{3} I_N + O\left(\frac{1}{N^4}\right)$$

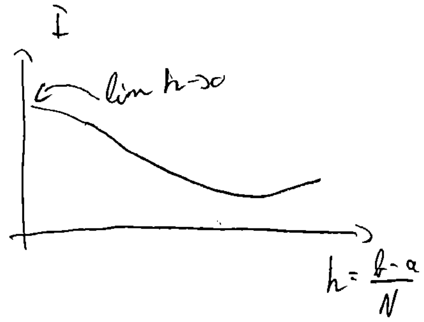
Romberg integrálás



$N \to \infty$

I egyre pontosabb, $N \to \infty$ lehet extrapolációval

⇐⇒



Nyílt formulák

- Ha valahol a függvény nem értelmezhető ki (a szingularitást a széleken vesszük)

$$f_i := f(x_i) \qquad f_{3/2} := f\left(\frac{x_1 + x_2}{2}\right)$$

$$\int_{x_0}^{x_{N-1}} f(x)\,dx = h\left(f_{3/2} + f_{5/2} + \ldots + f_{N-1/2}\right) + O\left(\frac{1}{N^2}\right)$$

- az integrálási határ $\pm\infty$

változó transzformáció, $(-\infty, -a]$ között, $(-a, b)$ és $[b, \infty)$ között integrálunk

$$x = \frac{1}{t} \qquad dx = \frac{1}{t^2}\,dt \qquad \int_a^b f(x)\,dx = -\int_{1/a}^{1/b} f\left(\frac{1}{t}\right)\frac{1}{t^2}\,dt = \int_{1/b}^{1/a} \frac{1}{t^2} f\left(\frac{1}{t}\right)dt$$

- ha a határon vagy a tartományban integrálható szingularitás van

pl: $f(x) = (x-a)^{\gamma-1}$ $\quad \gamma \in [0,1)$ $\quad$ a-ban szingularitás

$$x = t^{\frac{1}{1-\gamma}} + a \qquad t = (x-a)^{1-\gamma}$$

$$dx = \frac{1}{1-\gamma}\, t^{\frac{1-(1-\gamma)}{1-\gamma}}\, dt$$

$$\int_a^b f(x)\,dx = \frac{1}{1-\gamma}\int_0^{(b-a)^{1-\gamma}} t^{\frac{\gamma}{1-\gamma}}\, f\left(t^{\frac{1}{1-\gamma}}+a\right) dt$$

Gauss formulák

Létezés, hibarend érhető el $\left(\frac{1}{N^2} \to \frac{1}{N^4}\right)$

Nem egyenletesen osztjuk fel a tartományt, hanem okosan

Csak bizonyos függvényekre megy. Lehet egzakt és közelítéses is.

pl: $\int_{-1}^{1} \frac{e^{-\cos^2 x}}{\sqrt{1-x^2}}\,dx$ ha polinom lenne, egzaktul lehet integrálni, ha csak polinommal közelíthető, közelítéses módszert kapunk

integrálható szingularitást tartalmazó függvény, $:= W(f)$

$$\int_a^b W(x)\cdot f(x)\,dx \approx \sum_{j=1}^{N} w_j\, f(x_j)$$

W(x)-et kifejtjük valamilyen polinombázisban $p_0, p_1, p_2 \ldots$, ami W(x)-re merőleges, vagyis

$$\langle p_i | p_j \rangle := \int_a^b p_i(x)\, W(x)\, p_j(x)\,dx = c\cdot\delta_{ij}$$

a polinomok gyökei legyenek x_j

$$\sum_j \underbrace{p_i(x_j)}_{\underline{\underline{Q}}}\, w_j = \int_a^b W(x)\, p_i(x)\, \delta_{i0}\,dx \qquad \underline{\underline{Q}}\cdot\underline{w} = \begin{pmatrix} \int_a^b W(x)\,p(x)\,dx \\ 0 \\ 0 \\ \vdots \end{pmatrix}$$

→ megoldás a w_j

Polinomok meghatározása rekurzióval $W(x)$-ből

$p_{-1}(x) = 0$

$p_0(x) = 1$

$p_{j+1}(x) = (x - a_j)\, p_j(x) - b_j\, p_{j-1}(x)$ ahol

$$a_j = \frac{\langle x p_j | p_j \rangle}{\langle p_j | p_j \rangle}$$

$$b_j = \frac{\langle p_j | p_j \rangle}{\langle p_{j-1} | p_{j-1} \rangle}$$

egyszerű példák

$W(x) = 1 \qquad x \in (-1,1)$

$(j+1)\, p_{j+1} = (2j+1)\, x\, p_j - j\, p_{j-1}$ ← ezt nem vezettük le

$W(x) = (1-x^2)^{-1/2} \qquad x \in (-1,1)$ CHEBYSEV polinomokat kapjuk

$W(x) = x^{\alpha} e^{-x} \qquad \alpha \in N, \quad x \in (0,\infty)$ Lager - Gauss polinomok

$W(x) = e^{-x^2} \qquad x \in (-\infty,\infty)$ Hermite polinomok

$W(x) = (1-x^{\alpha})(1-x^{\beta}) \qquad x \in (-1,1) \qquad \alpha \in \mathbb{R} \ni \beta$

Többdimenziós integrálok

Baj: Az osztó pontok száma $N \sim \underset{\uparrow}{n}^{\overset{\swarrow}{d}}$ (dimenzió; 1d-s felontás)

Másik baj: már az integrálási határok sem triviálisak

De szerencsére néha szimmetria segítségével csökkenthető a dimenzió

– Ha a határok explicit módon adottak, $I = \int_V f(x,y,z)\, dx\, dy\, dz = \int_{x_1}^{x_2} dx \int_{y_1(x)}^{y_2(x)} dy \int_{z_1(y,z)}^{z_2(y,z)} dz\, f(x,y,z)$

Erre ugyanúgy vannak Newton-Cotes formulák és több D-s Gauss formulák

– Monte-Carlo módszerek, ha nincs könnyen meghatározott tartomány 04.27 / 4

kb. véletlenek generálunk, megnézzük hogy a kapott pont benne van-e a tartományban

$$\int f\,dV \approx V\langle f\rangle \qquad \langle f\rangle = \frac{1}{N}\sum_{i=1}^{N} f(x_i)$$

$$\int f\,dV = V\langle f\rangle \pm V\sqrt{\frac{\langle f^2\rangle - \langle f\rangle^2}{N}} \quad + O\,\frac{1}{\sqrt{N}}$$

Persze van jobb módszer, mint egyenletesen szórni a pontokat. Ehhez kell olyan véletlenszám generátor, ami tetszőleges, kívánt eloszlásban generál.

Azért kell a véletlen, mert egy reguláris rács szisztematikus hibákat csinálhat.

De egy kis szabályossággal $O\left(\frac{1}{N}\right)$ elérhető

05.04.

Numerikus optimalizáció

Nemlineáris egyenletek gyökeinek megtalálása ilyen, csak iteratív megoldások vannak.

Általában: N egyenlet, M ismeretlen van. Ha $N > M$ gyakran nincs megoldás, ha $N = M$ vagy $N < M$ akkor lehet megoldás, akár több is, akár végtelen sok is.

Nincsenek nagyon jó módszerek, nehéz is.

1D-s eset

I/ Nem tudjuk a függvény deriváltat, vagy nem akarjuk kiszámolni

1/ gyök bekerítése: keresünk a, b-t, hogy $f(a)\cdot f(b) < 0$. Ha éppen nem igaz, és wlog $|f(a)| < |f(b)|$, $a \mapsto a - |a-b|$

2/ gyök kereső iteráció

2/a gyökkeresés: felezős módszer (BISECTION

felezem az intervallumot, c a felezőpont, $f(c)\cdot f(a) > 0$ vagy $f(c) < f(b) < 0$

akkor kidobjuk a-t, $c := a'$. Ha épp $f(c)=0$ akkor megvan a gyök.

Elég stabil megoldás, mindig közelebb kerülünk a megoldáshoz. Ha ε_0 a kezdeti, ε pedig az elvárt, akkor $n = \log_2 \frac{\varepsilon_0}{\varepsilon_0}$ lépésben elérjük a kívánt pontosságot.

2/b secant, húr módszer

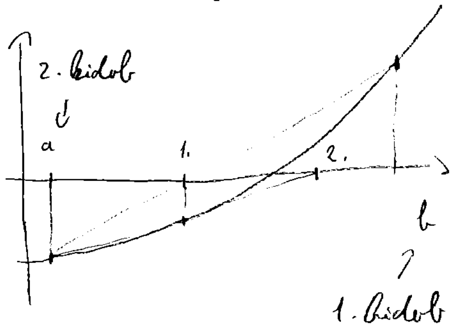
nem felezzük az intervallumot, hanem lineárisan megfelelően vesszük az osztópontot, és a régebbit hagyjuk el. Kezdetben régebbi: $|f(b)| > |f(a)|$. Baj: nem marad a

$\uparrow$ régebbi

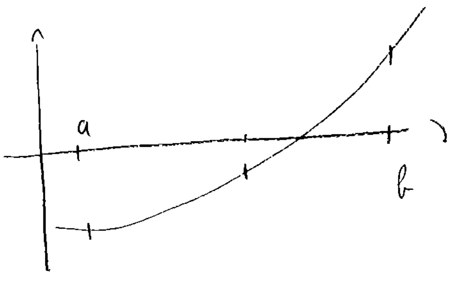
$\frac{1+\sqrt{5}}{2}$ gyök bekeretezve. De ez gyakran nem szokott

$$\lim_{n\to\infty} |\varepsilon_{n+1}| = c\cdot|\varepsilon_n|^{1,618}$$ bajt okozni.

„regula falsi" hogy a gyök mindig belül maradjon, az azonos előjelűt dobjuk el



2/c Lehetne 2-odfokút illesztve:



a pontokra feltett parabolát illesztünk. A tengelymetszetet megtartjuk, a régebbit kidobjuk.

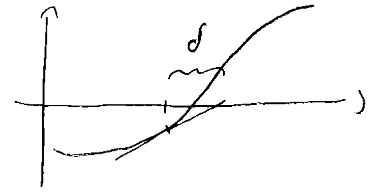
Gyorsaság vs stabilitás.

2/d kombinált módszerek: inverz kvadratikus (gyors) és felezős (stabil, jó a keret):

ha egy kvadratikus lépés legalább felez, és bekeretezve tart, akkor accept, ha nem, akkor felez.

II/ Ismerjük a deriváltat

szar példát rajzoltam



Newton Raphson módszer

$$f(x+\delta) = \frac{f(x)}{0!} + \frac{f'(x)\delta}{1!} + \frac{f''(x)\delta}{2!} \overset{!}{=} 0$$

ha $f'' \ll f'$, $\delta = -\frac{f(x)}{f'(x)}$ elhanyagolt lépés

Nincs keverés! $\varepsilon_{k+1} \hat{=} c \cdot \varepsilon_k^2$

Ha nem ismert a deriváltat, valahogy közelítjük, mert csomó időbe kerül. Polinomot azért még szabad.

Több gyök esetén: $Q(x) = \frac{P(x)}{(x-r)}$ — eredeti polinom; r: gyök

Hibák nagyon halmozódnak, mert a gyökök nem pontosak.

Több D-s eset

két függvény metszéspontjai, ahol 0-k, $f(x,y)=0$ és $g(x,y)=0$

NEWTON-RAPHSON módszert általánosítjuk több D-ra.

$$f_i(\underline{x}+\delta\underline{x}) = f_i(\underline{x}) + \sum_{i=1}^{N} \frac{\partial f_i}{\partial x_j}\delta x_j + \dots \qquad \frac{\partial f_i}{\partial x_j} = J_{ij} \text{ Jakobi Mátrix}$$

(N: dim)

$\underline{f}(\underline{x}+\delta\underline{x}) = \underline{f}(x) + \underline{\underline{J}}\,\delta\underline{x}$ Most $f(\underline{x}+\delta\underline{x})=0$ pontot keressük: $\underline{\underline{J}}\,\delta\underline{x} = -\underline{f}(\underline{x})$

Lineáris egyenletrendszerre vezet, LU-dekomp.

NUM OPT: Gradiens, szimulált hűtés. 1D sem triviális, de több D már katasztrófa.

$P(\text{arra megyünk, amerre növekszik}) = e^{-\frac{\Delta E}{kT}}$

$P(\text{arra, merre csökken}) = 1$

Sajátérték probléma

$\underline{\underline{A}}\,\underline{x} = \lambda\,\underline{x}$

$\det \left| \underline{\underline{A}} - \lambda \underline{\underline{I}} \right| = 0$

↓

karakterisztikus polinom, $N \times N$-es mátrix → N-edfokú egyenlet

ha $N \geq 4$ a polinom nem oldható meg egzaktul, numerikus megoldás nem optimális

Más megoldások: - iteratív megoldás

- karakterisztikus polinom gyökeinek megkeresése

Szimmetrikus valós mátrixnak N valós s.é. és N s.v. van

$$\underline{\underline{X}}_R = \begin{pmatrix} \underline{x}_1 & \underline{x}_2 & \cdots \end{pmatrix} \qquad \underline{\underline{X}}_L = \begin{pmatrix} (& x_1 &) \\ (& x_2 &) \\ & \vdots & \end{pmatrix} \qquad \underline{\underline{X}}_R^{-1}\,\underline{\underline{A}}\,\underline{\underline{X}}_R = \begin{pmatrix} \lambda_1 & & \\ & \lambda_2 & \\ & & \lambda_3 \\ & & & \ddots \end{pmatrix}$$

$\underline{\underline{P}}^T \underline{\underline{A}}\,\underline{\underline{P}}$ -nek ugyan annyi a sajátvektorainak értékei

ortog.

$$P_k^T \cdots P_2^T P_1^T \, A \, P_1 P_2 \cdots P_k = \begin{pmatrix} \ddots & & \varepsilon \\ & \ddots & \\ \varepsilon & & \ddots \end{pmatrix}$$

JACOBI TRANSZFORMÁCIÓ

síkforgatások sorozatával diagonalizáljuk

Többdimenziós forgatás

$$\begin{array}{c} \\ P_{pq} = \begin{array}{c} \\ p \\ \\ q \\ \end{array} \end{array} \begin{array}{c} \begin{array}{cccc} \quad & p & q & \end{array} \\ \begin{pmatrix} 1 & & & \\ & c & s & \\ & & 1 & \\ & -s & c & \\ & & & \ddots \end{pmatrix} \end{array}$$

ortogonális, $P^T = P^{-1}$

$c^2 + s^2 = 1$ teljesüljön

$$\underline{\underline{P}}^T \underline{\underline{A}}\,\underline{\underline{P}} = \underline{\underline{A}}' = \begin{pmatrix} & & a'_{1p} & a'_{1q} \\ & & \vdots & \vdots \\ a'_{p1} & & a'_{pp} & a'_{pq} \\ a'_{q1} & & a'_{qp} & a'_{qq} \end{pmatrix}$$

csak a p, q sorok és oszlopok változnak

05.11
2

$r \neq p, \; r \neq q$

$a'_{pr} = a'_{rp} = c\,a_{rp} - s\,a_{rq}$ szimm. valós az $\underline{\underline{A}}$

$a'_{qr} = a'_{rq} = c\,a_{rq} + s\,a_{rp}$

$a'_{pp} = c^2 a_{pp} + s^2 a_{qq} - 2sc\,a_{pq}$

$a'_{qq} = s^2 a_{pp} + c^2 a_{qq} - 2sc\,a_{pq}$

$a'_{pq} = a'_{qp} = (c^2 - s^2)\,a_{pq} + sc\,(a_{pp} - a_{qq})$ ← ez legyen 0

$$(c^2 - s^2)\,a_{pq} = sc\,(a_{qq} - a_{pp})$$

$2\,\mathrm{ctg}(2\varphi) = \frac{c^2 - s^2}{sc} = \frac{a_{qq} - a_{pp}}{a_{pq}}$ ezzel 0 lesz. Állítás: a többi cellaértékek (a diagonálison kívüli) elemek négyzetösszege $S = \sum_{r \neq s} |a_{rs}|^2$ teljesíti: $S_{i+1} = S_i - 2|a_{pq}|^2$

$$\underbrace{\cdots P_{pq}}_{\underline{\underline{X}}_n^T} \cdot \underline{\underline{A}} \cdot \underline{\underline{P}}_{pq}\, \underline{\underline{P}}_{p'q'} \cdots$$

Így monoton módon csökken a 0-hoz

p és q megválasztásánál jó kiválasztani a maxot.

Az egész N^3-bel skálázik.

Numerikusan nem optimális a maximum elem keresése, akkor már sorba vehetünk minden elemet.

<u>Householder redukció + QR módszer</u>

Ez is N^3-bel skálázik, de 5-ször gyorsabb.

$\underline{\underline{A}} \to$ tridiag $\to$ háromszög m. $\to$ s.é., s.v.

Pontok a térben, v_i

$$\sum_i \underline{v}_i \circ \underline{v}_i^T = \underline{\underline{C}}$$ s. é. probléma a pontfelhőnek „irányát" adja meg. zajmentesítésre jó.

Nem kell az összeset, elég csak néhányat megtalálni (a „legnagyobbakat")

Hatvány iteráció

Legnagyobb sajátérték meghatározása

$$y_0 = \sum_{k=1}^{N} \alpha_k \underline{x}_k$$ jobboldali s.v. tetszőleges vektor

$$\underline{y}_1 =: \underline{\underline{A}}\, y_0 = \sum_{k=1}^{N} \alpha_k \underline{\underline{A}}\, x_k = \sum_{k=1}^{N} \alpha_k \lambda_k x_k$$

$\lambda_1 > \lambda_2 > \lambda_3 \dots$

$$\underline{y}_m = \underline{\underline{A}}^m y_0 = \sum_{k=1}^{N} \alpha \lambda_k^m x_k$$

0-hoz tart

$$\lim_{m\to\infty} \frac{1}{\lambda_1^m} \underline{\underline{A}}^m y_0 = \alpha_1 \underline{x}_1 \qquad \left(\frac{\lambda_1}{\lambda_1}\right)^m x_1 + \left(\frac{\lambda_2}{\lambda_1}\right)^m \underline{x}_2$$

$$\lim_{m\to\infty} \underline{\underline{A}}^m y_0 = \lambda_1^m \alpha_1 \underline{x}_1 \longrightarrow \underline{x}_1$$

mert legyen $\underline{v}^T$ nem ort. $\underline{x}_1$-re, tetszőleges

$$\lambda_1 = \lim_{m\to\infty} \frac{\underline{v}^T y_{m+1}}{\underline{v}^T y_m}$$

$\underline{\underline{P}} = \underline{\underline{I}} - 2\,\underline{w} \circ \underline{w}$ (egységvektor)

$\underline{\underline{P}}^2 = \underline{\underline{I}}$

$\underline{\underline{P}}^T = \underline{\underline{P}}^{-1} = \underline{\underline{P}}$

$$\underline{\underline{A}} = \begin{pmatrix} \underline{x} & \cdots \end{pmatrix}$$

hogy válasszuk meg $\underline{w}$?

$\underline{u} = \underline{x} \pm |\underline{x}|\, \underline{e}_1 \qquad H = \frac{1}{2}|\underline{u}|^2$

$$\underline{\underline{P}} = \underline{\underline{I}} - \frac{\underline{u} \cdot \underline{u}^T}{H}$$

$\underline{\underline{P}}\,\underline{x} = \dots = \pm|\underline{x}|\,\underline{e}_1$

$$\underline{\underline{P}}_1 \underline{\underline{A}} = \begin{pmatrix} 1 & 0 \\ 0 & \underline{\underline{P}} \end{pmatrix} \cdot \underline{\underline{A}} = \begin{pmatrix} a_{11} & a_{12} & \dots \\ k & & \\ 0 & & \\ \vdots & & \end{pmatrix}$$

$$\underline{\underline{P}}_1 A \underline{\underline{P}}_1 = \begin{pmatrix} a_{11} & k' & 0 & \dots \\ k & & & \\ 0 & & & \\ \vdots & & & \end{pmatrix}$$

$$\underline{\underline{P}}_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \underline{\underline{P}} \end{pmatrix} \rightarrow \underline{\underline{P}}_2 \underline{\underline{P}}_1 \underline{\underline{A}}\, \underline{\underline{P}}_1 \underline{\underline{P}}_2 = \begin{pmatrix} a_{11} & k_1 & 0 & 0 \\ k_1' & a_{22} & k_2 & 0 \\ 0 & k_2' & & \\ 0 & 0 & & \end{pmatrix}$$

további Householder mátrixokkal alakú lesz, $\underline{\underline{A}} = \underline{\underline{P}}\,\underline{\underline{Q}}$

Ilyen alakra a tridiag. alak nélkül is lehet, de a pontosság miatt így jobb, mert a sok 0 és 1 elem miatt nem igazi mátrixszorzást kell végezni.