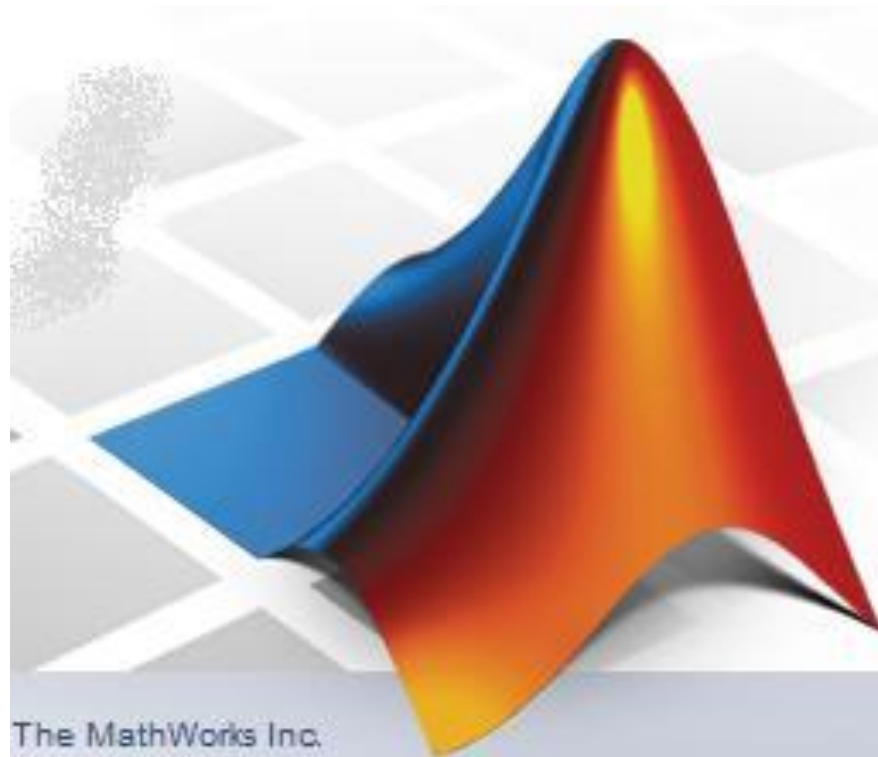


# Programovanie geofyzikálnych úloh v prostredí MATLAB



# Obsah

- Blakelyho metóda v gravimetrii
- načítanie a ukladanie Surfer ASCII grid súboru - funkcie
- riešenie systémov lineárnych rovníc v Matlabe  
aplikácie v mnohých geof. aplikáciách, príklad:
  - 2D hustotná inverzia (metóda Last-Kubik)
- Hilbertova transformácia (ako príklad IIR filtra)



# Programovanie geofyzikálnych úloh v prostredí MATLAB

## tzv. Blakelyho metóda lineárnych prvkov v gravimetrii

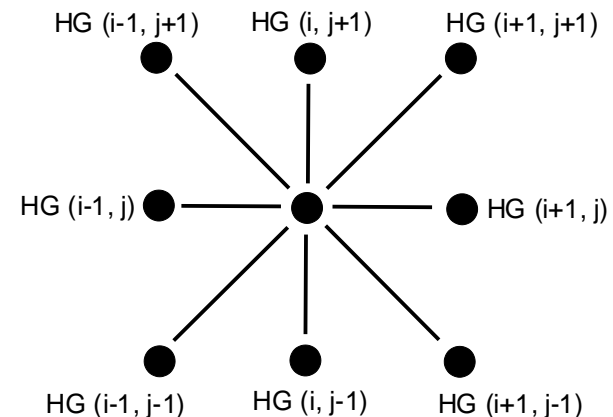
1. spočítajú sa horizontálne gradienty  $\partial\Delta g_B/\partial x$  a  $\partial\Delta g_B/\partial y$

2. vypočíta sa totálny horizontálny gradient:

$$HG = \sqrt{(\partial\Delta g_B/\partial x)^2 + (\partial\Delta g_B/\partial y)^2}$$

3. zistia sa polohy lokálnych maxím HG v okne 3 x 3 body podľa schémy:

overujú sa podmienky v N-S, E-W a diagonálnom smere (či je stredná hodnota väčšia, ako krajné) a pre každú polohu okna sa registruje počet splnených podmienok; wynášajú sa iba tie body, ktoré splňujú 3 a 4 tieto podmienky



Pozn.:  $\Delta g_B$  znamená ÚBA.

# Načítanie (a uloženie) gridu formátu GS Surfer ASCII

```
Lister - [d:\Roman\skola\prednasky\Matlab\3_predn\bram.grd]
Súbor Upraviť Možnosti Kódovanie Pomocník
DSAA
64 64
3370306.5 3520576.2
5698489.5 5851003.1
-21.024914883207 33.638097572244
8.7836697219637 8.7192318352404 8.7511441782683 8.90671
8.9483941143178 8.2594140698399 7.3300588225985 6.02358
0.1589426811709 0.34874959558678 0.83678387206162 1.941
7.6525150833535 7.7559344882513 7.4875566401002 6.73971
0.71383434597499 0.15434167952452 -0.97832050847555 -1.
4.5424592210783 2.8101533328325 1.3524793313003 -0.4938
-9.4783774989215 -8.7722879076653 -7.8780315215616 -7.1
8.5086991873055 8.4025308748885 8.2507723392527 7.99085
8.6995031113981 8.3614847609599 7.50863112266 6.3692309
0.2480656905315 0.23826407878389 0.11415347043415 1.301
7.766944139408 7.9698232042287 7.8898679642282 7.274371
1.4715298111948 0.77531891701374 0.47361998650868 -0.41
4.6077301993764 3.2696236482341 1.6594513707813 -0.3059
-9.4314336045015 -8.6152398390269 -7.6141310885153 -6.7
```

Obsah Register Hľadat'

Zadajte hľadane slová:

grid format

Zobrazit' témy

Zobrazit'

Výber témy:

Nájdenej: 58

Názov	Umi...	Hodnoter
Grid Data	Surf...	1
Surfer 7 Grid File For...	Surf...	2
Grid Convert	Surf...	3
Volumes and Areas	Surf...	4
ASCII Grid File Format	Surf...	5
Grid Blank	Surf...	6
Breaklines and Faults	Surf...	7
Surfer 6 Grid File For...	Surf...	8
Producing a Grid Fil...	Surf...	9
Grid Menu Commands	Surf...	10
Grid Files	Surf...	11
GridData	Surf...	12
GridMosaic	Surf...	13
New Features	Surf...	14
Triangulation with Li...	Surf...	15
GridMath	Surf...	16
File Format Chart	Surf...	17
Error	Surf...	18
Save As	Surf...	19
Natural Neighbor	Surf...	20
Grid Transform	Surf...	21
GridConvert	Surf...	22
SrfGridFormat Values	Surf...	23
SaveFile [Grid]	Surf...	24
Error	Surf...	25
Menu Commands	Surf...	26
List of All Methods a...	Surf...	27
Import	Surf...	28
GridExtract	Surf...	29
Vector Length Lege...	Surf...	30
Color Spectrum File ...	Surf...	31

Prehľadat' predchádzajúce výsledky

Zahrnúť aj podobné slová

## ASCII Grid File Format

ASCII grid files [.GRD] contain five header lines that provide information about the size and limits of the grid, followed by a list of Z values. The fields within ASCII grid files must be space delimited.

The listing of Z values follows the header information in the file. The Z values are stored in row-major order starting with the minimum Y coordinate. The first Z value in the grid file corresponds to the lower left corner of the map. This can also be thought of as the southwest corner of the map, or, more specifically, the grid node of minimum X and minimum Y. The second Z value is the next adjacent grid node in the same row (the same Y coordinate but the next higher X coordinate). When the maximum X value is reached in the row, the list of Z values continues with the next higher row, until all the rows of Z values have been included.

The general format of an ASCII grid file is:

id                    The identification string DSAA that identifies the file as an ASCII grid file.

nx ny                nx is the integer number of grid lines along the X axis (columns)  
ny is the integer number of grid lines along the Y axis (rows)

xlo xhi              xlo is the minimum X value of the grid  
xhi is the maximum X value of the grid

ylo yhi              ylo is the minimum Y value of the grid  
yhi is the maximum Y value of the grid

zlo zhi              zlo is the minimum Z value of the grid  
zhi is the maximum Z value of the grid

grid row 1

grid row 2

grid row 3

# Načítanie (a uloženie) gridu formátu GS Surfer ASCII

## Funkcie:

`grid_read_SRF_ASCII.m`

a

`grid_write_SRF_ASCII.m`

## Hlavičky:

*`function [matrix, xmin, xmax, ymin, ymax] = grd_read_v2(namefile)`*

*`function grd_write(matrix,xmin,xmax,ymin,ymax,namefile)`*

## Testovací skript:

`test_grid_read_write_SRF_ASCII.m`

# Programovanie geofyzikálnych úloh v prostredí MATLAB

## d'alsie dôležité príkazy:

riešenie systému lineárnych rovníc:

$$Ax = b$$

kde  $A$  je tzv. matica systému ( $N$ -riadkov  $\times$   $M$ -stĺpcov) ,

$x$  je vektor hľadaných neznámych – jednotĺpcová matica ( $M$ -riadkov  $\times$   $1$ -stĺpec)

a  $b$  tzv. pravá strana systému - jednotĺpcová matica ( $N$ -riadkov  $\times$   $1$ -stĺpec)

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1M} \\ a_{21} & a_{22} & \dots & a_{2M} \\ \dots & \dots & \dots & \dots \\ a_{N1} & a_{N2} & \dots & a_{NM} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_M \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \dots \\ b_N \end{pmatrix}$$

systém je riešiteľný, keď  $N = M$  (počet rovníc = počet neznámych),

prípadne keď  $N > M$ , kde ide o tzv. preurčený systém (rieši sa metódou LSQ)

# Programovanie geofyzikálnych úloh v prostredí MATLAB

## d'alsie dôležité príkazy:

riešenie systému lineárnych rovníc:  $Ax = b$

kde  $A$  je tzv. matica systému ( $m$ -riadkov  $\times$   $n$ -stĺpcov) ,

$x$  je vektor hľadaných neznámych – jednotstĺpcová matica ( $n$ -riadkov  $\times$   $1$ -stĺpec)

a  $b$  tzv. pravá strana systému - jednotstĺpcová matica ( $m$ -riadkov  $\times$   $1$ -stĺpec)

- poznáme veľké množstvo metód – Matlab ich poskytuje v značnom množstve

- hlavný príkaz v Matlabe:  $x = A \backslash b$ ;      (\-'backslash' na rozdiel od /-'slash')

( $x=A/b$  by riešil tzv. delenie sprava  
t.j. súčin inv. matice z  $b$  krát matica  $A$ )

Tento príkaz  $\backslash$  je však „čiernou skrinkou = black box“ – nevieme, že pre akú metódu sa Matlab pri konečnej realizácii presne rozhodne.

príklad:

```
clear all; close all; clc;
```

```
A=magic(3)
```

```
b=[3; 1; 4]
```

```
x = A \ b    % použitie „black-box“ metódy
```



# Programovanie geofyzikálnych úloh v prostredí MATLAB

riešenie systému lineárnych rovníc:  $Ax = b$

- hlavný príkaz v Matlabe:  $x = A \backslash b$ ;

- ďalšie možnosti:

- LU-rozklad (LU-decomposition),

- Choleského-rozklad (Cholesky-decomposition),

- SVD-rozklad (SVD-decomposition),

- QR-rozklad (QR-factorization),

- použitie pseudo-inverznej matice (v prípade, že matica systému  $A$  je singulárna)

# Programovanie geofyzikálnych úloh v prostredí MATLAB

riešenie systému lineárnych rovníc:  $Ax = b$

príklady:

```
clear all;
A=magic(3)
b=[3; 1; 4]
```

$$\begin{bmatrix} \alpha_{11} & 0 & 0 & 0 \\ \alpha_{21} & \alpha_{22} & 0 & 0 \\ \alpha_{31} & \alpha_{32} & \alpha_{33} & 0 \\ \alpha_{41} & \alpha_{42} & \alpha_{43} & \alpha_{44} \end{bmatrix} \cdot \begin{bmatrix} \beta_{11} & \beta_{12} & \beta_{13} & \beta_{14} \\ 0 & \beta_{22} & \beta_{23} & \beta_{24} \\ 0 & 0 & \beta_{33} & \beta_{34} \\ 0 & 0 & 0 & \beta_{44} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix}$$

```
[L,U] = lu(A) % použitie LU-rozkladu
```

```
y = L\b % rieši spodný trojuholníkový systém
```

```
x = U\y % rieši horný trojuholníkový systém
```

Instead of seeking arbitrary lower and upper triangular factors  $L$  and  $U$ , Cholesky decomposition constructs a lower triangular matrix  $L$  whose transpose  $L^T$  can itself serve as the upper triangular part. In other words we replace equation (2.3.1) by

```
clear all;
```

```
A=magic(3)
```

```
b=[3; 1; 4]
```

$$L \cdot L^T = A \quad (2.9.2)$$

```
R = chol(A) % použitie Choleského-rozkladu; pozor matica A musí spĺňať
```

```
y = R'\b % určité podmienky (tzv. pozitívne definitná, t.j.  $x^T A x > 0$ )
```

```
x = R\y
```

```
clear all;
```

```
A=magic(3)
```

```
b=[3; 1; 4]
```

```
x = pinv(A)*b %použitie pseudo-inverznej matice
```

# Programovanie geofyzikálnych úloh v prostredí MATLAB

riešenie systému lin. rovníc:

príklady:

`clear all;`

`A=magic(3)`

`b=[3; 1; 4]`

`x1 = A\b`

`[U,S,V] = svd(A)` % použitie SVD-rozkladu (Singular Value Decomposition)

`for i=1: sqrt(numel(A))` % preklopenie diagonálnych prvkov matice S

`SD(i,i) = 1/S(i,i);`

`end;`

`x = V*SD*(U'*b)`

`c = cond(A)` % tzv. condition number (hovorí o miere nestability systému)

`clear all;`

`A=magic(3)`

`b=[3; 1; 4]`

`x1 = A\b`

`[Q,R] = qr(A)` % použitie QR-rozkladu

`y = Q\b`

`x = R\y`

$$\begin{pmatrix} & & & \\ & \mathbf{A} & & \\ & & & \\ & & & \end{pmatrix} = \begin{pmatrix} & & & \\ & \mathbf{U} & & \\ & & & \\ & & & \end{pmatrix} \cdot \begin{pmatrix} w_1 & & & \\ & w_2 & & \\ & & \dots & \\ & & & \dots \\ & & & & w_N \end{pmatrix} \cdot \begin{pmatrix} & & & \\ & & & \\ & & & \\ & & & \mathbf{V}^T \end{pmatrix} \quad (2.6.1)$$

$$\mathbf{A}^{-1} = \mathbf{V} \cdot [\text{diag}(1/w_j)] \cdot \mathbf{U}^T$$

$$\mathbf{A} = \mathbf{Q} \cdot \mathbf{R}$$

Here  $\mathbf{R}$  is upper triangular, while  $\mathbf{Q}$  is orthogonal, that is,

$$\mathbf{Q}^T \cdot \mathbf{Q} = \mathbf{1}$$

## riešenie systémov lineárnych rovníc:

### príklady použitia z geofyziky:

- LSQ; v metóde najmenších štvorcov sa vždy rieši systém tzv. normálnych rovníc, ktoré majú toľko neznámych, koľko je rovníc (získaných deriváciou sumy štvorcov odchýliek podľa hľadaných neznámych koeficientov),
- fitovanie rôznych lineárnych a nelineárnych funkcií,
- riešenia pre tzv. semi-automatické interpretačné metódy (Wernerova a Eulerova dekonvolúcia),
- riešenie obrátenej úlohy pre mnohé metódy (mnohé gravimetrické a magnetometrické riešenia; ERT – „Locke“),  
**tzv. *inversions***  
väčšinou systém rovníc popisuje vzťah medzi výpočt. bodmi a bunkami modelu (napr. pravouhlé hranoly),
- atď., atď., atď., ...

# Inverzie

Priama úloha sa prepíše z integrálneho do diskkrétneho tvaru:  
(pole  $g$  je zistené v  $N$  diskrétnych bodoch):

$$|g| = \kappa \iiint_{\tau} \rho(\mathbf{r}) \Psi(\mathbf{r} - \mathbf{r}') d\tau \rightarrow g_i \approx \kappa \sum_{j=1}^M \rho_j \psi_{ij} \Delta\tau \quad \begin{array}{l} i = 1 \dots N \text{ (počet bodov poľa),} \\ j = 1 \dots M \text{ (počet hranolov),} \end{array}$$

celý modelový priestor sa rozdelí na  $M$  malých telies (buniek).

To vedie ku vzniku systému lineárnych rovníc (s  $M$  neznámymi):

$$\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$$

$\mathbf{A}$  je tzv. hlavná matica systému (vzťah medzi bodmi a bunkami),

$\mathbf{x}$  sú hľadané hustoty a  $\mathbf{b}$  je tzv. pravá strana systému (merané hodnoty  $g$ ).

---

V podstate by stačilo vyriešiť tento systém rovníc a problém obrátenej úlohy by bol vyriešený...

Ale také jednoduché to nie je – prejavujú sa tu všetky zdedené vlastnosti od integrálnych rovníc. Naplno sa prejavuje nestabilita (prispieva ku nej veľmi šum v nameraných údajoch). A aj mnohoznačnosť – tendencia ku vzniku ekvivalentných (hlavne plytkých) riešení.

V oblasti numerickej matematiky sa to prejavuje tak, že tieto systémy rovníc sú blízke singulárnym (determinant matice systému je blízky nule).

## metóda najmenších štvorcov (LSQ) pri riešení inverzie (1/2):

Často neriešime základný systém rovníc  $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$  priamo, ale ako minimalizačný problém (lepšie pri pridávaní ďalších vlastností):

$$S = \|\mathbf{b} - \mathbf{A} \cdot \mathbf{x}\|_{L_2} \rightarrow \min$$

napísané po prvkoch:

$$S = \sum_{i=1}^N \left( g_i - \sum_{j=1}^M a_{ij} x_j \right)^2 \rightarrow \min$$

Metóda najmenších štvorcov (LSQ) je postavená na vlastnosti hľadania minima funkcie:

$$\frac{\partial S}{\partial x_k} = 0, \quad k = 1, 2, \dots, M$$

čo vedie ku systému rovníc (k môže byť maximálne rovné M):

$$\frac{\partial S}{\partial x_k} = \sum_{i=1}^N 2 \left( g_i - \sum_{j=1}^M a_{ij} x_j \right) a_{ik} = 0, \quad k = 1, 2, \dots, M$$

## metóda najmenších štvorcov (LSQ) pri riešení inverzie (2/2):

$$\frac{\partial S}{\partial \mathbf{x}_k} = \sum_{i=1}^N 2 \left( \mathbf{g}_i - \sum_{j=1}^M \mathbf{a}_{ij} \mathbf{x}_j \right) \mathbf{a}_{ik} = 0, \quad k = 1, 2, \dots, M$$

Po úprave získavame:

$$\sum_{i=1}^N \mathbf{a}_{ik} \mathbf{g}_i = \sum_{i=1}^N \mathbf{a}_{ik} \left( \sum_{j=1}^M \mathbf{a}_{ij} \mathbf{x}_j \right), \quad k = 1, 2, \dots, M$$

Toto sa dá vyjadriť v tvare maticovej notifikácie:

$$\mathbf{A}^T \cdot \mathbf{b} = (\mathbf{A}^T \cdot \mathbf{A}) \cdot \mathbf{x}$$

Napokon z tohto vyjadríme riešenie pre hľadanú maticu  $\mathbf{x}$ :

$$\mathbf{x} = (\mathbf{A}^T \cdot \mathbf{A})^{-1} \cdot \mathbf{A}^T \cdot \mathbf{b}$$

Toto riešenie sa niekedy nazýva ako Gauss-Newtonove.

dôležité:

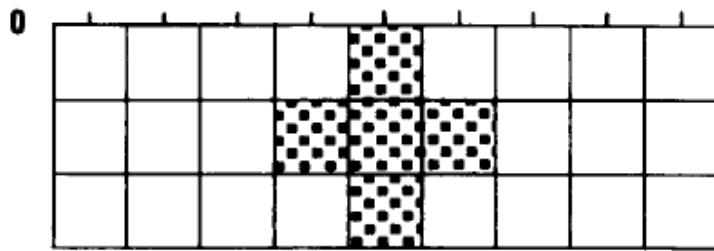
$$\mathbf{A}^T \cdot \mathbf{A}$$

výsledkom tohto súčinu je štvorcová symetrická matica, ktorá má svoj inverzný ekvivalent

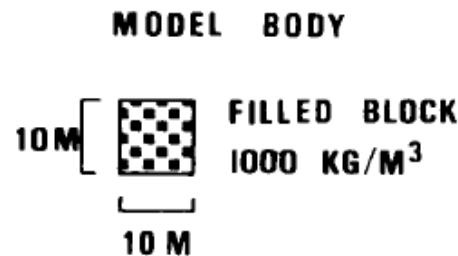
Často neriešime základný systém rovníc  $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$  priamo, ale ako minimalizačný problém (lepšie pri pridávaní ďalších vlastností):

$$\mathbf{x} = \left( \mathbf{A}^T \cdot \mathbf{A} \right)^{-1} \cdot \mathbf{A}^T \cdot \mathbf{b} \quad \left\| \mathbf{A} \mathbf{x} - \mathbf{b} \right\|_{L_2} = \min$$

Takéto jednoduché riešenie systému lineárnych rovníc však nevedie ku skutočnému riešeniu problému, výsledky sú väčšinou veľmi plytké (hneď pod povrchom – dôvodom sú vlastnosti hlavnej matice  $\mathbf{A}$ ).

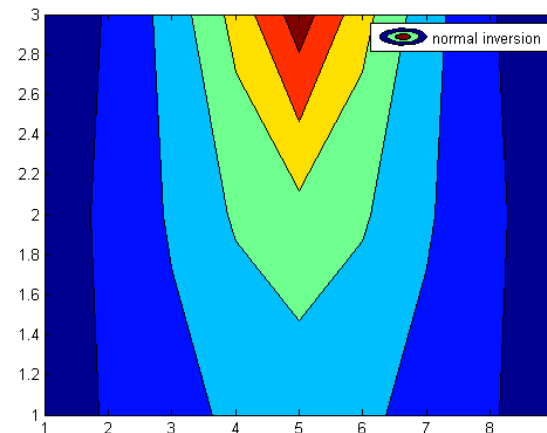
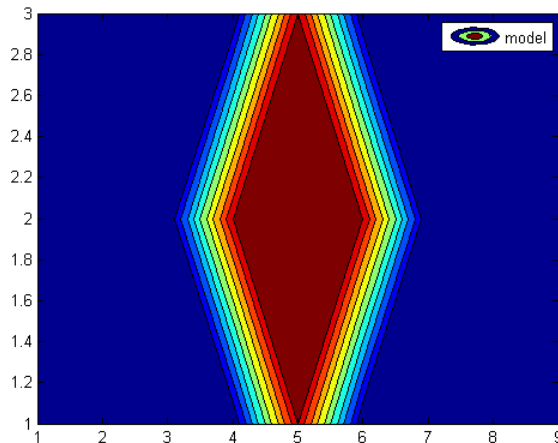


zobrazenie modelu v Matlabe



model  
z článku  
Last and Kubik  
(1983)

LSQ riešenie v Matlabe





Často neriešime základný systém rovníc  $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$  priamo, ale ako minimizačný problém (lepšie pri pridávaní ďalších vlastností):  $\|\mathbf{Ax} - \mathbf{b}\|_{L_2} = \min$

$$\mathbf{x} = (\mathbf{A}^T \cdot \mathbf{A})^{-1} \cdot \mathbf{A}^T \cdot \mathbf{b}$$

Do tohto relatívne jednoduchého zápisu je možné pridávať určité **špeciálne matice**, ktoré svojimi vlastnosťami vylepšujú riešenie a potláčajú:

1. **nestabilitu (regularizácia)** a
2. **mnohoznačnosť (váhové matice)**.

## 1. REGULARIZÁCIA

V hlavnej rovnici LSQ riešenia „vylepšíme“ štvorcovú maticu  $(\mathbf{A}^T \cdot \mathbf{A})$  tým, že ku nej pridáme jednotkovú maticu  $\mathbf{I}$ :

$$\mathbf{x} = (\mathbf{A}^T \cdot \mathbf{A} + \lambda \mathbf{I})^{-1} \cdot \mathbf{A}^T \cdot \mathbf{b}$$

vplyv tej jednotkovej matice je riadený regularizačným parametrom  $\lambda$ , ktorý sa určuje buď metódou pokusu a omylu alebo pomocou trošku sofistikovanejších metód.

Výsledné riešenia sú mierne hodnotovo posunuté (biased) oproti správnym. Táto metóda sa tiež nazýva Tichonov-Philipsova (Marquardt-Levenbergova).

Často neriešime základný systém rovníc  $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$  priamo, ale ako minimalizačný problém (lepšie pri pridávaní ďalších vlastností):  $\|\mathbf{Ax} - \mathbf{b}\|_{L_2} = \min$

$$\mathbf{x} = (\mathbf{A}^T \cdot \mathbf{A})^{-1} \cdot \mathbf{A}^T \cdot \mathbf{b}$$

Do tohto relatívne jednoduchého zápisu je možné pridávať určité **špeciálne matice**, ktoré svojimi vlastnosťami vylepšujú riešenie a potláčajú:

1. **nestabilitu (regularizácia)** a
2. **mnohoznačnosť (váhové matice)**.

## 2. VÁHOVÉ MATICE

Vyjadrujú prídavnú vlastnosť, ktorú by malo mať výsledné riešenie a je možné ich rozdeliť na 2 hlavné skupiny:

- a) matice vyjadrujúce kompaktnosť (hutnosť) riešení,
- b) matice ošetrojúce korektným spôsobom hĺbkovú závislosť riešenia.

## 2. VÁHOVÉ MATICE

a) matice vyjadrujúce kompaktnosť (hutnosť) riešení (Last and Kubik, 1983),

$$\mathbf{x} = \left( \mathbf{W}^{-1} \cdot \mathbf{A}^T \cdot \mathbf{A} \right)^{-1} \cdot \mathbf{W}^{-1} \cdot \mathbf{A}^T \cdot \mathbf{b}$$

kde  $\mathbf{W}$  je diagonálna matica s prvkami:  $w_j = (\sigma_j^2 + \varepsilon)^{-1}$ ,

( $\sigma_j$  je hustota j-teho modelového bloku,  $\varepsilon$  je malé číslo, napr.  $10^{-6}$ , aby nenastalo delenie nulou v prípade, že hustota bude nulová).

Využíva sa pri tom nasledujúca vlastnosť:

$$\lim_{\varepsilon \rightarrow 0} \left( \frac{\sigma^2}{\sigma^2 + \varepsilon} \right) = \begin{cases} 0 & \text{pre } \sigma = 0 \text{ (element modelu mimo spravneho riesenia)} \\ 1 & \text{pre } \sigma \neq 0 \text{ (element modelu v ramci spravneho riesenia)} \end{cases}$$

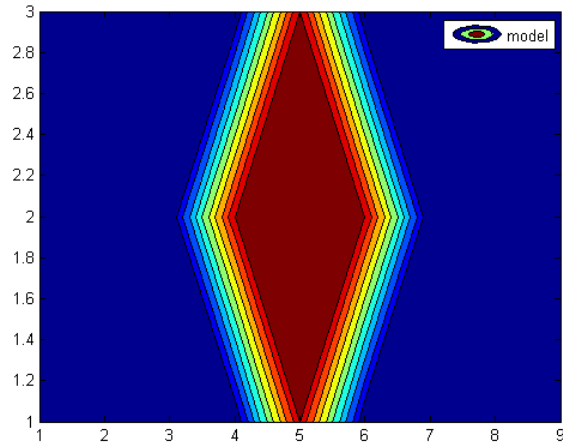
pozn.: limita pre  $\varepsilon$  idúce do nuly je samozrejme hypotetická, neráta sa.

Táto vlastnosť zabezpečuje to, že sa namiesto rozmazaných riešení získavajú ostrejšie obrysy anomálnych telies.

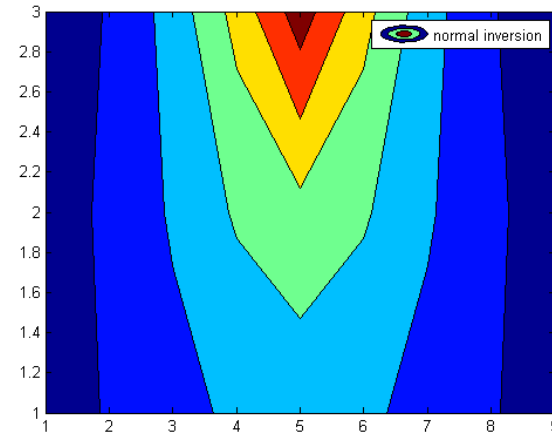
Účinok matice  $\mathbf{W}$  sa nedosiahne hneď na prvý pokus – je potrebné tento krok „liečenia“ opakovať po sebe niekoľko krát (iteratívny prístup).

# Porovnanie – klasické LSQ riešenie VS Last-Kubik (1983)

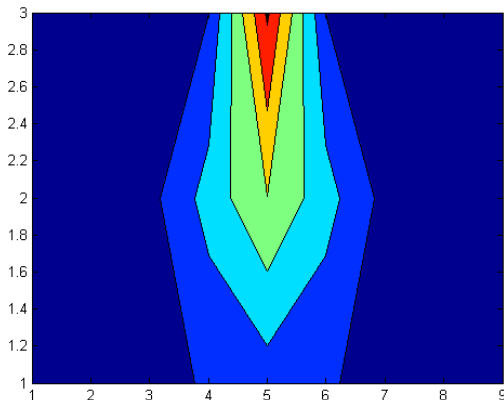
zobrazenie modelu v Matlabe



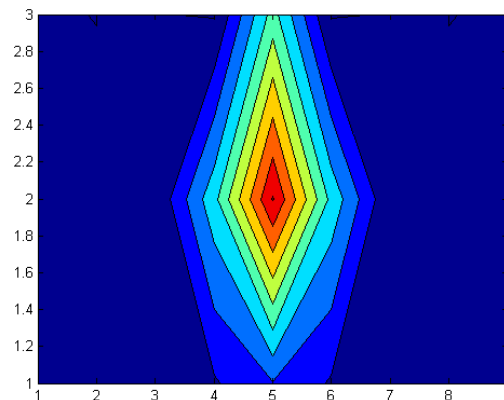
LSQ riešenie v Matlabe



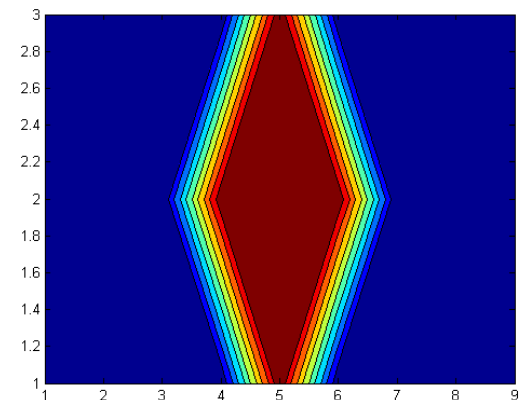
riešenie podľa Last and Kubik (1983), 2. iterácia



riešenie podľa Last and Kubik (1983), 4. iterácia



riešenie podľa Last and Kubik (1983), 10. iterácia



## Algoritmus metódy Last-Kubik (1983):

$$\mathbf{x} = \left( \mathbf{W}^{-1} \cdot \mathbf{A}^T \cdot \mathbf{A} \right)^{-1} \cdot \mathbf{W}^{-1} \cdot \mathbf{A}^T \cdot \mathbf{b} \quad \text{základná rovnica (v maticovom tvare)}$$

---

N – počet meraní, M – počet elementov modelu

- $\mathbf{A}$  je hlavná matica systému rovníc (N x M),
- matica  $\mathbf{W}$  je váhová štvorcová štvorcová matica (M x M)
- jednotípcová matica  $\mathbf{b}$  je tzv. pravá matica systému (N x 1)
- jednotípcová matica  $\mathbf{x}$  je matica riešení = hustôt (M x 1)

$$\mathbf{x} = \left( \mathbf{W}^{-1} \cdot \mathbf{A}^T \cdot \mathbf{A} \right)^{-1} \cdot \mathbf{W}^{-1} \cdot \mathbf{A}^T \cdot \mathbf{b} \quad \text{základná rovnica (v maticovom tvare)}$$


---

Pri konkrétnom riešení (Last a Kubik, 1983) zavádzajú autori okrem váhovej (kompaktnej)  $\mathbf{W}_v$  aj ďalšiu tzv. maticu chýb  $\mathbf{W}_e$  ( $N \times N$ ):

In previous work  $w_v$  and  $w_e$  have generally been stated to be independent of  $v$  and  $e$ , respectively. The procedure (5) then reduces to the classical least-squares problem, whose solution is, in matrix notation,

$$\hat{\mathbf{V}} = \mathbf{W}_v^{-1} \mathbf{A}^T (\mathbf{A} \mathbf{W}_v^{-1} \mathbf{A}^T + \mathbf{W}_e^{-1})^{-1} \mathbf{G}. \quad (9)$$

In particular if we set  $\mathbf{W}_v = \mathbf{I}$  (the identity matrix) and assign all noise to the blocks (geologic noise), which amounts to removing

Ktovej inverzná verzia je daná vzťahom :

$$\mathbf{W}_e^{-1} = \ell_0^2 \text{diag} (\mathbf{A} \mathbf{W}_v^{-1} \mathbf{A}^T), \quad (11)$$

$$\mathbf{x} = \left( \mathbf{W}^{-1} \cdot \mathbf{A}^T \cdot \mathbf{A} \right)^{-1} \cdot \mathbf{W}^{-1} \cdot \mathbf{A}^T \cdot \mathbf{b} \quad \text{základná rovnica (v maticovom tvare)}$$


---

Hlavný vzťah pre riešenie (pre k-tý krok iterácie) je daný vzťahom:

### COMPUTATIONAL PROCEDURE

As we have already seen,  $\mathbf{W}_v$  and  $\mathbf{W}_e$  are not constant weighting matrices. Thus the computational procedure is an iterative one, with the application of equation (9) at each step. The  $k$ th step in the process can be written

$$\hat{\mathbf{V}}^{(k)} = [\mathbf{W}_v^{(k-1)}]^{-1} \mathbf{A}^T \{ \mathbf{A} [\mathbf{W}_v^{(k-1)}]^{-1} \mathbf{A}^T + [\mathbf{W}_e^{(k-1)}]^{-1} \}^{-1} \mathbf{G}. \quad (15)$$

Pozn.: Váhová matica  $\mathbf{W}_v$  začína pri prvej iterácii ako jednotková matica  $\mathbf{I}$ , neskôr je už upravovaná cez vzorec  $w_j = (\sigma_j^2 + \varepsilon)^{-1}$ .  
 ( $\varepsilon$  je malé číslo, napr.  $10^{-6}$ ,  $\sigma_j$  sú hustoty s predchádzajúcej iterácie)

# IIR filter - Hilbertova transformácia

(prepočet jednej kolmej zložky na druhú):

IIR filtre (tzv. Infinite Impulse Response) slúžia na mnohé transformácie zložiek geofyzikálnych polí alebo signálu. IIR filtre nemajú konečnú dĺžku (ako tomu bolo napr. pri car\_box filtroch), ale nekonečnú dĺžku. V praxi sa ich dĺžka rovná presne dĺžke profilových dát (matici), ktorú ideme transformovať.

Hilbertova transformácia prepočítava ortogonálne zložky poľa (napr. v magnetometrii zložku H na Z alebo v gravimetrii pole  $V_{xz}$  na  $V_{zz}$ , a naopak).

Výpočet (koeficienty filtra v priestorovej oblasti):

Koeficienty IIR filtra:

$$f_v^{(H)} = \begin{cases} 0 & \text{pre } v = 0, \pm 2, \pm 4, \dots \text{ (v párne),} \\ 2/(\pi v) & \text{pre } v = \pm 1, \pm 3, \pm 5, \dots \text{ (v nepárny).} \end{cases}$$

Využite jednoduchú konvolučnú sumáciu:  $V_{zz_i} = \sum_{j=1}^n f_{j-i}^{(H)} V_{xz_j}$ , (pre  $i = 1, 2, \dots, n$ ).

Zdroj – učebnica:

Gutdeutsch R., 1986: Anwendungen der Potentialtheorie auf geophysikalische Felder. Hochschultext, Springer, s.83.



# IIR filter - Hilbertova transformácia

(prepočet jednej kolmej zložky na druhú):

Úloha:

Načítajte z pripojeného súboru (*profilove\_udaje.dat*) hodnoty zložky poľa  $V_{xz}$  a transformujte ich pomocou uvedeného IIR filtra (koeficienty sú uvedené dolu) na pole  $V_{zz}$ . Následne porovnajete váš výsledok s teoretickými hodnotami poľa  $V_{zz}$ , ktoré je tiež uvedené v súbore (súbor má 3 stĺpce:  $x$ ,  $V_{xz}$ ,  $V_{zz}$ ). Získané výsledky zobrazte v grafoch.

