# NLA: prva domača naloga

**Rok oddaje: 3. maj 2017**

Na spletno učilnico oddajte `.zip` arhiv (ali kaj podobnega), ki vsebuje vse datoteke (skripta in slike), s katerimi ste prišli do rezultatov, rezultate in poročilo, v katerem jih opišete.

Metode, ki jih implementirate, naj bodo zapisane v skriptih kot funkcije z ustreznimi argumenti. Poleg tega dodajte še skripta, v katerih te metode uporabite na konkretnih podatkih (oglej si primer DN na učilnici).

Sledi opis ozadja problema, ki ga rešujemo. Nato pokažemo, kako določene stvari izvedemo v Matlabu. Zaključimo z opisom nalog.

## Ozadje problema

Ukvarjali se bomo z ostrenjem slike $\widehat{B}$, ki je zaradi atmosferskih motenj ter dodatnega naključnega šuma zamegljena podoba prave slike $X$. Atmosferskim motnjam ustreza t. i. Gaussova zameglitev. Izberemo jo zato, ker lahko v tem primeru končno zameglitev razcepimo na zameglitev po stolpcih in vrsticah. Ko si izberemo matriko zameglitve $A$, lahko torej najdemo taki matriki $A_c$ in $A_r$, da velja

$$A \operatorname{vec}(X) = \operatorname{vec}(A_c X A_r^T). \tag{1}$$

Tehnično podrobnost, zaradi katere smo izbrali $A_r$ namesto $A_r^T$, bomo kmalu pojasnili. Ker matriki $X \in \mathbb{R}^{m\times n}$ s stolpci $x_i$ pripada vektor

$$\operatorname{vec}(X) = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^{mn},$$

je matrika zamegljevanja $A \in \mathbb{R}^{mn\times mn}$ lahko zelo velika: za sliko velikosti $500\times500$ dobimo matriko $A$ velikosti $250000\times250000$, kar onemogoči direktno računanje z njo.

Rezultat zamegljevanja je matrika $B = A_c X A_r^T$. Dodali ji bomo še nekaj šuma, tako da izberemo parameter $\sigma$ in definiramo

$$\widehat{B} = B + E,$$

kjer so elementi matrike $E$ realizacije neodvisnih slučajnih spremenljivk, porazdeljenih kot $N(0, \sigma)$. Poskrbite, da se ob zagonu ustrezne `.m` datoteke vedno zgenerirala ista matrika $E$ (to omogoča preverljivost rezultatov in olajša odkrivanje hroščev v kodi). Pri tem vam lahko koristi funkcija `rng`.

Zaradi dodanega šuma naivna rešitev

$$\widehat{X} = A_c^{-1}\widehat{B}A_r^{-T} = A_c^{-1}(B+E)A_r^{-T} = X + A_c^{-1}EA_r^{-T} \tag{2}$$

ni dobra. Če najdemo singularna razcepa

$$A_c = U_c\Sigma_c V_c^T = \begin{bmatrix} u_{c,1} & \cdots & u_{c,m} \end{bmatrix} \operatorname{diag}(\sigma_{c,1}, \ldots, \sigma_{c,m}) \begin{bmatrix} v_{c,1} & \cdots & v_{c,m} \end{bmatrix}^T$$

in

$$A_r = U_r\Sigma_r V_r^T = \begin{bmatrix} u_{r,1} & \cdots & u_{c,n} \end{bmatrix} \operatorname{diag}(\sigma_{r,1}, \ldots, \sigma_{r,n}) \begin{bmatrix} v_{r,1} & \cdots & v_{r,n} \end{bmatrix}^T,$$

potem lahko (2) zapišemo kot

$$\begin{aligned} \widehat{X} &= \left( \sum_{i=1}^{m} \frac{1}{\sigma_{c,i}} v_{c,i} u_{c,i}^T \right) \widehat{B} \left( \sum_{j=1}^{n} \frac{1}{\sigma_{r,j}} u_{r,j} v_{r,j}^T \right) \\ &= \sum_{i=1}^{m}\sum_{j=1}^{n} \underbrace{\frac{u_{c,i}^T \widehat{B} u_{r,j}}{\sigma_{c,i}\sigma_{r,j}}}_{\widehat{\beta}_{ij}} \underbrace{v_{c,i} v_{r,j}^T}_{V_{ij}}. \end{aligned}$$

Lahko je preveriti, da iz ortogonalnosti $V_c$ in $V_r$ sledi $\|V_{ij}\|_F = 1$, zato koeficienti $\widehat{\beta}_{ij}$ ne smejo biti preveliki. Ker je $\widehat{B} = B + E$, lahko koeficient $\widehat{\beta}_{ij}$ zapišemo kot vsoto $\widehat{\beta}_{ij} = \beta_{ij} + \varepsilon_{ij}$, kjer je

$$\beta_{ij} = \frac{u_{c,i}^T B u_{r,j}}{\sigma_{c,i}\sigma_{r,j}} \quad \text{in} \quad \varepsilon_{ij} = \frac{u_{c,i}^T E u_{r,j}}{\sigma_{c,i}\sigma_{r,j}}.$$

Ker so lahko produkti $\sigma_{c,i}\sigma_{r,j}$ zelo majhni, hkrati pa se zaradi naključnosti šuma vrednost $|u_{c,i}^T E u_{r,j}|$ ne spreminja močno z $i$ in $j$, lahko dobimo pri naivnem reševanju povsem zmoteno rešitev. Zato - podobno kot smo to storili pri regularizaciji rešitve $Ax = b$ v vektorskem primeru - vpeljemo faktorje regularizacije $\phi_{ij}$ in raje zapišemo

$$\widehat{X} = \sum_{i=1}^{m}\sum_{j=1}^{n} \phi_{ij}\widehat{\beta}_{ij} V_{ij}. \tag{3}$$

Za to, da postopamo analogno kot v vektorskem primeru, imamo torej dobro praktično motivacijo, vendar je vse tudi teoretično utemeljeno. Izkaže se, da za matriko zamegljevanja $A$ in njena faktorja $A_c$ ter $A_r$ velja zveza

$$A = A_r \otimes A_c,$$

kjer je Kroneckerjev produkt $\otimes$ med splošnima matrikama $C \in \mathbb{R}^{m\times n}$ in $D \in \mathbb{R}^{p\times q}$ definiran kot

$$C \otimes D = \begin{bmatrix} c_{11}D & \cdots & c_{1n}D \\ \vdots & & \vdots \\ c_{m1}D & \cdots & c_{mn}D \end{bmatrix}.$$

Z neposrednim računom se da hitro prepričati, da je $\otimes$ povezan s standardnimi matričnimi operacijami preko naslednjih zvez:

a) $(A \otimes B)\mathrm{vec}(X) = \mathrm{vec}(BXA^T)$,

b) $(A_1 \otimes B_1)(A_2 \otimes B_2) = (A_1A_2) \otimes (B_1B_2)$,

c) $(A \otimes B)^T = A^T \otimes B^T$,

d) $(A \otimes B)^{-1} = A^{-1} \otimes B^{-1}$.

Zveza a) nam omogoča računanje z matriko zameglitve $A$, če poznamo njena faktorja $A_c$ in $A_r$. Tiho smo jo uporabili že pri (1) in sedaj je tudi jasno, zakaj dobimo na desni $A_r^T$. Iz zvez b) in c) lahko izpeljemo

$$(U_r\Sigma_rV_r^T) \otimes (U_c\Sigma_cV_c^T) = (U_r \otimes U_c)(\Sigma_r \otimes \Sigma_c)(V_r \otimes V_c)^T.$$

Lahko se je prepričati, da je $A = (U_r \otimes U_c)(\Sigma_r \otimes \Sigma_c)(V_r \otimes V_c)^T$ singularni razcep matrike $A$, vendar moramo paziti, saj diagonalni elementi $\sigma_{ij}$ v matriki $\Sigma_r \otimes \Sigma_c$, ki so produkti singularnih vrednosti $A_r$ in $A_c$, **niso** urejeni po velikosti. S pomočjo zveze d) pridemo do (izpeljave) rešitve (3).

## 0.1 Filtri regularizacije

Neurejenost singularnih vrednosti $\sigma_{ij}$ pri regularizaciji Tihonova ne pride do izraza, še vedno definiramo filtre kot

$$\phi_{ij} = \frac{\sigma_{ij}^2}{\sigma_{ij}^2 + \alpha^2}$$

za neko izbrano vrednost parametra $\alpha$.

Tudi pri odrezanem singularnem razcepu neurejenosti ni težko zaobiti. Namesto indeksa $k$, od koder naprej so v standardnem primeru vsi filtri enaki 0, postavimo ekvivalenten pogoj

$$\phi_{ij} = \begin{cases} 1 & ; \quad \sigma_{ij} \geq t \\ 0 & ; \quad \text{sicer} \end{cases} \tag{4}$$

pri izbrani vrednosti parametra $t$.

## 0.2 Delo z Matlabom

Oglejmo si nekatere uporabne funkcije za delo s slikami ter nekatere druge, ki nam bodo koristile.

### 0.2.1 Branje in shranjevanje slik

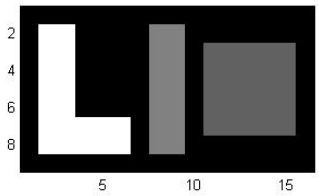
Slike so predstavljene kot matrike. Če v $A$ shranimo matriko

$$\begin{bmatrix} 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&8&8&0&0&0&0&4&4&0&0&0&0&0&0&0 \\ 0&8&8&0&0&0&0&4&4&0&3&3&3&3&3&0 \\ 0&8&8&0&0&0&0&4&4&0&3&3&3&3&3&0 \\ 0&8&8&0&0&0&0&4&4&0&3&3&3&3&3&0 \\ 0&8&8&0&0&0&0&4&4&0&3&3&3&3&3&0 \\ 0&8&8&8&8&8&0&4&4&0&3&3&3&3&3&0 \\ 0&8&8&8&8&8&0&4&4&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \end{bmatrix},$$

v Matlabu z ukazi

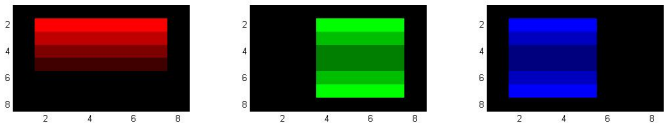
```
imagesc(A), axis image, colormap(gray)
```

dobimo

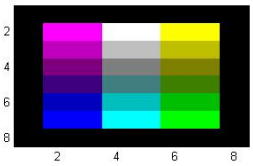



Na podoben način so barvne slike predstavljene s tridimenzionalnimi matrikami.

$$R = \begin{bmatrix} 0&0&0&0&0&0&0&0 \\ 0&8&8&8&8&8&8&0 \\ 0&6&6&6&6&6&6&0 \\ 0&4&4&4&4&4&4&0 \\ 0&2&2&2&2&2&2&0 \\ 0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0 \end{bmatrix}, \quad G = \begin{bmatrix} 0&0&0&0&0&0&0&0 \\ 0&0&0&8&8&8&8&0 \\ 0&0&0&6&6&6&6&0 \\ 0&0&0&4&4&4&4&0 \\ 0&0&0&4&4&4&4&0 \\ 0&0&0&6&6&6&6&0 \\ 0&0&0&8&8&8&8&0 \\ 0&0&0&0&0&0&0&0 \end{bmatrix}, \quad B = \begin{bmatrix} 0&0&0&0&0&0&0&0 \\ 0&8&8&8&8&0&0&0 \\ 0&6&6&6&6&0&0&0 \\ 0&4&4&4&4&0&0&0 \\ 0&4&4&4&4&0&0&0 \\ 0&6&6&6&6&0&0&0 \\ 0&8&8&8&8&0&0&0 \\ 0&0&0&0&0&0&0&0 \end{bmatrix}$$



```
X(:,:,1)=R/8; X(:,:,2)=G/8; X(:,:,3)=B/8; imagesc(X)
```




Zunanje slike preberemo v Matlabu s pomočjo ukaza `imread`. Tako npr. z

```
A = imread('butterflies.tif');
imagesc(A)
axis image
```

v Matlabu dobimo tridimenzionalno matriko $A \in \mathbb{R}^{474\times852\times3}$, ki predstavlja sliko

Če smo prebrali črnobelo sliko, uporabimo še ukaz `colormap gray`, da bo narisana slika res črnobela (ukaz spremeni zgolj prikaz slike in ne slike same).

Za pretvorbo barvne slike `A` v črnobelo uporabimo klic `rgb2gray(A)`. Če ni na voljo, potem lahko uporabimo ukaz

```
CB = 0.2989 * A(:,:,1) + 0.5870 * A(:,:,2) + 0.1140 * A(:,:,3);
```

(glej dokumentacijo za `rgb2gray`). Tipično so števila $a_{ij}$ v matriki $A$ tipa `int` in zavzemajo vrednosti $0 \leq a_{ij} \leq 255$. Če želimo sliki prišteti nekaj šuma, ki je po navadi realna matrika, in rezultat shraniti, lahko to storimo tako, da

1) pretvorimo števila $a_{ij}$ v realna: ukaz `A = im2double(A)` opravi transformacijo $a_{ij} \mapsto a_{ij}/255$,

2) prištejemo šum $A = A + E$,

3) shranimo novo sliko z ukazom `imwrite(A, 'mojaNovaSlika.jpg')`.

Opazimo lahko, da bo ukaz `imread('mojaNovaSlika.jpg')` vseeno vrnil matriko z elementi tipa `int`.

### 0.2.2 Paket HNO

Algoritmi za zameglitev slik v Matlabu so na voljo v paketu HNO, ki ga je pripravil Per Christian Hansen. Paket dobimo v datoteki `HNO.zip`, ki se nahaja na naslovu

`http://www.imm.dtu.dk/~pcha/HNO/`

Denimo, da imate sliko velikosti $m \times n$. Matriki $A_r$ in $A_c$, na kateri razcepimo Gaussovo zameglitev, dobite v Matlabu z ukazi:

```
[PSF, center] = psfGauss([n,m],s);
[Ar, Ac] = kronDecomp(PSF, center);
```

Tu je $s$ parameter Gaussove zameglitve (večji ko je, bolj je slika zamegljena, normalna vrednost je npr. 2). Ukaz

```
imagesc(PSF)
```

prikaže, kako zameglitev, ki je podana z matriko `PSF` (point spread function), deluje na sliko, ki ima en sam bel piksel v sredini. Zameglitev potem na isti način razmaže vse piksle v sliki, ukaz `kronDecomp` pa iz `PSF` izračuna ustrezni matriki $A_r$ in $A_c$.

Pomagate si lahko tudi z ukazoma `tsvd_sep` in `tik_sep`, katerih uporabo raziščite sami.

### 0.2.3 Razno

Singularni razcep dobimo z ukazom `[U, S, V] = svd(A)`. Če se želite poigrati z vektorizacijo matričnih enačb, sta koristna npr. `reshape`[1] (za funkcijo vec) in `kron` (za izračun $\otimes$).

Regularizirano rešitev $\widehat{X}$ iz (3) lahko dobimo tako:

1) Izračunamo števila $\phi_{ij}\widehat{\beta}_{ij}$ in jih shranimo v matriko $M$:

   - izračunamo $W = \Sigma_c^{-1} U_c^T \widehat{B} U_r \Sigma_r^{-1} \in \mathbb{R}^{m \times n}$: `W = Sc\Uc'*B*Ur/Sr`,
   - izberemo filtre $\phi_{ij}$ in jih shranimo v matriko `F`,
   - izračunamo produkt po komponentah: `M = F .* W`,

2) Izračunamo še `Xreg = Vc * M * Vr'`.

[1] V Octavu `vec`.

Za konec dodajmo še naslednje:

- Kubični polinomski zlepek razreda $\mathcal{C}^2(\mathbb{R})$ dobimo s pomočjo funkcije `spline`[2]. Odvod polinoma lahko dobimo s `polyder`.
- Podatke iz datotek tipa `<ime>.mat` naložimo z ukazom `load <ime>.mat`.
- Ukaz `[urejen, indeksi] = sort(seznam)` vrne seznama `urejen` in `indeksi`, za katera velja

  $$\texttt{urejen(i)} \leq \texttt{urejen(i + 1)} \quad \text{in} \quad \texttt{seznam(indeksi) = urejen},$$

  kar bomo potrebovali pri nalogi 2 pri urejanju singularnih vrednosti $\sigma_{ij}$ in izrisovanju števil $u_{c,i}^T \widehat{B} u_{r,j}$ v skladu s to ureditvijo.

# Osnovni opis naloge

Ukvarjali se bomo s tremi slikami: dvema, ki ju dobimo iz datotek `tabla.mat` in `skrivnost.mat`, ter eno, ki si jo izberete sami (naj bo ostra in primerne velikosti, lahko je črnobela). V `.mat` datotekah se poleg zamegljene črnobele slike, ki je shranjena v spremenljivki `B`, nahajajo še matrike `Ac` in `Ar` ter `PSF` in `center`.

Pri opisu sledečih nalog bomo zapis nekoliko standardizirali ter iskanje začetne matrike $X$, ki predstavlja sliko, obravnavali kot reševanje enačbe $A\boldsymbol{x} = \boldsymbol{b}$, pri čemer je $A$ naša matrika zamegljevanja in $\boldsymbol{b} = \widehat{B}$.

V nalogah bomo na različne načine skušali priti do faktorjev filtra $\phi_i$ in rešitev zapisali kot

$$\boldsymbol{x} = \sum_i \phi_i \frac{u_i^T \boldsymbol{b}}{\sigma_i} v_i,$$

kjer vektorje $v_i$ in $u_i$ ter števila $\sigma_i$ dobimo iz singularnega razcepa matrike $A$.

# 1 Regularizacija Tihonova

Začetno sliko bomo poiskali s pomočjo regularizacije Tihonova in poskušali najti optimalno vrednost $\alpha \geq 0$ za minimizacijski problem

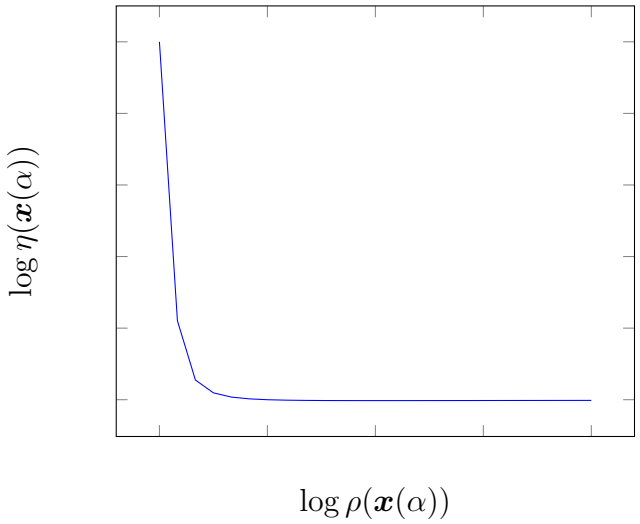
$$\min_{\boldsymbol{x}} \|A\boldsymbol{x} - \boldsymbol{b}\|_2^2 + \alpha^2 \|\boldsymbol{x}\|_2^2 . \tag{5}$$

[2] V Octavu `interp1` z dodatnim argumentom `'spline'`.

Vidimo, da gre pri gornjem problemu za iskanje ravnotežja med členoma $\eta(\boldsymbol{x}) = \|\boldsymbol{x}\|_2^2$ ter $\rho(\boldsymbol{x}) = \|A\boldsymbol{x} - \boldsymbol{b}\|_2^2$. Pri majhnih vrednostih $\alpha$ bo $\rho$ majhen in $\eta$ sorazmerno velik. Z večanjem vrednosti $\alpha$ pričakujemo rast $\rho$ in padanje $\eta$. Če za vse $\alpha \in [0, \infty)$ izračunamo rešitev problema (5) in jo označimo z $\boldsymbol{x}(\alpha)$, potem dobimo t. i. L-krivuljo, podano s parametrizacijo

$$L(\alpha) = (\log \rho(\boldsymbol{x}(\alpha)), \log \eta(\boldsymbol{x}(\alpha))).$$

Ime je dobila po svoji značilni obliki, ki je razvidna s slike 1. Namesto $\rho$ in $\eta$ narišemo raje njuna logaritma, saj je sta tako repa bolj poudarjena in je laže najti *točko preloma*, ki jo iščemo. Točka preloma bomo definirali kot točko,




Slika 1: L-krivulja (slika je simbolična)

v kateri ima $L(\alpha)$ največjo ukrivljenost $\kappa(\alpha)$. Vpeljimo oznaki

$$\widehat{\rho} = \log \rho \quad \text{in} \quad \widehat{\eta} = \log \eta,$$

pri čemer smo opustili pisanje argumenta $\alpha$. Spomnimo se, da je

$$\kappa = \frac{|\widehat{\rho}'\widehat{\eta}'' - \widehat{\rho}''\widehat{\eta}'|}{\|(\widehat{\rho}, \widehat{\eta})'\|_2^3}. \tag{6}$$

Da bi lahko uporabili zgornjo formulo, moramo dokazati naslednji enakosti, v katerih smo vpeljali $\beta_i = u_i^T \boldsymbol{b}$.

$$\eta(\alpha) = \|\boldsymbol{x}(\alpha)\|_2^2 = \sum_i \frac{\sigma_i^2 \beta_i^2}{(\alpha^2 + \sigma_i^2)^2} \tag{7}$$

$$\rho(\alpha) = \|A\boldsymbol{x}(\alpha) - \boldsymbol{b}\|_2^2 = \|r_\perp\|_2^2 + \sum_i \frac{\alpha^4 \beta_i^2}{(\alpha^2 + \sigma_i^2)^2} \tag{8}$$

Vektor $r_\perp$ je definiran kot $r_\perp = \boldsymbol{b} - \sum_i \beta_i u_i$, tj. ostanek iz ortogonalnega komplementa podprostora, ki ga razpenjajo $u_i$, ki bi ga dobili pri metodi najmanjših kvadratov oz. pri $\alpha = 0$.

## 1.1 Predpriprava

Dokažite zvezi (7) in (8) in izračunajte (prve in druge) odvode, ki nastopajo v (6). Verjetno bo koristno, da izračun $\kappa(\alpha)$ shranite kot funkcijo v neko `.m` datoteko, saj bo to sestavni del vseh sledečih podnalog te naloge.

## 1.2 Neposreden izračun ukrivljenosti

Najdite točko $\alpha_{\max}$, v kateri je dosežena maksimalna ukrivljenost, tako da maksimizirate (6).

## 1.3 Približen izračun

V praksi se lahko zgodi, da je minimizacijski problem, ki ga rešujemo v (5), časovno zahteven, zato bi radi prišli do približka za $\alpha_{\max}$ z malo izračuni $r(\alpha)$. Možen algoritem je sledeč:

**Algoritem 1** približni $\kappa$

1: izračunaj nekaj točk $L(\alpha_i)$ navpičnega dela L-krivulje, $1 \leq i \leq N_1$
2: izračunaj nekaj točk $L(\alpha_i)$ vodoravnega dela L-krivulje, $N_1 < i \leq N$
3: $S \leftarrow \{(\alpha_i, L(\alpha_i)) \mid 1 \leq i \leq N\}$
4: **for** $k = 1, 2, \ldots$ **do**
5: $\quad z_S \leftarrow$ kubični zlepek skozi točke iz $S$
6: $\quad \alpha_{\max}^k \leftarrow \alpha$, pri kateri ima krivulja, ki jo določa $z_S$, max ukrivljenost
7: $\quad \kappa_k = \kappa_{z_S}(\alpha_{\max}^k)$
8: $\quad$ dodaj $(\alpha_{\max}^k, L(\alpha_{\max}^k))$ v $S$
9: **end for**

Kubični zlepek $z_S$ je torej oblike

$$z_S : \alpha \to (\bar{\rho}(\alpha), \bar{\eta}(\alpha)),$$

torej najdemo približek ločeno za $\widehat{\rho}$ in $\widehat{\eta}$. Naj bo dvakrat zvezno odvedljiv, saj je cilj izračunati približek za $\kappa$.

Zaustavitveni pogoj ter število izračunanih točk na začetku izberite sami. Lahko preizkusite različne možnosti. Pri izbiri izhajajte iz tega, da je izračun $L(\alpha)$ najdražja operacija v algoritmu.

# 2 Odrezani singularni razcep

Tu faktorje filtra $\phi_i$ definiramo kot

$$\phi_i = \mathbf{1}[i \leq K],$$

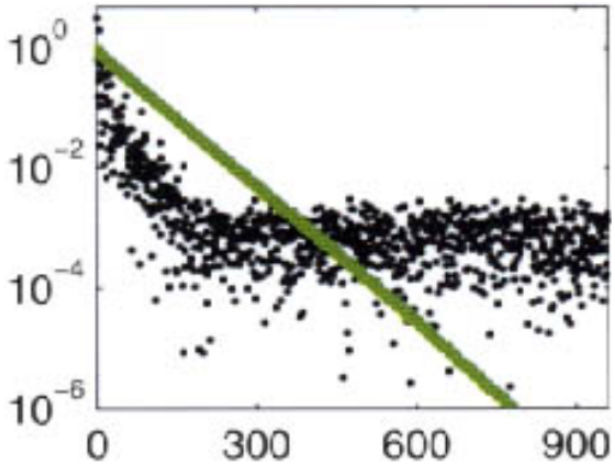
kjer je $\mathbf{1}$ indikatorska funkcija. Vemo, da je

$$\widehat{\boldsymbol{x}} = A^{-1}\widehat{\boldsymbol{b}} = A^{-1}\boldsymbol{b} + A^{-1}\boldsymbol{e} = \boldsymbol{x} + A^{-1}\boldsymbol{e},$$

kje je $\boldsymbol{e}$ vektor šuma in $\boldsymbol{b}$ zamegljena slika začetne slike $\boldsymbol{x}$. Ker je vektor $\boldsymbol{e}$ naključen in je

$$A^{-1}\boldsymbol{e} = \sum_i \frac{u_i^T \boldsymbol{e}}{\sigma_i} v_i,$$

so vsi skalarni produkti $u_i^T \boldsymbol{e}$ približno enako veliki, zato bodo pri majhnih $\sigma_i$ koeficienti $u_i^T \boldsymbol{e}/\sigma_i$ veliki in bo pravi $\boldsymbol{x}$ izgubljen (po navadi namreč $|u_i^T \boldsymbol{b}|$ pada). Želimo si torej $K$, pri katerem bomo v rešitev vključili dovolj komponent, hkrati pa bo dovolj majhen, da šum ne bo prišel pretirano do izraza. (S slike 2 je razvidno, da bo od nekega $i$ naprej šum popolnoma zasenčil pravo sliko.)




Slika 2: Singularne vrednosti (zelene pike) ter vrednosti $|u_i^T \widehat{\boldsymbol{b}}|$ v odvisnosti od $i$. Vir: Str. 60 v Per Christian Hansen, James G. Nagy, Dianne O. O'Leary. *Deblurring Images, Matrices, Spectra and Filtering*, Society for Industrial and Applied Mathematics. Philadelphia, PA, ZDA.

Zapišite vsaj en algoritem, ki iz velikosti skalarnih produktov $|u_i^T \widehat{\boldsymbol{b}}|$ ter singularnih vrednosti $\sigma_i$ najde smiseln $K$, pri katerem opravimo odrez. Preizkusite ga na izbranih slikah. Je morda mogoče *na roke* dobiti boljše rezultate?

Spomnimo, da v resnici namesto $K$ iščemo mejo $t$ iz enačbe (4).

# 3 Obdelava rezultatov

Poženite algoritme in najdite rešitve za različne stopnje šuma in zamegljevanja (kjer matriki $A_r$ in $A_c$ nista že določeni). Rezultate prikažite tudi grafično. Pri regularizaciji Tihonova analogu slike 1 dodajte še točko, ki ustreza optimalnemu $\alpha$, ki ste ga našli.

Pri odrezanem singularnem razcepu pa lahko analogu slike 2 dodate razmejitveno črto, ki označuje mesto odreza. Tu bo treba singularne vrednosti $\sigma_{ij}$ matrike $A$ urediti.

- Do kako velike stopnje šuma oz. zamegljevanja še dobimo smiselne rezultate?
- Ali algoritmi vračajo občutno slabše rezultate pri velikih motnjah?
- Kakšne so kvalitete rešitev, ki jih dobimo pri prvi in drugi nalogi? Se močno razlikujejo?
- Ali manjša razlika $\left\|\hat{X} - X\right\|$ med začetno sliko in najdeno rešitvijo vedno pomeni tudi večjo kvaliteto slike $\widehat{X}$?