Prelucrarea avansată a semnalelor

Capitolul 2: Reprezentări rare

Bogdan Dumitrescu

Facultatea de Automatică şi Calculatoare

Universitatea Politehnica Bucureşti

Cuprins

- Soluţii rare ale sistemelor liniare: definiţii, unicitate
- Soluţii rare aproximative
- Algoritmi lacomi (greedy)
  - Matching pursuit
  - Algoritmi cu ortogonalizare
- Relaxări convexe (basis pursuit)
- CMMP reponderate iterativ
- Proiectarea dicţionarului (bazei)
- Exemple de aplicaţii

Problema

- Problema studiată în acest capitol: găsirea unei soluţii rare pentru sistemul liniar $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$
- Matricea $\boldsymbol{A}$ are dimensiune $m \times N$
- De obicei $m < N$, adică sistemul este subdeterminat şi, în general, are o infinitate de soluţii
- Soluţie rară: $\boldsymbol{x}$ are "puţine" elemente nenule
- Soluţia cea mai rară: cea cu cel mai mic număr de elemente nenule
- Orice sistem liniar subdeterminat are (cel puţin) o cea mai rară soluţie, dar ea nu este obligatoriu rară, adică poate avea multe elemente nenule

Interpretare

- Coloanele matricei $\boldsymbol{A}$ sunt vectori într-o *bază supracompletă* (numită şi dicţionar)
- Aici *bază* se foloseşte în sensul larg, de mulţime de vectori reprezentativi
- Căutăm reprezentarea vectorului $\boldsymbol{b}$ cu un număr cât mai mic de vectori din bază
- Motiv: în realitate, multe semnale utile, transformate în spaţii adecvate (de exemplu cu DCT sau DWT) au reprezentări (aproximativ) rare în cel puţin unul din acele spaţii
- Reprezentarea rară produce informaţie condensată asupra lui $\boldsymbol{b}$, care poate fi folosită pentru compresie, eliminarea zgomotului, etc.
- Concatenând mai multe baze (DCT$+$DWT, de exemplu), se obţine un dicţionar care poate acoperi mai multe categorii de semnale

Comentarii

- Problema găsirii celei mai rare soluţii este NP-completă, deci în esenţă necesită căutare exhaustivă
- În practică, problema se relaxează (vom vedea mai târziu cum) la căutarea unei soluţii rare aproximative ($\boldsymbol{A}\boldsymbol{x} \approx \boldsymbol{b}$)
- Cercetarea recentă (problema aproape că nu a fost studiată înainte de anul 1990) are două direcţii majore
  - (Teorie:) formularea unor condiţii de unicitate a soluţiei celei mai rare (i.e. optimă) şi de evaluare a optimalităţii unei soluţii date
  - (Implementare:) descrierea unor algoritmi rapizi (i.e. în timp polinomial) care să găsească soluţia cea mai rară suficient de des

”Norma” 0

- "Norma" ℓ_0, notată $\|\boldsymbol{x}\|_0$, este numărul de elemente nenule ale vectorului $\boldsymbol{x}$
- Nu este o normă pentru că nu satisface inegalitatea $\|\alpha\boldsymbol{x}\| = \alpha\|\boldsymbol{x}\|$, $\forall\alpha \geq 0$
- Într-adevăr, $\|\alpha\boldsymbol{x}\|_0 = \|\boldsymbol{x}\|_0$ pentru $\alpha \neq 0$ (înmulţirea cu un scalar nu modifică numărul de elemente nenule ale unui vector)
- Altfel, inegalitatea triunghiului $\|\boldsymbol{x} + \boldsymbol{y}\|_0 \leq \|\boldsymbol{x}\|_0 + \|\boldsymbol{y}\|_0$ este satisfăcută în general (suma a doi vectori nu poate avea mai multe elemente nenule decât fiecare vector în parte)
- Mai mult, funcţia $\|\cdot\|_0$ este discontinuă, deci nu este convexă

Unicitate—rezultate teoretice

- $spark(\boldsymbol{A})$: numărul minim de coloane liniar dependente ale matricei $\boldsymbol{A}$
- Pentru multe baze (de exemplu generate aleator), $spark(\boldsymbol{A}) = m + 1$ (adică orice n coloane sunt liniar independente)
- Teoremă: dacă o soluţie a sistemului $\boldsymbol{Ax} = \boldsymbol{b}$ satisface $\|\boldsymbol{x}\|_0 < spark(\boldsymbol{A})/2$, atunci este cea mai rară
- Demonstraţie: fie $\boldsymbol{y}$ o soluţie mai rară. Atunci $\|x - y\|_0 < spark(\boldsymbol{A})$. Dar $\boldsymbol{A}(\boldsymbol{x} - \boldsymbol{y}) = 0$, ceea ce contrazice definiţia $spark(\boldsymbol{A})$
- Utilitatea rezultatului este limitată: calcularea $spark(\boldsymbol{A})$ este mai dificilă decât rezolvarea sistemului

Coerenţă mutuală

- Coerenţa mutuală a unei matrice $\boldsymbol{A}$ este

$$\mu(\boldsymbol{A}) = \max_{i \neq j} \frac{|\boldsymbol{a}_i^T \boldsymbol{a}_j|}{\|\boldsymbol{a}_i\| \cdot \|\boldsymbol{a}_j\|}$$

- E o măsură a celui mai mic unghi între două coloane ale matricei (cu atât mai aproape de 1 cu cât unghiul e mai mic)
- Cu cât N creşte (m fiind constant), $\mu(\boldsymbol{A})$ se apropie de 1 (spre deosebire de $spark$, care poate rămâne $m+1$)
- Se poate demonstra că

$$spark(\boldsymbol{A}) \geq 1 + \frac{1}{\mu(\boldsymbol{A})}$$

- Deşi dă margini mai pesimiste privind raritatea, coerenţa mutuală se poate calcula uşor

Problemă de optimizare

- Vectorul $\boldsymbol{x} \in \mathbb{R}^N$ este rar dacă $\|\boldsymbol{x}\|_0 \ll N$ (pe noi ne interesează ca $\|\boldsymbol{x}\|_0 \ll m$)
- Găsirea soluţiei celei mai rare poate fi formulată ca o problemă de optimizare

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \|\boldsymbol{x}\|_0 \\ c.r. \quad & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \end{aligned} \tag{1}$$

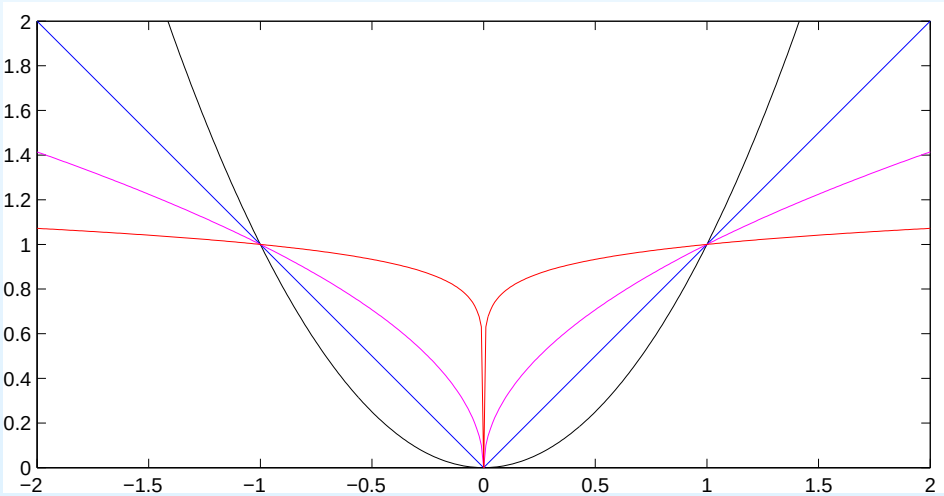
- Asta nu simplifică problema, dar poate conduce la aproximarea ei
- Problema (1) nu este convexă
- Mai rău, funcţia $\|\boldsymbol{x}\|_0$ este discontinuă

Normele p

- Norma 0 poate fi văzută ca o limită a normelor p

$$\|\boldsymbol{x}\|_0 = \lim_{p \to 0} \|\boldsymbol{x}\|_p^p = \lim_{p \to 0} \sum_{k=1}^{m} |x_k|^p$$

- Pentru $0 < p < 1$, norma p nu mai este o funcţie convexă, deci de fapt nu mai este o normă
- Graficul lui $|x|^p$, pentru $p = 2, 1, 0.5, 0.1$ (negru, albastru, magenta, roşu)



Soluţii aproximative

- În practică, datele sunt afectate de erori, deci, chiar dacă sistemul $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ ar avea o soluţie rară, ea n-ar putea fi găsită
- De aceea se modifică problema (1) în

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \|\boldsymbol{x}\|_0 \\ c.r. \quad & \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\| \leq \delta \end{aligned} \tag{2}$$

- De obicei, toleranţa δ este aleasă empiric
- Tipic se utilizează norma 2 în restricţia din (2)
- Problema nu se simplifică, adică e la fel de greu de rezolvat ca (1), şi este mai greu de caracterizat teoretic

Tipuri de algoritmi

- Două categorii mari de algoritmi pentru rezolvarea problemei (2), algoritmi lacomi şi aproximări (relaxări) convexe
- Algoritmii lacomi (greedy) selectează coloanele lui $\boldsymbol{A}$ (vectorii bazei) una câte una, de fiecare dată alegând-o pe cea mai "promiţătoare"
- Algoritmi: matching pursuit, orthogonal matching pursuit, greedy LS
- Relaxările convexe se obţin înlocuind norma 0 cu o funcţie convexă, de exemplu cu $\|\boldsymbol{x}\|_1$ (care e cea mai apropiată normă convexă)
- Algoritmi: basis pursuit

Matching pursuit—ideea

- Pentru simplitate (şi fără pierderea generalităţii), presupunem $\|\boldsymbol{a}_i\|_2 = 1$, adică vectorii din bază sunt normaţi
- Algoritmul matching pursuit (MP) a fost propus în 1993 de Mallat & Zhang, dar era cunoscut dinainte în alte domenii
- MP construieşte iterativ mulţimea indicilor $\mathcal{I}$ ai elementelor nenule, adăugând câte un indice la fiecare iteraţie
- Se porneşte de la soluţia nulă ($\mathcal{I} = \emptyset$), pentru care reziduul sistemului $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ este $\boldsymbol{r} \stackrel{def}{=} \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$
- Ideea MP: coloana care aproximează cel mai bine reziduul este cea care face unghiul cel mai mic cu acesta, deci este coloana k pentru care

$$|\boldsymbol{a}_k^T \boldsymbol{r}| = \max_{i=1:N} |\boldsymbol{a}_i^T \boldsymbol{r}|$$

MP—procesul iterativ

- Eroarea de aproximare este $\|\boldsymbol{r} - x_k \boldsymbol{a}_k\|$ şi e minimizată de

$$x_k = (\boldsymbol{a}_k^T \boldsymbol{a}_k)^{-1} \boldsymbol{a}_k^T \boldsymbol{r} = \boldsymbol{a}_k^T \boldsymbol{r}$$

- Se introduce k în $\mathcal{I}$ şi se calculează noul reziduu

$$\boldsymbol{r} \leftarrow \boldsymbol{r} - (\boldsymbol{a}_k^T \boldsymbol{r}) \boldsymbol{a}_k$$

- Se trece la iteraţia următoare, căutând din nou coloana cea mai apropiată de reziduu
- Criteriu de oprire: norma reziduului este mai mică decât δ, caz în care a fost găsită o soluţie pentru (2)
- Criteriu alternativ: numărul de coloane selectate atinge o valoare prestabilită (ne interesează mai mult numărul de elemente nenule din $\boldsymbol{x}$ decât eroarea)

MP—algoritmul

- Intrare: matricea $\boldsymbol{A} \in \mathbb{R}^{m \times N}$, vectorul $\boldsymbol{b} \in \mathbb{R}^m$, toleranţa δ

1. $\ell = 1$, $\mathcal{I} = \emptyset$, $\boldsymbol{r}^{(0)} = \boldsymbol{b}$, $\boldsymbol{x} = 0$
2. Caută coloana cea mai aproape de reziduu

$$k_\ell = \arg\max_{i=1:N} |\boldsymbol{a}_i^T \boldsymbol{r}^{(\ell-1)}| \tag{3}$$

3. $\mathcal{I} \leftarrow \mathcal{I} \cup \{k_\ell\}$
4. Coeficientul curent al soluţiei: $x_{k_\ell} = \boldsymbol{a}_{k_\ell}^T \boldsymbol{r}^{(\ell-1)}$
5. Noul reziduu: $\boldsymbol{r}^{(\ell)} \leftarrow \boldsymbol{r}^{(\ell-1)} - (\boldsymbol{a}_{k_\ell}^T \boldsymbol{r}^{(\ell-1)}) \boldsymbol{a}_{k_\ell}$
6. Dacă $\|\boldsymbol{r}^{(\ell)}\|_2 > \delta$, pune $\ell \leftarrow \ell + 1$ şi reia de la 2

- Ieşire: soluţia $\boldsymbol{x}$ (cu ℓ elemente nenule)

Comentarii

- Nu există nici o garanţie a optimalităţii algoritmului
- Reziduul $\boldsymbol{r}^{(\ell)}$ este prin construcţie ortogonal pe vectorul $\boldsymbol{a}_{k_\ell}$
- Totuşi, în general, nu este ortogonal pe vectorii selectaţi anterior (cum ar fi într-o soluţie CMMP)
- Deci nu se garantează optimalitate nici în sensul următor: dată o mulţime $\mathcal{I}$ de indici, algoritmul produce soluţia CMMP a sistemului $\boldsymbol{A}_{\mathcal{I}}\boldsymbol{x}_{\mathcal{I}} = \boldsymbol{b}$, unde $\boldsymbol{A}_{\mathcal{I}}$ este submatricea lui $\boldsymbol{A}$ cu coloanele având indici din $\mathcal{I}$
- Deci nu se minimizează $\|\boldsymbol{A}_{\mathcal{I}}\boldsymbol{x}_{\mathcal{I}} - \boldsymbol{b}\|_2$, chiar dacă mulţimea $\mathcal{I}$ este aleasă optim
- De fapt, algoritmul nici măcar nu garantează că o coloană nu este selectată de mai multe ori
- Număr de operaţii: $O(mN)$ pe iteraţie, în (3)

Variantă de implementare

- Dacă o matrice $\boldsymbol{A}$ este folosită pentru rezolvarea multor sisteme, se pot precalcula produsele scalare între coloane $\boldsymbol{a}_i^T \boldsymbol{a}_j$, $1 \leq i < j \leq N$
- Costul este $O(mN^2)$, dar apare o singură dată
- Produsele scalare între coloane şi reziduul curent se pot calcula la fiecare iteraţie prin

$$\boldsymbol{a}_i^T \boldsymbol{r}^{(\ell)} \leftarrow \boldsymbol{a}_i^T \boldsymbol{r}^{(\ell-1)} - (\boldsymbol{a}_{k_\ell}^T \boldsymbol{r}^{(\ell-1)})(\boldsymbol{a}_i^T \boldsymbol{a}_{k_\ell}), \quad i = 1 : N$$

- Sunt necesare $O(N)$ operaţii la fiecare iteraţie pentru calculul maximului (3), elementului x_{k_ℓ} şi actualizării produselor scalare de mai sus
- La început sunt necesare $O(mN)$ operaţii pentru iniţializarea produselor $\boldsymbol{a}_i^T \boldsymbol{r}^{(\ell)} = \boldsymbol{a}_i^T \boldsymbol{b}$

Soluţia CMMP pentru suport dat

- Dacă suportul (indicii elementelor nenule) $\mathcal{I}$ este disponibil (printr-o metodă neprecizată), atunci soluţia sistemului $\boldsymbol{A}_{\mathcal{I}}\boldsymbol{x}_{\mathcal{I}} = \boldsymbol{b}$ (adică $\boldsymbol{x}_{\mathcal{I}}$ care minimizează $\|\boldsymbol{A}_{\mathcal{I}}\boldsymbol{x}_{\mathcal{I}} - \boldsymbol{b}\|_2$) se calculează folosind triangularizarea ortogonală
- Sistemul este supradeterminat fiindcă $|\mathcal{I}| < m$ ($\boldsymbol{x}$ e rar)
- Algoritmul are deci structura tipică (pres. $\mathcal{I}$ ordonată)
  1. pentru $\ell = 1 : |\mathcal{I}|$ (k_ℓ este indicele ℓ din $\mathcal{I}$)
     1. Calculează reflectorul Householder $\boldsymbol{U}_\ell$ astfel încât $\boldsymbol{U}_\ell \boldsymbol{a}_{k_\ell}$ este zero în poziţiile $\ell + 1 : m$
     2. Actualizează $\boldsymbol{A}_{\mathcal{I}} \leftarrow \boldsymbol{U}_\ell \boldsymbol{A}_{\mathcal{I}}$, $\boldsymbol{b} \leftarrow \boldsymbol{U}_\ell \boldsymbol{b}$
  2. Rezolvă sistemul triunghiular $\boldsymbol{A}_{\mathcal{I}}\boldsymbol{x}_{\mathcal{I}} = \boldsymbol{b}$
- Mulţimea $\mathcal{I}$ nu trebuie să fie disponibilă toată de la început, e suficient să apară un nou indice în fiecare iteraţie
- În acest caz, noua coloană trebuie actualizată cu reflectorii Householder anteriori

Soluţie CMMP cu selecţie separată a coloanelor

- Orice algoritm de selecţie a indicilor din $\mathcal{I}$ poate fi utilizat în acest context
- Structura algoritmului devine

0. Iniţializează $\ell = 1$, $\boldsymbol{A}_{\mathcal{I}} = []$, $\boldsymbol{r} = \boldsymbol{b}$, $\tilde{\boldsymbol{b}} = \boldsymbol{b}$
1. cât timp $\|\boldsymbol{r}\| > \delta$
    1. Selectează un nou indice k_ℓ
    2. $\boldsymbol{a}_{k_\ell} \leftarrow \boldsymbol{U}_{k_{\ell-1}} \ldots \boldsymbol{U}_1 \boldsymbol{a}_{k_\ell}$
    3. $\boldsymbol{A}_{\mathcal{I}} \leftarrow [\boldsymbol{A}_{\mathcal{I}}\ \boldsymbol{a}_{k_\ell}]$
    4. Calculează reflectorul Householder $\boldsymbol{U}_\ell$ astfel încât $\boldsymbol{U}_\ell \boldsymbol{a}_{k_\ell}$ este zero în poziţiile $\ell + 1 : m$
    5. Actualizează $\boldsymbol{A}_{\mathcal{I}} \leftarrow \boldsymbol{U}_\ell \boldsymbol{A}_{\mathcal{I}}$, $\tilde{\boldsymbol{b}} \leftarrow \boldsymbol{U}_\ell \tilde{\boldsymbol{b}}$
    6. Rezolvă sistemul triunghiular $\boldsymbol{A}_{\mathcal{I}} \boldsymbol{x}_{\mathcal{I}} = \tilde{\boldsymbol{b}}$
    7. Calculează reziduul $\boldsymbol{r} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}$
    8. $\ell \leftarrow \ell + 1$

Comentarii

- În 1.7, reziduul e calculat folosind datele iniţiale ale sistemului, adică $\boldsymbol{A}$ şi $\boldsymbol{b}$
- $\boldsymbol{A}_{\mathcal{I}}$ şi $\tilde{\boldsymbol{b}}$ sunt actualizate cu transformările ortogonale
- Soluţia $\boldsymbol{x}$ de la pasul ℓ are elementele nenule egale cu valorile din $\boldsymbol{x}_{\mathcal{I}}$
- Calculele se pot eficientiza în funcţie de procedura de selecţie şi se pot organiza astfel încât să fie $O(mN)$ operaţii pe iteraţie

Orthogonal Matching Pursuit

- OMP foloseşte aceeaşi procedură de selecţie a indicilor $\mathcal{I}$ ca MP
- În general, dă rezultate mult mai bune decât MP
- Calculul soluţiei CMMP permite micşorarea reziduului; la rândul lui, un reziduu mai mic permite selectarea unui suport mai adecvat
- Complexitatea OMP este de acelaşi ordin de mărime cu complexitatea MP
- Nu există garanţii generale de optimalitate
- Caz special: dacă soluţia sistemului $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ satisface

$$\|\boldsymbol{x}\|_0 \leq \frac{1}{2}\left(1 + \frac{1}{\mu(\boldsymbol{A})}\right)$$

atunci ea este găsită de OMP cu $\delta = 0$

Greedy Least Squares

- În algoritmul GLS (numit şi optimized OMP sau orthgonal LS), la fiecare iteraţie se alege coloana care adăugată la $\mathcal{I}$ minimizează norma reziduului sistemului $\boldsymbol{A}_{\mathcal{I}}\boldsymbol{x}_{\mathcal{I}} = \boldsymbol{b}$
- Deci, spre deosebire de OMP, care selectează coloana curentă considerând doar relaţia ei directă cu reziduul, GLS tine cont explicit de alegerile anterioare
- Algoritmul poate fi implementat eficient ca o factorizare QR cu pivotare, costul fiind $O(mN)$ pe iteraţie
- Poate da rezultate mai bune decât OMP, din nou fără garanţii generale de optimalitate

Relaxări convexe

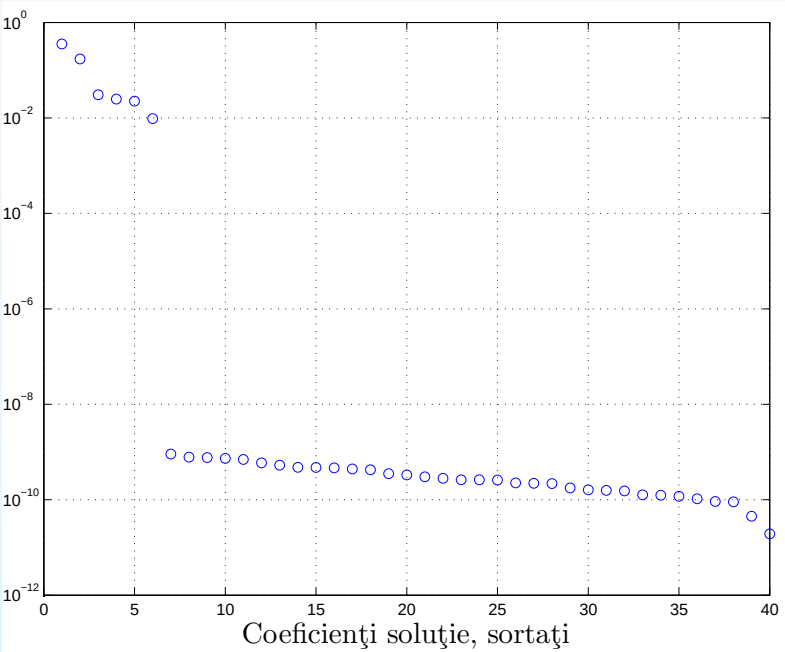
- Cea mai simplă relaxare convexă a problemei (2) se obţine prin utilizarea normei 1 în loc de "norma" 0

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \|\boldsymbol{W}\boldsymbol{x}\|_1 \\ c.r. \quad & \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2 \leq \delta \end{aligned} \tag{4}$$

- Matricea constantă $\boldsymbol{W}$ are rol de ponderare; $\boldsymbol{W}$ este diagonală, cu elementele diagonale pozitive; în lipsa unor informaţii care să permită o alegere adecvată, se ia $\boldsymbol{W} = \boldsymbol{I}$
- Problema (4) este convexă (criteriul se poate exprima cu restricţii liniare, restricţia este de tip con de ordinul doi)
- Tipic, soluţia problemei (4) nu este rară, dar are multe elemente foarte mici (în modul)
- Forţând aceste elemente la zero se obţine o soluţie rară fără a modifica semnificativ norma reziduului

Exemplu

- Figura: soluţia problemei (4) pentru date generate aleator, cu $N = 40$, $m = 10$, $\delta = 1$
- Coeficienţii lui x sunt ordonaţi în ordine descrescătoare a modulelor
- Sunt 6 coeficienţi semnificativi, restul neglijabili

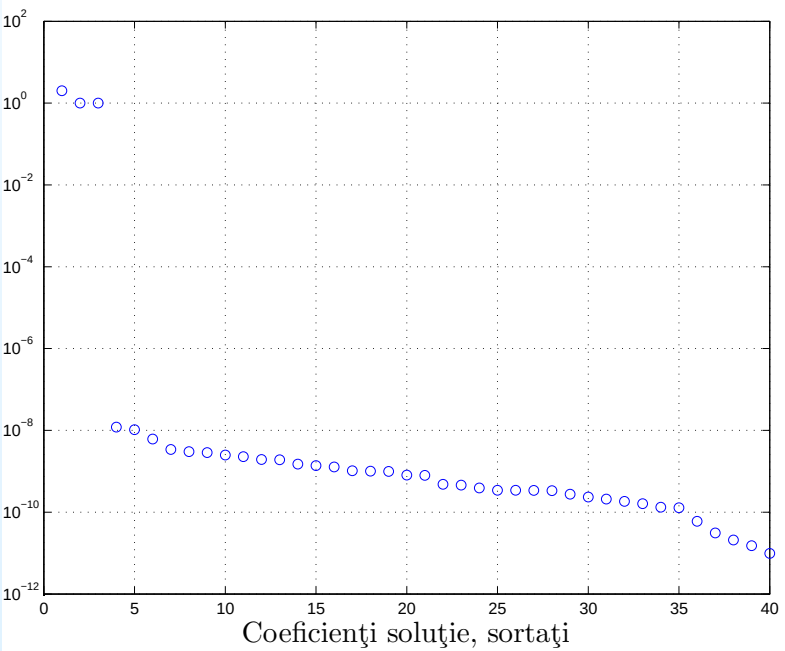


Optimalitate

- În general, nu există garanţii de optimalitate; totuşi, soluţia sistemului $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ este şi soluţia problemei (4) cu $\delta = 0$, dacă

$$\|\boldsymbol{x}\|_0 \leq \frac{1}{2}\left(1 + \frac{1}{\mu(\boldsymbol{A})}\right)$$

- Figura: soluţia (4) pentru date generate astfel încât $\boldsymbol{x}$ să aibă 3 elemente nenule (şi $\delta = 0$)



Formulare alternativă

- O alternativă la (4) este problema

$$\min_{\boldsymbol{x}} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|^2 + \lambda \|\boldsymbol{W}\boldsymbol{x}\|_1 \tag{5}$$

- Ponderea λ corespunde unei alegeri a toleranţei δ (care depinde însă şi de $\boldsymbol{A}$ şi $\boldsymbol{b}$)
- Practic, alegerea unor constante λ sau δ potrivite este o problemă destul de dificilă, rezolvată eminamente empiric
- Avantajul formulării (5) este lipsa restricţiilor
- Problemele (4) şi (5) sunt numite *basis pursuit*
- Atunci când matricea $\boldsymbol{A}$ este mare, metodele de optimizare convexă pot deveni lente şi sunt înlocuite cu metode (mai mult sau mai puţin) euristice

CMMP reponderate iterativ—ideea de bază

- Problemele în care apar diverse norme p pot fi rezolvate iterativ prin probleme CMMP ponderate
- Motivul: problemele CMMP se rezolvă uşor numeric
- Pentru o problemă cu variabila $\boldsymbol{x} \in \mathbb{R}^n$, se face aproximarea

$$\|\tilde{\boldsymbol{x}}\|_p^p = \sum_{i=1}^{n} |\tilde{x}_i|^p \approx \sum_{i=1}^{n} |x_i|^{p-2} |\tilde{x}_i|^2$$

- În procesul iterativ, $\tilde{\boldsymbol{x}}$ este variabila curentă, iar $\boldsymbol{x}$ este valoarea variabilei disponibilă de la pasul precedent
- Variabila $\tilde{\boldsymbol{x}}$ apare pătratic în aproximaţie
- Atunci când $\tilde{\boldsymbol{x}} = \boldsymbol{x}$, formula de mai sus este exactă. Un proces iterativ în care aproximaţia este folosită pentru optimizare, iar la sfarşitul fiecărei iteraţii se pune $\boldsymbol{x} = \tilde{\boldsymbol{x}}$, poate converge spre soluţie (fără garanţii generale)

Cazul normei 1

- În cazul normei 1, formula de aproximare devine

$$\|\tilde{\boldsymbol{x}}\|_1 = \sum_{i=1}^{n} |\tilde{x}_i| \approx \sum_{i=1}^{n} |x_i|^{-1} |\tilde{x}_i|^2 = \tilde{\boldsymbol{x}}^T \boldsymbol{X}^{-1} \tilde{\boldsymbol{x}}$$

- Matricea $\boldsymbol{X}$ este diagonală, cu elementele diagonale egale cu $|x_i|$
- Presupunând că dispunem de o soluţie aproximativă $\boldsymbol{x}$ pentru problema (5), o aproximaţie $\tilde{\boldsymbol{x}}$ (probabil) mai bună se obţine minimizând funcţia pătratică convexă

$$(\boldsymbol{A}\tilde{\boldsymbol{x}} - \boldsymbol{b})^T (\boldsymbol{A}\tilde{\boldsymbol{x}} - \boldsymbol{b}) + \lambda \tilde{\boldsymbol{x}}^T \boldsymbol{W} \boldsymbol{X}^{-1} \tilde{\boldsymbol{x}} \qquad (6)$$

- Anulând gradientul, minimul este soluţia sistemului

$$(\boldsymbol{A}^T \boldsymbol{A} + \lambda \boldsymbol{W} \boldsymbol{X}^{-1}) \tilde{\boldsymbol{x}} = \boldsymbol{A}^T \boldsymbol{b}$$

CMMP reponderate iterativ pentru basis pursuit

- Dacă $x_i \approx 0$ (ceea ce dorim pentru majoritatea elementelor !), atunci $|x_i|^{-1}$ devine foarte mare, ceea ce poate pune probleme numerice
- Soluţie posibilă: înlocuim $|x_i|^{-1}$ cu $(|x_i| + \eta)^{-1}$, unde η este o constantă mică
- Algoritmul CMMP reponderate iterativ are forma
  0. Date de intrare: $\boldsymbol{A}$, $\boldsymbol{b}$, λ, $\boldsymbol{W}$, η, ε (toleranţă de oprire)
  1. Iniţializează $\boldsymbol{x}$
  2. Pune $\boldsymbol{X} = diag(|x| + \eta)$
  3. Rezolvă sistemul $(\boldsymbol{A}^T\boldsymbol{A} + \lambda \boldsymbol{W}\boldsymbol{X}^{-1})\tilde{\boldsymbol{x}} = \boldsymbol{A}^T\boldsymbol{b}$
  4. Dacă $\|\tilde{\boldsymbol{x}} - \boldsymbol{x}\| < \varepsilon$, stop, soluţia este $\tilde{\boldsymbol{x}}$
  5. Altfel, pune $\boldsymbol{x} = \tilde{\boldsymbol{x}}$ şi reia de la 2

Comentarii

- Convergenţa se atinge atunci când diferenţa dintre soluţiile calculate la iteraţii consecutive este mică
- Trebuie prevăzut un număr maxim de iteraţii, pentru cazul în care algoritmul nu converge (de obicei converge !)
- Sistemul de la pasul 3 poate fi rezolvat doar aproximativ, de exemplu folosind câţiva paşi ai unei metode iterative (e.g. gradient conjugat)
- În acest caz nu este necesară formarea matricei $\boldsymbol{A}^T\boldsymbol{A}$, ci doar înmulţirea ei cu diverşi vectori

Alegerea dicţionarului

- Până acum am presupus că dicţionarul (baza supracompletă formată din coloanele matricei $\boldsymbol{A}$) este dat
- Ne punem acum problema alegerii (proiectării) matricei $\boldsymbol{A}$, astfel încât să fie adecvată reprezentării rare a semnalelor dintr-o anume clasă
- Alegerea cea mai simplă este prin concatenarea unor baze ortogonale, $\boldsymbol{A} = [\boldsymbol{\Phi}_1 \ \boldsymbol{\Phi}_2 \ \dots]$, cu $\boldsymbol{\Phi}_i \in \mathbb{R}^{m \times m}$ matrice ortogonale
- De exemplu, aceste baze pot fi cele standard:
  - $\boldsymbol{\Phi}_1 = \boldsymbol{I}$, adică baza temporală standard
  - $\boldsymbol{\Phi}_2$ matricea de transformare DCT sau DFT
  - $\boldsymbol{\Phi}_3$ o transformare wavelet, etc.

Proiectarea dicţionarului

- Alternativ, dicţionarul se poate obţine prin optimizare
- El poate fi structurat, în sensul că (de exemplu) este format din matrice ortogonale care depind de câţiva parametri
- În cazul extrem, vectorii dicţionarului sunt complet liberi
- Proiectarea se face având la dispoziţie un set de vectori de antrenare $\boldsymbol{b}_\ell$, $\ell = 1 : L$, cu L suficient de mare
- Se presupune implicit că există o bază supracompletă $\boldsymbol{A}$ în care soluţiile sistemelor $\boldsymbol{A}\boldsymbol{x}_\ell = \boldsymbol{b}_\ell$ sunt rare (dacă nu toate, măcar o bună parte)

Problema de optimizare

- Inspirându-ne din (4), putem formula proiectarea dicţionarului ca problema de optimizare

$$\begin{array}{ll} \min\limits_{\boldsymbol{A},\boldsymbol{x}_\ell} & \sum_{\ell=1}^{L} \|\boldsymbol{x}_\ell\|_0 \\ c.r. & \|\boldsymbol{A}\boldsymbol{x}_\ell - \boldsymbol{b}_\ell\|_2 \leq \delta,\ \ell = 1:L \end{array} \tag{7}$$

- Optimizarea dicţionarului este evident mai dificilă decât găsirea unei reprezentări rare, deci rezolvarea problemei (7) se poate face doar prin metode euristice
- Algoritmi de rezolvare: MOD (method of directions), K-SVD
- Discutăm doar principial algoritmul MOD

Algoritmul MOD—ideea

- Dacă $\boldsymbol{A}$ ar fi fixat, atunci în (7) variabilele $\boldsymbol{x}_\ell$ ar fi decuplate, deci pentru fiecare ℓ, soluţia rară $\boldsymbol{x}_\ell$ ar putea fi găsită prin metodele discutate anterior (matching pursuit, basis pursuit)
- Notăm $\boldsymbol{X} = [\boldsymbol{x}_\ell]_{\ell=1:L}$, $\boldsymbol{B} = [\boldsymbol{b}_\ell]_{\ell=1:L}$
- Dacă soluţiile $\boldsymbol{x}_\ell$ ar fi cunoscute, atunci $\boldsymbol{A}$ ar putea fi găsit prin rezolvarea problemei CMMP $\boldsymbol{AX} = \boldsymbol{B}$
- (Se minimizează astfel $\|\boldsymbol{AX} - \boldsymbol{B}\|_F$)
- Ideea de optimizare: se porneşte cu un dicţionar iniţial $\boldsymbol{A}_0$
- Presupunând $\boldsymbol{A}_{k-1}$ cunoscut (iniţial $k = 1$), se calculează $\boldsymbol{X}_k$ în funcţie de $\boldsymbol{A}_{k-1}$ folosind algoritmi pentru reprezentări rare
- Se află $\boldsymbol{A}_k$ minimizând $\|\boldsymbol{A}_k\boldsymbol{X}_k - \boldsymbol{B}\|_F$
- Se pune $k \leftarrow k + 1$ şi se repetă operaţiile de mai sus până la convergenţă

Exemple de aplicaţii

- Mai târziu, eventual doar la tablă !