Calcul numeric

Algoritmi iterativi pentru rezolvarea sistemelor

Paul Irofti
Andrei Pătrașcu
Cristian Rusu

Departmentul de Informatică

Facultatea de Matematică și Informatică

Universitatea din București

Email: prenume.nume@fmi.unibuc.ro

Cuprins

- Ce am făcut data trecută: descompunerea valorilor proprii (în special cazul simetric)
- Motivație pentru metode iterative
- Metode de optimizare
- Actualizări pe coordonate
- Metode iterative
- Spații Krylov

Descompunerea valorilor proprii (cazul general)

Pentru $\mathbf{A} \in \mathbb{R}^{n \times n}$.
Dacă avem $\mathbf{A}\mathbf{v}_i = \lambda_i \mathbf{v}_i$ pentru $i = 1, \ldots, n$ atunci avem defapt

$$\mathbf{A} \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \ldots & \mathbf{v}_n \end{bmatrix} = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \ldots & \mathbf{v}_n \end{bmatrix} \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n \end{bmatrix} \tag{1}$$

adică

$$\mathbf{AV} = \mathbf{VD} \text{ sau } \mathbf{A} = \mathbf{VDV}^{-1}. \tag{2}$$

Descompunerea valorilor proprii (cazul simetric)

Pentru $\mathbf{A} \in \mathbb{R}^{n \times n}$ și $\mathbf{A} = \mathbf{A}^T$.
Dacă avem $\mathbf{A}\mathbf{v}_i = \lambda_i \mathbf{v}_i$ pentru $i = 1, \ldots, n$ atunci avem defapt

$$\mathbf{A} \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \ldots & \mathbf{v}_n \end{bmatrix} = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \ldots & \mathbf{v}_n \end{bmatrix} \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n \end{bmatrix} \tag{3}$$

adică

$$\mathbf{A}\mathbf{V} = \mathbf{V}\mathbf{D} \text{ sau } \mathbf{A} = \mathbf{V}\mathbf{D}\mathbf{V}^T. \tag{4}$$

(adică, $\mathbf{V}^{-1} = \mathbf{V}^T$)

Motivația: metode iterative de rezolvare a sistemelor liniare

Sunt situații în care nu vrem sau nu putem să calculăm soluția sistemului $\mathbf{Ax} = \mathbf{b}$.

- dimensiunea n a lui $\mathbf{A}$ este prea mare, deci nu putem calcula foarte precis soluția
- este posibil să nici nu putem memora matricea $\mathbf{A}$ în memorie (procesăm doar pe blocuri)
- nu vrem o soluție foarte bună, este suficient $\mathbf{Ax} \approx \mathbf{b}$
- $\mathbf{A}$ are foarte multe zerouri

Motivația: metode iterative de rezolvare a sistemelor liniare

Sistemul **A** poate să fie sparse (rar) dar factorii săi nu vor fi

$$\mathbf{A} = \begin{bmatrix} 5 & -2 & 0 & -2 & -2 \\ -2 & 5 & -2 & 0 & 0 \\ 0 & -2 & 5 & -2 & 0 \\ -2 & 0 & -2 & 5 & -2 \\ -2 & 0 & 0 & -2 & 5 \end{bmatrix}$$

$$\mathbf{L} = \begin{bmatrix} 2.24 & 0 & 0 & 0 & 0 \\ -0.89 & 2.05 & 0 & 0 & 0 \\ 0 & -0.98 & 2.02 & 0 & 0 \\ -0.89 & -0.39 & -1.18 & 1.63 & 0 \\ -0.89 & -0.39 & -0.19 & -1.95 & 0.45 \end{bmatrix}$$

În acest caz, factorul Cholesky este aproape plin

Motivația: metode iterative de rezolvare a sistemelor liniare

Câteva caracteristici generale pentru metodele iterative

- vom face mai mulți pași în rezolvare, for fi iterații
- am vrea fiecare iterație să fie $O(n)$ sau $O(n^2)$ cel mult
- vom folosi $\mathbf{A}$, probabil vom avea nevoie să face operații Ay
- am vrea unii pași să fie paralelizabili
- vrem să facem maxim $O(n)$ pași
- am vrea să știm dacă funcționează mereu
- condiții teoretice când merg bine

Metoda de gradient

Cum credeți că vom folosi metoda de gradient pentru a rezolva un sistem de ecuații?

Metoda de gradient

- vom folosi tehnica clasică de optimizare cu gradient
- considerăm functia cost $f(\mathbf{x}) = \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2$
- gradientule este $\nabla f(\mathbf{x}) = 2\mathbf{A}^T(\mathbf{A}\mathbf{x} - \mathbf{b})$
- formula de actualizare este $\mathbf{x}^{k+1} = \mathbf{x}^k - \gamma \nabla f(\mathbf{x}^k)$
- cum actualizăm gamma?
  - rezolvăm problema $\underset{\gamma}{\text{minimize}} \ \|\mathbf{A}\mathbf{x}^{k+1} - \mathbf{b}\|_2^2$
  - soluția $\gamma = \frac{(\mathbf{r}^k)^T \mathbf{A}\mathbf{A}^T \mathbf{r}^k}{\|\mathbf{A}\mathbf{A}^T \mathbf{r}^k\|_2^2}$

Metoda de gradient

Algoritmul

- repeat
- $\mathbf{r}^k = \mathbf{A}\mathbf{x}^k - \mathbf{b}$
- $\gamma = \frac{(\mathbf{r}^k)^T \mathbf{A}\mathbf{A}^T \mathbf{r}^k}{\|\mathbf{A}\mathbf{A}^T \mathbf{r}^k\|_2^2}$
- $\mathbf{x}^{k+1} = \mathbf{x}^k - \gamma \mathbf{A}^T \mathbf{r}^k$
- dacă $\|\mathbf{r}^k\|_2$ este mic atunci return

Metoda de gradient

Algoritmul simplificat pentru minimize $\mathbf{x}^T\mathbf{A}\mathbf{x} - 2\mathbf{x}^T\mathbf{b}$ pentru că în acest caz avem $\nabla f(\mathbf{x}^k) = 2(\mathbf{A}\mathbf{x} - \mathbf{b})$.

- repeat
- $\mathbf{r}^k = \mathbf{A}\mathbf{x}^k - \mathbf{b}$
- $\gamma = \frac{(\mathbf{r}^k)^T\mathbf{r}^k}{\|(\mathbf{r}^k)^T\mathbf{A}\mathbf{r}^k\|_2^2}$
- $\mathbf{x}^{k+1} = \mathbf{x}^k - \gamma\mathbf{r}^k$
- dacă $\|\mathbf{r}^k\|_2$ este mic atunci return

Facem înmulțiri $\mathbf{A}\mathbf{y}$ de două ori. Cum eliminăm o asfel de înmulțire?

Metoda de gradient

Algoritmul simplificat pentru minimize $\mathbf{x}^T\mathbf{A}\mathbf{x} - 2\mathbf{x}^T\mathbf{b}$ pentru că în acest caz avem $\nabla f(\mathbf{x}^k) = 2(\mathbf{A}\mathbf{x} - \mathbf{b})$.

- $\mathbf{r}^k = \mathbf{A}\mathbf{x}^k - \mathbf{b}$
- repeat
- $\gamma = \frac{(\mathbf{r}^k)^T\mathbf{r}^k}{(\mathbf{r}^k)^T\mathbf{A}\mathbf{r}^k}$
- $\mathbf{x}^{k+1} = \mathbf{x}^k - \gamma\mathbf{r}^k$
- dacă $\|\mathbf{r}^k\|_2$ este mic atunci return
- $\mathbf{r}^{k+1} = \mathbf{r}^k - \gamma\mathbf{A}\mathbf{r}^k$

Am folosit faptul că $\mathbf{x}^{k+1} = \mathbf{x}^k - \gamma\mathbf{r}$ implică $\mathbf{r}^{k+1} = \mathbf{r}^k - \gamma\mathbf{A}\mathbf{r}^k$.

Optimizare pe coordonate

Ideea: modificăm o singură coordonată din $\mathbf{x}$ la un pas.
Cum arată algoritmul?

Optimizare pe coordonate

Ideea: modificăm o singură coordonată din $\mathbf{x}$ la un pas.

- repeat
- alege o coordonată i
- $x_i = \frac{\mathbf{a}_i^T \mathbf{r}_{-i}}{\|\mathbf{a}_i^T \mathbf{a}_i\|_2^2}$, $\mathbf{r}_{-i} = \mathbf{A}_{-i}\mathbf{x}_{-i} - \mathbf{b}$, unde $\mathbf{A}_{-i}$ este matricea $\mathbf{A}$ dar fară coloana i iar $\mathbf{x}_{-i}$ este vectorul $\mathbf{x}$ dar fară linia i.

Metode iterative

Ideea generală:

- Inspirate de formula de radical babiloniană

$$x_{k+1} = \frac{1}{2}\left(x_k + \frac{S}{x_k}\right) \tag{5}$$

- ideea: să "spargem problema" în componente care sunt ușor de rezolvat[1]
- iterăm până converge

[1] provine din iterația Newton $x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$

Metode iterative

Ideea generală:

- Inspirate de formula de radical babiloniană
- ideea: să "spargem problema" în componente care sunt ușor de rezolvat
- iterăm până converge

Structura algoritmului:

- $\mathbf{x}^{k+1} = F(\mathbf{x}^k)$
- ne va interesa să înțelegem cantitatea $\mathbf{e}^k = \mathbf{x}^k - \mathbf{x}^\star$
- relația cu reziduul este $\mathbf{A}\mathbf{e}^k = \mathbf{A}\mathbf{x}^k - \mathbf{A}\mathbf{x}^\star = \mathbf{A}\mathbf{x}^k - \mathbf{b} = \mathbf{r}^k$

Metode iterative

Ideea generală:

- vom sparge matricea cu care lucrăm: $\mathbf{A} = \mathbf{M} - \mathbf{N}$
- vom rezolva $\mathbf{M}\mathbf{x}^{k+1} = \mathbf{N}\mathbf{x}^{k} + \mathbf{b}$
- soluția este $\mathbf{x}^{k+1} = \mathbf{M}^{-1}\mathbf{N}\mathbf{x}^{k} + \mathbf{M}^{-1}\mathbf{b}$
- deci avem nevoie să alegem $\mathbf{M}$ astfel încât să fie ușor de inversat (i.e., diagonal, inferior sau super triunghiular)

Cum măsurăm progresul?

- definim $\mathbf{e}^{k+1} = \mathbf{x}^{k+1} - \mathbf{x}^{\star}$ și $\mathbf{e}^{k} = \mathbf{x}^{k} - \mathbf{x}^{\star}$
- vrem să scriem $\mathbf{e}^{k+1} = \mathbf{C}\mathbf{e}^{k}$
- avem $\mathbf{M}\mathbf{e}^{k+1} = \mathbf{N}\mathbf{e}^{k}$
- deci avem $\mathbf{e}^{k+1} = \mathbf{M}^{-1}\mathbf{N}\mathbf{e}^{k} = (\mathbf{I} - \mathbf{M}^{-1}\mathbf{A})\mathbf{e}^{k}$

Metode iterative

De ce importantă relația dintre erorile consecutive?

- deci avem $\mathbf{e}^{k+1} = \mathbf{M}^{-1}\mathbf{N}\mathbf{e}^{k} = (\mathbf{I} - \mathbf{M}^{-1}\mathbf{A})\mathbf{e}^{k}$
- deci avem $\mathbf{e}^{k+1} = \mathbf{C}^{k}\mathbf{e}^{0}$
- când scade precis eroarea? când spectrul matricei $\mathbf{C}$ este sub-unitar, adică cea mai mare valoare proprie este sub-unitară
- condiția este: $\rho(\mathbf{C}) \leq 1$, $\rho(\mathbf{C})$ este spectrul matricei $\mathbf{C}$

Cum alegem $\mathbf{M}$ și $\mathbf{N}$?

- considerăm $\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{U}$
- metoda Jacobi: $\mathbf{M} = \mathbf{D}$
- metoda Gauss-Seidel: $\mathbf{M} = \mathbf{D} + \mathbf{L}$

Spații Krylov

Ideea de bază

- orice matrice are un polinom caracteristic minimal $p(\lambda)$
- avem că $p(\lambda) = \lambda^n + c_{n-1}\lambda^{n-1} + \cdots + c_1\lambda + c_0$
- teorema Cayley-Hamilton spune că $p(\mathbf{A}) = \mathbf{0}$
- pentru $\mathbf{A}$ vom avea:

$$-\frac{1}{c_0}\mathbf{A}(\mathbf{A}^{n-1} + \cdots + c_1\mathbf{I}) = \mathbf{I} \tag{6}$$

- de unde rezultă faptul că:

$$\mathbf{A}^{-1} = -\frac{1}{c_0}(\mathbf{A}^{n-1} + + \ldots c_1\mathbf{I}) \tag{7}$$

- de ce este important acest lucru?

Spații Krylov

Ideea de bază

- orice matrice are un polinom caracteristic minimal $p(\lambda)$
- avem că $p(\lambda) = \lambda^n + c_{n-1}\lambda^{n-1} + \cdots + c_1\lambda + c_0$
- teorema Cayley-Hamilton spune că $p(\mathbf{A}) = \mathbf{0}$
- pentru $\mathbf{A}$ vom avea:

$$-\frac{1}{c_0}\mathbf{A}(\mathbf{A}^{n-1} + \cdots + c_1\mathbf{I}) = \mathbf{I} \tag{6}$$

- de unde rezultă faptul că:

$$\mathbf{A}^{-1} = -\frac{1}{c_0}(\mathbf{A}^{n-1} + + \ldots c_1\mathbf{I}) \tag{7}$$

- de ce este important acest lucru? pentru că inversa lui $\mathbf{A}$ este o combinație liniară de puteri a lui $\mathbf{A}$
- deci $\mathbf{A}^{-1}\mathbf{b}$ se află într-o combinație liniară de elemente $\mathbf{A}^k\mathbf{b}$

Spații Krylov

Definim spațiul Krylov de dimensiune m ca fiind

$$\mathcal{K}_m = \{\mathbf{b}, \mathbf{Ab}, \mathbf{A}^2\mathbf{b}, \ldots, \mathbf{A}^{m-1}\mathbf{b}\} \tag{8}$$

Și vom caută aproximarea curentă $\mathbf{x}^m$ în spațiul Krylov de dimensiune m.

Spații Krylov

Un mod special de a construi o aproximare pentru spațiul Krylov este metoda de Gradienți Conjugați (Conjugate Gradient):

- vrem sa minimizăm cantitatea $f(\mathbf{x}) = \mathbf{x}^T\mathbf{A}\mathbf{x} - 2\mathbf{x}^T\mathbf{b}$
- ne reamintim că $\nabla f(\mathbf{x}) = 2(\mathbf{A}\mathbf{x} - \mathbf{b})$
- vom aplica metoda de gradient, dar în loc să mergem pe vectorii de gradient, vom merge pe alți vectori care îndeplinesc condiția $\mathbf{d}_i^T\mathbf{A}\mathbf{d}_j = 0$ pentru $i \neq j$ (aceasta este proprietate de ortogonalitate conjugată)
- vom avea:
  - $\mathbf{d}^0 = -\mathbf{g}^0$
  - $\mathbf{d}^1 = -\mathbf{g}^1 - \beta^1\mathbf{d}^0$ cu $\beta^1 = \frac{(\mathbf{g}^1)^T\mathbf{g}^1}{(\mathbf{g}^0)^T\mathbf{g}^0}$ (este Gram-Schimdt)
  - ...
- o proprietate importantă care va rezulta este: $\mathbf{g}^k - \mathbf{g}^{k-1} = \mathbf{A}(\mathbf{x}^k - \mathbf{x}^{k-1}) = \gamma^{k-1}\mathbf{A}\mathbf{d}^{k-1}$

Spații Krylov

- $\mathbf{r}^0 = \mathbf{b} - \mathbf{A}\mathbf{x}^0$
- $\mathbf{d}^0 = \mathbf{r}^0$
- repeat
- $\gamma^k = \frac{(\mathbf{r}^k)^T \mathbf{r}^k}{(\mathbf{d}^k)^T \mathbf{A}\mathbf{d}^k}$
- $\mathbf{x}^{k+1} = \mathbf{x}^k + \gamma^k \mathbf{d}^k$
- $\mathbf{r}^{k+1} = \mathbf{r}^k - \gamma^k \mathbf{A}\mathbf{d}^k$
- dacă $\|\mathbf{r}^k\|_2^2$ este suficient de mic, return
- $\beta^k = \frac{(\mathbf{r}^{k+1})^T \mathbf{r}^{k+1}}{(\mathbf{r}^k)^T \mathbf{r}^k}$
- $\mathbf{d}^{k+1} = \mathbf{r}^{k+1} + \beta^k \mathbf{d}^k$

Final

- acest domeniu este relativ proaspăt, se face în continuare multă cercetare
- foarte popular pentru ML/AI
- notițele de curs sunt bazate pe o sursă majoră:
  - Y. Saad, Iterative Methods for Sparse Linear Systems Second Edition, 2003