Közelítő és szimbolikus számítások

7. gyakorlat

Hatványmódszer

Készítette:

Gelle Kitti

Csendes Tibor

Somogyi Viktor

Deák Gábor

Faragó István

Horváth Róbert

jegyzetei alapján

1. Vektornormák

Definíció 1.1. *A vektornorma olyan vektortéren vagy függvénytéren értelmezett d valós értéket felvevő leképezés, amelyre érvényesek a következő tulajdonságok:*

- $d(\boldsymbol{x}) > 0$, *ha az* $\boldsymbol{x}$ *egy nem zérusokból álló vektor*
- $d(\mathbf{0}) = 0$
- $d(\boldsymbol{x} + \boldsymbol{y}) \leq d(\boldsymbol{x}) + d(\boldsymbol{y})$
- $d(\lambda \boldsymbol{x}) = |\lambda| d(\boldsymbol{x})$

Gyakran használt norma az úgynevezett *p-norma*, amely a következő általános alakot veszi fel n-dimenziós vektorok esetén:

$$\|\boldsymbol{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}$$

A $p = 2$ választással egy olyan normát kapunk, amelyre teljesül a Pitagorasz tétel. Ha vektornormára gondolunk, akkor általában ez utóbbit szoktuk használni. Az alakja tehát:

$$\|\boldsymbol{x}\|_2 = \sqrt{\sum_{i=1}^{n} |x_i|^2}$$

2. Hatványmódszer

A múlt órán elhangzott, hogy a sajátértékeket nehéz kiszámítani. Ez azért van, mert a sajátértékeket a mátrix karakterisztikus polinomja segítségével adjuk meg, és ha az ebből képzett egyenlet negyedfokúnál nagyobb, akkor nem létezik rá explicit megoldóképlet. További problémánk, hogy gyakran nincs is szükség az összes sajátértékre, hanem csak egyre (általában a legnagyobb vagy legkisebb abszolútértékűre).

Ezt a problémát úgy hidaljuk át, hogy (ahogy az egyenletrendszerek megoldásának közelítésére is) iterációs módszereket alkalmazunk, amelyek egy kezdeti, akár véletlenszerűen választott megoldásból lépések sorozatával egyre pontosabb eredményeket adnak, de sosem érik el a pontos eredményt, csak ahhoz tetszőlegesen közel kerülnek.

Egy ilyen módszer a hatványmódszer. Ez iteratív lépések sorozatával a legnagyobb abszolútértékű sajátértéket és egy hozzá tartozó sajátvektort állítja elő. Az alap algoritmus a következő iteráción alapul:

$$\boldsymbol{y}^{(k)} = A\boldsymbol{x}^{(k)}$$

$$\boldsymbol{x}^{(k+1)} = \frac{\boldsymbol{y}^{(k)}}{\|\boldsymbol{y}^{(k)}\|}.$$

Először tehát kiszámolunk egy $\boldsymbol{y}$ vektort, majd ennek a normált értékét adjuk a következő iterációban szereplő $\boldsymbol{x}^{(k+1)}$-nek. A kiindulási $\boldsymbol{x}^{(0)}$ vektornak továbbá nem szabad merőlegesnek lennie a legnagyobb abszolútértékű sajátértékhez tartozó sajátvektorra (így pl. nem lehet nullvektor sem).

Ha $|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n|$, akkor a legnagyobb abszolútértékű sajátérték megkapható a következő módon:

$$|\lambda_1| = \lim_{k\to\infty} \frac{\|\boldsymbol{y}^{(k)}\|}{\|\boldsymbol{x}^{(k)}\|},$$

a hozzá tartozó sajátvektort ($\boldsymbol{u}_1$) pedig a

$$\boldsymbol{u}_1 = \lim_{k\to\infty} \operatorname{sign}(\lambda_1)^{(k)} \boldsymbol{x}^{(k)}$$

adja.

Ennek az algoritmusnak létezik egy másik változata is, ez a legkisebb abszolútértékű sajátértéket keresi meg. Tegyük fel, hogy T reguláris, ekkor minden sajátértéke nullától különböző. A $T\boldsymbol{x} = \lambda\boldsymbol{x}$ egyenletből T^{-1}-zel beszorozva:

$$\boldsymbol{x} = T^{-1}(\lambda\boldsymbol{x}) = \lambda T^{-1}\boldsymbol{x}$$

adódik, és innen (λ^{-1}-zel való beszorzással):

$$\lambda^{-1}\boldsymbol{x} = \lambda^{-1}\lambda T^{-1}\boldsymbol{x} = T^{-1}\boldsymbol{x}.$$

Azaz, ha T sajátértéke λ, és az ehhez tartozó sajátvektor $\boldsymbol{x}$, akkor a T^{-1} mátrix egy sajátértéke λ^{-1} az $\boldsymbol{x}$ sajátvektorral. Ebből adódik az *inverz hatványmódszer*:

$$\boldsymbol{x}^{(k)} = \frac{\boldsymbol{y}^{(k)}}{\|\boldsymbol{y}^{(k)}\|},$$

$$\boldsymbol{y}^{(k+1)} = T^{-1}\boldsymbol{x}^{(k)}$$

Ezzel a módszerrel egy tetszőleges sajátértéket és a hozzá tartozó sajátvektort közelítjük. Ehhez válasszuk a $T = A - \sigma I$ mátrixot, ahol σ egy tetszőleges érték, az eltolás, $\boldsymbol{y}^{(0)}$ pedig egy szintén tetszőleges kezdővektor.

Az eljárást nyilvánvalóan azért hívjuk inverz iterációnak, mert az A mátrix helyett az $T = A - \sigma I$ mátrix inverzével hajtjuk végre a hatványmódszert. Ha $\sigma = 0$, akkor a nullához legközelebbi, azaz a legkisebb abszolút értékű sajátértéket találja meg a módszer.

Legyen továbbá minden iterációban $\theta^{(k)} = \boldsymbol{x}^{(k)^T}\boldsymbol{y}^{(k)}$. Itt a sajátértéket a

$$|\lambda_n| = \sigma + \frac{1}{\theta^{(k)}}$$

adja, a hozzá tartozó $\boldsymbol{u}_n$ sajátvektort pedig

$$\boldsymbol{u}_n = \frac{\boldsymbol{y}^{(k)}}{\theta^{(k)}}.$$

Vegyük észre, hogy az $\boldsymbol{y}^{(k+1)} = T^{-1}\boldsymbol{x}^{(k)}$ épp a $T\boldsymbol{y}^{(k+1)} = \boldsymbol{x}^{(k)}$ lineáris egyenletrendszer definíció szerinti megoldása $\boldsymbol{y}$-ra, így a műveletigény is jóval nagyobb lesz. A műveletszám csökkentésére jól alkalmazható az LU- vagy QR-felbontás. Mivel minden iterációs lépés egyenletrendszerében az $(A-\sigma I)$ mátrix az együtthatómátrix, így ennek pl LU-felbontását az első lépésben $2n^3/3 + O(n^2)$ művelettel kiszámítva a többi lépésben már csak $2n^2$ műveletre van szükség. Tehát a következőképpen is felírhatjuk az iterációt:

$$\boldsymbol{x}^{(k)} = \frac{\boldsymbol{y}^{(k)}}{\|\boldsymbol{y}^{(k)}\|},$$

$$T\boldsymbol{y}^{(k+1)} = \boldsymbol{x}^{(k)}$$

Megjegyezhetjük, hogy a normalizálás mindegy, hogy az iteráció elején, vagy végén (az új értékkel) történik. Ez azért fontos, hogy elkerüljük az alul- illetve túlcsordulást.

Nézzük továbbá mire is szolgál a $T = A-\sigma I$ választásban a σ paraméter: Ha $A \in \mathbb{R}^{n\times n}$ invertálható, szimmetrikus mátrix és $\sigma \neq \lambda_i$ $(i = 1, \ldots, n)$, akkor a $T = A - \sigma I$ mátrix invertálható, és $(A - \sigma I)^{-1}$ sajátvektorai megegyeznek A sajátvektoraival, sajátértékei pedig $(\lambda_i-\sigma)^{-1}$. Ha σ elegendően közel van valamelyik λ_j sajátértékhez, akkor a domináns sajátérték $(\lambda_j - \sigma)^{-1}$ lesz, és az $(A - \sigma I)^{-1}$ mátrixszal végrehajtva a hatványmódszert, λ_j és u_j meghatározható.

A fenti képleteket pedig a következők alapján kapjuk: ha az $(A-\lambda I)$-t kifejtjük, akkor a λ-k lesznek a karakterisztikus polinom gyökei. A fenti T választással az $(A - \sigma I - \lambda I)$ sajátértékei épp a $(\sigma+\lambda)$-k lesznek. Ekkor ha $\boldsymbol{x}$ egységvektor, akkor θ épp a bezárt szögük koszinusza szorozva $\boldsymbol{y}$ hosszával, ami határértékben (mert 0 vagy 180 fokot fognak zárni) $\pm\boldsymbol{y}$ hosszához fog közeledni, ezért lesz $\sigma+1/\theta$ a sajátérték (mivel T^{-1} - amin végrehajtjuk a hatványmódszert - sajátértékei T sajátértékeinek inverzei, σ pedig a választott eltolás). A sajátvektor pedig, ha θ az $\boldsymbol{y}$ hosszához konvergál, akkor $\boldsymbol{y}/\theta$ a normált sajátvektorhoz magához.

Másrészt ennek a konvergenciasebessége annál gyorsabb, minél nagyobb a hányadosa a legkisebb és a második legkisebb sajátértéknek, amit egy jó σ választással nagyobbá tudunk tenni.

Ha például az A mátrix legkisebb sajátértéke 1, a második legkisebb meg 1.1, akkor nagyon lassan konvergál a módszer. De ha $\sigma = 0.99$-cel eltoljuk (mert van egy ilyen közelítésünk, mondjuk), akkor az $A - \sigma I$ mátrixnak már 0.01 lesz a legkisebb és 0.11 a második legkisebb sajátértéke, ami meg egy nagy hányados és ezért gyorsabb lesz a konvergencia. Annál jobb lesz a sebesség, minél közelebb van a σ a legkisebb sajátértékhez.

Példa. Határozzuk meg a következő mátrix legnagyobb sajátértékét a hatványmódszerrel:

$$\begin{pmatrix} 1 & 0 & 2 \\ -2 & 4 & 2 \\ 1 & 0 & 0 \end{pmatrix}$$

Megoldás. Kezdetben válasszunk egy tetszőleges vektort az $\boldsymbol{x}^{(0)}$ értékének:

$$\boldsymbol{x}^{(0)} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

Ezután kezdjük el az iterációt. Az első lépésben tehát:

$$\boldsymbol{y}^{(0)} = A\boldsymbol{x}^{(0)} = \begin{pmatrix} 1 & 0 & 2 \\ -2 & 4 & 2 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 3 \\ 4 \\ 1 \end{pmatrix}$$

Ezután normáljuk le $\boldsymbol{y}^{(0)}$-t, hogy megkapjuk $\boldsymbol{x}^{(1)}$-et:

$$\|\boldsymbol{y}^0\| = \sqrt{3^2 + 4^2 + 1^2} = 5,099$$

$$\boldsymbol{x}^{(1)} = \frac{\boldsymbol{y}^{(0)}}{\|\boldsymbol{y}^{(0)}\|} = \begin{pmatrix} 3/5,099 \\ 4/5,099 \\ 1/5,099 \end{pmatrix} = \begin{pmatrix} 0,5883 \\ 0,7845 \\ 0,1961 \end{pmatrix}$$

A kiszámolt sajátérték az első iterációban:

$$\lambda^{(1)} = \frac{\|\boldsymbol{y}^{(0)}\|}{\|\boldsymbol{x}^{(0)}\|} = \frac{5,099}{1,7321} = 2,9439$$

Ezután jön a második iteráció. Számoljuk ki $\boldsymbol{y}^{(1)}$-et (helyettesítsünk be az általános képletbe):

$$\boldsymbol{y}^{(1)} = A\boldsymbol{x}^{(1)} = \begin{pmatrix} 1 & 0 & 2 \\ -2 & 4 & 2 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0,5883 \\ 0,7845 \\ 0,1961 \end{pmatrix} = \begin{pmatrix} 0,9806 \\ 2,3534 \\ 0,5883 \end{pmatrix}$$

Ezután normáljuk le $\boldsymbol{y}^{(1)}$-t, hogy megkapjuk $\boldsymbol{x}^{(2)}$-t:

$$\|\boldsymbol{y}^{(1)}\| = 2,6165$$

$$x^{(2)} = \frac{\boldsymbol{y}^{(1)}}{\|\boldsymbol{y}^{(1)}\|} = \begin{pmatrix} 0.3748 \\ 0.8994 \\ 0.2249 \end{pmatrix}$$

A kiszámolt sajátérték a második iterációban:

$$\lambda^{(2)} = \frac{\|\boldsymbol{y}^{(1)}\|}{\|\boldsymbol{x}^{(1)}\|} = \frac{2,6165}{1} = 2,6165.$$

Az iterációt lehetne tovább folytatni, minden lépésben egy egyre közelebbi megoldáshoz jutnánk el.

3. Feladatok

1. Írjunk egy programot, amely megvalósítja a hatványmódszert! Minden lépésben írjuk ki az aktuális sajátértéket.

2. Tekintsük a legkisebb sajátértéket megkereső inverz hatványmódszer egy lehetséges megoldását:

```
%inverz iteracio az A matrix legkisebb sajatertekenek megkeresesere
function [dominantEigenValue, v, err, iterations] = ...
    invPowerMethod(A, x, tol, maxiter)

err = inf;
iterations = 0;

%az iteracio addig megy, amig el nem ertuk a max iteracioszamot
%vagy a hiba elerte a megadott toleranciaszinet
while err ≥ tol & iterations < maxiter,
    x = y/norm(y,2); %normalizaljuk a kapott erteket
    y = A\x; %oldjuk meg az Ay=x egyenletrendszert y-ra
    alpha1 = x'*y; %az aktualis kozelites
    if iterations>0
        err = abs( alpha1-alpha0 )/abs( alpha0 );
    end
    alpha0 = alpha1; %jegyezzuk fel a regi erteket
    iterations = iterations+1; %uj iteracio
end

dominantEigenValue = 1/alpha1; %a keresett sajatertek
v = x; %a hozza tartozo sajatvektor

end
```

Kérdés: hogyan módosítsuk az algoritmust, hogy *bármelyik* sajátérték közelítését megkaphassuk?