Oefeningen Numerieke Wiskunde

Oefenzitting 10 (PC): Het berekenen van eigenwaarden.

In de oefenzitting ga je m.b.v. Matlab verschillende methodes voor het berekenen van de eigenwaarden van een matrix uitproberen op een voorbeeld. Het relevante deel van de cursus is Hoofdstuk 6 van Deel 2: 'Het berekenen van eigenwaarden'. Je past ook de methode van Jacobi uit Hoofdstuk 4 van Deel 2 toe op een stelsel van lineaire vergelijkingen en je onderzoekt het verband tussen de convergentie van deze methode en de methode van de machten voor eigenwaarden.

Voor het uitwerken van de opgaven gebruik je `.m`-bestanden die je kan vinden op Toledo.

Theorie

Berekenen van eigenwaarden Een willekeurige startvector $X^{(0)}$ is te schrijven als een lineaire combinatie van de eigenvectoren $\{V_k\}_{k=1}^n$ van een matrix A die van eenvoudige structuur is. Er geldt dan dat

$$A^k X^{(0)} = \lambda_1^k \left(\alpha_1 V_1 + \alpha_2 \left(\frac{\lambda_2}{\lambda_1} \right)^k V_2 + \ldots + \alpha_n \left(\frac{\lambda_n}{\lambda_1} \right)^k V_n \right).$$

We nemen aan dat

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_n|.$$

Bij de methode van de machten zal hierdoor de benadering van de eigenvector convergeren naar V_1 en men kan aantonen dat de convergentiefactor gelijk is aan $\rho = \lambda_2/\lambda_1$. Dit is ook de convergentiefactor voor de opeenvolgende benaderingen van de eigenwaarde λ_1.

Bij de methode van de inverse machten met verschuiving σ past men de methode van de machten toe op de matrix $(\sigma I - A)^{-1}$. Deze matrix heeft dezelfde eigenvectoren als de matrix A en heeft eigenwaarden $1/(\sigma - \lambda_i),\ i = 1, \ldots, n$. Hoe beter σ een goede benadering is van een bepaalde eigenwaarde, des te sneller de convergentie.

Indien men bij de methode van de inverse machten de verschuiving adaptief maakt, en voor σ steeds de nieuwe benadering van de eigenwaarde neemt, dan wordt de convergentie zelfs kwadratisch.

Iteratief oplossen van stelsels lineaire vergelijkingen Voor de methoden van Jacobi en Gauss-Seidel voor het oplossen van een stelsel $AX = B$ geldt er dat

$$E^{(k)} = GE^{(k-1)}, \qquad \text{met} \quad \begin{cases} G = -D^{-1}(U + L) & \text{voor Jacobi} \\ G = -(L + D)^{-1}U & \text{voor Gauss-Seidel} \end{cases}$$

met $E^{(k)} = X^{(k)} - X^*$ de k-de iteratiefout. Indien we $E^{(0)}$ schrijven als een linaire combinatie van de eigenvectoren $\{V_k\}_{k=1}^n$ van G, dan krijgen we

$$E^{(k)} = G^k E^{(0)} = \alpha_1 \lambda_1^k V_1 + \alpha_2 \lambda_2^k V_2 + \ldots + \alpha_n \lambda_n^k V_n.$$

Hieruit ziet men dat als $\alpha_1 \neq 0$ de fout afneemt zoals $\mathcal{O}(\lambda_1^k)$, en dat de convergentiefactor in modulus gelijk is aan de spectraalradius $\rho(G) = \max_i (|\lambda_i(G)|)$.

Beschrijving van de Matlab-bestanden

- `[x, mu] = machten(A, x0, N)` past de methode van de machten met scalering toe op de matrix `A` met `x0` als startvector en `N` het aantal iteraties. Als output krijg je de benaderde eigenvector `x` en de opeenvolgende benaderingen `mu` van de overeeenkomstige eigenwaarde.

- `[x,mu] = invmachten(A, x0, sigma, N)` past de methode van de inverse machten met verschuiving `sigma` toe op de matrix `A` met `x0` als startvector en `N` het aantal iteraties. Dit komt overeen met het toepassen van de methode van de machten op $(\sigma I - A)^{-1}$. Hoe beter de schatting σ voor een bepaalde eigenwaarde, hoe sneller de methode zal convergeren. Als output krijg je de benaderde eigenvector `x` en de opeenvolgende benaderingen `mu` van de overeeenkomstige eigenwaarde.

- `[x,mu] = invmachten_adaptief(A, x0, sigma, N)` past de methode van de inverse machten met adaptieve verschuiving toe op de matrix `A` met `x0` als startvector en `N` het aantal iteraties. Als output krijg je de benaderde eigenvector `x` en de opeenvolgende benaderingen `mu` van de overeeenkomstige eigenwaarde.

- `fout = jacobi(A, b, x0, xe, n)` past de methode van Jacobi toe voor het iteratief oplossen van een stelsel $Ax = b$ met startvector `x0`, exacte oplossing `xe` en aantal iteraties `n` . Als output krijg je de normsgewijze relatieve fout van de verschillende iteraties, berekend t.o.v. de exacte oplossing `xe`.

Oefeningen

Open een nieuw script en sla het op als `oefz10.m`. Schrijf hierin je code voor elke oefening, zodat je gemakkelijk vorige resultaten opnieuw kan berekenen.

Probleem 1. Definieer de matrix

$$A = \begin{bmatrix} 3 & 1 & -1 \\ 0 & 4 & -6 \\ 0 & 0 & -2 \end{bmatrix}$$

met

```
>> A = [3 1 -1; 0 4 -6; 0 0 -2]
```

Bereken met de hand de eigenwaarden en de bijhorende eigenvectoren van deze matrix. Verifieer je resultaten m.b.v. het Matlab commando

```
>> [V,D] = eig(A)
```

Probleem 2. Pas de verschillende methodes toe op de matrix A. Doe $N = 40$ iteraties, gebruik als startvector $X^{(0)} = [1\ 0\ 1]^T$ en gebruik voor de verschuiving $\sigma = 3.8$:

```
>> N = 40
>> x0 = [1;0;1]
>> sigma = 3.8
```

Je kan de methodes dan als volgt toepassen:

```
[x1,mu1] = machten(A,x0,N)
[x2,mu2] = invmachten(A,x0,sigma,N)
[x3,mu3] = invmachten_adaptief(A,x0,sigma,N)
```

- Bekijk de uitvoer `x1` en `mu1`. Naar welke eigenwaarde en eigenvectoren is er convergentie? Is dit ook zo voor de andere methoden?[1] Sla de waarde van deze eigenwaarde op in `mu`.

- Plot nu de relatieve fouten als volgt

  ```
  figure(1),clf
  plot(abs(mu1-mu)/abs(mu),'--*')
  hold on
  plot(abs(mu2-mu)/abs(mu),'--*r')
  plot(abs(mu3-mu)/abs(mu),'--*g')
  ```

  Kan je hier veel op zien? Vervang nu `plot` door `semilogy`. Welke schaal krijg je op de assen?

- Leid uit de figuur de orde van convergentie af voor de verschillende methodes. Bepaal ook de convergentiefactoren op basis van de theorie.

- Doe `format long` en bekijk de fouten in je Command Window. Kan je hier ook de orde van convergentie en de convergentiefactoren op aflezen?

- Ga de waarden van de convergentiefactoren voor de lineaire convergerende methodes na door de fout van twee iteraties k en $k+m$ te vergelijken in het gebied van convergentie, met m groot genoeg, bvb.

  ```
  rho1 = (abs(mu1(40)-mu)/abs(mu1(20)-mu))^(1/20)
  ```

  Komen deze waarden overeen met wat je haalde uit de theorie?

- Geen van de methodes bevat een stopcriterium. Wat loopt er precies mis bij `invmachten_adaptief`? (Tip: kijk naar de eerste warning die je krijgt en kijk naar de relevante lijn in de code. Wat is het probleem?)

[1] Negeer even de **NaN** waarden in **mu3**, we komen hierop terug.

Probleem 3. (Gevoeligheid voor initiële condities) Doe dertig Jacobi-iteraties om het volgende stelsel op te lossen:

$$\left\{\begin{array}{rcrcrcr} 3x_1 & - & 2x_2 & + & 2x_3 & = & 3 \\ -2x_1 & + & 3x_2 & - & 2x_3 & = & -1 \\ 2x_1 & - & 2x_2 & + & 3x_3 & = & 3 \end{array}\right.$$

Neem als startwaarden $\left[\begin{array}{ccc} 3 & 2 & 0 \end{array}\right]^T$ en $\left[\begin{array}{ccc} 3.0001 & 2 & 0 \end{array}\right]^T$ en plot de fouten. De Matlab commando's hiervoor zijn:

```
A = [ 3 -2 2; -2 3 -2; 2 -2 3];
b = [3; -1; 3]
xe = A\b
x0a = [3; 2; 0]
x0b = [3.0001; 2; 0]
n = 30
err1 = jacobi(A,b,x0a,xe,n)
err2 = jacobi(A,b,x0b,xe,n)
figure(2),clf
semilogy(err1)
hold on
semilogy(err2,'r')
```

Verklaar wat er gebeurt:

- Bereken de iteratiematrix $G = -D^{-1}(L+U)$ en bereken de eigenwaarden en eigenvectoren van G:

  ```
  G = -diag(diag(A))\(A-diag(diag(A)))
  [V,D] = eig(G)
  ```

- Wat is de spectraalradius $\rho(G)$? Wat besluit je over de convergentie?

- Bepaal de initiële foutvector $E^{(0)}$

  ```
  e0a = x0a-xe
  ```

  Verklaar nu in detail het convergentiegedrag dat je bekomt in de figuur. Zoek hiervoor de coëfficiënten van $E^{(0)}$ als lineaire combinatie van de eigenvectoren

  ```
  V\e0a
  ```

  Doe hetzelfde voor de tweede startvector. Wat is het verschil?

Probleem 4. Bewijs dat de matrix $B = (\sigma I - A)^{-1}$ dezelfde eigenvectoren heeft als de matrix A en dat als λ een eigenwaarde is van A, dan is $1/(\sigma - \lambda)$ een eigenwaarde van B.