Oefeningen Numerieke Wiskunde

Oefenzitting 7 (PC): Het oplossen van stelsels lineaire vergelijkingen

In deze oefenzittingen ga je m.b.v. Matlab aspecten zoals conditie, stabiliteit en complexiteit onderzoeken voor stelsels vergelijkingen. Het relevante deel van de cursus is *Hoofdstuk 3 'Stelsels lineaire vergelijkingen'*, de slides over QR, SVD en kleinste kwadraten benaderingen en de slides over het creëren van nullen in een matrix. Voor het uitwerken van de opgaven gebruik je de `.m`-bestanden die je kan vinden op Toledo.

1 MATLAB–functies voor deze oefenzitting

Genereren van matrices

De volgende functies genereren een stelsel $Ax = b$, waarvan de oplossingsvector x bestaat uit natuurlijke getallen. Ieder commando genereert een matrix $M = [A\ b]$ met A een $(n \times n)$–matrix en b een $(n \times 1)$–vector.

- `M = genmatrix1(n)` : De matrix A is een goed geconditioneerde random matrix.
- `M = genmatrix2(n)` : De matrix A is een goed geconditioneerde random matrix met een specifiek permutatiegedrag.
- `M = genmatrixc(n)` : De matrix A is een slecht geconditioneerde random matrix.

Stelsels oplossen

- `G = gauss1(M)` waarbij $M = [A\ b]$ met A een $n \times n$–matrix en b een $n \times 1$–vector. Deze functie maakt de matrix

$$\left(\begin{array}{c|c|c} & & 1 \\ A & b & \vdots \\ & & n \end{array}\right)$$

  aan en past Gauss–eliminatie toe (zie handboek, Algoritme 3.4).

- `G = gauss2(M)` is analoog aan `gauss1`, maar met optimale rij–pivotering (zie handboek, Algoritme 3.5). Merk op dat dit equivalent is met `A\b` in Matlab voor vierkante matrices A.

- `[Q,R] = qr(M)` berekent een QR-factorisatie van de matrix M.
- `x = asubst(G)` voert achterwaartse substitutie uit op het resultaat van `gauss1` of `gauss2` (`x = asubst(R)` resp. op het resultaat van `qr`), en geeft de oplossing x van het stelsel $Ax = b$.

2 Conditie en achterwaartse stabiliteit

Een algoritme $\hat{F}(x)$ om $F(x)$ te berekenen is achterwaarts stabiel als

$$\hat{F}(x) = F(x + \Delta x), \quad \text{én} \quad \frac{||\Delta x||}{||x||} \leq C \cdot \epsilon_{mach}.$$

Dit betekent dat het effect van de voortplanting van afrondingsfouten gemaakt in $\hat{F}(x)$ kan worden teruggebracht tot een kleine relatieve perturbatie op de gegevens. In het geval van het oplossen van stelsels is een algoritme $\hat{x}(A, b)$ dus achterwaarts stabiel als

$$(A + \Delta A)\hat{x} = b + \Delta b, \quad \text{met} \quad \frac{||\Delta A||}{||A||} \leq C_1 \cdot \epsilon_{mach}, \frac{||\Delta b||}{||b||} \leq C_b \cdot \epsilon_{mach}. \tag{1}$$

We zullen in deze oefenzitting stelsels oplossen met verschillende algoritmes en niet enkel de relatieve fouten $\|\hat{x} - x\| / \|x\|$ op de berekende $\hat{x}$ vergelijken, maar ook de residu's

$$r = A\hat{x} - b.$$

Als het residu r klein is t.o.v. het rechterlid b, dan geldt (1) met $\Delta A = 0$ en $\Delta b = r$ en dan is het algoritme dus achterwaarts stabiel voor de gebruikte gegevens. Ga dit na!

In het algemeen geldt er voor een achterwaarts stabiel algoritme dat

$$\frac{||\hat{F}(x) - F(x)||}{||F(x)||} = \frac{||F(x + \Delta x) - F(x)||}{||F(x)||} \leq \kappa_F \frac{||\Delta x||}{||x||} \overset{(*)}{\leq} C \cdot \kappa_F \cdot \epsilon_{mach},$$

dus de voorwaartse fout is begrensd door het conditiegetal en de machine-nauwkeurigheid. Merk op dat de achterwaartse stabiliteit eigenlijk enkel de laatste ongelijkheid $(*)$ impliceert. De gelijkheid definieert gewoon de achterwaartse fout Δx en de eerste ongelijkheid volgt uit de definitie van het conditiegetal. Ga na dat, in het geval van het oplossen van stelsels, er uit de definitie van de achterwaartse fout en het conditiegetal van een matrix volgt dat

$$\frac{||\hat{x} - x||}{||x||} \leq \kappa_A \left(\frac{||\Delta b||}{||b||} + \frac{||\Delta A||}{||A||} \right), \tag{2}$$

en dat bijgevolg voor een achterwaarts stabiel algoritme geldt dat

$$\frac{||\hat{x} - x||}{||x||} \leq C \cdot \kappa_A \cdot \epsilon_{mach}.$$

3 Oefeningen

Probleem 1. (De LU ontbinding)

(a) Stel

$$A = \begin{pmatrix} 7 & 8 & 0 \\ 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}.$$

Bereken de LU-ontbinding van A met `[L,U] = lu(A)`. Wat is $L*U$? Geef de Matlab-bevelen om uit L en U de determinant van A te berekenen.

(b) Stel

$$A = \begin{pmatrix} 1 & 4 & 3 \\ 4 & 3 & 5 \\ 9 & 8 & 0 \end{pmatrix}.$$

Bereken opnieuw de LU-ontbinding van A. Wat merk je op als je de matrices L en U bekijkt? Geef een verklaring voor wat er gebeurd is. (Hint: `doc lu`.)

Probleem 2. (Gauss-eliminatie) Gebruik de functie `genmatrix1` om een 6×7–matrix $M = [A\ b]$ te genereren. Los het stelsel $Ax = b$ op m.b.v. de functies `gauss1`, `gauss2` en `qr`. Je bekomt dus drie oplossingen voor hetzelfde stelsel.

(a) Bereken de relatieve fout van de oplossingen en de verhouding van de residu's t.o.v het rechterlid b, indien je weet dat de exacte oplossing een vector met natuurlijke getallen is. Vul in de onderstaande tabel de grootte-orde (bvb. 10^3, 10^6, 10^{11}, ...) van de fouten en de relatieve residu's in. (Hint: gebruik `norm`.)

(b) Wat kan je zeggen over de stabiliteit van de methodes?

Doe nu hetzelfde, maar met de functie `genmatrix2`.

(c) Wat is het verschil t.o.v. je resultaten met het eerste stelsel? Verklaar.

(d) Wat kan je nu zeggen over de stabiliteit van de methodes?

(Tip: schrijf een Matlab script zodat je gemakkelijk je resultaten kan herberekenen. Deze zullen telkens anders zijn, want we werken met random matrices. De grootte-ordes zullen echter redelijk gelijk blijven.)

Probleem 3. (Conditie en achterwaartse stabiliteit) Voer opnieuw (a) uit van de vorige oefening, maar nu voor de functie `genmatrixc`. Vul de tabel verder aan.

(a) Bereken met `cond` of met `norm` het 2-norm conditiegetal van de matrix A voor de drie verschillende stelsels. Vul telkens de grootte-orde in de tabel in.

(b) Ga na dat de relatieve fouten van de oplossingen en de relatieve residu's voldoen aan formule (2).

(c) Wat is je algemene besluit over de stabiliteit van de methodes?

	$\kappa_2(A)$		gauss1 gauss2 qr
genmatrix1		$\|\hat{x}-x\|_2 / \|x\|_2$ $\|\|r\|\|_2/\|\|b\|\|_2$	
genmatrix2		$\|\hat{x}-x\|_2 / \|x\|_2$ $\|\|r\|\|_2/\|\|b\|\|_2$	
genmatrixc		$\|\hat{x}-x\|_2 / \|x\|_2$ $\|\|r\|\|_2/\|\|b\|\|_2$	

Probleem 4. (Implementatie-opdracht) Schrijf een functie `[Q,R]=qrstep(M)`, die een QR-factorisatie berekent van een matrix $M = [A\ B]$ van de volgende structuur:

$$M = \left[\begin{array}{ccccc|ccc} a_{1,1} & a_{1,2} & \cdots & a_{1,n-1} & a_{1,n} & b_{1,1} & \cdots & b_{1,p} \\ 0 & a_{2,2} & & a_{2,n-1} & a_{2,n} & b_{2,1} & & b_{2,p} \\ \vdots & & \ddots & \vdots & \vdots & \vdots & & \vdots \\ 0 & \cdots & 0 & a_{n-1,n-1} & a_{n-1,n} & b_{n-1,1} & \cdots & b_{n-1,p} \\ a_{n,1} & \cdots & a_{n,n-2} & a_{n,n-1} & a_{n,n} & b_{n,1} & \cdots & b_{n,p} \end{array}\right]$$

De implementatie moet gebeuren met maximaal $n-1$ Givens rotaties. Een Givens rotatie kan berekend worden met de standaard Matlab functie `planerot`.

Test je algoritme (is Q orthogonaal, R bovendriehoeks en is $M = Q * R$?). Vergelijk met het resultaat van de Matlab functie `qr`. Merk op dat dit niet noodzakelijk hetzelfde is, waaruit nogmaals blijkt dat de QR-factorisatie essentieel uniek is. D.w.z. dat de kolommen van Q uniek zijn op het teken na. Wat betekent dit voor de rijen van R?

Hints:

- `A([1 3],:) = G*A([1 3],:)` past de Givens rotatie G toe op de eerste en derde rij van de matrix A.
- Een matrix M met de bovenstaande structuur kan je bv. genereren met

```
M = triu(rand(n,n+p));
M(n,1:n-1) = rand(1,n-1);
```

Probleem 5*. (QR-factorisatie) Door je algoritme $n-1$ keer toe te passen kan je een QR-factorisatie van een willekeurige $n \times (n+p)$-matrix berekenen. Schrijf zulke routine. Nu heb je een alternatief voor het schema uit je cursus, met hetzelfde aantal Givens rotaties.