Numerieke Modellering en Benadering:
Practicum 1

vrijdag 20 maart 2015

In dit practicum onderzoeken we een toepassing van de QR-factorizatie bij een evoluerend kleinste kwadraten probleem en we bekijken enkele methoden voor het bepalen van eigenwaarden van volle matrices.

1 Kleinste kwadraten benadering

Stel

$$A^{(k)} = \begin{bmatrix} A_0 \\ M_1 \\ \vdots \\ M_k \end{bmatrix} \qquad \text{en} \qquad b^{(k)} = \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_k \end{bmatrix},$$

met $A_0 \in \mathbb{R}^{n\times n}$ niet-singulier, $b_0 \in \mathbb{R}^n$ en $M_i \in \mathbb{R}^{d\times n}, b_i \in \mathbb{R}^d$ voor $i = 1, \ldots, k$ and $d \ll n$. We noteren met $x^{(k)} \in \mathbb{R}^n$ de kleinste kwadraten oplossing van het overgedetermineerde stelsel

$$A^{(k)}x^{(k)} = b^{(k)}.$$

In toepassingen met metingen in real-time moeten dit soort overgedetermineerde stelsels opgelost worden voor continu stijgende k. Het doel is om de resultaten uit de berekeningen voor $x^{(k)}$ te kunnen herbruiken bij het berekenen van $x^{(k+1)}$.

Opgave 1. Beschrijf een algoritme, steunend op de QR-factorizatie, om de achtereenvolgende kleinste kwadraten benaderingen $x^{(k)}$ efficiënt te berekenen. Het rekenwerk in het algoritme moet vanaf $k = 1$ even groot zijn in elke stap, bovendien mag het gebruikte geheugen niet toenemen naarmate k groter wordt. Licht ook toe welke transformatie je gebruikt in de QR-factorizatie en schat de hoeveelheid rekenwerk die per stap vereist is. **Hint:** los in elke stap een probleem op van de vorm

$$\begin{bmatrix} R \\ K \end{bmatrix} x = \begin{bmatrix} b \\ c \end{bmatrix}$$

met $R \in \mathbb{R}^{n\times n}$ bovendriehoeks, $K \in \mathbb{R}^{d\times n}$, $b \in \mathbb{R}^n$ en $c \in \mathbb{R}^d$

2 Eigenwaardenproblemen

In heel wat toepassingen maakt men gebruik van de eigenwaarden en eigenvectoren van een volle matrix. Bijvoorbeeld, wanneer men in de statistiek de covariantiematrix van een grote dataset berekent, kunnen de eigenwaarden en eigenvectoren gebruikt worden om de dataset te reduceren zonder belangrijke statistische informatie te verliezen. Eerst onderzoeken we enkele theoretische eigenschappen van de methoden, daarna worden de convergentie-eigenschappen van de methoden onderzocht aan de hand van MATLAB-experimenten.

2.1 Theoretische eigenschappen

Opgave 2. Beschouw de Rayleigh quotiënt iteratie.

a) Gegeven een matrix $A \in \mathbb{R}^{n\times n}$ en een vector $x \in \mathbb{R}^{n\times 1}$, met x een benadering voor een eigenvector van A. Toon aan dat de oplossing $\rho \in \mathbb{R}$ van het minimalisatieprobleem

$$\min_{\rho\in\mathbb{R}} ||Ax - \rho x||_2$$

overeenkomt met het Rayleigh quotiënt van x.

b) Naarmate de benaderende eigenwaarde $\lambda^{(k-1)}$ dichter in de buurt komt van de exacte eigenwaarde, moet er in de inverse iteratie stap van het algoritme een stelsel opgelost worden dat meer en meer singulier wordt. Geef een beknopte (intuïtieve) verklaring waarom het in dit geval geen numerieke problemen geeft een (bijna) singulier stelsel op te lossen.

2.2 Convergentie-experimenten

Op Toledo vind je een aantal MATLAB-programma's die je kan gebruiken voor de rest van het practicum. Plaats de bestanden in een directory en gebruik in MATLAB het commando `cd` om naar deze directory te gaan. Lees grondig de documentatie bij de MATLAB-programma's. Typ `help pract` voor meer informatie.

Je kan experimenten uitvoeren door commando's in te geven aan de MATLAB invoerregel, maar het is aangewezen om zelf m-files (scripts en functies) te schrijven. Op die manier kan je gemakkelijk wijzigingen aanbrengen en vermijd je het herhaald intypen van gelijkaardige bevelen. Met het commando `help` kan je wat meer te weten komen over een bepaald programma. Je mag gerust de code van de MATLAB-programma's op Toledo aanpassen.

Enkele nuttige commando's zijn: `help`, `lookfor`, `figure`, `subplot`, `plot`, `semilogy`, `semilogx`, `loglog`, `xlabel`, `ylabel`, `title`, `legend`, `print`, `save`, `load`. Voor nog meer nuttige commando's zie ook de MATLAB inleiding op Toledo.

In het tweede deel van het practicum beschouwen we een gegeven volle, symmetrische matrix. Een voorbeeld is te vinden in het bestand `mat1.txt`. Deze matrix kan in MATLAB ingeladen worden met behulp van het commando `load mat1.txt`.

Opgave 3. Beschouw de matrix `mat1.txt`. Reduceer deze matrix naar Hessenberg vorm en toon de structuur met behulp van `spy`. Waarom is het belangrijk om de matrix te reduceren naar Hessenbergvorm alvorens de QR-methode toe te passen?

Opgave 4. In deze opgave bekijken we

- de QR-methode zonder shift,
- de QR-methode met Rayleigh quotiënt shift, en
- de QR-methode met Wilkinson shift.

Pas deze methoden toe om alle eigenwaarden van de matrix `mat1.txt` te berekenen. Illustreer grafisch en in tabelvorm de convergentie. Is de convergentie lineair, kwadratisch of kubisch? Komen de numerieke resultaten overeen met de theorie? Bespreek.

Opgave 5. De cursus vermeldt een verband

- tussen het QR-algoritme met Rayleigh quotiënt shift en de Rayleigh quotiënt iteratie, en
- tussen het QR-algoritme en de gelijktijdige iteratie.

Verduidelijk aan de hand van de matrix `mat1.txt` in welk opzicht het convergentiegedrag van het *QR-algoritme met Rayleigh quotiënt shift* gelijkenissen vertoont met de *gelijktijdige iteratie* en de *Rayleigh quotiënt iteratie*.

Opgave 6. Naast (quasi)-directe methoden om eigenwaardenproblemen op te lossen, kunnen we ook iteratieve methoden beschouwen, zoals de Arnoldi methode. Iteratieve methoden zijn in het bijzonder geschikt wanneer de matrix een ijle structuur heeft. Genereer een random, ijle matrix $A \in \mathbb{R}^{1000\times 1000}$ met het commando `sprand`. Pas een aantal Arnoldi-iteratiestappen toe op deze matrix (bv. 100). Bereken per iteratiestap de Ritz waarden. Daarbij mag je gebruik maken van het ingebouwde MATLAB-commando `eig`. Maak een grafiek waarin je toont hoe de Ritz waarden per iteratiestap convergeren naar de eigenwaarden van A. Toon daarbij enkel de reële delen. Bespreek bondig het convergentiegedrag.

2.3 Alternatieve eigenwaardenalgoritmen

Naast de besproken methoden in lecture 27 t.e.m. lecture 29 van het boek 'Numerical Linear Algebra' (L. Trefethen en D. Bau, 1997) bestaan er nog andere belangrijke methoden voor het oplossen van eigenwaardenproblemen. Enkele voorbeelden zijn bisectie, Jacobi, verdeel- en heers en Arnoldi/Lanczos. In dit gedeelte van het practicum bekijken we de bisectie-methode in meer detail.

Opgave 7. De bisectie-methode steunt op de 'interlacing'-eigenschap van de eigenwaarden van een tridiagonale, symmetrische matrix. Wat is die eigenschap en illustreer voor een zelf gekozen voorbeeld. Geef ook aan welk voorbeeld je gebruikt hebt.

Opgave 8. Schrijf een algoritme uit in pseudo-code om met de bisectie-methode al de eigenwaarden van een symmetrische matrix A in een interval $[a, b)$ te bepalen. Elke eigenwaarde moet bepaald worden met een fout kleiner dan een opgeven tolerantie `tol`. Implementeer dit algoritme als een functie in MATLAB: `function E = bisection(A,a,b,tol)`, met `E` de vector van gevonden eigenwaarden. Toon de correctheid van je implementatie aan door je code toe te passen op enkele 'interessante' testproblemen. Beschrijf duidelijk welke testproblemen je gekozen hebt en waarom.

Opgave 9. Genereer een aantal symmetrische, random, tridiagonale matrices van verschillende groottes (bijvoorbeeld 20, 40, 80, 160, 320, ...). Bepaal met het QR-algoritme met Rayleigh quotiënt shift en met de bisectie-methode 7 eigenwaarden en de bijhorende eigenvectoren van deze matrices. Gebruik inverse iteratie om de eigenvectoren te bepalen met het bisectie algoritme. Maak een grafiek van de rekentijd van beide methoden in functie van de matrixgrootte. Wat observeer je?

Praktische richtlijnen

- Dit practicum wordt gemaakt in groepjes van twee.
- Het verslag kan je elektonisch inleveren via mail ten laatste op **vrijdag 24 april 2015** om 15:00 uur.
- Gebruik figuren en tabellen om je bevindingen te verduidelijken. Gebruik logaritmische grafieken (`semilogx`, `semilogy`, `loglog`) waar nodig. Kies een gepaste schaal voor je figuren, vooral als twee verschillende figuren vergeleken moeten worden.
- Bespreek bondig je resultaten en je aanpak in een duidelijke vergezellende tekst. Hou de lengte evenwel onder de 15 pagina's, inclusief tabellen en figuren.
- Voeg naast de pseudo-code ook je MATLAB-code (de m-file) voor de bisectie-methode uit opgave 8 toe aan je verslag.

Veel succes!

Ben Jeuris (200A 01.160)
`ben.jeuris@cs.kuleuven.be`
`http://toledo.kuleuven.be`