Kuis 4 Analisis Numerik

September 22, 2021

Suatu matriks T berukuran $n \times n$ disebut matriks tridiagonal jika $T_{ij} = 0$ untuk setiap $i, j \in \{1, 2, \dots, n\}$ dengan $|i - j| > 1$, misalnya

$$T = \begin{pmatrix} 1 & 2 & 0 & 0 & 0 & 0 \\ 3 & 4 & 5 & 0 & 0 & 0 \\ 0 & 6 & 7 & 8 & 0 & 0 \\ 0 & 0 & 9 & 0 & 1 & 0 \\ 0 & 0 & 0 & 2 & 3 & 4 \\ 0 & 0 & 0 & 0 & 5 & 6 \end{pmatrix}$$

adalah sebuah matriks tridiagonal.

Program `lutrid.m` menerima masukan sebuah matriks tridiagonal T dan memberikan luaran matriks segitiga bawah L dan segitiga atas U sehingga $T = LU$. Jawablah masing-masing pertanyaan berikut di dalam sebuah berkas yang dispesifikasi penamaan dan formatnya di awal setiap nomor pertanyaan.

0. [soal0.txt, 10 poin] Jalankan lutrid.m untuk matriks T di atas Tuliskan nilai L dan U yang Anda peroleh pada berkas soal0.txt. Berikan komentar, apakah matriks segitiga bawah L memiliki bentuk spesial? Bagaimana dengan U?

1. [soal1.txt, 10 poin] Dinyatakan dalam n, berapa kalikah operasi +,-,*,/ dilakukan dalam satu eksekusi `lutrid(T)`? Lakukan perhitungan eksak dalam n; jangan gunakan notasi asimtotik. Berikan penjelasan perhitungan baris per baris untuk masing-masing operasi. [Petunjuk: Untuk mendapatkan atau memverifikasi polanya, Anda boleh tambahkan beberapa variabel `count_add, count_mul`, dll. untuk masing-masing operasi yang mencacah berapa kali ia dilakukan.]

   Catatan: Jika c skalar dan u adalah vektor berukuran $n \times 1$, maka operasi perkalian skalar-vektor $c * u$ akan menjalankan n operasi perkalian floating-point numbers. Begitu pula jika u dan v adalah vektor berukuran $n \times 1$, maka operasi tambah vektor $u + v$ akan menjalankan n operasi pejumlahan floating-point numbers.

2. [soal2.m, 10 poin] Modifikasi implementasi lutrid.m agar operasi floating point $+, -, *, /$ masing-masing dilakukan sebanyak $O(n)$ kali untuk matriks T berukuran $n \times n$. Ini berarti faktorisasi LU untuk matriks tridiagonal memiliki kompleksitas flop $O(n)$, yang jauh lebih efisien pada kasus umum yang memiliki kompleksitas $O(n^3)$. File `soal2.m` merupakan fungsi yang dapat menerima parameter T.

3. [soal3.m, 10 poin] Buat sebuah m-script yang:

   (a) menginisiasi nilai $n = 100$;

   (b) membangkitkan matriks tridiagonal T berukuran $n \times n$ secara acak

   (c) membangkitkan vektor b berukuran $n \times 1$ secara acak

   (d) menyelesaikan SPL $Tx = b$ melalui faktorisasi LU pada T terlebih dahulu

   (e) mencetak L, U, x yang dihasilkan dan juga backward error dari solusi x.

   Saat dieksekusi, soal3.m harus melakukan hanya $O(n)$ flop secara keseluruhan.

4. [soal4.txt, 20 poin] Lakukan faktorisasi LU umum dengan partial pivoting pada contoh matriks tridiagonal T di atas. Anda boleh gunakan fungsi `lu` umum atau implentasi yang ada di kelas pada MATLAB/Octave yang dapat mengembalikan P, L, dan U. Tuliskan masing-masing nilai P, L, dan U pada berkas soal4.txt. Berikan komentar, apakah matriks segitiga bawah L masih memiliki bentuk spesial? Bagaimana dengan U? Apa efek partial pivoting secara umum pada matriks tridiagonal?

5. [soal5.m, 20 poin] Lengkapi algoritma pada `soal2.m` dengan fitur partial pivoting. Perhatikan bahwa melakukan pivoting dapat sedikit[1] merusak sifat tridiagonal dari matriks yang diperoleh, namun kompleksitas flop yang $O(n)$ masih dapat dipertahankan. Berkas `soal5.m` merupakan fungsi yang menerima parameter T.

6. [soal6.m, 20 poin] Lakukan lagi perhitungan penyelesaian SPL soal3.m namun sekarang menggunakan algoritma soal5.m yang faktorisasi LU-nya dilengkapi partial pivoting. Saat dieksekusi, soal6.m harus melakukan hanya $O(n)$ flop.

Pengayaan (tidak dinilai): Matriks tridiagonal T hanya membutuhkan $O(n)$ memori alih-alih $O(n^2)$ untuk matriks umum $n \times n$. Matriks T dapat dipandang sebagai tiga buah vektor a, b, c:

$$
\begin{aligned}
a &= [T_{11}\, T_{22}\, \ldots\, T_{nn}] \\
b &= [T_{21}\, T_{32}\, \ldots\, T_{n,n-1}] \\
c &= [T_{12}\, T_{23}\, \ldots\, T_{n-1,n}]
\end{aligned}
$$

Dapatkah Anda implementasi algoritma-algoritma di atas dengan menggunakan strategi penyimpanan seperti ini dan tetap hanya membutuhkan $O(n)$ flop?

Aspek lain yang menarik adalah, untuk mengetahui bagaimana perubahan yang terjadi pada faktorisasi LDLT atau Cholesky jika diterapkan pada matriks tridiagonal secara efisien.

Salah satu soal tugas kelompok akan menunjukkan bagaimana matriks tridiagonal atau yang lebih umum, matriks pita, dapat muncul pada permasalahan pemodelan matematis dari dunia nyata.

[1] Tugas Anda pada soal 4 adalah memahami efek pivoting ini pada faktorisasi LU dari matriks tridiagonal.