Vv471
Dr Jing Liu

Assignment 3
Due: Jun 11, 2019

Question1 (20 points)

Suppose $\mathbf{A}$ is a matrix of $n \times n$. Write a piece of pseudo-code in computing $\det(\mathbf{A})$ by each of the followings. In the worst scenario, how many additions/subtractions and multiplications/divisions are needed in computing $\det(\mathbf{A})$ by each of the followings?

(a) (5 points) Leibniz's method. (See this wiki page)

(b) (5 points) Laplace's method. (Cofactor expansion)

(c) (5 points) Gauss's method. (Gaussian Elimination)

(d) (5 points) Dodgson's method. (See the description below)

Dodgson's method involves creating smaller and smaller matrixes to obtain the determinant:

$\mathbf{A}^{(0)}$ by removing all 0s from the interior of $\mathbf{A}$ using elementary row/column operations.

$\mathbf{A}^{(1)}$ by using the corresponding consecutive minors of 2×2 of $\mathbf{A}^{(0)}$.

$\mathbf{A}^{(2)}$ by using the corresponding consecutive minors of 2×2 of $\mathbf{A}^{(1)}$.

$\mathbf{A}^{(3)}$ by dividing each element of $\mathbf{A}^{(2)}$ by the corresponding element of the interior of $\mathbf{A}^{(0)}$.

$\mathbf{A}^{(4)}$ by using the corresponding consecutive minors of 2×2 of $\mathbf{A}^{(3)}$.

$\mathbf{A}^{(5)}$ by dividing each element of $\mathbf{A}^{(4)}$ by the corresponding element of the interior of $\mathbf{A}^{(1)}$.

Set $\mathbf{A}^{(1)} = \mathbf{A}^{(3)}$ and $\mathbf{A}^{(3)} = \mathbf{A}^{(5)}$, repeat the last 2 steps until a single scale is obtained.

The interior of a matrix is the submatrix formed by removing the first and last rows as well as the first and last columns of the matrix. A consecutive minor of 2×2 of a matrix $\mathbf{A}$ is the determinant of any 2×2 submatrix consisting the 4 elements in any two adjacent rows and adjacent columns from the original matrix $\mathbf{A}$. For a matrix of $n \times n$, notice there are

$$(n-1)^2$$

such minors, and those minors can be put into a matrix of $(n-1) \times (n-1)$ according to the relative positions of the two adjacent rows and two adjacent columns in the Dodgson's method.

Question2 (2 points)

Given the Dodgson's method above, what is the key difference between Dodgson's method and Gauss's method that might make Dodgson's method the preferred method under some circumstances?

Question3 (2 points)

Suppose $\mathbf{A} = \mathbf{P}^{\mathrm{T}}\hat{\mathbf{L}}\mathbf{U}$, where $\mathbf{P}$ is a permutation matrix, $\hat{\mathbf{L}}$ is unit lower triangular matrix, and $\mathbf{U}$ is an upper triangular matrix.

(a) (2 points) Determine the number of additions/subtractions and multiplications/divisions needed to obtain the factorisation using the algorithm given in class.

Question4 (12 points)

It can be shown that using LU takes approximately $\frac{2}{3}n^3$ flops. This is better than calculating $\mathbf{A}^{-1}\mathbf{b}$, which takes approximately $2n^3$ flops, thus both methods are $O(n^3)$. This raises the question as to whether there are sub-cubic methods, in other words, can you find a method that requires $O(n^\omega)$ flops, where $\omega < 3$? Considerable research has been invested in this question, and the usual approach to answering it is to change the question. It can be proved that if you can multiply two $n \times n$ matrices using $O(n^\omega)$ flops then you can solve $\mathbf{Ax} = \mathbf{b}$ using $O(n^\omega)$ flops. The direct method for multiplying two matrices requires n^3 multiplications/divisions and $n^3 - n^2$ additions/subtractions, in other words, $O(n^3)$ flops. The cubic barrier for matrix multiplication was first broken by Strassen, who was able to produce an algorithm that uses $O(n^\omega)$ flops, where $\omega = \log_2 7 \approx 2.8$. This question is about this algorithm, which is often known as the fast matrix multiplication algorithm or Strassen's method. Consider the block matrix product,

$$\begin{bmatrix} \mathbf{W} & \mathbf{X} \\ \mathbf{Y} & \mathbf{Z} \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} \begin{bmatrix} \mathbf{E} & \mathbf{F} \\ \mathbf{G} & \mathbf{H} \end{bmatrix}$$

where for simplicity, all the matrices are assumed to be square and of the same size.

(a) (2 points) Given $\mathbf{A}, \mathbf{B}, \ldots, \mathbf{G}, \mathbf{H}$, how many matrix additions/subtractions, and matrix multiplications/divisions does it take to compute the product directly.

(b) (2 points) Find $\mathbf{W}$, $\mathbf{X}$, $\mathbf{Y}$ and $\mathbf{Z}$ in terms of $\mathbf{P}_i$, where $\mathbf{P}_i$s are given by

$$\mathbf{P}_1 = (\mathbf{A} + \mathbf{D})(\mathbf{E} + \mathbf{H}) \qquad \mathbf{P}_5 = (\mathbf{A} + \mathbf{B})\mathbf{H}$$
$$\mathbf{P}_2 = (\mathbf{C} + \mathbf{D})\mathbf{E} \qquad \mathbf{P}_6 = (\mathbf{C} - \mathbf{A})(\mathbf{E} + \mathbf{F})$$
$$\mathbf{P}_3 = \mathbf{A}(\mathbf{F} - \mathbf{H}) \qquad \mathbf{P}_7 = (\mathbf{B} - \mathbf{D})(\mathbf{G} + \mathbf{H})$$
$$\mathbf{P}_4 = \mathbf{D}(\mathbf{G} - \mathbf{E})$$

(c) (2 points) How many matrix additions/subtractions, and matrix multiplications/divisions does it take to compute the product using Strassen's method.

(d) (2 points) Why do you think Strassen's methods is faster for large matrices?

(e) (2 points) Show by applying Strassen's method recursively, one can obtain an algorithm for multiplying matrices of size $n \times n$, where $n = 2^k$ with an operation count $O(n^{\log_2 7})$ as $n \to \infty$.

(f) (2 points) Write a recursive program in Matlab that implements this idea.

- For a matrix whose size is not $2^k \times 2^k$, you can simply add 0s to the matrix.
- If you test your implementation, it is very likely that your implementation is slower than Matlab's implement, which uses highly optimised Fortran/C implementation.

Question5 (4 points)

Imagine you are the instructor for an introductory course on linear algebra. To please your students, you want to focus on concepts rather than tedious arithmetics. How would you generate a random square matrix $\mathbf{A}$ such that $\mathbf{A}$ is invertible and $\mathbf{A}^{-1}$ as well as $\mathbf{A}$ has only integer elements?