Determinants

Group 12

Catterwell, A.   Smith, M.   Wang, R.   Watson, K.

University of Edinburgh

March 29, 2019

Overview

1. Using LU decomposition to compute determinants
   - What is LU decomposition?
   - How it helps us compute determinants
2. Other algorithms for computing determinants
   - Laplace Expansion
   - Leibniz Formula
   - Gaussian Elimination
   - Bareiss Algorithm
   - Bird's Algorithm
3. Summary
   - A comparison of the determinant algorithms

What is LU decomposition?

Somebody else do this section

- What LU decomposition is
- Its limitations
- How PLU decomposition addresses these limitations

How it helps us compute determinants

some maths.

Laplace Expansion

The Laplace (1st row) expansion for computing determinants is usually the first method taught for computing determinants of 3×3 matrices and larger.

Theorem

The formula for the (1st row) Laplace expansion of $A \in \mathrm{Mat}(n, \mathbb{R})$ is given as:

$$|A| = \sum_{j=1}^{n} a_{1j} \cdot C_{1j}$$

where C_{ij} is the (i, j) cofactor of A.

Its runtime complexity of $\mathcal{O}(N!)$ is poor.

Leibniz Formula

Definition

The Leibniz formula defines the determinant of an $n \times n$ matrix A as follows:

$$|A| = \sum_{\sigma \in \mathfrak{S}_n} \mathrm{sgn}(\sigma)_{i=1}^{n} a_{i\sigma(i)}$$

where $\mathfrak{S}_n$ is the set of permutations length n.

Computing the determinant using this method is slow with runtime $\mathcal{O}((N+1)!)$.

Gaussian Elimination

- The determinant of a triangular matrix can be computed by taking the product of its diagonal entries (which is a quick $\mathcal{O}(N)$ operation).
- Any invertible square matrix can be transformed into echelon form by performing Gaussian elimination, which has runtime $\mathcal{O}(N^3)$.

Conventional Gaussian elimination requires division, meaning that solutions maybe inexact, so precision is lost.
This is addressed by. . .

Bareiss Algorithm

- Addresses the issue of precision-loss by performing *integer-preserving* Gaussian elimination on integer matrices.
- The runtime complexity is $\mathcal{O}(N^3)$ which is the same as conventional Gaussian Elimination, whilst preserving exactness.
- To be continued.

Bird's Algorithm

Define $\mu : \mathrm{Mat}(n, \mathbb{R}) \to \mathrm{Mat}(n, \mathbb{R})$:

$$\mu(X) = \begin{bmatrix} \mu_{2,2} - x_{2,2} & x_{1,2} & \cdots & x_{1,n-1} & x_{1,n} \\ 0 & \mu_{3,3} - x_{3,3} & \cdots & x_{2,n-1} & x_{2,n} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & \mu_{n,n} - x_{n,n} & x_{n-1,n} \\ 0 & 0 & \cdots & 0 & 0 \end{bmatrix}$$

and $F_A : \mathrm{Mat}(n, \mathbb{R}) \to \mathrm{Mat}(n, \mathbb{R})$, with $A \in \mathrm{Mat}(n, \mathbb{R})$

$$\begin{aligned} F_A(X) &= \mu(X) \cdot A \\ F_A^2(X) &= \mu(F_A(X)) \cdot A \\ &\vdots \\ F_A^n(X) &= \mu(F_A^{n-1}(X)) \cdot A \end{aligned}$$

Bird's Algorithm (cont.)

Bird's Theorem

$$F_A^{n-1}(A) = \begin{bmatrix} d & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 \end{bmatrix} \text{ with } d = \begin{cases} |A| & \text{odd } n \\ -|A| & \text{even } n \end{cases}$$

- Enables the *division-free* computation of determinants in $\mathcal{O}(n \cdot M(n))$ where $M(n)$ is the runtime complexity of the matrix multiplication algorithm used.
- Given a good matrix multiplication algorithm with runtime $\mathcal{O}(n^{2.376})$ (*Coppersmith-Winograd*), this algorithm runs in $\mathcal{O}(n^{3.376})$.

A comparison of the determinant algorithms

Algorithm	*Runtime*	*Exact*
Laplace Expansion	$\mathcal{O}(N!)$	Yes
Leibniz Formula	$\mathcal{O}((N+1)!)$	Yes
LU Decomposition	$\mathcal{O}(N^3)$	No
Gaussian Elimination	$\mathcal{O}(N^3)$	No
Bareiss' Algorithm	$\mathcal{O}(N^3)$	Yes
Bird's Algorithm	$\mathcal{O}(N^{3.376})$	Yes