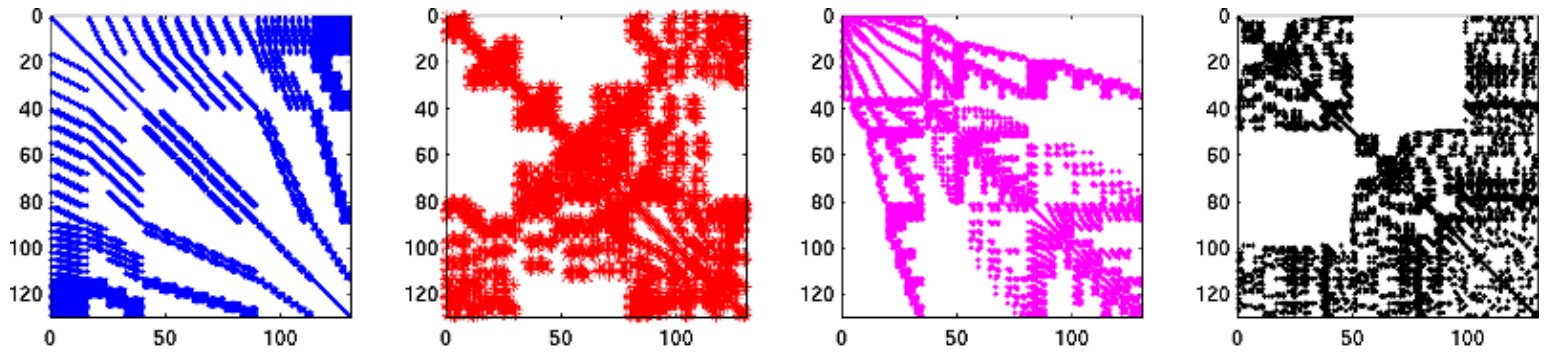

MA385 Part 3: Linear Algebra 1

3.1: Introduction to solving linear systems

Dr Niall Madden

October 2025

0.

For more, see Section 2.1 of Suli and Mayers:
https://ebookcentral.proquest.com/lib/nuig/reader.action?docID=221072&ppg=51&c=UERG

1. Solving Linear Systems

This section of the course is concerned with solving systems of linear equations ("simultaneous equations"). All problems are of the form: *find the set of real numbers* $x_1, x_2, \ldots, x_n$ *such that*

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n &= b_n \end{aligned}$$

where the a_{ij} *and* b_i *are real numbers.*
It is natural to rephrase this using the language of linear algebra:
Find $\mathrm{x} = (x_1, x_2, \ldots, x_n)^T \in \mathbb{R}^n$ *such that*

$$A\mathrm{x} = \mathrm{b}, \qquad \text{where } A \in \mathbb{R}^{n\times n}, \text{ and } \mathrm{b} \in \mathbb{R}^n. \tag{1}$$

1

1. Solving Linear Systems

In this section, we'll try to find a clever way to solving this system. In particular,

1. we'll argue that its unnecessary and (more importantly) expensive to try to compute A^{-1};
2. we'll have a look at **Gaussian Elimination**, but will study it in the context of LU-factorisation;
3. after a detour to talk about **matrix norms**, we calculate the **condition number** of a matrix;
4. this last task will require us to be able to estimate the **eigenvalues** (and singular values) of matrices: that is the main subject of the next and final section of the module.

2. A short review of linear algebra

- A vector $\mathrm{x} \in \mathbb{R}^n$, is an ordered n-tuple of real numbers, $\mathrm{x} = (x_1, x_2, \ldots, x_n)^T$.

- A matrix $A \in \mathbb{R}^{m \times n}$ is a rectangular array of n columns of m real numbers. (But we will only deal with **square** matrices, i.e., ones where $m = n$).

- You already know how to multiply a matrix by a vector, and a matrix by a matrix (right??).

2. A short review of linear algebra

- You can write the **scalar** product as $(\mathrm{x}, \mathrm{y}) = \mathrm{x}^T \mathrm{y} = \sum_{i=1}^{n} x_i y_i$.

- Recall that $(A\mathrm{x})^T = \mathrm{x}^T A^T$, so
$(\mathrm{x}, A\mathrm{y}) = \mathrm{x}^T A\mathrm{y} = (A^T \mathrm{x})^T \mathrm{y} = (A^T \mathrm{x}, \mathrm{y})$.

- Letting I be the identity matrix, if there is a matrix B such that $AB = BA = I$, when we call B the **inverse** of A and write $B = A^{-1}$. If there is no such matrix, we say that A is **singular**.

Theorem 3.1.1

The following statements are equivalent.

1. For **any** b, $Ax = b$ has a solution.
2. If there is a solution to $Ax = b$, it is unique.
3. If $Ax = 0$ then $x = 0$.
4. The columns (or rows) of A are linearly independent.
5. There exists a matrix $A^{-1} \in \mathbb{R}^{n \times n}$ such that $AA^{-1} = I = A^{-1}A$.
6. $\det(A) \neq 0$.
7. A has rank n.

Item 5 of the above lists suggests that we could solve $Ax = b$ as follows: find A^{-1} and then compute $x = A^{-1}b$. However this is not the best way to proceed. We only need x and computing A^{-1} requires quite a bit a work.

3. The time it takes to compute det(A)

In introductory linear algebra courses, you might compute

$$A^{-1} = \frac{1}{\det(A)} A^*$$

where A^* is the adjoint matrix. But it turns out that computing $\det(A)$ directly can be **very** time-consuming.
Let d_n be the number of multiplications required to compute the determinant of an $n \times n$ matrix using the **Method of Minors** (also called "the Laplacian determinant expansion"). We know that if

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

then $\det(A) = ad - bc$. So $d_2 = 2$.

3. The time it takes to compute det(A)

If

$$A = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & j \end{pmatrix}$$

then

$$\det(A) = \begin{vmatrix} a & b & c \\ d & e & f \\ g & h & j \end{vmatrix} = a \begin{vmatrix} e & f \\ h & j \end{vmatrix} - b \begin{vmatrix} d & f \\ g & j \end{vmatrix} + c \begin{vmatrix} d & e \\ g & h \end{vmatrix},$$

so $d_3 \geq 3d_2$.

3. The time it takes to compute det(A)

3. The time it takes to compute det(A)

In general we find that it takes $(n)(n-1)\cdots(4)(3)(2)(1) = n!$ operations to compute the determinant this way. And that is a lot of operations.

4. Exercises

Exercise 3.1.1

Suppose you had a computer that computer perform 2 billion operations per second. Give a lower bound for the amount of time required to compute the determinant of a 10-by-10, 20-by-20, and 30-by-30, matrix using the method of minors.

Exercise 3.1.2

The Cauchy-Binet formula for determinants tells us that, if A and B are square matrices of the same size, then $\det(AB) = \det(A)\det(B)$. Using it, or otherwise, show that $\det(\sigma A) = \sigma^n \det(A)$ for any $A \in \mathbb{R}^{n\times n}$ and any scalar $\sigma \in \mathbb{R}$. *Note: this exercise gives us another reason to avoid trying to calculate the determinant of the coefficient matrix in order to find the inverse, and thus the solution to the problem. For example $A \in \mathbb{R}^{16\times 16}$ and the system is rescaled by $\sigma = 0.1$, then $\det(A)$ is rescaled by 10^{-16}. On standard computing platforms, like numpy/scipy/MATLAB, it would be indistinguishable from a singular matrix!*