Numerical Linear Algebra and Sparse Matrices

Víctor Ballester Ribó v.ballester-ribo24@imperial.ac.uk
Joseph Whitaker Schaefer j.schaefer24@imperial.ac.uk

Department of Aeronautics
Imperial College London

We live in a world driven by matrices - Equations describing global economies, AI models, computer graphics, and more are regularly posed and solved in matrix form. A matrix can be defined as the mathematical object used to represent a system of linear equations. Here are a few examples:

$$y = 3x + 2 \qquad (y) = (3)\,(x) + (2)$$ 1-dimensional matrix

$$\begin{cases} y_1 = 4x_1 + 2x_2 + 7 \\ y_2 = -x_1 + 2 \end{cases} \qquad \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} 4 & 2 \\ -1 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} 7 \\ 2 \end{pmatrix}$$ 2-dimensional matrix

$$\begin{cases} y_1 = 7x_1 + x_2 - 1/2x_3 + 2 \\ y_2 = 4x_1 - 5x_3 \\ y_3 = 3/2x_1 + 3x_2 - 4x_3 + 1 \end{cases} \qquad \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} 7 & 1 & -1/2 \\ 4 & 0 & -5 \\ 3/2 & 3 & -4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} 2 \\ 0 \\ 1 \end{pmatrix}$$ 3-dimensional matrix

In practice, the vector in the left-hand side of the equations is known, and so we need to find the vector $\boldsymbol{x}$ by solving the system of equations

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}.$$

Since $\boldsymbol{y}$ is known, we can replace $\boldsymbol{y} - \boldsymbol{b}$ by a new, and also known, vector $\tilde{\boldsymbol{y}}$ and solve for $\boldsymbol{x}$ in the system

$$\tilde{\boldsymbol{y}} = \boldsymbol{A}\boldsymbol{x}.$$

This implies we can just consider the problem of solving homogeneous systems of equations, where $\boldsymbol{b} = \boldsymbol{0}$.

A first approach to solve these systems might be to extend the classical definition of inverse of a number ($\frac{1}{a} \cdot a = a \cdot \frac{1}{a} = 1$) to matrices, and then use the inverse of $\boldsymbol{A}$ to find $\boldsymbol{x}$ as

$$\boldsymbol{x} = (\boldsymbol{A}^{-1}\boldsymbol{A})\boldsymbol{x} = \boldsymbol{A}^{-1}(\boldsymbol{A}\boldsymbol{x}) = \boldsymbol{A}^{-1}\tilde{\boldsymbol{y}}.$$

We will see that this approach is not efficient for large size matrices. An alternative approach is to use the Gaussian elimination method, which consists of isolting every variable in terms of the next ones. For example in the 3-dimensional matrix example above, we can isolate x_1 in the first equation as

$$x_1 = 1/7(y_1 - x_2 + 1/2x_3 - 2),$$

and then substitute this expression in the second and third equations to obtain a system of two equations with two unknowns. Then, isolate x_2, as a function of x_3, in one of the two resulting equations and introduce it to the remaining equation to obtain a linear system of one equation with one unknown. Finally, we can isolate x_3 and substitute it back in the previous equations to find x_2 and x_1.

The first aim of this project will explore and implement some of the many advanced methods for solving systems of matrix equations in a programming language of your choice.

The second aim will be to extend this work to cover some forms of sparse matrices, a matrix in which most of the elements are zero. See the example below of a sparse linear system of dimension n:

$$\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_{n-1} \\ y_n \end{pmatrix} = \begin{pmatrix} 2 & -1 & 0 & \cdots & 0 \\ -1 & 2 & -1 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & -1 & 2 & -1 \\ 0 & \cdots & 0 & -1 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{pmatrix}.$$

The density (e.g. number of non-zero elements divided by the total number of elements) of this matrix is $\frac{3n-2}{n^2}$, which tends to 0 as n goes to infinity. Because of this, using the same algorithms as for dense (e.g. matrices with many non-zero elements) matrices is not efficient, taking into account the prior knowledge of the null values. Contrary to what one might think, sparse matrices are used in many applications, such as solving differential equations (e.g. heating up your cup of milk) or graphs theory (e.g. finding the best route through Google Maps).

In this part of the project we will use algorithms specifically designed to solve sparse matrices, and the comparison of the performance of these algorithms with the ones used for dense matrices, in terms of the density of the matrix and the size of the matrix.

This project will suit students with interested in growing their applied mathematics experience through the formulation and implementation of algorithms for scientific computing. Interest in low-level (C, Rust, C++, ...) or high-level (Python, Matlab, Julia, ...) programming languages is useful, but not essential.

Below are materials regarding general numerical linear algebra [TB97] and sparse matrices [Dav06] for further reading.

References

[Dav06] Timothy A. Davis. *Direct Methods for Sparse Linear Systems*. Vol. 2. Fundamentals of Algorithms. SIAM, 2006. ISBN: 97-80898-716-1-3-9. DOI: 10.1137/1.9780898718881.

[TB97] Lloyd N. Trefethen and III David Bau. *Numerical Linear Algebra*. SIAM, 1997. ISBN: 97-80898-713-6-1-9. DOI: 10.1137/1.9780898719574.