Computational Physics 301 (2018-19)

Exercise 1: Matrices and Linear Algebra

Submission deadline: Tuesday 19th Feb. 2019, 17:00

1 Objectives of Exercise 1

The manipulation of matrices is a core topic in numerical analysis. Matrix methods can be used to solve many classes of physical problems, although (as with all numerical techniques) care has to be taken in the choice and implementation of appropriate algorithms.

This exercise introduces simple matrix techniques to solve mathematical problems and investigates some of the considerations of stability and efficiency in choosing appropriate algorithms. This is then applied to a specific physics problem.

You should address all the Tasks and Physics Problems outlined below in your report.

2 Matrix Inversion for Linear Algebra

A set of simultaneous linear equations can always be written as a matrix equation. For example, two equations in two unknowns (x_1 and x_2)

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 &= b_1 \\ a_{21}x_1 + a_{22}x_2 &= b_2 \end{aligned}$$

can be rewritten as

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

An arbitrary set of equations is

$$\mathrm{A}\mathbf{x} = \mathbf{c} \tag{1}$$

where A is the matrix of coefficients, $\mathbf{x}$ is the vector of unknown variables $x_1, x_2, ...$ and $\mathbf{b}$ is the known vector of constants. The solution to Equation 1 can be obtained by pre-multiplying by the inverse of A:

$$\begin{aligned} \mathrm{A}^{-1}\mathrm{A}\mathbf{x} &= \mathrm{A}^{-1}\mathbf{b} \\ \Leftrightarrow \mathbf{x} &= \mathrm{A}^{-1}\mathbf{b} \end{aligned}$$

Task 1

Write **your own routine** to calculate the inverse of an arbitrary square $n \times n$ matrix (with n at least in the range $2 \leq n \leq 4$) using the standard inversion formula:

$$\mathrm{A}^{-1} = \frac{1}{\det \mathrm{A}} \mathrm{C}^T$$

where C is the matrix of cofactors of A. In the process of implementing this, you may become aware of a way to perform this calculation for an arbitrary $n \times n$ matrix. Verify that your routine works by testing it against a few hand-calculated inversions, and then investigate its performance: how accurate is it, how robust is it and how does the time it takes scale with the number of rows? **Tip :** You will find it useful later if you use numpy *ndarray* for your matrices.

3 Algorithms for Linear Algebra

A number of algorithms are used in preference to the analytical inversion method. A basic technique is "Gauss-Jordan elimination" in which a multiple of one row is subtracted from another to eliminate variables (just as you would do in solving simultaneous equations by hand). Two further common techniques are LU Decomposition and Singular Value Decomposition.

3.1 LU Decomposition

The solution to Equation 1 is straightforward if the matrix is triangular, i.e. all of the entries either above or below the diagonal are zero. For example, with a lower-diagonal matrix:

$$\begin{pmatrix} a_{11} & 0 & 0 & \dots \\ a_{21} & a_{22} & 0 & \dots \\ \vdots & & & \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \end{pmatrix}$$

The first row directly gives the solution $x_1 = b_1/a_{11}$. This can then be substituted into the second row to give x_2 and so on in similar fashion.

LU decomposition was introduced by Alan Turing in 1948, and works by reformulating the general matrix A as the product LU of a lower- and an upper-triangular matrix. For example, if :

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}, L = \begin{pmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{pmatrix}, U = \begin{pmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{pmatrix}$$

The general solution to Equation 1 is then reduced to the successive solution of two triangular equations:

$$\mathrm{L}\mathbf{y} = \mathbf{b}$$

and

$$\mathrm{U}\mathbf{x} = \mathbf{y}$$

3.2 Singular Value Decomposition

SVD involves writing a general matrix in the form

$$\mathrm{A} = \mathrm{USV}$$

where U and V are orthnormal matrices, and S is a diagonal matrix containing the "singular" values.

Task 2 :

Compare the speed of LUD, SVD and the analytic inversion formula in solving simultaneous linear equations. You should use LUD and SVD routines from scipy.linalg. In particular, investigate how the speed scales with the number of rows. Also investigate the behaviour when the set of equations is close to singular, e.g.

$$\begin{aligned} x + y + z &= 5 \\ x + 2y - z &= 10 \\ 2x + 3y + kz &= 15 \end{aligned}$$

for small k.

Physics Problem:

A trapeze artist is suspended above a stage by a system of 3 wires. Each wire is fed from a motorised drum, such that the artist appears to fly around above the stage by controlling the length of the wires. The stage is 15 m wide, 8 m deep and the wire drums are 8 m above the stage. Two drums are fixed above the front corners of the stage, and the third is in the centre at the rear. The artist has a mass of 70kg - you can ignore the mass of the wires.

Using an appropriate matrix algorithm, calculate the tension in each cable as a function of the artist's position. Assuming the artist is flown in the vertical plane above the front of the stage, with a maximum height of 7 m, plot the tension in one of the front pair of wires as a function of their position. What is the maximum tension and at which position does this occur? Now assume the artist uses the third wire to move forwards and backwards. Given the same maximum height, repeat the exercise to find the maximum tension.

4 Submitting your work

You need to submit the following on Blackboard:

- your report in either pdf or Word format
- your code, as a single text file (change .py to .txt)

A few guidance notes on report and code:

- The report should be concise, maximum 4 pages, including tables and figures as appropriate.
- A very brief introduction to each problem is sufficient, you do not need to restate the problem.
- Give a brief description of the method you used to solve the problem and focus your discussion on the results you have obtained and their interpretation, including possible suggestions for future improvement.

Should you encounter any problems with submitting your work, please contact Dr. Brooke (james.brooke@bristol.ac.uk) or ask a demonstrator.