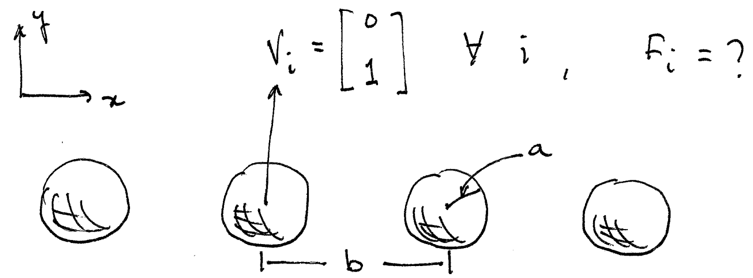
M3/4/5N9 – Project 2
Due 27 Nov 2017

Reminder: High level MATLAB functions, such as 'backslash' are not permitted to be used unless explicitly instructed to do so.

Settling of a line of N particles in a viscous fluid

In this project, you will explore how to exploit the structure and properties of a matrix to select specialized algorithms that solve directly linear systems involving that matrix. While this will be done using an example from fluid mechanics, the ideas developed here apply more widely and demonstrate that if a matrix has structure, it can usually be exploited algorithmically!



As we know from our basic mechanics course, a particle moving through a fluid experiences a drag force that opposes the motion of the particle. For a particle that is sufficiently small and moving sufficiently slowly, the drag force will be linearly related to the particle velocity. When there is more than one particle, one must also account for the fact that the particles will interact through the disturbances they create in the surrounding fluid. Again, when the particles are sufficiently small and moving sufficiently slowly, the forces on the particles are linearly related to particles velocities, but now through a matrix. This results in a linear system of the form

$$\begin{bmatrix} V_1 \\ V_2 \\ \vdots \\ V_N \end{bmatrix} = \begin{bmatrix} M_{11} & M_{12} & \dots & M_{1N} \\ M_{21} & M_{22} & \dots & M_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ M_{N1} & M_{N2} & \dots & M_{NN} \end{bmatrix} \begin{bmatrix} F_1 \\ F_2 \\ \vdots \\ F_N \end{bmatrix} = M \begin{bmatrix} F_1 \\ F_2 \\ \vdots \\ F_N \end{bmatrix}. \tag{1}$$

where (in our case where the motion is planar) V_i and F_i are, respectively, 2×1 vectors that are the velocity of and force on particle i and the 2×2 matrix M_{ij} provides the contribution to the velocity of particle i by the force on particle j. The matrix M depends on the positions of the particles and, in general, its entries will change as the particles move. One approximation for the matrices M_{ij} for spherical particles is known as the *Rotne-Prager-Yamakawa tensor*, or RPY tensor,

$$M_{ij} = \begin{cases} \frac{1}{8\pi\eta r_{ij}} \left[\left(1 + \frac{2a^2}{3r_{ij}^2}\right) I + \left(1 - \frac{2a^2}{r_{ij}^2}\right) \frac{(X_i - X_j)(X_i - X_j)^T}{r_{ij}^2} \right], & r_{ij} > 2a \\ \frac{1}{6\pi\eta a} \left[\left(1 - \frac{9r_{ij}}{32a}\right) I + \frac{3r_{ij}}{32a} \frac{(X_i - X_j)(X_i - X_j)^T}{r_{ij}^2} \right], & r_{ij} \le 2a \end{cases} \tag{2}$$

where η is the viscosity of the fluid, a is the radius of the particles, X_i is the position of particle i, $r_{ij} = |X_i - X_j|$, and I is the 2×2 identity.

Here, we will consider the case where the particles are regularly spaced along a line such that $X_i = [x_i, 0]^T$, where $x_i = (i-1)b$ and b is the spacing between the particles. The file `Msetup.m` builds the matrix M for this configuration assuming that $a = 1$ and $\eta = 1$, but for user specified b and N.

1. Consider the case where $V_i = [0, 1]^T$ for each particle. Using general LU decomposition with partial pivoting, solve the linear system $MF = V$ for different b and N. You may use the function `parpivgelim.m` which performs the LU decomposition, but you must write the other needed codes yourself.

   (a) In your report, provide a table of $\|F\|_2$ for $N = 100, 200$, and 400 with $b = 2, 4$ and 10. Provide a plot of the y-component of the force on each particle as a function of x_i/x_N. Comment on how the force changes with N and b.

   (b) Record the time it takes for your code to run for $b = 4$ and $N = 100, 200, 400, 800$, and 1600. In Matlab, this can be done using `tic` and `toc`. Plot these times vs. N on log-log scale and indicate whether you observe $O(N^3)$ as N becomes large.

   (c) For the $b = 4$ case in Question 1a, compute the residual for each LU decomposition and provide numerical evidence of backward stability by checking the norms of L, U and M. All matrix norms for this question can be computed using `norm` in MATLAB. Discuss your results.

2. We will now look at our matrix M more closely with the aim of revealing structure that can be exploited in computation.

   (a) Examine the structure of the matrix M and describe it in your report. Does the structure vary with b or N?

   (b) Show that

$$PMP^T = \begin{bmatrix} M_1 & 0 \\ 0 & M_2 \end{bmatrix} \tag{3}$$

   where M_1 and M_2 are $N \times N$ matrices and P is a permutation matrix. The matrix 0 is the $N \times N$ matrix of zeros. In your report, describe the permutation matrices for $N = 2$ and $N = 3$, as well as for general N. Describe the matrices M_1 and M_2 in terms of the original matrix, i.e. which force components do they relate to which velocity components, and how the vectors F and V would also need to be modified (*Hint:* $P^T P = I$). Also, describe how you readily obtain M_1 and M_2 without ever having to multiply by P.

3. Consider again the case where $V_i = [0, 1]^T$ for each particle.

   (a) Using Eq. (3) and your other observations from Question 2b, discuss how the solution can be found more rapidly using standard LU. Perform the computation for $b = 4$ and $N = 100, 200, 400, 800$, and 1600, ensuring that you obtain identical values of F to those from Question 1b. Plot the times vs. N on log-log scale and compare you results with those from Question 1b.

   (b) Carefully examine M_2 for properties that may be further exploited for computation and discuss them in your report. Study the algorithms presented in Chapter 4 of *Matrix Computations* by Golub and Van Loan. Are there any algorithms that will allow you to take advantage of these properties to find the solution more efficiently? Discuss your choice and implement it in Matlab. Again, perform the computation for $b = 4$ and $N = 100, 200, 400, 800$, and 1600 and again, making sure that your values of F are correct. Plot the computation times vs. N on log-log scale and compare you results with those from Question 1b and 3a.