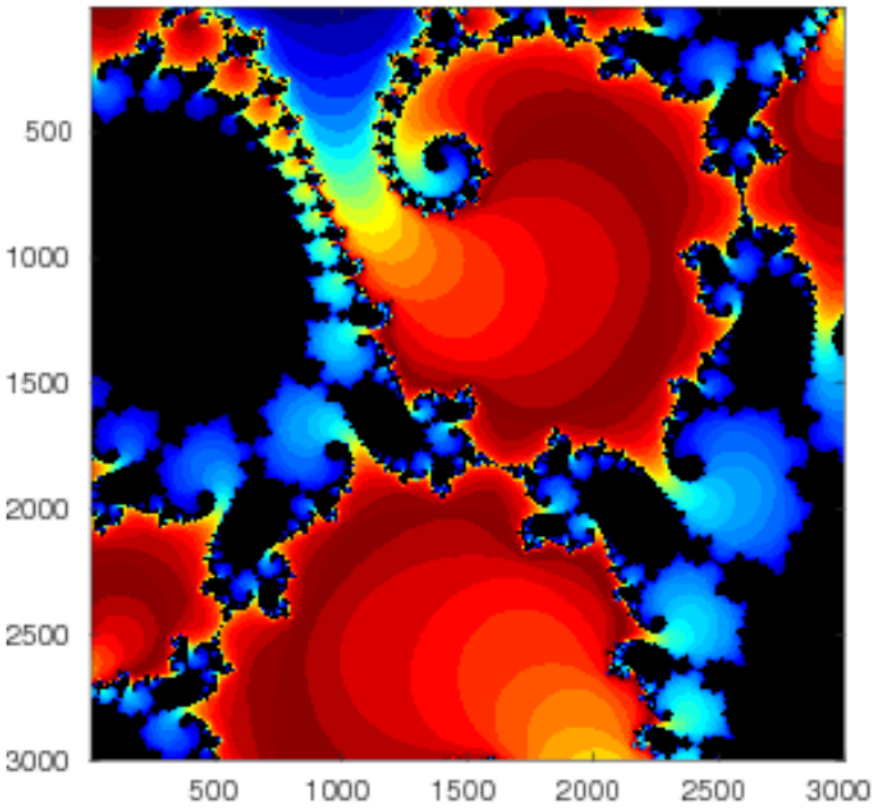
InFoMM CDT

University of Oxford

Scientific Computing

Class exercises

Michaelmas Term 2019



Acknowledgments: Some exercises in this course are taken from the course 'Introduction to Matlab' for the Financial Mathematics MSc and the Financial Modelling and Optimization MSc (University of Edinburgh, 2011) by Stuart Murray.

Session 1: Variables, arrays, functions and plots

In this session you will gain practice in:

- Using the MATLAB work environment;
- Entering, manipulating and indexing arrays in MATLAB ;
- Using basic mathematical functions in MATLAB ;
- Numerical root finding;
- Using MATLAB to plot simple line graphs.

Exercise 1.1 Calculate the following in MATLAB .

1. $0.8 \times \left(\frac{7}{13} + \frac{5}{7}\right)$
2. $\dfrac{2-3i}{1+\frac{1-4i}{3-2i}}$
3. $\gamma = 1/\sqrt{1-\frac{\nu^2}{c^2}}$ where $\nu = 5 \times 10^7$ and $c = 3 \times 10^8$

Exercise 1.2

1. Use the `magic` command to create a 3×3 magic square A.
2. Use `diag` to create a column vector d which is the leading diagonal of A.
3. Create the matrix
$$B = \begin{bmatrix} 0 & d^T \\ d & A \end{bmatrix}.$$
4. Check that its dimensions are correct using `size`.

You will need A, d and B in Exercise 1.4, so keep them in your workspace.

Exercise 1.3

1. Use the colon operator to make a row vector of the integers from -10 to 10.
2. Create a row vector of the real numbers from 7 to 2, descending in steps of $1/3$.
3. Create a row vector of the pure imaginary numbers from $2i$ to $3i$ in steps of $0.05i$.

Exercise 1.4 Using the arrays A, d and B from Exercise 1.2, calculate:

1. $d^T d$ and dd^T
2. $A^T d$ and $B \begin{bmatrix} 1^T \\ A \end{bmatrix}$, where 1 denotes a vector of all ones
3. $C = A + iA^{-1}$
4. C^T, C^* and $C^\dagger$, where * and † represent complex conjugate and pseudoinverse respectively

5. The determinant of A and the eigenvalues of A.

Exercise 1.5

1. Create a 5×5 magic square M.
2. Vectorize M to give m.
3. Reverse the order of the entries of m to give m_2.
4. Reshape m_2 back into a 5×5 matrix M_2.
5. Use `sort` to sort each column of M_2. How would you sort each row?
6. Find the maximum value of each column of M_2 using `max`. What single MATLAB expression gives the maximum value of the entire matrix?
7. Use `size` and `prod` together to calculate the number of elements in M_2. Does this agree with the result of `numel`?

Exercise 1.6 This exercise concerns the evaluation of the integral $\frac{2}{\sqrt{\pi}} \int_0^1 e^{-t^2} \, \mathrm{d}t$.

1. Use the `linspace` command to create a vector t of 10 points from 0 to 1 inclusive.
2. Create a vector `integrand` whose elements are the values of the integrand $2/\sqrt{\pi}e^{-t^2}$, where t runs over each value in `t`. *Hint: use elementwise operations.*
3. Use `sum` to find a rough approximation to the definite integral. Remember to multiply your answer by the size of the intervals in `t`.
4. Make a 5×2 matrix of zeros called `results`. Store the number of points used in `linspace` and the result of your integration as the first and second entries in the first row of this matrix.
5. Repeat the last few steps using successively more points in your vector `t`. Each time add the number of points used and the result of integration to a row of your matrix `results`. The command history might be helpful here. The aim is to end up with something like this:

points	result
10	...
100	...
1000	...
...	...
...	...

6. The integral in question is equal to erf(1), where erf is the error function. Plot a `loglog` plot of the absolute value of the error between your results and the true value erf(1) against number of points. Can you draw any conclusions from the plot?
7. Create an anonymous function called f that computes $\frac{2}{\sqrt{\pi}}e^{-t^2}$ (you will have to use an elementwise operation for the power). Use `quad` to calculate the integral. How does the accuracy of the result compare with that of your own?

Exercise 1.7 Use MATLAB to evaluate the following integrals numerically to six digit accuracy:

1. $\int_0^1 e^{x^3} \mathrm{d}x$
2. $\int_0^{10} \frac{1}{\sqrt{1+x^4}} \mathrm{d}x$
3. $\int_0^5 \sin\left(e^{x/2}\right) \mathrm{d}x.$

Exercise 1.8 The Gamma function $\Gamma(x)$ has a minimum for $1 < x < 2$. Use the `fminsearch` function as discussed in the lecture to find this minimum.

Exercise 1.9

1. The Airy function $\mathrm{Ai}(x)$ has a zero on the real line for $-3 < x < -2$. Use the MATLAB functions `fsolve` and `airy` to solve $\mathrm{Ai}(x) = 0$, and find this zero.
2. Can you find another zero on the negative real line?

Exercise 1.10 The Riemann Zeta Function $\zeta(z)$ is defined for real $z > 1$ as

$$\zeta(z) = \sum_{n=1}^{\infty} n^{-z},$$

and the definition can be extended to an analytic function on the entire complex plane except $z = 1$.

1. Using the inbuilt `zeta` function, plot $\zeta(z)$ on the negative real axis for $-13 < z < -1.5$. *Note: it turns out that $\zeta(z)$ is real-valued on the whole of the negative real axis.*
2. Use `fsolve` to find all the zeros of $\zeta(z)$ in this range.
3. Now plot the real and imaginary part of $\zeta(\frac{1}{2} + ib)$ on the same graph for $-30 < b < 30$. Plot the real part in blue and the imaginary part in red.
4. Use `fsolve` to find all the zeros of $\zeta(z)$ in this range.
5. How do your findings relate to the Riemann Hypothesis? *Wikipedia it!*

Exercise 1.11 Create two square matrices P and Q, both of size 100, using any of the special matrix constructions in MATLAB . Let I denote the (100×100) identity matrix, let $\otimes$ denote the Kronecker product, and let $\mathrm{vec}(X)$ denote the columnwise vectorization of a matrix X. Verify numerically that

$$(I \otimes P)\mathrm{vec}(Q) = \mathrm{vec}(PQ).$$

Exercise 1.12 In this exercise, we explore the function $\operatorname{sinc} x$ which is defined as

$$\operatorname{sinc} x = \begin{cases} \dfrac{\sin x}{x} & x \neq 0, \\ 1 & x = 0. \end{cases}$$

1. Plot $\operatorname{sinc} x$ for $-30 \leq x \leq 30$.
2. It can be shown that each maximum or minimum of the graph of $\operatorname{sinc} x$ corresponds to a point of intersection of the graphs of $\operatorname{sinc} x$ and $\cos x$. Illustrate this result by drawing the graph of $\cos x$ on the same plot.
3. Can you prove the result?

Exercise 1.13

1. Use the MATLAB `quad` function to evaluate the integral

$$\mathcal{I}_p = \int_0^1 \left[\ln\left(\frac{1}{x}\right)\right]^p \mathrm{d}x$$

   for integer values of p.
2. Plot a graph of $\mathcal{I}_p$ against p for real-valued p between 0 and 4.
3. Can you prove a result about $\mathcal{I}_p$?

Exercise 1.14 An integral solution of the ODE

$$x\frac{\mathrm{d}^3 y}{\mathrm{d}x^3} + 2y = 0$$

is given by

$$y(x) = \int_0^\infty e^{\left(-t - \frac{x}{\sqrt{t}}\right)} \mathrm{d}t. \tag{1}$$

1. Try using `quad` along with the function handle for the integrand in (1) and upper limit `Inf` to evaluate $y(2)$. What is the result? Vary the upper limit in $(0, \infty)$ to explore what is happening.
2. Try the above with the function `quadgk` instead.
3. Plot the solution to the equation for $x \geq 0$.

Exercise 1.15 The cycloid $x = t - \sin t$, $y = 1 - \cos t$ is the curve traced out by a point on a wheel as the wheel turns. Plot this curve for $0 \leq t \leq 6\pi$.

Exercise 1.16 Use `ezpolar` to plot (both leaves of) the lemniscate $r^2 = \cos 2\theta$.

Exercise 1.17 The hypotrochoid curve is defined implicitly by the equations

$$x(t) = (a-b)\cos t + c\cos\left[\left(\frac{a}{b} - 1\right)t\right], \quad y(t) = (a-b)\sin t - c\sin\left[\left(\frac{a}{b} - 1\right)t\right].$$

Plot the hypotrochoid for $a = 1$, $b = 7/5$ and $c = 7/13$. Experiment with other choices of a, b and c.

Session 2: Logical operations, `m`-files and functions

In this session you will gain practice in:

- Using array logic;
- Writing `m`-files;
- Writing recursive loops;
- Writing your own MATLAB functions.

Exercise 2.1 Write and test a logical expression that returns

1. Logical `true` if $1 < a < 2$. *Hint: simply using the expression* $1 < a < 2$ *will not work.*
2. Logical `false` if either a or b is strictly positive.

Exercise 2.2 Use `rand` to create a 10×10 matrix A of uniformly distributed random numbers, and use it to perform the following tasks.

1. Type the expression $A > 0.5$. What is the result? Use `nnz` to count how many entries of A are greater than 0.5.
2. Use logical indexing to display all of the entries of A that are greater than 0.5. Use `find` to display their indices.
3. Use array logic to count how many elements of A are either less than 0.5 or greater than 0.9.

Exercise 2.3 Create a vector of the numbers from 1 to 1000.

1. Use `mod` to count how many numbers are divisible by 17.
2. Do the same thing to work out which numbers are divisible by 5 or by 7.

Exercise 2.4

1. Write and test a `for` loop which generates the sequence by means of the iteration $x_{n+1} = x_n + \sin x_n$, trying different starting values. What does the iteration converge to?
2. Modify your loop by adding an `if` statement and a `break` statement, so that the loop stops if $|x_{n+1} - x_n| < 10^{-8}$.
3. Rewrite your code using a `while` loop to achieve the same thing.

Exercise 2.5 The sequence of square triangular numbers (numbers which are both square numbers and triangle numbers) is generated by the recursion

$$G_{n+2} = 34G_{n+1} - G_n + 2,$$

with $G_0 = 0$ and $G_1 = 1$. Generate the first ten square triangular numbers.

Exercise 2.6 The Hadamard matrix H_{2^m} of size 2^m is obtained by repeated Kronecker products as follows:

$$H_{2^m} = H_2 \otimes H_2 \otimes \ldots \otimes H_2, \text{ where } H_2 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix},$$

and where the matrix H_2 occurs m times in the Kronecker product.

1. Write a program that generates H_{2^m} for any positive integer m.
2. Consider the case $m = 5$ and let us write $H = H_{32}$ for convenience. Display H, $H - H^T$, $H^T H$ and the eigenvalues of H. What do these displays tell you about H? Check whether the same properties hold for other choices of m.

Exercise 2.7 Euler's totient function $\phi(n)$ counts how many numbers less than or equal to n are coprime to n, i.e. have no common factors greater than 1. With the help of `gcd`, write a function which takes a single argument n and calculates $\phi(n)$. Given that `gcd` works elementwise, can you think how to calculate $\phi(n)$ using a single MATLAB expression?

Exercise 2.8 The function `lcm` calculates the least common multiple of two numbers. Generalize the function by writing a program that calculates the least common multiple l of all the numbers in a vector v. You will need to loop over all possible values of l, and you may find it useful to use the elementwise property of `mod`.

Exercise 2.9 Given any two real numbers x_0 and y_0, calculate their arithmetic and geometric means

$$x_1 = \frac{x_0 + y_0}{2}; \quad y_1 = \sqrt{x_0 y_0}. \tag{2}$$

The two sequences $\{x_n\}_{n\geq 0}$ and $\{y_n\}_{n\geq 0}$, produced by iterating (2), both converge to the same limit $M(x, y)$, known as the arithmetic-geometric mean.

1. Write a function that calculates the arithmetic-geometric mean of two numbers, and stops when $|x_n - y_n| < \epsilon$ for a given accuracy ϵ.
2. The arithmetic-geometric mean is related to the complete elliptic integral of the first kind. Find out the relation and use an appropriate inbuilt MATLAB function to check your code output. *Take care: the complete elliptic integral of the first kind can be defined in slightly different ways.*
3. Find out how MATLAB calculates this elliptic integral.

Exercise 2.10 A group of n gentlemen enter a restaurant and check their hats. Unfortunately, the waiter is having a bad day due to spending the previous night out with the InFoMM CDT, and he forgets which hats belong to which gentlemen. Optimistically, he returns the hats to the gentlemen at random.

1. Write a function which takes an input n, simulates the waiter's behaviour using the `rand` function, and outputs the number of correct assignments.
2. By performing a large number of trials, estimate the probability that no gentleman gets his own hat.
3. Investigate the limiting value of this probability as $n \to \infty$.

Exercise 2.11 The binary representation of numbers is often useful in computational mathematics. For example, $26 = \underline{1} \times 16 + \underline{1} \times 8 + \underline{0} \times 4 + \underline{1} \times 2 + \underline{0} \times 1$, so 26 has binary form 11010.

1. First the easy way round, write a function `outofbinary` which converts a binary vector into decimal.
2. A little harder, write a function `intobinary` to convert a decimal number into a binary vector.

Note: there are inbuilt functions `bin2dec` *and* `dec2bin` *which work with binary numbers in string form. This exercise is to give you practice in coding it yourself, but feel free to use these inbuilt functions to check your code is working properly.*

Exercise 2.12 Given $\beta > 0$, define a sequence by the recursion

$$x_{n+2} = x_{n+1} \pm \beta x_n,$$

where the sign in the sum is chosen at random for each n independently with equal probabilities for $+$ and $-$.

1. Investigate how the limiting behaviour (as $n \to \infty$) of the sequence $\left\{|x_n|^{1/n}\right\}_{n\geq 0}$ depends upon the parameter β.
2. For what values of β does the sequence $\{x_n\}_{n\geq 0}$ exhibit exponential growth, and for which values of β does it exhibit exponential decay?

Exercise 2.13

1. The multiplication of a vector of length 2^m by a Hadamard matrix of size $2^m \times 2^m$ (see Exercise 2.6) is often referred to as the Walsh-Hadamard transform (WHT). Use the decomposition formula

$$H_{2^m} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} H_{2^{m-1}}(x_1 + x_2) \\ H_{2^{m-1}}(x_1 - x_2) \end{bmatrix}$$

   to write a recursive 'divide-and-conquer' algorithm to compute the WHT.
2. Check your algorithm by comparing the output with the inbuilt `fwht` function.
3. Time the computation of the WHT of a random vector of length 2^{14} using your method above, and using naïve multiplication by a (pre-generated) Hadamard matrix.
4. For what m does it become impossible to store H_{2^m} in memory? Is the memory capacity any different on one of the CDT remote machines?
5. *Harder:* Write an iterative algorithm to compute the WHT. *Hint: you might find reshaping useful.*

Exercise 2.14 Define the Discrete Fourier Transform (DFT) $y = \{y_k\}_{0\leq k<n}$ of a vector $x = \{x_j\}_{0\leq j<n}$ of length n to be

$$y_k = \frac{1}{\sqrt{n}} \sum_{j=0}^{n-1} e^{\frac{-2\pi i j k}{n}} x_j, \quad 0 \leq k < n.$$

If $n = 2^m$ for some integer m, define $e = \{e_k\}_{0 \le k < n/2}$ and $o = \{o_k\}_{0 \le k < n/2}$ to be the DFTs of the even and odd components of x respectively. Then the following recursive formula holds.

$$\begin{aligned} y_k &= \tfrac{1}{\sqrt{2}} \left(e_k + e^{\frac{-2\pi i k}{n}} o_k \right) && 0 \le k < n/2 \\ y_{n/2+k} &= \tfrac{1}{\sqrt{2}} \left(e_k - e^{\frac{-2\pi i k}{n}} o_k \right) && n/2 \le k < n. \end{aligned}$$

1. Use this recursive formula to code up a 'divide-and-conquer' algorithm for the DFT.
2. Check your algorithm by comparing the output with the inbuilt `fft` function.

Session 3: Sparse matrices and numerical solution of ODEs

In this session you will gain practice in:

- Solving systems of linear equations;
- Working with sparse matrices;
- Optimizing the efficiency of code;
- Numerical solution of ODEs.

Exercise 3.1

1. Create a column vector b consisting of $10,000$ ones. Then type the following to create a sparse tridiagonal matrix A:
   `A = spdiags([b -2*b b],[-1 0 1], 10000, 10000);`
2. Use `spy` to visualize the sparsity pattern of the matrix.
3. Use the backslash operator to solve the system $Ax = b$. How long does it take?
4. Now convert A from a sparse matrix to a full matrix, and solve the equation once more. What do you find?
5. The calculation being performed here is $x = A^{-1}b$. Find how long it takes to calculate x using this explicit expression with both sparse and full matrices.

Exercise 3.2

1. Create a sparse matrix J of size 500 of the form
$$J = \begin{bmatrix} & & & 1 \\ & & 1 & \\ & \iddots & & \\ 1 & & & \end{bmatrix}.$$
   Avoid creating a full matrix at any point.
2. Create the sparse matrix
$$M = \begin{bmatrix} I & J \\ J & I \end{bmatrix},$$
   where I is the identity matrix of size 500. Again, work entirely with sparse matrices.
3. Visualize the matrix M using `spy`.

Exercise 3.3

1. Solve numerically the ODE $y' = x - y^2$ with initial condition $y(0) = 0$ over the range $0 \leq x \leq 4$.
2. Plot a graph of the solution.

Exercise 3.4 Consider the 1D Poisson equation

$$\frac{\mathrm{d}^2 y}{\mathrm{d}x^2} + f(x) = 0, \quad 0 \leq x \leq 2\pi,$$

where $f(x) = \frac{2\cos x}{e^x}$, and where we are given the Dirichlet boundary conditions $y(0) = 0$, $y(2\pi) = 0$. Discretization of the partial second derivative operator using central differences and step length $h = \frac{2\pi}{n-1}$ leads to the system of equations $Ay = b$ where

$$A = \begin{bmatrix} 1 & 0 & 0 & \dots & \dots & 0 \\ -1 & 2 & -1 & & & \vdots \\ 0 & -1 & 2 & -1 & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & 0 \\ \vdots & & & -1 & 2 & -1 \\ 0 & \dots & \dots & 0 & 0 & 1 \end{bmatrix}, \qquad y = \begin{bmatrix} 0 \\ h^2 f(h) \\ h^2 f(2h) \\ \vdots \\ h^2 f((n-2)h) \\ 0 \end{bmatrix}.$$

Note: the first and last rows are different due to the boundary conditions.

1. Create A and b for the case $n = 10$ in sparse form. Hence obtain a numerical solution to the differential equation.
2. Obtain a closed form solution for the differential equation.
3. Investigate the approximation error for different n.

Session 4: Parallel computing

In this session you will gain practice in:

- Doing parallel computations in MATLAB;
- Running tasks on remote compute machines.

Exercise 4.1 This exercise concerns the distribution of the extreme value of a certain number of i.i.d. random variables. You will need the code `extreme_value.m` which, given sample size n and number of trials p, returns p instances of the maximum value over n instances of the standard Normal distribution $N(0,1)$.

1. Access one of the departmental compute machines with multiple cores using `ssh`. Run the script `extreme_value.m` for $n = 10^6$ and $p = 10^4$. Parallelize the code using `smpd` and vary the number of cores from 1 to the maximum number of cores available. For each number of cores do five trials and compute the average computing time over these trials. Plot a graph of the speed-up versus the number of cores. Check what type of speedup you get (e.g. linear, logistic, etc.). Check with your cohort fellows who have used a different compute machine what type of speedup they get and discuss what can influence the behaviour(s) you observe.

2. Parallelise the code using `parfor` and vary the number of cores from 1 to the maximum number of cores possible. Compare the speedup using `parfor` and `smpd` for the same number of cores. Check which approach is faster and discuss amongst you why?

3. It is known that the pdf f_n of the maximum of n independent standard Normal random variables satisfies
$$f_n(x) \to g_n(x) := \frac{1}{\beta_n} e^{-(z+e^{-z})}$$
as $n \to \infty$, where $z = (x - \mu_n)/\beta_n$ and where
$$\mu_n = \Phi^{-1}\left(1 - \frac{1}{n}\right) \text{ and } \beta_n = [n\phi(\mu_n)]^{-1},$$
in which $\Phi(x)$ and $\phi(x)$ represent the cumulative distribution function and probability density function for the standard Normal distribution respectively. By superimposing $g_n(x)$ onto normalized histograms of the distribution of your maximum values, explore visually the convergence to this distribution.

Exercise 4.2 This exercise explores the behaviour of the maximum squared singular value[1] of certain large random matrices. You will need the code `max_sing.m` which, given a matrix size n, an aspect ratio $\rho \in (0,1]$ and a number of trials p, returns p instances of the maximum squared singular value of an $n \times \rho n$ matrix whose entries are i.i.d. $N(0, 1/n)$.

1. Take $\rho = 1$. Taking $n = 10^r$ for various r, verify numerically that the expected value of the maximum squared singular value tends to 4 as $n \to \infty$. Make sure to parallelize the code and run it on one of the departmental machines using multiple cores.

[1] If you are not familiar with the singular value decomposition check online what it is about.

2. Do similar exploration for other choices of $\rho \in (0, 1)$. Can you make a conjecture about an expression for the limiting value of the expected maximum squared singular value in terms of ρ?

3. Repeat the previous task for the minimum squared singular value.

4. It is known that, given $\rho \in (0, 1]$, the pdf f_ρ of the maximum squared singular value satisfies

$$f_\rho(x) \to g_\rho(x) := \begin{cases} \frac{\sqrt{(\lambda_+ - x)(x - \lambda_-)}}{2\pi\rho x} & x \in (\lambda_-, \lambda_+) \\ 0 & \text{otherwise,} \end{cases}$$

where λ_+ and λ_- are the expected maximum squared singular value. By superimposing $g_\rho(x)$ onto normalized histograms of the distribution of your maximum values, explore visually the convergence to this distribution.