M.Sc. in High-Performance Computing
5633 - Numerical Methods
Assignment 2

Marina Krstic Marinkovic
(mmarina@maths.tcd.ie)
School of Mathematics, TCD

Rules

To submit, make a single tar-ball with all your code and a pdf of any written part you want to include. Submit this via `msc.tchpc.tcd.ie` or via email to `mmarina@maths.tcd.ie` by the end of **Friday January 20th**. Instead of R, you may use Matlab or Python for the numerical/plotting part. Late submissions without prior arrangement or a valid explanation will result in reduced marks.

Question

1. (10p) Write an R code that does LU factorisation with partial pivoting for an arbitrary $n \times n$ matrix (n<100). Use it to solve the systems of equations $Ax = b$:

   (a) $A = \begin{bmatrix} 9 & 3 & 2 & 0 & 7 \\ 7 & 6 & 9 & 6 & 4 \\ 2 & 7 & 7 & 8 & 2 \\ 0 & 9 & 7 & 2 & 2 \\ 7 & 3 & 6 & 4 & 3 \end{bmatrix}$, $\quad b = [35, 58, 53, 37, 39]^T$;

   (b) $A = H_5$, $\quad b = [5.0, 3.550, 2.81428571428571, 2.34642857142857, 2.01746031746032]^T$;

   (c) $A = A_5$, $\quad b = [-4, -7, -6, -5, 16]^T$,

   where H_n and A_n denote the families of matrices parameterised by their dimension in the following way:

$$H_n = [h_{ij}], \quad h_{ij} = \frac{1}{i + j - 1},$$

$$A_n = [a_{ij}], \quad a_{ij} = \begin{cases} 1, & i = j; \\ 4, & i - j = 1; \\ -4, & i - j = -1; \\ 0, & \text{otherwise.} \end{cases}$$

   In each case, have your code multiply out the L and U factors to check that the routine is working properly.

2. (5p) Use the algorithm for the condition number estimator studied in class to produce a plot of κ^* versus n for the matrix families from the previous question:

   (a) $H_n, \quad 4 \leq n \leq 20$;

   (b) $A_n, \quad 4 \leq n \leq 20$.

3. (10p) Let A be the following 16×16 matrix:

$$A = \begin{bmatrix} -4 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -4 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -4 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -4 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -4 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -4 \end{bmatrix}$$

   Write an R code that applies Jacobi and Gauss-Seidel iterative solver on the system of equations Ax=b, where

$$b = [5, 11, 18, 21, 29, 40, 48, 48, 57, 72, 80, 76, 69, 87, 94, 85]^T.$$

   Try to make the code as efficient as possible, e.g. by only storing the nonzero diagonals of A.

4. (15p) Write an R code that solves the following initial value problems (IVP):

   i) $y' = -y \ln y, \quad y(0) = \frac{1}{2}$;

   ii) $y' + 4y = 1, \quad y(0) = 1$;

   iii) $y' = y, \quad y(0) = 1$,

   using forward Euler's method.

   (a) For each of the IVP above, approximate the solution using a sequence of decreasing grids $n = h^{-1} = 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024$. Find an analytic solution of the problem and compare the accuracy achieved with the Euler's method over the interval [0,1] to the theoretical accuracy by computing the error over the interval:

$$E_n = \max_{k \leq n} |y(t_k) - y_k|,$$

where $y(t_k)$ is the analytic solution of the differential equation evaluated at a grid point $t_k = t_0 + kh,\ k = 0, \ldots, n$. Write out your findings in a text file `forward_{i}.txt` of the format:

$$n = h^{-1} \qquad y_n(t = 1) \qquad E_n,$$

where i=1,2,3 for each of the IVP given above.

(b) Modify your code to include trapezoid rule predictor-corrector method and apply it to the IVP `i)`. Write out your result in a file named `implicit.txt` of the same format as before:

$$n = h^{-1} \qquad y_n(t = 1) \qquad E_n,$$

for $n = h^{-1} = 2, 4, 8, \ldots, 1024$. Comment on the order of accuracy of this method, compared to the ordinary (forward) Euler's method implemented in part `a)`.

(c) Solve the same IVP `i)` using a second and a fourth order Runge-Kutta method and write the output in the file rungekutta.txt of the format:

$$n = h^{-1} \qquad \text{RK2 } y_n(t = 1) \qquad \text{RK2 } E_n \qquad \text{RK4 } y_n(t = 1) \qquad \text{RK4 } E_n$$

Plot the analytic solution of the IVP `i)` on the interval [0,1], along with the 2nd and 4th order RK approximate values for $h = \frac{1}{4}$.