Direct methods for linear systems

Francesco Marchetti

Lab of Fundamentals of Computational Mathematics

First exercise

Exercise

In a script called `direct_solvers.py`, write two functions `lu_solver.py` and `chol_solver.py` that, given a square invertible matrix A and a vector $\boldsymbol{b}$ as input, solve the linear system $A\boldsymbol{x} = \boldsymbol{b}$ by using the LU factorization and the Cholesky method, respectively. In the functions, you can use the built-in routines `lu`, `cholesky` and `solve` provided by the scipy package (check the documentation!).

Second exercise

Exercise

The scipy function `hilbert(n)` provides the Hilbert matrix of order n, which is symmetric and very ill-conditioned. Moreover, `invhilbert(n)` provides its analytical inverse (not computed by numerical approaches). In a new script, varying $n = 2, \dots, 12$, consider the right-hand term $\boldsymbol{b} = (1, \dots, 1)^\intercal$ of proper dimensions and the Hilbert matrix A of order n. Then, compare the solutions of $A\boldsymbol{x} = \boldsymbol{b}$ obtained by using the two methods, imported from the other script (let us call them `x_lu` and `x_chol`), with the true analytical solution `x_true` computed by multiplying the exact inverse of A times b. By using the scipy function `norm`, compute the relative errors in Euclidean norm (e.g. `norm(x_lu-x_true,2)/norm(x_true,2)`) varying n, then producing a semilogarithmic plot.