UNIVERSITY OF SUSSEX
Scientific Computing
Tutor: Dr. Ilian Iliev, Office: Pev 3 4C5

Problem Sheet 2 (Problem 2 will be assessed)
Deadline: 12pm on Monday, October 31st, 2016.
Penalties will be imposed for submissions beyond this date.
Final submission date: Tuesday, November 1st, 2016.
No submissions will be accepted beyond this date.

1. Consider the n simultaneous equations $\mathbf{Ax} = \mathbf{b}$, where

$$A_{ij} = (i+j)^2,\ b_i = \sum_{j=0}^{n-1} A_{ij},\ i = 0, 1, ..., n-1,\ j = 0, 1, ..., n-1$$

Clearly, the solution is $\mathbf{x} = (1, 1, \ldots, 1)^T$. Write a program that solves these equations for any given n using both the provided code `GaussPivot` and the internal `solve` routine. Run your program with $n = 2, 3$, and 4 and comment on the results.

2. **(Assessed) Ill-conditioned matrices:** Consider the system $Hx = b$ where H is the Hilbert matrix of size n, defined by

$$H[i, j] = \frac{1}{i+j+1}$$

and $b_i = \sum_j H_{ij}$ (so that the vector $x = [1, 1, \ldots, 1]$ solves the system). You can use the provided function `hilb(n)`to generate the Hilbert matrix of size $n \times n$. For $n = 2 : 20$, use the in-build NumPy linear solver `solve(A,b)` to find x and plot the maximal error in x as a function of n (use a logarithmic scale in the error). Notice how the error grows quickly, and becomes of the same order as the solution for $n > 12$, at which point your solutions of $Hx = b$ become meaningless, although of course the computer will cheerfully continue to give you results for larger n unless you stop it from doing so.

Try to do the same as above using the provided code for Gauss elimination with partial pivoting (`gaussPivot.py`). Compare the results with the in-build code above. What go you conclude? [30]