# Homework 9

ECON 6204 (8204) - 001

Fall 2017

100 Points

Reading Assignment: Lecture 12

Due: Tuesday, November 28 Student Name: _____

**Instruction**. On the due date, you should email your completed homework with a single zipped folder named "hw9_YourName. The folder should at least include a driver file named "main.py" and an implementation file named "thomas.py for your Python program.

**Problem**. The SOR algorithm,

$$x_i^{(k+1)} = (1-\omega)x_i^{(k)} + \frac{\omega}{a_{ii}}(b_i - \sum_{j=0}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^{N-1} a_{ij}x_j^{(k)}),\ i = 0, 1, \ldots, N-1,$$

can be used to solve a general linear system: $A \cdot x = b$, where $A \equiv [a_{ij}]$ is a $N \times N$ matrix, $x = [x_0, \ldots x_{N-1}]$ is a vector of $N$ unknowns, and $b = [b_0, \ldots, b_{N-1}]$ is a constant vector. You are asked to use Python to design a SOR solver that can be used to solve any tridiagonal systems like the following

$$\begin{bmatrix} \alpha_0 & \beta_0 & & & 0 \\ \gamma_1 & \alpha_1 & \beta_1 & & \\ & \ddots & \ddots & \ddots & \\ & & \gamma_{N-2} & \alpha_{N-2} & \beta_{N-2} \\ 0 & & & \gamma_{N-1} & \alpha_{N-1} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{N-1} \end{bmatrix} = \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_{N-1} \end{bmatrix}$$

You then need to write a main() function together with the SOR solver to solve the following $10 \times 10$ tridiagonal system,

$$\begin{bmatrix} 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \end{bmatrix} = \begin{bmatrix} 10 \\ 10 \\ 10 \\ 10 \\ 10 \\ 10 \\ 10 \\ 10 \\ 10 \\ 10 \end{bmatrix}$$

Your Python program must at least include these two files: main.py and sor.py. The sor.py function should serve as as a general tridiagonal matrix solver. That is, it can be used to solve any tridiagonal system, and should not be limited to the $10 \times 10$ system example.

To enhance the speed of the Thomas algorithm, you can apply just-in-time compiling from *numba*'s @jit.

Note that when your Python program is executed, it should be able to generate console outputs that are well tabulated. It is also required that your Python program should include remarks specifying the homework's purpose, algorithms, and author (i.e., your name) on top of your main.py function. To make your code reader-friendly, you should add remarks when necessary.