Maths 6141 - Numerical Methods

Computer Lab 4

Computer Lab 4

In past lectures we have looked at the solution of linear systems. `numpy` provides a simple command for solving the linear system

$$A\boldsymbol{x} = \boldsymbol{b}.$$

The command is simply

```
x = numpy.linalg.solve(A, b)
```

Note that in order to use this command, $\boldsymbol{b}$ must be a vector, but that python does not distinguish between row and column vectors.
In this lab you should try to implement your own Gaussian elimination algorithm. At first, do this without any pivoting; only later try to improve your algorithm to include pivoting. See the end of this worksheet for a detailed description of the algorithm.
There is a file, `gauss_elimination.py`, which you can download from Blackboard. This contains a function to perform back substitution (`MyBackSubstitution`), which returns the solution once the system is written in the form

$$U\boldsymbol{x} = \boldsymbol{b}$$

with U an upper triangular matrix. You can use this if you wish, or you can implement your own. The script also contains templates for the functions to implement, and a number of tests that your function should pass.

Modules and functions

To use the provided script it is essential to remember the following. When you write functions in a file, you need to `import` the filename within the console in order to call the function. So to test your code in the console you could do

```
> import gauss_elimination
> gauss_elimination.MyGaussianElimination(A, b)
```

However, within the file itself, you can call functions defined within the file directly. So the function `MyBackSubstitution` is defined within the file, and is called *from within the file* using `MyBackSubstitution`.

Tests

Testing your functions against a variety of different input is a good way to build code. It ensures that each function does as expected, and can be built upon. The `pytest` module allows you to run tests automatically, either from the console or from the script. The file provided defines a number of tests that your functions should pass; running the file uses `pytest` to check that they behave as expected, reporting only on those that fail.

Tasks

Write a function

```
MyGaussianElimination(A, b)
```

to solve the linear system. The function should

1. solve the linear system using Gaussian Elimination *without* pivoting. You may use the function `MyBackSubstitution` if you wish.
2. check its own input. A should be a real numeric square matrix. $\boldsymbol{b}$ should be a real numeric vector of the appropriate dimension. A should also be nonsingular, although it need not be well conditioned. If the input violates these conditions, a *meaningful* error message should be returned as early as possible.

Once you have written the function, you should apply the test functions at the end of the given script. To do this, you can – from the console – run

```
> import pytest
> pytest.main("-x gauss_elimination.py")
```

This will run all functions starting with `test_` within the file, and report on whether they pass (i.e., give the expected output). Alternatively, you can run the file from the Editor which should do the same thing.
Secondly, you should write a function

```
MyGaussianEliminationWithPivoting(A, b)
```

to solve the linear system. There are a variety of tests associated with this function also defined within the file.
If you are happy with both of your algorithms, try writing a general function that takes an optional argument `options`, and use an appropriate entry in the options table to select whether or not to use pivoting in the algorithm. There are three possible options:

1. No pivoting
2. Pivoting to avoid dividing by zero
3. Pivoting for accuracy.

You may wish to use a *dictionary* for the options.

Algorithm 1 Gaussian elimination algorithm without pivoting, given $n \times n$ matrix A and known vector $\boldsymbol{b}$, solving for $\boldsymbol{x}$, assuming consistent and well-conditioned input.

$aug \leftarrow [A\ b]$
for $i = 1$ to n **do**
  for $k = i + 1$ to n **do**
    $\epsilon \leftarrow \frac{aug(k,i)}{aug(i,i)}$
    $aug(k,:) \leftarrow aug(k,:) - \epsilon\ aug(i,:)$
  end for
end for
$\boldsymbol{x} =$`MyBackSubstitution`$(aug(:, 1:n), aug(:, n+1))$

Algorithm 2 Gaussian elimination algorithm with pivoting, given $n \times n$ matrix A and known vector $\boldsymbol{b}$, solving for $\boldsymbol{x}$, assuming consistent and well-conditioned input.

$aug \leftarrow [A\ b]$
for $i = 1$ to n **do**
  $ind \leftarrow$ Row number giving $max(abs(aug(i:n, i)))$
  $aug \leftarrow aug$, swapping row i and row ind
  for $k = i + 1$ to n **do**
    $\epsilon \leftarrow \frac{aug(k,i)}{aug(i,i)}$
    $aug(k,:) \leftarrow aug(k,:) - \epsilon\ aug(i,:)$
  end for
end for
$\boldsymbol{x} =$`MyBackSubstitution`$(aug(:, 1:n), aug(:, n+1))$