Algorithm 21.1. Gaussian Elimination with Partial Pivoting

$U = A,\ L = I,\ P = I$
for $k = 1$ **to** $m - 1$
  Select $i \geq k$ to maximize $|u_{ik}|$
  $u_{k,k:m} \leftrightarrow u_{i,k:m}$ (interchange two rows)
  $\ell_{k,1:k-1} \leftrightarrow \ell_{i,1:k-1}$
  $p_{k,:} \leftrightarrow p_{i,:}$
  for $j = k + 1$ **to** m
    $\ell_{jk} = u_{jk}/u_{kk}$
    $u_{j,k:m} = u_{j,k:m} - \ell_{jk} u_{k,k:m}$

Import Dependncies

In [ ]:
```
import helper as hp
import pandas as pd

from IPython.display import display

pd.set_option('display.float_format', '{:.5e}'.format)
```

Comparison Table generation

Block to generate tables for comparing execution Times and Norms between 3 methods:

1. Built function
2. `Scipy.linalg.lu` method

   > decomposes given matrix **A** into 3 matrices **P L U**, where degree(**A**) == degree(**P**, **L**, **U**)

3. `Scipy.linalg.lu_factor` method

   > decomposes given matrix **A** into a matrix **lu** and **piv**, where, **lu**: (M,N) Matrix containing **U** in its upper triangle, and **L** in its lower triangle. The unit diagonal elements of **L** are not stored. **piv** : (N,) ndarray. Pivot indices representing the permutation matrix **P**: row `i` of matrix was interchanged with row **piv**[`i`].

Function for generating two tables

In [ ]:
```
def final_comparison(degree_arr):
    headers1 = ['Time (Built)','Time (Scipy.lu)', 'Time (Scipy.lu_factor)',
                '|| PA-LU (Built) ||','|| PA-LU (Scipy.lu) ||']

    headers2 = ['Time (Built)','Time (Scipy.lu)', 'Time (Scipy.lu_factor)',
                '|| Ax-b (Built) ||','|| Ax-b (Scipy.lu) ||','|| Ax-b (Scipy.lu_factor) ||']

    decomp_stat_table = pd.DataFrame(columns=headers1, index=degree_arr)
    soln_stat_table = pd.DataFrame(columns=headers2, index=degree_arr)

    decomp_stat_table.index.names = ['Degree']
```

```
    soln_stat_table.index.names = ['Degree']

    for degree in degree_arr:
        A, b = hp.get_matrix(degree), hp.get_array(degree)

        my_result = hp.my_system_of_equations_solver(A, b)
        scipy_result = hp.scipy_system_of_equations_solver(A, b)

        if my_result == None or scipy_result == None:
            decomp_stat_table.loc[degree]= ['Singular Matrix'] * len(headers1)
            soln_stat_table.loc[degree]= ['Singular Matrix'] * len(headers2)
        else:
            decomp_stat_table.loc[degree] = [my_result['time_decomp'], scipy_result['time_de
                                             my_result['palu_norm'], scipy_result['palu_norm

            soln_stat_table.loc[degree] = [my_result['time_solve'], scipy_result['time_solve
                                           my_result['axb_norm'], scipy_result['axb_norm_lu'

    return decomp_stat_table, soln_stat_table
```

Execution on matrices of different sizes

In [ ]:
```
degree_array = [10, 100, 500, 1000, 1500]         # size of the square matrices
decomp_stat, soln_stat = final_comparison(degree_array)
```

Comparision table for LU decomposion using different methods

> `Time (Built)` : Time taken by Built method

> `Time (Scipy.lu)` : Time taken by `Scipy.linalg.lu_factor` method

> `Time (Scipy.lu_factor)` : `Scipy.linalg.lu` method

> `|| PA-LU (Built) ||` : Norm of **PA-LU** matrix, where **P**, **L**, **U** calculated using Built method

> `|| PA-LU (Scipy.lu) ||` : Norm of **PA-LU** matrix, where **P**, **L**, **U** calculated using `Scipy.linalg.lu` method

Please Note: Norm of **PA-LU** using `Scipy.linalg.lu_factor` method could not be easily calculated because the Permutation matrix is in LEPACK's permutation array format.

In [ ]:
```
decomp_stat
```

Out[ ]:

Degree	Time (Built)	Time (Scipy.lu)	Time (Scipy.lu_factor)	|| PA-LU (Built) ||	|| PA-LU (Scipy.lu) ||
10	1.14733e-03	5.28351e-03	3.03670e-05	7.40125e-14	8.48059e-14
100	8.81207e-02	1.20167e-03	2.44632e-04	4.34047e-12	2.82582e-12
500	7.34248e-01	3.79214e-02	7.72033e-03	7.08826e-11	6.57570e-11
1000	3.09493e+00	2.54351e-01	6.61428e-02	2.45249e-10	1.47423e-10
1500	8.62392e+00	5.54371e-01	1.39441e-01	5.05743e-10	2.56425e-10

Comparision table for solving `x` in `Ax=b` using different methods

> `Time (Built)` : Time taken by Built method

> `Time (Scipy.lu)` : Time taken by `Scipy.linalg.lu_factor` method

> `Time (Scipy.lu_factor)` : `Scipy.linalg.lu` method

> `|| Ax-b (Built) ||` : Norm of **Ax-b** matrix, where **x** calculated using Built method

> `|| Ax-b (Scipy.lu) ||` : Norm of **Ax-b** matrix, where **x** calculated using `Scipy.linalg.lu` method

> `|| Ax-b (Scipy.lu_factor) ||` : Norm of **Ax-b** matrix, where **x** calculated using `Scipy.linalg.lu_factor` method

In [ ]:
```
soln_stat
```

Out[ ]:

Degree	Time (Built)	Time (Scipy.lu)	Time (Scipy.lu_factor)	|| Ax-b (Built) ||	|| Ax-b (Scipy.lu) ||	|| Ax-b (Scipy.lu_factor) ||
10	1.36449e-04	5.31795e-04	2.00330e-05	4.45154e-14	8.24237e-14	8.24237e-14
100	1.18491e-02	1.95176e-04	3.58980e-05	3.34850e-10	2.22700e-10	2.22700e-10
500	6.13307e-02	1.22498e-03	3.14779e-04	1.76565e-10	9.17914e-11	9.17914e-11
1000	2.41155e-01	4.02436e-03	1.33226e-03	9.50075e-09	5.86093e-09	5.86093e-09
1500	5.40438e-01	8.18021e-03	2.16545e-03	4.31686e-09	2.49956e-09	2.49956e-09