

Gauss Elimination (1A)

Copyright (c) 2016 Young W. Lim.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

Please send corrections (or suggestions) to youngwlim@hotmail.com.

This document was produced by using OpenOffice.

Based on

Lab Manual for Linear Algebra

<http://joshua.smcvt.edu/linearalgebra/lab.pdf>

QQ : for the rational numbers
RR : for real numbers with arbitrary precision
RDF: for real numbers with double-length floats; for
CC : for the complex numbers with arbitrary precision
CDF: for the complex numbers with double floats
ZZ : for the integers.

```
M = matrix ( QQ, [[1,2,3],[4,5,6],[7,8,9]] )
```

```
M[1,2]
```

```
M.nrows()
```

```
M.ncols()
```

```
v = vector ( QQ , [2/3, -1/3, 1/2] )
```

```
M1 = M.augment(v)
```

```
M1 = M.augment(v, subdivide=True)
```

```
M1 = M.swap_rows(0, 1)
```

```
M1 = M.rescale_row(0, 1/2)
```

```
M1 = M.add_multiple_of_the_row(2, 0, -2)
```

`M1.echelon_form()`

`M1.rref()`

`M1.pivots()`

```
var ( 'x , y , z' )
```

```
eqns = [ -3/4*z == -1 , 2*y + z == 2 , x + 2*z == 1/2 ]
```

```
solve ( eqns , x , y , z )
```

```
(x , y , z)  
[[ x == (-13/6), y == (1/3) , z == (4/3) ]]
```

```
def check_nonsingular(mat):  
    if not ( mat.is_square()):  
        print " ERROR : mat must be square "  
        return  
    p = mat . pivots ()  
    for col in range ( mat . ncols ( )):  
        if not ( col in p ):  
            print " nonsingular "  
            break
```

```
N = Matrix ( QQ , [[1,2,3], [4,5,6], [7,8,9]] )  
check_nonsingular (N)  
N = Matrix ( QQ , [[1,0,0], [0,1,0], [0,0,1]] )  
check_nonsingular (N)
```


References

[1] <http://joshua.smcvt.edu/linearalgebra/lab.pdf>