Nyquist Assignment Python and MATLAB

Ishaan Kharbanda

UE179039

In [28]:
```
pip install control
```

```
Requirement already satisfied: control in c:\users\dell\anaconda3\lib\site-packages
(0.8.3)Note: you may need to restart the kernel to use updated packages.
Requirement already satisfied: numpy in c:\users\dell\anaconda3\lib\site-packages (fro
m control) (1.18.5)
Requirement already satisfied: matplotlib in c:\users\dell\anaconda3\lib\site-packages
(from control) (3.2.2)
Requirement already satisfied: scipy in c:\users\dell\anaconda3\lib\site-packages (fro
m control) (1.5.0)
Requirement already satisfied: python-dateutil>=2.1 in c:\users\dell\anaconda3\lib\sit
e-packages (from matplotlib->control) (2.8.1)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in c:\users\de
ll\anaconda3\lib\site-packages (from matplotlib->control) (2.4.7)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\dell\anaconda3\lib\site-p
ackages (from matplotlib->control) (1.2.0)
Requirement already satisfied: cycler>=0.10 in c:\users\dell\anaconda3\lib\site-packag
es (from matplotlib->control) (0.10.0)
Requirement already satisfied: six>=1.5 in c:\users\dell\anaconda3\lib\site-packages
(from python-dateutil>=2.1->matplotlib->control) (1.15.0)
```

Importing necessary libraries

In [2]:
```python
import control as co
import sympy as sym
from sympy import Poly
import numpy as np
import matplotlib.pyplot as plt
from math import degrees
```

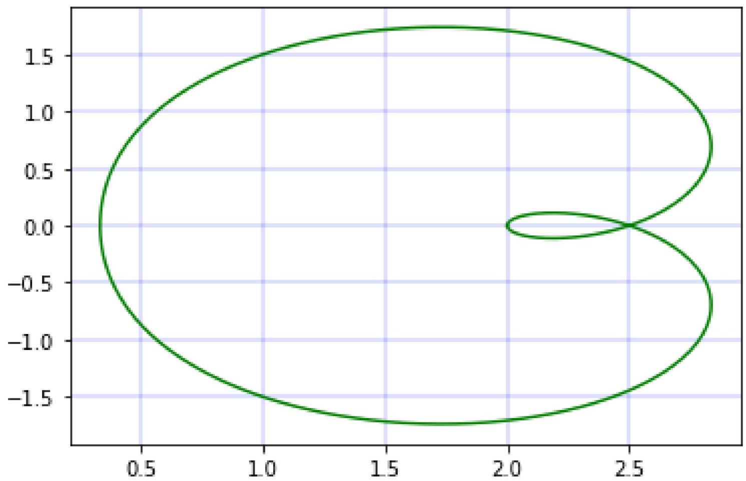
Funtion to create nyquist plot

In [33]:
```python
def draw_nyquist():

    # Asking user for the coefficeints of the numerator polynomial
    N = []
    while True:
        n = input("Enter numerator polynomial or 'done' to stop : ")
        if n == 'done':
            break
        else:
            N.append(int(n))

    # Asking user for the coefficeints of the denominator polynomial
    D = []
    while True:
        d = input("Enter denominator polynomial or 'done' to stop : ")
        if d == 'done':
            break
        else:
            D.append(int(d))


    print('\n******************************************')
    print("Numerator polynomial coefficients : ",N)
    print("\nDenominator polynomial coefficients : ",D)


    # Creating transfer function based on the input
    omega = sym.symbols('w')
    T_F = co.tf(N,D)

    print('******************************************')
    print("\nTransfer Function is : \n",T_F)


    # Replacing s in Transfer function with j*w
    p = Poly(N,omega)/Poly(D,omega)
    p = p.subs(omega,1j*omega)


    omega_range = np.arange(-250,250,0.01)
    Real = []
    Imaginary = []

    for k in omega_range:
        a = complex(p.subs(omega,k))

        R = a.real
        I = a.imag

        Real.append(R)
        Imaginary.append(I)


    print("**** Nyquist plot for the above transfer function ****")
    plt.plot(Real,Imaginary,'g')
    plt.grid(color='b', linestyle='-', linewidth=0.25)
```

Testing the function

In [34]:
```
draw_nyquist()
```

```
Enter numerator polynomial or 'done' to stop : 2
Enter numerator polynomial or 'done' to stop : 5
Enter numerator polynomial or 'done' to stop : 1
Enter numerator polynomial or 'done' to stop : done
Enter denominator polynomial or 'done' to stop : 1
Enter denominator polynomial or 'done' to stop : 2
Enter denominator polynomial or 'done' to stop : 3
Enter denominator polynomial or 'done' to stop : done

******************************************
Numerator polynomial coefficients :  [2, 5, 1]

Denominator polynomial coefficients :  [1, 2, 3]
*****************************************

Transfer Function is :

2 s^2 + 5 s + 1
---------------
 s^2 + 2 s + 3

**** Nyquist plot for the above transfer function ****
```



Matlab M file

```
clc

n = input('Enter numerator polynomial coefficients : ');
d = input('Enter denominator polynomial coefficients : ');

sys_tf = tf(n,d)

syms s w
num_poly = poly2sym(n,s);
den_poly = poly2sym(d,s);

%fprintf('Transfer function is : %X',sys_tf)

system = num_poly/den_poly;

system_omega = subs(system,s,1i*w)

omega = -250:0.01:250;



output_complex_numbers = subs(system_omega,w,[omega]);

R = real(output_complex_numbers);
I = imag(output_complex_numbers);

plot(R,I)
grid on
```

Matlab Command Window

```
Command Window
Enter numerator polynomial coefficients : [2 5 1]
Enter denominator polynomial coefficients : [1 2 3]

sys_tf =

  2 s^2 + 5 s + 1
  ---------------
   s^2 + 2 s + 3

Continuous-time transfer function.


system_omega =

(- 2*w^2 + w*5i + 1)/(- w^2 + w*2i + 3)

>>
```

Matlab Nyquist Graph