

Joe Stanley

ECE522 - EXAM1

```
In [2]: 1 # Import Necessary Libraries
2 import numpy as np
3 import matplotlib.pyplot as plt
4 from scipy.optimize import fsolve
5 import electricpy as ep
6 from electricpy.constants import *
7
8 # Set Boolean Control for Report Style
9 debug = False
```

Problem III:

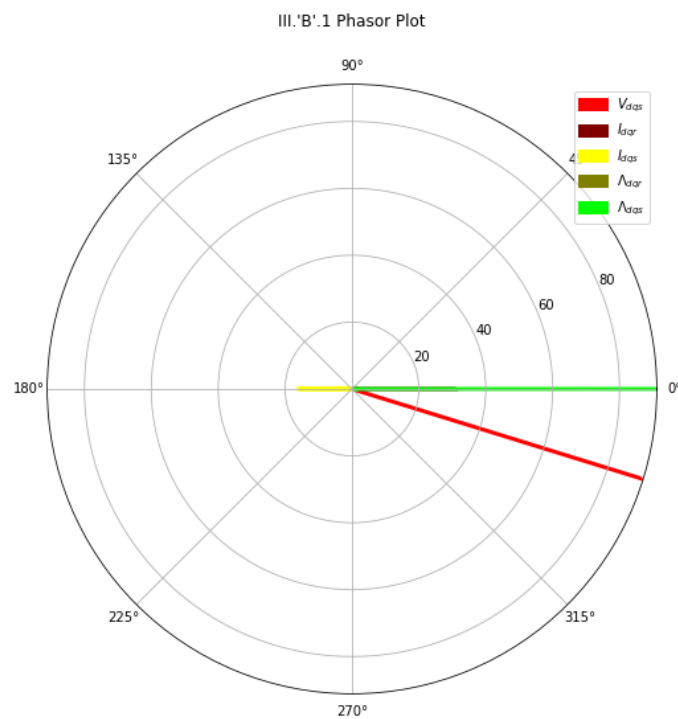
Repeat Parts B and C of Problem I for the situation where the parameter L_r/r in the "slip calculator" is in error by -25%.

Comment on the effect on steady state performance of such "detuning" of the controller.

Just like Problem II, since we know: $s = \frac{\omega_{es} - \omega_r}{\omega_{es}}$, we can manipulate the equation into the form: $(\omega_{es} - \omega_r) = s \cdot \omega_{es}$. In this form, we can substitute it into our equations to solve.

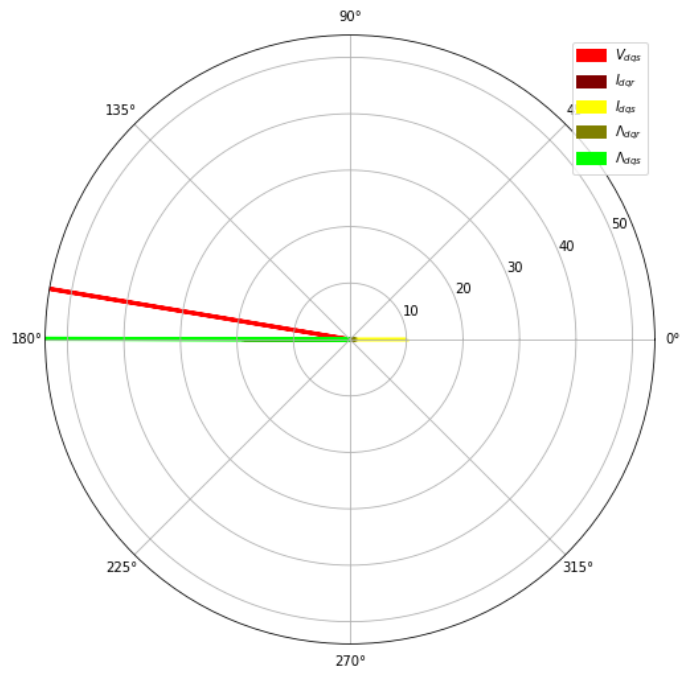
Part 'B' (since we're only repeating parts B and C of problem 1):

Sub-Part 1:



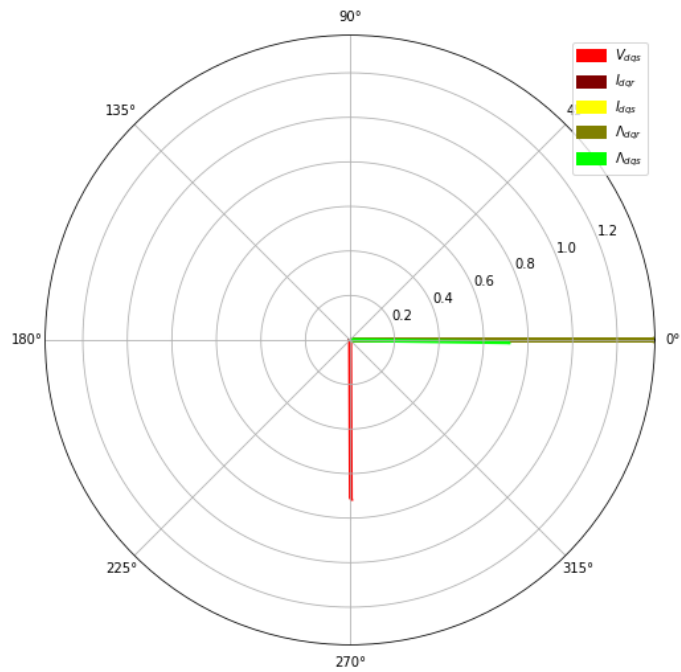
Sub-Part 2:

III.'B'.3 Phasor Plot



Sub-Part 3:

III.'B'.2 Phasor Plot



In [4]:

```
1  # Define Provided Machine Parameters
2  rs = 0.03 #pu
3  Lls = 0.1 #pu
4  Lm = 2.0 #pu
5  LLr = 0.1 #pu
6  rr = 0.03 #pu
7
8  # Define Rated Criteria
9  VdqsMag = 1
10 wes = 1
11 Tem = 0
12
13 # Assumptions
14 p = 4
15 at = 2
16
17 # Calculate Additional Inductance Terms
18 Ls = Lls + Lm
19 Lr = LLr + Lm
20 Lmr = 2/3 * (Lr - LLr)
21 Lsr = Lmr * at
22
23 #####
24 # Calculated from Problem 1
25 lamdr_rated = 1.36883
26 w_rated = 0.99468
27 #####
28
29 # Define Equations Function as Solver
30 def equations_B(val,wr,LAMdr):
31     Idr,Iqr,Ids,Iqs,LAMqr,LAMds,LAMqs,Vds,Vqs = val
32     #####
33     s = (wes - wr) / wes # Find S
34     s *= 0.75 # Apply -25% Error
35     #####
36     A = (rs*Ids - wes*LAMqs) - Vds
37     B = (rs*Iqs - wes*LAMds) - Vqs
38     C = rr*Idr - (wes*s)*LAMqr
39     D = rr*Iqr + (wes*s)*LAMdr
40     E = (Ls*Ids + 3/2*Lsr*Idr) - LAMds
41     F = (Ls*Iqs + 3/2*Lsr*Iqr) - LAMqs
42     G = (3/2*Lsr*Ids+Lr*Idr) - LAMdr
43     H = (3/2*Lsr*Iqs+Lr*Iqr) - LAMqr
44     I = (3*p/4*(3/2*Lsr)/Lr*(LAMdr*Iqs-LAMqr*Ids)) - Tem
45     return(A,B,C,D,E,F,G,H,I)
46
47 # Define Initial Guesses
48 Vds0 = 1
49 Vqs0 = 0
50 Idr0 = -1
51 Iqr0 = -1
52 Ids0 = 1
53 Iqs0 = -1
54 LAMqr0 = 3/2*Lsr*Iqs0 + Lr*Iqr0
55 LAMds0 = Ls*Ids0 + 3/2*Lsr*Idr0
56 LAMqs0 = Ls*Iqs0 + 3/2*Lsr*Iqr0
57
58 # Define Plotting Labels
59 labels = [
60     "$V_{dqs}$",
61     "$I_{dqr}$",
62     "$I_{dqs}$",
63     "$\\Lambda_{dqr}$",
64     "$\\Lambda_{dqs}$",
65 ]
66
67 #####
68 # "B".1)
69 wr = 0.0
70 LAMdr = lamdr_rated
71 b1 = lambda x: equations_B(x,wr,LAMdr)
72
73 # Use Iterative Solver to Find Results
74 Idr,Iqr,Ids,Iqs,LAMqr,LAMds,LAMqs,Vds,Vqs = fsolve(b1,(
75     Idr0,Iqr0,Ids0,Iqs0,LAMqr0,LAMds0,LAMqs0,Vds0,Vqs0))
76
77 # Generate Phasor Plot
78 clist = ep.compose([[Vds,Vqs],[Idr,Iqr],[Ids,Iqs],[LAMdr,LAMqr],[LAMds,LAMqs]])
79 clist *= ep.phs(-np.angle(clist[3],deg=True))
80 ep.phasorplot(clist,"III. 'B'.1 Phasor Plot",labels,filename="III-B-1",size=8,linewidth=3,plot=debug)
81
82 #####
83 # "B".2)
84 wr = w_rated
85 LAMdr = lamdr_rated
86 b2 = lambda x: equations_B(x,wr,LAMdr)
```

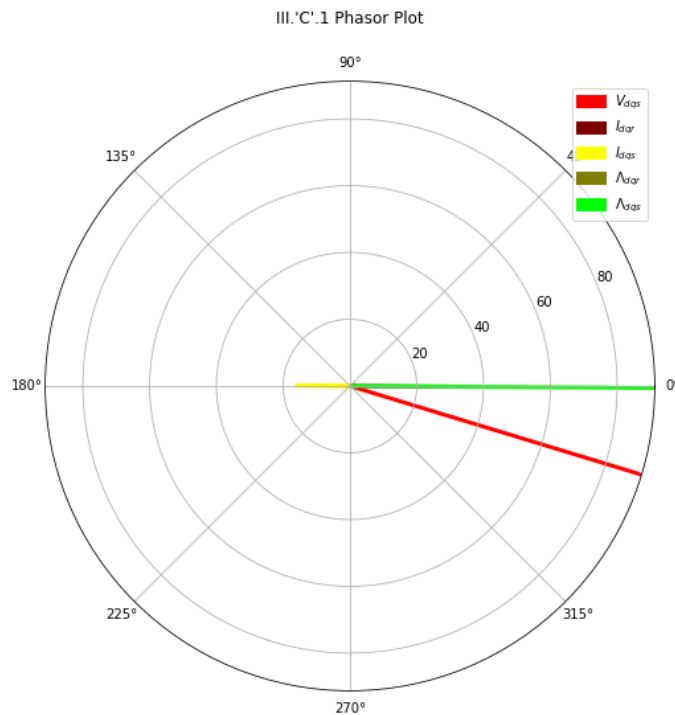
```

87
88 # Use Iterative Solver to Find Results
89 Idr,Iqr,Ids,Iqs,LAMqr,LAMds,LAMqs,Vds,Vqs = fsolve(b2,(
90     Idr0,Iqr0,Ids0,Iqs0,LAMqr0,LAMds0,LAMqs0,Vds0,Vqs0))
91
92 # Generate Phasor Plot
93 clist = ep.compose([[Vds,Vqs],[Idr,Iqr],[Ids,Iqs],[LAMdr,LAMqr],[LAMds,LAMqs]])
94 clist *= ep.phs(-np.angle(clist[3],deg=True))
95 ep.phasorplot(clist,"III.'B'.2 Phasor Plot",labels,filename="III-B-2",size=8,linewidth=3,plot=debug)
96
97 #####
98 # "B".3)
99 wr = 2*w Rated
100 LAMdr = lamdr Rated/2
101 b3 = lambda x: equations_B(x,wr,LAMdr)
102
103 # Use Iterative Solver to Find Results
104 Idr,Iqr,Ids,Iqs,LAMqr,LAMds,LAMqs,Vds,Vqs = fsolve(b3,(
105     Idr0,Iqr0,Ids0,Iqs0,LAMqr0,LAMds0,LAMqs0,Vds0,Vqs0))
106
107 # Generate Phasor Plot
108 clist = ep.compose([[Vds,Vqs],[Idr,Iqr],[Ids,Iqs],[LAMdr,LAMqr],[LAMds,LAMqs]])
109 clist *= ep.phs(-np.angle(clist[3],deg=True))
110 ep.phasorplot(clist,"III.'B'.3 Phasor Plot",labels,filename="III-B-3",size=8,linewidth=3,plot=debug)

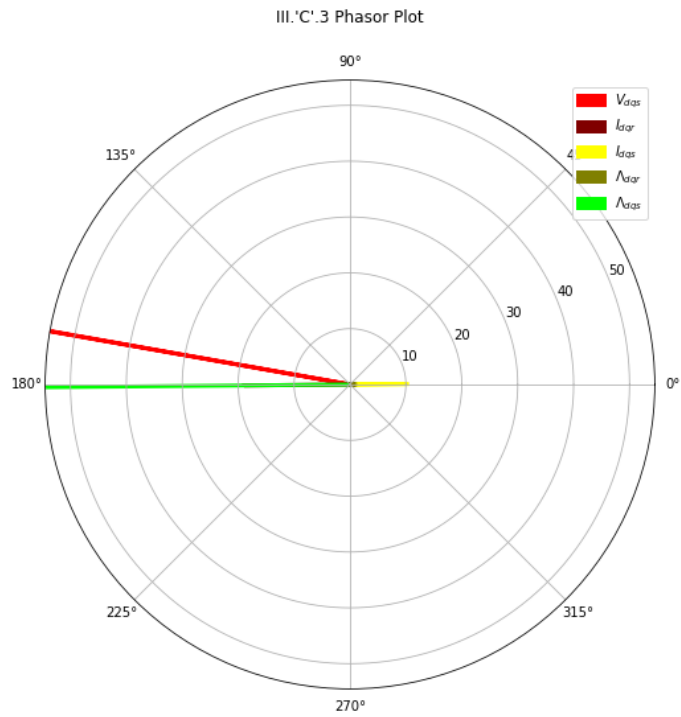
```

Part 'C' (since we're only repeating parts B and C of problem 1):

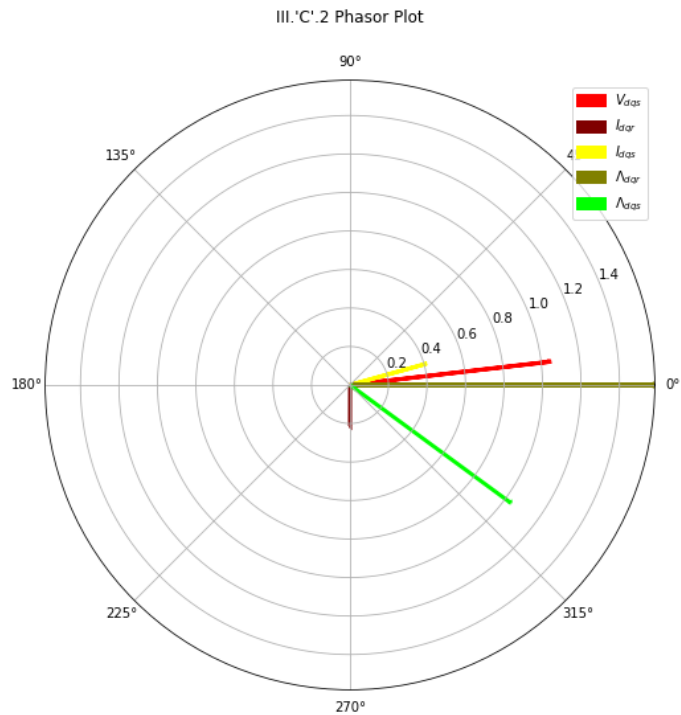
Sub-Part 1:



Sub-Part 2:



Sub-Part 3:



Comments and Analysis:

From comparison, it seems clear that these results are not too dissimilar from those found in the first problem (problem I). Perhaps the only truly notable difference is that magnitude difference between the results. Angle differences and general relations between the vectors appear to be largely the same between the Problem I results and these Problem III results. It is interesting to see that the magnitudes again seem to reflect the error in slip. It seems that perhaps it could be drawn that slip is directly proportional to these terms.

In [5]:

```
1  # Define Equations Function as Solver
2  def equations_C(val,wr,LAMdr,Tem):
3      Idr,Iqr,Ids,Iqs,LAMqr,LAMds,LAMqs,Vds,Vqs = val
4      #####
5      s = (wes - wr) / wes # Find S
6      s *= 0.75 # Apply -25% Error
7      #####
8      A = (rs*Ids - wes*LAMqs) - Vds
9      B = (rs*Iqs - wes*LAMds) - Vqs
10     C = rr*Idr - (wes*s)*LAMqr
11     D = rr*Iqr + (wes*s)*LAMdr
12     E = (Ls*Ids + 3/2*Lsr*Idr) - LAMds
13     F = (Ls*Iqs + 3/2*Lsr*Iqr) - LAMqs
14     G = (3/2*Lsr*Ids+Lr*Idr) - LAMdr
15     H = (3/2*Lsr*Iqs+Lr*Iqr) - LAMqr
16     I = (3*p/4*(3/2*Lsr)/Lr*(LAMdr*Iqs-LAMqr*Ids)) - Tem
17     return(A,B,C,D,E,F,G,H,I)
18
19 # Define Initial Guesses
20 Vds0 = 1
21 Vqs0 = 0
22 Idr0 = -1
23 Iqr0 = -1
24 Ids0 = 1
25 Iqs0 = -1
26 LAMqr0 = 3/2*Lsr*Iqs0 + Lr*Iqr0
27 LAMds0 = Ls*Ids0 + 3/2*Lsr*Idr0
28 LAMqs0 = Ls*Iqs0 + 3/2*Lsr*Iqr0
29
30 #####
31 # "C".1)
32 Tem = 1.0
33 wr = 0.0
34 LAMdr = lamdr_rated
35 c = lambda x: equations_C(x,wr,LAMdr,Tem)
36
37 # Use Iterative Solver to Find Results
38 Idr,Iqr,Ids,Iqs,LAMqr,LAMds,LAMqs,Vds,Vqs = fsolve(c,(
39     Idr0,Iqr0,Ids0,Iqs0,LAMqr0,LAMds0,LAMqs0,Vds0,Vqs0))
40
41 # Generate Phasor Plot
42 clist = ep.compose([[Vds,Vqs],[Idr,Iqr],[Ids,Iqs],[LAMdr,LAMqr],[LAMds,LAMqs]])
43 clist *= ep.phs(-np.angle(clist[3],deg=True))
44 ep.phasorplot(clist,"III.'C'.1 Phasor Plot",labels,filename="III-C-1",size=8,linewidth=3,plot=debug)
45
46 #####
47 # "C".2)
48 Tem = 1.0
49 wr = w_rated
50 LAMdr = lamdr_rated
51 c = lambda x: equations_C(x,wr,LAMdr,Tem)
52
53 # Use Iterative Solver to Find Results
54 Idr,Iqr,Ids,Iqs,LAMqr,LAMds,LAMqs,Vds,Vqs = fsolve(c,(
55     Idr0,Iqr0,Ids0,Iqs0,LAMqr0,LAMds0,LAMqs0,Vds0,Vqs0))
56
57 # Generate Phasor Plot
58 clist = ep.compose([[Vds,Vqs],[Idr,Iqr],[Ids,Iqs],[LAMdr,LAMqr],[LAMds,LAMqs]])
59 clist *= ep.phs(-np.angle(clist[3],deg=True))
60 ep.phasorplot(clist,"III.'C'.2 Phasor Plot",labels,filename="III-C-2",size=8,linewidth=3,plot=debug)
61
62 #####
63 # "C".3)
64 Tem = 0.5
65 wr = 2*w_rated
66 LAMdr = lamdr_rated/2
67 c = lambda x: equations_C(x,wr,LAMdr,Tem)
68
69 # Use Iterative Solver to Find Results
70 Idr,Iqr,Ids,Iqs,LAMqr,LAMds,LAMqs,Vds,Vqs = fsolve(c,(
71     Idr0,Iqr0,Ids0,Iqs0,LAMqr0,LAMds0,LAMqs0,Vds0,Vqs0))
72
73 # Generate Phasor Plot
74 clist = ep.compose([[Vds,Vqs],[Idr,Iqr],[Ids,Iqs],[LAMdr,LAMqr],[LAMds,LAMqs]])
75 clist *= ep.phs(-np.angle(clist[3],deg=True))
76 ep.phasorplot(clist,"III.'C'.3 Phasor Plot",labels,filename="III-C-3",size=8,linewidth=3,plot=debug)
```

In []:

1