Grating Efficiency Measurements

Blaise C. Kuo Tiong - Macquarie University - 24/10/2018

Setup

- Richardson Grating Labs R6 echelle grating, 80.7 blaze angle. Silver coating. 150 mm x 60 mm x 25 mm.
- 2mW 543 nm HeNe laser from Newport
- 0.8 mW 633 HeNe laser from Thorlabs
- both lasers attached to Fiber port collimators
- Corning 8.2 μm with 0.14 NA single mode fiber
- fiber attached to a Thorlabs 90^o Off-axis Parabola that collimates a 4 mm beam
- the beam is shone on the channel 1 S120C photodiode set to 10 1mW scale and sampling at 10 times per second
- power levels are measured using a Thorlabs Dual Channel Power and Energy Meter (PM320E) with two S120C photodiodes attached
- for method 2, a Thorlabs 50:50 BS03 non-polarizing cube beam splitter is placed in the path
- two photodiodes are used to measure the reflectance and transmission paths

Method 1

No beamsplitter used, laser power level measured after off angle measurements of difracted orders.

Measure the power level of the two laser. Check the run quality: gratingefficiency%2Frunquality.ipynb

Calcualte mean of power level.

Diffract beam from OAP with grating onto a position as close to the OAP as possible. Measure the power level at this position and then at increasing off plane angles.

Plot grating efficiency as a percentage of the measured power level. (solid lines in plot)

Method 2

Use a beamsplitter to get fraction of power level from off angle measurements of diffracted orders over laser power level. Apply transmission/reflectance ratio afterwards.

Measure the power levels of the lasers. Then put a non-polarizing cube beam splitter in the path. Measure the power levels in the reflectance and transmission paths. Determine the measured ratio for the two lasers.

meassure beam diffrated from grating at off plane angles like in method 1.

Determine efficiency as a ratio been the two beam splitter paths and correct for the measured ratio between the beam splitters paths.

Plot the results. (dashed lines in plot)

In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import glob
import re

#win
#datastore_path="c:/cloudstor/datastore/grating-efficiency/"
#local
datastore_path="measurements/"
```

Function for calculating averages from data files

In [2]:

```
def PD_mean(str_run,PD):
    disp_list =[]

    for filestring in (glob.glob(datastore_path + str_run + "/*0.txt")):
        with open(filestring, 'r') as f:
            file2list = pd.read_csv(f,header=None,delimiter='\t')
        if PD == 1:
            for item in file2list[2]:
                disp_list.append(item)
        elif PD == 2:
            for item in file2list[4]:
                disp_list.append(item)

    mean = np.array(disp_list).mean()
    std = np.array(disp_list).std()

    print("mean: " + str(mean * 1000000) + ' \u03BCW')
    print("std: " + str(std * 1000000) + ' \u03BCW')

    return(np.array(disp_list).mean())
```

Laser Power Levels and Averages

543 nm laser

In [3]:

```
mean_run543 ="543nm_nosplit_lasercal_run22"
lasermean543=PD_mean(mean_run543,1)
```

```
mean: 443.0509672197491 μW
std: 13.408263233598111 μW
```

Power levels on the 543 nm laser varies significantly. Observed between 373-443 μW between runs but stays fairly consistent between runs. See gratingefficiency%2Frunquality.ipynb for other runs. The laser power level should therefore be measured immediately before or after a run.

633 nm laser

In [4]:

```
mean_run633 ="633nm_nosplitter_lasercalpost_run21"
lasermean633=PD_mean(mean_run633,1)
```

```
mean: 113.5376156554491 µW
std: 3.5734473360119896 µW
```

Power levels for the 633 nm laser stays consistent between runs at this level.

Beam splitter ratio

543 nm

In [5]:

```
#measure power level again before beam splitter measurement runs
run="543nm_lasercal_run29"
fullbeam=PD_mean(run,1)

run ="543nm_beamsplittercal_run30"

#transmission path
BS1_543=(PD_mean(run,1))/fullbeam
#reflectance path
BS2_543=(PD_mean(run,2))/fullbeam

print("Beamsplitter grating (transmission) position: " + str(BS1_543*100) + "%")
print("Beamsplitter reference (reflectance) position: " + str(BS2_543*100) + "%")
```

```
mean: 373.0033330378513 µW
std: 5.053126275774238 µW
mean: 175.07325089833017 µW
std: 3.1938911352199173 µW
mean: 171.9713168463327 µW
std: 3.119337404238464 µW
Beamsplitter grating (transmission) position: 46.93610898124715%
Beamsplitter reference (reflectance) position: 46.104498704005294%
```

633 nm

In [6]:

```
# measure power levels again before beam splitter measurement runs
run="633nm_lasercal_run31"
fullbeam=PD_mean(run,1)

run ="633nm_beamsplittercal_run32"

# transmission path
BS1_633=(PD_mean(run,1))/fullbeam
# reflectance path
BS2_633=(PD_mean(run,2))/fullbeam

print("Beamsplitter grating (transmission) position: " + str(BS1_633*100) + "%")
print("Beamsplitter reference (reflectance) position: " + str(BS2_633*100) + "%")
```

```
mean: 113.75754265595377 µW
std: 2.9069301770381197 µW
mean: 47.986337774957704 µW
std: 1.434343181908866 µW
mean: 58.09587880710661 µW
std: 1.7417936272362493 µW
Beamsplitter grating (transmission) position: 42.18299433566942%
Beamsplitter reference (reflectance) position: 51.069913652065004%
```

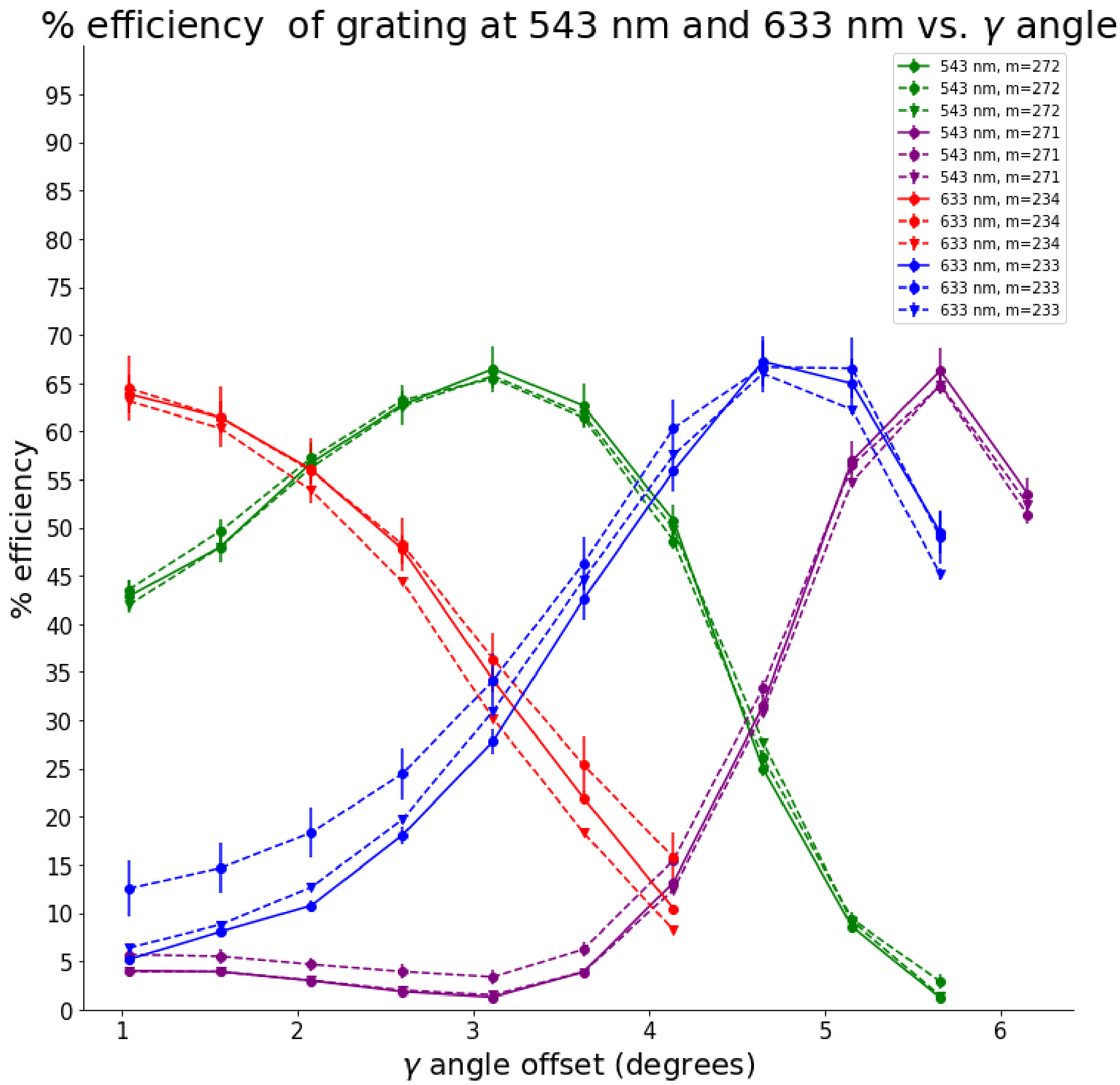
Grating Efficiency Plot

In [20]:

```
# Prepare plot area
plt.figure(figsize=(12, 12))

#Prepare axes
ax = plt.subplot(111)
ax.spines["top"].set_visible(False)
ax.spines["bottom"].set_visible(True)
ax.spines["right"].set_visible(False)
ax.spines["left"].set_visible(True)

ax.get_xaxis().tick_bottom()
ax.get_yaxis().tick_left()

# set y range, percentage
plt.ylim(0, 100)
#plt.xlim(2, 1278)

# put labels for every 5%
plt.yticks(np.arange(0,100,5),fontsize=15)
plt.xticks(fontsize=15)

#axis labels
plt.xlabel(r'$\gamma$ angle offset (degrees)',fontsize=20)
plt.ylabel("% efficiency",fontsize=20)

# title
plt.title(r'% efficiency  of grating at 543 nm and 633 nm vs. $\gamma$ angle',fontsize=
25)

plt.tick_params(axis="both", which="both", bottom=True, top=False,
                labelbottom=True, left=True, right=False, labelleft=True)


# function for plotting runs
def plot_run(run_glob,lasermean,series_marker,series_color,labelstring):

    i=0
    plot_list = []

    for file in (glob.glob(datastore_path + run_glob)):
        with open(file, 'r') as f:
            disp_list = pd.read_csv(f,header=None,delimiter='\t')

        # calculate off plane angle in degrees
        # beam hits the grating about 550 mm from the OAP
        # measurements of spots spaced about 10 mm apart
        offplaneangle = (np.arctan((10*i+20)/550))*(90/np.pi)

        # std dev calculations needs work
        if lasermean == 0:
            if re.match('543',run_glob):
                efficiency = ( (disp_list[2]/disp_list[4]).mean() / (BS1_543/BS2_543) )
*100
                stddev = ( (disp_list[2]/disp_list[4]).std() / (BS1_543/BS2_543) )*100
            elif re.match('633',run_glob):
                efficiency = ( (disp_list[2]/disp_list[4]).mean() / (BS1_633/BS2_633) )
*100
                stddev = ( (disp_list[2]/disp_list[4]).std() / (BS1_633/BS2_633) )*100
```

```
        else:
            efficiency = (disp_list[2].mean()/lasermean)*100
            stddev = (disp_list[2].std()/lasermean)*100

        plot_list.append([offplaneangle,efficiency,stddev])

        i=i+1

    num_plot=np.array(plot_list)
    plotted = plt.errorbar(num_plot[:,0],num_plot[:,1],num_plot[:,2],fmt=series_marker,
color=series_color,label = labelstring)

    print(labelstring + " efficiency: " + str(round(np.max(num_plot[:,1]),1)) + "%")

    return plotted


#plot runs
# line is using method 1 ( no beam splitter), dash is using method 2 (using beam splitt
er ratios)
green_line = plot_run("543nm_nosplitter_run23/pos*_0.txt",lasermean543,'-o',"green",'54
3 nm, m=272')

green_dash = plot_run("543nm_run8/pos*_0.txt",0,'--o',"green",'543 nm, m=272')
green_dash = plot_run("543nm_run2/pos*_0.txt",0,'--v',"green",'543 nm, m=272')

purple_line = plot_run("543nm_no1_nosplitter_run24/pos*_0.txt",lasermean543,'-o',"purpl
e",'543 nm, m=271')

purple_dash = plot_run("543nm_on1_run8/pos*_0.txt",0,'--o',"purple",'543 nm, m=271')
purple_dash = plot_run("543nm_on1_run2/pos*_0.txt",0,'--v',"purple",'543 nm, m=271')

red_line = plot_run("633nm_nosplitter_run19/pos*_0.txt",lasermean633,'-o',"red",'633 n
m, m=234')

red_dash = plot_run("633nm_run9/pos*_0.txt",0,'--o',"red",'633 nm, m=234')
red_dash = plot_run("633nm_run2/pos*_0.txt",0,'--v',"red",'633 nm, m=234')

blue_line = plot_run("633nm_no1_nosplitter_run20/pos*_0.txt",lasermean633,'-o',"blue",
'633 nm, m=233')

blue_dash = plot_run("633nm_on1_run9/pos*_0.txt",0,'--o',"blue",'633 nm, m=233')
blue_dash = plot_run("633nm_on1_run2/pos*_0.txt",0,'--v',"blue",'633 nm, m=233')

plt.legend(loc='best')

plt.savefig("GratingEfficiency.png", bbox_inches="tight")
```

```
543 nm, m=272 efficiency: 66.5%
543 nm, m=272 efficiency: 65.5%
543 nm, m=272 efficiency: 65.8%
543 nm, m=271 efficiency: 66.3%
543 nm, m=271 efficiency: 64.7%
543 nm, m=271 efficiency: 64.8%
633 nm, m=234 efficiency: 63.9%
633 nm, m=234 efficiency: 64.5%
633 nm, m=234 efficiency: 63.1%
633 nm, m=233 efficiency: 67.3%
633 nm, m=233 efficiency: 66.7%
633 nm, m=233 efficiency: 66.0%
```


% efficiency of grating at 543 nm and 633 nm vs. γ angle
543 nm, m=272
543 nm, m=272
543 nm, m=272
543 nm, m=271
543 nm, m=271
543 nm, m=271
633 nm, m=234
633 nm, m=234
633 nm, m=234
633 nm, m=233
633 nm, m=233
633 nm, m=233
% efficiency
γ angle offset (degrees)
0
5
10
15
20
25
30
35
40
45
50
55
60
65
70
75
80
85
90
95
1
2
3
4
5
6