photoelecv2

January 21, 2026

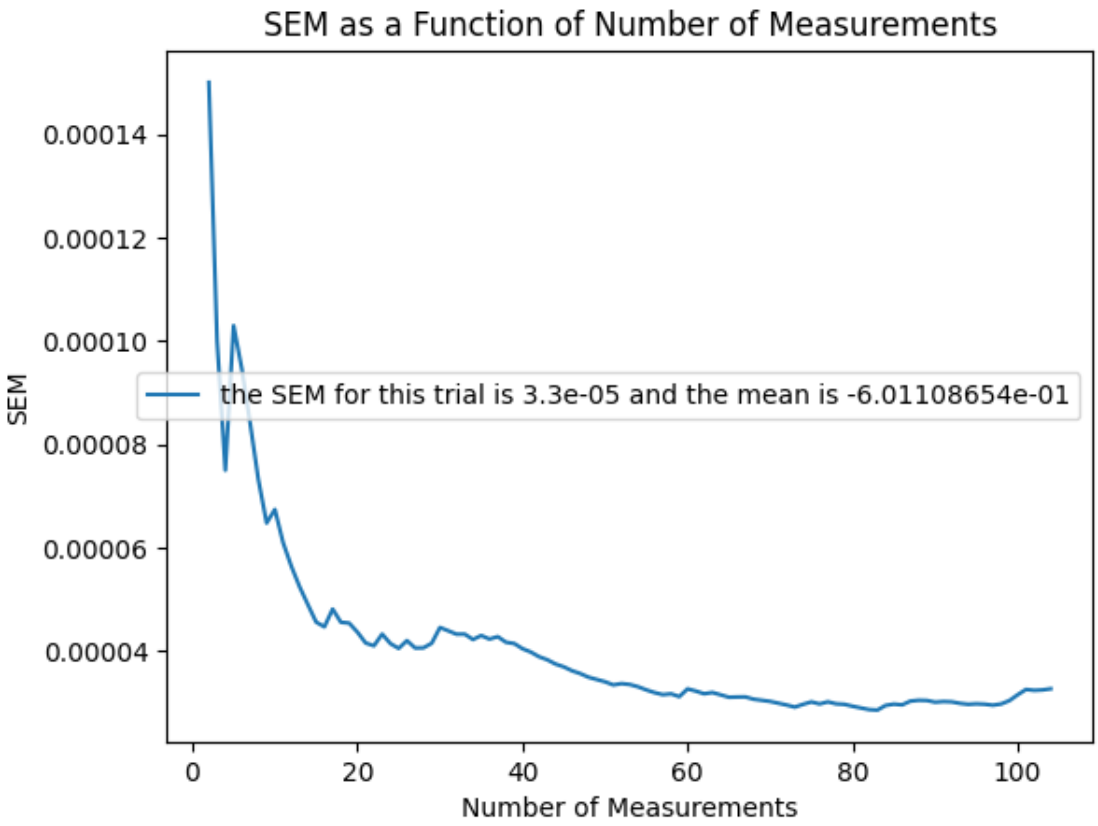
[16]:
```
import os
import numpy as np
import matplotlib.pyplot as plt
import labtools as lbts #goated file
import importlib
importlib.reload(lbts)
```

[16]:
```
<module 'labtools' from 'c:\\Adult
life\\McGill\\2025-2026\\Winter_Sem\\PHYS_258\\lab3_photoelec\\labtools.py'>
```

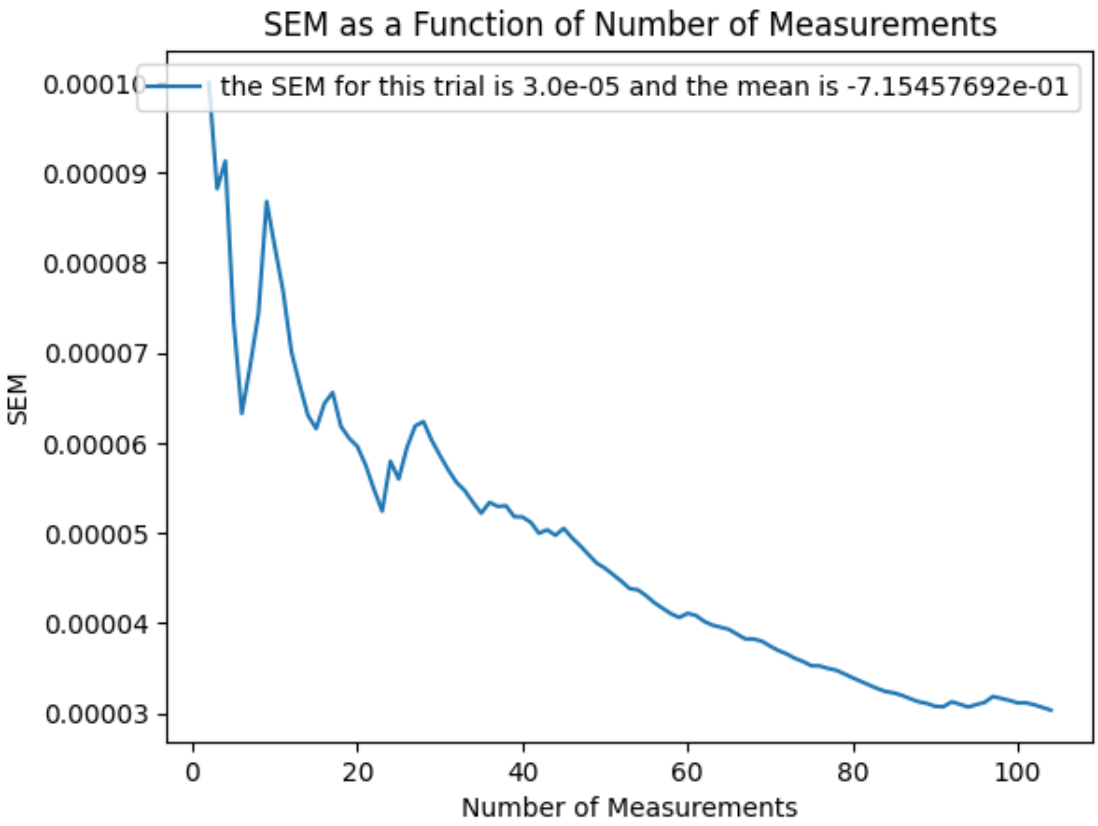
[4]:
```
#data sorting and filtering for the different runs and experiments, pretty␣
 ↪boring ngl
#This is for run 1, we are mesuring stopping volatage on y axis against input␣
 ↪(1/wavelength on the y axis)
os.getcwd()
a =os.listdir()
#print(a)
uglyrun1_vol_wl = np.genfromtxt("trial 1.csv",delimiter = ",",skip_header =1 )
#print(uglyrun1_vol_wl)

run1_vols = uglyrun1_vol_wl[:,1::3][0:104,:] #run 1, every column is a␣
 ↪different wl
#the slice [0:104,:] was taken to remove the "nan" out of everywhere. this will␣
 ↪be done
#for every run
#find out how many measurements to take from there

for i in range (run1_vols[0:1, :].size):
  print(f" SEM on voltage measurement associated with a particular wavelength␣
 ↪number {i+1} for run number 1")
  lbts.num_of_trials_visual(run1_vols[:,i])
```

```
 SEM on voltage measurement associated with a particular wavelength number 1 for
run number 1
```

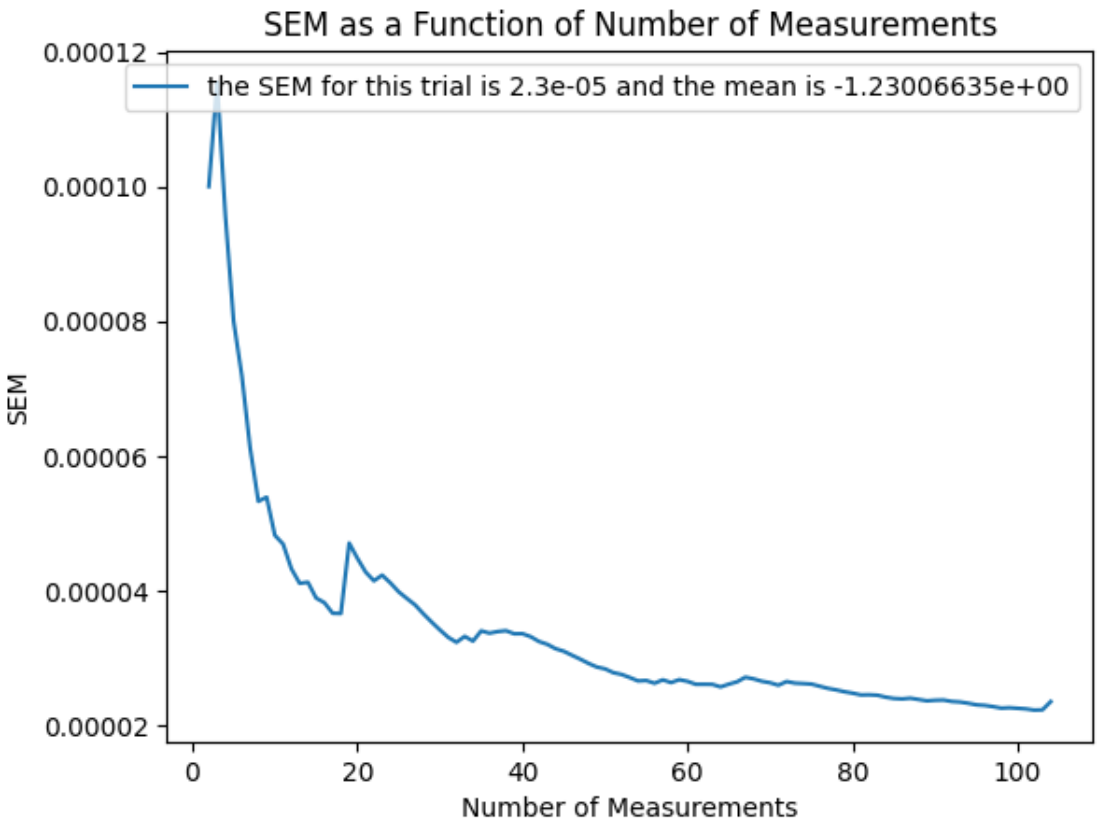
```
c:\Adult
life\McGill\2025-2026\Winter_Sem\PHYS_258\lab3_photoelec\labtools.py:71:
RuntimeWarning: invalid value encountered in divide
  var = var_pop * n / (n - ddof)
```



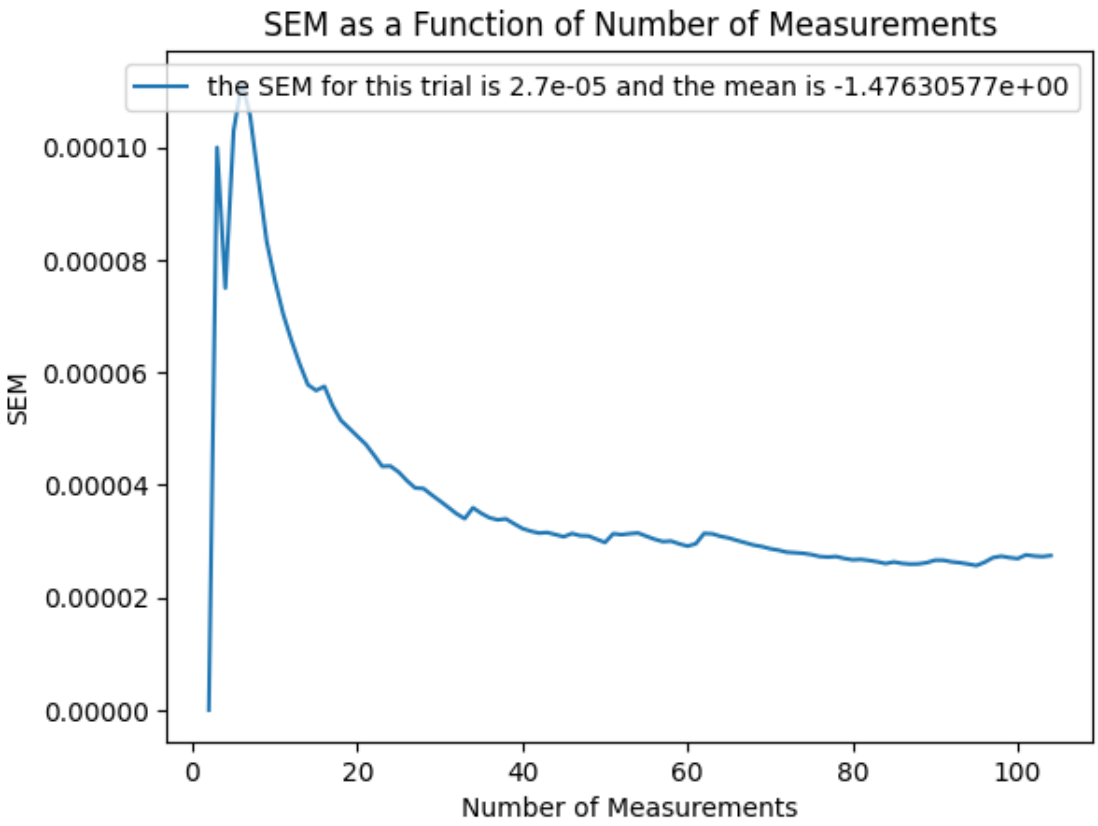
SEM on voltage measurement associated with a particular wavelength number 2 for run number 1



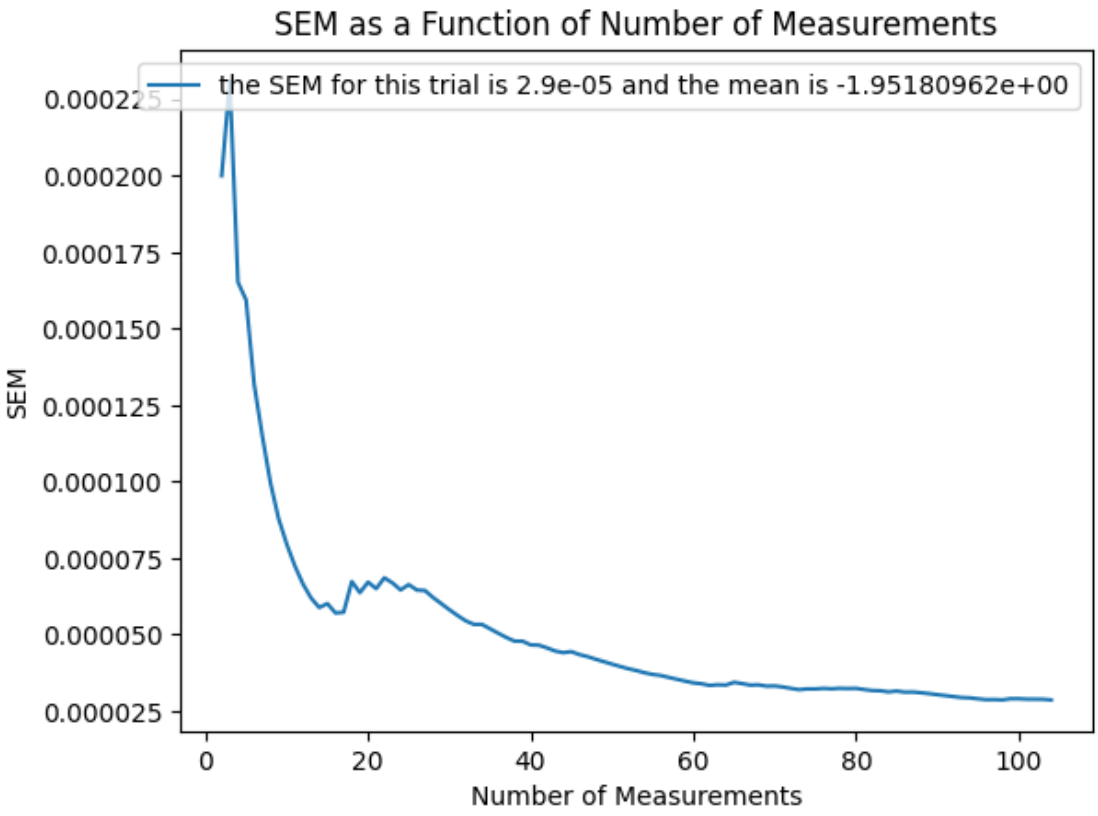
SEM on voltage measurement associated with a particular wavelength number 3 for run number 1



SEM on voltage measurement associated with a particular wavelength number 4 for run number 1



SEM on voltage measurement associated with a particular wavelength number 5 for run number 1



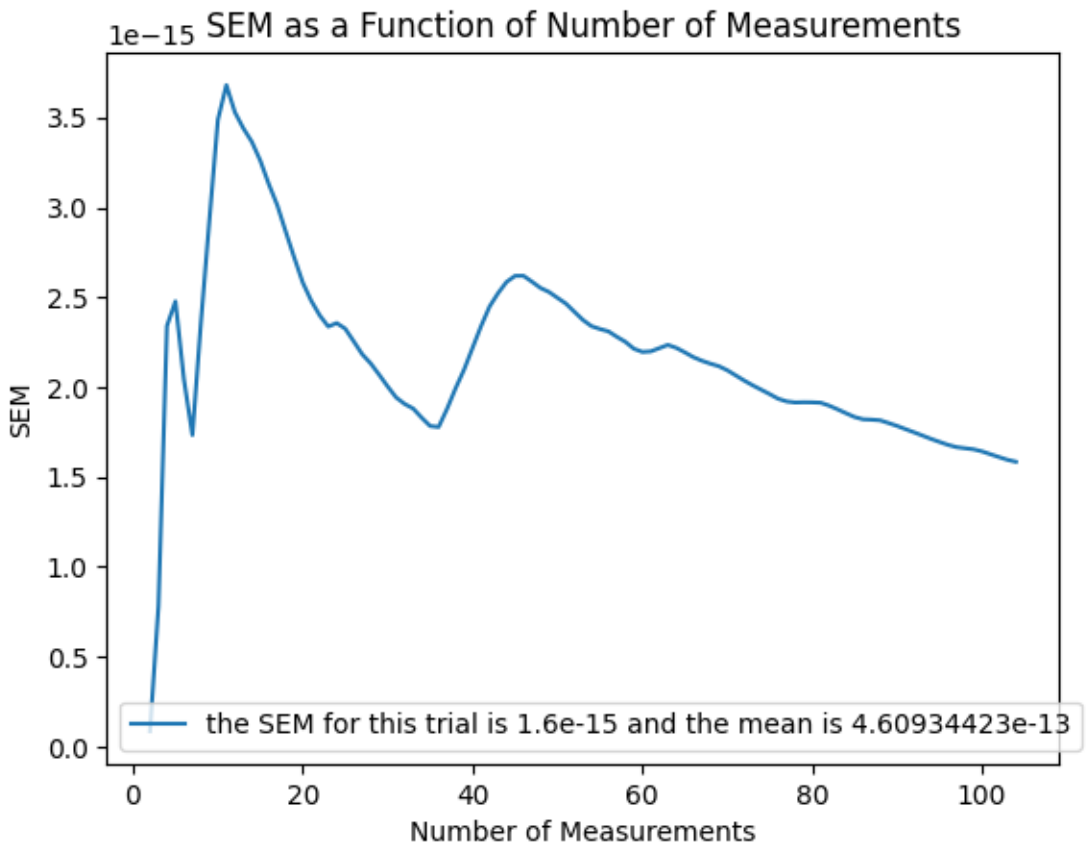
[33]:
```
#we are going to repeat this process of plotting the SEM against the the number
 ↪of data points per measured (1/wavelength)
#to see how many measumrents are enough for every single run of every single
 ↪type of graph we are going to plot
uglyrun_2_3_vols_wls = np.genfromtxt("trials 2_3.csv", delimiter = ",",
 ↪skip_header=1)
vols_wls_run2 = uglyrun_2_3_vols_wls[:,1::6][0:104,:]
#for i in range(vols_wls_run2[0:1,:].size):
  #print(f" SEM on voltage associated with a specific wavelength number {i+1}
 ↪for run number 2")
  #lbts.num_of_trials_visual(vols_wls_run2[:,i])

vols_wls_run3=  uglyrun_2_3_vols_wls[:,4::6][0:104,:]
#for i in range(vols_wls_run3[0:1,:].size):
  #print(f" SEM on voltage associated with a specific wavelength number {i+1}
 ↪for run number 3")
  #lbts.num_of_trials_visual(vols_wls_run3[:,i])
```

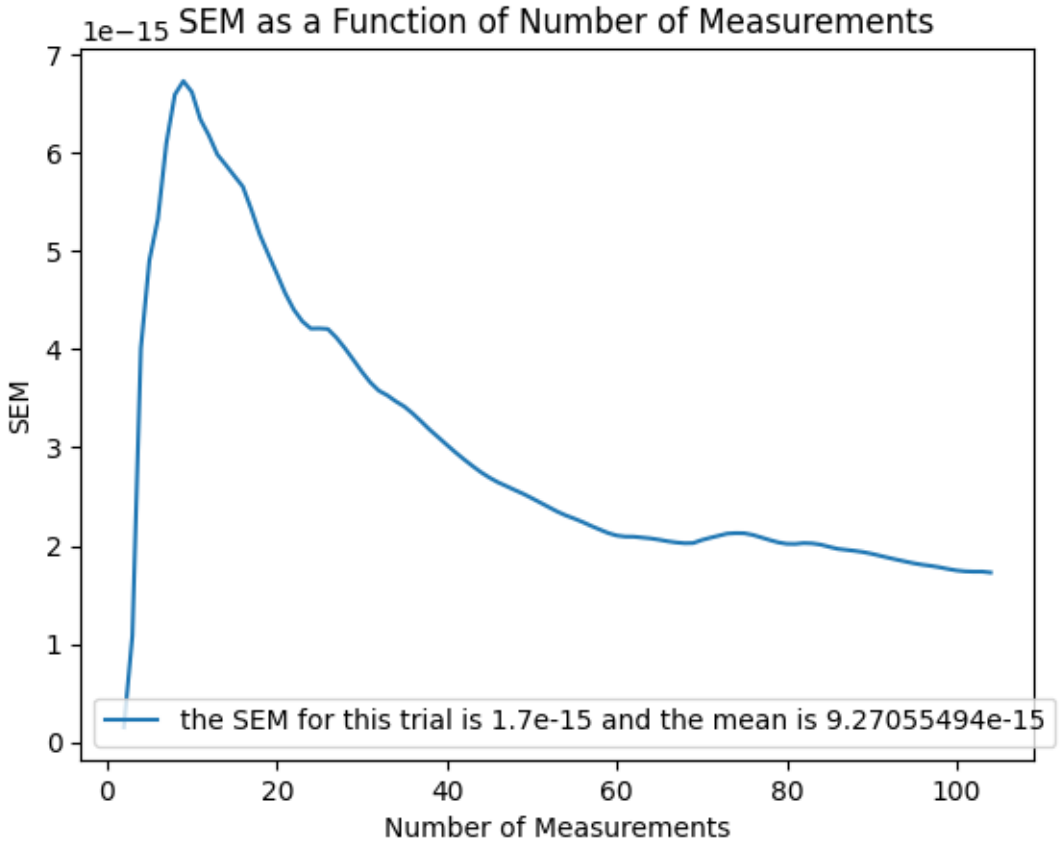
```
ugly405_apt = np.genfromtxt("405nm-8-4-2.csv", delimiter = ",", skip_header=1)
apt405_vols = ugly405_apt[:,1::3][0:100,:]
#for i in range((apt405_vols[0:1,:].size)):
  #print(f"SEM on voltage number {i+1} associated with a specific apeture size␣
 ↪run, indicent beam of 405nm")
  #lbts.num_of_trials_visual(apt405_vols[:,i])

fakeapt405_ins = ugly405_apt[:,2::3][0:102,:]
apt405_ins = fakeapt405_ins[:,::-1]
#for i in range((apt405_ins[0:1,:].size)):
  #print(f"SEM on current number {i+1} associated with a specific apeture size,␣
 ↪indicent beam of 405nm")
  #lbts.num_of_trials_visual(apt405_ins[:,i])


ugly436_apt  = np.genfromtxt("436nm-8-4-2.csv", delimiter = ",", skip_header=1)
apt436_vols = ugly436_apt[:,1::3][0:104,:]
#for i in range((apt436_vols[0:1,:].size)):
  #print(f"SEM on specific voltage number {i+1} associated with an apeture␣
 ↪size, indicent beam of 436nm ")
  #lbts.num_of_trials_visual(apt436_vols[:,i])

fakeapt436_ins = ugly436_apt[:,2::3] [0:104,:]
apt436_ins = fakeapt436_ins[:,::-1]
#for i in range(apt436_ins[0:1,:].size):
  #print(f"SEM on specific current number {i+1} associated with an apeture␣
 ↪size, indcident beam of 436nm ")
  #lbts.num_of_trials_visual(apt436_ins[:,i])

ugly_distances1_405_436 = np.genfromtxt("distances10-20-30 for␣
 ↪wavelengths405-436 trial 1.csv",  delimiter = ",", skip_header=1)

dists_run1_436_vols = ugly_distances1_405_436[:, 16:0:-6] #kept constant at␣
 ↪about 1 V to avoid current sensor overflow
dists_run1_436_ins = ugly_distances1_405_436[:,17:1:-6] [0:104,:]
#for i in range(dists_run1_436_ins[0:1,:].size):
  #print(f"SEM on intensity measurement number {i+1} for a specific distance␣
 ↪run number 1 with incident beam of 436 nm")
  #lbts.num_of_trials_visual(dists_run1_436_ins[:,i])
dists_run1_405_vols =  ugly_distances1_405_436[:,13:0:-6] #kept constant at ␣
 ↪about 1V to avoid current sensor overflow
dists_run1_405_ins = ugly_distances1_405_436[:,14:1:-6][0:104,:]
#for i in range(dists_run1_405_ins[0:1,:].size):
  #print(f"SEM on intensity measurment number {i+1} for a specific distance run␣
 ↪number 1 with incident beam of 405 nm")
  #lbts.num_of_trials_visual(dists_run1_436_ins[:,i])
```

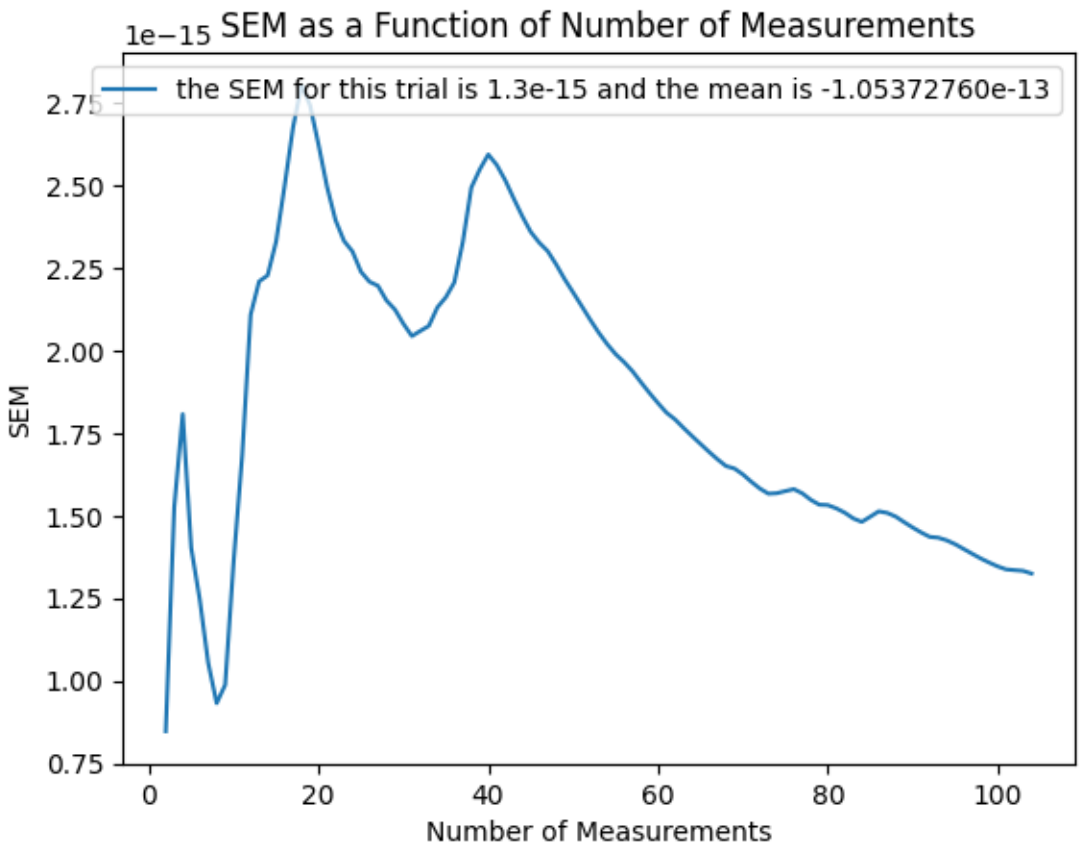
```
ugly_distances2_405_436 = np.genfromtxt("distances10-20-30 for␣
 ↪wavelengths405-436 trial 2.csv",  delimiter = ",", skip_header=1)

dists_run2_436_vols = ugly_distances2_405_436[:, 16:0:-6] #kept constant at ␣
 ↪about 1V to avoid current sensor overflow
dists_run2_436_ins = ugly_distances2_405_436[:,17:1:-6][0:104,:]
#for i in range(dists_run2_436_ins[0:1,:].size):
  #print(f"SEM on intensity measurement number {i+1} for varying distances, run␣
 ↪number 2, incident beam of 436 nm")
  #lbts.num_of_trials_visual(dists_run2_436_ins[:,i])
dists_run2_405_vols =  ugly_distances2_405_436[:, 13:0:-6]#kept constant at ␣
 ↪about 1V to avoid current sensor overflow
dists_run2_405_ins = ugly_distances2_405_436[:,14:1:-6][0:104,:]

#for i in range(dists_run2_405_ins[0:1,:].size):
  #print(f"SEM on intensity measurement number {i+1} for varying distances, run␣
 ↪number 2, incident beam of 405 nm ")
  #lbts.num_of_trials_visual(dists_run2_405_ins[:,i])
ugly_distances3_405_436 = np.genfromtxt("distances10-20-30 for␣
 ↪wavelengths405-436 trial 3.csv",  delimiter = ",", skip_header=1)

dists_run3_436_vols = ugly_distances3_405_436[:, 16:0:-6] #kept constant at ␣
 ↪about 1V to avoid current sensor overflow
dists_run3_436_ins = ugly_distances3_405_436[:,17:1:-6][0:104,:]
#for i in range(dists_run3_436_ins[0:1,:].size):
  #print(f'SEM on intensity measurement number {i+1} for varying distances, run␣
 ↪number 3, incident beam of 436 nm')
  #lbts.num_of_trials_visual(dists_run3_436_ins[:,i])
dists_run3_405_vols =  ugly_distances3_405_436[:, 13:0:-6] #kept constant at ␣
 ↪about 1V to avoid current sensor overflow
dists_run3_405_ins = ugly_distances3_405_436[:,14:1:-6][0:104,:]
#for i in range(dists_run3_405_ins[0:1,:].size):
  #print(f"SEM on intensity measurement number {i+1} for varying distances, run␣
 ↪number 3 incident beam of 405 nm")
  #lbts.num_of_trials_visual(dists_run3_405_ins[:,i])
volstop_and_int_as_apt = np.genfromtxt("stopping voltage from intensity-8-4-2.
 ↪csv",  delimiter = ",", skip_header=1)
szgd_volstop_and_int_as_apt = volstop_and_int_as_apt
vols8_4_2 = volstop_and_int_as_apt[:,1::3][0:104,:]
#for i in range(vols8_4_2[0:1,:].size):
  #print(f'SEM on voltage observed number {i+1} for varying apeture sizes')
  #lbts.num_of_trials_visual(vols8_4_2[:,i])

int8_4_2 = volstop_and_int_as_apt[:,2::3][0:104,:]
```

```
for i in range(int8_4_2 [0:104,:][0:1,:].size):
  lbts.num_of_trials_visual(int8_4_2[:,i])

print(int8_4_2[:,1])
print("As we can see for graph above the one just above this statement, this␣
 ↪trial will have to be discarded "
      "(4mm apeture run), " \
      "because the variations in data were inconsistent (there were many␣
 ↪changes in current magnitude throughout this trial)" \
      "that caused the SEM to be of such a high magnitude with respect to the␣
 ↪measurements")


'''####### MAJOR WARNING for the trial with a 4 mm apeture was discarded,
###as literally there was too much variation in the current on that specific␣
 ↪trial. not just noise variations, but also
###direction of current changes, because holy shit that thing went from␣
 ↪positive current to negative current at least twice
####A plot will be shown to jsutify us deleting this thing. Furthermore, that␣
 ↪means that we cannot use either the 8mm apeture trial
# or the 2mm apeture trial, as we will have no dof left, and it is not␣
 ↪statistically valid to fit a two param curve onto two points
# and draw conclusions. Yeah'''
```

```
c:\Adult
life\McGill\2025-2026\Winter_Sem\PHYS_258\lab3_photoelec\labtools.py:72:
RuntimeWarning: invalid value encountered in divide
  sem = np.sqrt(var) / np.sqrt(n)
```


SEM as a Function of Number of Measurements
1e−15
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
SEM
0
20
40
60
80
100
Number of Measurements
the SEM for this trial is 1.6e-15 and the mean is 4.60934423e-13


SEM as a Function of Number of Measurements
1e−15
7
6
5
4
3
2
1
0
SEM
0
20
40
60
80
100
Number of Measurements
the SEM for this trial is 1.7e-15 and the mean is 9.27055494e-15




```
[-5.3125e-14 -5.3430e-14 -5.0037e-14 -3.6450e-14 -2.9285e-14 -2.3401e-14
 -1.2158e-14 -5.2612e-15 -2.3926e-15 -1.9897e-15 -3.7720e-15  1.2207e-15
  1.8311e-15  6.8237e-15  8.6182e-15  1.0437e-14  3.2227e-15 -4.7607e-16
  3.3691e-15  2.5269e-15 -5.8594e-16  3.2104e-15  7.9956e-15  1.3159e-14
  2.1155e-14  2.2290e-14  1.4270e-14  1.0596e-14  7.0557e-15  3.9551e-15
  5.6641e-15  1.1218e-14  1.8274e-14  1.5979e-14  1.7566e-14  1.2781e-14
  8.3984e-15  3.9795e-15  7.1167e-15  2.7100e-15 -9.7656e-17 -1.5137e-15
 -1.1719e-15  2.2461e-15  8.0444e-15  1.0828e-14  1.4075e-14  1.3220e-14
  1.3367e-14  1.0217e-14  7.8979e-15  5.2734e-15  3.1738e-16 -7.1533e-15
 -1.0681e-14 -8.0933e-15 -2.0264e-15  5.1758e-15  5.1880e-15  1.3147e-14
  2.0068e-14  2.4146e-14  2.1729e-14  2.2034e-14  2.0117e-14  1.9263e-14
  2.1118e-14  2.3486e-14  2.7734e-14  3.8049e-14  3.7024e-14  3.7500e-14
  3.8647e-14  3.3411e-14  3.1409e-14  2.3022e-14  1.3123e-14  9.7656e-15
  1.4282e-14  2.3401e-14  3.1836e-14  3.7329e-14  3.1824e-14  2.6270e-14
  1.4563e-14  1.6431e-14  2.7515e-14  2.7759e-14  2.5781e-14  2.0459e-14
  1.3281e-14  6.3232e-15  3.8330e-15  3.3936e-15  2.2217e-15 -3.4302e-15
 -7.2266e-15 -2.9663e-15  3.0396e-15  1.8347e-14  2.5830e-14  3.1116e-14
  3.3850e-14  2.6111e-14]
As we can see for graph above the one just above this statement, this trial will
have to be discarded (4mm apeture run), because the variations in data were
```

inconsistent (there were many changes in current magnitude throughout this trial)that caused the SEM to be of such a high magnitude with respect to the measurements

[33]:
```
'####### MAJOR WARNING for the trial with a 4 mm apeture was discarded,\n###as literally there was too much variation in the current on that specific trial. not just noise variations, but also \n###direction of current changes, because holy shit that thing went from positive current to negative current at least twice\n####A plot will be shown to jsutify us deleting this thing. Furthermore, that means that we cannot use either the 8mm apeture trial \n# or the 2mm apeture trial, as we will have no dof left, and it is not statistically valid to fit a two param curve onto two points \n# and draw conclusions. Yeah'
```

[ ]:
```
##Finally, after 44 hours wasted, finally time to start plotting and fitting some graphs!!
#First, graph of stopping voltage as a function of the incident wavelength
#first, before plotting, perofmr a weighted mean of all three runs and their uncs
#to then plot most representative graph

rawxdata = np.asarray(np.transpose([[1/((577)*(10**-9))],[1/(546*(10**-9))],[1/((436*(10**-9)))],[1/((405*(10**-9)))],[1/((365*(10**-9)))]]))
rawxerr = np.asarray(np.full((1,5),None))
rawydatar1 = np.asarray(run1_vols[0:105,:])
rawyerr = np.asarray(np.full((104,5),0.0001))

plottable_xdata = lbts.find_plottabl_stuff(rawxdata, rawxerr)[0]# throw these into goated function

plottable_xuncs = lbts.find_plottabl_stuff(rawxdata,rawxerr)[1] #subject to comparison when plotting
#print(plottable_xuncs.shape)
plottable_ydatar1 = lbts.find_plottabl_stuff(rawydatar1,None)[0]#throw these into goated function
plottable_yuncsr1 = lbts.find_plottabl_stuff(rawydatar1,None)[1]#subject to comparison when plotting
#print(plottable_ydatar1)
#print(plottable_yuncsr1)


plottable_ydatar2 = lbts.find_plottabl_stuff(vols_wls_run2)[0]
plottable_yuncsr2 = lbts.find_plottabl_stuff(vols_wls_run2)[1]
#print(plottable_ydatar2)
#print(plottable_yuncsr2)
plottable_ydatar3 = lbts.find_plottabl_stuff(vols_wls_run3)[0]
plottable_yuncsr3 = lbts.find_plottabl_stuff(vols_wls_run3)[1]
#print(plottable_ydatar3)
```

```
#print(plottable_yuncsr3)

#final_xunc = lbts.uncer_compare(plottable_xuncs,rawxerr)
#final_yunc = lbts.uncer_compare(plottable_yuncsr3,rawyerr)
```

[49]:
```
runs = [run1_vols[0:105,:],vols_wls_run2,vols_wls_run3]
useful = lbts.run_stacker(runs)
FINAL_yplotdata = lbts.weighted_mean(useful[0],useful[1])[0]
almost_yplotunc = lbts.weighted_mean(useful[0],useful[1])[1]
FINAL_yplotunc = np.sqrt((0.0001**2)+(almost_yplotunc)**2)
rawxerr = None
raw_xdata = np.asarray(np.transpose([[1/((577)*(10**-9))],[1/(546*(10**-9))],[1/
 ↪((436*(10**-9)))],[1/((405*(10**-9)))],[1/((365*(10**-9)))]]))
plottable_xdata = lbts.find_plottabl_stuff(raw_xdata, rawxerr)[0]
#print(FINAL_yplotdata.shape)
#print(FINAL_yplotunc)
#print(almost_yplotunc)

print(FINAL_yplotdata)
print(FINAL_yplotunc)
plz = lbts.linear_fitter(plottable_xdata,FINAL_yplotdata,FINAL_yplotunc)
#print(plz.red_chi2)
#print(plz.pvalue)
#print(plz.chi2)
#print(plottable_xdata.shape)
xticks = ["1/577","1/546","1/436","1/405","1/365"]

#lbts.plot_shower(plottable_xdata,FINAL_yplotdata,xerr = None,yerr=␣
 ↪FINAL_yplotunc,plot = plz,xticks = xticks)

print(f"relationship 1 has an R2 value of {plz.R2:.3e}")
print(plz)


planck = (-(FINAL_yplotdata)*(-1.6*(10**-19)) *(1/plottable_xdata))/(3*(10**8))␣
 ↪+ ((plz.params[1])*(1/plottable_xdata))/(3*(10**8))
print(planck)

better_planck = ((plz.params[0])* (-1.6*(10**(-19)))) /   (3*(10**8))
print(better_planck)

unc_planck  = abs((-1.6*(10**(-19))) * plz.params_errs[0]/(3*(10**8)))
print(unc_planck)

work_func = plz.params[1]*(-1.6*(10**(-19)))
uncer_work_func = (-1.6*(10**(-19)))*plz.params_errs[1]
print(work_func)
```

```
print(uncer_work_func)
```

```
[-0.60447294 -0.71750664 -1.23329432 -1.47620928 -1.95310781]
[0.00010162 0.00010126 0.00010109 0.00010133 0.00010206]
relationship 1 has an R2 value of 9.887e-01
FitResult(model='linear fit', params=array([-1.29513330e-06,  1.67004846e+00]),
params_errs=array([1.19390201e-10, 2.68010347e-04]), covar=array([[
1.42540201e-20, -3.15358095e-14],
       [-3.15358095e-14,  7.18295461e-08]]), chi2=1348485.0230828403, dof=3,
red_chi2=449495.00769428007, pvalue=0.0, yhat=array([-0.57454998, -0.70199055,
-1.30044076, -1.52781154, -1.87826195]), R2=np.float64(0.9887096253904784),
residuals=array([-0.02992296, -0.01551609,  0.06714644,  0.05160226,
-0.07484586]))
[3.21205987e-15 3.03948820e-15 2.42713710e-15 2.25456542e-15
 2.03189229e-15]
6.9073776033167675e-34
6.36747737698302e-38
-2.672077537951827e-19
-4.288165553310172e-23
```

[ ]:
```
#now time to investigate the relationshipe between voltage and the apeture size.
#405nm light shown upon
runs = [apt405_vols]
useful = lbts.run_stacker(runs)

useful = lbts.run_stacker(runs)
FINAL_yplotdata = lbts.weighted_mean(useful[0],useful[1])[0]
almost_yplotunc = lbts.weighted_mean(useful[0],useful[1])[1]
FINAL_yplotunc = np.sqrt((0.0001**2)+(almost_yplotunc)**2)

plottable_xdata = np.asarray([np.pi, 4*np.pi, 16*np.pi])

plz = lbts.linear_fitter(plottable_xdata,FINAL_yplotdata,FINAL_yplotunc)
xticks = ["pi", "4pi","16pi"]

lbts.plot_shower(plottable_xdata,FINAL_yplotdata,xerr = None,yerr=␣
 ↪FINAL_yplotunc,plot = plz,xticks = xticks)
```

[ ]:
```
#Same idea, just now with a different incident beam, 436 nm

runs = [apt436_vols]
useful = lbts.run_stacker(runs)

useful = lbts.run_stacker(runs)
FINAL_yplotdata = lbts.weighted_mean(useful[0],useful[1])[0]
almost_yplotunc = lbts.weighted_mean(useful[0],useful[1])[1]
FINAL_yplotunc = np.sqrt((0.0001**2)+(almost_yplotunc)**2)
```

```
plottable_xdata = np.asarray([np.pi, 4*np.pi, 16*np.pi])

plz = lbts.linear_fitter(plottable_xdata,FINAL_yplotdata,FINAL_yplotunc)
xticks = ["pi", "4pi","16pi"]

lbts.plot_shower(plottable_xdata,FINAL_yplotdata,xerr = None,yerr=␣
 ↪FINAL_yplotunc,plot = plz,xticks = xticks)
```

[ ]:
```
## Now plot current against the area illuminated, this we expect a proportional␣
 ↪linear relationship
runs = [apt405_ins]
useful = lbts.run_stacker(runs)


useful = lbts.run_stacker(runs)
FINAL_yplotdata = lbts.weighted_mean(useful[0],useful[1])[0]
almost_yplotunc = lbts.weighted_mean(useful[0],useful[1])[1]
FINAL_yplotunc = np.sqrt(((10**(-12))**2)+(almost_yplotunc)**2)

plottable_xdata = np.asarray([np.pi, 4*np.pi, 16*np.pi])

plz = lbts.linear_fitter(plottable_xdata,FINAL_yplotdata,FINAL_yplotunc)
xticks = ["pi", "4pi","16pi"]

lbts.plot_shower(plottable_xdata,FINAL_yplotdata,xerr = None,yerr=␣
 ↪FINAL_yplotunc,plot = plz,xticks = xticks)
```

[ ]:
```
#same idea but with a 436 nm incident beam

runs = [apt436_ins]
useful = lbts.run_stacker(runs)


useful = lbts.run_stacker(runs)
FINAL_yplotdata = lbts.weighted_mean(useful[0],useful[1])[0]
almost_yplotunc = lbts.weighted_mean(useful[0],useful[1])[1]
FINAL_yplotunc = np.sqrt(((10**(-12))**2)+(almost_yplotunc)**2)

plottable_xdata = np.asarray([np.pi, 4*np.pi, 16*np.pi])

plz = lbts.linear_fitter(plottable_xdata,FINAL_yplotdata,FINAL_yplotunc)
xticks = ["pi", "4pi","16pi"]

lbts.plot_shower(plottable_xdata,FINAL_yplotdata,xerr = None,yerr=␣
 ↪FINAL_yplotunc,plot = plz,xticks = xticks)
```

[ ]:
```
runs = [dists_run1_436_vols[0:102,:],
        dists_run2_436_vols[0:102,:], dists_run3_436_vols[0:102,:]]
useful = lbts.run_stacker(runs)
#print(np.any(np.isnan(useful)))

#print(runs)

useful = lbts.run_stacker(runs)
FINAL_yplotdata = lbts.weighted_mean(useful[0],useful[1])[0]
almost_yplotunc = lbts.weighted_mean(useful[0],useful[1])[1]
FINAL_yplotunc = np.sqrt(((0.0001)**2)+(almost_yplotunc)**2)

#print(FINAL_yplotdata)
#print(FINAL_yplotunc)
#print()

r= np.array([44,34,24])
plottable_xdata = np.asarray([1/(44**2),1/(34**2),1/(24**2)])

plz = lbts.linear_fitter(plottable_xdata,FINAL_yplotdata,FINAL_yplotunc)
xticks = ["1/(44^2)", "1/(34^2)","1/(24^2)"]
errx = []
for i in range(len(xticks)):
  errxx = 2*(0.05)/(r[i]**3)
  errx.append(errxx)

xerr = np.asarray(errx)
#print(xerr.shape)
#print(xerr)
#print(FINAL_yplotdata)
#print(FINAL_yplotunc)
lbts.plot_shower(plottable_xdata,FINAL_yplotdata,xerr =xerr ,yerr=␣
 ↪FINAL_yplotunc,plot = plz,xticks = xticks)
#plt.errorbar(plottable_xdata,FINAL_yplotdata,yerr= FINAL_yplotunc)
#plt.show()
```

[ ]:
```
runs = [dists_run1_436_ins[0:102,::-1],
        dists_run2_436_ins[0:102,::-1], dists_run3_436_ins[0:102,::-1]]
useful = lbts.run_stacker(runs)
#print(np.any(np.isnan(useful)))

#print(runs)

useful = lbts.run_stacker(runs)
FINAL_yplotdata = lbts.weighted_mean(useful[0],useful[1])[0]
almost_yplotunc = lbts.weighted_mean(useful[0],useful[1])[1]
FINAL_yplotunc = np.sqrt(((10**(-12))**2)+(almost_yplotunc)**2)
```

```
print(FINAL_yplotdata)
print(FINAL_yplotunc)


r= np.array([44,34,24])
plottable_xdata = np.asarray([1/(44),1/(34),1/(24)])

plz = lbts.linear_fitter(plottable_xdata,FINAL_yplotdata,FINAL_yplotunc)
xticks = ["1/(44^2)", "1/(34^2)","1/(24^2)"]
errx = []
for i in range(len(xticks)):
  errxx = 2*(0.05)/(r[i]**3)
  errx.append(errxx)

xerr = np.asarray(errx)

lbts.plot_shower(plottable_xdata,FINAL_yplotdata,xerr =xerr ,yerr=␣
 ↪FINAL_yplotunc,plot = plz,xticks = xticks)
```

[ ]:
```
runs = [dists_run1_405_ins[0:102,::-1],
        dists_run2_405_ins[0:102,::-1], dists_run3_405_ins[0:102,::-1]]
useful = lbts.run_stacker(runs)
print(np.any(np.isnan(useful)))

#print(runs)

useful = lbts.run_stacker(runs)
FINAL_yplotdata = lbts.weighted_mean(useful[0],useful[1])[0]
almost_yplotunc = lbts.weighted_mean(useful[0],useful[1])[1]
FINAL_yplotunc = np.sqrt(((10**(-12))**2)+(almost_yplotunc)**2)

print(FINAL_yplotdata)
print(FINAL_yplotunc)


r= np.array([44,34,24])
plottable_xdata = np.asarray([1/(44),1/(34),1/(24)])

plz = lbts.linear_fitter(plottable_xdata,FINAL_yplotdata,FINAL_yplotunc)
xticks = ["1/(44^2)", "1/(34^2)","1/(24^2)"]
errx = []
for i in range(len(xticks)):
  errxx = 2*(0.05)/(r[i]**3)
  errx.append(errxx)

xerr = np.asarray(errx)
```

```
lbts.plot_shower(plottable_xdata,FINAL_yplotdata,xerr =xerr ,yerr=␣
 ↪FINAL_yplotunc,plot = plz,xticks = xticks)
```

[25]:
```
print(lbts.num_sigfigs(dists_run1_405_ins))
```

```
1
```

[50]:
```
##Merge the wls for the IV of distance, DV is the current
xticks = ["1/(44^2)", "1/(34^2)","1/(24^2)"]
errx = []
r= np.array([44,34,24])
for i in range(len(xticks)):
  errxx = 2*(0.05)/(r[i]**3)
  errx.append(errxx)

plottable_xdata = np.asarray([1/(44**2),1/(34**2),1/(24**2)])

##Stuff for 405 nm
runs405 = [dists_run1_405_ins[0:102,::-1],
        dists_run2_405_ins[0:102,::-1], dists_run3_405_ins[0:102,::-1]]
useful405 = lbts.run_stacker(runs405)
#print(np.any(np.isnan(useful405)))

FINAL_yplotdata405 = lbts.weighted_mean(useful405[0],useful405[1])[0]
almost_yplotunc405 = lbts.weighted_mean(useful405[0],useful405[1])[1]
FINAL_yplotunc405 = np.sqrt(((10**(-12))**2)+(almost_yplotunc405)**2)

#stuff for 436 nm
runs436 = [dists_run1_436_ins[0:102,::-1],
        dists_run2_436_ins[0:102,::-1], dists_run3_436_ins[0:102,::-1]]
useful436 = lbts.run_stacker(runs436)
#print(np.any(np.isnan(useful436)))

FINAL_yplotdata436 = lbts.weighted_mean(useful436[0],useful436[1])[0]
almost_yplotunc436 = lbts.weighted_mean(useful436[0],useful436[1])[1]
FINAL_yplotunc436 = np.sqrt(((10**(-12))**2)+(almost_yplotunc436)**2)

plz405 = lbts.
 ↪linear_fitter(plottable_xdata,FINAL_yplotdata405,FINAL_yplotunc405)

plz436 = lbts.
 ↪linear_fitter(plottable_xdata,FINAL_yplotdata436,FINAL_yplotunc436)
'''
lbts.
 ↪multiple4_plot_shower(plottable_xdata,plottable_xdata,FINAL_yplotdata405,plz405,
                        FINAL_yplotdata436,plz436,xerr1= errx,xerr2=errx,
```

```
                            xticks1 =␣
 ↪xticks,xticks2=xticks,yerr1=FINAL_yplotunc405,
                            yerr2 = FINAL_yplotunc436)
'''

print(f"relationship 2 with an incident beam of 405 nm has an R2 value of␣
 ↪{plz405.R2:.3e}")
print(f"relationship 2 with an incident beam of 436 nm has an R2 value of␣
 ↪{plz436.R2:.3e}")
print(plz405)
print(plz436)
print("break")
print(FINAL_yplotdata405)
print(FINAL_yplotunc405)
print(FINAL_yplotdata436)
print(FINAL_yplotunc436)
```

```
relationship 2 with an incident beam of 405 nm has an R2 value of 9.992e-01
relationship 2 with an incident beam of 436 nm has an R2 value of 9.983e-01
FitResult(model='linear fit', params=array([ 2.75156679e-07, -5.05237597e-11]),
params_errs=array([1.13299929e-09, 1.30966018e-12]), covar=array([[
1.28368739e-18, -1.33000251e-21],
       [-1.33000251e-21,  1.71520978e-24]]), chi2=46.453371044936475, dof=1,
red_chi2=46.453371044936475, pvalue=9.382480237262174e-12,
yhat=array([9.16026239e-11, 1.87501049e-10, 4.27178807e-10]),
R2=np.float64(0.9992153279181992), residuals=array([ 3.84839432e-12,
-5.44741031e-12,  1.56508689e-12]))
FitResult(model='linear fit', params=array([ 1.32183476e-07, -9.40047433e-12]),
params_errs=array([1.13032348e-09, 1.30722423e-12]), covar=array([[
1.27763117e-18, -1.32465695e-21],
       [-1.32465695e-21,  1.70883520e-24]]), chi2=22.81189781581761, dof=1,
red_chi2=22.81189781581761, pvalue=1.7865626251893856e-06,
yhat=array([5.88761147e-11, 1.04945093e-10, 2.20084728e-10]),
R2=np.float64(0.9983409978795308), residuals=array([-2.70083599e-12,
3.79735396e-12, -1.09313030e-12]))
break
[9.54510182e-11 1.82053639e-10 4.28743894e-10]
[1.00127324e-12 1.00676052e-12 1.00946250e-12]
[5.61752787e-11 1.08742447e-10 2.18991597e-10]
[1.00051522e-12 1.00261322e-12 1.00627962e-12]
```

[51]:
```
##Time to combine the graphs and residuals for area and current experiment
xticks = ["pi", "4pi","16pi"]
#stuff for 436nm
runs436 = [apt436_ins]
useful436 = lbts.run_stacker(runs436)
```

```
FINAL_yplotdata436 = lbts.weighted_mean(useful436[0],useful436[1])[0]
almost_yplotunc436 = lbts.weighted_mean(useful436[0],useful436[1])[1]
FINAL_yplotunc436 = np.sqrt(((10**(-12))**2)+(almost_yplotunc436)**2)

plottable_xdata = np.asarray([np.pi, 4*np.pi, 16*np.pi])

plz436 = lbts.
 ↪linear_fitter(plottable_xdata,FINAL_yplotdata436,FINAL_yplotunc436)
#stuff for 405nm
runs405 = [apt405_ins]
useful405 = lbts.run_stacker(runs405)

FINAL_yplotdata405 = lbts.weighted_mean(useful405[0],useful405[1])[0]
almost_yplotunc405 = lbts.weighted_mean(useful405[0],useful405[1])[1]
FINAL_yplotunc405 = np.sqrt(((10**(-12))**2)+(almost_yplotunc405)**2)

plz405 = lbts.
 ↪linear_fitter(plottable_xdata,FINAL_yplotdata405,FINAL_yplotunc405)
#405 called in first
'''
lbts.
 ↪multiple4_plot_shower(plottable_xdata,plottable_xdata,FINAL_yplotdata405,plz405,
                        FINAL_yplotdata436,plz436,xerr1= errx,xerr2=errx,
                        xticks1 =␣
 ↪xticks,xticks2=xticks,yerr1=FINAL_yplotunc405,
                        yerr2 = FINAL_yplotunc436)
'''
print(f"relationship 3 with an incident beam of 405 nm has an R2 value of␣
 ↪{plz405.R2:.3e}")
print(f"relationship 3 with an incident beam of 436 nm has an R2 value of␣
 ↪{plz436.R2:.3e}")
print(plz405)
print(plz436)
print(FINAL_yplotdata405)
print(FINAL_yplotunc405)
print(FINAL_yplotdata436)
print(FINAL_yplotunc436)
```

```
relationship 3 with an incident beam of 405 nm has an R2 value of 9.998e-01
relationship 3 with an incident beam of 436 nm has an R2 value of 9.972e-01
FitResult(model='linear fit', params=array([3.52959621e-12, 1.68456536e-12]),
params_errs=array([3.06052132e-14, 8.54464864e-13]), covar=array([[
9.36679072e-28, -1.86653412e-26],
       [-1.86653412e-26,  7.30110204e-25]]), chi2=2.8228420422643725, dof=1,
red_chi2=2.8228420422643725, pvalue=0.09293172260880488,
yhat=array([1.27731189e-11, 4.60387795e-11, 1.79101422e-10]),
R2=np.float64(0.9998159370603877), residuals=array([-1.03295223e-12,
```

```
1.29505385e-12, -3.24755182e-13]))
FitResult(model='linear fit', params=array([1.73602917e-12, 3.28970680e-12]),
params_errs=array([2.84476557e-14, 8.50065878e-13]), covar=array([[
8.09269115e-28, -1.77230906e-26],
       [-1.77230906e-26,  7.22611997e-25]]), chi2=10.479934940298728, dof=1,
red_chi2=10.479934940298728, pvalue=0.001206780058904562,
yhat=array([8.74360328e-12, 2.51052927e-11, 9.05520505e-11]),
R2=np.float64(0.9972097719624556), residuals=array([-1.99761194e-12,
2.49808227e-12, -5.04329346e-13]))
[1.17401667e-11 4.73338333e-11 1.78776667e-10]
[1.00003073e-12 1.00152575e-12 1.12145344e-12]
[6.74599135e-12 2.76033750e-11 9.00477212e-11]
[1.00000625e-12 1.00021994e-12 1.00492641e-12]
```

[28]:
```
##and now last graph, i think, merge the two beams, where IV was area and DV␣
 ↪was voltage
#436 nm beam stuff right now
runs436 = [apt436_vols]
useful436 = lbts.run_stacker(runs436)

FINAL_yplotdata436 = lbts.weighted_mean(useful436[0],useful436[1])[0]
almost_yplotunc436 = lbts.weighted_mean(useful436[0],useful436[1])[1]
FINAL_yplotunc436 = np.sqrt((0.0001**2)+(almost_yplotunc436)**2)

plottable_xdata = np.asarray([np.pi, 4*np.pi, 16*np.pi])
xticks = ["pi", "4pi","16pi"]

plz436 = lbts.
 ↪linear_fitter(plottable_xdata,FINAL_yplotdata436,FINAL_yplotunc436)


#405 nm stuff right now
runs405 = [apt405_vols]
useful405 = lbts.run_stacker(runs405)

FINAL_yplotdata405 = lbts.weighted_mean(useful405[0],useful405[1])[0]
almost_yplotunc405 = lbts.weighted_mean(useful405[0],useful405[1])[1]
FINAL_yplotunc405 = np.sqrt((0.0001**2)+(almost_yplotunc405)**2)

plz405 = lbts.
 ↪linear_fitter(plottable_xdata,FINAL_yplotdata405,FINAL_yplotunc405)

plz405 = lbts.
 ↪linear_fitter(plottable_xdata,FINAL_yplotdata405,FINAL_yplotunc405)
#405 called in first
'''
```

```
lbts.
 ↪multiple4_plot_shower(plottable_xdata,plottable_xdata,FINAL_yplotdata405,plz405,
                        FINAL_yplotdata436,plz436,xerr1= errx,xerr2=errx,
                        xticks1 =␣
 ↪xticks,xticks2=xticks,yerr1=FINAL_yplotunc405,
                        yerr2 = FINAL_yplotunc436)
'''
print(f"relationship with an incident beam of 405 nm has an R2 value of {plz405.
 ↪R2:.3e}")
print(f"relationship with an incident beam of 436 nm has an R2 value of {plz436.
 ↪R2:.3e}")
print(plz405)
print(plz436)
```

```
relationship with an incident beam of 405 nm has an R2 value of 5.747e-01
relationship with an incident beam of 436 nm has an R2 value of 1.166e-01
FitResult(model='linear fit', params=array([-2.61977278e-06, -1.02637930e+00]),
params_errs=array([2.99893383e-06, 8.80537764e-05]), covar=array([[
8.99360411e-12, -1.92128856e-10],
       [-1.92128856e-10,  7.75346753e-09]]), chi2=0.5793827778715157, dof=1,
red_chi2=0.5793827778715157, pvalue=0.4465543540336766, yhat=array([-1.02638753,
-1.02641222, -1.02651099]), R2=np.float64(0.5747213175993765), residuals=array([
4.85311813e-05, -6.07780420e-05,  1.29850650e-05]))
FitResult(model='linear fit', params=array([-3.92632781e-06, -1.02603699e+00]),
params_errs=array([2.94273808e-06, 8.78470595e-05]), covar=array([[
8.65970742e-12, -1.89372331e-10],
       [-1.89372331e-10,  7.71710586e-09]]), chi2=13.752167555348311, dof=1,
red_chi2=13.752167555348311, pvalue=0.00020857999451537595,
yhat=array([-1.02604932, -1.02608633, -1.02623435]),
R2=np.float64(0.11659432704464279), residuals=array([-2.36254141e-04,
2.95942935e-04, -5.98841479e-05]))
```