```
 1
 2  import numpy as np
 3  import matplotlib
 4  matplotlib.use('TkAgg') #Must use this backend to get squiggly lines on OSX.
 5  import matplotlib.pyplot as plt
 6
 7  test1_data = np.array([274,307,305,310,309,308,308,309,312,401,456,466,382,388,44
    6,429,311,355,373,392])
 8  test1_description = 'IR remote\nAll auto'
 9
10  test2_data = np.array([197,184,221,193,221,219,179,211,221,220,217,220,221,220,22
    0,220,220,217,225,220])
11  test2_description = 'IR remote\nManual focus \n& no live view'
12
13  test3_data = np.array([131,267,260,259,287,317,254,281,249,253,269,249,321,252,25
    0,494,301,244,276,247])
14  test3_description = 'IR remote\nManual focus \n& live view \n(lock up)'
15
16  test4_data = np.array([145,145,235,307,247,165,324,277,294,215,140,239,287,286,22
    1,324,328,217,208,219])
17  test4_description = 'IR remote\nManual focus \n& live view \n(no lockup)'
18
19  test5_data = np.array([488,175,107,170,135,171,171,104,171,171,125,170,122,137,17
    1,114,112,113,117,135])
20  test5_description = 'Wired direct\nManual focus \n& no live view\n(no lockup)'
21
22  test6_data = np.array([135,157,125,101,179,180,179,179,179,
23                         179,179,179,179,179,179,179,179,179,179,
24                         145,179,179,179,151,179,180,179,179,179])
25  test6_description = 'Wired 5v relay\nManual focus \n& no live view\n(no lockup)'
26
27  #test7_data = np.array([177,179,178,185,179,179,179,179,140,179,
28  #                       179,179,180,179,180,170,189,180,179,112])
29  #test7_description = 'Wired opto\nManual focus \n& no live view\n(no lockup)'
30
31  #Second iteration of test 7, better firmware on LED:
32  test7_data = np.array([178,157,149,178,133,178,178,178,178,178,126,112,137,178,13
    2,136,135,178,131])
33  #test7_description = 'Wired opto\nManual focus \n& no live view\n(no lockup)'
34  test7_description = 'My Canon 550D\nManual focus \n& no live view\n(no lockup
    )'
35
36
37  test8_data = np.array([149,225,195,184,222,100,279,98,134,110,195,289,160,245,169,
38                         226,210,229,218,216,109,138,278,108,216,256,146,160,306,129,197,2
    29,129,300,216])
39  test8_description = 'Wired opto\nManual focus \n& live view\n(lockup)'
40
41  test9_data = np.array([169,169,188,109,169,169,169,188,118,188,118,169,188,168,16
    9,117,169,137,127,188])
42  test9_description = 'Wired opto\nMF \n& dumb (fisheye)\n lens'
43
44  test10_data = np.array([269,264,248,228,249,219,229,269,269,264,
45                          237,210,230,222,264,232,214,230,232,264])
46  test10_data = test10_data - 95
47  test10_description = 'Wired opto\nMF \n& dumb (fisheye) \nlens + 95ms offset'
48
49  test11_data = np.array([169,169,178,121,129,169,154,118,133,147,
50                          169,131,132,119,136,137,128,178,169,168,
```

```
                    159,169,113,111,117,130,194,140,130,125,
                    129,133,151,115,138,169,167,121,124,125])
test11_description = 'Bright/Dim test\nMF & dumb lens'


test12_data = np.array([114,127,157,167,127,147,167,163,170,114,159,137,149,155,,
test12_description = 'Mark\'s Nikon D50 \n1/4000 F5 MF'

#JD's
test13_data = np.array([159,159,160,160,160,159,159,160,159,160,160,160,160,160,,
                    159,159,158,159,160,159,104,159,158,159,160,159,159,159,160,159,
test13_description = 'JD\'s Canon 600D\n1/4000 F5 MF'

#Sujith's:
test14_data = np.array([160,159,160,161,159,160,160,161,160,87,160,160,160,161,1
61,161,161,161,160,161])
test14_description = 'Sujith\'s Canon 600D\n1/4000 F5 MF'


#Plot the information:
background_colour='none'
scale = 2.0
fig =plt.figure(1,figsize=(8*scale,3*scale),facecolor=background_colour)
fig.subplots_adjust(left=0.1,top=0.90,bottom=0.15,right=0.97,wspace=0.3,hspace=0.
3)


plt.xkcd()


#Hacks to make XKCD style work nicely with non-white backgrounds:
from matplotlib import patheffects
from matplotlib import rcParams, rcParamsDefault, get_backend
from matplotlib import rc_context
context = rc_context()
rcParams['path.effects'] = [    patheffects.withStroke(linewidth=4*scale, foreground=
rcParams['figure.facecolor'] = background_colour

data_sets = [test1_data,
            test2_data,
            test3_data,
            test4_data,
            test5_data,
            test6_data,
            test7_data,
            test8_data,
            test9_data,
            test10_data,
            test11_data,
            test12_data,
            test13_data,
            test14_data]
data_descriptions = [test1_description,
                    test2_description,
                    test3_description,
                    test4_description,
                    test5_description,
                    test6_description,
                    test7_description,
```

```
108                 test8_description,
109                 test9_description,
110                 test10_description,
111                 test11_description,
112                 test12_description,
113                 test13_description,
114                 test14_description]
115 data_markersets = ['y.','g.','r.','b.','y.','g.','r.','y.','g.','r.','y.','b.','y.','g.']
116 x_loc = 0
117
118 # ax = fig.gca()
119 # ax.spines['right'].set_color('none')
120 # ax.spines['top'].set_color('none')
121 test_count = 0
122 for i in [6,11,12,13]: #range(0,len(data_sets)):
123     x_loc = test_count * 50 + 10
124     test_count += 1
125     x_width = 30
126     plt.plot( np.linspace(x_loc-x_width/2,x_loc+x_width/2,data_sets[i].T.size),data_sets
127     plt.text(x_loc, 520, data_descriptions[i],fontsize=10*scale,horizontalalignment='
    center')
128
129     # plt.annotate(data_descriptions[i], xy=(x_loc, 900),  xycoords='data',
130     #             xytext=(0.6, 0.3), textcoords='figure fraction',
131     #             size=20,
132     #             color='red',
133     #             #bbox=dict(boxstyle="round", fc="0.8"),
134     #             arrowprops=dict(arrowstyle="simple",
135     #                             fc="0.6", ec="none",
136     #                             #patchB=el,
137     #                             connectionstyle="arc3,rad=0.1"),
138     #             )
139
140 plt.ylim([0.0,700])
141 plt.xlim([-20, x_loc+30])
142 plt.xticks([])
143
144 plt.ylabel('delay (ms)',fontsize=10*scale)
145 plt.title('Camera delay measurements',fontsize=16*scale)
146 plt.savefig("camera shutter lag v01.png",facecolor=fig.get_facecolor(), edgecolor='
    none',transparent=True)
147 plt.show()
148
149
```