Heat Maps using Pandas groupby, and Matplotlib part 1

August 24, 2016

Heat Maps using Matplotlib

In [29]:
```
%pylab inline
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
```

```
Populating the interactive namespace from numpy and matplotlib
```

In [30]:
```
helix = pd.read_csv('Data/helix_parameters.csv')
helix.head() # just seeing that data was imported properly by outputing first 5 cells
```

Out[30]:

	job_n	Energy	n_helices	r0_A	r0_B	r0_C	omega0	delta_omega0_A	\
0	36019	-387.167	3	6.0	6.0	6.0	0	0	
1	36022	-402.606	3	6.0	6.0	6.0	0	0	
2	36020	-395.944	3	6.0	6.0	6.0	0	0	
3	36002	-389.788	3	6.0	6.0	6.0	0	0	
4	36005	-388.016	3	6.0	6.0	6.0	0	0	

	delta_omega0_B	delta_omega0_C	...	invert_B	invert_C	z1_offset_A	\
0	120	240	...	1	0	0	
1	120	240	...	1	0	0	
2	120	240	...	1	0	0	
3	120	240	...	1	0	0	
4	120	240	...	1	0	0	

	z1_offset_B	z1_offset_C	delta_t_A	delta_t_B	delta_t_C	omega1	z1
0	0.6	-3.0	0	0	0	99.999268	1.51
1	0.6	0.6	0	0	0	99.999268	1.51
2	0.6	-1.8	0	0	0	99.999268	1.51
3	-3.0	-1.8	0	0	0	99.999268	1.51
4	-3.0	1.8	0	0	0	99.999268	1.51

[5 rows x 27 columns]

In [31]:
```
# shape of the dataframe
helix.shape
```

Out[31]:
```
(47475, 27)
```

In [32]:
```
# checking what the columns are
helix.columns
```

Out[32]:
```
Index([u'job_n', u'Energy', u'n_helices', u'r0_A', u'r0_B', u'r0_C', u'omega0',
       u'delta_omega0_A', u'delta_omega0_B', u'delta_omega0_C', u'z0_offset_A',
```

```
       u'z0_offset_B', u'z0_offset_C', u'helix1 phase', u'helix 2 phase',
       u'helix3 phase', u'invert_A', u'invert_B', u'invert_C', u'z1_offset_A',
       u'z1_offset_B', u'z1_offset_C', u'delta_t_A', u'delta_t_B',
       u'delta_t_C', u'omega1', u'z1'],
      dtype='object')
```

Selecting Columns (by different methods)

In [33]:
```
# selecting a couple columns
couple_columns = helix[['Energy','helix 2 phase', 'helix1 phase']]
couple_columns.head()
```

Out[33]:

	Energy	helix 2 phase	helix1 phase
0	-387.167	0	0
1	-402.606	0	0
2	-395.944	0	0
3	-389.788	0	0
4	-388.016	0	0

In [34]:
```
# selecting same columns a different way
helix.ix[:,['Energy','helix 2 phase', 'helix1 phase']].head()
```

Out[34]:

	Energy	helix 2 phase	helix1 phase
0	-387.167	0	0
1	-402.606	0	0
2	-395.944	0	0
3	-389.788	0	0
4	-388.016	0	0

Heat Map

In [35]:
```
# this is essentially would be taking the average of each unique combination.
# one important mention is notice how little the data varies from eachother.
phase_1_2 = couple_columns.groupby(['helix1 phase', 'helix 2 phase']).mean()
print phase_1_2.shape
phase_1_2.head(10)
```

```
(100, 1)
```

Out[35]:

		Energy
helix1 phase	helix 2 phase	
0	0	-392.419841
	20	-389.622691
	40	-390.318620
	60	-392.198537
	80	-393.661624
	100	-392.226253
	120	-390.955112
	140	-394.319969
	160	-392.594862
	180	-389.254009

In [36]:
```
phase_1_2 = phase_1_2.reset_index()
phase_1_2.head()
```

Out[36]:

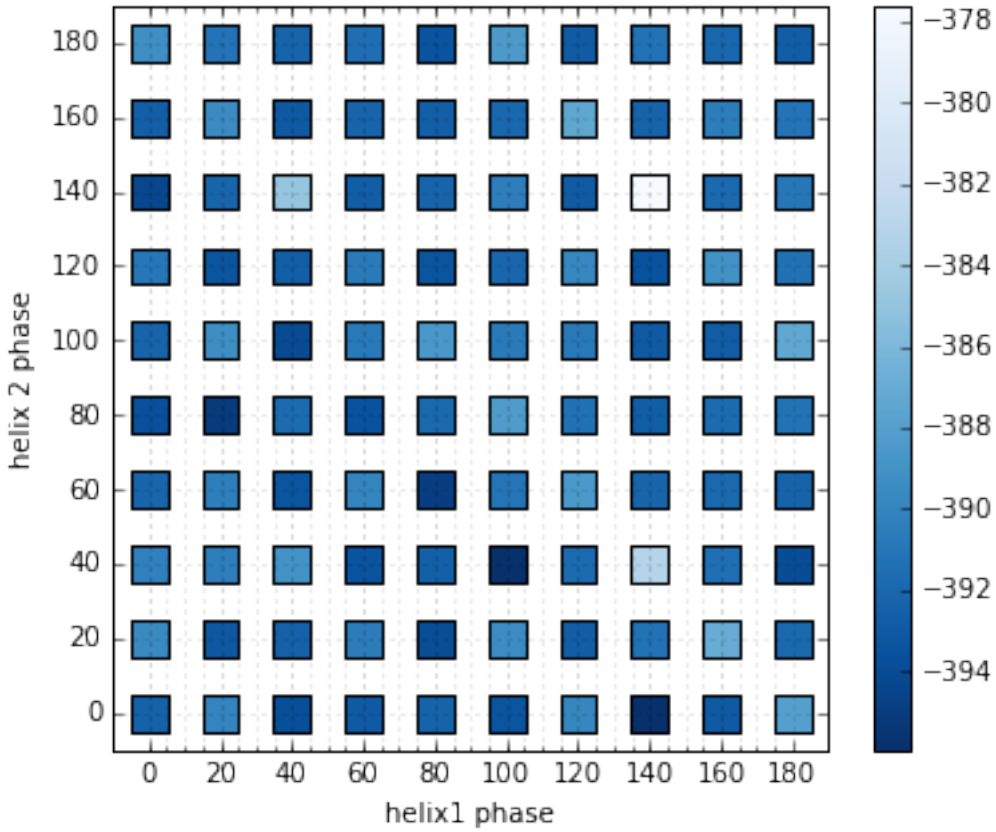
	helix1 phase	helix 2 phase	Energy
0	0	0	-392.419841
1	0	20	-389.622691
2	0	40	-390.318620
3	0	60	-392.198537
4	0	80	-393.661624

In [39]:
```
major_ticks = np.arange(0, 200, 20)
minor_ticks = np.arange(0, 180, 5)

fig = plt.figure(figsize = (6,5))
ax = fig.add_subplot(1,1,1)
s = ax.scatter('helix1 phase', 'helix 2 phase', c = 'Energy',data = phase_1_2, cmap = 'Blues_r
ax.axis([phase_1_2['helix1 phase'].min()-10, phase_1_2['helix1 phase'].max()+10, phase_1_2['he
ax.set_xticks(major_ticks)
ax.set_xticks(minor_ticks, minor=True)
ax.set_yticks(major_ticks)
ax.grid(which='both', alpha = 0.3)
ax.grid(which='major', alpha=0.3)
ax.set_xlabel('helix1 phase', fontsize=10);
ax.set_ylabel('helix 2 phase', fontsize=10);

# http://stackoverflow.com/questions/13943217/how-to-add-colorbars-to-scatterplots-created-lik
cbar = plt.colorbar(mappable = s,ax = ax)

plt.show()
```



In [ ]: