In [1]:
```
import pandas as pd
import numpy as np
from statsmodels.formula.api import ols
from statsmodels.graphics.factorplots import interaction_plot
import statsmodels.api
import scipy.stats.mstats as mstats
kw_test = mstats.kruskalwallis
import matplotlib.pyplot as plt
import matplotlib as mpl
from scipy.stats.stats import pearsonr
from scipy.stats import shapiro
from scipy.stats import histogram
from statsmodels.stats.anova import anova_lm
from scipy.stats import ttest_ind

from scipy.stats import spearmanr
from scipy.stats import mannwhitneyu
from scipy.stats import fisher_exact
from scipy.stats import chi2_contingency
import scikits.statsmodels.api as sm

import math

import misc
import corrstats
from nemenyi import kw_nemenyi

#pd.options.display.mpl_style = 'default'   # makes pretty colors f
or plots

%matplotlib inline
```

```
(2.4410345510875615, 0.015524222131207077)
(0.36402849584643454, 0.7158367305270712)
(0.04153184045164876, 0.3892118258766578)
(-0.19258265869547644, 0.11358641065961)
```

```
/usr/local/lib/python2.7/dist-packages/pandas/computation/expressio
ns.py:21: UserWarning: The installed version of numexpr 1.4.2 is no
t supported in pandas and will be not be used
The minimum supported version is 2.1

  "version is 2.1\n".format(ver=ver), UserWarning)
```

In [2]:
```
data_all = pd.read_csv('../Analysis/data_20160417.csv')
```

In [28]:
```
data_u = data_all[:][data_all['usable'] == 'Y']
data_pen = data_u[:][(data_u['pen_pickup'] == 'P') | (data_u['pen_p
ickup'] == 'N')] # only data for experiments with pen
data_crfn = data_all[:][(data_all['usable'] == 'Y') | (data_all['se
ries'] == 'C')]
```

In [4]:
```
turk_all = pd.read_csv('../Analysis/turk_20160418.csv')
```

In [8]:
```
turk_all['source'] = 'turk'
```

In [27]:
```
data_all['source'] = 'irl'
```

In [19]:
```
turk_condition_map = {'D': 'B', 'B': 'A'}
turk_all['condition'] = [turk_condition_map[x[0]] for x in turk_all
['ID']]
```

In [ ]:
```
```

In [29]:
```
combined = data_u.append(turk_all)
```

In [167]:
```
for d in (data_crfn, data_u, data_pen, turk_all, combined):
    for X in ('EC', 'FS', 'PT', 'PD', 'Age'):
        d['high_' + X] = ['Y' if x >= np.median(d[X]) else 'N' for
x in d[X]]
```

In [30]:
```
combined.columns
```

Out[30]:
```
Index([u'Age', u'Apathetic/Responsive', u'Artificial/Lifelike', u'A
wful/Nice', u'Dead/Alive', u'Dislike/Like', u'EC', u'FS', u'Fake/Na
tural', u'Foolish/Sensible', u'Gender', u'ID', u'Ignorant/Knowledge
able', u'Incompetent/Competent', u'Inert/Interactive', u'Irresponsi
ble/Responsible', u'Machinelike/Humanlike', u'Mechanical/Organic',
u'Moving rigidly/Moving elegantly', u'PD', u'PT', u'Stagnant/Livel
y', u'Unconscious/Conscious', u'Unfriendly/Friendly', u'Unintellige
nt/Intelligent', u'Unkind/Kind', u'Unpleasant/Pleasant', u'animac
y', u'anthropomorphic', u'condition', u'empathize', u'intelligenc
e', u'likeability', u'pen_pickup', u'pets', u'post_bad', u'pre_ba
d', u'revisit', u'series', u'source', u'stressful', u'usable', u'wi
ll_erase'], dtype='object')
```

In [31]:

```
d = combined
for metric in ('pre_bad', 'post_bad', 'anthropomorphic', 'animacy',
'likeability', 'intelligence'):
    group1 = d[metric][d['source'] == 'turk']
    group2 = d[metric][d['source'] == 'irl']
    print '>>>>>', metric
    print 'shapiro', shapiro(d[metric])
    print 'turk:', np.mean(group1), np.median(group1), len(group1)
    print 'irl:', np.mean(group2), np.median(group2), len(group2)
    print 'MW-U', mannwhitneyu(group1, group2)
```

```
>>>>> pre_bad
shapiro (0.8399193286895752, 3.041851926594008e-12)
turk: 2.25833333333 2.0 120
irl: 3.17021276596 3.0 47
MW-U (1839.5, 0.00015034743799694601)
>>>>> post_bad
shapiro (nan, 1.0)
turk: nan nan 120
irl: 3.46808510638 4.0 47
MW-U (0.0, 5.0741552809941472e-24)
>>>>> anthropomorphic
shapiro (0.9245650172233582, 1.2191345888368232e-07)
turk: 2.04666666667 1.8 120
irl: 2.74893617021 2.6 47
MW-U (1671.0, 2.0595092573314166e-05)
>>>>> animacy
shapiro (0.9799643754959106, 0.016177764162421227)
turk: 2.61666666663 2.5 120
irl: 3.1014184397 3.166666667 47
MW-U (1984.5, 0.0014525521183927038)
>>>>> likeability
shapiro (0.9405418634414673, 1.920487875395338e-06)
turk: 3.61666666667 3.6 120
irl: 4.06914893617 4.2 47
MW-U (1941.5, 0.00084464784381427173)
>>>>> intelligence
shapiro (0.9870237708091736, 0.1251356154680252)
turk: 2.87722222227 2.866666667 120
irl: 3.1917730496 3.1 47
MW-U (2336.5, 0.042791911418458707)
```

Effect of stories on how bad people feel for robot

I am going to see if the two conditions show a difference in reported pre_bad

then check breakdown by turk (video and story only), irl pre_bad (video, story + short interaction), irl post_bad (video, story + erasure)

In [306]:

```
d = combined
for metric in ('pre_bad',):
    group1 = d[metric][d['condition'] == 'A']
    group2 = d[metric][d['condition'] == 'B']
    print '>>>>>', metric
    print 'shapiro', shapiro(d[metric])
    print 'A:', np.mean(group1), np.median(group1), len(group1)
    print 'B:', np.mean(group2), np.median(group2), len(group2)
    print 'MW-U', mannwhitneyu(group1, group2)
```

```
>>>>> pre_bad
shapiro (0.8399193286895752, 3.041851926594008e-12)
A: 2.16470588235 2.0 85
B: 2.87804878049 3.0 82
MW-U (2528.0, 0.0007527632626668846)
```

report that pre_bad is higher for experiment condition

shapiro (0.8399193286895752, 3.041851926594008e-12)

A: 2.16470588235 2.0 85

B: 2.87804878049 3.0 82

MW-U (2528.0, 0.0007527632626668846)

> now breakdown by groups

In [282]:

```
# condition A
groupA_t = combined['pre_bad'][(combined['condition'] == 'A') & (co
mbined['source'] == 'turk')]
groupA_i = combined['pre_bad'][(combined['condition'] == 'A') & (co
mbined['source'] == 'irl')]
groupA_im = combined['post_bad'][(combined['condition'] == 'A') &
(combined['source'] == 'irl')]

# condition B
groupB_t = combined['pre_bad'][(combined['condition'] == 'B') & (co
mbined['source'] == 'turk')]
groupB_i = combined['pre_bad'][(combined['condition'] == 'B') & (co
mbined['source'] == 'irl')]
groupB_im = combined['post_bad'][(combined['condition'] == 'B') &
(combined['source'] == 'irl')]

condA_means = [np.mean(x) for x in (groupA_t, groupA_i, groupA_im)]
condA_stdev = [np.std(x)/math.sqrt(len(x)) for x in (groupA_t, grou
pA_i, groupA_im)]

condB_means = [np.mean(x) for x in (groupB_t, groupB_i, groupB_im)]
condB_stdev = [np.std(x)/math.sqrt(len(x)) for x in (groupB_t, grou
pB_i, groupB_im)]
```

In [62]:

```
condA
```

Out[62]:

```
[1.8333333333333333, 2.96, 3.3999999999999999]
```

In [63]:

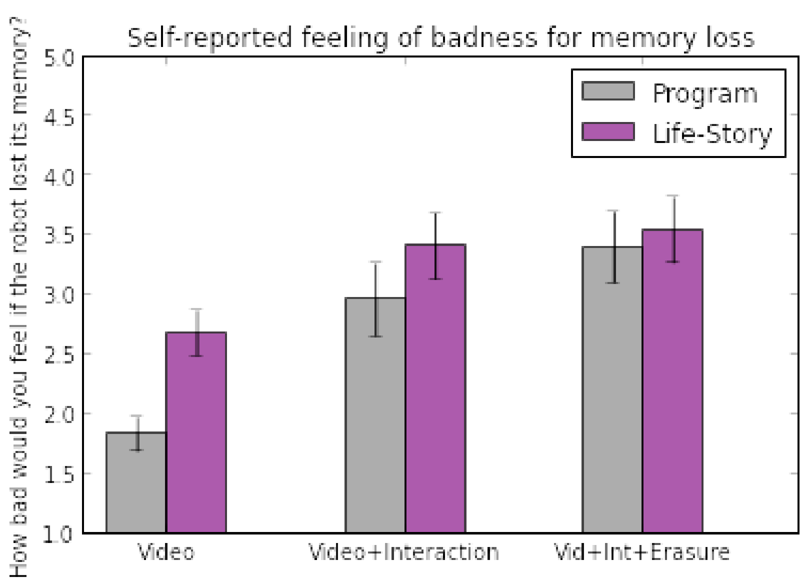
```
condB
```

Out[63]:

```
[2.6833333333333331, 3.4090909090909092, 3.5454545454545454]
```

In [298]:

```
# bar chart of above
ind = np.arange(3)
width = 0.25
fig, ax = plt.subplots()
rects1 = ax.bar(ind + 0.1, condA_means, width, color='gray', alpha=
0.8, yerr=condA_stdev, ecolor='k')
rects2 = ax.bar(ind + width + 0.1, condB_means, width, color='purpl
e', alpha=0.8, yerr=condB_stdev, ecolor='k')

# add some text for labels, title and axes ticks
ax.set_ylabel('How bad would you feel if the robot lost its memor
y?')
ax.set_title('Self-reported feeling of badness for memory loss')
ax.set_xticks(ind + width + 0.1)
ax.set_xticklabels(('Video', 'Video+Interaction', 'Vid+Int+Erasur
e'))
ax.set_ylim([1.0, 5.0])

ax.legend((rects1[0], rects2[0]), ('Program', 'Life-Story'))

plt.show()
```



In [67]:
```
for (A, B) in zip((groupA_t, groupA_i, groupA_im), (groupB_t, group
B_i, groupB_im)):
    print
    print 'A:', np.mean(A), np.std(A)/math.sqrt(len(A)), len(A)
    print 'B:', np.mean(B), np.std(B)/math.sqrt(len(B)), len(B)
    print 'MW-U', mannwhitneyu(A, B)
```

```
A: 1.83333333333 0.145614915801 60
B: 2.68333333333 0.190746583873 60
MW-U (1218.0, 0.00063493363709897911)

A: 2.96 0.307141661127 25
B: 3.40909090909 0.277675496655 22
MW-U (229.5, 0.16334141368796795)

A: 3.4 0.299332590942 25
B: 3.54545454545 0.278182358342 22
MW-U (265.0, 0.41741573670875343)
```

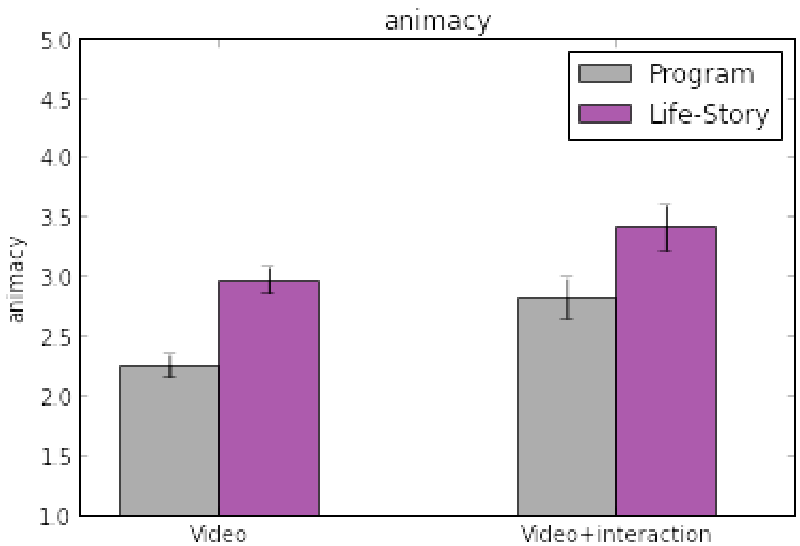
Godspeed

In [70]:
```
d = combined
for metric in ('anthropomorphic', 'animacy', 'likeability', 'intell
igence'):
    group1 = d[metric][d['condition'] == 'A']
    group2 = d[metric][d['condition'] == 'B']
    print '>>>>>', metric
    print 'shapiro', shapiro(d[metric])
    print 'A:', np.mean(group1), np.median(group1), len(group1)
    print 'B:', np.mean(group2), np.median(group2), len(group2)
    print 'MW-U', mannwhitneyu(group1, group2)
```

```
>>>>> anthropomorphic
shapiro (0.9245650172233582, 1.2191345888368232e-07)
A: 1.89882352941 1.8 85
B: 2.60243902439 2.4 82
MW-U (2121.0, 5.9528827758279519e-06)
>>>>> animacy
shapiro (0.9799643754959106, 0.016177764162421227)
A: 2.42470588228 2.333333333 85
B: 3.09349593498 3.0 82
MW-U (2086.5, 3.6548094694948634e-06)
>>>>> likeability
shapiro (0.9405418634414673, 1.920487875395338e-06)
A: 3.48529411765 3.4 85
B: 4.01219512195 3.8 82
MW-U (2054.5, 2.1080677233249681e-06)
>>>>> intelligence
shapiro (0.9870237708091736, 0.1251356154680252)
A: 2.69976470589 2.733333333 85
B: 3.24146341466 3.0833333335 82
MW-U (2251.0, 3.9170197146365035e-05)
```

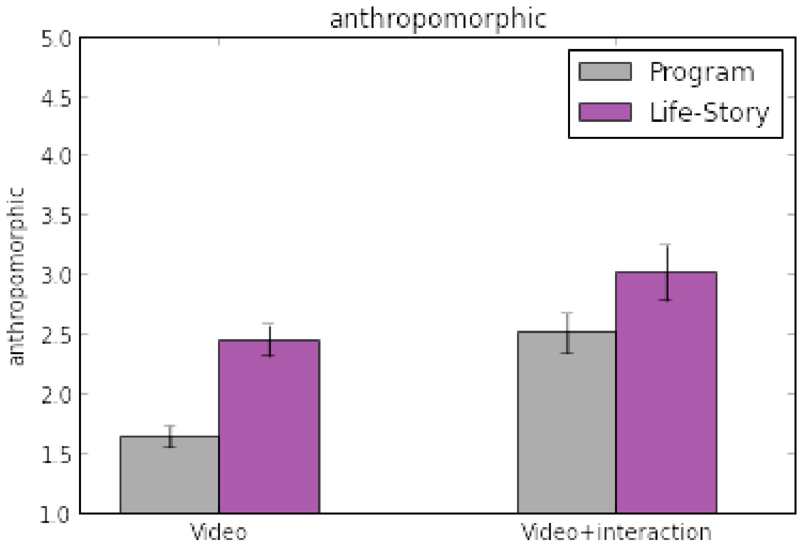
In [289]:
```
def barchart_turk_irl_measure(measure, dataset, filename=None):
    # condition A
    groupA_t = dataset[measure][(dataset['condition'] == 'A') & (da
taset['source'] == 'turk')]
    groupA_i = dataset[measure][(dataset['condition'] == 'A') & (da
taset['source'] == 'irl')]


    # condition B
    groupB_t = dataset[measure][(dataset['condition'] == 'B') & (da
taset['source'] == 'turk')]
    groupB_i = dataset[measure][(dataset['condition'] == 'B') & (da
taset['source'] == 'irl')]


    condA_means = [np.mean(x) for x in (groupA_t, groupA_i)]
    condA_stdev = [np.std(x)/math.sqrt(len(x)) for x in (groupA_t,
groupA_i)]

    condB_means = [np.mean(x) for x in (groupB_t, groupB_i)]
    condB_stdev = [np.std(x)/math.sqrt(len(x)) for x in (groupB_t,
groupB_i)]

    # bar chart of above
    ind = np.arange(2)
    width = 0.25
    fig, ax = plt.subplots()
    rects1 = ax.bar(ind + 0.1, condA_means, width, color='gray', al
pha=0.8, yerr=condA_stdev, ecolor='k')
    rects2 = ax.bar(ind + width + 0.1, condB_means, width, color='p
urple', alpha=0.8, yerr=condB_stdev, ecolor='k')

    # add some text for labels, title and axes ticks
    ax.set_ylabel(measure)
    ax.set_title(measure)
    ax.set_xticks(ind + width + 0.1)
    ax.set_xticklabels(('Video', 'Video+interaction'))
    ax.set_ylim([1.0, 5.0])

    ax.legend((rects1[0], rects2[0]), ('Program', 'Life-Story'))

    plt.show()
    for (A, B) in zip((groupA_t, groupA_i), (groupB_t, groupB_i)):
        print
        print 'A:', np.mean(A), np.std(A)/math.sqrt(len(A)), len(A)
        print 'B:', np.mean(B), np.std(B)/math.sqrt(len(B)), len(B)
        print 'MW-U', mannwhitneyu(A, B)
```

```
In [290]: barchart_turk_irl_measure('animacy', combined)
```



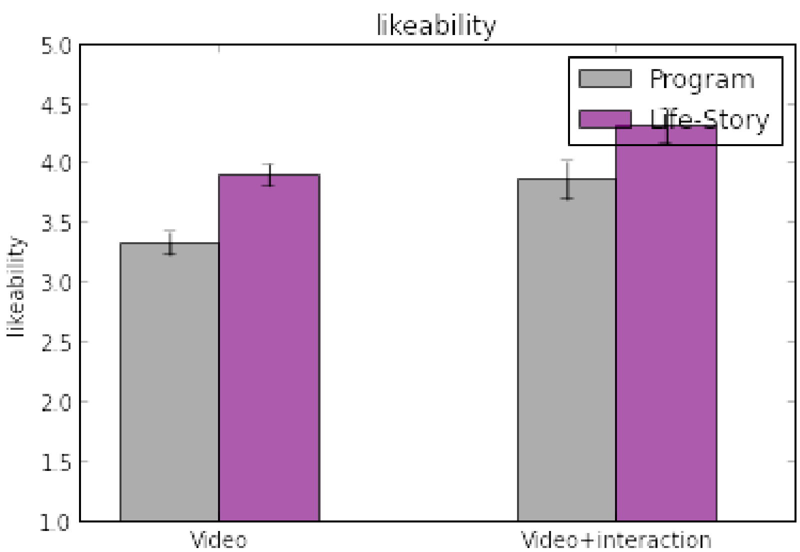

```
A: 2.25833333327 0.0940699273649 60
B: 2.975 0.113163339288 60
MW-U (1002.0, 1.3485425629042308e-05)

A: 2.82399999992 0.172031211622 25
B: 3.41666666673 0.192772888635 22
MW-U (184.5, 0.027174663014348528)
```

In [291]:
```
barchart_turk_irl_measure('anthropomorphic', combined)
```



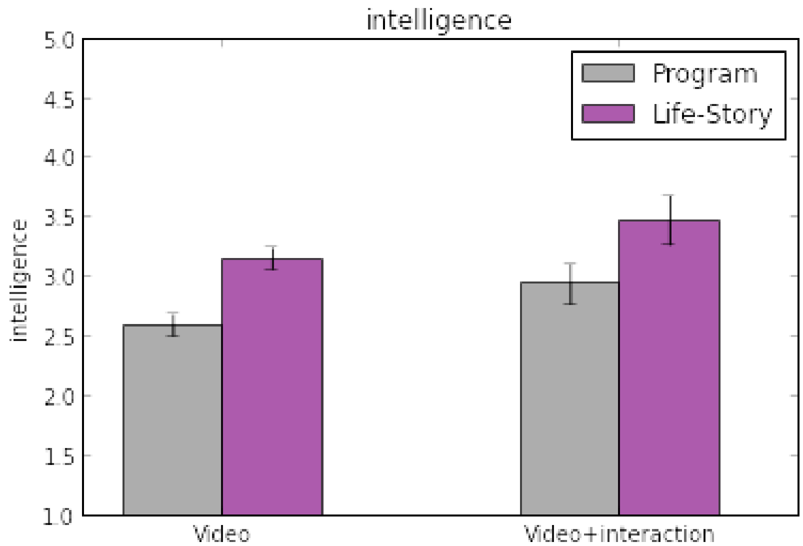

```
A: 1.64333333333 0.0923870681988 60
B: 2.45 0.133551904184 60
MW-U (937.0, 2.6583007468329544e-06)

A: 2.512 0.169735794693 25
B: 3.01818181818 0.234224265954 22
MW-U (220.0, 0.12183551694810446)
```

In [292]:
```
barchart_turk_irl_measure('likeability', combined)
```




```
A: 3.33 0.0989753055621 60
B: 3.90333333333 0.0948087744025 60
MW-U (917.0, 1.6085215175977795e-06)

A: 3.858 0.162571338187 25
B: 4.30909090909 0.145867183292 22
MW-U (185.5, 0.02746930454177943)
```

In [293]:
```
barchart_turk_irl_measure('intelligence', combined)
```

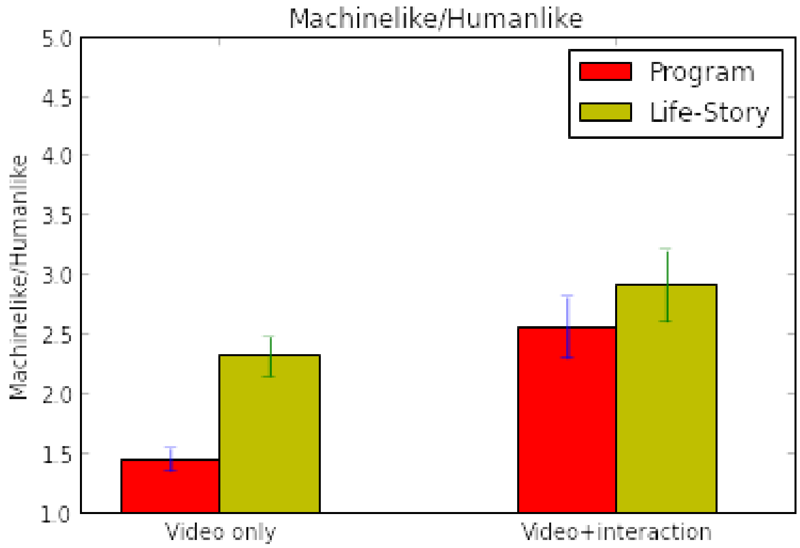



```
A: 2.59888888893 0.093995556326 60
B: 3.1555555556 0.0941618866882 60
MW-U (1060.5, 5.2291084812735744e-05)

A: 2.9418666666 0.170517234591 25
B: 3.47575757573 0.209652099539 22
MW-U (206.0, 0.07201708017642526)
```

In [97]:
```
ax.boxplot
```

Out[97]:
```
<bound method AxesSubplot.boxplot of <matplotlib.axes.AxesSubplot o
bject at 0x679a4d0>>
```

In [102]:
```
barchart_turk_irl_measure('Machinelike/Humanlike', combined)
```

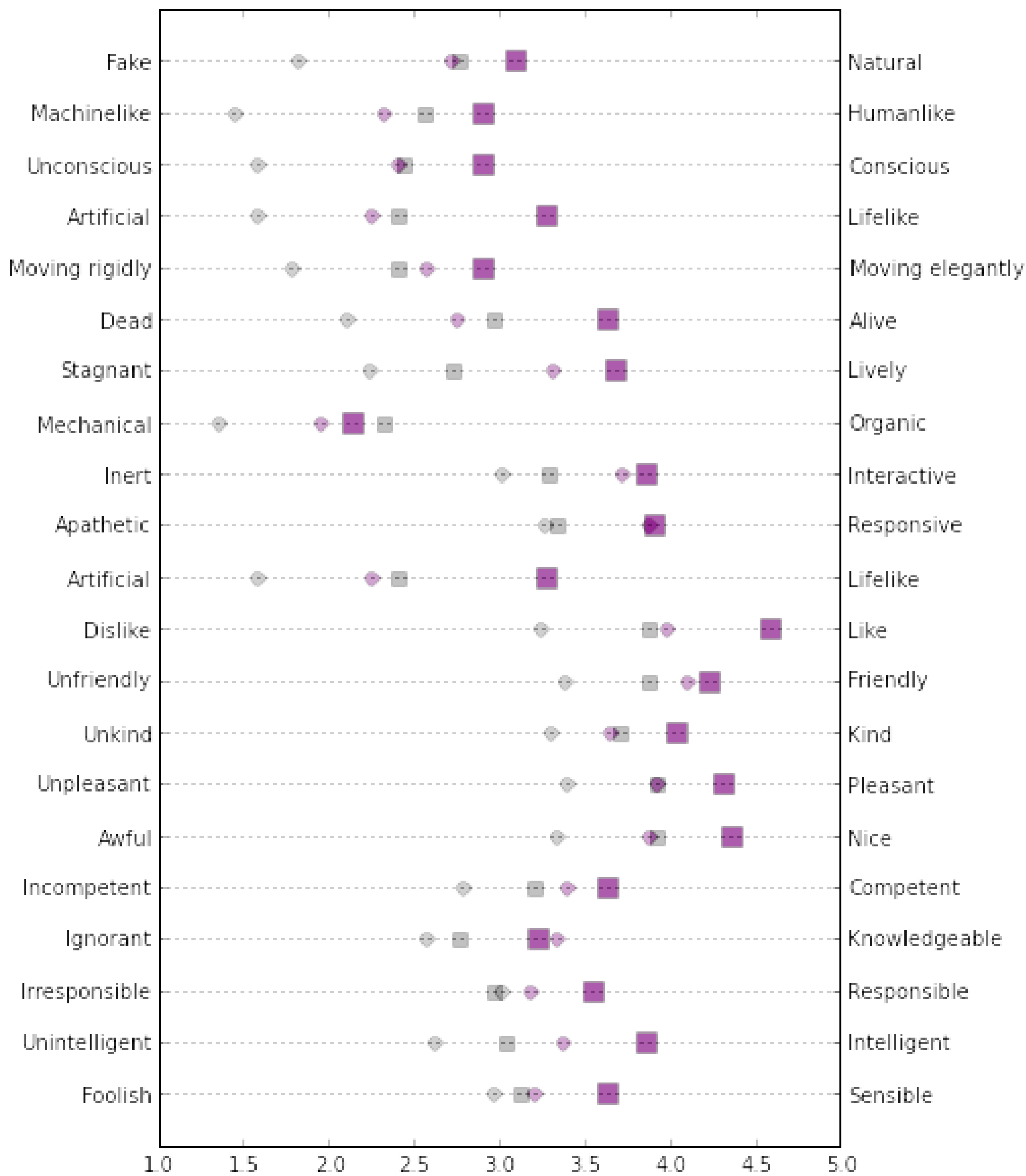



```
A: 1.45 0.0983898142876 60
B: 2.31666666667 0.165817280072 60
MW-U (1100.5, 3.1288922830395992e-05)

A: 2.56 0.259722929292 25
B: 2.90909090909 0.30088774237 22
MW-U (237.0, 0.20669541003261604)
```

In [165]:

```
df = combined







gs_anthropomorphic = [u'Fake/Natural', u'Machinelike/Humanlike',
u'Unconscious/Conscious',
              u'Artificial/Lifelike', u'Moving rigidly/Moving elega
ntly',]
gs_animacy = [u'Dead/Alive',u'Stagnant/Lively', u'Mechanical/Organi
c', u'Inert/Interactive',
              u'Apathetic/Responsive', u'Artificial/Lifelike']
gs_likeability = [u'Dislike/Like', u'Unfriendly/Friendly', u'Unkin
d/Kind',
                  u'Unpleasant/Pleasant', u'Awful/Nice']
gs_intelligence = [u'Incompetent/Competent', u'Ignorant/Knowledgeab
le', u'Irresponsible/Responsible',
                   u'Unintelligent/Intelligent', u'Foolish/Sensibl
e']

gs_columns = gs_anthropomorphic + gs_animacy + gs_likeability + gs_
intelligence
gs_columns.reverse()
gs_left = [x.split('/')[0] for x in gs_columns]
gs_right = [x.split('/')[1] for x in gs_columns]

indices = np.arange(len(gs_columns))

meansB_t = [np.mean(df[x][(df['condition'] == 'B') & (df['source']
== 'turk')]) for x in gs_columns]
meansB_i = [np.mean(df[x][(df['condition'] == 'B') & (df['source']
== 'irl')]) for x in gs_columns]
meansA_t = [np.mean(df[x][(df['condition'] == 'A') & (df['source']
== 'turk')]) for x in gs_columns]
meansA_i = [np.mean(df[x][(df['condition'] == 'A') & (df['source']
== 'irl')]) for x in gs_columns]



fig, ax = plt.subplots(figsize=(6,10))
ax2 = ax.twinx()

rects = ax.plot(meansA_i, indices, marker='s', color='gray', linewi
dth=0, ms=7.0, alpha=0.7, label='Program - Interaction')
rects = ax.plot(meansB_t, indices, marker='o', color='purple', line
width=0, ms=7.0, alpha=0.6, label='Life-Story - Video')
rects = ax.plot(meansA_t, indices, marker='o', color='gray', linewi
dth=0, ms=7.0, alpha=0.6, label='Program - Video',)

rects = ax2.plot(meansB_i, indices, marker='s', color='purple', lin
ewidth=0, ms=10.0, alpha=0.8, label='Life-Story - Interaction')
```

```
ax.yaxis.grid(True)
ax.set_xlim([1.0, 5.0])
ax.set_ylim([-1, 21])
ax.set_yticks(indices)
ax.set_yticklabels(gs_left,  ha='right')
#ax.yaxis.set_visible(False)


ax2.yaxis.grid(True)
ax2.set_xlim([1.0, 5.0])
ax2.set_ylim([-1, 21])
ax2.set_yticks(indices)
ax2.set_yticklabels(gs_right, ha='left')

#plt.legend()

plt.margins(0.1)
plt.show()
```



Empathy

In [182]:
```
reload(misc)
```

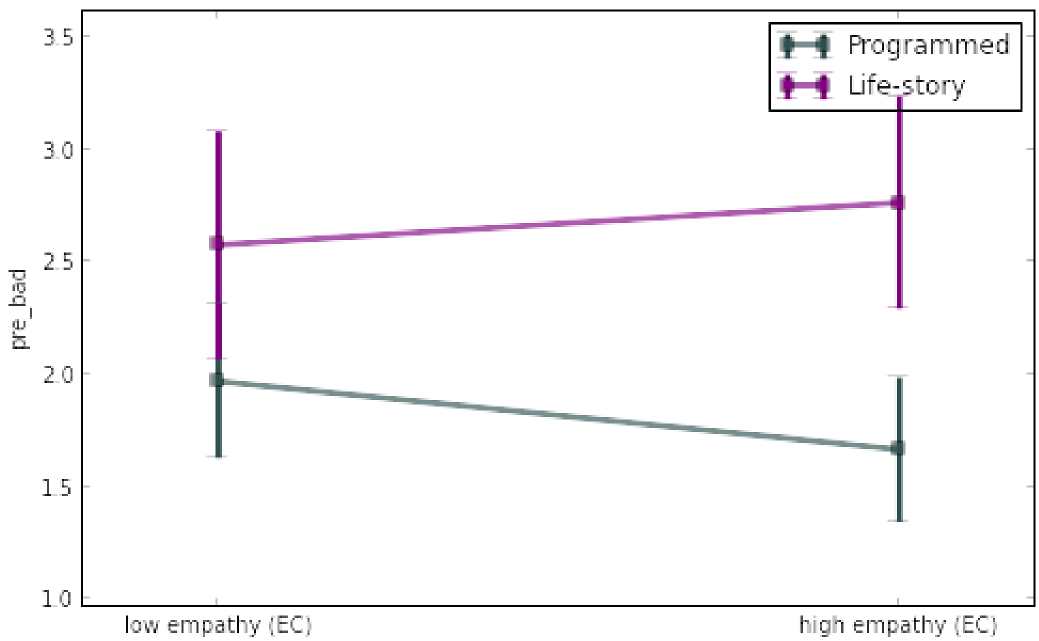
Out[182]:
```
<module 'misc' from 'misc.py'>
```

In [301]:
```
def plot_interaction(cat, df, measure='pre_bad'):
    print 'analyzing' + cat
    misc.pub_my_interaction_plot(df, 'high_' + cat, 'condition', me
asure,
                                 x_levels=['N', 'Y'],
                                x_labels = ['low empathy (' + cat +
')', 'high empathy (' + cat +')'],
                                 trace_labels = ['Programmed', 'Lif
e-story']
                                )

    group0 = d[measure][(d['condition'] == 'A') & (d['high_' + cat]
== 'Y')]
    group1 = d[measure][(d['condition'] == 'B') & (d['high_' + cat]
== 'Y')]
    group2 = d[measure][(d['condition'] == 'A') & (d['high_' + cat]
== 'N')]
    group3 = d[measure][(d['condition'] == 'B') & (d['high_' + cat]
== 'N')]

    print 'KW:', kw_test(group0, group1, group2, group3)
    # compare=((0,1), (2,3), (0,2), (1,3)) - compared b and d with
empathy high and low, then high and low empathy for each of b and d
    print 'Nemenyi', kw_nemenyi((group0, group1, group2, group3), t
o_compare=((0,1), (2,3), (0,2), (1,3)))

    formula = measure + ' ~ C(condition)*' + cat
    lm = ols(formula, df).fit()
    print lm.summary()
    #fig, ax = plt.subplots()
    #fig = statsmodels.api.graphics.plot_fit(lm, 2, ax=ax)
    print anova_lm(lm)
    print '> normality of residuals', shapiro(lm.resid)
```

In [302]:
```
plot_interaction('EC', turk_all)
```

```
analyzingEC
got x_levels ['N', 'Y']
got trace_levels set(['A', 'B'])
0 N A 1.9696969697 1.21816674195 33 0.342880231428
1 Y A 1.66666666667 0.981306762925 27 0.32075014955
0 N B 2.57692307692 1.59742766595 26 0.505376194395
1 Y B 2.76470588235 1.37324912117 34 0.474143147159
KW: (3.5863632130693981, 0.30973263005428181)
Nemenyi (3.5863632130693981, 0.30973263005428181, array([ 0.4201236
4,  0.9       ,  0.9       ,  0.39767015]), array([False, False, Fa
lse, False], dtype=bool))
                            OLS Regression Results
====================================================================
===========
Dep. Variable:                pre_bad   R-squared:
0.162
Model:                            OLS   Adj. R-squared:
0.140
Method:                 Least Squares   F-statistic:
7.466
Date:                Wed, 20 Apr 2016   Prob (F-statistic):
0.000130
Time:                        22:23:07   Log-Likelihood:
-198.45
No. Observations:                 120   AIC:
404.9
Df Residuals:                     116   BIC:
416.1
Df Model:                           3
Covariance Type:            nonrobust
====================================================================
=====================
                           coef    std err          t      P>|t|
[95.0% Conf. Int.]
--------------------------------------------------------------------
---------------------
Intercept                3.8012      0.947      4.015      0.000
1.926     5.676
C(condition)[T.B]       -2.6268      1.181     -2.225      0.028
-4.965    -0.288
EC                      -0.0735      0.035     -2.111      0.037
-0.142    -0.005
C(condition)[T.B]:EC     0.1295      0.043      3.003      0.003
0.044     0.215
====================================================================
===========
Omnibus:                       11.178   Durbin-Watson:
2.161
Prob(Omnibus):                  0.004   Jarque-Bera (JB):
7.471
Skew:                           0.472   Prob(JB):
0.0239
Kurtosis:                       2.224   Cond. No.
388.
```

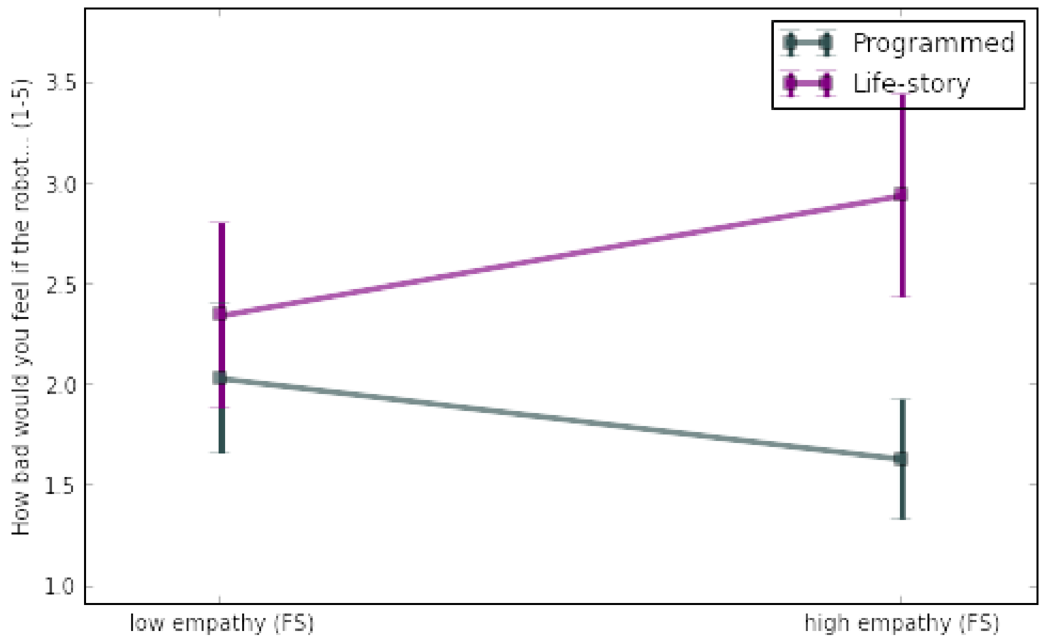
```
==================================================================
===========

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors
is correctly specified.
                  df      sum_sq    mean_sq          F    PR(>F)
C(condition)       1   21.675000  21.675000  13.100040  0.000439
EC                 1    0.465538   0.465538   0.281364  0.596823
C(condition):EC    1   14.920423  14.920423   9.017677  0.003276
Residual         116  191.930706   1.654575        NaN       NaN
> normality of residuals (0.9491804838180542, 0.0001882250944618135
7)
```



In [193]:
```
plot_interaction('FS', turk_all)
```

```
analyzingFS
got x_levels ['N', 'Y']
got trace_levels set(['A', 'B'])
0 N A 2.03333333333 1.22429117815 30 0.371234177865
1 Y A 1.63333333333 0.982626864866 30 0.298204503531
0 N B 2.34615384615 1.4660860425 26 0.460118624747
1 Y B 2.94117647059 1.43365383611 34 0.50440760336
KW: (12.429726841154752, 0.00604713196687262)
Nemenyi (12.429726841154752, 0.00604713196687262, array([ 0.0156046
6,  0.63007325,  0.9       ,  0.45368808]), array([ True, False, Fa
lse, False], dtype=bool))
                            OLS Regression Results
====================================================================
===========
Dep. Variable:                pre_bad   R-squared:
0.126
Model:                            OLS   Adj. R-squared:
0.103
Method:                 Least Squares   F-statistic:
5.552
Date:                Wed, 20 Apr 2016   Prob (F-statistic):
0.00135
Time:                        17:35:27   Log-Likelihood:
-200.99
No. Observations:                 120   AIC:
410.0
Df Residuals:                     116   BIC:
421.1
Df Model:                           3
Covariance Type:            nonrobust
====================================================================
=====================
                           coef    std err          t      P>|t|
[95.0% Conf. Int.]
--------------------------------------------------------------------
---------------------
Intercept                2.0568      0.694      2.962      0.004
0.682     3.432
C(condition)[T.B]       -0.7355      0.988     -0.745      0.458
-2.692     1.221
FS                      -0.0095      0.029     -0.332      0.741
-0.066     0.047
C(condition)[T.B]:FS     0.0644      0.040      1.622      0.107
-0.014     0.143
====================================================================
===========
Omnibus:                       11.618   Durbin-Watson:
2.234
Prob(Omnibus):                  0.003   Jarque-Bera (JB):
7.571
Skew:                           0.471   Prob(JB):
0.0227
Kurtosis:                       2.209   Cond. No.
272.
```

```
================================================================================
===========

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors
is correctly specified.
                     df       sum_sq    mean_sq          F    PR(>F)
C(condition)          1    21.675000  21.675000  12.556575  0.000570
FS                    1     2.536175   2.536175   1.469235  0.227929
C(condition):FS       1     4.542769   4.542769   2.631678  0.107466
Residual            116   200.237722   1.726187        NaN       NaN
> normality of residuals (0.9392876029014587, 3.833742448478006e-0
5)
```

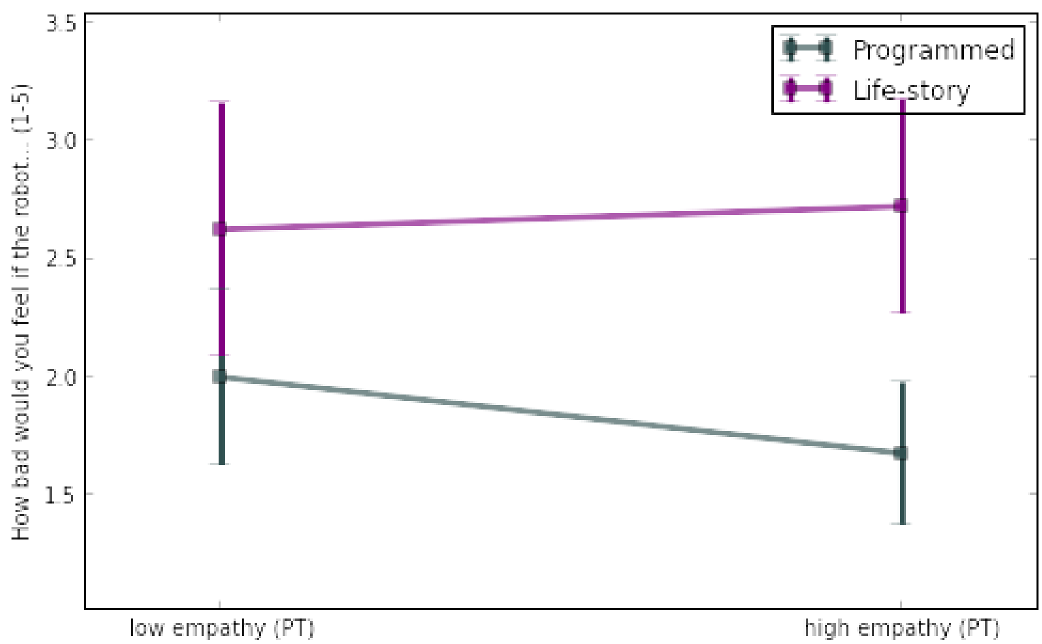


In [194]:

```
plot_interaction('PT', turk_all)
```

```
analyzingPT
got x_levels ['N', 'Y']
got trace_levels set(['A', 'B'])
0 N A 2.0 1.3130643286 29 0.371390676354
1 Y A 1.67741935484 0.893961707132 31 0.301273409851
0 N B 2.625 1.60240704359 24 0.535825881234
1 Y B 2.72222222222 1.38666488604 36 0.453703703704
KW: (10.921022477198598, 0.012160693101594067)
Nemenyi (10.921022477198598, 0.012160693101594067, array([ 0.021497
91,  0.51118264,  0.9       ,  0.89385319]), array([ True, False, F
alse, False], dtype=bool))
                            OLS Regression Results
===================================================================
===========
Dep. Variable:                 pre_bad   R-squared:
0.103
Model:                             OLS   Adj. R-squared:
0.079
Method:                  Least Squares   F-statistic:
4.421
Date:                 Wed, 20 Apr 2016   Prob (F-statistic):
0.00556
Time:                         20:23:13   Log-Likelihood:
-202.55
No. Observations:                  120   AIC:
413.1
Df Residuals:                      116   BIC:
424.2
Df Model:                            3
Covariance Type:            nonrobust
===================================================================
=====================
                           coef    std err          t      P>|t|
[95.0% Conf. Int.]
-------------------------------------------------------------------
---------------------
Intercept                2.7791      0.949      2.929      0.004
0.900     4.658
C(condition)[T.B]       -0.0769      1.299     -0.059      0.953
-2.650     2.496
PT                      -0.0357      0.035     -1.014      0.313
-0.105     0.034
C(condition)[T.B]:PT     0.0350      0.048      0.735      0.464
-0.059     0.129
===================================================================
===========
Omnibus:                        20.128   Durbin-Watson:
2.268
Prob(Omnibus):                   0.000   Jarque-Bera (JB):
8.883
Skew:                            0.455   Prob(JB):
0.0118
Kurtosis:                        2.026   Cond. No.
391.
```

```
==============================================================================
===========

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors
is correctly specified.
                    df      sum_sq    mean_sq          F    PR(>F)
C(condition)         1   21.675000  21.675000  12.235294  0.000666
PT                   1    0.862673   0.862673   0.486969  0.486679
C(condition):PT      1    0.958313   0.958313   0.540957  0.463521
Residual           116  205.495680   1.771514        NaN       NaN
> normality of residuals (0.9137037992477417, 1.0561462886471418e-0
6)
```

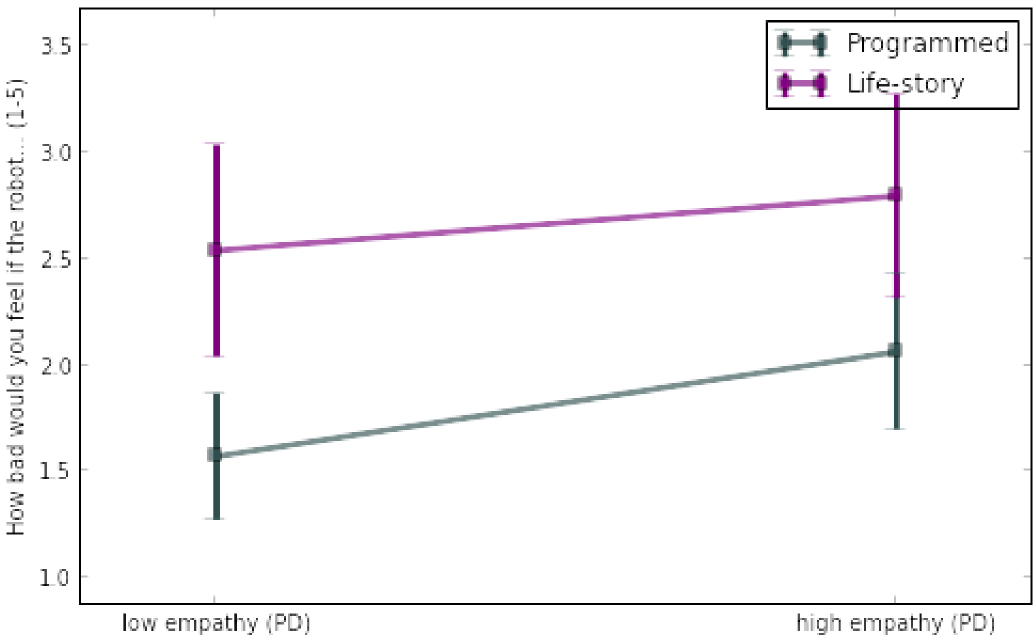


In [195]:

```
plot_interaction('PD', turk_all)
```

```
analyzingPD
got x_levels ['N', 'Y']
got trace_levels set(['A', 'B'])
0 N A 1.57142857143 0.979379228629 28 0.296972085936
1 Y A 2.0625 1.19732775379 32 0.364601934049
0 N B 2.53846153846 1.36524918072 26 0.49783326612
1 Y B 2.79411764706 1.54880241714 34 0.479187223192
KW: (12.545539118883154, 0.0057299560143549881)
Nemenyi (12.545539118883154, 0.0057299560143549881, array([ 0.21961
023,  0.08089563,  0.55071004,  0.85734729]), array([False, False,
False, False], dtype=bool))
                            OLS Regression Results
==============================================================================
===========
Dep. Variable:                pre_bad   R-squared:
0.105
Model:                            OLS   Adj. R-squared:
0.082
Method:                 Least Squares   F-statistic:
4.551
Date:                Wed, 20 Apr 2016   Prob (F-statistic):
0.00472
Time:                        20:24:01   Log-Likelihood:
-202.37
No. Observations:                 120   AIC:
412.7
Df Residuals:                     116   BIC:
423.9
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
=====================
                           coef    std err          t      P>|t|
[95.0% Conf. Int.]
------------------------------------------------------------------------------
---------------------
Intercept                1.4081      0.507      2.776      0.006
0.403     2.413
C(condition)[T.B]        0.9223      0.706      1.306      0.194
-0.476     2.321
PD                       0.0242      0.027      0.891      0.375
-0.030     0.078
C(condition)[T.B]:PD    -0.0043      0.038     -0.114      0.909
-0.079     0.070
==============================================================================
===========
Omnibus:                       17.933   Durbin-Watson:
2.288
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
8.677
Skew:                           0.463   Prob(JB):
0.0131
Kurtosis:                       2.062   Cond. No.
145.
```

```
================================================================================
===========

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors
is correctly specified.
                  df      sum_sq    mean_sq          F    PR(>F)
C(condition)       1   21.675000  21.675000  12.272295  0.000654
PD                 1    2.417643   2.417643   1.368859  0.244406
C(condition):PD    1    0.022923   0.022923   0.012979  0.909494
Residual         116  204.876100   1.766173        NaN       NaN
> normality of residuals (0.9296949505805969, 9.210341886500828e-0
6)
```

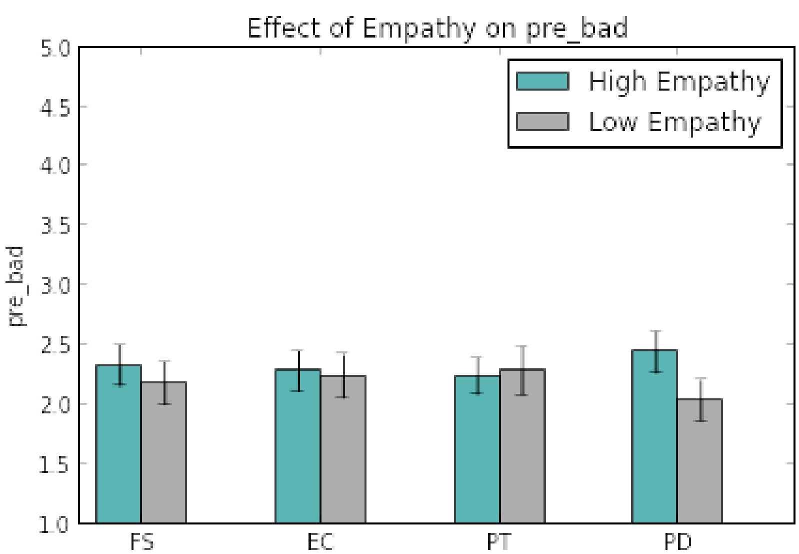


In [303]:

```
def barchart_empathy(dataset, measure='pre_bad', filename=None):
    # condition A
    means_high = []
    stdev_high = []
    means_low = []
    stdev_low = []

    empathy_cats = ('FS', 'EC', 'PT', 'PD')
    for cat in empathy_cats:
        group_high = dataset[measure][dataset['high_' + cat] ==
'Y']
        group_low = dataset[measure][dataset['high_' + cat] == 'N']

        means_high.append(np.mean(group_high))
        stdev_high.append(np.std(group_high)/math.sqrt(len(group_hi
gh)))

        means_low.append(np.mean(group_low))
        stdev_low.append(np.std(group_low)/math.sqrt(len(group_lo
w)))

        A = group_high
        B = group_low
        print '>>>>> ' + cat
        print 'A:', np.mean(A), np.std(A)/math.sqrt(len(A)), len(A)
        print 'B:', np.mean(B), np.std(B)/math.sqrt(len(B)), len(B)
        print 'MW-U', mannwhitneyu(A, B)


    # bar chart of above
    ind = np.arange(len(means_high))
    width = 0.25
    fig, ax = plt.subplots()
    rects1 = ax.bar(ind + 0.1, means_high, width, color='darkcyan',
alpha=0.8, yerr=stdev_high, ecolor='k')
    rects2 = ax.bar(ind + width + 0.1, means_low, width, color='gra
y', alpha=0.8, yerr=stdev_low, ecolor='k')

    # add some text for labels, title and axes ticks
    ax.set_ylabel(measure)
    ax.set_title('Effect of Empathy on ' + measure)
    ax.set_xticks(ind + width + 0.1)
    ax.set_xticklabels(empathy_cats)
    ax.set_ylim([1.0, 5.0])

    ax.legend((rects1[0], rects2[0]), ('High Empathy', 'Low Empath
y'))

    plt.show()
```

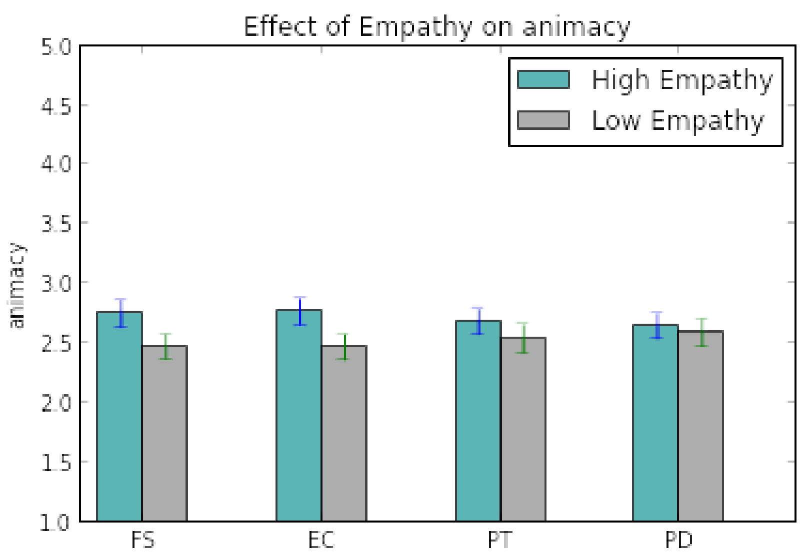
In [304]:
```
barchart_empathy(turk_all)
```

```
>>>>> FS
A: 2.328125 0.175466297707 64
B: 2.17857142857 0.180537644573 56
MW-U (1694.5, 0.29502009871863621)
>>>>> EC
A: 2.27868852459 0.170572407713 61
B: 2.23728813559 0.186192412595 59
MW-U (1727.0, 0.34491592669198623)
>>>>> PT
A: 2.23880597015 0.158081548585 67
B: 2.28301886792 0.203873767286 53
MW-U (1734.5, 0.41060208818063981)
>>>>> PD
A: 2.43939393939 0.176862758109 66
B: 2.03703703704 0.173645972723 54
MW-U (1503.5, 0.060756693366021616)
```



In [231]:
```
barchart_empathy(turk_all, measure='animacy')
```

```
>>>>> FS
A: 2.74479166661 0.115289534827 64
B: 2.47023809523 0.108139146249 56
MW-U (1489.0, 0.055271291275820318)
>>>>> EC
A: 2.76229508193 0.112019126743 61
B: 2.46610169488 0.112483283581 59
MW-U (1411.0, 0.020549885713333087)
>>>>> PT
A: 2.67910447758 0.107531385408 67
B: 2.53773584902 0.120620463028 53
MW-U (1593.5, 0.16805140672284158)
>>>>> PD
A: 2.64141414141 0.108629450043 66
B: 2.58641975302 0.119834255675 54
MW-U (1734.5, 0.40182318172590 27)
```



hmm... I wonder what the empathy interaction looks like with GS measure

In [218]:
```
reload(misc)
```

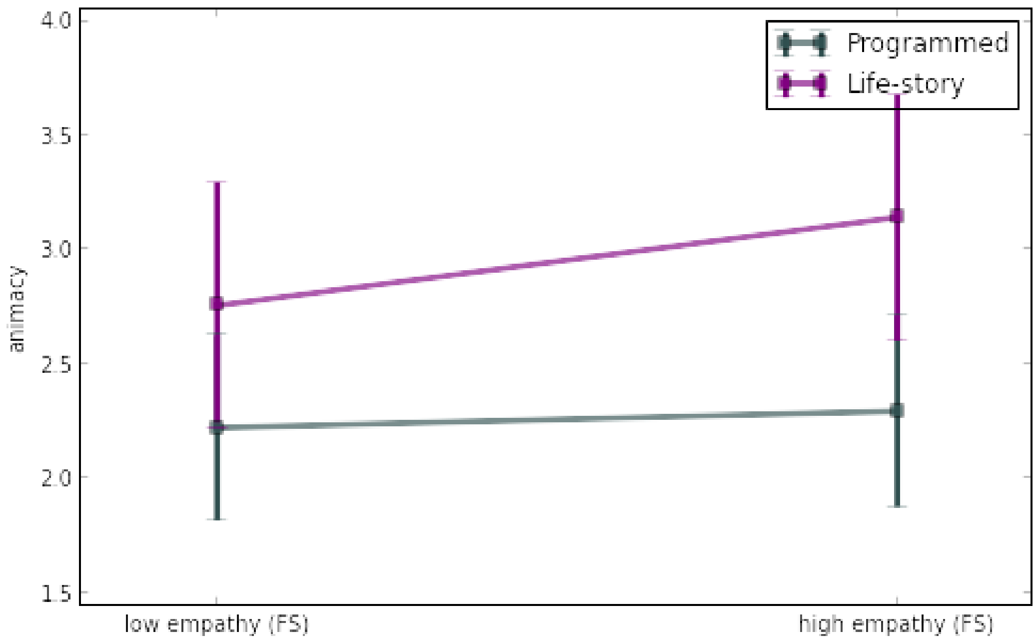
Out[218]: <module 'misc' from 'misc.py'>

In [221]:

```
plot_interaction('FS', turk_all, measure='animacy')
```

```
analyzingFS
got x_levels ['N', 'Y']
got trace_levels set(['A', 'B'])
0 N A 2.22222222217 0.721794745251 30 0.405720412957
1 Y A 2.29444444437 0.733690990248 30 0.418906326374
0 N B 2.75641025646 0.810232067139 26 0.540576526352
1 Y B 3.14215686271 0.888499830823 34 0.53887545625
KW: (22.682176171908715, 4.70356227262776e-05)
Nemenyi (22.682176171908715, 4.70356227262776e-05, array([ 0.009097
41,  0.0133272 ,  0.58734303,  0.72344873]), array([ True,  True, F
alse, False], dtype=bool))
                            OLS Regression Results
====================================================================
===========
Dep. Variable:                animacy   R-squared:
0.212
Model:                            OLS   Adj. R-squared:
0.192
Method:                 Least Squares   F-statistic:
10.43
Date:                Wed, 20 Apr 2016   Prob (F-statistic):
3.95e-06
Time:                        20:45:04   Log-Likelihood:
-140.88
No. Observations:                 120   AIC:
289.8
Df Residuals:                     116   BIC:
300.9
Df Model:                           3
Covariance Type:            nonrobust
====================================================================
=====================
                           coef    std err          t      P>|t|
[95.0% Conf. Int.]
--------------------------------------------------------------------
---------------------
Intercept                1.8384      0.421      4.369      0.000
1.005     2.672
C(condition)[T.B]        0.1305      0.599      0.218      0.828
-1.055     1.316
FS                       0.0179      0.017      1.029      0.305
-0.017     0.052
C(condition)[T.B]:FS     0.0227      0.024      0.943      0.348
-0.025     0.070
====================================================================
===========
Omnibus:                        2.908   Durbin-Watson:
2.157
Prob(Omnibus):                  0.234   Jarque-Bera (JB):
2.546
Skew:                           0.257   Prob(JB):
0.280
Kurtosis:                       2.506   Cond. No.
272.
```

```
==============================================================================
===========

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors
is correctly specified.
                   df     sum_sq    mean_sq          F    PR(>F)
C(condition)        1  15.408333  15.408333  24.308929  0.000003
FS                  1   3.867956   3.867956   6.102274  0.014955
C(condition):FS     1   0.563212   0.563212   0.888551  0.347829
Residual          116  73.527165   0.633855        NaN       NaN
> normality of residuals (0.9835394620895386, 0.15140587091445923)
```

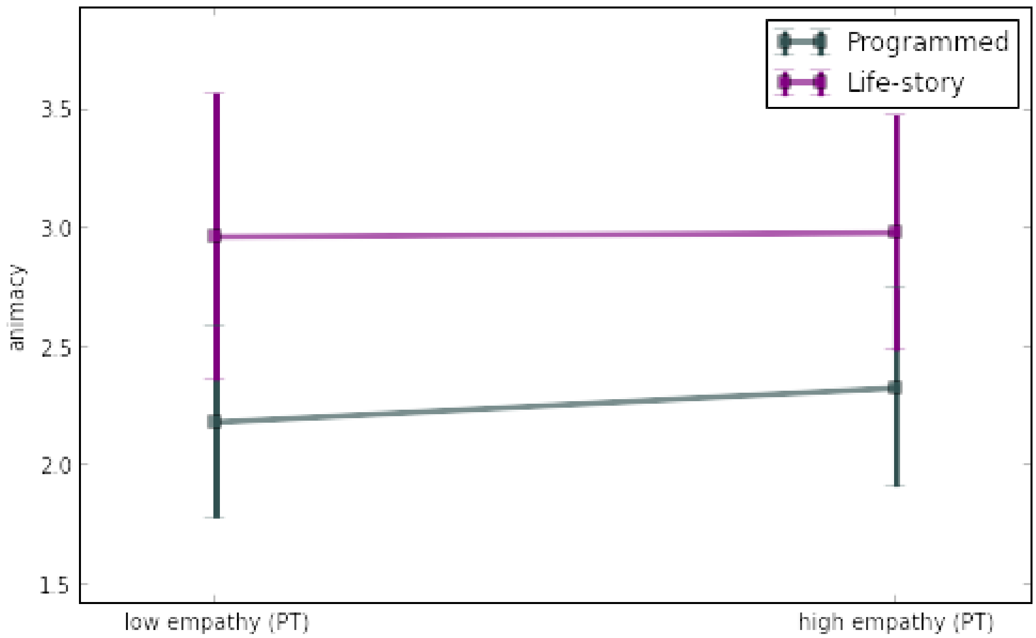


In [222]:

```
plot_interaction('PT', turk_all, measure='animacy')
```

```
analyzingPT
got x_levels ['N', 'Y']
got trace_levels set(['A', 'B'])
0 N A 2.1839080459 0.672880278301 29 0.40554154313
1 Y A 2.32795698919 0.770751008058 31 0.418113418149
0 N B 2.96527777779 0.906482781947 24 0.605284791767
1 Y B 2.98148148147 0.855968314776 36 0.496913580245
KW: (20.169222716645567, 0.00015657203986895908)
Nemenyi (20.169222716645567, 0.00015657203986895908, array([ 0.0052
6633,  0.01317092,  0.9       ,  0.9       ]), array([ True,  True,
False, False], dtype=bool))
                            OLS Regression Results
===================================================================
===========
Dep. Variable:                animacy   R-squared:
0.171
Model:                            OLS   Adj. R-squared:
0.150
Method:                 Least Squares   F-statistic:
7.994
Date:                Wed, 20 Apr 2016   Prob (F-statistic):
6.87e-05
Time:                        20:45:34   Log-Likelihood:
-143.94
No. Observations:                 120   AIC:
295.9
Df Residuals:                     116   BIC:
307.0
Df Model:                           3
Covariance Type:            nonrobust
===================================================================
=====================
                           coef    std err          t      P>|t|
[95.0% Conf. Int.]
-------------------------------------------------------------------
---------------------
Intercept                2.0281      0.582      3.484      0.001
0.875     3.181
C(condition)[T.B]        0.4940      0.797      0.620      0.537
-1.085     2.073
PT                       0.0087      0.022      0.402      0.688
-0.034     0.051
C(condition)[T.B]:PT     0.0080      0.029      0.273      0.786
-0.050     0.066
===================================================================
===========
Omnibus:                        2.261   Durbin-Watson:
2.253
Prob(Omnibus):                  0.323   Jarque-Bera (JB):
2.276
Skew:                           0.290   Prob(JB):
0.320
Kurtosis:                       2.654   Cond. No.
391.
```

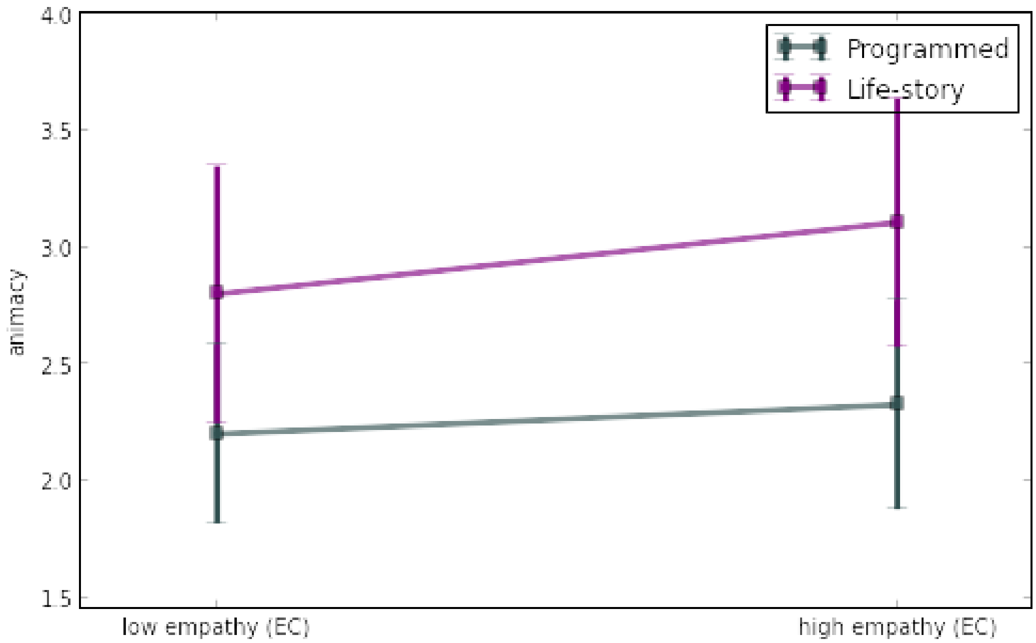
```
==================================================================
===========

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors
is correctly specified.
                  df     sum_sq    mean_sq          F    PR(>F)
C(condition)       1  15.408333  15.408333  23.101281  0.000005
PT                 1   0.537838   0.537838   0.806365  0.371057
C(condition):PT    1   0.049608   0.049608   0.074375  0.785555
Residual         116  77.370888   0.666990        NaN       NaN
> normality of residuals (0.9838168025016785, 0.16051313281059265)
```



In [224]:
```
plot_interaction('EC', turk_all, measure='animacy')
```

```
analyzingEC
got x_levels ['N', 'Y']
got trace_levels set(['A', 'B'])
0 N A 2.20202020191 0.66762251644 33 0.383322515115
1 Y A 2.32716049381 0.791520755229 27 0.44786224585
0 N B 2.80128205135 0.963011691958 26 0.549376609342
1 Y B 3.10784313721 0.778470883592 34 0.532990700876
KW: (22.947228903511327, 4.1418817647843138e-05)
Nemenyi (22.947228903511327, 4.1418817647843138e-05, array([ 0.0032
3218,  0.03877317,  0.78644469,  0.48617886]), array([ True,  True,
False, False], dtype=bool))
                            OLS Regression Results
====================================================================
===========
Dep. Variable:                animacy   R-squared:
0.195
Model:                            OLS   Adj. R-squared:
0.174
Method:                 Least Squares   F-statistic:
9.353
Date:                Wed, 20 Apr 2016   Prob (F-statistic):
1.38e-05
Time:                        20:46:21   Log-Likelihood:
-142.22
No. Observations:                 120   AIC:
292.4
Df Residuals:                     116   BIC:
303.6
Df Model:                           3
Covariance Type:            nonrobust
====================================================================
=====================
                           coef    std err          t      P>|t|
[95.0% Conf. Int.]
--------------------------------------------------------------------
---------------------
Intercept                2.6091      0.592      4.404      0.000
1.436     3.783
C(condition)[T.B]       -0.4842      0.739     -0.655      0.514
-1.948     0.979
EC                      -0.0131      0.022     -0.601      0.549
-0.056     0.030
C(condition)[T.B]:EC     0.0446      0.027      1.654      0.101
-0.009     0.098
====================================================================
===========
Omnibus:                        3.394   Durbin-Watson:
2.299
Prob(Omnibus):                  0.183   Jarque-Bera (JB):
3.424
Skew:                           0.392   Prob(JB):
0.180
Kurtosis:                       2.737   Cond. No.
388.
```

```
==============================================================================
===========

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
                    df     sum_sq    mean_sq          F    PR(>F)
C(condition)         1  15.408333  15.408333  23.773940  0.000003
EC                   1   1.003253   1.003253   1.547947  0.215947
C(condition):EC      1   1.773318   1.773318   2.736101  0.100807
Residual           116  75.181762   0.648119        NaN       NaN
> normality of residuals (0.981813907623291, 0.10496324300765991)
```



Pen Pickup

I want to see what fraction of people are picking up pens in each category

In [237]:
```
np.median(data_pen['post_bad'])
```

Out[237]: 4.0

In [239]:
```
len(data_pen['ID'][data_pen['post_bad'] < 4.0])
```

Out[239]: 16

In [240]:
```
data_pen['high_post_bad'] = ['Y' if x >= np.median(data_pen['post_bad']) else 'N' for x in data_pen['post_bad']]
```

In [268]:

```
data_pen['high_pre_bad'] = ['Y' if x >= np.median(data_pen['pre_ba
d']) else 'N' for x in data_pen['pre_bad']]
```

In [252]:

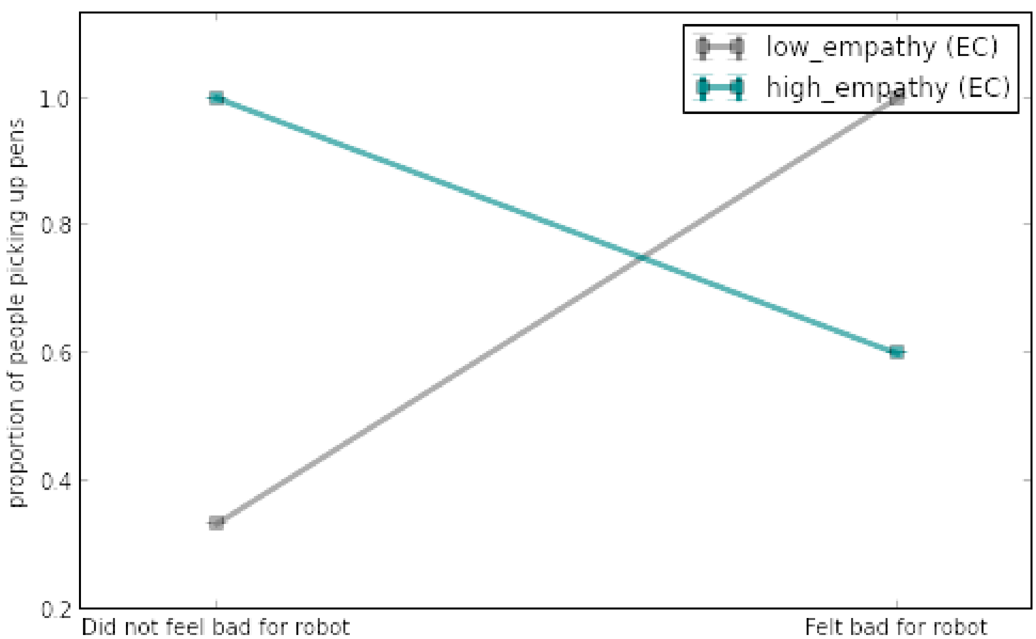
```
def pen_interaction_plot(df, x='high_post_bad', trace='high_EC',
                         x_levels=None, trace_levels=None, filename
=None,
                            x_labels = None, trace_labels = None):
    """
    For publication...

    does a statsmodel style interaction plot grouping the
    'response' column values by x, trace and plotting their mean,
    stdev of mean with a line per level of trace

    provide x_levels and trace_levels if you care about the order

    needs
        import matplotlib.pyplot as plt
        import numpy as np
        import math

    """
    colors = ['darkcyan', 'gray']
    if x_levels is None:
        x_levels = set(df[x])
    if trace_levels is None:
        trace_levels = set(df[trace])

    if x_labels is None:
        x_labels = x_levels
    if trace_labels is None:
        trace_labels = trace_levels

    colors = (colors*len(trace_levels))[:len(trace_levels)]
    print 'got x_levels', x_levels
    print 'got trace_levels', trace_levels

    fig, ax = plt.subplots(figsize=(8, 5)) # was 4,3
    plt.ylabel('proportion of people picking up pens')
    #plt.xlabel(x)

    for (trace_val, trace_label) in zip(trace_levels, trace_label
s):
        x_num_list = []
        y_list = []
        yerr_list = []
        for x_num, x_val in zip(range(len(x_levels)), x_levels):
            responses = df['pen_pickup'][(df[x] == x_val) & (df[tra
ce] == trace_val)]
            num_pickups = sum(responses == 'P')
            num_no_pickups = sum(responses == 'N')
            pickup_prop = float(num_pickups) /  float(num_pickups +
num_no_pickups)

            x_num_list.append(x_num)
            y_list.append(pickup_prop)
            yerr_list.append(0)
```

```
            print x_num, x_val, trace_val, pickup_prop, len(respons
es)

        plt.errorbar(x_num_list,y_list, yerr=yerr_list,
                     linewidth=3.0, alpha=0.80, capsize=5.0, marker
='s',
                     color=colors.pop(), label=trace_label)
    plt.legend()
    plt.xticks(range(len(x_levels)), list(x_labels))
    plt.margins(0.2)
```

In [308]:

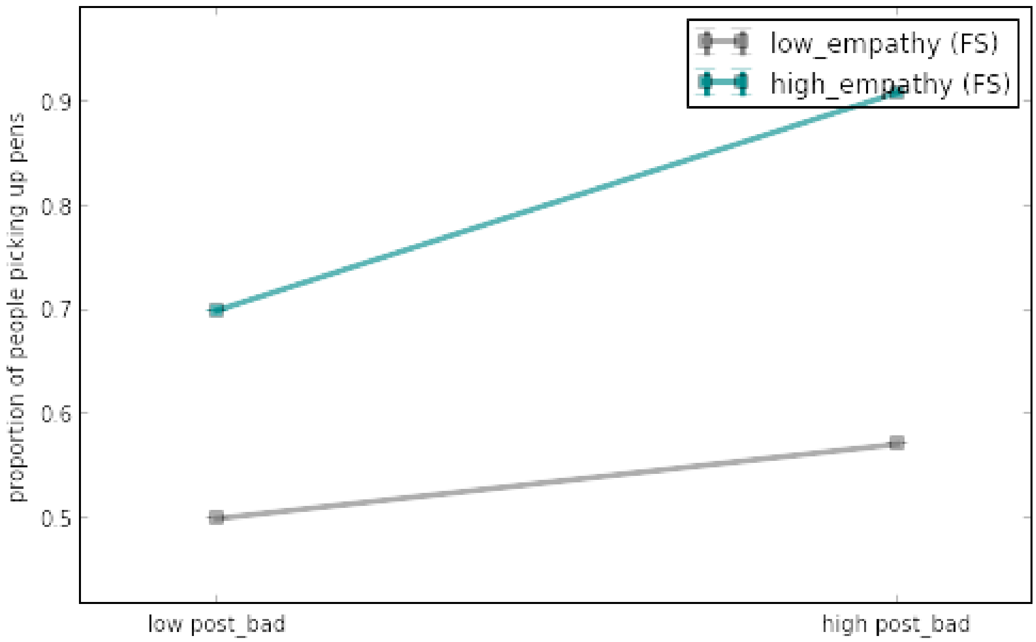
```
pen_interaction_plot(data_pen, trace = 'high_EC', x_levels = ['N',
'Y'], trace_levels = ['N', 'Y'],
                    x_labels = ['Did not feel bad for robot', 'Felt
bad for robot'],
                    trace_labels = ['low_empathy (EC)', 'high_empat
hy (EC)'])
```

```
got x_levels ['N', 'Y']
got trace_levels ['N', 'Y']
0 N N 0.333333333333 9
1 Y N 1.0 8
0 N Y 1.0 7
1 Y Y 0.6 10
```



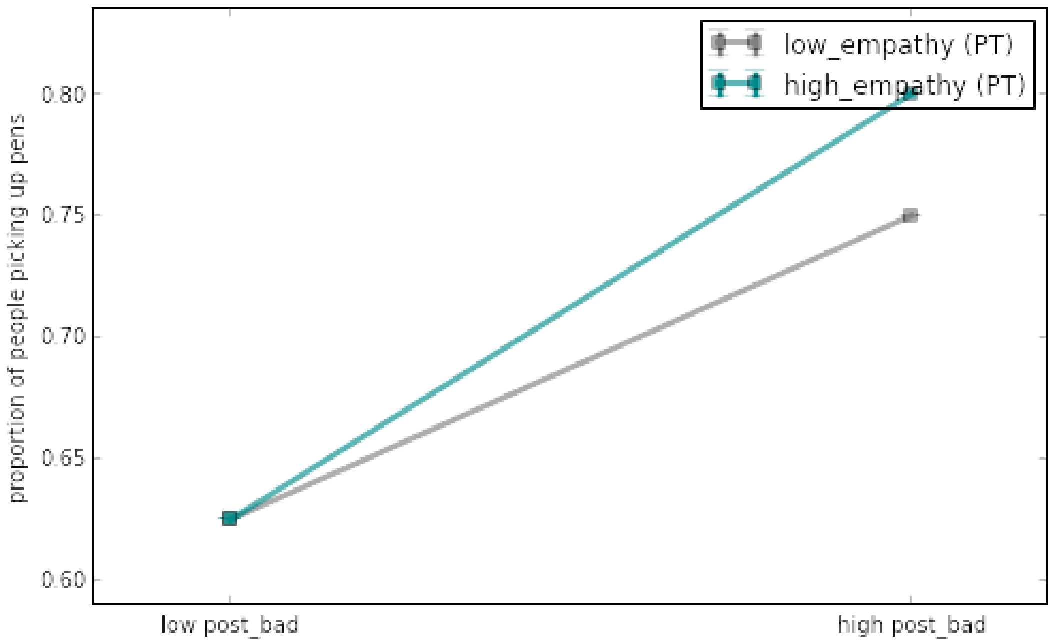
In [255]:
```
pen_interaction_plot(data_pen, trace = 'high_FS', x_levels = ['N',
'Y'], trace_levels = ['N', 'Y'],
                    x_labels = ['low post_bad', 'high post_bad'],
                    trace_labels = ['low_empathy (FS)', 'high_empat
hy (FS)'])
```

```
got x_levels ['N', 'Y']
got trace_levels ['N', 'Y']
0 N N 0.5 6
1 Y N 0.571428571429 7
0 N Y 0.7 10
1 Y Y 0.909090909091 11
```



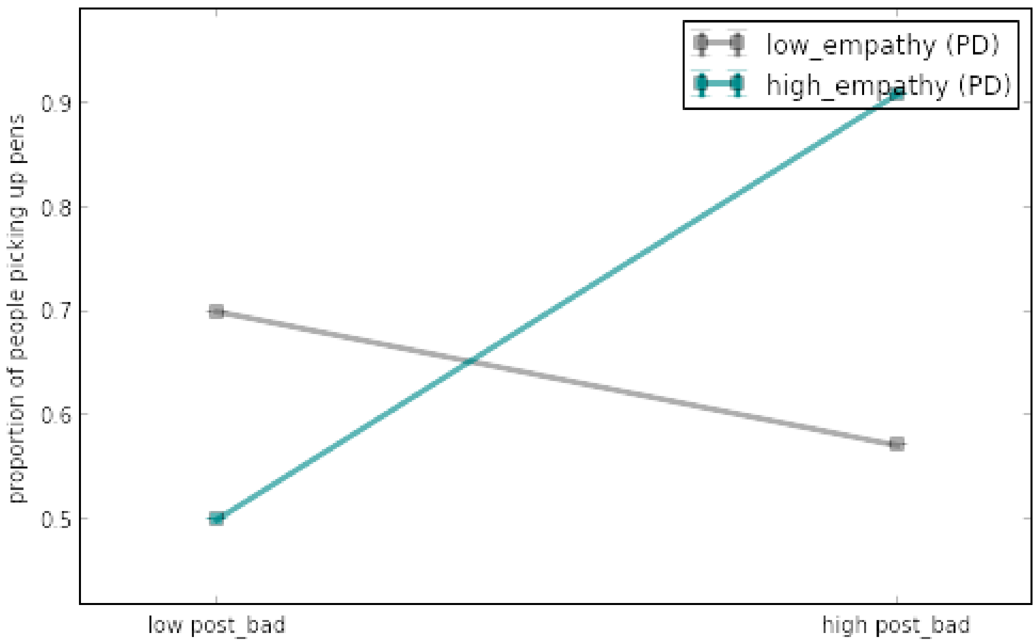
In [256]:
```
pen_interaction_plot(data_pen, trace = 'high_PT', x_levels = ['N',
'Y'], trace_levels = ['N', 'Y'],
                    x_labels = ['low post_bad', 'high post_bad'],
                    trace_labels = ['low_empathy (PT)', 'high_empat
hy (PT)'])
```

```
got x_levels ['N', 'Y']
got trace_levels ['N', 'Y']
0 N N 0.625 8
1 Y N 0.75 8
0 N Y 0.625 8
1 Y Y 0.8 10
```



In [258]:
```
pen_interaction_plot(data_pen, trace = 'high_PD', x_levels = ['N',
'Y'], trace_levels = ['N', 'Y'],
                     x_labels = ['low post_bad', 'high post_bad'],
                     trace_labels = ['low_empathy (PD)', 'high_empat
hy (PD)'])
```

```
got x_levels ['N', 'Y']
got trace_levels ['N', 'Y']
0 N N 0.7 10
1 Y N 0.571428571429 7
0 N Y 0.5 6
1 Y Y 0.909090909091 11
```



In [270]:

```
d = data_pen
# checking to see how to explain more people picked up the pen when they felt bad for the robot
formula = 'post_bad ~ C(pen_pickup)*C(high_EC)'
lm = ols(formula, d).fit()
print lm.summary()
#fig, ax = plt.subplots()
#fig = statsmodels.api.graphics.plot_fit(lm, 2, ax=ax)
print anova_lm(lm)
print '> normality of residuals', shapiro(lm.resid)
```

```
                            OLS Regression Results
==============================================================================
===========
Dep. Variable:               post_bad   R-squared:
0.392
Model:                            OLS   Adj. R-squared:
0.331
Method:                 Least Squares   F-statistic:
6.453
Date:                Wed, 20 Apr 2016   Prob (F-statistic):
0.00167
Time:                        21:35:35   Log-Likelihood:
-50.128
No. Observations:                  34   AIC:
108.3
Df Residuals:                      30   BIC:
114.4
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
===================================
                                          coef    std err          t
P>|t|      [95.0% Conf. Int.]
------------------------------------------------------------------------------
-----------------------------------
Intercept                               2.0000      0.459      4.354
0.000         1.062     2.938
C(pen_pickup)[T.P]                      1.9091      0.571      3.343
0.002         0.743     3.075
C(high_EC)[T.Y]                         3.0000      0.726      4.130
0.000         1.517     4.483
C(pen_pickup)[T.P]:C(high_EC)[T.Y]   -3.5245      0.860     -4.097
0.000        -5.281    -1.768
==============================================================================
===========
Omnibus:                        0.552   Durbin-Watson:
2.082
Prob(Omnibus):                  0.759   Jarque-Bera (JB):
0.537
Skew:                          -0.273   Prob(JB):
0.764
Kurtosis:                       2.713   Cond. No.
8.98
==============================================================================
===========

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors
is correctly specified.
                         df     sum_sq    mean_sq          F    PR
(>F)
C(pen_pickup)             1   1.275000   1.275000   1.006950  0.32
3660
C(high_EC)                1   1.986715   1.986715   1.569037  0.22
```

```
0024
C(pen_pickup):C(high_EC)   1  21.252271  21.252271  16.784286  0.00
0292
Residual                  30  37.986014   1.266200        NaN
NaN
> normality of residuals (0.9377186298370361, 0.052742067724466324)
```

In [272]:

```
d = data_pen
# checking to see how to explain more people picked up the pen when
they felt bad for the robot
formula = 'post_bad ~ C(pen_pickup)*C(high_FS)'
lm = ols(formula, d).fit()
print lm.summary()
#fig, ax = plt.subplots()
#fig = statsmodels.api.graphics.plot_fit(lm, 2, ax=ax)
print anova_lm(lm)
print '> normality of residuals', shapiro(lm.resid)
```

```
                            OLS Regression Results
==============================================================================
===========
Dep. Variable:               post_bad   R-squared:
0.042
Model:                            OLS   Adj. R-squared:
-0.053
Method:                 Least Squares   F-statistic:
0.4434
Date:                Wed, 20 Apr 2016   Prob (F-statistic):
0.724
Time:                        21:46:54   Log-Likelihood:
-57.856
No. Observations:                  34   AIC:
123.7
Df Residuals:                      30   BIC:
129.8
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
===================================
                                          coef    std err          t
P>|t|      [95.0% Conf. Int.]
------------------------------------------------------------------------------
-----------------------------------
Intercept                               3.5000      0.577      6.070
0.000         2.322     4.678
C(pen_pickup)[T.P]                      0.0714      0.786      0.091
0.928        -1.533     1.676
C(high_FS)[T.Y]                        -0.7500      0.912     -0.823
0.417        -2.612     1.112
C(pen_pickup)[T.P]:C(high_FS)[T.Y]     0.8256      1.111      0.743
0.463        -1.443     3.094
==============================================================================
===========
Omnibus:                        2.383   Durbin-Watson:
2.009
Prob(Omnibus):                  0.304   Jarque-Bera (JB):
1.833
Skew:                          -0.404   Prob(JB):
0.400
Kurtosis:                       2.200   Cond. No.
9.82
==============================================================================
===========

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors
is correctly specified.
                          df     sum_sq   mean_sq         F    PR(>
F)
C(pen_pickup)              1   1.275000  1.275000  0.639134  0.4303
09
C(high_FS)                 1   0.275963  0.275963  0.138335  0.7125
```

```
56
C(pen_pickup):C(high_FS)   1   1.102399  1.102399  0.552612  0.4630
35
Residual                  30  59.846639  1.994888       NaN       N
aN
> normality of residuals (0.9284204244613647, 0.028145471587777138)
```

In [273]:

```
d = data_pen
# checking to see how to explain more people picked up the pen when
they felt bad for the robot
formula = 'post_bad ~ C(pen_pickup)*C(high_PT)'
lm = ols(formula, d).fit()
print lm.summary()
#fig, ax = plt.subplots()
#fig = statsmodels.api.graphics.plot_fit(lm, 2, ax=ax)
print anova_lm(lm)
print '> normality of residuals', shapiro(lm.resid)
```

```
                            OLS Regression Results
==============================================================================
===========
Dep. Variable:               post_bad   R-squared:
0.022
Model:                            OLS   Adj. R-squared:
-0.075
Method:                 Least Squares   F-statistic:
0.2297
Date:                Wed, 20 Apr 2016   Prob (F-statistic):
0.875
Time:                        21:47:05   Log-Likelihood:
-58.208
No. Observations:                  34   AIC:
124.4
Df Residuals:                      30   BIC:
130.5
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
===================================
                                         coef    std err          t
P>|t|      [95.0% Conf. Int.]
------------------------------------------------------------------------------
-----------------------------------
Intercept                              3.2000      0.638      5.014
0.000         1.897     4.503
C(pen_pickup)[T.P]                     0.3455      0.770      0.449
0.657        -1.226     1.917
C(high_PT)[T.Y]                    -6.661e-16      0.903  -7.38e-16
1.000        -1.843     1.843
C(pen_pickup)[T.P]:C(high_PT)[T.Y]     0.1469      1.075      0.137
0.892        -2.049     2.343
==============================================================================
===========
Omnibus:                        3.125   Durbin-Watson:
1.935
Prob(Omnibus):                  0.210   Jarque-Bera (JB):
1.990
Skew:                          -0.374   Prob(JB):
0.370
Kurtosis:                       2.081   Cond. No.
9.22
==============================================================================
===========

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors
is correctly specified.
                          df     sum_sq   mean_sq         F    PR(>
F)
C(pen_pickup)              1   1.275000  1.275000  0.626059  0.4350
16
C(high_PT)                 1   0.090517  0.090517  0.044446  0.8344
```

```
51
C(pen_pickup):C(high_PT)   1   0.037979  0.037979  0.018649  0.8922
90
Residual                  30  61.096503  2.036550       NaN       N
aN
> normality of residuals (0.9325859546661377, 0.03722474351525307)
```

In [274]:

```
d = data_pen
# checking to see how to explain more people picked up the pen when they felt bad for the robot
formula = 'post_bad ~ C(pen_pickup)*C(high_PD)'
lm = ols(formula, d).fit()
print lm.summary()
#fig, ax = plt.subplots()
#fig = statsmodels.api.graphics.plot_fit(lm, 2, ax=ax)
print anova_lm(lm)
print '> normality of residuals', shapiro(lm.resid)
```

```
                              OLS Regression Results
==============================================================================
===========
Dep. Variable:               post_bad   R-squared:
0.136
Model:                            OLS   Adj. R-squared:
0.050
Method:                 Least Squares   F-statistic:
1.581
Date:                Wed, 20 Apr 2016   Prob (F-statistic):
0.215
Time:                        21:47:26   Log-Likelihood:
-56.099
No. Observations:                  34   AIC:
120.2
Df Residuals:                      30   BIC:
126.3
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
===================================
                                         coef    std err          t
P>|t|      [95.0% Conf. Int.]
------------------------------------------------------------------------------
-----------------------------------
Intercept                              3.6667      0.548      6.696
0.000         2.548     4.785
C(pen_pickup)[T.P]                    -0.4848      0.681     -0.712
0.482        -1.875     0.905
C(high_PD)[T.Y]                       -1.1667      0.866     -1.348
0.188        -2.935     0.601
C(pen_pickup)[T.P]:C(high_PD)[T.Y]     1.9848      1.025      1.936
0.062        -0.109     4.079
==============================================================================
===========
Omnibus:                        1.039   Durbin-Watson:
1.761
Prob(Omnibus):                  0.595   Jarque-Bera (JB):
0.812
Skew:                          -0.368   Prob(JB):
0.666
Kurtosis:                       2.825   Cond. No.
8.98
==============================================================================
===========

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors
is correctly specified.
                          df    sum_sq   mean_sq         F    PR(>
F)
C(pen_pickup)              1  1.275000  1.275000  0.708731  0.4065
27
C(high_PD)                 1  0.515130  0.515130  0.286344  0.5965
```

```
18
C(pen_pickup):C(high_PD)   1   6.740173  6.740173  3.746643  0.0623
88
Residual                  30  53.969697  1.798990       NaN       N
aN
> normality of residuals (0.9694252610206604, 0.44580239057540894)
```

In [ ]: