HW4

March 2, 2015

1 STAT 541 HW4

The following code is Python. I checked it against the SAS output. Seems legit.

1.1 Problem 1

In [52]:
```
import pandas as pd
import numpy as np
import statsmodels.api as sm
from statsmodels.formula.api import ols


### enter data
prob1_data = pd.DataFrame({'temperature' : np.repeat(['1520', '1600', '1680', '1760',\
 '1840'], 6),
                           'fail_time' : [1253, 1435, 1771, 4027, 5434, 5614,
                                          1190, 1286, 1550, 2125, 2557, 2845,
                                          751, 837, 848, 1038, 1361, 1443,
                                          611, 691, 751, 772, 808, 859,
                                          513, 546, 517, 420, 471, 556]})

### find logged group means and std
logmeans = np.log(prob1_data.groupby('temperature').mean())
logstd = np.log(prob1_data.groupby('temperature').std())

### fit model
logmeans = sm.add_constant(logmeans)
prob1_model = sm.OLS(logstd, logmeans).fit()

### calculate lambda
lambdaa = 1 - prob1_model.params[1]
lambdaa
```

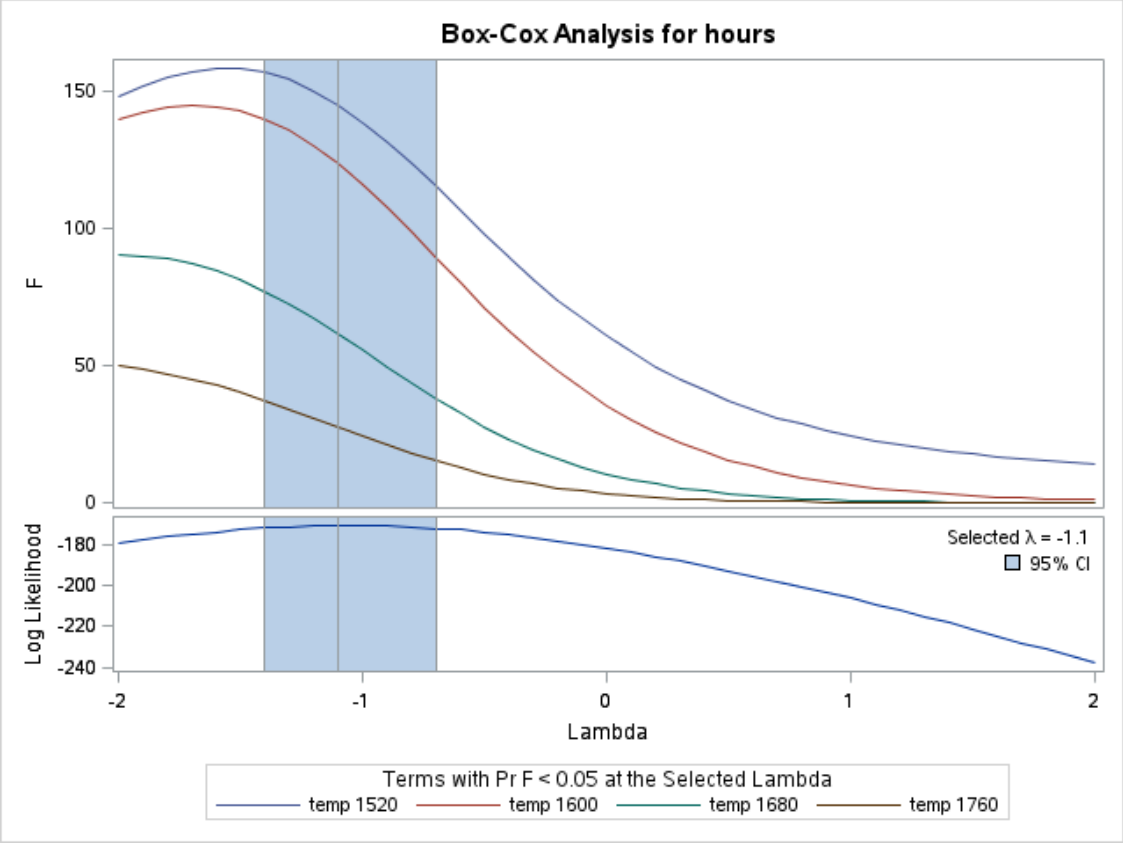
Out[52]:
```
-0.99756363490905109
```

The empirical method results in $\lambda \approx -1$ which is approximately a reciprical transformation.

1.2 Problem 2

In [53]:
```
from IPython.display import Image
Image(filename='hw4_boxcox.png')
```

Out[53]:



The Box-Cox function in Python doesn't work for the purposes of this context. The SAS output says that the recommended transformation is roughly the reciprical tranformation as well ($\lambda \approx -1.1$).

1.3 Problem 3

1.3.1 Part (a)

The means model can be notated as

$$y_{ij} = \mu_i + \epsilon_{ij} \qquad 1 \leq i \leq 5, 1 \leq j \leq 6$$

$$\epsilon_{ij} \overset{iid}{\sim} N(0, \sigma^2)$$

where i is the index for group and j is the index for observations within groups. The parameter μ_i represents the mean failure time in hours of group i, ϵ_{ij} is the error term for observation j in group i, and σ^2 is the variance of the errors.

1.3.2 Part (b)

```
In [69]: from scipy import stats

         ### seperate data by temperature level
         g1=prob1_data.query('temperature =="1520"').fail_time
         g2=prob1_data.query('temperature =="1600"').fail_time
         g3=prob1_data.query('temperature =="1680"').fail_time
         g4=prob1_data.query('temperature =="1760"').fail_time
```

```
g5=prob1_data.query('temperature =="1840"').fail_time

### run levene's test
levene_out = stats.levene(g1, g2, g3, g4, g5, center='mean')
{'F-statistic' : levene_out[0], 'p-value' : levene_out[1]}
```

```
Out[69]: {'F-statistic': 37.404536440145407, 'p-value': 3.2862755117618303e-10}
```

The results of Levene's test will inform a hypothesis test of

$$H_0 : \sigma_i = \sigma_j \quad \text{for all} \quad i, j$$

$$H_a : \sigma_i \neq \sigma_j \quad \text{for some} \quad i, j$$

but Levene's test is actually testing the following hypotheses: let μ_i^* denote the mean $\hat{\mu}_i^*$ absolute deviation for group i which is estimated by $\hat{\mu}_i^* = \frac{1}{n}\sum_j |y_{ij} - \bar{y}_i|$

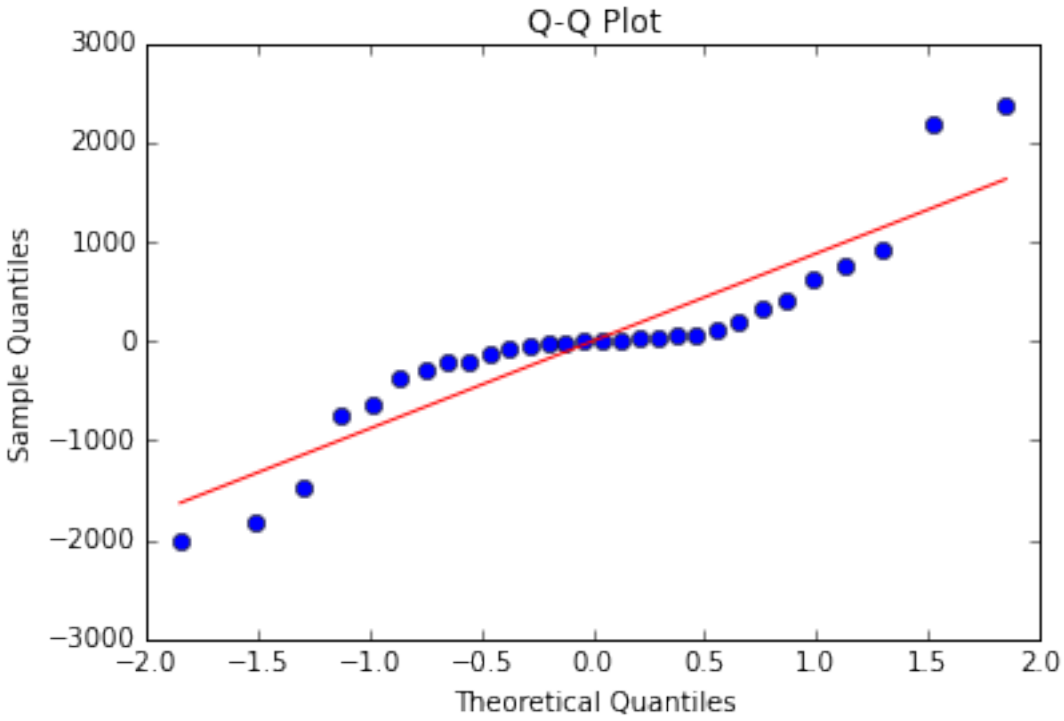
$$H_0 : \mu_i^* = \mu_j^* \quad \text{for all} \quad i, j$$

$$H_a : \mu_i^* \neq \mu_j^* \quad \text{for some} \quad i, j$$

The F-statatistic of about 37.4 results in a p-value of .000.... This provides plenty of evidence against the null hypothesis and we conclude that the variances in failure time between the different temperature levels are not equal.

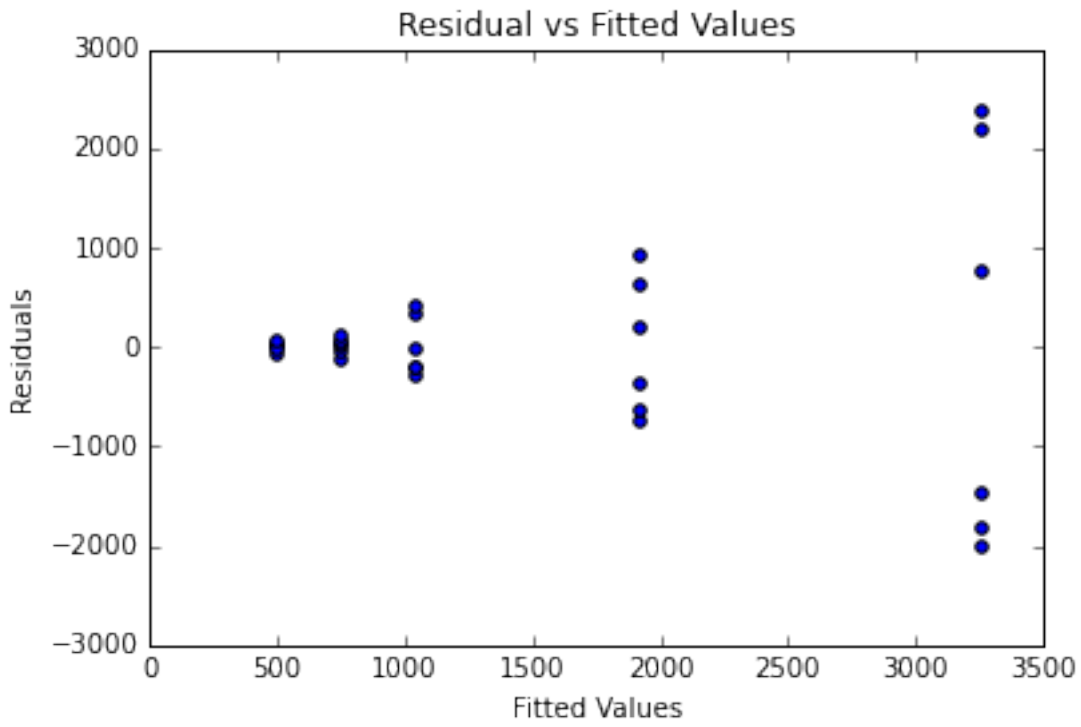
1.3.3 Part (c)

```
In [75]: import matplotlib.pyplot as plt
         %matplotlib inline

         ### fit model and get residuals and fitted values
         prob1_model3 = ols('fail_time ~ temperature', prob1_data).fit()
         resid = prob1_model3.resid
         fitted = prob1_model3.fittedvalues

         ### normal probability plot
         sm.qqplot(resid, line='s')
         title('Q-Q Plot')
         plt.show()
```



In the normal probability plot the points don't lie closely to the diagonal line indicating that the residuals are not normally distributed.

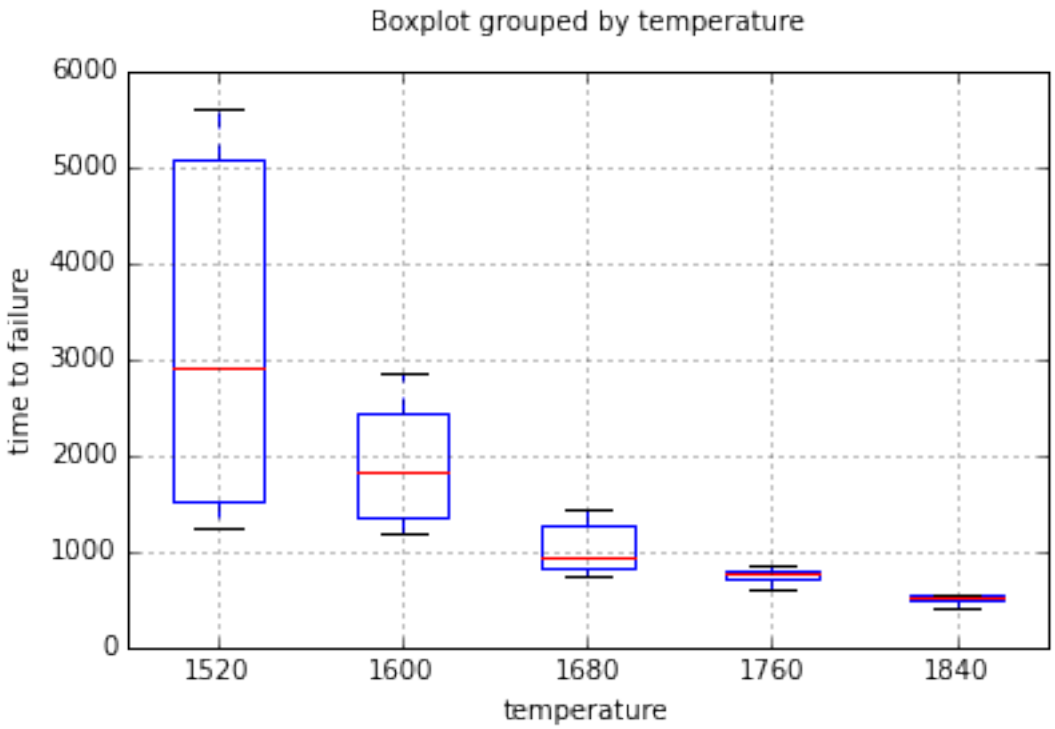
```
In [57]: ### fitted values vs residuals plot
         plt.scatter(fitted, resid)
         xlabel('Fitted Values')
         ylabel('Residuals')
         title('Residual vs Fitted Values')
         plt.show()
```



In the residuals versus predicted value plot there is an obvious fanning of variance as the fitted values increase. This implies that the homogeneity of variance assumption is violated.

1.3.4 Part (d)

```
In [61]: prob1_data.boxplot(column = 'fail_time', by = 'temperature')
         ylabel('time to failure')
         title('')
         plt.show()
```



It seems pretty clear that as the mean failure time increases, so does the variance for failure time per temperature grouping.

1.3.5 Pard (e)

```
In [63]: from statsmodels.stats.multicomp import MultiComparison

         tukey_test = MultiComparison(prob1_data['fail_time'], prob1_data['temperature'])
         print tukey_test.tukeyhsd()
```

```
Multiple Comparison of Means - Tukey HSD,FWER=0.05
=====================================================
group1 group2  meandiff    lower      upper    reject
-----------------------------------------------------
  0      1    -1330.1667 -2966.6521  306.3187  False
  0      2    -2209.3333 -3845.8187 -572.8479   True
  0      3     -2507.0   -4143.4854 -870.5146   True
  0      4    -2751.8333 -4388.3187 -1115.3479  True
  1      2    -879.1667  -2515.6521  757.3187  False
  1      3    -1176.8333 -2813.3187  459.6521  False
  1      4    -1421.6667 -3058.1521  214.8187  False
  2      3    -297.6667  -1934.1521 1338.8187  False
  2      4      -542.5   -2178.9854 1093.9854  False
  3      4    -244.8333  -1881.3187 1391.6521  False
-----------------------------------------------------
```

It appears that the 1520 temperature group is significantly different from the 1680, 1760, and 1840 groups and all other group means are not considered significantly different.

1.4 Problem 4

1.4.1 Part (a)

```
In [64]: prob1_data['fail_time_recip'] = 1/prob1_data['fail_time']
```

The model is similar which only the response being transformed. Let $w_{ij} = \frac{1}{y_{ij}}$. The means model can be notated as

$$w_{ij} = \mu_i + \epsilon_{ij} \qquad 1 \leq i \leq 5, 1 \leq j \leq 6$$

$$\epsilon_{ij} \overset{iid}{\sim} N(0, \sigma^2)$$

where i is the index for group and j is the index for observations within groups. The parameter μ_i represents the mean reciprical failure time in hours of group i, ϵ_{ij} is the error term for observation j in group i, and σ^2 is the variance of the errors.

1.4.2 Part (b)

```
In [68]: ### seperate data by temperature level
         g1_recip=prob1_data.query('temperature == "1520"').fail_time_recip
         g2_recip=prob1_data.query('temperature == "1600"').fail_time_recip
         g3_recip=prob1_data.query('temperature == "1680"').fail_time_recip
         g4_recip=prob1_data.query('temperature == "1760"').fail_time_recip
         g5_recip=prob1_data.query('temperature == "1840"').fail_time_recip

         ### run levene's test
         levene_out_recip = stats.levene(g1_recip, g2_recip, g3_recip, g4_recip, g5_recip, center='mean
         {'F-statistic' : levene_out_recip[0], 'p-value' : levene_out_recip[1]}
```

```
Out[68]: {'F-statistic': 1.3373187128746151, 'p-value': 0.28378015800416206}
```

The results of Levene's test will inform a hypothesis test of

$$H_0 : \sigma_i = \sigma_j \quad \text{for all} \quad i, j$$

$$H_a : \sigma_i \neq \sigma_j \quad \text{for some} \quad i, j$$

but Levene's test is actually testing the following hypotheses: let μ_i^{**} denote the mean absolute deviation for group i (estimated by $\mu_i^{**} = \frac{1}{n}\sum_j |w_{ij} - \bar{w}_i|$)

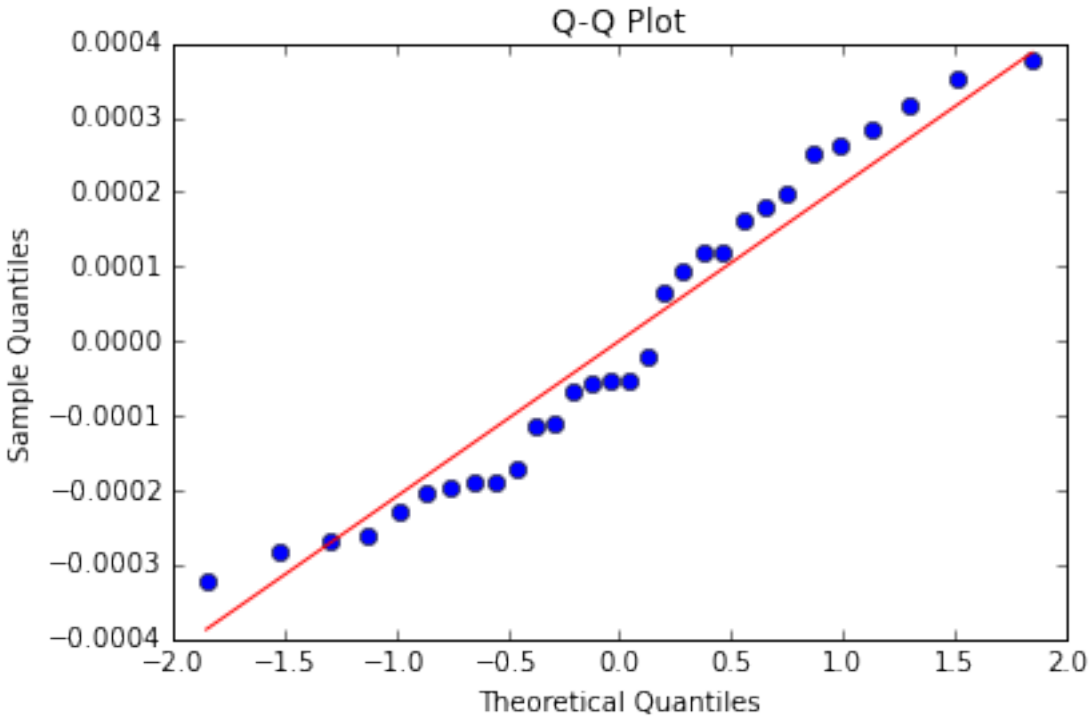
$$H_0 : \mu_i^{**} = \mu_j^{**} \quad \text{for all} \quad i, j$$

$$H_a : \mu_i^{**} \neq \mu_j^{**} \quad \text{for some} \quad i, j$$

The F-statatistic of about 1.337 results in a p-value of .284. This provides little evidence that after taking the reciprical of the failure times that there is a difference in variance between groups.

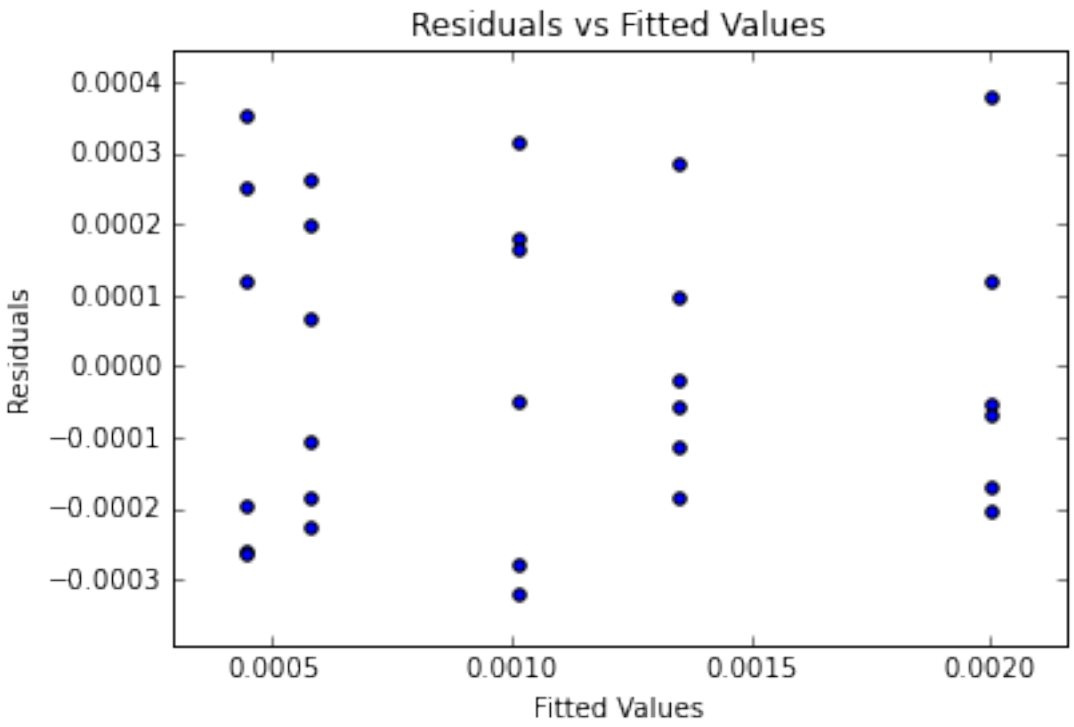
1.4.3 Part (c)

```
In [76]: ### fit model and get residuals and fitted values
         prob1_model4 = ols('fail_time_recip ~ temperature', prob1_data).fit()
         resid_recip = prob1_model4.resid
         fitted_recip = prob1_model4.fittedvalues

         ### normal probability plot
         sm.qqplot(resid_recip, line='s')
         title('Q-Q Plot')
         plt.show()
```



The normal probability plot looks pretty good. The dots lie close to the diagonal line, indicating that the residuals are approx normally distributed.

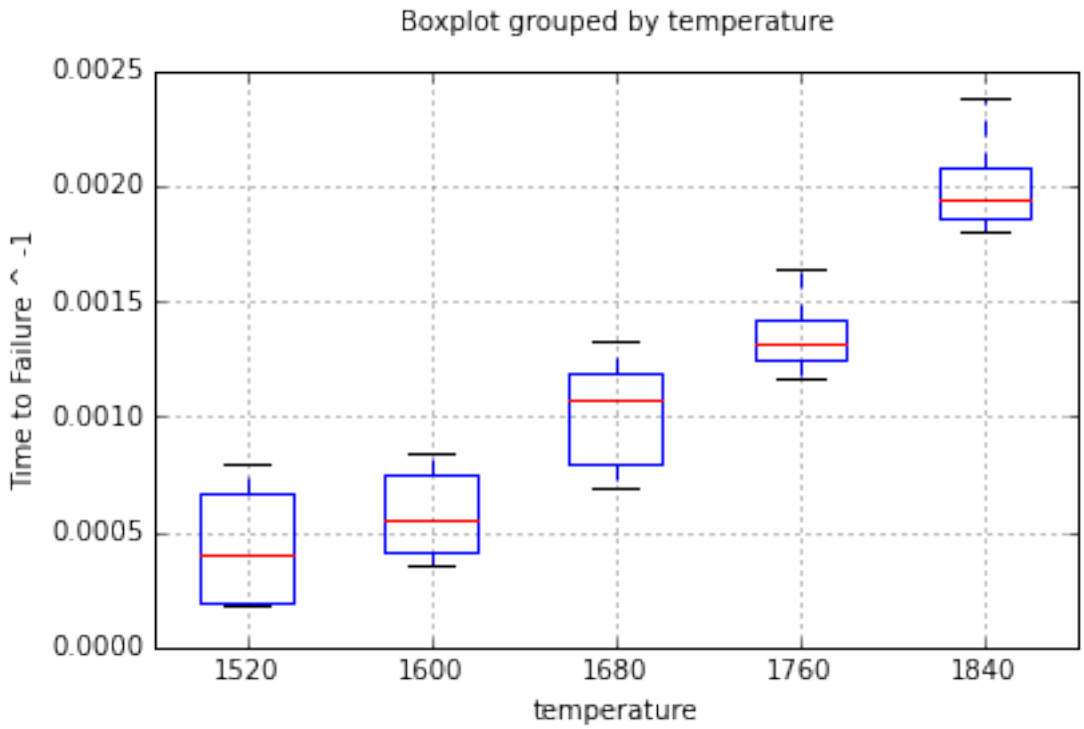
```
In [73]: ### residuals vs fitted value plot
         plt.scatter(fitted_recip, resid_recip)

         ### because range of values is so small, the plot has to be adjusted
         range_x = max(fitted_recip) - min(fitted_recip)
         range_y = max(resid_recip) - min(resid_recip)
         plt.xlim(min(fitted_recip) - .1*range_x, max(fitted_recip) + .1*range_x)
         plt.ylim(min(resid_recip) - .1*range_y, max(resid_recip) + .1*range_y)
         xlabel('Fitted Values')
         ylabel('Residuals')
         title('Residuals vs Fitted Values')
         plt.show()
```



In the residual versus fitted value plot there is no obvious fanning implying that there is no homogeneity of variance violation.

1.4.4 Part (d)

```
In [78]: prob1_data.boxplot(column = 'fail_time_recip', by = 'temperature')
         ylabel('Time to Failure ^ -1')
         title('')
         plt.show()
```



It appears that the variance remains constant across the different temperature groups, regardless of mean reciprical time to failure.

1.4.5 Part (e)

```
In [79]: tukey_test_recip = MultiComparison(prob1_data['fail_time_recip'], prob1_data['temperature'])
         print tukey_test_recip.tukeyhsd()
```

```
Multiple Comparison of Means - Tukey HSD,FWER=0.05
============================================
group1 group2 meandiff  lower  upper  reject
--------------------------------------------
  0      1     0.0001  -0.0003 0.0005 False
  0      2     0.0006   0.0002 0.001   True
  0      3     0.0009   0.0005 0.0013  True
  0      4     0.0016   0.0012 0.0019  True
  1      2     0.0004    0.0   0.0008  True
  1      3     0.0008   0.0004 0.0012  True
  1      4     0.0014   0.001  0.0018  True
  2      3     0.0003  -0.0001 0.0007 False
  2      4     0.001    0.0006 0.0014  True
  3      4     0.0007   0.0003 0.001   True
--------------------------------------------
```

It appears that the 1520 temperature group is not significantly different from the 1600 group and the 1680 group is not significantly different from the 1760 group. All other group comparisons are considered sig different.

1.5 Problem 5

I would definitely go with the transformed data. The diagnostic plots look much better indicating that we can assume the model assumptions are met. Looking at the original boxplot of the untransformed data, it appears that there were actually more differences than what the Tukey comparisons suggested. By taking the reciprical of the failure times we have 'tamed' the increasing nature of spread which allows us to see more accurately differences in our analysis.

1.6 Problem 6

```
In [82]: prob1_model5 = ols('fail_time_recip ~ C(temperature, Poly)', prob1_data).fit()
         prob1_model5.summary()
```

```
Out[82]: <class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:        fail_time_recip   R-squared:                       0.879
Model:                            OLS   Adj. R-squared:                  0.859
Method:                 Least Squares   F-statistic:                     45.20
Date:                Sun, 01 Mar 2015   Prob (F-statistic):           4.31e-11
Time:                        23:00:30   Log-Likelihood:                 211.59
No. Observations:                  30   AIC:                            -413.2
Df Residuals:                      25   BIC:                            -406.2
Df Model:                           4
==================================================================================================
                                     coef    std err          t      P>|t|      [95.0% Conf. In
--------------------------------------------------------------------------------------------------
Intercept                          0.0011   4.19e-05     25.782      0.000         0.001      0
C(temperature, Poly).Linear        0.0012   9.36e-05     13.139      0.000         0.001      0
C(temperature, Poly).Quadratic     0.0002   9.36e-05      2.662      0.013      5.64e-05      0
C(temperature, Poly).Cubic      3.962e-06   9.36e-05      0.042      0.967        -0.000      0
C(temperature, Poly)^4          9.787e-05   9.36e-05      1.046      0.306     -9.49e-05      0
==============================================================================
Omnibus:                        6.560   Durbin-Watson:                   1.315
Prob(Omnibus):                  0.038   Jarque-Bera (JB):                2.202
Skew:                           0.240   Prob(JB):                        0.332
Kurtosis:                       1.762   Cond. No.                         2.24
==============================================================================
"""
```

As expected, it looks like there is a linear trend ($t = 13.139$, p-value $= .001$) and there is evidence for curvature ($t = 2.662$, p-value $= .013$). There is not any evidence for a cubic or quartic term.

1.7 Problem 7

1.7.1 Part (a)

I don't think it is reasonable to perform a WLS ANOVA because it appears that there is a relationship between mean response and variability. If there wasn't such a relationship then WLS would be an alright choice. Using WLS would not capture this relationship.

1.7.2 Part (b)

I woulds say performing a Box-Cox transformation does sound reasonable as it is apparent that the variance is proportional to some power of the mean response per group. The Box-Cox procedure seeks to find this power.

1.8 Problem 8

In [104]:
```
import math
import statsmodels.stats.power as smp

sigma = 25
mus = [60., 50., 60., 55.]
mu = np.mean(mus)
f = math.sqrt(((60 - mu)**2/4 + (50 - mu)**2/4 + (60 - mu)**2/4 + (55 - mu)**2/4)/sigma)

seq1 = [.9, .91, .92, .93, .94]
sample_size = [smp.FTestAnovaPower().solve_power(effect_size=f, alpha=0.05, power=x, k_groups=
pd.DataFrame({'power (1 - beta)' : seq1, 'Group Sample Size (n)' : sample_size})
```

Out[104]:

	Group Sample Size (n)	power (1 - beta)
0	6.231466	0.90
1	6.400805	0.91
2	6.587819	0.92
3	6.797018	0.93
4	7.035120	0.94

It appears that for each of the four groups, sample sizes of $n = 7$ are required to have a power of at least $1 - \beta = .9$.

1.9 Problem 9

1.9.1 Part (a)

In [105]:
```
sigma = 36
mus = [60., 50., 60., 55.]
mu = np.mean(mus)
f = math.sqrt(((60 - mu)**2/4 + (50 - mu)**2/4 + (60 - mu)**2/4 + (55 - mu)**2/\
4)/sigma)

seq1 = [.9, .91, .92, .93, .94]
sample_size2 = [smp.FTestAnovaPower().solve_power(effect_size=f, alpha=0.05, power=x, k_groups
pd.DataFrame({'power (1 - beta)' : seq1, 'Group Sample Size (n)' : sample_size2})
```

Out[105]:

	Group Sample Size (n)	power (1 - beta)
0	8.470832	0.90
1	8.715748	0.91
2	8.986171	0.92
3	9.288634	0.93
4	9.632826	0.94

It appears that for each of the four groups, sample sizes of $n = 9$ are required to have a power at least $1 - \beta = .9$

1.9.2 Part (b)

In [106]:
```
sigma = 49
mus = [60., 50., 60., 55.]
mu = np.mean(mus)
f = math.sqrt(((60 - mu)**2/4 + (50 - mu)**2/4 + (60 - mu)**2/4 + (55 - mu)**2/\
4)/sigma)
```

```
seq1 = [.9, .91, .92, .93, .94]
sample_size2 = [smp.FTestAnovaPower().solve_power(effect_size=f, alpha=0.05, power=x, k_groups
pd.DataFrame({'power (1 - beta)' : seq1, 'Group Sample Size (n)' : sample_size2})
```

Out[106]:

	Group Sample Size (n)	power (1 - beta)
0	11.132370	0.90
1	11.466476	0.91
2	11.835269	0.92
3	12.247781	0.93
4	12.717156	0.94

It appears that for each of the four groups, sample sizes of $n = 12$ are required to have a power at least $1 - \beta = .9$

1.9.3 Parts (c) and (d)

As the variability increases, the sensitivity of an ANOVA F-test decreases. Thus, the need for more data to detect an actual difference is required. As σ increases, so should the sample sizes n.