Appendix 1. Detailed notes on OLS and log OLS models

ORDINARY LEAST SQUARES REGRESSION (OLS)

\[ Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_k X_k + \varepsilon, \]

- Coefficient on treatment indicator produces an estimate of \( \beta \)
- Assumptions generally violated with cost data:
  - Variance of error term is constant.
    * Heteroskedasticity - Unbiased coefficients, but potentially inefficient
  - Error term is normally distributed
    * Violation has no effect on coefficients
    * Necessary only for test of statistical significance
    * In large samples we can rely on the central limit theorem
    * Has been shown that regression analysis is robust against violations of normality
    * Difficult to defend in practice
    * See tests for normality in Appendix 2

See Computer Output, p. 11
OLS RESULT

<table>
<thead>
<tr>
<th>Cost</th>
<th>S.E.</th>
<th>t</th>
<th>p</th>
<th>95% CI</th>
</tr>
</thead>
<tbody>
<tr>
<td>4027</td>
<td>853.8</td>
<td>4.72</td>
<td>0.000</td>
<td>2343 to 5711</td>
</tr>
</tbody>
</table>

replace treat=0
predict ols_0
replace treat=1
predict ols_1
replace treat=tmptreat
gen ols_dif=ols_1-ols_0

. tabstat ols_1 ols_0 ols_dif
stats | _ ols_1  ols_0  ols_dif
        -----------------------------
        mean | 10845.52  6818.93  4026.585
        ----------------------------------------

Univariate Result:

. tabstat cost if treat==1
  variable |    mean
            -----------------------------
            cost | 10832.22
            ----------------------------------------

. tabstat cost if treat==0
  variable |    mean
            -----------------------------
            cost | 6832.225
            ----------------------------------------

LOG OF COSTS: ORDINARY LEAST SQUARES REGRESSION

- Coefficient on treatment indicator produces an estimate of the percentage difference in mean costs between treatment groups

- For cost effectiveness analysis we are interested in the predicted mean costs of treatment. With a log of cost regression we are predicting log of costs, not costs
  - Is it possible to reliably transform predicted log of costs into an unbiased prediction of mean costs?
  - Not trivial because the mean of the log of costs is not equal to the log of the mean of costs
    i.e. E(ln(y)/x) is not equal to ln(E(y/x))

- Residual may not be normally distributed even after log transformation

See Computer Output, p. 11
RETRANSFORMATION AFTER THE LOG OF COST REGRESSION (I)

- The result of the regression is the predicted log of costs \( \hat{Z}_i \) where:

\[
\hat{Z}_i = \hat{\alpha} + X_i \hat{\beta}_1 + T_i \hat{\beta}_T
\]

- Estimation of the effect of treatment on predicted costs is a nonlinear retransformation of the regression coefficients.

- Nonlinear retransformations have to account for two complexities:
  - In the logged scale, the (multiplicative) effect of the treatment group is estimated holding all else equal; however, exponentiation of predicted log costs (to estimate costs) reintroduces the covariate imbalances.
  - Simple exponentiation of the predicted log costs leads to biased estimates (Duan, 1983).

RETRANSFORMATION AFTER THE LOG OF COST REGRESSION (II)

- Avoiding reintroduction of covariate imbalance (method of recycled predictions):
  - Code everyone as if they were in treatment group A and predict \( \bar{Z}_{iA} \).
  - Code everyone as if they were in treatment group B and predict \( \bar{Z}_{iB} \).

- Avoiding the bias of simple exponentiation of the predicted log (smearing retransformation):
  - Retransform the predicted log of costs (both \( \bar{Z}_{iA} \) and \( \bar{Z}_{iB} \)) into the original scale of costs by use of a smearing factor \( N \) as follows:

\[
E(Y_{iA}/X_i) = \Phi e^{\bar{Z}_{iA}}
\]

\[
E(Y_{iB}/X_i) = \Phi e^{\bar{Z}_{iB}}
\]

where

\[
\Phi = \frac{1}{N} \sum_{i=1}^{N} e^{(\bar{Z}_i - \bar{Z}_i)}
\]
SMEARING AND HETEROSKEDASTICITY (I)

- Use of a single smearing factor introduces bias when there is heteroskedasticity in the data

- For example:
  - If the variance of the residuals is greater for the treatment group than for the comparison group, the smearing correction parameter will underestimate the costs for the treatment group and overestimate the costs for the standard

- Solution:
  - Estimate the smearing correction parameter separately for the two treatment groups
    * For those actually in group A
      \[ \Phi_A = \frac{1}{N_A} \sum_{i=1}^{N_A} e^{(z_A - z_A)} \]
    * For those actually in group B
      \[ \Phi_B = \frac{1}{N_B} \sum_{i=1}^{N_B} e^{(z_B - z_B)} \]

SMEARING AND HETEROSKEDASTICITY (II)

- Retransformation in the face of heteroskedasticity
  - When everyone is coded as if they are in treatment group A
    \[ E(Y_{A_i}/X_i) = \Phi_A \cdot e^{(z_A)} \]
  - When everyone is coded as if they are in treatment group B
    \[ E(Y_{B_i}/X_i) = \Phi_B \cdot e^{(z_B)} \]

- See Breusch-Pagan test for heteroskedasticity in Appendix 2
- See Manning (1998) for unbiased estimation when heteroskedasticity exists in the data
- See Ai and Norton (2000) for estimates of standard errors for the retransformation problem with heteroskedasticity
LOG RESULT - SMEARING FACTOR:

RUN LOGIT

. regress lcost treat $ivar

ESTIMATE THETA

**smearing factor comes from the regression

* residual
predict res, r
replace res=exp(res)

***HOMOSKEDASTIC SMEARING FACTOR
sum res
scalar sm=r(mean)

***HETEROSKEDASTIC SMEARING FACTORS
sum res if treat==0
scalar sm0=r(mean)
sum res if treat==1
scalar sm1=r(mean)

. display sm
1.3056459

. display sm0
1.4702791

. display sm1
1.1410127

LOG RESULT - RETRANFORMATION:

****ESTIMATE Z’S FOR EACH TREATMENT GROUP
replace treat=0
predict lc0
replace treat=1
predict lc1
replace treat=tmp

*****HETEROSKEDASTIC PREDICTION
***RETRANSFORM AND APPLY SMEARING FACTOR
gen lols_0=exp(lc0)*sm0
gen lols_1=exp(lc1)*sm1
gen lols_t=lols_0 if treat==0
   replace lols_t=lols_1 if treat==1
gen lols_dif=lols_1-lols_0

. tabstat lols_1 lols_0 lols_dif
   stats |  lols_1  lols_0  lols_dif
----------+------------------------------
   mean |  11156.22  6888.652  4267.573

----------+

*****HOMOSKEDASTIC PREDICTION
***RETRANSFORM AND APPLY SMEARING FACTOR
gen lolsho_0=exp(lc0)*sm
gen lolsho_1=exp(lc1)*sm
gen lolsho_dif=lolsho_1-lolsho_0
gen lolsho_t=lolsho_0 if treat==0
   replace lolsho_t=lolsho_1 if treat==1

. tabstat lolsho_1 lolsho_0 lolsho_dif
   stats |  lolsho_1  lolsho_0  lolsho_dif
----------+------------------------------
   mean |  12765.92  6117.301  6648.621

----------+
*** OLS (untransformed)

```
. regress cost treat $ivar

Source | SS  df  MS
---------+------------------------------
Model | 1.1771e+09   6  196177149
Residual | 6.9525e+09  193  36023463.2
---------+------------------------------
Total | 8.1296e+09  199  40852217.5

Number of obs = 200
F(  6,   193) = 5.45
Prob > F = 0.0000
R-squared = 0.1448
Adj R-squared = 0.1182
Root MSE = 6002

------------------------------------------------------------------------------
|          Cost | Coef. Std. Err.   t   P>|t|  [95% Conf. Interval]
|--------------|-------------------+-------------------+-------------------
| treat | 4026.585  853.8104   4.72  0.000  2342.587    5710.582
| age | 75.1648  42.17426   1.78  0.076 -8.016836    158.3464
| ejfract | -76.18302  64.84257  -1.17  0.241 -204.0741    51.70804
| sex | -652.0594  975.3994  -0.67  0.505  -2575.87    1271.752
| etiology | 2264.755  887.3851   2.55  0.011  514.5371   4014.972
| race | 481.5117  1284.211   0.37  0.708 -2051.377   3014.401
| _cons | 2676.532   3087.71   0.87  0.387 -3413.456    8766.52

------------------------------------------------------------------------------
```

. ***Log costs

```
. regress lcost treat $ivar

Source | SS  df  MS
---------+------------------------------
Model | 39.2814791     6  6.54691318
Residual | 110.692964  193  .573538672
---------+------------------------------
Total | 149.974443  199  .753640416

Number of obs = 200
F(  6,   193) = 11.41
Prob > F = 0.0000
R-squared = 0.2619
Adj R-squared = 0.2390
Root MSE = .75732

------------------------------------------------------------------------------
|          lcost | Coef. Std. Err.   t   P>|t|  [95% Conf. Interval]
|----------------+-------------------+-------------------+-------------------
| treat | .7356582  .107733  6.83  0.000   .5231724    .9481441
| age | .0072738  .0053215   1.37  0.173  -.003222    .0177696
| ejfract | -.016828  .0081818  -2.06  0.041  -.0329652   -.0006908
| sex | -.112238  .1230754  -0.91  0.363  -.3549834    .1305074
| etiology | .3244058  .1119698   2.90  0.004   .1035643    .5452473
| race | .3329655  .162041  2.05  0.041   .1033669    .6525641
| _cons | 7.998734  .3896055  20.53  0.000   7.230302    8.767165

------------------------------------------------------------------------------
```
APPENDIX 2. Technical Notes on Diagnostic Tests

This section provides details on the statistical tests discussed in this lecture.

1. Skewness/Kurtosis
2. Heteroskedasticity test (Breusch-Pagan test)
3. Modified Park test (GLM family test)
4. Pregibon Link test
5. Modified Hosmer Lemeshow test
6. Pearson’s Correlation test
7. Copas test

Tests for Normality and Heteroskedasticity

<table>
<thead>
<tr>
<th>Test</th>
<th>Purpose</th>
<th>Diagnosis</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Skewness/ kurtosis</td>
<td>Test normality of residuals</td>
<td>If skewness ( \neq 0 ) or kurtosis&gt;3 then residuals not normally distributed (OLS assumption violated)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If kurtosis &gt;3 on log-scale residuals GLM may suffer precision losses</td>
</tr>
<tr>
<td>Breusch-Pagan</td>
<td>Test whether residuals are</td>
<td>If reject null then residuals are heteroskedastic</td>
</tr>
<tr>
<td></td>
<td>heteroskedastic</td>
<td>If log-scale residuals are heteroskedastic, Log OLS will be biased if appropriate smearing correction not applied</td>
</tr>
</tbody>
</table>

Test for Heteroskedasticity (Breusch-Pagan test)

Is the treatment variable heteroskedastic?

After OLS cost regression:

e.g. regress cost treat $ivar

hettest treat

Breusch-Pagan / Cook-Weisberg test for heteroskedasticity

Ho: Constant variance

Variables: treat

\[
\text{chi}^2(1) = 2.03 \\
\text{Prob} > \text{chi}^2 = 0.1546
\]

After log OLS cost regression:

e.g. regress lcost treat $ivar

hettest treat

Breusch-Pagan / Cook-Weisberg test for heteroskedasticity

Ho: Constant variance

Variables: treat

\[
\text{chi}^2(1) = 32.26 \\
\text{Prob} > \text{chi}^2 = 0.0000
\]

YES in log model! (Also heteroskedastic in race & etiology)

See Computer Output, pp. 24 and 25
Summary of Results: Normality and Heteroskedasticity

After OLS cost regression:

Normality of Residuals
Co-efficient of Skewness: 1.96
Co-efficient of Kurtosis: 8.85
Joint test statistic of Skewness/Kurtosis: 72.1 (p<0.0000)

Heteroskedasticity in treatment variable
Breusch-Pagan test statistic: 2.0 (p=0.15)

After Log-OLS regression:

Normality of Residuals
Co-efficient of Skewness: -0.12
Co-efficient of Kurtosis: 2.9
Joint test statistic of Skewness/Kurtosis: 0.52 (p=0.77)

Heteroskedasticity in treatment variable
Breusch-Pagan test statistic: 32.3 (p=0.0000)

- OLS residuals are homoskedastic in treatment variable BUT they are not normally distributed!
- Log-OLS residuals are normal BUT heteroskedastic in treatment variable!

Tests for Determining Link Function and Family Distribution for GLM

<table>
<thead>
<tr>
<th>Test</th>
<th>Purpose</th>
<th>Diagnosis</th>
</tr>
</thead>
<tbody>
<tr>
<td>4. Modified Park</td>
<td>Determine family distribution for GLM</td>
<td>If 8=0 Gaussian</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If 8=1 Poisson</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If 8=2 Gamma</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If 8=3 Inverse</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Gaussian</td>
</tr>
</tbody>
</table>

Modified Park Test (GLM family test)

Purpose: To determine family distribution for GLM model (i.e. relationship between mean and variance)

AFTER GLM COST REGRESSION:
e.g. glm cost treat $ivar, family(gamma) link(log)

(1) Predict value of y and log transform it
    predict yhat
    gen lnyhat=ln(yhat)

(2) Save raw scale residuals and square them
    gen res=cost-yhat
    gen r2=((res)^2)

(3) Regress ln(r2) on ln(yhat) and a constant using GLM with gamma distribution
    glm r2 lnyhat, link(log) family(gamma) robust nolog

(4) Coefficient on ln(yhat) gives the family
    If 8=0 Gaussian NLLS
    If 8=1 Poisson
    If 8=2 Gamma
    If 8=3 Inverse Gaussian or Wald
    test lnyhat==0
    test lnyhat==1
    test lnyhat==2
    test lnyhat==3

See Computer Output, p. 15
Modified Park Test Results

Modified Park Test found $8 = 0.81$ (Poisson!)

- Test Ho $8=1$ (p=0.74) $\sim$ Poisson
- Test Ho $8=2$ (p=0.04) $\sim$ Not Gamma
- Test Ho $8=3$ (p=0.000) $\sim$ Not Inverse Gaussian

(Gaussian) $0 < 8 < 1$ (Poisson) Compromise!

Within-sample Diagnostics for Model Fit

<table>
<thead>
<tr>
<th>Test</th>
<th>Purpose</th>
<th>Diagnosis</th>
</tr>
</thead>
<tbody>
<tr>
<td>4. Pregibon Link</td>
<td>Test linearity of response on scale of estimation</td>
<td>If reject null then problem with the functional form of $X$</td>
</tr>
<tr>
<td>5. Modified Hosmer Lemeshow</td>
<td>Determine systematic bias in fit on raw scale</td>
<td>If reject null and observe systematic pattern then problem with either the link function or functional form of $X$</td>
</tr>
<tr>
<td>6. Pearson Correlation</td>
<td>Determine systematic bias in fit on raw scale</td>
<td>If reject null and high correlation co-efficient then problem with either the link function or functional form of $X$</td>
</tr>
</tbody>
</table>

Pregibon's Link Test

- Purpose: To determine linearity of response on scale of estimation

After COST REGRESSION:
e.g. glm cost treat $ivar$, family(gamma) link(log)

(1) Create two new variables
(a) Create prediction of (xb)
   predict xb, xb

(b) Create variable of squared prediction
gen xbsq=xb^2

(2) Refit model with the two new variables as the only predictors
   glm cost xb xbsq, family(gamma) link(log) robust

(3) Co-efficient on square of the prediction should not be significantly different from zero. i.e. test Ho: co-efficient on xbsq=0
   lincom xbsq (alternatively, test xbsq)

(4) If fail to reject null (i.e. xbsq not significant predictor) then keep model the same; if reject null then problem with functional form of $x$

See Computer Output, p. 27
Modified Hosmer Lemeshow Test

Purpose: To determine systematic pattern of bias in model fit on raw scale

After COST REGRESSION:
e.g. glm cost treat $ivar, family(gamma) link(log)

(1) Obtain predicted value of y on raw scale
predict yhat

(2) Compute residual on raw scale
gen res=cost-yhat

(3) Create 10 groups, sorted by x or xb
xtile xtile=xb, nq(10)
tab xtile, gen(xbt)

(4) Conduct an F-test of whether the mean of the raw scale residuals across all groups of the deciles are not significantly different from zero
reg res xbt1 xbt2 xbt3 xbt4 xbt5 xbt6 xbt7 xbt8 xbt9 xbt10, nocons robust
test xbt1 xbt2 xbt3 xbt4 xbt5 xbt6 xbt7 xbt8 xbt9 xbt10

(5) Look for systematic patterns by plotting mean residuals by deciles (e.g. U-shape)

(6) If reject null and find systematic pattern then there is a problem with either the left hand side (wrong power function) or right hand side (wrong functional form of x)

See Computer Output, p. 28

Pearson’s Correlation Test

Purpose: To determine systematic bias in the prediction of E(y/x)

After COST REGRESSION:
e.g. glm cost treat $ivar, family(gamma) link(log)

(1) Obtain predicted value of y on raw scale
predict yhat

(2) Compute residual on raw scale
gen res=cost-yhat

(3) Check correlation between res and yhat
pwcorr yhat res, sig

<table>
<thead>
<tr>
<th></th>
<th>yhat</th>
<th>res</th>
</tr>
</thead>
<tbody>
<tr>
<td>yhat</td>
<td>1.0000</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>res</td>
<td>-0.0335</td>
<td>1.0000</td>
</tr>
</tbody>
</table>
|      | 0.6372=> p-value

(4) If statistic is significantly different from zero then model is providing a biased prediction (i.e. estimated impact of the x on y (slope) is either too high or too low) and suggests that either the link function or linear specification is incorrect
Summary of Results

<table>
<thead>
<tr>
<th>Test</th>
<th>Stat</th>
<th>p</th>
<th>Stat</th>
<th>p</th>
<th>Stat</th>
<th>p</th>
</tr>
</thead>
<tbody>
<tr>
<td>OLS</td>
<td>0.17</td>
<td>0.68</td>
<td>0.69</td>
<td>0.73</td>
<td>0.000</td>
<td>1.00</td>
</tr>
<tr>
<td>Log OLS (Homo)</td>
<td>11.78</td>
<td>0.001</td>
<td>1.70</td>
<td>0.08</td>
<td>-0.309</td>
<td>0.000</td>
</tr>
<tr>
<td>Log OLS (Hetero)</td>
<td>11.78</td>
<td>0.001</td>
<td>0.70</td>
<td>0.73</td>
<td>-0.154</td>
<td>0.03</td>
</tr>
<tr>
<td>GLM gamma/log</td>
<td>1.39</td>
<td>0.24</td>
<td>0.65</td>
<td>0.77</td>
<td>-0.034</td>
<td>0.64</td>
</tr>
<tr>
<td>GLM poisson/log</td>
<td>0.09</td>
<td>0.77</td>
<td>0.58</td>
<td>0.83</td>
<td>-0.002</td>
<td>0.98</td>
</tr>
</tbody>
</table>

- Both Log OLS models (with homoskedastic as well as heteroskedastic retransformation) fail the Pregibon Link test and Pearson Correlation test!

Test for Model Fit Within and Out-of-Sample

<table>
<thead>
<tr>
<th>Test</th>
<th>Purpose</th>
<th>Diagnosis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Copas</td>
<td>Test for overfitting using split sample cross-validation</td>
<td>If reject null, then evidence of overfitting</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Use to rank models since test fails often</td>
</tr>
</tbody>
</table>

Copas Test

Purpose: To test for overfitting using split sample cross validation

(1) Randomly split sample into two equal groups A & B
(2) Estimate model on sample A and retain its co-efficients
(3) Forecast to sample B
\[ \hat{y}_b = X_b \hat{\beta}_A \]

(4) Regression model for sample B
\[ y_b = \delta_0 + \delta_1 \hat{y}_b + \eta \]
Test \( \hat{\delta}_1 = 0 \)

Alternatively,
\[ \text{res}_b = \delta_0 + \delta_1 \hat{y}_b + \eta \]
Test \( \hat{\delta}_1 = 0 \)

(5) If reject null hypothesis, then overfitting may be a problem - need to prune the model and check for outliers

(6) Repeat split sample experiment (Steps (1) to (4)) for 1000 times to get distribution and report the % of times the null is rejected – use to rank models, since test of \( \ast \_1 = 1 \) often fails
Summary of Results

<table>
<thead>
<tr>
<th></th>
<th>Using Own-Sample Predictions</th>
<th>Using Cross-Sample Predictions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Statistic</td>
<td>P *</td>
</tr>
<tr>
<td>OLS</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Log OLS (Homo)</td>
<td>7.65</td>
<td>0.707</td>
</tr>
<tr>
<td>Log OLS (Hetero)</td>
<td>2.84</td>
<td>0.284</td>
</tr>
<tr>
<td>GLM gamma/log</td>
<td>0.84</td>
<td>0.044</td>
</tr>
<tr>
<td>GLM poisson/log</td>
<td>0.70</td>
<td>0.029</td>
</tr>
</tbody>
</table>

* Represents proportion of 1000 replicates for which the F-statistic is significant at 5% level

- OLS model shows significant degree of overfitting (34.5% of the replicates fail Copas test using out-of-sample predictions)
- GLM poisson/log link model performs best (2.9% & 3.8% replicates fail Copas test within-sample & out-of-sample, respectively)

See Summary Table, p. 29
Tests for Normality of Residuals

Are the residuals normally distributed?

(1) Skewness for a normal distribution is zero
   Negative values – left skewed data
   Positive values – right skewed data

(2) Kurtosis for a standard normal distribution is 3

After OLS cost regression:

. sum res_ols, detail

Residuals

Percentiles   Smallest
1%    -6865.266     -6954.6
5%    -6285.974   -6933.082
10%    -5321.376    -6797.45       Obs                 200
25%    -3998.92   -6778.228       Sum of Wgt.         200
50%    -1597.021                   Mean           2.78e-06
Largest       Std. Dev.      5910.781
75%     2502.182    17393.61
90%     7335.55    17773.35       Variance       3.49e+07
95%     10781.04    25293.73       Skewness       1.960199
99%     21533.54    32942.74       Kurtosis        8.85317

. sktest res_ols

Skewness/Kurtosis tests for Normality

--- joint ------

Variable | Pr(Skewness)  Pr(Kurtosis)  adj chi2(2)  Prob>chi2
----------+--------------------------------------------------
res_ols  |   0.000        0.000         72.09      0.0000

NO FOR OLS
After log OLS cost regression:

```
.sum res_log, detail

Residuals

----------------------------------------
Percentiles    Smallest
1%    -1.879498   -2.256186
5%     -1.2353    -2.033567
10%    -1.025368   -1.725428      Obs                 200
25%    -.4435903   -1.623668      Sum of Wgt.         200
50%    -.055706    -7.28e-10      Largest
75%     .5797378   1.407953      Mean
90%     .9839813   1.572954      Std. Dev.      .745819
95%     1.207211   1.787229      Variance       .556246
99%     1.680091   1.860877      Skewness      -.1200774
      1.860877      Kurtosis     2.902033

.sktest res_log

Skewness/Kurtosis tests for Normality

----------------------------------------
Variable |  Pr(Skewness)   Pr(Kurtosis)  adj chi2(2)    Prob>chi2
-------------+-------------------------------------------------------
res_log      |  0.475           0.945           0.52        0.7707

YES FOR LOG
```
/* Modified Park Test */
. gen r2 = ((cost-yhat)^2)
. gen lnyhat = ln(yhat)

. glm r2 lnyhat , link(log) family(gamma) robust nolog

Generalized linear models                  No. of obs  =       200
Optimization     : ML: Newton-Raphson          Residual df  =       198
                  Scale parameter =   5.37055
Deviance         =  556.0966603                (1/df) Deviance =  2.808569
Pearson          =  1063.368955                (1/df) Pearson  =   5.37055

Variance function: V(u) = u^2                   [Gamma]
Link function    : g(u) = ln(u)                  [Log]
Standard errors  : Sandwich

Log pseudo-likelihood = -3667.729811            AIC          =   36.6973
BIC                   =-492.9701783

------------------------------------------------------------------------------
|               Robust
|             Coef.   Std. Err.     z  P>|z|      [95% Conf. Interval]
|-------------+--------------------------------------------------
r2    | .8059514   .6058605   1.33  0.183    -.3815133    1.993416
     lnyhat  | 10.04718   5.417169   1.85  0.064    -.5702812    20.66463
_cons    |                   |
------------------------------------------------------------------------------

.test lnyhat==1
   ( 1)  [r2]lnyhat = 1

         chi2(  1) =  0.100
     Prob > chi2 =  0.7488

.test lnyhat==2
   ( 1)  [r2]lnyhat = 2

         chi2(  1) =  3.880
     Prob > chi2 =  0.0487

.test lnyhat==3
   ( 1)  [r2]lnyhat = 3

         chi2(  1) =  13.109
     Prob > chi2 =  0.0003
From pp. 17

```
. /* PREGIBON LINK TEST */
. gen xbsq=xb^2
. glm cost xb xbsq, family(gamma) link(log) robust nolog

Generalized linear models                  No. of obs      =       200
Optimization     : ML: Newton-Raphson         Residual df     =       197
Scale parameter =  .5366878                 (1/df) Deviance =  .5128996
Deviance         =  101.0412175             (1/df) Pearson =  .5366878
Pearson          =   105.727504              
Variance function: V(u) = u^2               [Gamma]
Link function    : g(u) = ln(u)               [Log]
Standard errors  : Sandwich
Log pseudo-likelihood = -2008.934778        AIC             =  20.11935
BIC                   =-942.7273038

<table>
<thead>
<tr>
<th>Robust</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.</td>
</tr>
<tr>
<td>---------------</td>
<td>--------</td>
</tr>
<tr>
<td>cost</td>
<td>16.67377</td>
</tr>
<tr>
<td>xbsq</td>
<td>-.863838</td>
</tr>
<tr>
<td>_cons</td>
<td>-71.02776</td>
</tr>
</tbody>
</table>

. test xbsq
( 1) [cost]xbsq = 0

    chi2(  1) =    1.39
    Prob > chi2 =   0.2389
```
/* MODIFIED HOSMER LEMESHOW TEST */
.xtile xbtile=xb, nq(10)
.tab xbtile, gen(xbt)

<table>
<thead>
<tr>
<th>10 quantiles of xb</th>
<th>Freq.</th>
<th>Percent</th>
<th>Cum.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20</td>
<td>10.00</td>
<td>10.00</td>
</tr>
<tr>
<td>2</td>
<td>20</td>
<td>10.00</td>
<td>20.00</td>
</tr>
<tr>
<td>3</td>
<td>20</td>
<td>10.00</td>
<td>30.00</td>
</tr>
<tr>
<td>4</td>
<td>20</td>
<td>10.00</td>
<td>40.00</td>
</tr>
<tr>
<td>5</td>
<td>20</td>
<td>10.00</td>
<td>50.00</td>
</tr>
<tr>
<td>6</td>
<td>20</td>
<td>10.00</td>
<td>60.00</td>
</tr>
<tr>
<td>7</td>
<td>20</td>
<td>10.00</td>
<td>70.00</td>
</tr>
<tr>
<td>8</td>
<td>20</td>
<td>10.00</td>
<td>80.00</td>
</tr>
<tr>
<td>9</td>
<td>20</td>
<td>10.00</td>
<td>90.00</td>
</tr>
<tr>
<td>10</td>
<td>20</td>
<td>10.00</td>
<td>100.00</td>
</tr>
<tr>
<td>Total</td>
<td>200</td>
<td>100.00</td>
<td></td>
</tr>
</tbody>
</table>

.reg res xbt1 xbt2 xbt3 xbt4 xbt5 xbt6 xbt7 xbt8 xbt9 xbt10, nocons robust

Regression with robust standard errors
Number of obs = 200
F( 10,   190) = 0.65
Prob > F = 0.7652
R-squared = 0.0293
Root MSE = 5983.8

|      | Coef.    | Std. Err. | t     | P>|t|  | 95% Conf. Interval |
|------|----------|-----------|-------|------|---------------------|
| xbt1 | -858.0814| 1340.247  | -0.64 | 0.523| -3501.756           | 1785.593 |
| xbt2 | -1082.153| 814.6391  | -1.33 | 0.186| -2689.051           | 524.7458 |
| xbt3 | 2533.632 | 1489.641  | 1.70  | 0.091| -404.7272           | 5471.991 |
| xbt4 | -390.9682| 870.4381  | -0.45 | 0.654| -2107.932           | 1325.995 |
| xbt5 | -10.61213| 1267.625  | -0.01 | 0.993| -2521.927           | 2500.702 |
| xbt6 | 960.5702 | 975.5737  | 0.98  | 0.326| -963.7763           | 2884.917 |
| xbt7 | 170.9792 | 1267.625  | 0.13  | 0.893| -2329.447           | 2671.405 |
| xbt8 | -330.7193| 1198.243  | -0.28 | 0.783| -2694.287           | 2032.848 |
| xbt9 | -519.5159| 1630.368  | -0.32 | 0.750| -3735.462           | 2696.43  |
| xbt10| -686.1769| 2047.685  | -0.34 | 0.738| -4725.293           | 3352.94  |

.test xbt1 xbt2 xbt3 xbt4 xbt5 xbt6 xbt7 xbt8 xbt9 xbt10
( 1) xbt1 = 0
( 2) xbt2 = 0
( 3) xbt3 = 0
( 4) xbt4 = 0
( 5) xbt5 = 0
( 6) xbt6 = 0
( 7) xbt7 = 0
( 8) xbt8 = 0
( 9) xbt9 = 0
(10) xbt10 = 0

F( 10,   190) = 0.65
Prob > F = 0.7652
### SUMMARY OF DIAGNOSTIC TEST RESULTS

<table>
<thead>
<tr>
<th>Model</th>
<th>Pregibon Link</th>
<th>Modified Hosmer-Lemeshow</th>
<th>Pearson correlation</th>
<th>Modified Park</th>
<th>Copas (Rank)</th>
</tr>
</thead>
<tbody>
<tr>
<td>OLS</td>
<td>Passed</td>
<td>Passed</td>
<td>Passed</td>
<td>-</td>
<td>3</td>
</tr>
<tr>
<td>Log OLS (Homo)</td>
<td>Failed</td>
<td>Passed</td>
<td>Failed</td>
<td>-</td>
<td>5</td>
</tr>
<tr>
<td>Log OLS (Hetero)</td>
<td>Failed</td>
<td>Passed</td>
<td>Failed</td>
<td>-</td>
<td>4</td>
</tr>
<tr>
<td>GLM gamma/log</td>
<td>Passed</td>
<td>Passed</td>
<td>Passed</td>
<td>Failed</td>
<td>2</td>
</tr>
<tr>
<td>GLM poisson/log</td>
<td>Passed</td>
<td>Passed</td>
<td>Passed</td>
<td>Passed</td>
<td>1</td>
</tr>
</tbody>
</table>

- GLM poisson/log link seems to have performed well on all tests as compared to other models!
- However, in this example dataset, the incremental costs estimated from the GLM poisson ($4,018) and untransformed OLS ($4,027) were not very different [although Log OLS (homoskedastic) was quite different ($6649)]
- In other examples, we have seen substantial differences across models!
APPENDIX 3.  STATA PROGRAMS

This section provides sample programs for the topics discussed in this lecture.

1. Multivariable.do
2. Insideboot.do
3. Bootmulti.do
4. Bootresult.do
5. Multivariable_tests.do
6. Mse_ape_copastest.do

1. Multivariable.DO
****************************THIS PROGRAM DOES THE MULTIVARIABLE ANALYSES***************************
clear
set memory 10000
capture log closelog using multivariable.log, replace
use mdmcea
******control variables
global ivar "age ejfract sex etiology race"

clear
set memory 10000
capture log close
log using multivariable.log, replace

use mdmcea

**create log of costs
gen lcost=log(cost)
gen tmptreat=treat

******OLS
regress cost treat $ivar
predict ols_t
replace treat=0
predict ols_0
replace treat=1

tabstat cost if treat==1
tabstat cost if treat==0
tabstat ols_1 ols_0 ols_dif

***log costs
regress lcost treat $ivar

***smearing factor
capture drop res
predict res, r
replace res=exp(res)
sum res
scalar sm=r(mean)
sum res if treat==0
scalar sm0=r(mean)
sum res if treat==1
scalar sm1=r(mean)
display sm
display sm0
display sm1

replace treat=0
predict l0
replace treat=1
predict l1
replace treat=tmptreat

gen lols_0=exp(l0)*sm0
gen lols_1=exp(l1)*sm1
gen lols_t=lols_0 if treat==0
replace lols_t=lols_1 if treat==1
gen lols_dif=lols_1-lols_0

tabstat lols_1 lols_0 lols_dif

****assume homoskedastic!
gen lolsho_0=exp(l0)*sm
gen lolsho_1=exp(l1)*sm
gen lolsho_dif=lolsho_1-lolsho_0
gen lolsho_t=lolsho_0 if treat==0
replace lolsho_t=lolsho_1 if treat==1

tabstat lolsho_1 lolsho_0 lolsho_dif

*****glm model (gamma/log link)
glm cost treat $ivar, family(gamma) link(log)
*predict gam_t
replace treat=0
predict gam_0
replace treat=1
predict gam_1
gen gam_dif=gam_1-gam_0
replace treat=tmtreat
gen gam_t=gam_0 if treat==0
replace gam_t=gam_1 if treat==1

tabstat gam_1 gam_0 gam_dif

*****glm model (poisson/log link)
glm cost treat $ivar, family(poisson) link(log)
*predict pois_t
replace treat=0
predict pois_0
replace treat=1
predict pois_1
gen pois_dif=pois_1-pois_0
replace pois_t=pois_1 if treat==0
replace pois_t=pois_1 if treat==1

tabstat pois_1 pois_0 pois_dif

sum ols_1 lols_1 lolsho_1 gam_1 pois_1
sum ols_0 lols_0 lolsho_0 gam_0 pois_0
sum ols_dif lols_dif lolsho_dif gam_dif pois_dif

log close

2. Insideboot.do
******************************
THIS PROGRAM IS JUST LIKE multivariable.do
*********BUT IT IS A VERSION THAT IS CALLED BY bootmulti.do
*********IN ORDER TO GET THE CONFIDENCE INTERVALS
******************************

*****control variables
global ivar "age ejfract sex etiology race"

**create log of costs
gen lcost=log(cost)

***
gen tmptreat=treat

*****OLS
regress cost treat $ivar
*predict ols_t
replace treat=0
predict ols_0
replace treat=1
predict ols_1
replace treat=tmptreat
gen ols_dif=ols_1-ols_0

tabstat ols_1 ols_0 ols_dif

tabstat cost if treat==1
tabstat cost if treat==0
tabstat ols_1 ols_0 ols_dif

***log costs
regress lcost treat $ivar

***smearing factor
capture drop res
predict res, r
replace res=exp(res)
sum res
scalar sm=r(mean)
sum res if treat==0
scalar sm0=r(mean)
scalar sm1=r(mean)
display sm
display sm0
display sm1
replace treat=0
predict lc0
replace treat=1
predict lc1
replace treat=tmpctreat
gen lols_0=exp(lc0)*sm0
gen lols_1=exp(lc1)*sm1
gen lols_t=lols_0 if treat==0
replace lols_t=lols_1 if treat==1
gen lols_diff=lols_1-lols_0
tabstat lols_1 lols_0 lols_diff

****assume homoskedastic!
gen lolsho_0=exp(lc0)*sm
gen lolsho_1=exp(lc1)*sm
gen lolsho_diff=lolsho_1-lolsho_0
gen lolsho_t=lolsho_0 if treat==0
replace lolsho_t=lolsho_1 if treat==1
tabstat lolsho_1 lolsho_0 lolsho_diff

****glm model (poisson/log link)
glm cost treat $ivar, family(poisson) link(log)
*predict pois_t
replace treat=0
predict pois_0
replace treat=1
predict pois_1
gen pois_diff=pois_1-pois_0
replace treat=tmpctreat
gen pois_t=pois_0 if treat==0
replace pois_t=pois_1 if treat==1

tabstat pois_1 pois_0 pois_diff

sum ols_1 lols_1 lolsho_1 gam_1 pois_1
sum ols_0 lols_0 lolsho_0 gam_0 pois_0
sum ols_diff lols_diff lolsho_diff gam_diff pois_diff

sum ols_1, meanonly
gen ols1=r(mean)
sum ols_0, meanonly
gen ols0=r(mean)
sum lolsho_1, meanonly
gen lolsho1=r(mean)
sum lolsho_0, meanonly
gen lolsho0=r(mean)
sum lolsho_1, meanonly
gen lolsho1=r(mean)
sum lolsho_0, meanonly
gen lolsho0=r(mean)

sum gam_1, meanonly
gen gam1=r(mean)
sum gam_0, meanonly
gen gam0=r(mean)

sum pois_1, meanonly
ngen pois1=r(mean)
sum pois_0, meanonly
ngen pois0=r(mean)
keep ols1 ols0 lols1 lols0 lolsho1 lolsho0 gam1 gam0 pois1 pois0
keep if _n==1

3. Bootmulti.do

***** THIS PROGRAM RUNS insideboot.do TO GET bootresult.dta
***** AFTER RUNNING THIS PROGRAM, RUN bootresult.do
***** TO GET CONFIDENCE INTERVALS
version 7
* BOOTSTRAP METHOD

capture erase temp1.dta
capture program drop loopm
program define loopm
  local i=`1'
  while `i'>0 {
    local i = `i' - 1
    use mdmcea
    preserve
    keep if treat==0
    bsample
    save temp1,replace
    restore
    keep if treat==1
    bsample
    append using temp1
    save temp1,replace
  }
end
set seed 123456789
set obs 1
gen temp = 1
save bootm, replace
quietly loopm 2000
drop if temp==1
drop temp
capture erase bootresult.dta
save bootresult, replace
capture erase templ.dta
capture erase bootm.dta

capture log closelog using bootresult.log, replace

4. Bootresult.do

********************************************************************
****THIS PROGRAM GIVES THE CI'S FOR THE MULTIVARIABLE
**REGRESSIONS
**RUN THIS PROGRAM AFTER GETTING bootresult.dta
********************************************************************

clear
set memory 10000
capture log close
log using bootresult.log, replace
use bootresult

gen olsdif=ols1-ols0
gen lolsdif=lols1-lols0
gen lhodif=lolsho1-lolsho0
gen gamdif=gam1-gam0
gen poisdif=pois1-pois0

capture program drop get95ci
program define get95ci
  sort `1'
  list `1' if _n==1+round((_N*0.025),0)|(_n==_N-round((_N*0.025),0))
end

clear
set more 1

quietly sum `1'
display "mean " r(mean)
display "st err " r(sd)
display "t-stat " abs(r(mean)/r(sd))

e
get95ci olsdif
get95ci lolsdif
get95ci lhsdif
get95ci gamdif
get95ci poisdif

log close

5. Multivariable_tests.do

******************************************************************************
**********THIS PROGRAM CONDUCTS THE STATISTICAL TESTS FOLLOWING MULTIVARIABLE ANALYSES
******************************************************************************
clear
set memory 10000
capture log close
log using multivariable_test.log, replace

use mdmcea, clear

*****control variables***************
global ivar "age ejfract sex etiology race"
******************************************************************************
global lst1 "xbt1 xbt2 xbt3 xbt4 xbt5 xbt6 xbt7 xbt8 xbt9 xbt10"

**create log of costs
gen lcost=log(cost)
gen tmptreat=treat

*****OLS
regress cost treat $ivar

predict xb, xb
gen mu=xb
gen res= cost-mu
summ res

/* PEARSON CORR TEST */
pwcorr mu res, sig

/* PREGIBON LINK TEST */
qui gen xbsq=xb^2
reg cost xb xbsq, robust
test xbsq

/* MODIFIED HOSMER LEMESHOW TEST */
qui xtile xbtile=xb, nq(10)
qui tab xbtile, gen(xbt)
reg res $lst1, nocons robust
test $lst1

drop xb mu res xbtile xbt1-xbt10 xbsq

***log costs
regress lcost treat $ivar
predict xb, xb

***smearing factor
capture drop res
predict res, r
replace res=exp(res)
sum res
scalar sm=r(mean)
sum res if treat==0
scalar sm0=r(mean)
sum res if treat==1
scalar sm1=r(mean)
drop res

display sm
display sm0
display sm1
replace treat=0
predict lc0
replace treat=1
predict lcl
replace treat=tkmp_treat

gen lols_0=exp(lc0)*sm0

gen lols_1=exp(lc1)*sm1

gen lols_t=lols_0 if treat==0
    replace lols_t=lols_1 if treat==1

gen lols_res=cost-lols_t

gen res=lols_res

gen mu=lols_t

/* PEARSON CORR TEST */
pwcorr mu res, sig 

/* PREGIBON LINK TEST */
gen xbsq=xb^2
reg lcost xb xbsq, robust

test xbsq

/* MODIFIED HOSMER LEMESHOW TEST */
xtile xbtile=xb, nq(10)
qui tab xbtile, gen(xbt)
reg res $lst1, nocons robust

test $lst1

drop xb mu res xbtile xbt1-xbt10 xbsq r2

****glm model (gamma/log link)
glm cost treat $ivar, family(gamma) link(log)
predict xb, xb
replace treat=0
predict gam_0
replace treat=1
predict gam_1

gen gam_dif=gam_1-gam_0
replace treat=tkmp_treat

gen gam_t=gam_0 if treat==0
replace gam_t=gam_1 if treat==1

gen gam_res=cost-gam_t

gen mu=gam_t

gen res=cost-gam_t

/* PEARSON CORR TEST */
pwcorr mu res, sig

/* PREGIBON LINK TEST */
gen xbsq=xb^2

glm cost xb xbsq, family(gamma) link(log) robust nolog

test xbsq

/* MODIFIED HOSMER LEMESHOW TEST */
xtile xbtile=xb, nq(10)
tab xbtile, gen(xbt)
reg res $lst1, nocons robust

test $lst1

/* Modified Park Test */
gen r2 = ((cost-mu)^2)
gen lnmu = ln(mu)
glm r2 lnmu, link(log) family(gamma) robust nolog
test lnmu==0
test lnmu==1
test lnmu==2
test lnmu==3

drop xb mu res xbtile xbt1-xtb10 xbsq r2 lnmu

****glm model (poisson/log link)
glm cost treat $ivar, family(poisson) link(log)
predict xb, xb
replace treat=0
predict pois_0
replace treat=1
predict pois_1
gen pois_dif=pois_1-pois_0
replace treat=tmpetreat
gen pois_t=pois_0 if treat==0
replace pois_t=pois_1 if treat==1

gen pois_res=cost-pois_t
gen mu=pois_t
gen res=cost-pois_t

/* PEARSON CORR TEST */
pwcorr mu res, sig

/* FREGIBON LINK TEST */
gen xbsq=xb^2
glm cost xb xbsq, family(gamma) link(log) robust nolog
test xbsq

/* MODIFIED HOSMER LEMESHOW TEST */
xtile xbtile=xb, nq(10)
tab xbtile, gen(xbt)
reg res $ist1, nocons robust
test $lstat1

/* Modified Park Test */
gen r2 = ((cost-mu)^2)
gen lnmu = ln(mu)
glm r2 lnmu, link(log) family(gamma) robust nolog
test lnmu==0
test lnmu==1
test lnmu==2
test lnmu==3

drop xb mu res xbtile xbt1-xtb10 xbsq r2 lnmu

log close

6. Mse_ape_copastest.do
******************************************************************************************
***THIS PROGRAM CONDUCTS THE SPLIT SAMPLE EXPERIMENT TO
**EXAMINE WITHIN-SAMPLE & OUT-OF-SAMPLE FIT
******************************************************************************************
clear all
capture log close
set memory 500000
set more 1

global seed=5735766

clear all
set obs 1

global seed = $seed +1000
save mcmcea_splithalf, replace
log using mcmcea_splithalf.log, replace

global seed = $seed +1000
set seed $seed

global seed = $seed +1000
set seed $seed
save `tmp1', replace
global seed = `seed'

use mcmcea, clear

gen tmptreat=treat
gen lcost=ln(cost)

global seed = $seed +10000
set seed $seed

global seed = $seed +10000
set seed $seed

global seed = $seed +10000
set seed $seed

keep patid
keep if patid != patid[_n-1]
sample 50
gen split=1
sort patid
save 'tmp2', replace
merge patid using 'tmp1'
drop _merge
replace split=0 if split==. /* Split data into two
equal parts */

*****control variables********************
global ivar "age ejfract sex etiology race"*******************************************

** For demonstration purposes Copas test for only OLS
** regression shown in this sample program - Can similarly
** include other multivariable models

***** OLS
regress cost treat $ivar if split==1
predict ols1_t
replace treat=0
predict ols1_0
replace treat=1
predict ols1_1
replace treat=tmptreat

/* In-sample Prediction */
gen own_ols = ols1_t if split==1
replace own_ols=. if split==0

/* Out-sample Prediction */
gen cross_ols = ols1_t if split==0
replace cross_ols=. if split==1

/* Efficiency Measures */
gen own_ols_mse = (cost-own_ols)^2
gen own_ols_ape = abs(cost-own_ols)
gen cross_ols_mse = (cost-cross_ols)^2
gen cross_ols_ape = abs(cost-cross_ols)

summ own_ols_mse
gen own_olsmse = r(mean)
summ own_ols_ape
gen own_olsape = r(mean)

summ cross_ols_mse
gen cross_olsmse = r(mean)
summ cross_ols_ape
gen cross_olsape = r(mean)

/* COPAS TEST */

**OLS
reg own_res own_ols, robust
gen own_b0_ols=_b[_cons]
gen own_b1_ols=_b[own_ols]
gen own_ols_copusmse=e(rmse)^2
test _b[_cons]=0, notest
test _b[own_ols]=0, accumulate
gen own_fstat_ols=r(F)
gen own_pval_ols=(r(p)<=0.05)

reg cross_res cross_ols, robust
gen cross_b0_ols=_b[_cons]
gen cross_b1_ols=_b[cross_ols]
gen cross_ols_copusmse=e(rmse)^2
test _b[_cons]=0, notest
test _b[cross_ols]=0, accumulate
gen cross_fstat_ols=r(F)
gen cross_pval_ols=(r(p)<=0.05)

drop own_res cross_res

gen simul=`k'
keep if _n==1
keep simul own_b0_ols own_b1_ols own_fstat_ols own_pval_ols
own_ols_copusmse/* */cross_b0_ols cross_b1_ols
cross_fstat_ols cross_pval_ols cross_ols_copusmse
append using mdmcea_split
save mdmcea_split, replace

ci own_b0_ols own_b1_ols own_fstat_ols own_pval_ols /*
*/ own_ols_copusmse ci cross_b0_ols cross_b1_ols
cross_fstat_ols /*
*/ cross_pval_ols cross_ols_copusmse

log close