# DeLong: Teaching Economics

November 29, 2019

Last edited: 2019-10-12

# 1 Deep Roots of Relative Development

## 1.1 Due ???? via upload to ???

### 1.1.1 J. Bradford DeLong

## 1.2 Derived from QuantEcon: Linear Regression in Python: https://python.quantecon.org/ols.html

You should have gotten to this point vis this link:

### 1.2.1 Table of Contents

1.

2.

3.

In [1]:
```
#libraries:

!pip install linearmodels

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import statsmodels.api as sm
from statsmodels.iolib.summary2 import summary_col
from linearmodels.iv import IV2SLS


# inline graphics

%matplotlib inline
```

In [2]:
```
ajr_df = pd.read_csv('https://delong.typepad.com/files/ajr.csv')
ajr_df.head()
```

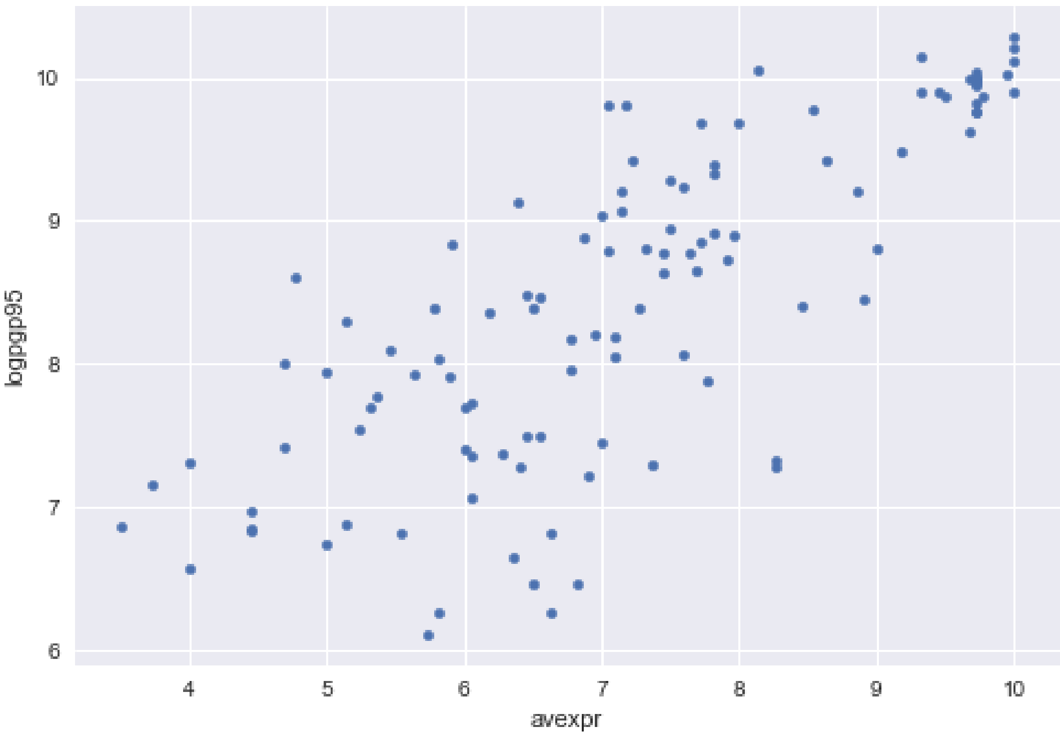
```
Out[2]:   shortnam   euro1900  excolony    avexpr  logpgp95  cons1  cons90  democ00a  \
0        AFG   0.000000       1.0       NaN       NaN    1.0     2.0       1.0
1        AGO   8.000000       1.0  5.363636  7.770645    3.0     3.0       0.0
2        ARE   0.000000       1.0  7.181818  9.804219    NaN     NaN       NaN
3        ARG  60.000004       1.0  6.386364  9.133459    1.0     6.0       3.0
4        ARM   0.000000       0.0       NaN  7.682482    NaN     NaN       NaN

   cons00a    extmort4    logem4  loghjypl  baseco
0      1.0   93.699997  4.540098       NaN     NaN
1      1.0  280.000000  5.634789 -3.411248     1.0
2      NaN         NaN       NaN       NaN     NaN
3      3.0   68.900002  4.232656 -0.872274     1.0
4      NaN         NaN       NaN       NaN     NaN
```

Let's use a scatterplot to see whether any obvious relationship exists between GDP per capita and the protection against expropriation index:

```
In [3]: plt.style.use('seaborn')

        ajr_df.plot.scatter(x='avexpr', y='logpgp95')
        plt.show()
```




Let's add three-letter country labels to the points:

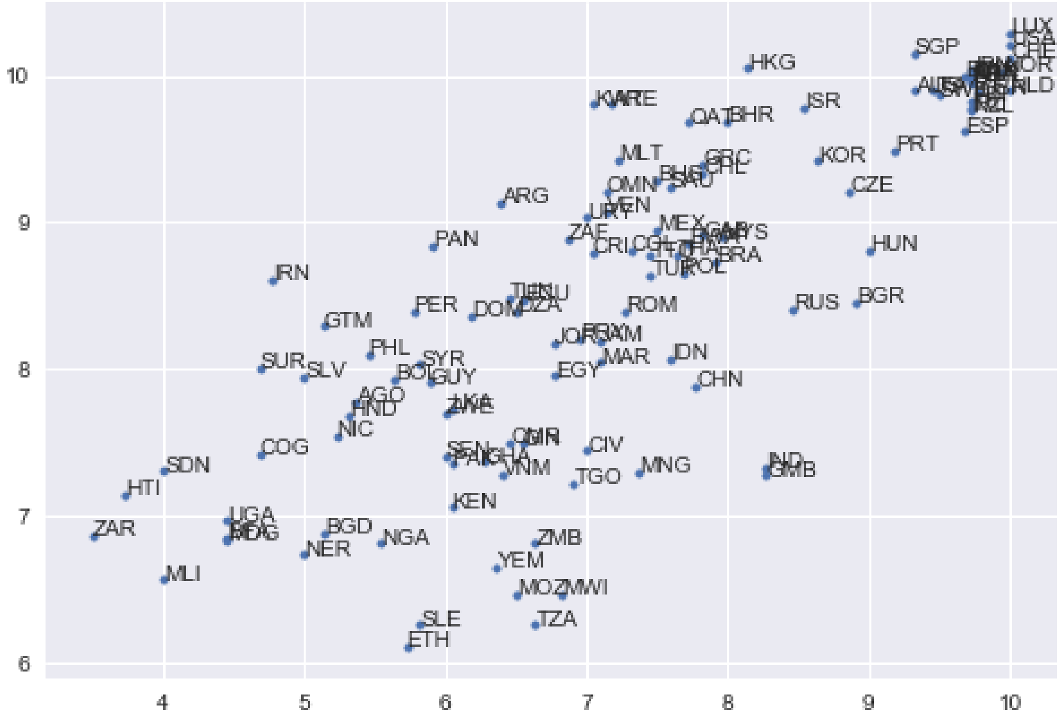
```
In [4]: x = ajr_df['avexpr'].tolist()
        y = ajr_df['logpgp95'].tolist()
        labels = ajr_df['shortnam'].tolist()

        fig, ax = plt.subplots()
        ax.scatter(x, y, marker='.')
```

```
for i, txt in enumerate(labels):
    ax.annotate(txt, (x[i], y[i]))

plt.show()

# no, I do not understand why the data and labels need to be
# coerced into a list before ax.annotate will do its thing...
```




Let’s fit a linear model to this scatter:

1. $\ln(pgp_9 5)_i = \beta_0 + \beta_1(avexpr_i) + u_i$

$\beta_1$ is the slope of the linear trend line, representing the marginal association of protection against against expropriation risk with log GDP per capita

$u_i$ is an error term.

Fitting this linear model chooses a straight line that best fits the data in a least-squares, as in the following plot (Figure 2 in AJR):

In [5]:
```
# dropping NA's is required to use numpy's polyfit...
# using only 'base sample' for plotting purposes...


ajr_df = ajr_df.dropna(subset=['logpgp95', 'avexpr'])
ajr_df = ajr_df[ajr_df['baseco'] == 1]

x = ajr_df['avexpr'].tolist()
y = ajr_df['logpgp95'].tolist()
labels = ajr_df['shortnam'].tolist()
```

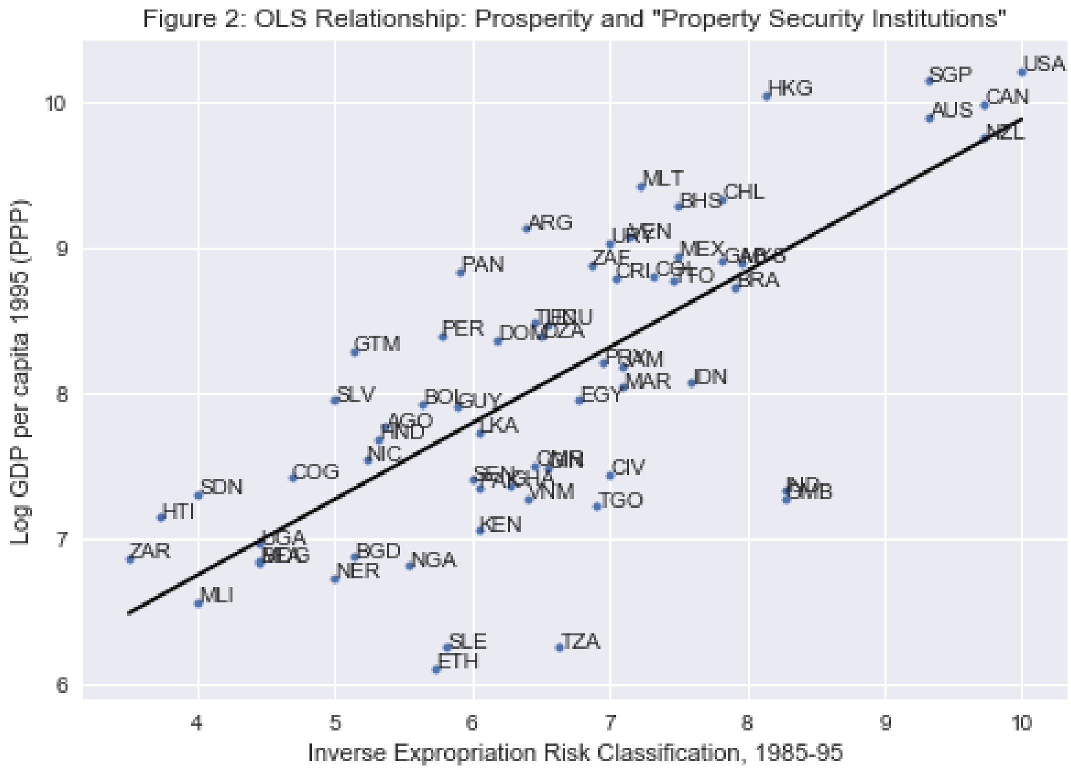
```
fig, ax = plt.subplots()
ax.scatter(x, y, marker='.')

for i, txt in enumerate(labels):
    ax.annotate(txt, (x[i], y[i]))

ax.plot(np.unique(x),
         np.poly1d(np.polyfit(x, y, 1))(np.unique(x)),
         color='black')

ax.set_xlabel('Inverse Expropriation Risk Classification, 1985-95')
ax.set_ylabel('Log GDP per capita 1995 (PPP)')
ax.set_title('Figure 2: OLS Relationship: Prosperity and "Property Security
Institutions"')

plt.show()
```




To estimate the constant term $\beta_0$, we need to add a column of 1's to our dataframe so that we can use statsmodels's OLS routines:

```
In [6]: ajr_df['constant'] = 1

        regression_1 = sm.OLS(endog=ajr_df['logpgp95'],
                              exog=ajr_df[['constant', 'avexpr']],
                              missing='drop')
        results_1 = regression_1.fit()
        print(results_1.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:               logpgp95   R-squared:                       0.540
```

```
Model:                            OLS   Adj. R-squared:                  0.533
Method:                 Least Squares   F-statistic:                     72.82
Date:                Fri, 29 Nov 2019   Prob (F-statistic):           4.72e-12
Time:                        17:07:14   Log-Likelihood:                -68.168
No. Observations:                  64   AIC:                             140.3
Df Residuals:                      62   BIC:                             144.7
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
constant       4.6604      0.409     11.408      0.000       3.844       5.477
avexpr         0.5221      0.061      8.533      0.000       0.400       0.644
==============================================================================
Omnibus:                        7.098   Durbin-Watson:                   2.064
Prob(Omnibus):                  0.029   Jarque-Bera (JB):                6.657
Skew:                          -0.781   Prob(JB):                       0.0358
Kurtosis:                       3.234   Cond. No.                         31.2
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

We extend our bivariate regression model to a multivariate regression model by adding in other factors correlated with $\ln(pgp_95)_i$:

- climate, as proxied by latitude
- the different culture and history of different continents

latitude is used to proxy this differences that affect both economic performance and institutions, eg. cultural, historical, etc.; controlled for with the use of continent dummies Let's estimate some of the extended models considered in the paper (Table 2) using data from

In [7]:
```
ajr2_df = pd.read_csv('https://delong.typepad.com/files/ajr2.csv')
ajr2_df['constant'] = 1

X1 = ['constant', 'avexpr']
X2 = ['constant', 'avexpr', 'lat_abst']
X3 = ['constant', 'avexpr', 'lat_abst', 'asia', 'africa', 'other']

regression_2 = sm.OLS(ajr2_df['logpgp95'], ajr2_df[X1], missing='drop').fit()
regression_3 = sm.OLS(ajr2_df['logpgp95'], ajr2_df[X2], missing='drop').fit()
regression_4 = sm.OLS(ajr2_df['logpgp95'], ajr2_df[X3], missing='drop').fit()

info_dict={'R-squared' : lambda x: f"{x.rsquared:.2f}",
           'No. observations' : lambda x: f"{int(x.nobs):d}"}

results_table = summary_col(results=[regression_2, regression_3, regression_4],
                            float_format='%0.2f',
                            stars = True,
                            model_names=['Model 1',
                                         'Model 3',
                                         'Model 4'],
                            info_dict=info_dict,
                            regressor_order=['constant',
                                             'avexpr',
```

```python
                                                   'lat_abst',
                                                   'asia',
                                                   'africa'])

results_table.add_title('Table 2 - OLS Regressions')

print(results_table)
```

```
            Table 2 - OLS Regressions
=========================================
                 Model 1 Model 3 Model 4
-----------------------------------------
constant         4.63*** 4.87*** 5.85***
                 (0.30)  (0.33)  (0.34)
avexpr           0.53*** 0.46*** 0.39***
                 (0.04)  (0.06)  (0.05)
lat_abst                 0.87*   0.33
                         (0.49)  (0.45)
asia                             -0.15
                                 (0.15)
africa                           -0.92***
                                 (0.17)
other                            0.30
                                 (0.37)
R-squared        0.61    0.62    0.72
No. observations 111     111     111
=========================================
Standard errors in parentheses.
* p<.1, ** p<.05, ***p<.01
```

In [8]:
```python
# Dropping NA's is required to use numpy's polyfit
df1_subset2 = ajr_df.dropna(subset=['logem4', 'avexpr'])

X = df1_subset2['logem4']
y = df1_subset2['avexpr']
labels = df1_subset2['shortnam']

# Replace markers with country labels
fig, ax = plt.subplots()
ax.scatter(X, y, marker='')

for i, label in enumerate(labels):
    ax.annotate(label, (X.iloc[i], y.iloc[i]))

# Fit a linear trend line
ax.plot(np.unique(X),
         np.poly1d(np.polyfit(X, y, 1))(np.unique(X)),
         color='black')

ax.set_xlim([1.8,8.4])
ax.set_ylim([3.3,10.4])
ax.set_xlabel('Log of Settler Mortality')
ax.set_ylabel('Average Expropriation Risk 1985-95')
ax.set_title('Figure 3: First-stage relationship between settler mortality \
    and expropriation risk')
plt.show()
```


```
In [9]: df4 = pd.read_stata('https://github.com/QuantEcon/QuantEcon.lectures.code/raw/
 ↪master/ols
        /maketable4.dta')
        df4 = df4[df4['baseco'] == 1]
        df4['const'] = 1

        iv = IV2SLS(dependent=df4['logpgp95'],
                    exog=df4['const'],
                    endog=df4['avexpr'],
                    instruments=df4['logem4']).fit(cov_type='unadjusted')

        print(iv.summary)
```

```
                          IV-2SLS Estimation Summary
==============================================================================
Dep. Variable:               logpgp95   R-squared:                      0.1870
Estimator:                    IV-2SLS   Adj. R-squared:                 0.1739
No. Observations:                  64   F-statistic:                    37.568
Date:                Fri, Nov 29 2019   P-value (F-stat)                0.0000
Time:                        17:07:18   Distribution:                  chi2(1)
Cov. Estimator:            unadjusted

                             Parameter Estimates
==============================================================================
            Parameter  Std. Err.     T-stat    P-value    Lower CI    Upper CI
------------------------------------------------------------------------------
const          1.9097     1.0106     1.8897     0.0588     -0.0710      3.8903
avexpr         0.9443     0.1541     6.1293     0.0000      0.6423      1.2462
==============================================================================

Endogenous: avexpr
```

```
Instruments: logem4
Unadjusted Covariance (Homoskedastic)
Debiased: False
```

## 2 Catch Our Breath—Further Notes:

https://tinyurl.com/20190119a-delong

- weblog support: https://github.com/braddelong/LS2019/blob/master/Deep-Roots-of-Relative-Development.ipynb
- nbViewer: https://nbviewer.jupyter.org/github/braddelong/LS2019/blob/master/Deep-Roots-of-Relative-Development.ipynb
- datahub: http://datahub.berkeley.edu/user-redirect/interact?account=braddelong&repo=LS2019&branch=master&path=Deep-Roots-of-Relative-Development.ipynb

In [10]:
```
pwt91_df = pd.read_csv('https://delong.typepad.com/files/pwt91-data.csv')
```

In [11]:
```
pwt91_df.head()
```

Out[11]:
```
  countrycode country   currency_unit  year  rgdpe  rgdpo  pop  emp  avh  hc  \
0         ABW   Aruba  Aruban Guilder  1950    NaN    NaN  NaN  NaN  NaN NaN
1         ABW   Aruba  Aruban Guilder  1951    NaN    NaN  NaN  NaN  NaN NaN
2         ABW   Aruba  Aruban Guilder  1952    NaN    NaN  NaN  NaN  NaN NaN
3         ABW   Aruba  Aruban Guilder  1953    NaN    NaN  NaN  NaN  NaN NaN
4         ABW   Aruba  Aruban Guilder  1954    NaN    NaN  NaN  NaN  NaN NaN

   …  csh_x  csh_m  csh_r  pl_c  pl_i  pl_g  pl_x  pl_m  pl_n  pl_k
0  …    NaN    NaN    NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN
1  …    NaN    NaN    NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN
2  …    NaN    NaN    NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN
3  …    NaN    NaN    NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN
4  …    NaN    NaN    NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN

[5 rows x 52 columns]
```