REPLICATION OF MRW 1992

In [5]:
```
import pandas as pd
import numpy as np
import statsmodels.api as sm
from statsmodels.iolib.summary2 import summary_col
import matplotlib.pyplot as plt
```

In [9]:
```
data = pd.read_csv ('mrw_2019.csv')
```

In [10]:
```
data= data.drop(data.columns[0], axis =1)
data= data.set_index('country')
data
```

Out[10]:

country	n	i	o	rgdpw1995	rgdpw2019	popgrowth	inv/output	school	rdgp_1995
Angola	1	0	0	4765.312781	13706.910920	0.032961	0.187624	1.481984	3.655082e+04
United Arab Emirates	0	0	0	172514.412000	117325.749600	0.014492	0.306448	2.746695	2.263044e+05
Argentina	1	1	0	35789.270760	48037.396490	0.009457	0.104180	3.096804	4.235772e+05
Australia	1	1	1	71766.792400	99574.429010	0.012252	0.218022	3.549666	5.895985e+05
Austria	1	1	1	72484.506080	109448.679100	0.007166	0.299841	3.381046	2.598537e+05
...	...	...	...	...	...	...	...	...	...
Uruguay	1	1	0	29034.909110	44913.446540	0.003609	0.151197	2.776406	4.000406e+04
United States	1	1	1	88413.211370	131778.647400	0.006019	0.218466	3.749341	1.126939e+07
Venezuela (Bolivarian Republic of)	1	1	0	23115.316190	612.820827	-0.012853	0.110155	2.893462	1.772251e+05
Yemen	0	0	0	3822.629989	9048.092216	0.023273	0.374125	1.842989	1.108384e+04
Zimbabwe	1	1	0	12712.491900	6191.766957	0.014313	0.074876	2.713408	6.422392e+04

105 rows × 17 columns

In [11]:
```
data['ln(n+g+\u03B4)']=np.log(data['popgrowth']/100 + 0.05) # Authors have assumed t
data ['ln(inv)']= np.log(data['inv/output']/100)
data ['ln(inv)-ln(n+g+\u03B4)']= data['ln(inv)'] - data['ln(n+g+\u03B4)']
data ['ln(sch)']=np.log(data['school']/100)
data ['ln(sch)-ln(n+g+\u03B4)'] = data['ln(sch)']-data['ln(n+g+\u03B4)']
data ['ln_95'] = np.log(data['rgdpw1995'])
data ['ln_19'] = np.log(data['rgdpw2019'])
data ['constant']=1

data
```

Out[11]:

country	n	i	o	rgdpw1995	rgdpw2019	popgrowth	inv/output	school	rdgp_1995
Angola	1	0	0	4765.312781	13706.910920	0.032961	0.187624	1.481984	3.655082e+04
United Arab Emirates	0	0	0	172514.412000	117325.749600	0.014492	0.306448	2.746695	2.263044e+05
Argentina	1	1	0	35789.270760	48037.396490	0.009457	0.104180	3.096804	4.235772e+05
Australia	1	1	1	71766.792400	99574.429010	0.012252	0.218022	3.549666	5.895985e+05
Austria	1	1	1	72484.506080	109448.679100	0.007166	0.299841	3.381046	2.598537e+05
...	...	...	...	...	...	...	...	...	...
Uruguay	1	1	0	29034.909110	44913.446540	0.003609	0.151197	2.776406	4.000406e+04
United States	1	1	1	88413.211370	131778.647400	0.006019	0.218466	3.749341	1.126939e+07
Venezuela (Bolivarian Republic of)	1	1	0	23115.316190	612.820827	-0.012853	0.110155	2.893462	1.772251e+05
Yemen	0	0	0	3822.629989	9048.092216	0.023273	0.374125	1.842989	1.108384e+04
Zimbabwe	1	1	0	12712.491900	6191.766957	0.014313	0.074876	2.713408	6.422392e+04

105 rows × 25 columns

In [12]:

```
# Dividing the data in three parts: A) No oil countries b) OECD c) population <1 mil
data_no = data.loc[data['n']==1,:] # Dummies of no oil will only be considered
data_oecd = data_no [data_no.o ==1] # considering the dummies for oecd only
data_pop = data_no [data_no.i==1] # population is less than 1 million
```

In [13]:

```
# TABLE 1: Estimation of the textbook Solow model

# Formulating the models: A) Unrestricted Model and B) Restricted Model

# UNRESTRICTED MODEL EACH FOR THREE CATEGORIES

reg1 = sm.OLS(endog = data_no['ln_19'],
              exog = data_no[['constant', 'ln(inv)', 'ln(n+g+\u03B4)']],
             missing = 'drop').fit()

reg2 = sm.OLS(endog = data_oecd['ln_19'],
              exog = data_oecd[['constant', 'ln(inv)', 'ln(n+g+\u03B4)']],
             missing = 'drop').fit()

reg3 = sm.OLS(endog = data_pop['ln_19'],
              exog = data_pop[['constant', 'ln(inv)', 'ln(n+g+\u03B4)']],
             missing = 'drop').fit()

#RESTRICTED MODEL: RESTRICTION ON COEFFICEINTS

regr1 = sm.OLS(endog = data_no['ln_19'],
```

```
                 exog = data_no[['constant', 'ln(inv)-ln(n+g+\u03B4)']],
                missing = 'drop').fit()

regr2 = sm.OLS(endog = data_oecd['ln_19'],
                 exog = data_oecd[['constant', 'ln(inv)-ln(n+g+\u03B4)']],
                missing = 'drop').fit()

regr3 = sm.OLS(endog = data_pop['ln_19'],
                 exog = data_pop[['constant', 'ln(inv)-ln(n+g+\u03B4)']],
                missing = 'drop').fit()
```

In [14]:

```
# constructing the table

info_dict = {'R^2': lambda x:x.rsquared_adj,
            'Observations': lambda x:x.nobs,
            's.e.e.': lambda x:np.sqrt(x.scale),
            'Implied \u03B1': lambda x:f"{x.params[1]/(1+x.params[1]):.2f}"}

results_unrestricted = summary_col(results=[reg1, reg3, reg2],
                                  float_format= '%0.2f',
                                  stars= True,
                                  model_names= ['Non-oil', 'Intermediate', 'OECD'],
                                  info_dict = info_dict,
                                  regressor_order = ['constant', 'ln(inv)', 'ln(n+g+

results_restricted = summary_col(results=[regr1, regr3, regr2],
                                  float_format= '%.2f',
                                  stars= True,
                                  model_names= ['Non-oil', 'Intermediate', 'OECD'],
                                  info_dict = info_dict,
                                  regressor_order = ['constant', 'ln(inv)-ln(n+g+\u0

# Providing titles
results_restricted.add_title('Restricted Regressions')
results_unrestricted.add_title('Unrestricted Regressions')
print('')
print(results_unrestricted)
print('')
print(results_restricted)
```

```
           Unrestricted Regressions
===============================================
                Non-oil   Intermediate   OECD
-----------------------------------------------
constant        -923.34*** -764.27***   -82.73
                (159.96)   (193.47)     (119.28)
ln(inv)         1.09***    1.26***      1.02***
                (0.28)     (0.30)       (0.19)
ln(n+g+δ)       -314.14*** -261.37***   -33.48
                (53.35)    (64.58)      (39.94)
R-squared       0.42       0.36         0.58
R-squared Adj.  0.40       0.34         0.54
R^2             0.4027     0.3377       0.5417
Observations    84.0000    70.0000      25.0000
s.e.e.          0.9612     0.9405       0.2074
Implied α       0.52       0.56         0.50
===============================================
Standard errors in parentheses.
* p<.1, ** p<.05, ***p<.01

             Restricted Regressions
================================================
```

```
                  Non-oil  Intermediate   OECD
------------------------------------------------
constant          14.48*** 14.70***     14.35***
                  (1.06)   (1.05)       (0.53)
ln(inv)-ln(n+g+δ) 1.36***  1.36***      0.97***
                  (0.33)   (0.33)       (0.18)
R-squared         0.17     0.20         0.57
R-squared Adj.    0.16     0.19         0.55
R^2               0.1597   0.1895       0.5484
Observations      84.0000  70.0000      25.0000
s.e.e.            1.1401   1.0404       0.2059
Implied α         0.58     0.58         0.49
================================================
Standard errors in parentheses.
* p<.1, ** p<.05, ***p<.01
```

In [16]:

```python
# TABLE 2: Estimation of the augmented Solow Model using human capital

#UNRESTRICTED
rega1= sm.OLS(endog = data_no['ln_19'],
             exog = data_no[['constant','ln(inv)','ln(n+g+\u03B4)','ln(sch)']],
             missing='drop').fit()

rega2= sm.OLS(endog = data_oecd['ln_19'],
             exog = data_oecd[['constant','ln(inv)','ln(n+g+\u03B4)', 'ln(sch)']],
            missing ='drop').fit()

rega3 = sm.OLS(endog = data_pop['ln_19'],
              exog = data_pop[['constant', 'ln(inv)', 'ln(n+g+\u03B4)', 'ln(sch)']],
            missing='drop').fit()

#RESTRICTED
reghr1 =sm.OLS(endog=data_no['ln_19'],
              exog = data_no[['constant','ln(inv)-ln(n+g+\u03B4)', 'ln(sch)-ln(n+g+\
              missing='drop').fit()

reghr2 = sm.OLS(endog = data_oecd['ln_19'],
               exog= data_oecd[['constant', 'ln(inv)-ln(n+g+\u03B4)', 'ln(sch)-ln(n+
               missing='drop').fit()

reghr3 = sm.OLS (endog= data_pop['ln_19'],
                exog = data_pop[['constant', 'ln(inv)-ln(n+g+\u03B4)','ln(sch)-ln(n+
                missing='drop').fit()


# Drawing the regression table

info_dictu = {'R^2': lambda x:x.rsquared_adj,
        'Observations': lambda x:x.nobs,
        's.e.e.': lambda x: np.sqrt(x.scale),
        'Implied \u03B1': lambda x:f" {x.params[1]/(1+x.params[1]+x.params[3]):.2f}"
        'Implied \u03B2': lambda x:f" {x.params[3]/(1+x.params[1] + x.params[3]):.2f

info_dictr = {'R^2': lambda x: x.rsquared_adj,
            'Observations': lambda x: x.nobs,
            's.e.e.': lambda x: np.sqrt(x.scale),
            'Implied α': lambda x: f"{x.params[1]/(1 + x.params[1] + x.params[2]):.2
            'Implied β': lambda x: f"{x.params[2]/(1 + x.params[1] + x.params[2]):.2

results_unrestricted = summary_col(results = [rega1, rega3, rega2],
                                  float_format = '%0.2f',
                                  stars= True,
                                  model_names = ['Non-Oil', 'Intermediate','OECD'],
```

```
                                   info_dict= info_dictu,
                                    regressor_order = ['constant','ln(inv)',  'ln(n+g+

results_restricted = summary_col(results = [reghr1, reghr3, reghr2],
                         float_format='%.2f',
                         stars = True,
                         model_names =['Non-Oil','Intermediate','OECD'],
                         info_dict= info_dictr,
                         regressor_order = ['constant', 'ln(inv)-ln(n+g+\u03B4)','ln


results_restricted.add_title('Restricted Regressions')
results_unrestricted.add_title('Unrestricted Regressions')
print(results_unrestricted)
print('')
print(results_restricted)
```

```
          Unrestricted Regressions
=============================================
                Non-Oil  Intermediate   OECD
---------------------------------------------
constant        155.30   168.23        -119.24
                (166.89) (182.65)      (102.53)
ln(inv)         0.83***  0.93***       1.01***
                (0.20)   (0.22)        (0.16)
ln(n+g+δ)       42.64    46.88         -46.72
                (55.42)  (60.72)       (34.37)
ln(sch)         3.37***  3.26***       0.94***
                (0.38)   (0.41)        (0.31)
R-squared       0.70     0.67          0.71
R-squared Adj. 0.69     0.66          0.67
R^2             0.6907   0.6551        0.6661
Observations    83.0000  70.0000       25.0000
s.e.e.          0.6870   0.6787        0.1771
Implied α       0.16     0.18          0.34
Implied β       0.65     0.63          0.32
=============================================
Standard errors in parentheses.
* p<.1, ** p<.05, ***p<.01

             Restricted Regressions
================================================
                   Non-Oil  Intermediate   OECD
------------------------------------------------
constant           14.95*** 15.25***      14.65***
                   (0.64)   (0.68)        (0.47)
ln(inv)-ln(n+g+δ) 0.83***  0.94***       0.95***
                   (0.20)   (0.22)        (0.16)
ln(sch)-ln(n+g+δ) 3.14***  3.04***       0.89***
                   (0.26)   (0.31)        (0.31)
R-squared          0.70     0.67          0.68
R-squared Adj.     0.69     0.66          0.66
R^2                0.6918   0.6567        0.6553
Observations       83.0000  70.0000       25.0000
s.e.e.             0.6858   0.6772        0.1799
Implied α          0.17     0.19          0.33
Implied β          0.63     0.61          0.31
================================================
Standard errors in parentheses.
* p<.1, ** p<.05, ***p<.01
```

In [17]:

```
# Now the Solow rate of convergence
```

```python
# UNCONDITIONAL
regcon1 = sm.OLS(endog= data_no['ln_19']-data_no['ln_95'],
                 exog= data_no[['constant', 'ln_95']],
                 missing='drop').fit()

regcon2 = sm.OLS( endog= data_oecd['ln_19']-data_oecd['ln_95'],
                 exog= data_oecd[['constant', 'ln_95']],
                 missing='drop').fit()

regcon3 = sm.OLS( endog= data_pop['ln_19']-data_pop['ln_95'],
                 exog= data_pop[['constant', 'ln_95']],
                 missing='drop').fit()

#CONDITIONAL(without the human capital component)
regcon4 = sm.OLS(endog= data_no['ln_19']-data_no['ln_95'],
                 exog= data_no[['constant','ln_95','ln(inv)','ln(n+g+\u03B4)']],
                 missing='drop').fit()

regcon5 = sm.OLS(endog=data_oecd['ln_19'] - data_oecd['ln_95'],
                 exog= data_oecd[['constant','ln_95','ln(inv)','ln(n+g+\u03B4)']],
                 missing='drop').fit()

regcon6 = sm.OLS(endog= data_pop['ln_19']-data_pop['ln_95'],
                 exog= data_pop[['constant','ln_95','ln(inv)','ln(n+g+\u03B4)']],\
                 missing='drop').fit()

#CONDITIONAL(with the human capital component)
regcon7 = sm.OLS(endog= data_no['ln_19']-data_no['ln_95'],
                 exog= data_no[['constant','ln_95','ln(inv)','ln(n+g+\u03B4)','ln(sch
                 missing='drop').fit()

regcon8 = sm.OLS(endog=data_oecd['ln_19'] - data_oecd['ln_95'],
                 exog= data_oecd[['constant','ln_95','ln(inv)','ln(n+g+\u03B4)','ln(s
                 missing='drop').fit()

regcon9 = sm.OLS(endog= data_pop['ln_19']-data_pop['ln_95'],
                 exog= data_pop[['constant','ln_95','ln(inv)','ln(n+g+\u03B4)','ln(sc
                 missing='drop').fit()


# constructing the regression tables
info_dictrcon = {'R^2' :lambda x: x.rsquared_adj,
                 'N': lambda x: x.nobs,
                 's.e.e.': lambda x:np.sqrt(x.scale),
                 'Implied \u03BB': lambda x: f"{-np.log(x.params[1]+1)/25:.5f}"}

info_dicturcon ={'R^2': lambda x: x.rsquared_adj,
                 'N': lambda x: x.nobs,
                 's.e.e.': lambda x: x.np.sqrt(x.scale),
                 'Implied \u03BB': lambda x: f"{-np.log(x.params[1]+1)/25:.5f}"}

info_dicturconh ={'R^2': lambda x: x.rsquared_adj,
                 'N': lambda x: x.nobs,
                 's.e.e.': lambda x: x.np.sqrt(x.scale),
                 'Implied \u03BB': lambda x: f"{-np.log(x.params[1]+1)/25:.5f}"}
# TABLE 3

table_rcon = summary_col(results =[regcon1, regcon3, regcon2],
                         float_format= '%.2f',
                         stars= True,
                         model_names = ['Non-Oil','Intermediate','OECD'],
                         info_dict = info_dictrcon,
                         regressor_order =['constant','ln_60'])
# TABLE 4
```

```
table_urcon = summary_col(results=[regcon4, regcon6, regcon5],
                         float_format='%.2f',
                         stars= True,
                         info_dict=info_dicturcon,
                         model_names=['Non-Oil','Intermediate','OECD'],
                         regressor_order=['constant','ln_60','ln(inv)','ln(n+g+\u03B
# TABLE 5

table_urconh = summary_col(results=[regcon7, regcon9, regcon8],
                         float_format='%.2f',
                         stars= True,
                         info_dict=info_dicturconh,
                         model_names=['Non-Oil','Intermediate','OECD'],
                         regressor_order=['constant','ln_60','ln(inv)','ln(n+g+\u03B

table_rcon.add_title('Test for unconditional convergence')
table_urcon.add_title ('Test for conditional convergence (Without Human Capital)')
table_urconh.add_title ('Test for conditional convergence (With Human capital)')

print(table_rcon)
print('')
print(table_urcon)
print('')
print(table_urconh)
```

```
    Test for unconditional convergence
===========================================
               Non-Oil Intermediate   OECD
-------------------------------------------
constant       1.60*** 1.80***      1.87
               (0.52)  (0.67)       (1.83)
ln_95          -0.11** -0.13*       -0.13
               (0.05)  (0.07)       (0.17)
R-squared      0.05    0.05         0.03
R-squared Adj. 0.04    0.04         -0.02
R^2            0.0428  0.0411       -0.0158
N              84.0000 70.0000      25.0000
s.e.e.         0.5956  0.6269       0.2111
Implied λ      0.00488 0.00573      0.00562
===========================================
Standard errors in parentheses.
* p<.1, ** p<.05, ***p<.01

Test for conditional convergence (Without Human Capital)
=============================================
               Non-Oil  Intermediate   OECD
---------------------------------------------
constant       127.09   128.65       30.71
               (123.57) (144.19)     (72.65)
ln(inv)        0.61***  0.71***      0.74***
               (0.17)   (0.19)       (0.12)
ln(n+g+δ)      40.65    40.82        7.43
               (41.45)  (48.35)      (24.44)
ln_95          -0.12*   -0.16**      -0.33***
               (0.07)   (0.08)       (0.10)
R-squared      0.20     0.24         0.70
R-squared Adj. 0.17     0.20         0.66
R^2            0.1702   0.2011       0.6574
N              84.0000  70.0000      25.0000
s.e.e.
Implied λ      0.00512  0.00700      0.01575
=============================================
Standard errors in parentheses.
```

```
* p<.1, ** p<.05, ***p<.01

Test for conditional convergence (With Human capital)
============================================
                Non-Oil  Intermediate   OECD
--------------------------------------------
constant        257.34*  236.11        69.29
                (132.44) (151.20)      (80.37)
ln(inv)         0.62***  0.72***       0.69***
                (0.16)   (0.19)        (0.13)
ln(n+g+δ)       82.28*   74.87         21.37
                (44.08)  (50.35)       (27.43)
ln(sch)         1.05**   1.03*         -0.37
                (0.45)   (0.52)        (0.34)
ln_95           -0.30*** -0.34***      -0.18
                (0.10)   (0.12)        (0.17)
R-squared       0.25     0.28          0.72
R-squared Adj. 0.22     0.23          0.66
R^2             0.2167   0.2349        0.6608
N               83.0000  70.0000       25.0000
s.e.e.
Implied λ       0.01405  0.01642       0.00808
============================================
Standard errors in parentheses.
* p<.1, ** p<.05, ***p<.01
```

In [18]:

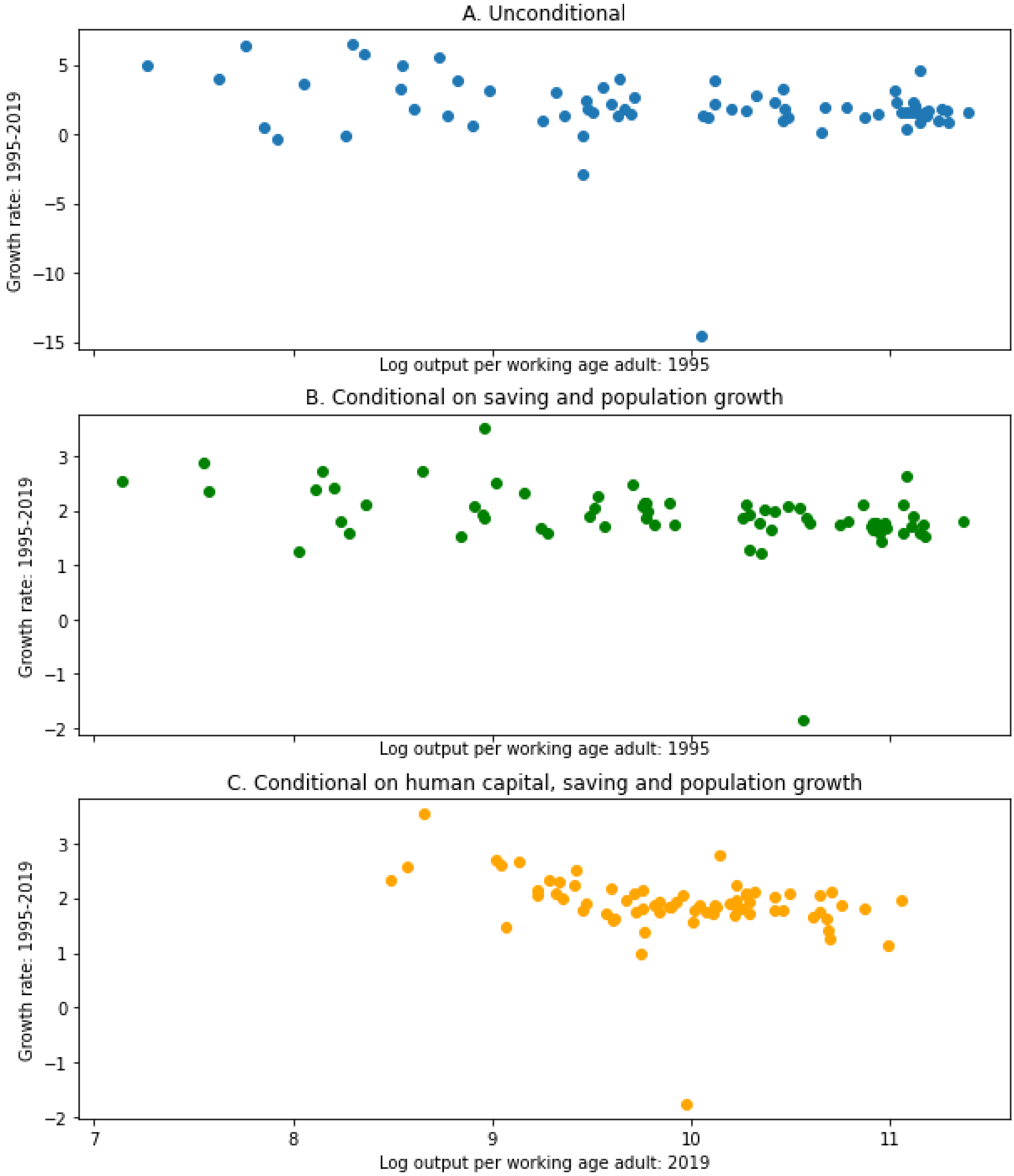
```python
# Rate of convergence RESTRICTED ESTIMATION
r1 = sm.OLS(endog=data_no['ln_19']-data_no['ln_95'],
            exog= data_no[['constant','ln_95','ln(inv)-ln(n+g+\u03B4)', 'ln(sch)-ln(n
            missing='drop').fit()
r2 = sm.OLS(endog=data_pop['ln_19']-data_pop['ln_95'],
            exog = data_pop[['constant', 'ln(inv)-ln(n+g+\u03B4)','ln(sch)-ln(n+g+\u0
            missing='drop').fit()
r3 = sm.OLS(endog=data_oecd['ln_19']-data_oecd['ln_95'],
            exog= data_oecd[['constant','ln_95','ln(inv)-ln(n+g+\u03B4)','ln(sch)-ln(
            missing='drop').fit()

#Constructing the table

info_dictrestricted = {'R^2': lambda x: x.rsquared_adj,
                       'Observations': lambda x:x.nobs,
                       's.e.e.': lambda x: np.sqrt(x.scale),
                       'Implied \u03BB': lambda x: f"{-np.log(x.params[1]+1)/25:.5f}"
results=summary_col(results=[r1, r2, r3],
                    float_format='%0.3f',
                    stars=True,
                    info_dict=info_dictrestricted,
                    model_names= ['Non-Oil','Intermediate','OECD'],
                    regressor_order= ['constant','ln_60','ln(inv)-ln(n+g+\u03B4)','ln

results.add_title('Rate of conditional convergence (Restricted Regression)')
print(results)
```

```
Rate of conditional convergence (Restricted Regression)
=================================================
                   Non-Oil  Intermediate   OECD
-------------------------------------------------
constant           6.077*** 6.784***      5.254***
                   (1.420)  (1.646)       (1.814)
ln(inv)-ln(n+g+δ) 0.630*** 0.744***      0.737***
                   (0.167)  (0.186)       (0.112)
ln(sch)-ln(n+g+δ) 0.718*   0.753         -0.252
                   (0.418)  (0.493)       (0.301)
```

```
ln_95               -0.319*** -0.353***    -0.249*
                    (0.101)   (0.118)      (0.143)
R-squared           0.221     0.254        0.708
R-squared Adj.      0.191     0.220        0.667
R^2                 0.1910    0.2198       0.6667
Observations        83.0000   70.0000      25.0000
s.e.e.              0.5508    0.5655       0.1210
Implied λ           0.01537   -0.02224     0.01143
=================================================
Standard errors in parentheses.
* p<.1, ** p<.05, ***p<.01
```

In [19]:

```
# Replicating the residual plots

regplt1= sm.OLS(endog= data_pop['ln_95'],
               exog=data_pop[['ln(inv)','ln(n+g+\u03B4)']],
               missing='drop').fit()

resid1= regplt1.resid

regplt2 = sm.OLS(endog=data_pop['ln_19']-data_pop['ln_95'],
                exog = data_pop[['ln(inv)','ln(n+g+\u03B4)']],
                missing='drop').fit()

resid2 = regplt2.resid

regplt3 = sm.OLS(endog=data_pop['ln_95'],
                exog=data_pop[['ln(inv)','ln(n+g+\u03B4)','ln(sch)']],
                missing='drop').fit()

resid3 = regplt3.resid

regplt4 = sm.OLS (endog=data_pop['ln_19']-data_pop['ln_95'],
                 exog =data_pop [['ln(inv)','ln(sch)','ln(n+g+\u03B4)']],
                 missing='drop').fit()

resid4 =regplt4.resid

# Scatter Plots

fig, ax =plt.subplots(3,1, sharex='col', figsize=(10,12))
ax[0].scatter(data_pop['ln_95'],(data_pop['ln_19']- data_pop['ln_95'])*100/25)
ax[0].set_xlabel('Log output per working age adult: 1995')
ax[0].set_ylabel('Growth rate: 1995-2019')
ax[0].set_title('A. Unconditional')

ax[1].scatter(resid1 + np.mean(data_pop['ln_95']),
             resid2 + np.mean(data_pop['ln_19']-data_pop['ln_95'])*100/25, color = "
ax[1].set_xlabel('Log output per working age adult: 1995')
ax[1].set_ylabel('Growth rate: 1995-2019')
ax[1].set_title('B. Conditional on saving and population growth')

ax[2].scatter(resid3 +np.mean(data_pop['ln_95']),
             resid4 + np.mean(data_pop['ln_19']-data['ln_95'])*100/25, color="orange
ax[2].set_xlabel('Log output per working age adult: 2019')
ax[2].set_ylabel('Growth rate: 1995-2019')
ax[2].set_title('C. Conditional on human capital, saving and population growth')
plt.show()
```


A. Unconditional
Growth rate: 1995-2019
Log output per working age adult: 1995
B. Conditional on saving and population growth
Growth rate: 1995-2019
Log output per working age adult: 1995
C. Conditional on human capital, saving and population growth
Growth rate: 1995-2019
Log output per working age adult: 2019