Project Appendix

May 12, 2024

0.1 Program for Model Calibrations

[ ]:
```
from scipy.optimize import fsolve

def equations(variables):
    x, t, w = variables
    eq1 = (1 - w) / 0.25 - x
    eq2 = 0.14 / (0.7 * (t ** -0.5)) - x
    eq3 = w - (0.35 + 0.5 * (0.14 * t + 1))
    return [eq1, eq2, eq3]

initial_guess = (0,1,0)

# Solve the system of equations
x, t, w = fsolve(equations, initial_guess)

print("Solution for the values in Jordan without an increase in human capital:")
print("The value of J_pi is =", x)
print("The value of theta=", t)
print("The value of the wage =", w)
```

```
Solution for the values in Jordan without an increase in human capital:
The value of J_pi is = 0.22992890156768905
The value of theta= 1.3216824944011112
The value of the wage = 0.9425177746080777
```

[ ]:
```
def equations(variables):
    x, t, w = variables
    eq1 = (1.2 - w) / 0.25 - x
    eq2 = 0.14 / (0.7 * (t ** -0.5)) - x
    eq3 = w - (0.35 + 0.5 * (0.14 * t + 1.2))
    return [eq1, eq2, eq3]


# Solve the system of equations
x, t, w = fsolve(equations, initial_guess)

print("Solution for the values in Jordan with an increase in human capital:")
```

```python
print("The value of J_pi is =", x)
print("The value of theta=", t)
print("The value of the wage =", w)
```

```
Solution for the values in Jordan with an increase in human capital:
The value of J_pi is = 0.31322605765246453
The value of theta= 2.4527640798126242
The value of the wage = 1.1216934855868839
```

[ ]:
```python
def equations(variables):
    x, t, w = variables
    eq1 = (1.1 - w) / 0.25 - x
    eq2 = 0.14 / (0.7 * (t ** -0.5)) - x
    eq3 = w - (0.35 + 0.5 * (0.14 * t + 1.1))
    return [eq1, eq2, eq3]


# Solve the system of equations
x, t, w = fsolve(equations, initial_guess)

print("Solution for the values in Jordan with an increase in human capital:")
print("The value of J_pi is =", x)
print("The value of theta=", t)
print("The value of the wage =", w)
```

```
Solution for the values in Jordan with an increase in human capital:
The value of J_pi is = 0.2740967606413665
The value of theta= 1.8782258548522628
The value of the wage = 1.0314758098396584
```

[ ]:
```python
def equations(variables):
    x, t, w = variables
    eq1 = (1 - w) / 0.28 - x
    eq2 = 0.115 / (1.08 * (t ** -0.4)) - x
    eq3 = w - (0.25 + 0.5 * (0.115 * t + 1))
    return [eq1, eq2, eq3]


# Solve the system of equations
x, t, w = fsolve(equations, initial_guess)

print("Solution for the values in India without tax relief or an increase in␣
 ↪human capital:")
print("The value of J_pi is =", x)
print("The value of theta=", t)
print("The value of the wage =", w)
```

```
Solution for the values in India without tax relief or an increase in human
capital:
```

```
The value of J_pi is = 0.17560574535547524
The value of theta= 3.4927024573994254
The value of the wage = 0.9508303913004669
```

[ ]:
```python
def equations(variables):
    x, t, w = variables
    eq1 = (0.8 - w) / 0.28 - x
    eq2 = 0.115 / (1.08 * (t ** -0.4)) - x
    eq3 = w - (0.25 + 0.5 * (0.115 * t + 0.8))
    return [eq1, eq2, eq3]


# Solve the system of equations
x, t, w = fsolve(equations, initial_guess)

print("Solution for the values in India with tax relief but without increase in␣
↪human capital:")
print("The value of J_pi is =", x)
print("The value of theta=", t)
print("The value of the wage =", w)
```

```
Solution for the values in India with tax relief but without increase in human
capital:
The value of J_pi is = 0.13862048173659192
The value of theta= 1.9336741758913791
The value of the wage = 0.7611862651137543
```

[ ]:
```python
def equations(variables):
    x, t, w = variables
    eq1 = (1.02 - w) / 0.28 - x
    eq2 = 0.115 / (1.08 * (t ** -0.4)) - x
    eq3 = w - (0.25 + 0.5 * (0.115 * t + 1.02))
    return [eq1, eq2, eq3]


# Solve the system of equations
x, t, w = fsolve(equations, initial_guess)

print("Solution for the values in India with tax relief and an increase in␣
↪human capital:")
print("The value of J_pi is =", x)
print("The value of theta=", t)
print("The value of the wage =", w)
```

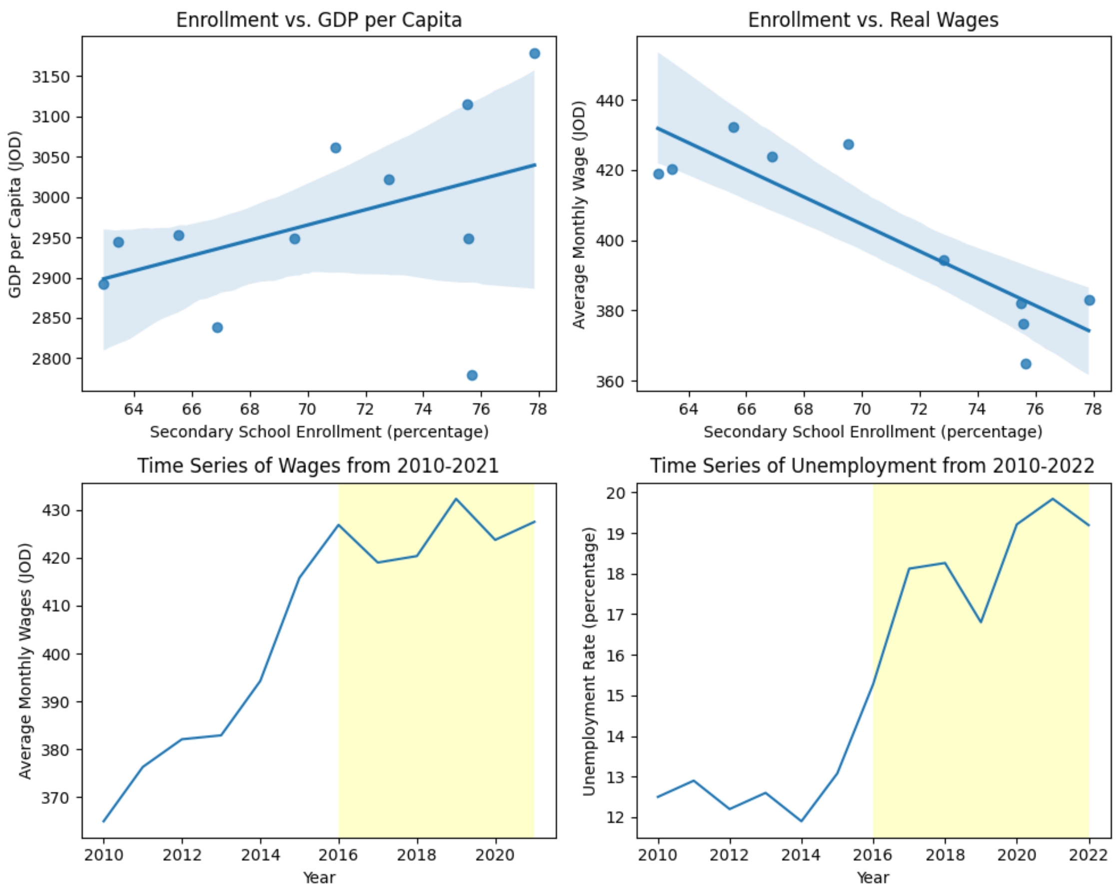
```
Solution for the values in India with tax relief and an increase in human
capital:
The value of J_pi is = 0.17875272354150024
The value of theta= 3.65129108536313
```

```
The value of the wage = 0.9699492374083799
```

0.2 Graphs for Jordan

```
[ ]: import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import linearmodels as lm
import statsmodels.api as sm
import statsmodels.formula.api as smf
from datetime import datetime
df = pd.read_excel('Desktop/Advanced_Macro_Data.xlsx')

fig, axes = plt.subplots(2, 2, figsize=(10, 8))

# Plot 1: Scatterplot of Secondary School Enrollment and GDP per Capita
sns.regplot(x='Enrollment', y='GDP per capita', data=df, ax=axes[0, 0])
axes[0, 0].set_title('Enrollment vs. GDP per Capita')
axes[0, 0].set_xlabel('Secondary School Enrollment (percentage)')
axes[0, 0].set_ylabel('GDP per Capita (JOD)')

df['Date'] = pd.to_datetime(df['Date'])
df = df.set_index('Date')

# Plot 2: Scatterplot of Secondary School Enrollment and Real Wages
sns.regplot(x='Enrollment', y='Real Wages', data=df, ax=axes[0, 1])
axes[0, 1].set_title('Enrollment vs. Real Wages')
axes[0, 1].set_xlabel('Secondary School Enrollment (percentage)')
axes[0, 1].set_ylabel('Average Monthly Wage (JOD)')

df_2021 = df
df_2021.drop(['2022-01-01'])

# Plot 3: Time Series of Wages from 2010-2021
axes[1, 0].plot(df.index, df['Real Wages'])
axes[1, 0].axvspan(datetime(2016, 1, 1), datetime(2021, 1, 1),␣
 ↪facecolor='yellow', alpha=0.2)
axes[1, 0].set_title('Time Series of Wages from 2010-2021')
axes[1, 0].set_xlabel('Year')
axes[1, 0].set_ylabel('Average Monthly Wages (JOD)')

# Plot 4: Time Series of Unemployment from 2010-2022
axes[1, 1].plot(df.index, df['Unemployment '])
axes[1, 1].axvspan(datetime(2016, 1, 1), datetime(2022, 1, 1),␣
 ↪facecolor='yellow', alpha=0.2)
axes[1, 1].set_title('Time Series of Unemployment from 2010-2022')
```

```
axes[1, 1].set_xlabel('Year')
axes[1, 1].set_ylabel('Unemployment Rate (percentage)')

plt.tight_layout()
plt.show()
```



0.3 Graphs for India

[ ]:
```
# Load the data
df = pd.read_excel('Desktop/India_Data.xlsx')

fig, axes = plt.subplots(2, 2, figsize=(8, 6))

# Plot GDP Per Capita vs. Secondary School Enrollment
sns.regplot(x='GDP Per Capita (Current US $)', y='Secondary School Enrollment␣
↪(Gross %)', data=df, ax=axes[0, 0])
axes[0, 0].set_title('GDP Per Capita vs. Secondary School Enrollment',␣
↪fontsize=10)
axes[0, 0].set_xlabel('GDP Per Capita (Current US $)', fontsize=8)
```

```python
axes[0, 0].set_ylabel('Secondary School Enrollment (Gross %)', fontsize=8)
axes[0, 0].tick_params(axis='both', labelsize=8)
axes[0, 0].grid(True)

# Plot Secondary School Enrollment vs. Average Monthly Wages
sns.regplot(x='Secondary School Enrollment (Gross %)', y='Average Monthly Wages (Indian Rupees)', data=df, ax=axes[0, 1])
axes[0, 1].set_title('Secondary School Enrollment vs. Average Monthly Wages', fontsize=10)
axes[0, 1].set_xlabel('Secondary School Enrollment (Gross %)', fontsize=8)
axes[0, 1].set_ylabel('Average Monthly Wages (Indian Rupees)', fontsize=8)
axes[0, 1].tick_params(axis='both', labelsize=8)
axes[0, 1].grid(True)

# Plot Secondary School Enrollment vs. Unemployment Rate
sns.regplot(x='Secondary School Enrollment (Gross %)', y='Unemployment Rate', data=df, ax=axes[1, 0])
axes[1, 0].set_title('Secondary School Enrollment vs. Unemployment Rate', fontsize=10)
axes[1, 0].set_xlabel('Secondary School Enrollment (Gross %)', fontsize=8)
axes[1, 0].set_ylabel('Unemployment Rate', fontsize=8)
axes[1, 0].tick_params(axis='both', labelsize=8)
axes[1, 0].grid(True)

# Plot Wages and Unemployment Rate Over Time
color = 'tab:red'
axes[1, 1].plot(df['Year'], df['Average Monthly Wages (Indian Rupees)'], color=color, label='Wages')
axes[1, 1].set_title('Wages and Unemployment Rate Before and After 2008', fontsize=10)
axes[1, 1].set_xlabel('Year', fontsize=8)
axes[1, 1].set_ylabel('Average Monthly Wages (Indian Rupees)', color=color, fontsize=8)
axes[1, 1].tick_params(axis='y', labelcolor=color, labelsize=8)
axes[1, 1].legend(loc='upper left', fontsize=8)

axes[1, 1].axvline(x=2008, color='gray', linestyle='--', label='Tax Relief Policy Amendment in 2008')

ax2 = axes[1, 1].twinx()
color = 'tab:blue'
ax2.plot(df['Year'], df['Unemployment Rate'], color=color, label='Unemployment Rate')
ax2.set_ylabel('Unemployment Rate', color=color, fontsize=8)
ax2.tick_params(axis='y', labelcolor=color, labelsize=8)
ax2.legend(loc='upper right', fontsize=8)
```

```
plt.suptitle('', fontsize=12)
plt.tight_layout(rect=[0, 0.03, 1, 0.95])  # Adjust layout to fit title
plt.show()
```