ECON272A - Replication Assignment

Christopher Saw

18 March 2022

1 Introduction

I perform a simulation study motivated by Doraszelski & Jaumandreu's 2018 JPE paper "Measuring the Bias of Technological Change" (henceforth D&J). The goal of this exercise is to implement the procedure in D&J to estimate labor-augmenting productivity and Hicks-neutral productivity from simulated panel data and recover structural parameters in the model. The model used for this study is a simplified model from the model in D&J, so as to reduce the parameter space and complexity in estimation. I will go through the model for this exercise and explain the estimation routine adapted from D&J. I then explain how I targeted moments from the original D&J dataset and the procedure for simulating the data. Lastly, I present my results. The code is written in Python and attached at the end (script file attached separately).

2 Model

All lowercase variables denote logs unless stated otherwise.

Production

I consider a single industry with N firms, and each firm is observed for T periods.[1] Assume that output Q of a firm i in period t follows CES technology:

$$Q_{it}(K_{it}, L_{it}, M_{it}) = X_{it}(K_{it}, L_{it}, M_{it})^{\frac{\nu\sigma}{\sigma-1}} \cdot \exp(\omega_{it}^H) \cdot \exp(\varepsilon_{it}) \tag{1}$$

where

$$X_{it}(K_{it}, L_{it}, M_{it}) = \beta_K K_{it}^{\frac{\sigma-1}{\sigma}} + [\exp(\omega_{it}^L) \cdot L_{it}]^{\frac{\sigma-1}{\sigma}} + \beta_M M_{it}^{\frac{\sigma-1}{\sigma}} \tag{2}$$

and (K, L, M)[2] denote capital, labor and materials input respectively, and (ω^L, ω^H) denote labor-augmenting and Hicks-neutral productivity respectively. ε is a random i.i.d. shock that is mean zero and independent across i and t. σ denotes the elasticity of substitution and ν denotes the elasticity of scale.

Assume that labor and materials are static inputs that are chosen each period to maximize short-run profits and that the firm is a price taker in input markets. Assume that firm capital follows the usual law of motion: $K_{it+1} = (1-\delta)K_{it} + I_{it}$. These assumptions on timing will be relevant for discussing instruments later.

[1] I assume a balanced panel for simplicity.

[2] I do not distinguish between different types of labor and different types of materials.

First-order conditions

Assume that each firm faces a demand curve $P_{it} = P(Q_{it}, D_{it})$ where D_{it} is an (observed) exogenous demand shifter. Capital is fixed at the start of each period. For the static inputs labor and materials, the firm maximizes per-period profits according to:

$$\max_{L_{it}, M_{it}} P(Q_{it}, D_{it}) \cdot Q_{it} - W_{it} L_{it} - P_{it}^M M_{it}$$

where W, P_{it}^M refer to wages and the price of materials respectively.

For Hicks-neutral productivity, consider the FOC with respect to materials:

$$\begin{aligned}
\frac{\partial P(Q_{it}, \cdot)}{\partial Q_{it}} \frac{\partial Q_{it}}{\partial M_{it}} Q_{it}(M_{it}, \cdot) + P(Q_{it}, \cdot) \frac{\partial Q_{it}}{\partial M_{it}} &= P_{it}^M \\
\left[\frac{\partial P(Q_{it}, \cdot)}{\partial Q_{it}} Q_{it} + P(Q_{it}, \cdot) \right] \frac{\partial Q_{it}}{\partial M_{it}} &= P_{it}^M \\
[1 + \eta_{it}] \frac{\partial Q_{it}}{\partial M_{it}} &= \frac{P_{it}^M}{P(Q_{it}, \cdot)}
\end{aligned}$$

Where η_{it} denotes the price elasticity of demand. Solving this equation gives:

$$\begin{aligned}
(1 + \eta_{it}) \frac{\nu\sigma}{\sigma - 1} X_{it}^{\frac{\nu\sigma}{\sigma-1} - 1} \cdot \beta_M \frac{\sigma - 1}{\sigma} M_{it}^{\frac{\sigma-1}{\sigma} - 1} \cdot \exp(\omega_{it}^H) \cdot \exp(\varepsilon_{it}) &= \frac{P_{it}^M}{P(Q_{it}, \cdot)} \\
(1 + \eta_{it}) \nu X_{it}^{\frac{1 + \nu\sigma - \sigma}{\sigma - 1}} \cdot \beta_M M_{it}^{-\frac{1}{\sigma}} \cdot \exp(\omega_{it}^H) \cdot \exp(\varepsilon_{it}) &= \frac{P_{it}^M}{P(Q_{it}, \cdot)}
\end{aligned}$$

Taking logs we get:

$$\omega_{it}^H = p_{it}^M - p_{it} - \frac{1 + \nu\sigma - \sigma}{\sigma - 1} x_{it} - \frac{1}{\sigma} m_{it} - \log(1 + \eta_{it}(p_{it}, D_{it})) - \log(\nu\beta_M) - \varepsilon_{it} \tag{3}$$

For labor-augmenting productivity, take the ratio of FOCs in L and M:

$$\begin{aligned}
\frac{\partial Q_{it} / \partial L_{it}}{\partial Q_{it} / \partial M_{it}} &= \frac{W_{it}}{P_{it}^M} \\
\frac{\exp(\omega_{it}^L)^{\frac{\sigma-1}{\sigma}} L_{it}^{-\frac{1}{\sigma}}}{\beta_M M_{it}^{-\frac{1}{\sigma}}} &= \frac{W_{it}}{P_{it}^M} \\
\exp(\omega_{it}^L)^{\frac{\sigma-1}{\sigma}} \left(\frac{M_{it}}{L_{it}} \right)^{\frac{1}{\sigma}} &= \left(\frac{W_{it}}{P_{it}^M} \right) \beta_M
\end{aligned}$$

And taking logs we get:

$$(1 - \sigma)\omega_{it}^L = m_{it} - l_{it} + \sigma(p_{it}^M - w_{it}) - \sigma \log \beta_M \tag{4}$$

Markov processes

Assume that ω^H and ω^L jointly follow first-order Markov processes:

$$\omega_{it}^H = \mathbb{E}\big[\omega_{it}^H | \omega_{it-1}^H, z_{it-1}\big] + \xi_{it}^H = g_{t-1}^H(\omega_{it-1}^H, z_{it-1}) + \xi_{it}^H \tag{5}$$

$$\omega_{it}^L = \mathbb{E}\big[\omega_{it}^L | \omega_{it-1}^L, z_{it-1}\big] + \xi_{it}^L = g_{t-1}^L(\omega_{it-1}^L, z_{it-1}) + \xi_{it}^L \tag{6}$$

where ξ_{it}^H and ξ_{it}^L are i.i.d. innovation shocks to productivity and z_{it} is an observable of the firm that affects both components of productivity (D&J use R&D expenditures).

3 Estimation

Step 1: Invert FOC for labor-augmenting productivity

Equation (4) gives a reduced-form equation that we can estimate with OLS:

$$m_{it} - l_{it} = \beta_0 + \beta_1(p_{it}^M - w_{it}) + e_{it} \tag{7}$$

where $\hat{\beta}_0 = \sigma \log \beta_M$, $\hat{\beta}_1 = -\sigma$ and $\hat{e}_{it} = (1-\sigma)\omega_{it}^L$, where $\hat{e}_{it}$ are the fitted residuals.

Step 2: Recover Hicks-neutral productivity

From (3) we can derive a control function: $\omega_{it}^H = h^H(p_{it}^M, p_{it}, k_{it}, m_{it}, D_{it})$. Using this with (1), (2), (3) and (5), we can form an equation to estimate:

$$q_{it} = \underbrace{\frac{\nu\sigma}{\sigma-1} x_{it}(k_{it}, m_{it}, l_{it}, \omega_{it}^L; \sigma, \theta) + g_{t-1}^H(h^H(p_{it-1}^M, p_{it-1}, k_{it-1}, m_{it-1}, D_{it-1}), z_{it-1})}_{\phi(\cdot)} + \tilde{\xi}_{it}^H$$

Where we let $\tilde{\xi}_{it}^H \equiv \xi_{it}^H + \varepsilon_{it}$. Pick any values for $\theta = (\nu, \beta_K, \beta_M)$. We can recover $\tilde{\omega}_{it}^H \equiv g_{t-1}^H(\cdot) + \tilde{\xi}_{it}^H$ from the above equation by (i) fitting a polynomial approximation $\phi(\cdot)$ over the observed variables and (ii) using σ and ω_{it}^L from Step 1 for x_{it}. Lastly, if we assume that $g_{t-1}^H(\cdot)$ is a linear function, we can use $(\tilde{\omega}_{it-1}^H, z_{it-1})$ to recover $\tilde{\xi}_{it}^H$ from (5). We can then construct moment conditions: $\mathbb{E}[Z\tilde{\xi}_{it}^H(\theta)] = 0$, where Z is a vector of instruments.

For instruments, I follow D&J's approach and exploit the timing assumption. $\tilde{\xi}_{it}^H$ is an innovation shock to $\tilde{\omega}_{it}^H$ in period t and unknown to the firm in period $t-1$. Past decisions of the firm (l_{it-1}, m_{it-1}) are therefore uncorrelated with $\tilde{\xi}_{it}^H$. Similarly, the firm's wages and prices of materials in the previous period (w_{it-1}, p_{it-1}^M) do not affect $\tilde{\xi}_{it}^H$. Lastly, the demand shifter D_{it} is exogenous by construction and can also be included as an instrument. Therefore, we have $Z = (l_{it-1}, m_{it-1}, w_{it-1}, p_{it-1}^M, D_{it})'$.

The GMM estimator is:

$$\min_\theta \left[\frac{1}{NT}\sum_{i=1}^N\sum_{t=1}^T Z\tilde{\xi}_{it}^H(\theta)\right]' \hat{\Omega} \left[\frac{1}{NT}\sum_{i=1}^N\sum_{t=1}^T Z\tilde{\xi}_{it}^H(\theta)\right] \tag{8}$$

where $\hat{\Omega} = [ZZ']^{-1}$ is the optimal weighting matrix.

4 Simulation

I simulate data for $N = 300$ firms from the same industry for $T = 10$ periods.

Capital, Labor, Materials

Table 1 provides key moments for the rates of growth of capital, labor and materials. However, there are no moments about their levels in the descriptive statistics. In order to proceed, I assume that at $t = 0$, firms draw their capital, labor and materials (in logs) from an initial state that follows a lognormal$(0, 1)$ distribution.[3]

$$\log(S) \sim N(0, 1)$$

$$k_{i0} = l_{i0} = m_{i0} = \log(S_i)$$

After the initial values have been drawn, I use the moments in Table 1 to construct a stochastic growth process and forward simulate for each $v \in \{k, l, m\}$:

$$v_{it} = v_{it-1} + g^v_{it} \quad \text{where} \quad g^v \sim N(\mu_v, \sigma_v)$$

I target moments for Industry 3 ("Chemical Products") using the descriptive statistics provided by D&J (Table 1):

v	μ_v	σ_v
k	0.062	0.182
l	0.015	0.170
m	0.044	0.274

Productivity

Recall from (5) and (6) that ω^H and ω^L jointly follow first-order Markov processes that both depend on some observable z. I specify the data generating process as:

$$\omega^H_{it} = \rho\omega^H_{it-1} + \theta z_{it-1}$$
$$\omega^L_{it} = \rho\omega^L_{it-1} + \theta z_{it-1}$$

I set $\rho = 0.8$ and $\theta = 0.2$. I assume that z follows a MA(1) process: $z_{it} = \bar{z} + \alpha_1\varepsilon^Z_{it-1} + \varepsilon^Z_{it}$ (where $\varepsilon^Z \sim N(0, 1)$ is white noise). I set $\bar{z} = z_{i0}$ and $\alpha_1 = \frac{1}{2}$. To forward simulate $(z_{it}, \omega^H_{it}, \omega^L_{it})$ for T periods, I assume that $z_{i0} = \frac{1}{2}x_{i0}$ and separately draw the initial productivities ω^H_{i0} and ω^L_{i0} from a Uniform(0,1) distribution.

[3] The idea being that, in a large enough industry, the distribution of firm-size has a heavy right tail. S is some sufficient statistic in the initial period to scale capital, labor and materials accordingly. A 1:1 ratio is assumed for simplicity.

Output

Using (1) and (2), and the structural parameters estimated by D&J, I can now simulate output for each i in each t, assuming that $\varepsilon \sim N(0,1)$ in (1) is also a standard white noise. For Industry 3, the structural parameters are: $\sigma = 0.695$ (Table 4 in D&J, Column 10), $\nu = 0.933$ and $\beta_k = 0.137$ (Table 7 in D&J, Columns 2 and 1 respectively). I assume that $\beta_m = 1 - \beta_k$ (same assumption as D&J).

Prices

Recall that I assume each firm faces a demand curve $P_{it} = P(Q_{it}, D_{it})$. I now specify:

$$P_{it} = \exp(D_{it}) \cdot (Q_{it})^{-\eta}$$
$$p_{it} = D_{it} - \eta q_{it}$$

I use $\eta = 2.431$ for Industry 3 as estimated by D&J (Table 7, Column 5) and assume that D follows a MA(1) process: $D_{it} = \bar{D} + \alpha_1 \varepsilon^D_{it-1} + \varepsilon^D_{it}$ ($\varepsilon^D \sim N(0,1)$ is white noise). I set $\bar{D} = D_{i0}$ and draw D_{i0} from a Uniform(0,1) distribution. These assumptions allow me to forward simulate p_{it}.

For the input prices W_{it} and P^M_{it}, the FOCs from the model imply:

$$W_{it} = [1+\eta] \cdot \nu X_{it}^{\frac{1+\nu\sigma-\sigma}{\sigma-1}} \cdot \exp(\omega^L_{it}) L_{it}^{-\frac{1}{\sigma}} \cdot \exp(\omega^H_{it}) \cdot \exp(\varepsilon_{it}) \cdot P_{it}$$
$$P^M_{it} = [1+\eta] \cdot \nu X_{it}^{\frac{1+\nu\sigma-\sigma}{\sigma-1}} \cdot \beta_M M_{it}^{-\frac{1}{\sigma}} \cdot \exp(\omega^H_{it}) \cdot \exp(\varepsilon_{it}) \cdot P_{it}$$

Taking logs,

$$w_{it} = \log[(1+\eta)\nu] + \frac{1+\nu\sigma-\sigma}{\sigma-1} x_{it} + \omega^L_{it} - \frac{1}{\sigma} l_{it} + p_{it} + \omega^H_{it} + \varepsilon_{it}$$
$$p^M_{it} = \log[(1+\eta)\nu\beta_m] + \frac{1+\nu\sigma-\sigma}{\sigma-1} x_{it} - \frac{1}{\sigma} m_{it} + p_{it} + \omega^H_{it} + \varepsilon_{it}$$

and I can simulate (w_{it}, p^M_{it}).

5 Results

The attached Python code implements the 2-step GMM estimator from D&J under the simplified version of the model that I described above. I obtain estimates for $(\hat{\sigma}, \hat{\nu}, \hat{\beta}_K, \hat{\beta}_M, \{\hat{\omega}^H_{it}\}, \{\hat{\omega}^L_{it}\})$. However, I am unable to reproduce the simulated structural parameters from estimation. After reviewing my work, the likely explanation for these results is that the process for generating the simulated data in Section 4 was not a realistic approximation to the actual data generating process in D&J's original dataset. In particular, I constructed a stochastic and independent growth processes for capital, labor and materials in order to match the growth moments in Table 1 of D&J, but this is likely unrealistic as the growth rates of capital, labor and materials are likely to be correlated under a real data generating process.

I found no issues with my replication code and it should work if applied to actual data. For completeness I present a comparison of the simulated and estimated parameters below, and OLS regressions of $\{\hat{\omega}_{it}^H\}$ on $\{\omega_{it}^H\}$ and $\{\hat{\omega}_{it}^L\}$ on $\{\omega_{it}^L\}$.

	Simulated	Estimated
σ	0.695	0.224
ν	0.933	-6.638
β_K	0.137	1.530
β_M	0.863	2.479

Table 1: Structural Parameters

	ω_{it}^H	ω_{it}^L
$\hat{\omega}_{it}^H$	0.0397***	-
	(0.0033)	-
$\hat{\omega}_{it}^L$	-	-1.0993***
	-	(0.0318)
Obs	2700	2700

Standard errors in parentheses
** p< 0.01, * p< 0.05, + p< 0.10

Table 2: OLS regression for $\{\omega_{it}^H\}$ and $\{\omega_{it}^L\}$

Python code for simulation

dj_simulation.py

```
import os
os.chdir("/Users/christophersaw/Desktop/Replication")
import pandas as pd
import numpy as np

# Setup
np.random.seed(123456789)
N=300
T=10
k=np.zeros((N,T+1)) # log capital
l=np.zeros((N,T+1)) # log labor
m=np.zeros((N,T+1)) # log materials
k1=np.zeros((N,T+1)) # 1 denotes lag
l1=np.zeros((N,T+1))
m1=np.zeros((N,T+1))

# Parameters
nu=0.933
sigma=0.695
```

```
eta=2.431
beta_k=0.137
beta_m=1-beta_k

# Capital, Labor, Materials
# Assume that the initial distribution of firms is lognormal(0,1) and k0=m0=l0
# From the initial distribution, growth of (k, l, m) is stochastic
# Follows Industry 3 in D&J (Table 1)
S=np.random.normal(0,1,(N,1))
for i in range(N):
        k[i,0]=S[i]
        l[i,0]=S[i]
        m[i,0]=S[i]
        for t in range(T):
                k[i,t+1]=k[i,t]+np.random.normal(0.062,0.182)
                l[i,t+1]=l[i,t]+np.random.normal(0.015,0.170)
                m[i,t+1]=m[i,t]+np.random.normal(0.044,0.274)
                k1[i,t+1]=k[i,t]
                l1[i,t+1]=l[i,t]
                m1[i,t+1]=m[i,t]

# Productivity
z=np.zeros((N,T+1)) # log Z (such as R&D expenditures)
WL=np.zeros((N,T+1)) # labor-augmenting productivity
WH=np.zeros((N,T+1)) # Hicks-neutral productivity
z1=np.zeros((N,T+1))

# Assume that z0 is proportional to the initial state
# Assume that z follows a MA(1) process
# Assume that WL, WH are randomly drawn in the initial state
e0=np.random.normal(0,1,(N,T+1))
e1=np.random.normal(0,1,(N,T+1))
xi1=np.random.normal(0,1,(N,T+1))
xi2=np.random.normal(0,1,(N,T+1))
alpha=0.5
rho=0.8
theta=0.2
for i in range(N):
        z[i,0]=0.5*S[i]
        WL[i,0]=np.random.uniform(0,1)
        WH[i,0]=np.random.uniform(0,1)
        for t in range(T):
                z[i,t+1]=z[i,0]+alpha*e1[i,t]+e1[i,t+1]
                WL[i,t+1]=rho*WL[i,t]+theta*z[i,t]+xi1[i,t+1]
                WH[i,t+1]=rho*WH[i,t]+theta*z[i,t]+xi2[i,t+1]
                z1[i,t+1]=z[i,t]

# Output (follows CES production function)
```

```
Q=np.zeros((N,T+1)) # Q_it (levels)
X=np.zeros((N,T+1)) # X_it
p=(sigma-1)/sigma
beta_l=np.exp(WL)
for t in range(T+1):
        for i in range(N):
                X[i,t]=beta_k*np.power(np.exp(k[i,t]),p)+\
                np.power(beta_l[i,t]*np.exp(l[i,t]),p)+\
                beta_m*np.power(np.exp(m[i,t]),p)
                Q[i,t]=np.power(X[i,t],nu/p)*np.exp(WH[i,t])*np.exp(e0[i,t])

q=np.log(Q)
x=np.log(X)

# Goods price
D=np.zeros((N,T+1))         # D_it
ed=np.random.normal(0,1,(N,T+1))
for i in range(N):
        D[i,0]=np.random.uniform(0,1)
        for t in range(T):
                D[i,t+1]=D[i,0]+alpha*ed[i,t]+ed[i,t+1]

p=D-eta*q

# Wages and Price of materials
# Use FOCs in logs
C1=np.log((1+eta)*nu)
C2=np.log((1+eta)*nu*beta_m)
C3=(1+nu*sigma-sigma)/(sigma-1)
C4=-(1/sigma)

w=C1+C3*x+WL+C4*l+p+WH+e0
pm=C2+C3*x+C4*m+p+WH+e0

# Lags for output, demand shifter and prices
q1=np.zeros((N,T+1))
p1=np.zeros((N,T+1))
D1=np.zeros((N,T+1))
w1=np.zeros((N,T+1))
pm1=np.zeros((N,T+1))
for i in range(N):
        for t in range(T):
                q1[i,t+1]=q[i,t]
                p1[i,t+1]=p[i,t]
                D1[i,t+1]=D[i,t]
                w1[i,t+1]=w[i,t]
                pm1[i,t+1]=pm[i,t]
```

```
# Reshape
q=pd.DataFrame(q.flatten('F'), columns=["q"])
q1=pd.DataFrame(q1.flatten('F'), columns=["q1"])
p=pd.DataFrame(p.flatten('F'), columns=["p"])
p1=pd.DataFrame(p1.flatten('F'), columns=["p1"])
D=pd.DataFrame(D.flatten('F'), columns=["D"])
D1=pd.DataFrame(D1.flatten('F'), columns=["D1"])
k=pd.DataFrame(k.flatten('F'), columns=["k"])
k1=pd.DataFrame(k1.flatten('F'), columns=["k1"])
l=pd.DataFrame(l.flatten('F'), columns=["l"])
l1=pd.DataFrame(l1.flatten('F'), columns=["l1"])
m=pd.DataFrame(m.flatten('F'), columns=["m"])
m1=pd.DataFrame(m1.flatten('F'), columns=["m1"])
z=pd.DataFrame(z.flatten('F'), columns=["z"])
z1=pd.DataFrame(z1.flatten('F'), columns=["z1"])
w=pd.DataFrame(w.flatten('F'), columns=["w"])
w1=pd.DataFrame(w1.flatten('F'), columns=["w1"])
pm=pd.DataFrame(pm.flatten('F'), columns=["pm"])
pm1=pd.DataFrame(pm1.flatten('F'), columns=["pm1"])
WL=pd.DataFrame(WL.flatten('F'), columns=["WL"])
WH=pd.DataFrame(WH.flatten('F'), columns=["WH"])

# Firm and time ids
firm=pd.DataFrame(np.linspace(1,N,num=N), columns=["firm_id"])
firm=pd.concat([firm]*(T+1),ignore_index=True)
time=pd.Series(np.linspace(0,T,num=T+1)).repeat(N)
time=pd.DataFrame(time, columns=["time"]).reset_index()

# Build dataframe
df=firm.join(time['time']).join(q).join(q1)\
.join(p).join(p1)\
.join(D).join(D1)\
.join(k).join(k1)\
.join(l).join(l1)\
.join(m).join(m1)\
.join(z).join(z1)\
.join(w).join(w1)\
.join(pm).join(pm1)\
.join(WL).join(WH)

df=df.loc[df['time']!=0]        # Drop t=0

# Save
df.to_csv("df.csv",index=False)
```

Python code for estimation

dj_estimation.py

```
import os
os.chdir("/Users/christophersaw/Desktop/Replication")
import pandas as pd
import numpy as np
import linearmodels
from linearmodels import OLS
import statsmodels.api as sm
import scipy as sp
from scipy.optimize import minimize
from sklearn.preprocessing import PolynomialFeatures
import warnings
warnings.filterwarnings("ignore", category=DeprecationWarning)
warnings.filterwarnings("ignore", category=FutureWarning)

# Load data
df=pd.read_csv('df.csv')
n_obs=int(len(df))
n_firms=int(max(df['firm_id']))
n_periods=int(max(df['time']))

### Step 1: Labor-augmenting productivity

# Create variables for m-l and pm-w
df['ml']=df['m']-df['l']                # m_it - l_it
df['pmw']=df['pm']-df['w']                # pm_it - w_it

# OLS regression for equation (4)
dep_var=df['ml']
lin_var=sm.add_constant(df[['pmw']])
ols=OLS(dep_var, lin_var).fit()

# Recover sigma
s=-ols.params[1]

# Recover labor-augmenting productivity
df['WL_hat']=ols.resids/(1-s)
bl=np.exp(df.WL_hat)



### Step 2: Hicks-neutral productivity

# Non-parameteric regression (polynomial approximation)
# set degree of polynomial
```

```
poly=PolynomialFeatures(degree=3)
dep_var=df['q']
# endogenous variables for polynominal regression:
poly_var=df[['k','l','m','pm1','p1','k1','m1','D1','z1','WL_hat']]
poly_var=pd.DataFrame(poly.fit_transform(poly_var))
poly_reg=OLS(dep_var,poly_var).fit() # Regression of q on phi(k,l,m)
phi=np.array(poly_reg.predict()) # Calculate fitted values

# Construct arrays
K=np.exp(df.k)
L=np.exp(df.l)
M=np.exp(df.m)

# Construct function f(x; theta)
def fx(theta):
        fx_theta=np.zeros((n_obs,1))
        X=np.zeros((n_obs,1))
        nu=theta[0]
        bk=theta[1]
        bm=theta[2]
        C=(nu*s)/(s-1)
        p=(s-1)/s
        for i in range(n_obs):
                        X[i]=bk*np.power(K[i],p)+\
                        np.power(bl[i]*L[i],p)+\
                        bm*np.power(M[i],p)
                        fx_theta[i]=C*np.log(X[i])
        return fx_theta


# Construct functions omega and xi
def omega(theta):
        omega_theta=phi-fx(theta) # this gives omega^H + xi^H
        return omega_theta


Z0=np.array(df.z)[0:n_obs-n_firms]
Z0=np.reshape(Z0,(n_obs-n_firms,1))
def xi(theta):
        W1=np.array(omega(theta))[n_firms:] # remove first n_firms in year 1
        W1=np.reshape(W1,(n_obs-n_firms,1))
        W0=np.array(omega(theta))[0:n_obs-n_firms] # remove last n_firms in year T
        W0=np.reshape(W0,(n_obs-n_firms,1))
        WZ=np.concatenate((W0,Z0),axis=1)
        beta=np.linalg.inv(WZ.transpose() @ WZ) @ WZ.transpose() @ W1 # OLS
        xi_theta=W1 - (WZ @ beta) # Calculate residuals xi^H
        return xi_theta
```

```
# Instruments for GMM
Z=np.array(sm.add_constant(df[['l1','m1','w1','pm1','D1']]))
Z=Z[n_firms:,0:]

# GMM objective function
weightmatrix=np.linalg.inv(Z.transpose() @ Z) # optimal weighting matrix for GMM
def gmmobjfn(theta):
        gmmobjfn=xi(theta).transpose() @ Z @ weightmatrix @ Z.transpose() @ xi(theta)
        return gmmobjfn

# Solve for theta^H
theta0=np.array([[1],[0.5],[0.5]])
results=minimize(gmmobjfn,theta0,\
        method='Nelder-Mead',options={'maxiter':1000})

theta_H=results.x

# Compare estimates with simulated values
df=df.loc[df['time']!=1]        # Drop t=1
df['WH_hat']=omega(theta_H)[n_firms:]-xi(theta_H)

simH=df.WH
estH=df.WH_hat
OLS(simH,estH).fit()

simL=df.WL
estL=df.WL_hat
OLS(simL,estL).fit()
```