Problem Set #1

DSGE Models, Prof. Kerk Phillips
Martina Fraschini

Exercise 1
In the Brock and Mirman's model the households solves the following dynamic program:

$$V\left(K_t, z_t\right) = \max_{K_{t+1}} \ln\left(e^{z_t} K_t^{\alpha} - K_{t+1}\right) + \beta E_t \left\{V\left(K_{t+1}, z_{t+1}\right)\right\},$$

where the law of motion is:

$$z_{t+1} = \rho z_t + \varepsilon_t; \quad \varepsilon_t \sim i.i.d\left(0, \sigma^2\right).$$

The associated Euler equation is:

$$\frac{1}{e^{z_t} K_t^{\alpha} - K_{t+1}} = \beta E_t \left\{\frac{\alpha e^{z_{t+1}} K_{t+1}^{\alpha-1}}{e^{z_{t+1}} K_{t+1}^{\alpha} - K_{t+2}}\right\}.$$

In order to find an algebraic solution for the policy function we use the "guess and verify" method. We guess that the policy function is in the form of $K_{t+1} = Ae^{z_t} K_t^{\alpha}$ and we substitute it in the Euler equation. We obtain:

$$\frac{1}{e^{z_t} K_t^{\alpha} - Ae^{z_t} K_t^{\alpha}} = \beta E_t \left\{\frac{\alpha e^{z_{t+1}} A^{\alpha-1} e^{(\alpha-1)z_t} K_t^{\alpha^2-\alpha}}{e^{z_{t+1}} A^{\alpha} e^{\alpha z_t} K_t^{\alpha^2} - A^{1+\alpha} e^{z_{t+1}} e^{\alpha z_t} K_t^{\alpha^2}}\right\},$$

$$\iff \frac{1}{(1-A) e^{z_t} K_t^{\alpha}} = \beta E_t \left\{\frac{\alpha A^{-1} e^{-z_t} K_t^{-\alpha} A^{\alpha} e^{\alpha z_t} K_t^{\alpha} e^{z_{t+1}}}{(1-A) A^{\alpha} e^{\alpha z_t} K_t^{\alpha^2} e^{z_{t+1}}}\right\},$$

$$\iff \frac{1}{(1-A) e^{z_t} K_t^{\alpha}} = \beta E_t \left\{\frac{\alpha}{(1-A) A e^{z_t} K_t^{\alpha}}\right\},$$

$$\iff \alpha\beta = \frac{(1-A) A e^{z_t} K_t^{\alpha}}{(1-A) e^{z_t} K_t^{\alpha}},$$

$$\iff A = \alpha\beta.$$

Therefore the policy function is $K_{t+1} = \alpha\beta\, e^{z_t} K_t^{\alpha}$.

Exercise 2
Our baseline model with:

$$u\left(c_t, \ell_t\right) = \ln c_t + a \ln\left(1 - \ell_t\right),$$

and

$$f\left(K_t, L_t, z_t\right) = e^{z_t} K_t^{\alpha} L_t^{1-\alpha},$$

has the following characterizing equations:

$$c_t = (1-\tau)\left[w_t\ell_t + (r_t-\delta)\,k_t\right] + k_t + T_t - k_{t+1},$$

$$\frac{1}{c_t} = \beta E_t\left\{\frac{1}{c_{t+1}}\left[(r_{t+1}-\delta)(1-\tau)+1\right]\right\},$$

$$\frac{a}{1-\ell_t} = \frac{1}{c_t}w_t(1-\tau),$$

$$r_t = \alpha e^{z_t}k_t^{\alpha-1}\ell_t^{1-\alpha},$$

$$w_t = (1-\alpha)e^{z_t}k_t^{\alpha}\ell_t^{-\alpha},$$

$$\tau\left[w_t\ell_t + (r_t-\delta)\,k_t\right] = T_t,$$

$$z_t = (1-\rho_z)\,\bar{z} + \rho_z z_{t-1} + \epsilon_t^z; \quad \epsilon_t^z \sim \text{ i.i.d. }\left(0,\sigma_z^2\right).$$

In this case we can't use the same tricks as in Exercise 1 to solve for the policy function because the model is too complex and it's impossible to guess a good algebraic solution.

Exercise 3
Our baseline model with:

$$u\left(c_t,\ell_t\right) = \frac{c_t^{1-\gamma}-1}{1-\gamma} + a\ln\left(1-\ell_t\right),$$

and

$$f\left(K_t,L_t,z_t\right) = e^{z_t}K_t^{\alpha}L_t^{1-\alpha},$$

has the following characterizing equations:

$$c_t = (1-\tau)\left[w_t\ell_t + (r_t-\delta)\,k_t\right] + k_t + T_t - k_{t+1},$$

$$c_t^{-\gamma} = \beta E_t\left\{c_{t+1}^{-\gamma}\left[(r_{t+1}-\delta)(1-\tau)+1\right]\right\},$$

$$\frac{a}{1-\ell_t} = c_t^{-\gamma}w_t(1-\tau),$$

$$r_t = \alpha e^{z_t}k_t^{\alpha-1}\ell_t^{1-\alpha},$$

$$w_t = (1-\alpha)e^{z_t}k_t^{\alpha}\ell_t^{-\alpha},$$

$$\tau\left[w_t\ell_t + (r_t-\delta)\,k_t\right] = T_t,$$

$$z_t = (1-\rho_z)\,\bar{z} + \rho_z z_{t-1} + \epsilon_t^z; \quad \epsilon_t^z \sim \text{ i.i.d. }\left(0,\sigma_z^2\right).$$

Exercise 4
Our baseline model with:

$$u\left(c_t,\ell_t\right) = \frac{c_t^{1-\gamma}-1}{1-\gamma} + a\frac{\left(1-\ell_t\right)^{1-\xi}-1}{1-\xi},$$

and

$$f\left(K_t, L_t, z_t\right)=e^{z_t}\left[\alpha K_t^\eta+(1-\alpha) L_t^\eta\right]^{\frac{1}{\eta}},$$

has the following characterizing equations:

$$c_t=(1-\tau)\left[w_t \ell_t+\left(r_t-\delta\right) k_t\right]+k_t+T_t-k_{t+1},$$

$$c_t^{-\gamma}=\beta E_t\left\{c_{t+1}^{-\gamma}\left[\left(r_{t+1}-\delta\right)(1-\tau)+1\right]\right\},$$

$$a\left(1-\ell_t\right)^{-\xi}=c_t^{-\gamma} w_t(1-\tau),$$

$$r_t=\alpha e^{z_t} k_t^{\eta-1}\left[\alpha k_t^\eta+(1-\alpha) \ell_t^\eta\right]^{\frac{1-\eta}{\eta}},$$

$$w_t=(1-\alpha) e^{z_t} \ell_t^{\eta-1}\left[\alpha k_t^\eta+(1-\alpha) \ell_t^\eta\right]^{\frac{1-\eta}{\eta}},$$

$$\tau\left[w_t \ell_t+\left(r_t-\delta\right) k_t\right]=T_t,$$

$$z_t=\left(1-\rho_z\right) \bar{z}+\rho_z z_{t-1}+\epsilon_t^z ; \quad \epsilon_t^z \sim \text { i.i.d. }\left(0, \sigma_z^2\right).$$

Exercise 5

Assume $\ell_t=1$. For the market clearing conditions we also have that $L_t=\ell_t=1$. Our baseline model with:

$$u\left(c_t, \ell_t\right)=\frac{c_t^{1-\gamma}-1}{1-\gamma},$$

and

$$f\left(K_t, L_t, z_t\right)=K_t^\alpha\left(L_t e^{z_t}\right)^{1-\alpha},$$

has the following characterizing equations:

$$c_t=(1-\tau)\left[w_t \ell_t+\left(r_t-\delta\right) k_t\right]+k_t+T_t-k_{t+1},$$

$$c_t^{-\gamma}=\beta E_t\left\{c_{t+1}^{-\gamma}\left[\left(r_{t+1}-\delta\right)(1-\tau)+1\right]\right\},$$

$$c_t^{-\gamma} w_t(1-\tau)=0,$$

$$r_t=\alpha k_t^{\alpha-1}\left(\ell_t e^{z_t}\right)^{1-\alpha},$$

$$w_t=(1-\alpha) k_t^\alpha \ell_t^{-\alpha} e^{(1-\alpha) z_t},$$

$$\tau\left[w_t \ell_t+\left(r_t-\delta\right) k_t\right]=T_t,$$

$$z_t=\left(1-\rho_z\right) \bar{z}+\rho_z z_{t-1}+\epsilon_t^z ; \quad \epsilon_t^z \sim \text { i.i.d. }\left(0, \sigma_z^2\right).$$

The steady state equations, considering $\ell_t=1$, are:

$$\bar{c}=(1-\tau)\left[\bar{w}+(\bar{r}-\delta) \bar{k}\right]+\bar{T},$$

$$\bar{c}^{-\gamma}=\beta E_t\left\{\bar{c}^{-\gamma}\left[(\bar{r}-\delta)(1-\tau)+1\right]\right\},$$

$$
\begin{aligned}
\bar{c}^{-\gamma}\bar{w}(1-\tau) &= 0, \\
\bar{r} &= \alpha\bar{k}^{\alpha-1}e^{(1-\alpha)\bar{z}}, \\
\bar{w} &= (1-\alpha)\bar{k}^{\alpha}e^{(1-\alpha)\bar{z}}, \\
\tau\left[\bar{w}+(\bar{r}-\delta)\,\bar{k}\right] &= \bar{T}.
\end{aligned}
$$

Therefore, the steady state for the value of k is given by:

$$
\bar{k} = \left[\frac{1}{\alpha}\left(\frac{1-\beta}{\beta(1-\tau)}+\delta\right)\right]^{\frac{1}{\alpha-1}} e^{\bar{z}}.
$$

The numerical solution is presented below.

In [1]:
```
# import packages
import numpy as np
from matplotlib import pyplot as plt
from scipy import optimize

# define system of characterizing equations
def charact_eq(x, p):
    gamma, beta, alpha, delta, tau = p
    return [x[0] - x[1] - (x[2]-delta)*x[3],
            x[0]**(-gamma) - beta*x[0]**(-gamma) * ((x[2]-delta)*(1-tau)+1),
            x[2] - alpha * x[3]**(alpha-1),
            x[1] - (1-alpha) * x[3]**alpha,
            x[4] - tau*(x[1] + (x[2]-delta)*x[3])]

# solve the system
param = [2.5, .98, .4, .1, .05]
X = optimize.root(charact_eq,[.5,.5,.5,.5,.5], args=param)

# present results
c, w, r, k, T = X.x
y = k ** .4
i = .1 * k

print("Steady state values:\n")
print("{:<15}{:<5}{:<5}".format('Consumption','c',round(c,4)))
print("{:<15}{:<5}{:<5}".format('Wage rate','w',round(w,4)))
print("{:<15}{:<5}{:<5}".format('Rental rate','r',round(r,4)))
print("{:<15}{:<5}{:<5}".format('Capital','k',round(k,4)))
print("{:<15}{:<5}{:<5}".format('Transfer','T',round(T,4)))
print("{:<15}{:<5}{:<5}".format('Output','y',round(y,4)))
print("{:<15}{:<5}{:<5}".format('Investment','i',round(i,4)))
```

```
gamma, beta, alpha, delta, tau = param
print("\nThe algebraic solution for capital is k = ", round(((1/alpha)*((1-beta)
```

```
Steady state values:

Consumption    c    1.4845
Wage rate      w    1.328
Rental rate    r    0.1215
Capital        k    7.2875
Transfer       T    0.0742
Output         y    2.2133
Investment     i    0.7287

The algebraic solution for capital is k =  7.2875
```

Exercise 6
Our baseline model with:

$$u\left(c_t,\ell_t\right)=\frac{c_t^{1-\gamma}-1}{1-\gamma}+a\frac{\left(1-\ell_t\right)^{1-\xi}-1}{1-\xi},$$

and

$$f\left(K_t,L_t,z_t\right)=K_t^{\alpha}\left(L_te^{z_t}\right)^{1-\alpha},$$

has the following characterizing equations:

$$c_t=(1-\tau)\left[w_t\ell_t+\left(r_t-\delta\right)k_t\right]+k_t+T_t-k_{t+1},$$

$$c_t^{-\gamma}=\beta E_t\left\{c_{t+1}^{-\gamma}\left[\left(r_{t+1}-\delta\right)(1-\tau)+1\right]\right\},$$

$$-a(1-\ell_t)^{-\xi}=c_t^{-\gamma}w_t(1-\tau),$$

$$r_t=\alpha k_t^{\alpha-1}\left(\ell_te^{z_t}\right)^{1-\alpha},$$

$$w_t=(1-\alpha)k_t^{\alpha}\ell_t^{-\alpha}e^{(1-\alpha)z_t},$$

$$\tau\left[w_t\ell_t+\left(r_t-\delta\right)k_t\right]=T_t,$$

$$z_t=\left(1-\rho_z\right)\bar{z}+\rho_zz_{t-1}+\epsilon_t^z;\quad \epsilon_t^z\sim \text{i.i.d.}\left(0,\sigma_z^2\right).$$

The steady state equations, considering $\ell_t=1$, are:

$$\bar{c}=(1-\tau)\left[\bar{w}\bar{\ell}+\left(\bar{r}-\delta\right)\bar{k}\right]+\bar{T},$$

$$\bar{c}^{-\gamma}=\beta E_t\left\{\bar{c}^{-\gamma}\left[\left(\bar{r}-\delta\right)(1-\tau)+1\right]\right\},$$

$$a(1-\bar{\ell})^{-\xi}=\bar{c}^{-\gamma}\bar{w}(1-\tau),$$

$$\bar{r}=\alpha\bar{k}^{\alpha-1}\bar{\ell}^{1-\alpha}e^{(1-\alpha)\bar{z}},$$

$$\bar{w} = (1-\alpha)\bar{k}^{\alpha}\bar{\ell}^{-\alpha}e^{(1-\alpha)\bar{z}},$$

$$\tau\left[\bar{w}\bar{\ell} + (\bar{r}-\delta)\,\bar{k}\right] = \bar{T}.$$

The numerical solution is presented below.

In [2]:
```
# import packages
import numpy as np
from matplotlib import pyplot as plt
from scipy import optimize

# define system of characterizing equations
def charact_eq(x, p):
    gamma, xi, beta, alpha, a, delta, tau = p
    return [x[0] - x[1]*x[5] - (x[2]-delta)*x[3],
            x[0]**(-gamma) - beta*x[0]**(-gamma) * ((x[2]-delta)*(1-tau)+1),
            x[0]**(-gamma)*x[1]*(1-tau) - a*(1-x[5])**(-xi),
            x[2] - alpha * x[3]**(alpha-1) * x[5]**(1-alpha),
            x[1] - (1-alpha) * x[3]**alpha * x[5]**(-alpha),
            x[4] - tau*(x[1]*x[5] + (x[2]-delta)*x[3])]

# solve the system
param = [2.5, 1.5, .98, .4, .5, .1, .05]
X = optimize.root(charact_eq,[.5,.5,.5,.5,.5,.5], args=param)

# present results
c, w, r, k, T, l = X.x
y = (k ** .4)*(l ** .6)
i = .1 * k

print("Steady state values:\n")
print("{:<15}{:<5}{:<5}".format('Consumption','c',round(c,4)))
print("{:<15}{:<5}{:<5}".format('Wage rate','w',round(w,4)))
print("{:<15}{:<5}{:<5}".format('Rental rate','r',round(r,4)))
print("{:<15}{:<5}{:<5}".format('Capital','k',round(k,4)))
print("{:<15}{:<5}{:<5}".format('Transfer','T',round(T,4)))
print("{:<15}{:<5}{:<5}".format('Labor','l',round(l,4)))
print("{:<15}{:<5}{:<5}".format('Output','y',round(y,4)))
print("{:<15}{:<5}{:<5}".format('Investment','i',round(i,4)))
```

```
Steady state values:

Consumption    c    0.8607
Wage rate      w    1.328
Rental rate    r    0.1215
Capital        k    4.2252
```

```
Transfer       T    0.043
Labor          l    0.5798
Output         y    1.2832
Investment     i    0.4225
```

Exercise 7

```
In [3]: # import packages
        import numpy as np
        from matplotlib import pyplot as plt
        from scipy import optimize

        # define system of characterizing equations
        def charact_eq(x, p):
            gamma, xi, beta, alpha, a, delta, tau = p
            return [x[0] - x[1]*x[5] - (x[2]-delta)*x[3],
                    x[0]**(-gamma) - beta*x[0]**(-gamma) * ((x[2]-delta)*(1-tau)+1),
                    x[0]**(-gamma)*x[1]*(1-tau) - a*(1-x[5])**(-xi),
                    x[2] - alpha * x[3]**(alpha-1) * x[5]**(1-alpha),
                    x[1] - (1-alpha) * x[3]**alpha * x[5]**(-alpha),
                    x[4] - tau*(x[1]*x[5] + (x[2]-delta)*x[3])]

        # define parameters and initial value
        param = [2.5, 1.5, .98, .4, .5, .1, .05]
        x0 = [.5,.5,.5,.5,.5,.5]

        # model at the steady state
        X = optimize.root(charact_eq,x0, args=param)
        c, w, r, k, T, l = X.x
        y = (k ** .4)*(l ** .6)
        i = .1 * k

        # compute derivatives
        h = 1e-5
        dd = np.empty([len(x0)+2,len(param)])
        for ip in range(len(param)):
            h_param = param.copy()
            h_param[ip] += h
            h_X = optimize.root(charact_eq,x0, args=h_param)
            for ix in range(len(x0)):
                dd[ix,ip] = (h_X.x[ix] - X.x[ix]) / h
            h_y = (h_X.x[3] ** .4)*(h_X.x[5] ** .6)
            h_i = .1 * h_X.x[3]
```

```python
        dd[-2,ip] = (h_y - y) / h
        dd[-1,ip] = (h_i - i) / h

    namevar = ['c', 'w', 'r', 'k', 'T', 'l', 'y', 'i']
    # present results
    print("{:<5}{:>10}{:>10}{:>10}{:>10}{:>10}{:>10}{:>10}".format('','gamma', 'xi',
    print("-"*75)
    for ix in range(len(x0)+2):
        print("{:<5}{:>10}{:>10}{:>10}{:>10}{:>10}{:>10}{:>10}".format(namevar[ix],r
```

```
          gamma        xi      beta     alpha         a     delta       tau
---------------------------------------------------------------------------
c        0.0283   -0.1633     1.751    2.0853   -0.3767   -3.5111   -0.2344
w          -0.0       0.0    7.9879    4.3961       0.0    -7.287   -0.1648
r          -0.0      -0.0    -1.096       0.0      -0.0       1.0    0.0226
k        0.1387   -0.8017   65.4385   25.9858   -1.8492  -48.3453   -2.3232
T        0.0014   -0.0082    0.0876    0.1043   -0.0188   -0.1756     0.849
l         0.019     -0.11    0.2603   -0.7694   -0.2538    1.3197   -0.1389
y        0.0421   -0.2435    8.2949     2.135   -0.5616   -4.1209   -0.4667
i        0.0139   -0.0802    6.5438    2.5986   -0.1849   -4.8345   -0.2323
```