# Theory of Income 3 PS 1: TA answer

Hyejin Park*

April 8, 2019

## 1 Neoclassical Growth Model with Trending Hours

1. Construct Lagrangian.

$$\mathcal{L} = \max_{\{C_t, H_t, K_{t+1}\}_{t=0}^{\infty}} \sum_{t=0} \beta^t \left\{ \frac{C_t^{1-\sigma} - 1}{1-\sigma} - \frac{\gamma\varepsilon}{1+\varepsilon} H_t^{\frac{1+\varepsilon}{\varepsilon}} + \lambda_t \left( K_t^{\alpha} (Z_t H_t)^{1-\alpha} - K_{t+1} + (1-\delta) K_t - C_t \right) \right\}$$

First order conditions are

$$\begin{aligned}
\partial C_t; \quad & \beta^t C_t^{-\sigma} = \beta^t \lambda_t \\
\partial H_t; \quad & \beta^t \gamma H_t^{\frac{1}{\varepsilon}} = \beta^t \lambda_t (1-\alpha) K_t^{\alpha} Z_t^{1-\alpha} H_t^{-\alpha} \\
\partial K_{t+1}; \quad & \beta^t \lambda_t = \beta^{t+1} \lambda_{t+1} \left( 1 - \delta + \alpha K_{t+1}^{\alpha-1} \left( Z_{t+1} H_{t+1} \right)^{1-\alpha} \right)
\end{aligned}$$

Combine $\partial C_t$ with $\partial H_t$ and then with $\partial K_{t+1}$.

$$\begin{aligned}
\gamma H_t^{\frac{1}{\varepsilon}} C_t^{\sigma} &= (1-\alpha) K_t^{\alpha} Z_t^{1-\alpha} H_t^{-\alpha} \\
\left( \frac{C_t}{C_{t+1}} \right)^{-\sigma} &= \beta \left( 1 - \delta + \alpha K_t^{\alpha-1} \left( Z_{t+1} H_{t+1} \right)^{1-\alpha} \right)
\end{aligned}$$

Impose Balanced Growth Path, i.e. let $C_t = (1+g)^t c^*$, $K_t = (1+g)^t k^*$, $Z_t = (1+g_z)^t$, and $H_t = (1+g_h)^t h^*$. Then the equations become

$$\begin{aligned}
\gamma (1+g_h)^{\frac{t}{\varepsilon}} h^{*\frac{1}{\varepsilon}} (1+g)^{\sigma t} c^{*\sigma} &= (1-\alpha)(1+g)^{\alpha t} k^{*\alpha} (1+g_z)^{(1-\alpha)t} (1+g_h)^{(1-\alpha)t} (1+g_h)^{-\alpha t} h^{*-\alpha} \\
(1+g)^{\sigma} &= \beta \left( 1 - \delta + \alpha (1+g)^{(\alpha-1)(t+1)} k^{*\alpha-1} (1+g_z)^{(1-\alpha)(t+1)} (1+g_h)^{-\alpha(t+1)} h^{*(1-\alpha)} \right)
\end{aligned}$$

*University of Chicago; Email Address: hjpark0619@uchicago.edu; Please email me if you see any error.

Note that this has to hold for all $t$. Then, the growth rates have to satisfy the two following equations.

$$(1+g_h)^{\frac{1}{\varepsilon}}(1+g)^{\sigma} = (1+g)^{\alpha}(1+g_z)^{1-\alpha}(1+g_h)^{-\alpha}$$
$$(1+g) = (1+g_z)(1+g_h)$$

Solving these for $1+g_h$ and $1+g$,

$$1+g_h = (1+g_z)^{\frac{(1-\sigma)\varepsilon}{1+\varepsilon\sigma}}$$
$$1+g = (1+g_z)^{\frac{1+\varepsilon}{1+\varepsilon\sigma}}$$

Since $g_z > 0$, $g_h < 0$ if and only if $\sigma > 1$.

Given $g_z$ and $\sigma > 1$, $g_h$ is larger when $\varepsilon$ is smaller and $\sigma$ is smaller. This is intuitive, since smaller $\varepsilon$ implies the substitution effect for labor supply is smaller, so income effect is more likely to dominate. Moreover, smaller $\sigma$ implies the consumption smoothing incentive of representative agent is smaller, so that the capital increases at a higher rate and the substitution with capital is faster.

2. Let's first start with the simpler ones

$$1.005 = 1+g == (1+g_z)(0.999)$$
$$\Rightarrow 1+g_z = 1.005/0.999 = 1.006$$
$$1.005 = 1+g == (1.006)^{\frac{3}{1+2\sigma}}$$
$$\Rightarrow \sigma = \frac{1}{2}\left(3\frac{\log 1.006}{\log 1.005} - 1\right) = 1.3009$$
$$\alpha = 1 - L_t w_t / Y_t = 1 - 0.6 = 0.4$$
$$0.26 = \frac{X_t}{Y_t} = \frac{(1+g)K_t - (1-\delta)K_t}{Y_t} = (g+\delta)\frac{K_t}{Y_t} = (0.005+\delta)3.2 * 4$$
$$\Rightarrow \delta = 0.0153$$
$$(1+g)^{\sigma} = \beta(1-\delta+\alpha Y_t/K_t)$$
$$\Rightarrow \beta = 0.9907;$$

Note the three equations that characterize $k^*$, $c^*$, and $h^*$.

$$\gamma h^{*\frac{1}{\varepsilon}} c^{*\sigma} = (1-\alpha) k^{*\alpha} h^{*-\alpha}$$

$$(1+g)^{\sigma} = \beta(1-\delta+\alpha(h^{*}/k^{*})^{1-\alpha})$$

$$h^{*(1-\alpha)} k^{*\alpha} = (\delta+g)k^{*} + c^{*}$$

Use the second equation to get the $h^*/k^* = 250/k^* = 0.0017$ and then $k^* = 17510$. Substitute this into the third equation, $c^* = 1012$, and $\gamma = 0.000025569$.

Calibration code follows.

```
g       = 0.005;
gh      = -.001;
kovery  = 3.2*4;
xovery  = 0.26;
epsilon = 2;
hstar   = 1000/4;

alpha   = 0.4;
gz      = (1+g)/(1+gh)-1;
sigma   = (3*log(1+gz)/log(1+g) - 1)/2;
delta   = xovery/kovery-g;
beta    = (1+g)^sigma/(1-delta+alpha/kovery);

hoverk  = (((1+g)^sigma/beta-1+delta)/alpha)^(1/(1-alpha));
kstar   = hstar/hoverk;

cstar   = hstar^(1-alpha)*kstar^alpha-(delta+g)*kstar;
ystar   = hstar^(1-alpha)*kstar^alpha;
xstar   = (g + delta)*kstar;

gamma   = (1-alpha)*kstar^alpha*hstar^(-alpha)/hstar^(1/epsilon)/
   cstar^sigma;
```

3. Now the question is to find $g$, $g_h$ $k^*$, $h^*$ and $c^*$ that satisfy the five equations

$$\begin{aligned}
\gamma h^{*\frac{1}{\varepsilon}} c^{*\sigma} &= (1-\alpha) k^{*\alpha} h^{*-\alpha} \\
(1+g)^\sigma &= \beta(1-\delta+\alpha(h^*/k^*)^{1-\alpha}) \\
h^{*(1-\alpha)} k^{*\alpha} &= (\delta+g)k^* + c^* \\
1+g_h &= (1+g_z)^{\frac{(1-\sigma)\varepsilon}{1+\varepsilon\sigma}} \\
1+g &= (1+g_z)^{\frac{1+\varepsilon}{1+\varepsilon\sigma}}
\end{aligned}$$

Again the second gives $\chi := h^*/k^* = 0.0017$. Then the third equation becomes

$$\begin{aligned}
h^* \chi^{-\alpha} &= (\delta+g)h^*/\chi + c^* \\
\Rightarrow c^* &= \left(\chi^{-\alpha} - \frac{\delta+g}{\chi}\right) h^*
\end{aligned}$$

Plugging this into the first equation,

$$\gamma \left(\chi^{-\alpha} - \frac{\delta+g}{\chi}\right)^\sigma h^{*\frac{1}{\varepsilon}+\sigma} = (1-\alpha)\chi^{-\alpha}$$

Solving these gives $g = 0.005$, $g_h = -0.001$, $h^* = 1000$, $c^* = 1012$, and $k^* = 17510$, same as in 2..

Code to find the steady state follows.

```
gh      = (1+gz)^((1-sigma)*epsilon/(1+epsilon*sigma))-1;
g       = (1+gz)^((1+epsilon)/(1+epsilon*sigma))-1;

chi     = (((1+g)^sigma/beta-1+delta)/alpha)^(1/(1-alpha));
hstar   = ((1-alpha)*chi^(-alpha)/(chi^(-alpha)-(delta+g)/chi)^
   sigma/gamma)^(1/(1/epsilon+sigma));
kstar   = hstar/chi;
cstar   = (chi^(-alpha)-(delta+g)/chi)*hstar;
ystar   = kstar^alpha*hstar^(1-alpha);
```

4. Let $\log k_t = \log K_t - \log(k^*(1+g)^t)$, $\log c_t = \log C_t - \log(c^*(1+g)^t)$, and $\log h_t = \log H_t - \log(h^*(1+g_h)^t)$.

$$\begin{aligned}
\gamma H_t^{\frac{1}{\varepsilon}} C_t^{\sigma} &= (1-\alpha) K_t^{\alpha} Z_t^{1-\alpha} H_t^{-\alpha} \\
\gamma (1+g_h)^{\frac{t}{\varepsilon}} h^{*\frac{1}{\varepsilon}} (1+g)^{\sigma t} c^{*\sigma} &= (1-\alpha)(1+g)^{\alpha t} k^{*\alpha} (1+g_z)^{(1-\alpha)t} (1+g_h)^{-\alpha t} h^{*-\alpha} \\
\Rightarrow h_t^{\frac{1}{\varepsilon}} c_t^{\sigma} &= k_t^{\alpha} h_t^{-\alpha}
\end{aligned}$$

Note that this equation is a contemporaneous relationship between $c_t, h_t, k_t$. We will use this equation to substitute $h_t$ in the following difference equations.

$$\begin{aligned}
\left(\frac{C_t}{C_{t+1}}\right)^{-\sigma} &= \beta\left(1-\delta+\alpha K_t^{\alpha-1}\left(Z_{t+1}H_{t+1}\right)^{1-\alpha}\right) \\
\Rightarrow (1+g)^{\sigma}\left(\frac{c_t}{c_{t+1}}\right)^{-\sigma} &= \beta\left(1-\delta+\alpha k_{t+1}^{\alpha-1} h_{t+1}^{1-\alpha} (h^*/k^*)^{1-\alpha}\right) \\
K_t^{\alpha}\left(Z_t H_t\right)^{1-\alpha} &= K_{t+1} - (1-\delta)K_t + C_t \\
\Rightarrow k_t^{\alpha} h_t^{1-\alpha} &= \frac{(1+g)k^*}{k^{*\alpha} h^{*(1-\alpha)}} k_{t+1} - (1-\delta)\frac{k^*}{k^{*\alpha} h^{*(1-\alpha)}} k_t + \frac{c^*}{k^{*\alpha} h^{*(1-\alpha)}} c_t
\end{aligned}$$

To find the $c_0$ on the stable arm, I coded the following algorithm, which is basically some variation of bisection method.

(a) Fix $[c_l, c_u]$, an interval where the initial consumption is. Note that this has to be wide enough so that $c_0 = c_l$ gives diverging capital path and $c_0 = c_u$ gives capital path shrinks to zero.

(b) Guess $c_0^{guess} = (c_l + c_u)/2$.

(c) Run the difference equation forward, to get $\{c_t\}_{t=0}^T$ and $\{k_t\}_{t=0}^T$, where $T$ is large enough to give $c_T \approx 1$ and $k_T \approx 1$ if $(k_0, c_0^{guess})$ is on the stable arm.

(d) Check if $||k_T - 1|| < \varepsilon$, where $\varepsilon$ is the tolerance level. If $k_T \geq 1+\varepsilon$, substitute $c_l = c_0^{guess}$ and go back to (b). If $k_T \leq 1-\varepsilon$, substitute $c_u = c_0^{guess}$ and go back to (b).

For implementation, see the following code. The results are drawn on the last page. Worked hours first starts above the balanced growth path, decreases over time, and converges to the balanced growth path.

```
k0           = kstar/2;
T            = 100; % has to be large enough
kovery       = kstar/ystar;
covery       = cstar/ystar;

% I will use Bisection method to find c0
% first lower bound
cl           = 0.001;
% check this is a good lower bound
cpath        = zeros(T, 1);
kpath        = zeros(T, 1);
hpath        = zeros(T, 1);
kpath(1)     = k0/kstar;
cpath(1)     = cl;
hpath(1)     = (kpath(1)^alpha*cpath(1)^(-sigma))^(1/(1/epsilon+
   alpha));

for t = 2:T
    kpath(t)     = (kpath(t-1)^alpha*hpath(t-1)^(1-alpha)+(1-delta)
       *kovery*kpath(t-1)-covery*cpath(t-1))/(1+g)/kovery;
    if kpath(t) > 2
        disp("k diverges; good lower bound for c")
        break
    end
    hfun         = @(c) (kpath(t)^alpha*c^(-sigma))^(1/(1/epsilon+
       alpha));
    eefun        = @(c) (1+g)^sigma*(cpath(t-1)/c)^(-sigma)-beta
       *(1-delta + alpha*kpath(t)^(alpha-1)*hfun(c)^(1-alpha)*(
       hstar/kstar)^(1-alpha));
    cpath(t)     = fzero(eefun, [0.001, 10]);
    hpath(t)     = hfun(cpath(t));
end

% then upper bound;
cu   = 2;
% check this is a good upper bound
cpath        = zeros(T, 1);
```

```
kpath        = zeros(T, 1);
hpath        = zeros(T, 1);
kpath(1)     = k0/kstar;
cpath(1)     = cu;
hpath(1)     = (kpath(1)^alpha*cpath(1)^(-sigma))^(1/(1/epsilon+
   alpha));

for t = 2:T
    kpath(t)     = (kpath(t-1)^alpha*hpath(t-1)^(1-alpha)+(1-delta)
       *kovery*kpath(t-1)-covery*cpath(t-1))/(1+g)/kovery;
    if kpath(t) < 0.01
        disp("k shrinks to zero; good upper bound for c")
        break
    end
    hfun         = @(c) (kpath(t)^alpha*c^(-sigma))^(1/(1/epsilon+
       alpha));
    eefun        = @(c) (1+g)^sigma*(cpath(t-1)/c)^(-sigma)-beta
       *(1-delta + alpha*kpath(t)^(alpha-1)*hfun(c)^(1-alpha)*(
       hstar/kstar)^(1-alpha));
    cpath(t)     = fzero(eefun, [0.001, 10]);
    hpath(t)     = hfun(cpath(t));
end

% bisection
tolx    = 1e-6;
x       = 1;
maxiter = 100;

for i = 1:maxiter
    c0           = (cl + cu)/2;
    cpath        = zeros(T, 1);
    kpath        = zeros(T, 1);
    hpath        = zeros(T, 1);
    kpath(1)     = k0/kstar;
    cpath(1)     = c0;
    hpath(1)     = (kpath(1)^alpha*cpath(1)^(-sigma))^(1/(1/epsilon
       +alpha));

    for t = 2:T
        kpath(t)     = (kpath(t-1)^alpha*hpath(t-1)^(1-alpha)+(1-
           delta)*kovery*kpath(t-1)-covery*cpath(t-1))/(1+g)/
           kovery;
```

```
        if kpath(t) > 2
            x = 10;
            break
        end
        if kpath(t) < 0.01
            x = 0.1;
            break
        end
        hfun        = @(c) (kpath(t)^alpha*c^(-sigma))^(1/(1/
           epsilon+alpha));
        eefun       = @(c) (1+g)^sigma*(cpath(t-1)/c)^(-sigma)-
           beta*(1-delta + alpha*kpath(t)^(alpha-1)*hfun(c)^(1-
           alpha)*(hstar/kstar)^(1-alpha));
        cpath(t)    = fzero(eefun, [0.001, 10]);
        hpath(t)    = hfun(cpath(t));
        x           = kpath(t);
    end
    if x > 1
        cl = c0;
    elseif x < 1
        cu = c0;
    end
    if abs(1-x) < tolx
        fprintf('Shooting algorithm found an equilibrium path; c0
           = %.4f \n', c0*cstar)
        break
    end
    fprintf('Bisection Iteration: c0/cstar= %.8f, k_T = %.8f \n',
       c0, x)
end
```

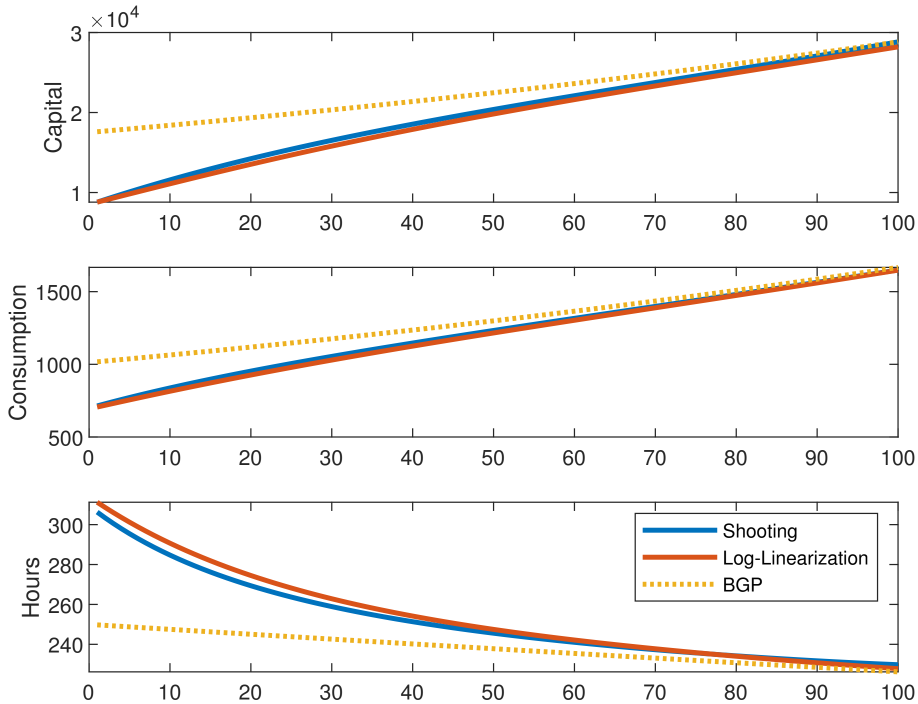
5. See the following code.

```
% for log-linearization you can;
% 1) manually get the derivatives and put them on the computer
% 2) can use symbolic toolbox of MATLAB
% 3) can compute numerical derivatives
% 4) can use automatic differentiation

% in this answer, I will use symbolic toolbox of MATLAB
```

```
% declare symbolic variables
syms capital cons hours capital1 cons1 hours1

mrs      = hours^(1/epsilon)*cons^sigma-capital^alpha*hours^(-alpha
   );
ee       = (1+g)^sigma*(cons/cons1)^(-sigma)-beta*(1-delta+alpha*
   capital1^(alpha-1)*hours1^(1-alpha)*(hstar/kstar)^(1-alpha));
mkcl     = capital^alpha*hours^(1-alpha)-(1+g)*kovery*capital1 +
   (1-delta)*kovery*capital - covery*cons;

dmrsdk   = diff(mrs, capital);
dmrsdc   = diff(mrs, cons);
dmrsdh   = diff(mrs, hours);

deedc    = diff(ee, cons);
deedk1   = diff(ee, capital1);
deedc1   = diff(ee, cons1);
deedh1   = diff(ee, hours1);

dmkcldk = diff(mkcl, capital);
dmkcldc = diff(mkcl, cons);
dmkcldh = diff(mkcl, hours);
dmkcldk1 = diff(mkcl, capital1);

capital = 1;
cons     = 1;
hours    = 1;
capital1 = 1;
cons1    = 1;
hours1   = 1;

dmrsdk   = double(subs(dmrsdk));
dmrsdc   = double(subs(dmrsdc));
dmrsdh   = double(subs(dmrsdh));

deedc    = double(subs(deedc));
deedk1   = double(subs(deedk1));
deedc1   = double(subs(deedc1));
deedh1   = double(subs(deedh1));

dmkcldk = double(subs(dmkcldk));
```

```
dmkcldc = double(subs(dmkcldc));
dmkcldh = double(subs(dmkcldh));
dmkcldk1 = double(subs(dmkcldk1));

% system Phi1*x_t1 = Phi0*x_t where x_t = [log k_t; log c_t; log
   h_t];
Phi1     = [  deedk1 deedc1 deedh1;
             dmkcldk1      0      0;
                    0      0      0];

Phi0     = [       0   -deedc        0;
             -dmkcldk -dmkcldc -dmkcldh;
               dmrsdk   dmrsdc   dmrsdh];

% system reduction Psi1*y_t1 = Psi0*y_t where y_t = [log k_t; log
   c_t];
Psi1     = Phi1(1:2, 1:2) + Phi1(1:2, 3)*(-Phi0(3, 1:2)/Phi0(3, 3))
   ;
Psi0     = Phi0(1:2, 1:2) + Phi0(1:2, 3)*(-Phi0(3, 1:2)/Phi0(3, 3))
   ;

Psi      = inv(Psi1)*Psi0;
[V_Psi, D_Psi] = eig(Psi);

% the first eigenvalue is unstable, while the second one is stable
% then [log k_0; log c_0] has to be in span(v2), while log k_0 =
   log(1/2);
y0       = log(k0/kstar)*V_Psi(:, 2)/V_Psi(1, 2);

llpath   = zeros(2, T);
llpath(:, 1) = y0;
for t=2:T
    llpath(:,t) = Psi*llpath(:,t-1);
end

llkpath      = exp(llpath(1, :));
llcpath      = exp(llpath(2, :));
llhpath      = exp(-Phi0(3, 1:2)*llpath/Phi0(3, 3));
```

Let's compare the equilibrium paths given from the two algorithm.




The two give slightly different path. Log-linearization yields quite a large error because the initial condition on $k_0$ is too far from the steady state.