Dynamic Macroeconomics using Matlab
Seminar 2

Educanet
Edinson Tolentino
email: et396@kent.ac.uk

Twitter: @edutoleraymondi

Content

Ramesy-Cass-Koopmans growth model: ejemplo

- Given the next max problem:

$$\max_{c_t, k_{t+1}} \sum_{t=0}^{\infty} \beta^t u(c_t)$$

$$\text{s.t: } c_t + k_{t+1} = f(k_t) + (1-\delta)k_t$$

- Using:
  - $u(c_t) = log(c_t)$
  - $y_t = k_t^{\alpha}$
  - Depreciation is 100 %
- You need to find:
  1. write the planner's problem
  2. find the steady state of consumption and capital

Social planner's problem: F.O.C

- Lagrangian with multiplier $\lambda_t > 0$ for each resource constraint

$$\mathcal{L} = \sum_{t=0}^{\infty} \beta^t log(c_t) + \sum_{t=0}^{\infty} \lambda_t \left[k_t^{\alpha} - c_t - k_{t+1}\right]$$

- Some key first order conditions

$$\begin{aligned} c_t : & \qquad \beta^t \frac{1}{c_t} - \lambda_t = 0 \\ k_{t+1} : & \qquad -\lambda_t + \lambda_{t+1}\left[\alpha k_{t+1}^{\alpha-1}\right] = 0 \\ \lambda_t : & \qquad k_t^{\alpha} - c_t - k_{t+1} = 0 \end{aligned}$$

- Using first and second equation

$$\lambda_t = \beta^t \frac{1}{c_t} \qquad \lambda_{t+1} = \beta^{t+1} \frac{1}{c-t+1}$$

- Eliminating the Lagrange multipliers (`Euler equation`)

$$\frac{1}{c_t} = \beta \frac{1}{c_{t+1}} \left[\alpha k_{t+1}^{\alpha-1}\right]$$

Dynamical system

- Gives a system of two nolinear difference equations in c_t, k_t

$$\frac{1}{c_t} = \beta \frac{1}{c_{t+1}} \left[\alpha k_{t+1}^{\alpha-1} \right]$$
$$c_t + k_{t+1} = k_t^{\alpha}$$

- Two boundary conditions: (i) initial $k_0 > 0$ given, and (ii) `transversality condition`

$$\lim_{T\to\infty} \beta^T \lambda_T k_{T+1} = 0$$

Steady state

- Steady state where $\triangle c_t = 0$ and $\triangle k_t = 0$, Let c_{ss} k_{ss} denote steady state values. These are determined by

$$c_{t+1} = \beta c_t \left[\alpha k_{t+1}^{\alpha-1}\right]$$
$$c_t + k_{t+1} = k_t^{\alpha}$$

$$c_{ss} = \beta c_{ss} \left[\alpha k_{ss}^{\alpha-1}\right]$$
$$1 = \beta \left[\alpha k_{ss}^{\alpha-1}\right]$$
$$k_{ss} = (\alpha\beta)^{\frac{1}{1-\alpha}}$$

$$c_{ss} + k_{ss} = k_{ss}^{\alpha}$$
$$c_{ss} = k_{ss}^{\alpha} - k_{ss}$$
$$c_{ss} = (\alpha\beta)^{\frac{\alpha}{1-\alpha}} - (\alpha\beta)^{\frac{1}{1-\alpha}}$$

Forward iteration

- We need to compute the Euler equation

$$c_{t+1} = \beta c_t \left[\alpha k_{t+1}^{\alpha-1}\right]$$

- Moreover, we include initial values for k_0 and c_0, after that these variables converge to `steady state`
- Therefore, once to achieve steady state, economy keep this situation at least shock
- Given the example:

$$U(c_t) = lnc_t$$
$$f(k_t) = k_t^{\alpha}$$

  - solve equations given T

$$0 = \left(\frac{k_1^{\alpha} - k_2}{k_0^{\alpha} - k_1}\right) - \alpha\beta k_1^{\alpha-1}$$
$$0 = \left(\frac{k_2^{\alpha} - k_3}{k_1^{\alpha} - k_2}\right) - \alpha\beta k_2^{\alpha-1}$$
$$\vdots$$
$$0 = \left(\frac{k_T^{\alpha} - k_{ss}}{k_{T-1}^{\alpha} - k_T}\right) - \alpha\beta k_T^{\alpha-1}$$

Forward iteration

- Code in matlab

Listing 1: Script

```
%===============================================
% Dynamic Macroeconomics
% Author: Edinson Tolentino
%===============================================
% Ramsey Cass-Koopmans
% FIRST METHODS

clc;
clear all;
clear close;

% parameters
% --------------------------------------------------

alpha   = 0.35;
k0      = 0.075;
beta    = 0.985;
tol     = 0.00001;
T       = 30;
```

Forward iteration

- Code in matlab

Listing 2: Script

```

% steady stacionary
%---------------------------------
ks       = (alpha*beta)^(1/(1-alpha));
x0       = ones(T,1)*0.8*ks;
param    = [alpha k0 beta ks];

% iteration
%----------------------------------
sol      = secant('sys', x0, param);

time = (0:1:T-1); %% time index

figure(1)
plot(time, sol,'r-')
ylabel('Capital Stock')
xlabel('Time')
name = [fig_path '\capital_time' ];
print('-depsc', name)
```

Forward iteration

- function in matlab: `secant`

Listing 3: `Script`

```
% Functions
%----------------------------------
function x = secant(sys, x0, param)

    del = diag(max(abs(x0)*1e-4,1e-8));
    n   = length(x0);
    for i=1:1000
        f=feval(sys,x0, param);
        for j=1:n
            J(:,j)=(f-feval(sys,x0-del(:,j),param))/del(j,j);
        end
        x = x0-inv(J)*f';
        if norm(x-x0)<0.00001
            break;
        end
        x0 = x;
    end
end
```

Forward iteration

- function in matlab: `f`

Listing 4: `Script`

```

function f=sys(z,p)

    a = p(1);
    b = p(2);
    c = p(3);
    d = p(4);

    f(1)=((((z(1)^a)-z(2))/((b^a)-z(1))))-a*c*(z(1)^(a-1));

    for i=2:30-1
    f(i)=((((z(i)^a)-z(i+1))/((z(i-1)^a)-z(i))))-a*c*(z(i)^(a-1));
    end

    i = 30;
    f(i)=((((z(i)^a)-d)/((z(i-1)^a)-z(i))))-a*c*(z(i)^(a-1));
end
```

Forward iteration

- Path for capital stock across 30 periods