Computing Equilibria in a Very Simple Growth Model

Let's start with the very simplest model.

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

subject to

$$c_t + k_{t+1} + (1-\delta)k_t = A_t k_t^{\alpha},$$

and k_0 given. For now, assume that $\delta = 1$, $u(c) = \ln(c)$, and $A_t = A$. Except in the case of the 'guess and verify' method, generalizing to the case with general δ and $u(\cdot)$ and stochastic A_t will hopefully be obvious.

Method 1. Dynamic programming. The Bellman equation in this case is

$$v(k) = \max_{c,k'} \{\ln(c) + \beta v(k')\} \quad \text{s.t.} c + k' = Ak^{\alpha}.$$

By definition, $v(k_0) = \sum_{t=0}^{\infty} \beta^t u(c_t)$ for optimal c_t. Writing this as the sum of two terms, the utility today plus the value of the stock tomorrow, gives Bellman's equation.

We want to compute $v(k)$. Given $v(k)$ we know the optimal $c(k)$ and $k'(k)$. Given k_0, we can generate time series $k_1, k_2, \ldots$ and $c_0, c_1, c_2, \ldots$. Here is where Minnesota grads are different from MIT grads: we compare the time series in our models with the time series in our data directly.

We can compute $v(k)$ many ways. For example, we can guess and verify. Guess a form of the solution to be: $v(k) = a + b\ln(k)$. If we substitute out c, we have

$$v(k) = \max_{k'} \{\ln(Ak^{\alpha} - k') + \beta v(k')\}.$$

Assuming I can differentiate, I have the following first-order condition for the right hand side:

$$\frac{1}{Ak^{\alpha} - k'} = \beta v'(k').$$

If we use our guess, we have

$$\frac{1}{Ak^{\alpha} - k'} = \beta \frac{b}{k'}$$

or

$$k' = \frac{\beta b}{1 + \beta b} Ak^{\alpha}.$$

At this optimum:

$$a + b \ln k = \ln(Ak^\alpha - \frac{\beta b}{1+\beta b} Ak^\alpha) + \beta(a + b \ln[\frac{\beta b}{1+\beta b} Ak^\alpha)]\}.$$

Note that we can equate the coefficient on $\ln k$ to solve for b and the constant terms to solve for a.

How else can I solve for $v(k)$? I can iterate backwards in time. The idea comes from finite time examples. At the last date, the value of any capital after death is zero. Start with $v(k) = 0$. Iterate until convergence. Let's do a few steps.

$$\begin{aligned} v_1(k) &= \max_{k' \geq 0} \{\ln(Ak^\alpha - k')\} \\ &= \ln(Ak^\alpha) \end{aligned}$$

and given $v_1(k)$, we have

$$v_2(k) = \max_{k' \geq 0} \{\ln(Ak^\alpha - k') + \beta \ln(A) + \beta\alpha \ln(k')\}.$$

Taking first order derivatives, we have

$$\frac{1}{Ak^\alpha - k'} = \frac{\beta\alpha}{k'}$$

which implies

$$k' = \frac{\beta\alpha}{1+\beta\alpha} k.$$

Fill this in and we have $v_2(k)$. Keep this going until $v_T(k) \approx v_{T+1}(k)$.

Can we do this on a computer? We have to think about how we are representing the solution. In the case of dynamic programming, we are looking for a solution to a functional equation. We need to think about how we are going to represent that solution. In the very crudest algorithms, we simply assume there is a vector of k, in other words a grid, and a value v for each k. In this very simple case you could think of the value function as a piecewise function.

The second thing that we need to think about is what it means to say that something (e.g., $\hat{v}$) is a solution. In the crudest case, we stop iterating when $\sqrt{\sum(\hat{v}_T - \hat{v}_{T-1})^2}/N$ is small where $\hat{v}_T$ is a vector of values of length N.

Finally, what matters? Suppose we completely miss on our approximation of v at very small or very large k. Suppose also that the question at hand is concerned with business

cycle facts. Then the bad approximation for very low probability events won't affect the analysis.

Method 2. Solving the first order conditions

We can guess and verify in this case as well. Let's jump ahead to do undetermined coefficients. But, here, as in most business cycle models, we will first log-linearize the first order conditions.

The equations we need to linearize are

$$c_t + k_{t+1} = Ak_t^\alpha$$
$$1/c_t = \beta\alpha A k_{t+1}^{\alpha-1}/c_{t+1}.$$

Here are the steps:

1. Get the steady state: $k_{ss} = (\beta\alpha A)^{-1/\alpha}$ and $c_{ss} = Ak_{ss}^\alpha - k_{ss}$
2. Replace all variables x_t by $\exp(\ln x_t) \equiv \exp \hat{x}_t$
3. Take first-order Taylor expansions around steady state of x. Note that we can ignore all of the constant terms until the end.

   The first-order condition is the resource constraint which we can write

$$e^{\hat{c}_t} + e^{\hat{k}_{t+1}} = Ae^{\alpha\hat{k}_t}$$

The left hand side can be approximated as

$$e^{\hat{c}} + e^{\hat{c}}(\hat{c}_t - \hat{c}) + e^{\hat{k}} + e^{\hat{k}}(\hat{k}_{t+1} - \hat{k}) = e^{\hat{c}}\hat{c}_t + e^{\hat{k}}\hat{k}_{t+1} + \text{constant}.$$

where recall that $f(x_t) \approx f(x) + f'(x)(x_t - x)$. The right hand side is

$$Ae^{\alpha\hat{k}} + A\alpha e^{\alpha\hat{k}}(\hat{k}_t - \hat{k}) = A\alpha e^{\alpha\hat{k}}\hat{k}_t + \text{constant}$$

So this means we have (ignoring constants)

$$e^{\hat{c}}\hat{c}_t + e^{\hat{k}}\hat{k}_{t+1} = A\alpha e^{\alpha\hat{k}}\hat{k}_t.$$

Using steady states this implies

$$c_{ss}\hat{c}_t + k_{ss}\hat{k}_{t+1} = \alpha y_{ss}\hat{k}_t.$$

Multiply by β and divide by k_{ss}:

$$(1/\alpha - \beta)\hat{c}_t + \beta\hat{k}_{t+1} = \hat{k}_t$$

since $\beta c_{ss}/k_{ss} = \beta(y_{ss} - k_{ss})/k_{ss} = \beta(1/(\beta\alpha) - 1)$.

The second first order condition is the dynamic Euler equation which we can write:

$$e^{-\hat{c}_t} = \beta\alpha A e^{(\alpha-1)\hat{k}_{t+1}} e^{-\hat{c}_{t+1}}$$

The left hand side (ignoring constants) is $-e^{-\hat{c}}\hat{c}_t$. The right hand side (ignoring constants) is

$$\beta\alpha A e^{-\hat{c}} e^{(\alpha-1)\hat{k}}[-\hat{c}_{t+1} + (\alpha-1)\hat{k}_{t+1}].$$

Using steady states this implies

$$\begin{aligned}\hat{c}_t &= \beta\alpha A e^{(\alpha-1)\hat{k}}[\hat{c}_{t+1} + (1-\alpha)\hat{k}_{t+1}] \\ &= \hat{c}_{t+1} + (1-\alpha)\hat{k}_{t+1}.\end{aligned}$$

The log-linear solution is

$$\begin{aligned}\hat{c}_t &= \gamma_c + \psi_c\hat{k}_t \\ \hat{k}_{t+1} &= \gamma_k + \psi_k\hat{k}_t\end{aligned}$$

Plug this into the log-linearized solution. Again, for now, ignore the constant terms. We will figure out the constant terms at the end. Rewriting the first-order conditions, we have

$$\begin{aligned}(1/\alpha - \beta)\psi_c\hat{k}_t + \beta\psi_k\hat{k}_t &= \hat{k}_t \\ \psi_c\hat{k}_t &= \psi_c\psi_k\hat{k}_t + (1-\alpha)\psi_k\hat{k}_t\end{aligned}$$

For this to be satisfied, ψ_c and ψ_k have to satisfy:

$$\begin{aligned}(1/\alpha - \beta)\psi_c + \beta\psi_k &= 1 \\ \psi_c &= \psi_c\psi_k + (1-\alpha)\psi_k.\end{aligned}$$

We can substitute out $\psi_c = (1 - \beta\psi_k)/(1/\alpha - \beta)$ to get a quadratic in ψ_k:

$$\beta\psi_k^2 - (1/\alpha + \alpha\beta)\psi_k + 1 = 0.$$

There are two roots (which you should note are β-reciprocals): α and $1/(\alpha\beta)$. This is what Vaughan's theorem says you should find!

Take the root inside the unit circle for ψ_k, namely $\psi_k = \alpha$. Plugging that in to the expression for ψ_c gives $\psi_c = \alpha$ and

$$\hat{c}_t = \gamma_c + \alpha \hat{k}_t$$
$$\hat{k}_{t+1} = \gamma_k + \alpha \hat{k}_t.$$

To get the γ's, use the steady state values.

Can we do this on a computer? Yes, but in this case we are computing the coefficients (the γ's and ψ's). In all cases (even the more general examples), solving the problem on the computer means solving a quadratic equation. (Refer to computer codes in UM8185 and UM8186.)

Method 3. Transforming the problem to an LQ maximization

If we substitute the resource constraint into the utility function, then we have

$$\max_{\{k_{t+1}\}} \sum_{t=0}^{\infty} \beta^t u(Ak_t^\alpha - k_{t+1})$$

given k_0. A standard method for solving this involves a second-order approximation of $u(Ak_t^\alpha - k_{t+1})$ so that the problem can be written in a general LQ way:

$$\max_{\{u_t\}} \sum_{t=0}^{\infty} \beta^t \{X_t' Q X_t + u_t' R u_t + 2 X_t' W u_t\}$$

subject to

$$X_{t+1} = AX_t + Bu_t.$$

For this, we can apply standard methods developed by engineers (e.g., dynamic programming methods or methods like Vaughan's based on computing eigenvalues of a particular matrix).