Econ 200 (MATLAB Course) - Problems

Flavien Moreau*

September 14, 2016

Problems

Please try your best to solve the following problems. Problem 3 introduces you to the canonical method of value function iteration that is extremely useful to solve fixed-point problems.

Problem 1: Convergence

Find the solutions of $x^2 - x - 8 = 0$ in the following ways:

a) analytically

b) using a while loop, find the negative and positive solutions

c) using fsolve find the negative and positive solutions

Problem 2: Finite Horizon Life-Cycle with Two Goods (Riley Essential Microeconomics Chapter 6)

A consumer has T periods to live. There are two consumption goods. The consumer has consumption sequence: $\{x_t\}_{t=1}^T = \{x_{1t}, x_{2t}\}_{t=1}^T$.

We will assume that preferences have the following form: $u(x_1, x_2) = x_1^a + bx_2^b$ where $a \neq b$.

$U(x_1, ..., x_T) = \sum_{t=1}^{t=T} u_t(x_t)$ where $u_t(x_t) = \delta^{t-1}u(x_t)$, $u'() > 0$, $u''() < 0$, $u'(0) = \infty$ and $\delta \in [0, 1]$

We have known income stream: $\{w_t\}_{t=1}$, interest rate r, and initial capital k_1. Also, normalize $p_{1t} = 1$ and assume $p_{2t} = \bar{p_2}$ fixed.

And note that $k_{T+1} = 0$ without a bequest motive.

Objective: solve for $\{x_t\}_{t=1}^T$ and $\{k_t\}_{t=1}^T$ when the growth equation is known to be:

$k_{t+1} \leq (1 + r)(k_t + w_t - x_{1t} - p_{2t}x_{2t})$.

Question (a): Rewrite the budget constraint without capital (except for initial capital).

Question (b): Now, solve for $\text{MRS}(x_t, x_{t+1})$ for both good 1 and good 2 and find the $x_{2,t}$ as a function of p_2, paramters, and $x_{1,t}$.

Question (c): Suppose $W_t = k_t + PV(\{w_\tau\}_{\tau=t}^{\tau=T})$

Notice, this implies $W_1 = \sum_{t=1}^{T} \frac{x_{1,t} + p_2 x_{2,t}}{(1+r)^{t-1}}$ from part (a).

Prove $W_{t+1} = (1 + r)W_t - (1 + r)(x_{1,t} + \bar{p_2}x_{2,t})$.

Question (d): Numerical Implementation

$\delta = .98$, $1 + r = 1.05$, $\bar{W_1} = 100$, $\bar{p_2} = 1.5$, $a = .6$, $b = .4$

Following from $\bar{W_1}$, using the MRS condition and the law of motion for wealth, compute the sequence of wealth and consumption. See chapter 3 of essential microeconomics for further explanation. This should then give you the sequence of $\{W_t\}_{t=1}^T$. Use fsolve to converge on the condition W(T+1)=0. Plot your solutions for the vectors x_1, x_2, and W as a function of time.

*These notes heavily rely on the ones generously provided by Robert Kurtzman. All errors are mine.

Question (e)

Now suppose we only have one good and new utility function $ln(\xi + x)$ where $\xi = 2$ with all other parameters the same.

What is the MRS? Notice, we have the same wealth equation. Solve for x_1 numerically using the same parameters as part (d). Check your answer with page 12 of the Chapter 6 slides of Essential Microeconomics .

Problem 3: Value Function Iteration

Consider the following problem:

$$V(k) = \max_{k'}\{\log(Ak^{\alpha} + (1-\delta)k - k') + \beta V(k')\}$$

The value function V is defined recursively by a Bellman equation. You will come across this problem extensively in the first quarter of macro. But, let's just say for now that we have this problem we want to solve numerically. First we discretize the problem: instead of solving for the function V, an infinite dimension problem we turn to a finite-dimensional version of the problem by evaluationg V on a grid. Second, we use the fact that this Bellman equation defines a contraction mapping of which our solution is a fixed point. As a consequence, an easy way to converge to a solution is to iterate the contraction mapping.

First we want to define a grid of capital, k. On the one hand, capital will always be greater than 0. On the other hand, there is a natural upper bound for capital; we get this by deriving the level at which $Ak^{\alpha} + (1-\delta)k - k' = 0$. This is the level beyond which, even by investing all the available capital, the output produced along with net-of-depreciation capital fall short of the amount of capital invested. Solving this equation, we have the following:

$$k = Ak^{\alpha} + (1-\delta)k$$

$$\implies k_{end} = (\frac{\delta}{A})^{\frac{1}{\alpha - 1}}$$

To create a grid of length n, if we want the end points to be equally spaced, we can do the following: $k = [0 : \frac{k_{end}}{(n-1)} : k_{end}]$. Or, we could use the function linspace, which will do this for us: $linspace\,(0,\, k_{end}\,, n)$. Start with $n = 50$.

Since Matlab is very efficient at linear algebra, we rewrite the Bellman equation with matrices. We also take advantage of a built-in function of Matlab: when applied to a vector, max returns the maximum value of the vector, when applied to a matrix, it returns a row vector containing the maximum value of each column. So, let's consider for a second the matrix multiplication we want to perform. Notice, we are only choosing k', and in the discretized version of the problem, this amounts to chosing the optimal point on the capital grid we have defined above. You should convince yourself that the discretized version of the Bellman equation should look like the following:

$$\begin{pmatrix} V_1 \\ V_2 \\ . \\ . \\ . \\ V_n \end{pmatrix}^t = \max\{\log(A \begin{bmatrix} k_1^{\alpha} & k_2^{\alpha} & . & . & . & k_n^{\alpha} \\ k_1^{\alpha} & k_2^{\alpha} & . & . & . & k_n^{\alpha} \\ . & . & . & & & . \\ . & . & & . & & . \\ . & . & & & . & . \\ k_1^{\alpha} & k_2^{\alpha} & . & . & . & k_n^{\alpha} \end{bmatrix} + (1-\delta) \begin{bmatrix} k_1 & k_2 & . & . & . & k_n \\ k_1 & k_2 & . & . & . & k_n \\ . & . & . & & & . \\ . & . & & . & & . \\ . & . & & & . & . \\ k_1 & k_2 & . & . & . & k_n \end{bmatrix} - \begin{bmatrix} k_1' & k_1' & . & . & . & k_1' \\ k_2' & k_2' & . & . & . & k_2' \\ . & . & . & & & . \\ . & . & & . & & . \\ . & . & & & . & . \\ k_n' & k_n' & . & . & . & k_n' \end{bmatrix}) + \beta \begin{bmatrix} V_1 & V_1 & . & . & . & V_1 \\ V_2 & V_2 & . & . & . & V_2 \\ . & . & . & & & . \\ . & . & & . & & . \\ . & . & & & . & . \\ V_n & V_n & . & . & . & V_n \end{bmatrix}\}$$

In our algorithm, we can define a matrix T:

$$T = log(A \begin{bmatrix} k_1^\alpha & k_2^\alpha & . & . & . & k_n^\alpha \\ k_1^\alpha & k_2^\alpha & . & . & . & k_n^\alpha \\ . & . & . & & & . \\ . & . & & . & & . \\ . & . & & & . & . \\ k_1^\alpha & k_2^\alpha & . & . & . & k_n^\alpha \end{bmatrix} + +(1-\delta) \begin{bmatrix} k_1 & k_2 & . & . & . & k_n \\ k_1 & k_2 & . & . & . & k_n \\ . & . & . & & & . \\ . & . & & . & & . \\ . & . & & & . & . \\ k_1 & k_2 & . & . & . & k_n \end{bmatrix} - \begin{bmatrix} k_1' & k_1' & . & . & . & k_1' \\ k_2' & k_2' & . & . & . & k_2' \\ . & . & . & & & . \\ . & . & & . & & . \\ . & . & & & . & . \\ k_n' & k_n' & . & . & . & k_n' \end{bmatrix})$$

We can use matrix algebra to create these matrices:

$T = log(max(ones(n,1) * (A. * k.^\alpha + (1-\delta). * k) - k' * ones(1,n), 10^{-1000}))$;[1]

Check the help to see how you can use the command *repmat* to define the last matrix of the right hand side of the equation.

As you will see in a few weeks, the Bellman equation is actually a contraction mapping. This implies that we can start from any initial value for V, and keep on iterating the mapping until it is close enough.

Set parameters at the following values: $\alpha = \frac{1}{3}$; $\beta = .96$; $A = 1$; $\delta = .08$.

Solve for V and the optimal policy rule for k' to a convergence criterion of 10^{-10}.

[1] note that the max operator here has nothing to do with the one evoked above but we just want to avoid Matlab setting some values to $-\infty$ when the number inside the log is 0.