Quantitative Macro Problem Set 3

Bingxue Li, Zhihang Liang, Tuba Seker

PSE-APE, Fall 2023

Consider the following income fluctuation problem for $t = 0, 1, \ldots, T$.

$$\max_{\{c_t, k_{t+1}, i_t\}_{t=1}^T} E_0 \sum_{t=0}^{T} \beta^t \frac{c_t^{1-\sigma} - 1}{1-\sigma}$$
$$\text{s.t.}$$
$$c_t + i_t \leq z_t k_t^\theta$$
$$k_{t+1} = (1-\delta) k_t + i_t$$
$$\log(z_{t+1}) = \rho \log(z_t) + \varepsilon_{t+1} \text{ where } \varepsilon_{t+1} \sim \text{ i.i.d. } \mathcal{N}\left(0, \sigma^2\right)$$
$$k_0 \text{ and } z_0 \text{ given}$$

where c_t is consumption, k_t capital, z_t productivity, i_t investment and ε_t an exogenous shock. In this problem set we consider the case without shocks, so assume $z_0 = 1$ and $\epsilon_t = 0, t = 0, 1, 2, \ldots$.

1 Problem 1 The Bellman equation and its properties.

1. Simplify the problem as a choice of $\{k_{t+1}\}$. Define the "givens" $(X, \Gamma, F,\ \beta)$ seen in class for this problem.

We already know that for optimality, we need to use all production, ie. the resource constraint should be binding:

$$c_t + i_t = k_t^\theta \tag{1}$$

Then, we substitute for i_t and get,

$$c_t = k_t^\theta + (1-\delta) k_t - k_{t+1} \tag{2}$$

Now, we can plug 2 into the utility function to simplify the problem as a choice of k_{t+1}. Note that we do not need to express E_0 and z_t for now since we assume no shocks:

$$\max_{\{k_{t+1}\}_{t=0}^T} \sum_{t=0}^{T} \beta^t \frac{[k_t^\theta + (1-\delta) k_t - k_{t+1}]^{1-\sigma} - 1}{1-\sigma} \tag{3}$$

Now, let's define the givens of this problem:

1. **X**: set of possible values of k_{t+1} where k_{t+1} ranges from 0 (full consumption of capital stock) to k^{max}.

2. Γ: This is feasibility correspondence which describes the set of feasible values of next period capital stock which ranges from 0 to k_{max} (where 0 consumption of the capital stock):

$$\Gamma(k) = \{k_{t+1} : 0 \leq k_{t+1} \leq (1-\delta) k_t + k_t^\theta\}$$

3. **F**: return function or one-period function which maps the set of current and next period capital stock into real numbers. It corresponds to the utility function at period t:

$$F(k_t, k_{t+1}) = \frac{[k_t^\theta + (1-\delta)k_t - k_{t+1}]^{1-\sigma} - 1}{1-\sigma}$$

4. β: discount factor

2. Write down the functional equation (FE) for this problem that defines the value function v.

The value function is the maximization of the utility today and the discounted lifetime utility of the next period and onwards. We can write the functional equation as

$$\begin{aligned} v(k_t) &= \max_{k_{t+1}\in\Gamma}[F(k_t, k_{t+1}) + \beta v(k_{t+1})] \\ &= \max_{k_{t+1}\in\Gamma}\Big[\frac{[k_t^\theta + (1-\delta)k_t - k_{t+1}]^{1-\sigma} - 1}{1-\sigma} + \beta v(k_{t+1})\Big] \end{aligned} \tag{4}$$

3. The return function F is not bounded, but (FE) nevertheless maps $C(X)$, the space of continuous bounded functions on $\mathbb{R}$, into itself. (Make sure you understand why). (FE) defines a contraction mapping using Blackwell's sufficient conditions. (Again, make sure you understand why). Can you further characterise v given the functional forms of F and Γ ?

Remember that with the given utility and production functions, we can write return function 4, which is unbounded. On the other hand, state space **X** ranges in between 0 to k^{max}, ie. compact set. In addition, Γ is compact and continuous while F is also a continuous function. Thus, the return function F will behave as if it is bounded.

Intuitively, to see why F is bounded above, assume that the planner temporarily consumes nothing and accumulates capital fully. After a while, capital stock will converge to k^{max} such that

$$\begin{aligned} k_t = k_t^\theta + (1-\delta)k_t - c_t &\Leftrightarrow k_{max} = k_{max}^\theta + (1-\delta)k_{max} \\ k_{max} &= \delta^{\frac{1}{\theta-1}} \end{aligned} \tag{5}$$

We know that the return function F reaches its maximum in $F(k^{max}, 0)$ since it's monotonic with k, ie. increasing with k. Thus, F is bounded above, given the constraints.

On the other hand, F is not bounded from below since 0 consumption and full capital accumulation will return $F \to \infty$, which is not optimal. Thus, we can ignore such a possibility. So, although the return function is unbounded, we can treat it as bounded.

Now, let's understand why (FE) nevertheless maps C(X) into C(X). Consider a mapping operator T as in the lecture notes such that

$$(Tv)(x) = \max_{y\in\Gamma(x)}[F(x,y) + \beta v(y)] \tag{6}$$

Our aim is to show $T : C(X) \to C(X)$, ie. operator T maps the space of continuous bounded functions on $\mathbb{R}$ into itself. We already discussed that F and v are bounded from above. Also, they both are continuous. Also, Γ is compact-valued and continuous. Thus, by the Theorem of Maximum, we can say that T is also continuous. All these show that the (FE) maps continuous bounded functions on X into continuous bounded functions.

Now, we show (FE) defines the contraction mapping using Blackwell's two sufficient conditions by using our problem in 6: monotonicity and discounting.

- **Monotonicity**: Assume $v(k_t) \leq w(k_t)$ for $\forall t$ implies $(Tv)(k_t) \leq (Tw)(k_t)$ where k^*_{t+1} defines optimal level of capital:

$$\begin{aligned}(Tv)(k_t) &= \max_{k_{t+1}\in\Gamma(k_t)}[F(k_t,k_{t+1})+\beta v(k_{t+1})] = F(k_t,k^*_{t+1})+\beta v(k^*_{t+1})\\ &\leq \max_{k_{t+1}\in\Gamma(k_t)}[F(k_t,k_{t+1})+\beta w(k_{t+1})]\\ &\leq F(k_t,k^*_{t+1})+\beta w(k^*_{t+1}) = (Tw)(k_t)\end{aligned} \tag{7}$$

- **Discounting**: There exists some $\beta \in (0,1)$ and $a \geq 0$,

$$\begin{aligned}T(v+a)(k_t) &= \max_{k_{t+1}\in\Gamma(k_t)}[F(k_t,k_{t+1})+\beta(v(k_{t+1})+a)]\\ &= \max_{k_{t+1}\in\Gamma(k_t)}[F(k_t,k_{t+1})+\beta v(k_{t+1})]+\beta a\\ &= (Tv)(k_t)+\beta a\end{aligned} \tag{8}$$

Now, let's characterize v given the functional forms of F and Γ. We showed that v is continuous and bounded. In addition, v is a concave function. Define,

$$x_i = \alpha x_0 + (1-\alpha)x_1 \tag{9}$$

where $x_0 \neq x_1$ and $i \in (0,1)$.
Let $y_i \in \Gamma(x_i)$ is complete $(Tv)(xi)$ for i=0,1. Since the feasibility correspondence Γ is convex, we have $y_i = \alpha y_0 + (1-\alpha)y_1$ is also $y_i \in \Gamma(x_i)$. If we assume F is strictly concave, then Tv is also concave,

$$\begin{aligned}Tv(x_i) &\geq F(x_i,y_i)+\beta v(y_i)\\ &> \alpha(F(x_0,y_0)+\beta v(y_0)+(1-\alpha)(F(x_1,y_1)+\beta v(y_1)))\end{aligned}$$

Since F is bounded and continuous, X is restricted to a compact set, and Γ is nonempty, compact valued and continuous. T has a unique fixed point $v \in C(X)$. Thus, v is also concave.
Also, v is increasing. F is increasing in its first element; for $k_t < k_t^{'}$, $\Gamma(k_t) \subset \Gamma(k_t^{'})$, as a higher level of capital can sustain a higher level of capital in the next period. From the corollary of the contraction mapping theorem, v is increasing.

4. Assume that $\delta = 1$ and $\sigma = 1$ (s.t. the utility function is $\log(c_t)$).

(a) Consider the finite-horizon problem, where T denotes the final period. Solve for the optimal path of the endogenous variables for $T = 2$ by backward induction.

For the finite horizon problem, we know that when t=T, $k_{T+1} = 0$, ie. eat all of your capital stock in the last period. Then, recall that $c_t = k_t^\theta + (1-\delta)k_t - k_{t+1}$ corresponds here to $c_T = k_T^\theta$. Then, we solve for

$$\begin{aligned}V_0(k_2) &= \max_{k_3\in[0,f(k_2)]} In(k_2^\theta - k_3)+\beta 0\\ &= \theta In k_2\end{aligned} \tag{10}$$

When t=T-1, the problem becomes $\max_{k_2} In(k_1^\theta - k_2)+\beta V_2(k_2)$. First-order conditions give,

$$-\frac{1}{k_1^\theta - k_2}+\frac{\beta\theta}{k_2} = 0 \Leftrightarrow k_2 = k_1^\theta\frac{\beta\theta}{1+\beta\theta} \tag{11}$$

and by using $c_t = k_t^\theta - k_{t+1}$ where $\delta = 1$, we get c_1 as

$$c_1 = \frac{k_1^\theta}{1+\beta\theta} \tag{12}$$

Now, you need to solve,

$$\begin{aligned} V_1(k_1) &= \max_{k_2 \in [0, f(k_1)]} In(k_1^\theta - k_2) + \beta V_0(k_2^\theta - k_3) \\ &= In(k_1^\theta - k_1^\theta \frac{\beta\theta}{1+\beta\theta}) + \beta\theta In(k_1^\theta \frac{\beta\theta}{1+\beta\theta}) \\ &= Ink_1^\theta + In\frac{1}{1+\beta\theta} + \beta\theta Ink_1^\theta + \beta\theta In\frac{\beta\theta}{1+\beta\theta} \\ &= \theta(1+\beta\theta)Ink_1 + In\frac{1}{1+\beta\theta} + \beta\theta In\frac{\beta\theta}{1+\beta\theta} \end{aligned} \tag{13}$$

When, t=T-2, we solve

$$V_2(k_0) = \max_{k_1 \in [0, f(k_0)]} In(k_0^\theta - k_1) + \beta V_1(k_1) \Leftrightarrow$$

$$V_2(k_0) = \max_{k_1 \in [0, f(k_0)]} In(k_0^\theta - k_1) + \beta\Big[\theta(1+\beta\theta)Ink_1 + In\frac{1}{1+\beta\theta} + \beta\theta In\frac{\beta\theta}{1+\beta\theta}\Big] \tag{14}$$

By taking FOCs with respect to k_1, we get

$$k_1 = \frac{\beta\theta(1+\beta\theta)k_0^\theta}{1+\beta\theta+(\beta\theta)^2} \tag{15}$$

and by $c_0 = k_0^\theta - k_1$, we get our policy function as

$$c_0 = \frac{k_0^\theta}{1+\beta\theta+(\beta\theta)^2} \tag{16}$$

Then, we plug our optimal policy function into the value function and get

$$\begin{aligned} V_2(k_0) = In\Big(\frac{k_0^\theta}{1+\beta\theta+(\beta\theta)^2}\Big) &+ \beta\theta(1+\beta\theta)In\Big(\frac{\beta\theta(1+\beta\theta)k_0^\theta}{1+\beta\theta+(\beta\theta)^2}\Big) + \beta In(\frac{1}{1+\beta\theta}) + \beta^2\theta In(\frac{\beta\theta}{1+\beta\theta}) \\ = \theta(1+\beta\theta+(\beta\theta)^2)Ink_0 &+ In(\frac{1}{1+\beta\theta+(\beta\theta)^2}) + \beta\theta(1+\beta\theta)In(\frac{\beta\theta+(\beta\theta)^2}{1+\beta\theta+(\beta\theta)^2}) \\ + \beta In(\frac{1}{1+\beta\theta}) &+ \beta^2\theta In(\frac{\beta\theta}{1+\beta\theta}) \end{aligned} \tag{17}$$

(b) Infer the optimal policy rule for the finite-horizon problem from this (or do one more iteration to $T = 3$).

As seen in 12 and 40, we can generalize our optimal policy rule for the finite-horizon problem as

$$c_{T-t} = \frac{k_{T-t}^\theta}{\sum_{n=0}^t (\beta\theta)^n} \tag{18}$$

$$k_{T-t+1} = \frac{\sum_{n=0}^t (\beta\theta)^n - 1}{\sum_{n=0}^t (\beta\theta)^n} k_{T-t}^\theta \tag{19}$$

In addition, we show the derivation for optimal capital stock k_1, consumption c_0, and V_0 in a T=3 case in the appendix.

(c) Infer the value function for the infinite horizon problem from $V_2(k_0)$. (Hint: From the value functions V_0, V_1 and V_2, you can guess the pattern of the finite horizon value functions. Then take the limit as $T \to \infty$ to obtain the infinite horizon value function.) Solve for the optimal stationary policy function that attains the maximum in (FE).

We denote the value function at the first period for a T-period living problem by $V^{(T)}(k)$, By mathematical induction, we can easily prove that:

$$V^{(T)}(k) = \theta \sum_{i=0}^{T} (\beta\theta)^i Ink + \sum_{t=1}^{T} \beta^{T-t} a_t \tag{20}$$

where

$$a_t = In(\frac{1}{\sum_{i=0}^{t}(\beta\theta)^i}) + \beta\theta(\sum_{i=0}^{t-1}(\beta\theta)^i)In(\frac{\sum_{i=0}^{t}(\beta\theta)^i - 1}{\sum_{i=0}^{t}(\beta\theta)^i})$$

Simplify a_t, we get:

$$a_t = In(\frac{1-\beta\theta}{1-(\beta\theta)^{t+1}}) + \beta\theta(\frac{1-(\beta\theta)^t}{1-\beta\theta})In(1 - \frac{1-\beta\theta}{1-(\beta\theta)^{t+1}}) \tag{21}$$

We consider the term $\beta^t a_{T-t}$, we let T tends to infinity and fix t to see what happens:

$$\lim_{T\to\infty} \beta^t a_{T-t} = \beta^t \left(In(1-\beta\theta) + \beta\theta(\frac{1}{1-\beta\theta})In(\beta\theta) \right) \equiv \beta^t \bar{a}$$

Next, we calculate the second term in equation 20:

$$\sum_{t=1}^{T} \beta^{T-t} a_t$$
$$= \sum_{j=0}^{T-1} \beta^j a_{T-j}$$

We use the $\epsilon - \delta$ language to prove that:

$$\lim_{T\to\infty} \sum_{j=0}^{T-1} \beta^j a_{T-j} = \lim_{T\to\infty} \sum_{j=0}^{T-1} \beta^j \bar{a} = \frac{\bar{a}}{1-\beta} \tag{22}$$

$\forall \epsilon > 0, \forall\, T_0 > 0, \exists\, \tilde{T}(T_0)$, s.t. when $T > \tilde{T}(T_0)$:

$$|\sum_{j=1}^{T_0} \beta^j (a_{T-j} - \bar{a})| < \epsilon, \tag{23}$$

which is an immediate result of $\lim_{T\to\infty} a_T = \bar{a}$.
At the same time, $\forall \epsilon < 0, \exists\, \tilde{T_0}(\epsilon)$, s.t. when $T_0 > \tilde{T_0}$, $\forall\, T$, we have:

$$|\sum_{j=T_0+1}^{T} \beta^j (a_{T-j} - \bar{a})| < \epsilon \tag{24}$$

To prove the above inequality, we need to see that the series $\{|a_t - \bar{a}|\}$ is bounded above by some positive numer M , as $\lim_{t\to\infty} a_t = \bar{a}$. Thus:

$$|\sum_{j=T_0+1}^{T} \beta^j (a_{T-j} - \bar{a})| < \sum_{j=T_0+1}^{T} \beta^j M = \beta^{T_0+1} \frac{1-\beta^{T-T_0}}{1-\beta} M < \beta^{T_0+1} \frac{1}{1-\beta} M$$

So, as long as T_0 is large enough, no matter what value T takes, $|\sum_{j=T_0+1}^{T}\beta^j(a_{T-j}-\bar{a})|$ can be as small as possible.
Combining inequality 23 and 24, we can see that $\forall\epsilon > 0$, $\exists\ \tilde{T_0}(\epsilon)$, $\exists\ \tilde{T}(\tilde{T_0}(\epsilon))$, s.t. when $T > \tilde{T}(\tilde{T_0}(\epsilon))$:

$$|\sum_{j=1}^{T_0}\beta^j(a_{T-j}-\bar{a})| < \epsilon,\ |\sum_{j=T_0+1}^{T}\beta^j(a_{T-j}-\bar{a})| < \epsilon \tag{25}$$

This implies that:

$$\lim_{T\to\infty}\sum_{j=0}^{T-1}\beta^j a_{T-j} = \lim_{T\to\infty}\sum_{j=0}^{T-1}\beta^j\bar{a} = \frac{\bar{a}}{1-\beta}.$$

Thus, we have:

$$\lim_{T\to\infty}V^{(T)}(k) = \theta\frac{1}{1-\beta\theta}In(k) + \frac{log(1-\theta\beta)}{1-\beta} + \frac{1}{1-\beta}\frac{\beta\theta}{1-\beta\theta}log(\beta\theta) \tag{26}$$

(d) Alternatively, you can find the value function with the method of "guess and verify" with $V(k) = \alpha_0 + \alpha_k\log(k))$ as an initial guess and determining the unknown coefficients θ_0, θ_k in terms of the coefficients of the model.

Given the value function, we can write the (FE) as follows (where k' denotes next period's capital stock for simplicity),

$$\alpha_0 + \alpha_k\log(k) = \max_{k'\in\Gamma(k)} log(k^\theta - k') + \beta(\alpha_0 + \alpha_k\log(k')) \tag{27}$$

FOCs with respect to k' gives,

$$-\frac{1}{k^\theta - k'} + \frac{\beta\alpha_k}{k'} = 0 \Leftrightarrow k' = \frac{\beta\alpha_k k^\theta}{1+\beta\alpha_k} \tag{28}$$

Plug this into the value function and then simplify,

$$\begin{aligned}\alpha_0 + \alpha_k\log(k) &= log(k^\theta - \frac{\beta\alpha_k k^\theta}{1+\beta\alpha_k}) + \beta(\alpha_0 + \alpha_k\log(\frac{\beta\alpha_k k^\theta}{1+\beta\alpha_k}))\\ &= \theta logk + log(\frac{1}{1+\beta\alpha_k}) + \beta\alpha_0 + \beta\alpha_k log(\frac{\beta\alpha_k}{1+\beta\alpha_k}) + \beta\alpha_k\theta logk\\ &= \theta(1+\beta\alpha_k)logk + log(\frac{1}{1+\beta\alpha_k}) + \beta\alpha_0 + \beta\alpha_k log(\frac{\beta\alpha_k}{1+\beta\alpha_k})\end{aligned} \tag{29}$$

Since 29 holds for $\forall t$,

$$\alpha_k = \theta(1+\beta\alpha_k) \tag{30}$$

$$\alpha_k = \frac{\theta}{1-\theta\beta} \tag{31}$$

The rest also holds as follows,

$$\alpha_0 = log(\frac{1}{1+\beta\alpha_k}) + \beta\alpha_0 + \beta\alpha_k log(\frac{\beta\alpha_k}{1+\beta\alpha_k}) \tag{32}$$

By plugging 30 and 31 into 32, we simplify and get a_0 as,

$$\alpha_0 = \frac{log(1-\theta\beta)}{1-\beta} + \frac{1}{1-\beta}\frac{\beta\theta}{1-\beta\theta}log(\beta\theta) \tag{33}$$

Moreover, we can get the optimal policy function:

$$k' = \beta\theta k^\theta \tag{34}$$

2 Problem 2 Numerical solution I: Discrete-grid value function iteration.

Consider $T = \infty$ from now on and the following parameter values:

β	θ	σ	δ
0.99	0.4	2	0.1

Calculate analytically the steady state level of capital k^* of this economy and calculate its value given parameters.

Let's rewrite (FE) in 4 as:

$$v(k_t) = \max_{k_{t+1}\in\Gamma} \Big[\frac{[k_t^\theta + (1-\delta)k_t - k_{t+1}]^{1-\sigma} - 1}{1-\sigma} + \beta\Big[\frac{[k_{t+1}^\theta + (1-\delta)k_{t+1} - k_{t+2}]^{1-\sigma} - 1}{1-\sigma}\Big]$$

Take the FOCs with respect to k_{t+1},

$$(k_t^\theta + (1-\delta)k_t - k_{t+1})^{-\sigma} = \beta(k_{t+1}^\theta + (1-\delta)k_{t+1} - k_{t+2})^{-\sigma}(\theta k_{t+1}^{\theta-1} + 1 - \delta)$$

which results in the steady state,

$$k^* = \Big[\frac{\theta\beta}{1-\beta(1-\delta)}\Big]^{\frac{1}{1-\theta}} = 8.5858 \tag{35}$$

(a) Rewrite the recursive continuous dynamic programming from Problem 1.1. as a discrete one (by constraining the state space to equal a discrete grid.)

To approximate the value function, we need to specify a domain of interval around the steady state which will reflect the grid points, k_1 to k_N where $k_1 = 3/4k^*$ and $k_N = 5/4k^*$. This interval of grid points is equally spaced and we will have an approximation of the value function v at each point of the grids. Here, we define the grid in a linear manner such that the grid points are equally spaced. To improve the accuracy of the approximation, one can set the grid to be a nonlinear one such that adding more grid points where the curvature of the policy function is high.

$$K = \{k_1 < k_2 < ... < k_N\} \tag{36}$$

Now, let's rewrite our value function as

$$v(k_t) = \max_{k_{t+1}\in\Gamma(k)\cap K, k_i\in K} \Big[\frac{[k_t^\theta + (1-\delta)k_t - k_{t+1}]^{1-\sigma} - 1}{1-\sigma} + \beta v(k_{t+1})\Big]$$

(b) Now solve the above problem by "discrete" value function iterations using matlab or some other program. In particular,
i. Choose a criterion ε for convergence of the value function (choose e.g. 10^{-6}).
ii. Choose an equally-spaced grid $\mathbb{K} = \{k_1 < k_2 < \dots k_N\}$ with $k_1 = 3/4k^*$ and $k_N = 5/4k^*$, where k^* denotes the steady-state value of the capital stock.
iii. For $i, l = 1, \dots, N$, calculate the one-period return function $F(k_i, k_l')$.
iv. Choose an initial value function $v_0(k_i)$, an $N \times 1$ vector.
v. Calculate for all $i = 1, .., N$

$$v_{s+1}[i] = \max_l \{F(k_i, k_l) + \beta v_s[l]\}$$

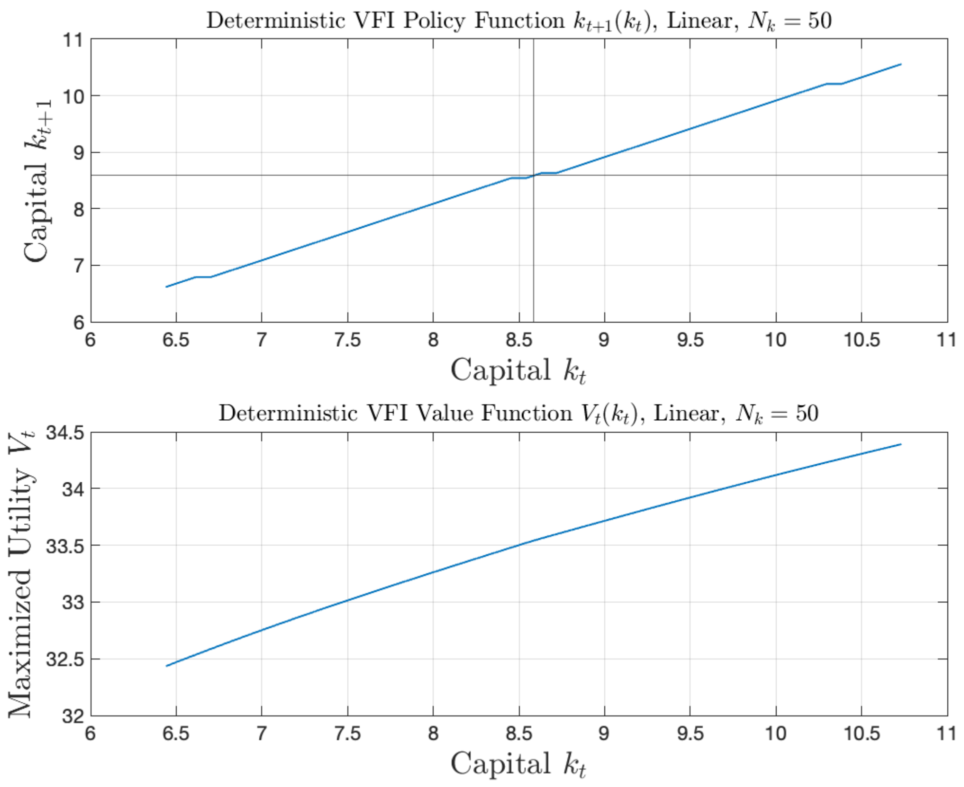

Figure 1: Deterministic VFI Solution ($N_k = 50$, Linear Grid, Iter=1494)

for $s = 0, 1, 2, \ldots$**, where** $X[j]$ **denotes the** j$'$ **the element of** X**.**
vi. If $\max_{i,j} |v_{s,i,j} - v_{s-1,i,j}| \geq \varepsilon$ **go back to b). Otherwise stop.**

See the attached code file.

(c) Plot the policy function $k'(k)$**.**

See 1 for $N_k = 50$, $V_0 = \mathbf{0}$,(Number of iteration= 1494).
To improve convergence speed, we try with an educated initial guess, the linear consumption rule where we set V_0 such that $V_0(k_i) = \sum_{t=0}^{\infty} \beta^t u(k_i^\theta - \delta k_i) = \frac{k_i^\theta - \delta k_i}{1-\beta}$ for $i = 1, 2, ..., N$. This initial guess allows the value function to converge with only 23 iterations. Although a good initial guess improves the convergence speed of the guesses, the trade-off between efficiency and accuracy persists.

To improve the accuracy of fitting linear approximation towards function with curvature although with a loss of efficiency, we try with a larger number of grids. See 2 for $N_k = 1000$, (Number of iteration= 66). Obviously, with more narrowly discretized grids, the linear approximations for a function with curvature become smoother and better. Our numerical solution gives a better approximation for the true analytical one now.

In addition, to check the robustness of our method, we try with $\delta = 1$ and $\sigma = 1$ where we have a corner solution with an analytical form of the policy function such that $k_{t+1} = \beta\theta k_t^\theta$. See 3 with $N_k = 200$ where the red plot denotes the analytical solution, and one can see that they resemble in terms of general trend.

Finally, with the curvature of the policy function observed, we try a nonlinear grid design with more weight for a higher degree of curvature. In the case of $\delta = 1$ and $\sigma = 1$, the smaller k_t is the more concave the policy function will be according to the analytical solution $k_{t+1} = \beta\theta k_t^\theta$. In the code, this is achieved by first raising the values of grid distances to the power of 5 and

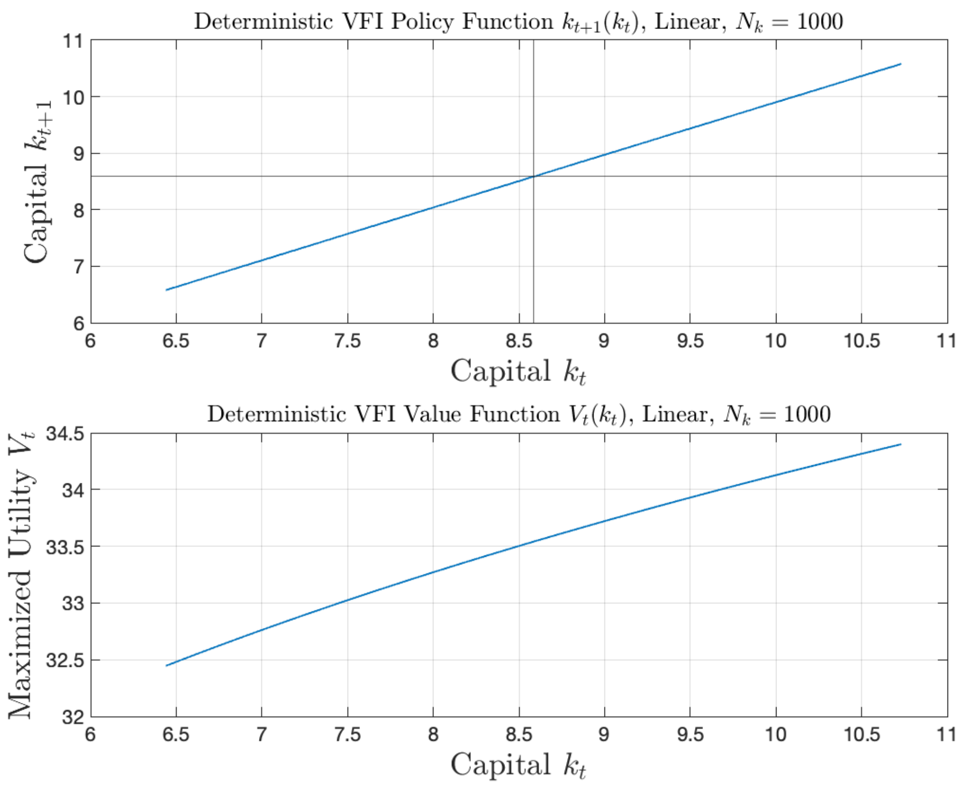

Figure 2: Deterministic VFI Solution ($N_k = 1000$, Linear Grid)

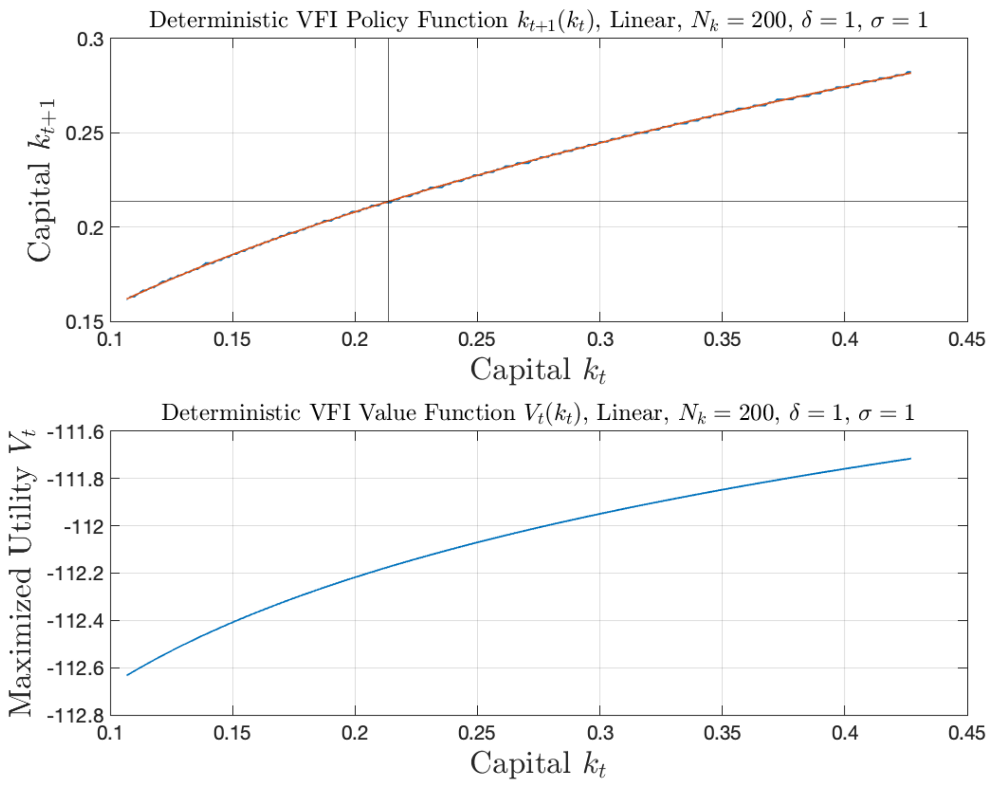

Figure 3: Deterministic VFI Solution ($N_k = 200$, Linear Grid, $\delta = 1, \sigma = 1$)

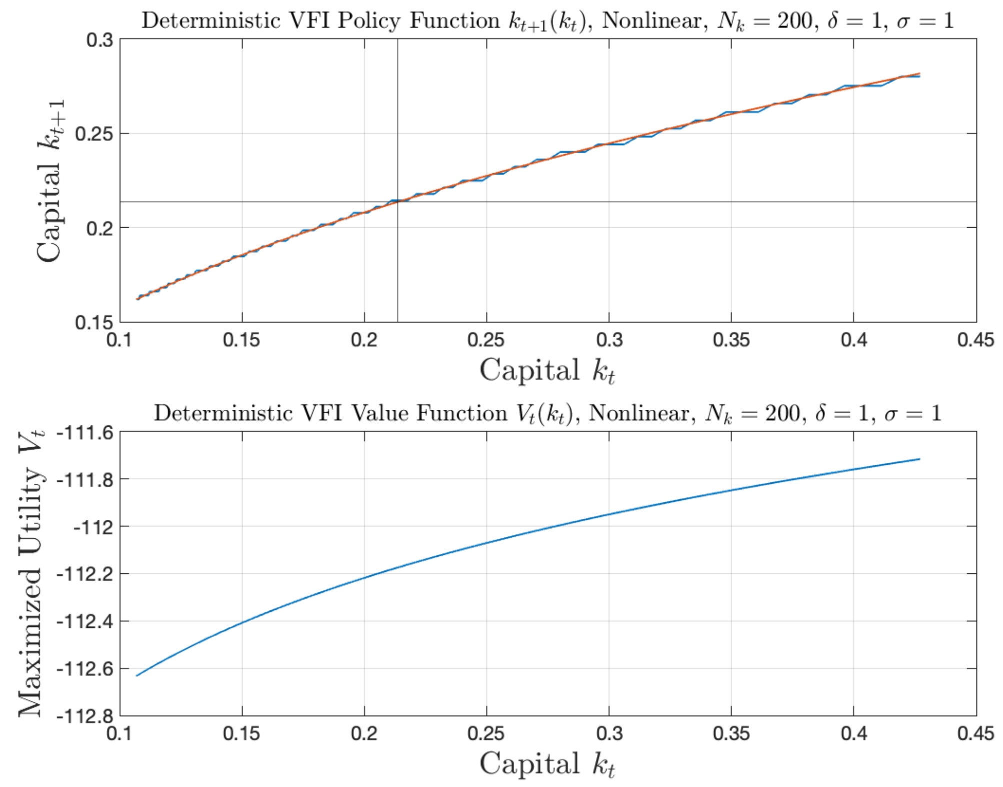

Figure 4: Deterministic VFI Solution ($N_k = 200$, Nonlinear Grid, $\delta = 1, \sigma = 1$)

normalizing it back to fit the range of k_t. See 4. Obviously, compared with 3, the approximation around smaller k_t is done with more grids (kinks) here. It appears that the improvement achieved with this method is rather modest when contrasted with the scenario of using equally spaced grids. Note that the value function now takes negative values, this is less of a concern as an outcome of parametrization of $\delta = 1$ (which drives k^* to a small level) and log-utility.

(d) Evaluate the Euler equation errors. In particular, for every value $k_i, i = 1, .., N$, calculate the maximum percentage difference between the consumption according to your policy function $c(k_i)$, and that which makes the Euler equation hold with equality given tomorrow's consumption and marginal productivity

$$c(k_i)^{imp} = \left[\beta\left(\theta * k'(k_i)^{\theta-1} + 1 - \delta\right) c(k'(k_i))^{-\sigma}\right]^{-1/\sigma}$$

See 5 for the plot of Euler Equation Error with respect to the level of k_t.
The Euler equation error provides a unit-free measure of the error in the first-order equation in terms of percentage and is a measure of the accuracy of the VFI method. The downs and spikes in 5 for specific k_t correspond to the non-smooth part in 1. Note that two strong deviations are close at the left and right-hand side of steady-state level $k^* = 8.5858$, when $k_t + 1$ infinitely close to k_t. To explain these sudden increases in the inaccuracy of VFI results, we checked the concavity of the policy function and found sudden changes in the concavity level around these points. We also checked with the case where $\delta = 1, \sigma = 1$, which gives a policy function with a uniform level of concavity across the level of k_t. Euler equation errors behave more consistently in this case, substantiating that spikes and downs here are the results of applying several linear approximations to a function with sudden changes in concavity.

(e) Bonus: Add a policy function iteration step, i.e. after k initial iterations on the Bellman equation, alternate between i) one iteration step on the Bellman equation

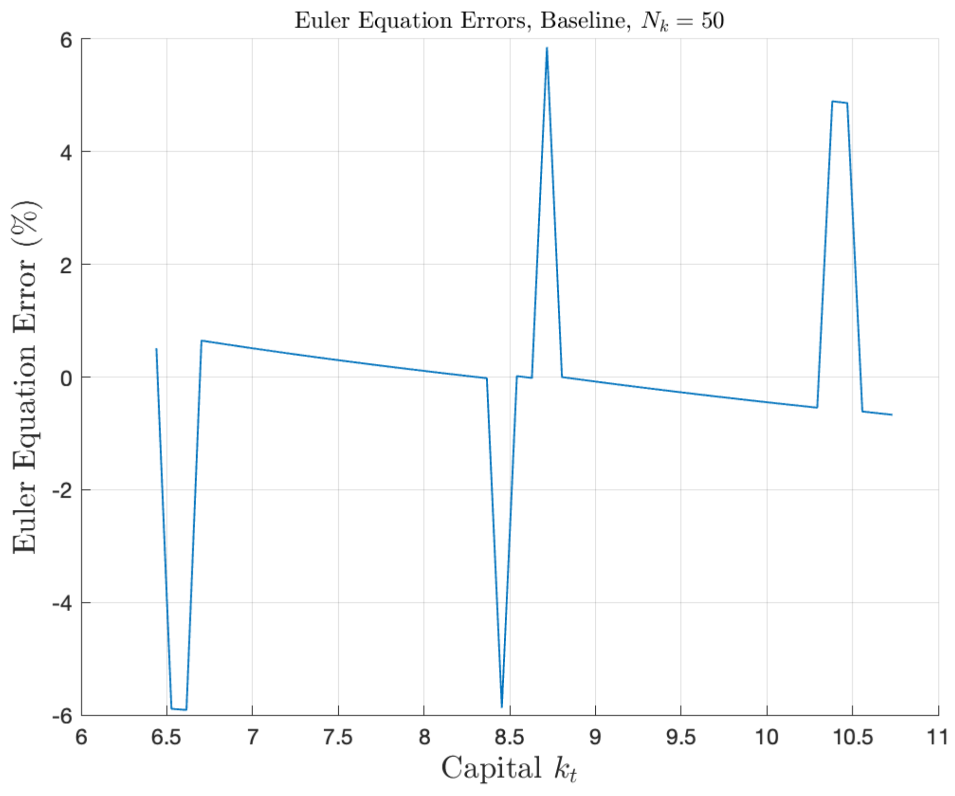

Figure 5: Deterministic VFI Euler Equation Errors ($N_k = 200$, Baseline, $N_k = 50$)

ii) iteration using the resulting policy (without changing it) until the value function converges. Go back to i) until the value function approximately solves the Bellman Equation. Does this speed up the solution? (You can experiment with k.)

The idea of iteration using the achieved policy function to let the value function converge comes from Howard's improvement. In fact, every time we iterate over a chosen i, we need to conduct the maximization of the Bellman equation by the dimension of j, which implies that we need to evaluate the Bellman N^2 times in each iteration under the while-loop. Doing maximization is computationally demanding, so we update the value function under each iteration instead of doing more policy function updating.

The steps for Howard improvement following each computation of the policy and value function vector (denote this as iteration h for the outer while-loop) is: i. Choose a criterion ε_v for convergence of the value function (choose e.g. 10^{-6}). ii. (Starting from the 3rd iteration of the outer while-loop where approximation of the value and policy function becomes more accurate) For $i = 1 : N$, let $V_i^{new} = U(k_i^\theta + (1-\delta)k_i - k_{m_i}) + \beta * V_{m_i}^*$. Note that k_{m_i} and $V_{m_i}^*$ are the approximation obtained under iteration h; iii. Repeat i and ii. until $\|V^{new} - V^*\| < \varepsilon_v$. As times of iteration following i and ii grow, the approximation of value function converges to the fixed point of $V_i = U(k_i^\theta + (1-\delta)k_i - k_{m_i}) + \beta * V_{m_i}$.

We count the number of iterations for the convergence of policy function under the outer while-loop with and without the element of Howard improvement. As mentioned with 1, the baseline VFI has 1494 iterations before convergence, and with Howard Improvement that starts from the 25th iteration, the total iteration number is reduced to 46. However, it takes more time for the value function to converge overall. One explanation is that the initial approximation V_h^* from iteration h of the outer while loop is not good enough and too far away from the fixed point that we are looking for, so the Howard Improvement loop itself takes a lot of time.

3 Problem 3 Numerical solution II: Value function iteration with interpolation.

(a) Now reformulate the maximisation step by allowing choices of k' that are not on the grid. Use linear interpolation (Matlab function interp1.m) to evaluate the Value function between grid points. Use the golden-search algorithm to find the optimal choice k' for each level of k. To find the bracket more efficiently, you can exploit that $k'(k)$ is an increasing function, such that $k^{i\star}(k_{i+j}) > k^{i\star}(k_i)$.

- Step 1, choose an initial guess of the value function.

Same as before, here we choose an initial value function. However, here, instead of choosing a vector of values, we choose an initial function in the matlab. Specifically, we set `V = @(x)(0)`. Also, we can choose $V(k) = \sum_{t=0}^{\infty} \beta^t u(k^\theta - \delta k) = \frac{k^\theta - \delta k}{1-\beta}$ to improve convergence speed as before.

- Step 2, update the value function by maximizing the RHS of the functional equation

We maximize the following equation for each state k^i:

$$V^{j+1}(k^i) = \max_{k'} \frac{[(k^i)^\theta + (1-\delta)k^i - k']^{1-\sigma} - 1}{1-\sigma} + \beta V^j(k') \tag{37}$$

Here, $V^j(\cdot)$ is the value function obtained by linear interpolation from the j-th iteration.
We use the method of golden-algorithm to do this, which requires monotonicity. Since the true value function is monotonic, monotonicity is likely to preserve while updating the value function. Basic steps are:

```
Fix the upper bound and lower bound for kprime.
Use the budget constraint to counstrct the bound.

Calculate the golden ratio: alpha1 = (3-sqrt(5))/2; alpha2 = (sqrt(5)-1)/2;

Calculate the searching point:
b1= kprimelow+alpha1*(kprimehigh-kprimelow);
b2= kprimelow+alpha2*(kprimehigh-kprimelow);

Evaluate the value function at b1, b2, kprimelow and kprimehigh:

dk=1;
criter_k=1e-12;

while dk>criter_k
% use golden search
    if Vb2>Vb1
        Update the lower bound of the searching range with b1
        Update b1 with b2
        Update b2 with kprimelow + alpha2*(kprimehigh-kprimelow)
        Revaluate the value function at every point
    else
        Update the upper bound of the searching range with b2
        Update b2 with b1
        Update b1 with kprimelow + alpha1*(kprimehigh-kprimelow)
        Revaluate the value function at every point
    end
    dk=abs(Vhigh-Vlow);
end
```

- Step 3, use linear interpolation to get the value function $V^{j+1}(\cdot)$ on the continuous state space.

We use the `griddedInterpolant` function in matlab to perform linear interpolation, instead of the `interp1` as suggested by the instruction. We do this for efficiency. `interp1` is slow because in the maximization step, we need to recalculate the interpolation every time when we update the value function at the lower and upper bound. This is too costly! By using the `griddedInterpolant` function, we can save a lot of time, because `griddedInterpolant` outputs a function handle, that can be reused again and again in the step 2.

- Step 4, calculate the difference between $V^{j+1}(\cdot)$ and $V^{j}(\cdot)$ on the discrete state space. If $\max_i |V^{j+1}(k^i) - V^j(k^i)|$ is smaller than some critical value, stop the iteration. Otherwise, go back to step 2.

Remark: The value function iteration with linear interpolation is much slower than the value function iteration with discrete maximization. With 100 grid points, the value function iteration with maximization over the discrete space takes approximately 1.7 seconds to finish. On the other hand, with linear interpolation, it takes 32 seconds to converge. There is substantial space to improve efficiency. For example, we can use other maximization algorithms for iteration.
Note that with `interp1` function, it takes 2 to 3 minutes to converge!

(b) Again plot the policy function $k'(k)$.

Figure 6 shows the policy function and the value function. Indeed, the calculated policy function is very similar to the previous result. Although it's not obvious, the policy function is concave, which ensures convergence back to the steady state.
In addition, we include the result created by value function iteration with and without linear interpolation in figure 7. We can see that with linear interpolation, the calculated policy function is much smoother than the calculation without interpolation. However, it should be noted that this comes with some loss of efficiency.

(c) Again evaluate the Euler equation errors. Comment on how they relate to those found in Problem 1.

The Euler equation errors are plotted in figure 8. Compared with the previous case, we can see that the error is much smaller, even with the same critical value for convergence. This suggests a trade-off between errors and calculation speed.

Also, the pattern is similar: there are some oscillations around some certain points. Ignoring these points of discontinuity, the errors are pretty smooth.

Why there is such pattern? We can find some inspiration from the polynomial interpolation with equally-spaced grids. When we use polynomial interpolation, especially high-degree polynomials, to approximate functions with discontinuities or steep gradients, the interpolated polynomial may exhibit oscillations or overshoot near the points of discontinuity. This phenomenon is called the Gibbs phenomenon.

Although value function iteration is not like polynomial interpolation, the intuition still carries through. For equally-spaced grid, in areas where the function changes abruptly, there may exist the problem of under-sampling: there are not enough grid points to capture these changes. We should note that value function iteration is a very well-performing algorithm, with few errors everywhere, and the spike of errors is very small indeed. But when the number of state variables increases and the model becomes more complicated, this might matter.

What does a substantial change in function mean? Derivative is the notion that comes to mind when we consider how a function changes. With discrete grids, we can actually capture the first-order derivative approximately. However, we cannot capture the second-order derivative: with linear interpolation, the derivative of the policy function and the value function change discontinuously, thus a discontinuous second-order derivative.

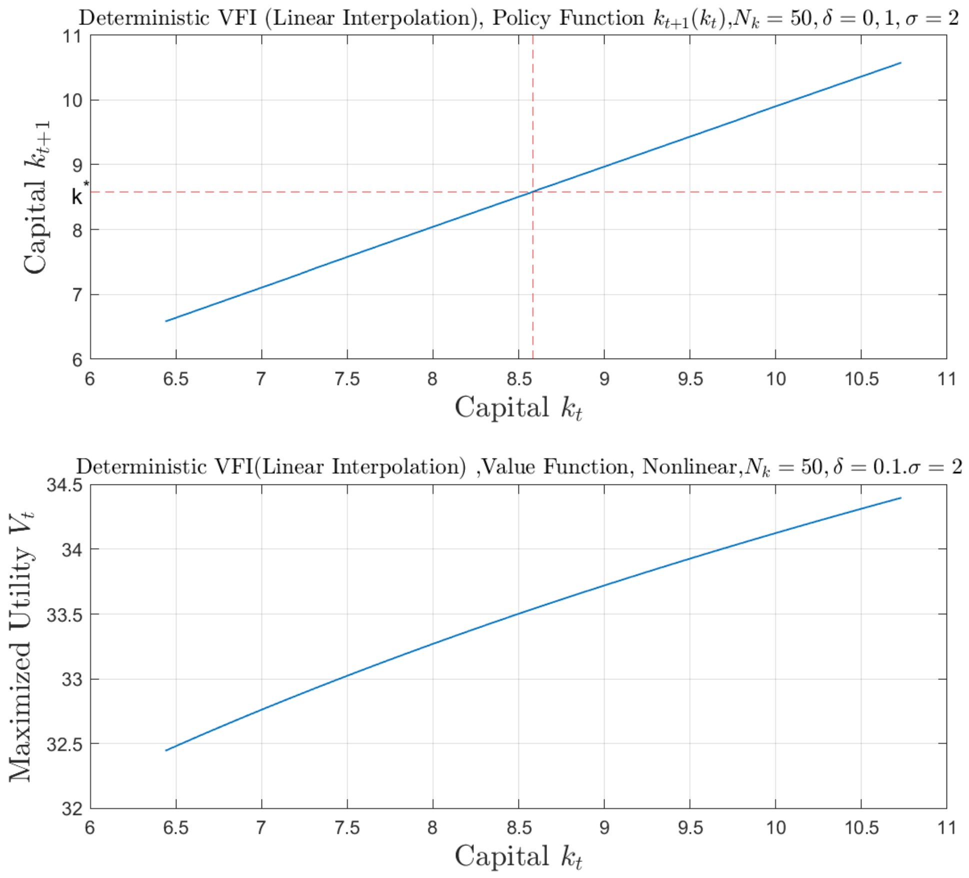

Figure 6: Deterministic VFI with Linear Interpolation, ($N_k = 50$, Linear Grid, $\delta = 0.1, \sigma = 2$)

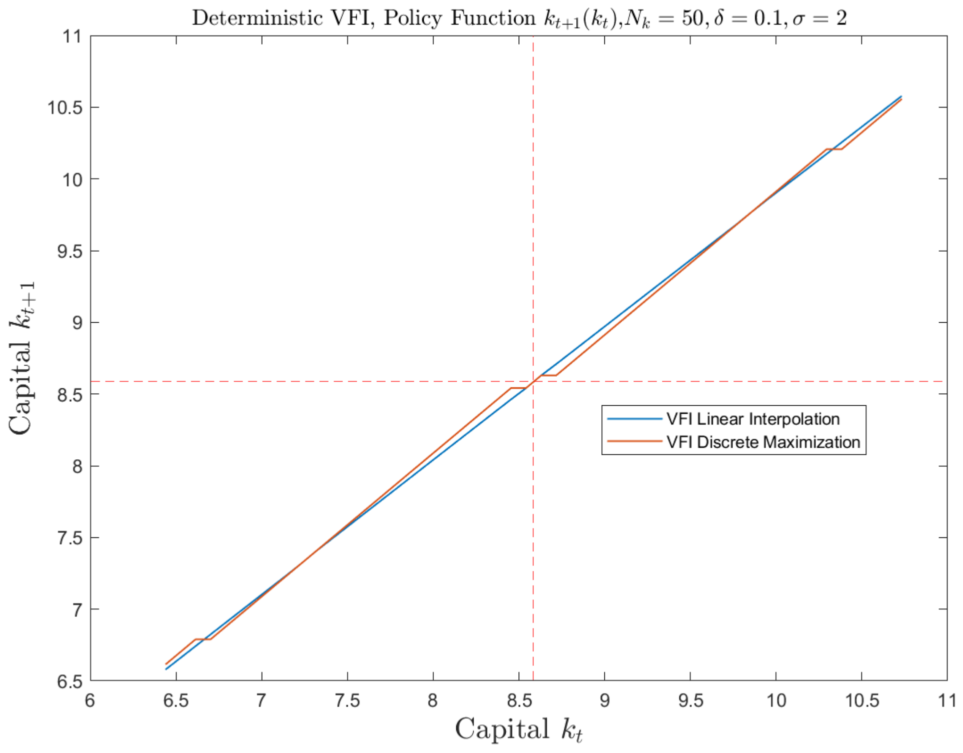

Figure 7: Comparison between VFI with and without Linear Interpolation

To show this idea in graphs, we apply the idea of relative risk aversion to the policy function. We calculate numerically

$$-\frac{k'(k) \cdot k'^{(2)}(k)}{k'^{(1)}(k)}$$

, where $k'(k)$ is the policy function, and $k'^{(1)}(k)$, $k'^{(2)}(k)$ are the first-order derivative and the second-order derivative of the policy function respectively. We use the central difference numerical differentiation to calculate the derivative on the grids. We call this notion "policy curvature" in our graph. We distinguish between the "numerical policy curvature" and the "true policy curvature". What we plot is the numerical policy curvature.

We plot the numerical policy curvature against the Euler equation error for the case of value function iteration without interpolation in figure 9. The result is stunning: the Euler equation error stays relatively stable when the numerical policy curvature is zero. When the numerical policy curvature shoots up, the Euler equation error also shoots up! There are two things to explain here: why the numerical policy curvature almost stays flat everywhere, and why the Euler equation error shoots up when the numerical policy curvature shoots up.

Firstly, the flatness of the numerical policy curvature can be explained by the discrete equally-spaced grid points. We iterate over the discrete grids, and the policy function takes value on the equally-spaced discrete grids. As the policy function increases, for most of the grids, the policy function simply maps one point in the discrete state space to the next point in the discrete state space. Also, the space between any two discrete points is equal, this leads to the fact that the first derivative approximated by the discretized policy function stays constant for most of the time. However, the contraction mapping ensures that the policy function will converge. Since the policy function is concave, to capture the changes in the first-order derivative, the approximated policy function will exhibit the pattern that there exist some spikes and troughs for the policy concavity around certain points. Another explanation is that the policy function here is not that concave. Indeed, one can barely tell whether the policy function is a straight line or not from the graph.

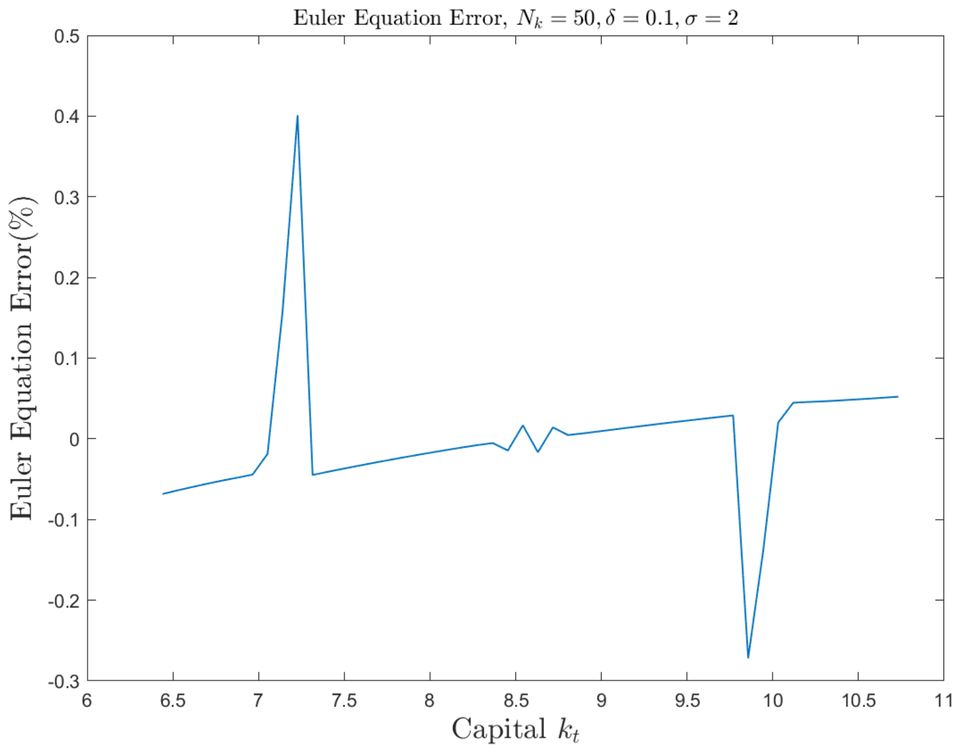

Figure 8: Euler Equation Error, VFI Linear Interpolation

Secondly, the estimation error connects closely to the numerical policy curvature. At points where the estimation doesn't need to adjust for the concavity of the policy function, VFI will focus on getting the value of the policy function correct. However, at points where VFI needs to adjust to capture the overall concavity of the policy function, VFI will prioritize changing the first-order derivative, ignoring the accuracy of the value of the value function, thus leading to high spikes in the numerical policy curvature and large errors at the same time.

In the linear interpolation case, the error is mitigated, because the policy function doesn't necessarily need to lie on the original discretized state space, thus capturing some changes in the first-order derivative. However, there are still some spikes in the Euler equation error, which can again be predicted by the numerical policy curvature. We show this in figure 10.

To further demonstrate this idea, I show the figure of the Euler equation error of the case where the utility is log utility and there is full depreciation. The policy function, as shown in equation 34, exhibits constant policy curvature, which is similar to the CRRA utility function. Thus, the estimation error generated by the value function iteration should stay stable. Figure 11 confirms this prediction.

In future numerical implementation, we should be aware of the importance of choosing the right grids and finding the right method to capture the second-order derivative of a function.

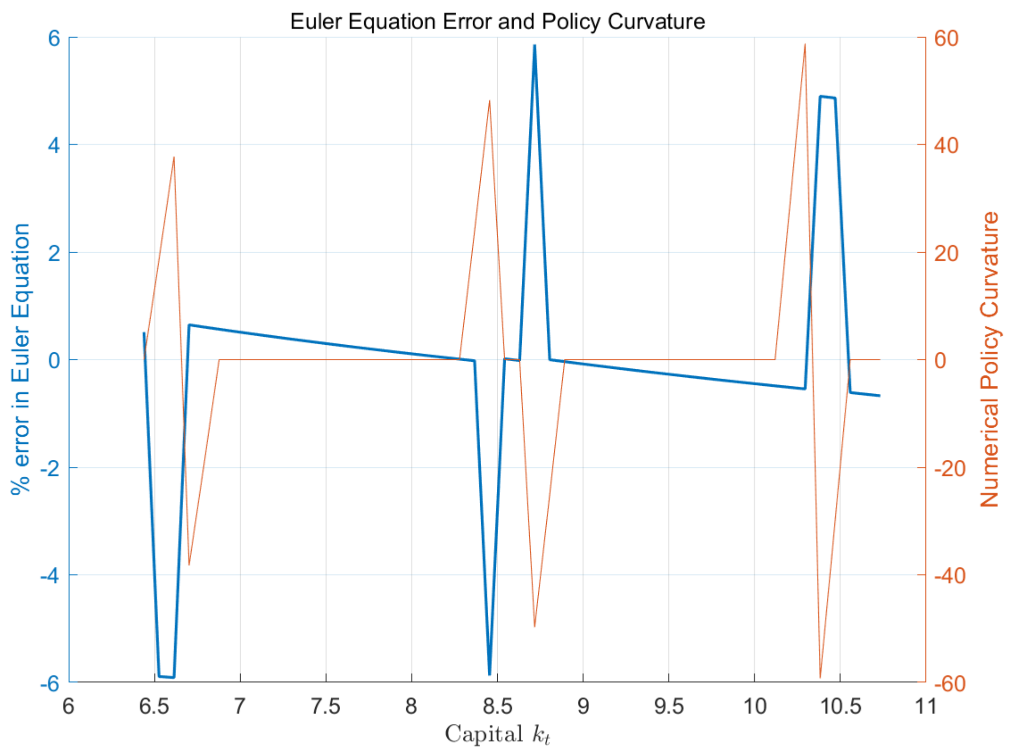

Figure 9: Numerical Policy Curvature and Euler Equation Error

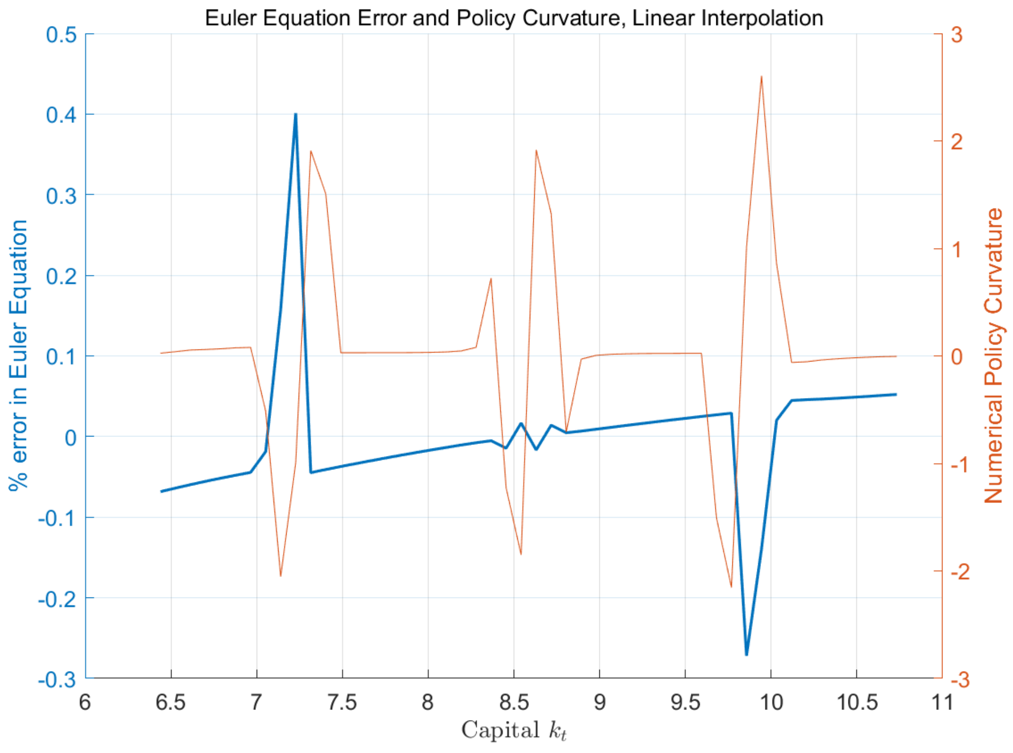

Figure 10: Numerical Policy Curvature and Euler Equation Error, VFI with Linear Interpolation

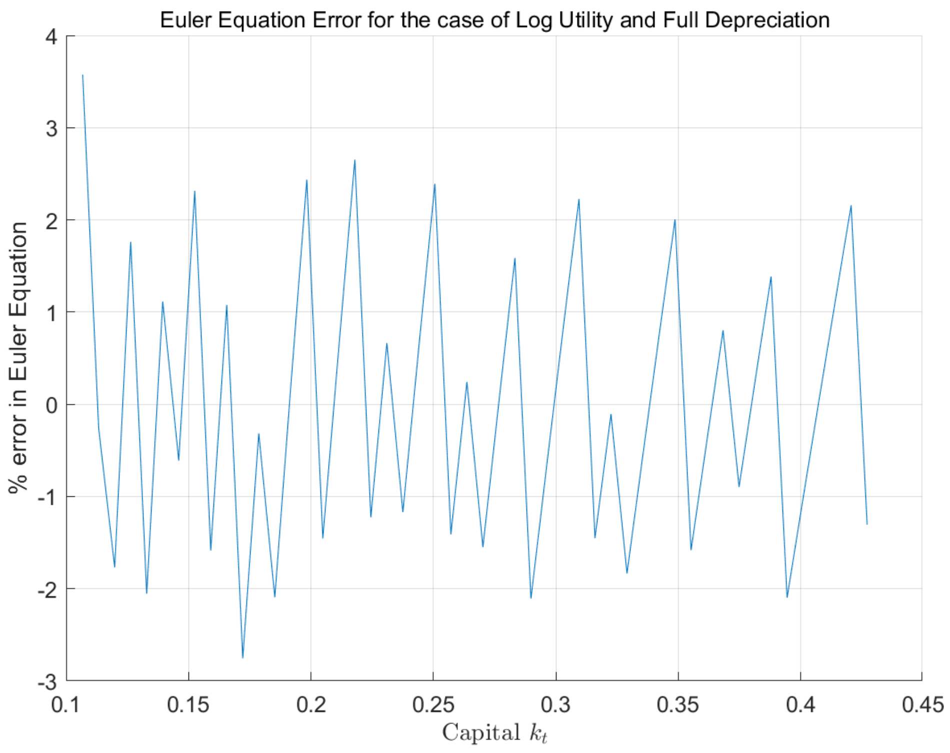

Figure 11: Euler Equation Error, Log Utility and Full Depreciation

4 Appendix

The following backward induction with T=3 is for Question 1. b) and c). Note that for periods t=3,2, and 1, we just take the derivation from Question 1 a) for corresponding t=2,1 and 0.

$$\begin{aligned} V_1(k_1) &= In\Big(\frac{k_1^\theta}{1+\beta\theta+(\beta\theta)^2}\Big) + \beta\theta(1+\beta\theta)In\Big(\frac{\beta\theta(1+\beta\theta)k_1^\theta}{1+\beta\theta+(\beta\theta)^2}\Big) + \beta In(\frac{1}{1+\beta\theta}) + \beta^2\theta In(\frac{\beta\theta}{1+\beta\theta}) \\ &= \theta(1+\beta\theta+(\beta\theta)^2)Ink_1 + In(\frac{1}{1+\beta\theta+(\beta\theta)^2}) + \beta\theta(1+\beta\theta)In(\frac{\beta\theta+(\beta\theta)^2}{1+\beta\theta+(\beta\theta)^2}) \\ &+ \beta In(\frac{1}{1+\beta\theta}) + \beta^2\theta In(\frac{\beta\theta}{1+\beta\theta}) \end{aligned} \tag{38}$$

$$k_1 = \frac{\beta\theta(1+\beta\theta+(\beta\theta)^2)k_0^\theta}{1+\beta\theta+(\beta\theta)^2+(\beta\theta)^3} \tag{39}$$

$$c_0 = \frac{k_0^\theta}{1+\beta\theta+(\beta\theta)^2+(\beta\theta)^3} \tag{40}$$

$$
\begin{aligned}
V_0(k_0) &= In(c_0) + \beta V_1(k_1) \\
&= In(c_0) + \beta(\theta(1+\beta\theta+(\beta\theta)^2)Ink_1 + In(\frac{1}{1+\beta\theta+(\beta\theta)^2}) + \beta\theta(1+\beta\theta)In(\frac{\beta\theta+(\beta\theta)^2}{1+\beta\theta+(\beta\theta)^2}) \\
&+ \beta In(\frac{1}{1+\beta\theta}) + \beta^2\theta In(\frac{\beta\theta}{1+\beta\theta})) \\
&= In(\frac{k_0^\theta}{1+\beta\theta+(\beta\theta)^2+(\beta\theta)^3}) + \beta\theta(1+\beta\theta+(\beta\theta)^2)In(\frac{\beta\theta(1+\beta\theta+(\beta\theta)^2)k_0^\theta}{1+\beta\theta+(\beta\theta)^2+(\beta\theta)^3}) \\
&+ \beta In(\frac{1}{1+\beta\theta+(\beta\theta)^2}) + \beta^2\theta(1+\beta\theta)In(\frac{\beta\theta+(\beta\theta)^2}{1+\beta\theta+(\beta\theta)^2}) + \beta^2 In(\frac{1}{1+\beta\theta}) + \beta^3\theta In(\frac{\beta\theta}{1+\beta\theta}) \\
&= \theta(1+\beta\theta+(\beta\theta)^2+(\beta\theta)^3)Ink_0 + In(\frac{1}{1+\beta\theta+(\beta\theta)^2+(\beta\theta)^3}) \\
&+ \beta\theta(1+\beta\theta+(\beta\theta)^2)In(\frac{\beta\theta(1+\beta\theta+(\beta\theta)^2)}{1+\beta\theta+(\beta\theta)^2+(\beta\theta)^3}) \\
&+ \beta In(\frac{1}{1+\beta\theta+(\beta\theta)^2}) + \beta^2\theta(1+\beta\theta)In(\frac{\beta\theta+(\beta\theta)^2}{1+\beta\theta+(\beta\theta)^2}) + \beta^2 In(\frac{1}{1+\beta\theta}) + \beta^3\theta In(\frac{\beta\theta}{1+\beta\theta})
\end{aligned}
\tag{41}
$$