Matthew Fitzgerald
Econ 204B
March 1, 2020

Problem Set # 4

Problem 1: Working in Academics

Individuals have preferences over consumption c and leisure l and are endowed with one unit of time. Workers maximize

$$\mathbb{E}\int_0^\infty e^{-rt}[c_t + v(l_t)]dt$$

where v is both increasing and concave with $v(1) = 0$. Suppose that there are three states of the world: unemployed, non-tenured, and tenured. Unemployed workers search for jobs and receive an unemployed utility flow of $b < \underline{w}$. They receive wage offers at a Poisson rate λ_0 from a distribution $F(w)$ where $F(\underline{w}) = 0$ and $F(\overline{w}) = 1$. If workers accept an offer, they enter the non-tenured state where they choose work intensity n such that $n + l = 1$. Non-tenured workers face the tenure-clock. Their tenure decision arrives at a Poisson rate λ_t and they have a probability $\alpha(n)$ that they receive tenure, otherwise they lose their job. Once tenured, workers are set for life. They receive their non-tenured wage w forever and cannot lose their job.

(a) Write down the flow Bellman for each state. Show that $V_T \geq V_{NT}$.

(b) Consider a world where $\alpha(n) = 1$. Derive the reservation wage and policy rule for work intensity.

(c) Assume that $\alpha(n)$ is strictly increasing and concave. Derive the reservation wage and policy rule for work intensity.

Problem 2: Value Function Iteration with Matrices

This problem will have you improve the efficiency of the code you completed in Problem 3 of Problem Set 3. You previously completed code that performed value function iteration by looping through all possible values of k in your capital grid. For each value of k in your grid, the code found the optimal k' to save for next period and computed the value of starting with k in this period (under the assumption that you will continue to act optimally in the future). In this problem, we will remove the inner loop and instead implement the code using matrices.

Open the file "ValueFnIterationWithMatricesIncomplete.jl".

(a) This file implements value function iteration, but it is incomplete. Fill in the code where necessary and make sure it runs properly by comparing it to the code from "ValueFunctionIteration.jl".

(b) Write additional code in the script that plots the value function and the policy function against k in separate graphs.

(c). Time your completed code in "ValueFnIterationWithMatricesIncomplete.jl" and in "ValueFunctionIteration.jl" using the following parameters: grid points = 1000, $\alpha = 0.3$, $\delta = 0.5$. Plugging in the appropriate parameter values, you can time your functions in these scripts using:

```
@time val_fun_iter_mats(prod_fun,num_grid_points, alpha, delta)
@time val_fun_iter(prod_fun,num_grid_points, alpha, delta)
```

Note that you should run each @time line twice and record the time from the second run. How much faster is your new code?

(d). What portion of code in "ValueFnIterationWithMatricesIncomplete.jl" has been taken out of the while loop compared to that in "ValueFunctionIteration.jl"? Why can we now remove this code from the while loop?