ECON 6130

Problem Set 3

Gabe Sekeres

October 12, 2024

Problem 1. Getting familiar with some basic facts about the U.S. Economy. Using FRED, find what the time series properties are over the last fifty years or so. What is the average, how much do they fluctuate, are there noticeable trends, what was their behavior over the latest recession, etc.

1. Consumption as a fraction of GDP.

   I used the series Personal consumption expenditures (DPCERE1Q156NBEA) from Q1 1974 to Q1 2024. This series is not seasonally adjusted, so there are some monthly trends we should ignore. Otherwise, it remained roughly constant from 1974 to 1980, then rose steadily from 60 percent to its maximum of 68.8 percent in 2009. It decreased slowly from the end of the Great Recession until it hit 66.1 in Q2 2020 (COVID Recession), then increased sharply to 68.5 in Q2 2021 and has remained roughly constant since. The average over the last fifty years has roughly been 65 percent, with monthly fluctuations but general upward trends during expansionary periods and downward trends during contractionary periods.

2. Investment as a fraction of GDP.

   I used the series Gross private domestic investment (A006RE1Q156NBEA) from Q1 1974 to Q1 2024. This series is not seasonally adjusted, so there are monthly trends we should ignore. Otherwise, it shows strong behavior of rising (fairly sharply) during expansionary periods, and falling even more sharply during contractionary periods. The exception to this is the period from 1984 to 1990, in which there was no recession but investment share of GDP still fell. On average, it was roughly 18 percent, but it fluctuated a lot over that period, from a low of 12.7 in Q3 2009 (right after the Great Recession) to a high of 20.8 from Q4 1978 to Q2 1979. During the COVID Recession, it dropped sharply for a single quarter and then rose again back to almost exactly the level before the recession. It has stayed fairly steady since.

3. Government (federal and all levels) spending as a fraction of GDP.

   I used the series Government consumption expenditures and gross investment (A822RE1Q156NBEA) from Q1 1974 to Q1 2024. This series is not seasonally adjusted, so there are some monthly trends we will ignore. Roughly this series is decreasing over the last fifty years. It tends to decrease during expansionary periods and then sharply increase during contractionary periods, as the government pursues countercyclical fiscal policy. On average, it is 20 percent over the period, from a high of 22.9 percent in Q1 1975 to a low of 17.2 percent in Q3 2022. It fluctuates less quarter to quarter than the previous time series, but tends to have strong long-run trends. During the COVID Recession, spending as a fraction of GDP increased sharply for a single period and then returned to the previous rate.

4. Payments to labor as a share of GDP.

   I used the series Share of Labour Compensation in GDP (LABSHPUSA156NRUG) from 1969 to 2019. First, a quick note about this series (and the next). It was created by a team of researchers from the University of Groningen and the University of California, Davis, rather than the U.S. Bureau of Economic Analysis. It is yearly, and only goes to 2019. It is harder to see trends with these yearly data than the above quarterly data, so any claims here will be lower confidence. This series is roughly decreasing over the course of the last fifty years, and it tends to stay steady or grow during expansionary periods and sharply drop during contractionary periods. The largest period of sustained growth was from 1996 to 2001, and the sharpest drop was during the 1974 recession. The minimum was in 2010, at 58.8 percent, the maximum was in 1970 at 64.9 percent, and the average was roughly 61.5 percent. Over the Great Recession (the latest in the sample), there was an extremely sharp drop in share.

5. Payments to capital as a share of GDP.

   I used the same series as part (4), Share of Labour Compensation in GDP (LABSHPUSA156NRUG) from 1969 to 2019. As all GDP that is not payments to labor is payments to capital, all trends are the inverse of that part of the question. If you take all of the numbers in (4), do 100 minus them, and change 'increase' to 'decrease' and vice versa, that's the answer to this.

6. The growth rate of output per capita.

   I used the series Real GDP per capita (A939RX0Q048SBEA), considering the percent change from one year ago, from Q2 1973 to Q2 2023. This series is roughly constant over the last fifty years, though it does see sharp peaks during expansionary periods and troughs during contractionary periods. It maintains an average of approximately 2 percent, though it fluctuates more than earlier series, despite being seasonally adjusted. During recessions, it is (definitionally) negative, though it often returns to the pre-recession trend fairly quickly. During the COVID Recession, it reached its global minimum of -7.9 percent in Q2 2020, followed shortly by its global maximum of 11.8 percent in Q2 2021.

7. The growth rate of consumption per capita.

   I used the series Real personal consumption expenditures per capita (A794RX0Q048SBEA), considering the percent change from one year ago, from Q1 1974 to Q1 2024. This series has almost precisely the same trends as the growth rate of output per capita (6), where it drops sharply during contractionary periods and rises sharply (and then maintains) in expansionary periods. It has a slightly higher average, of about 2.5 percent, though its peaks and troughs are higher and lower than (6). With one exception (the 2001 recession) it is negative during recessions, and during the COVID Recession it also reached its global minimum of -9.0 percent in Q2 2020 followed by its global maximum of 16.2 percent in Q2 2021.

8. Civilian unemployment rate.

   I used the series Unemployment Rate (UNRATE), from Aug 1974 to Aug 2024. This series rises sharply during contractionary periods, and then slowly decreases during expansionary periods. Over the last fifty years, it maintained an average of approximately 6 percent, though it generally returns to a lower level after long periods of expansion than it was before the recession in question. It mostly fluctuates during recessions, and has a fairly steady trend during expansionary periods. It attained its global minimum of 3.5 percent in Feb 2020, right before the COVID Recession, and attained its global maximum of 14.8 in Apr 2020.

9. Average duration of unemployment.

   I used the series Average Weeks Unemployed (UEMPMEAN), from Aug 1974 to Aug 2024. Over the last fifty years, this series has tended to rise during contractionary periods, reaching peaks after recessions, and then falling slowly until the next recession. It has broadly risen over time, largely driven by a massive rise following the Great Recession. It maintained an average of 18 weeks until that recession, and an average of 28 weeks since. Its maximum was 40.5, reached in Aug 2011, and it actually reached its minimum of 7.1 in Apr 2020. This is clearly a function of the COVID lockdowns creating a large set of short-term unemployed workers, evidenced by the massive jump in civilian unemployment in Apr 2020.

Problem 2. Consider the same problem as in slides 20 to 27 of Neoclassical Growth, except $\delta = 0.75$

1. Formulate the planner's optimization problem as a dynamic programming problem.

We have that $U(c) = \ln(c)$, $F(k, n) = k^\alpha n^{1-\alpha}$, and $\delta = 0.75$. This implies that $f(k) = F(k, 1) + (1 - \delta)k = k^\alpha + (1 - \delta)k$. We can write the Bellman equation

$$v(k) = \max_{0 \leq k' \leq f(k)} \{U(f(k) - k') + \beta v(k')\}$$

In our terms, we have that

$$v(k) = \max_{0 \leq k' \leq k^\alpha + (1-\delta)k} \{\ln(k^\alpha + (1 - \delta)k - k') + \beta v(k')\}$$

Where $k' = g(k)$ for some optimal policy function g. The state variable is k, and the choice variable is $k' = g(k)$.

2. Using a value function iteration approach, solve for the value and policy functions in this problem. Plot them as functions of k.

As in the notes, we will assume that $\alpha = 0.3$ and $\beta = 0.6$. It will also be necessary to discretize the space. To find what range to discretize around, we can solve for the steady state of k, using the Euler equation method. First, we will impose a transversality condition, where we assume that

$$\lim_{t \to \infty} \beta U'(f(k_t) - k_{t+1}) f'(k_t) k_0 = 0$$

Assuming that, we can find the steady state. Our Euler equation is

$$U'(f(k_t) - k_{t+1}) = \beta U'(f(k_{t+1}) - k_{t+2}) f'(k_{t+1})$$

which is

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

Define $z_t := \frac{k_{t+1}}{k_t^\alpha + (1-\delta)k_t}$. Then this equation becomes

$$\frac{\beta(\alpha k_{t+1}^\alpha + (1 - \delta)k_{t+1})}{k_{t+2}} \left(\frac{1}{z_t} - 1\right) = \frac{1}{z_{t+1}} - 1$$

which implies that

$$\underbrace{\frac{\beta(1 - \alpha)k_{t+1}^\alpha}{k_{t+2}}}_{A} \frac{\beta}{z_{t+1}} \left(\frac{1}{z_t} - 1\right) = \frac{1}{z_{t+1}} - 1$$

So we have

$$z_{t+1} = 1 + A\beta - \frac{A\beta}{z_t}$$

A steady state is reached whenever $z_{t+1} = z_t = z$ for all t, so when

$$z = 1 + A\beta - \frac{A\beta}{z} \implies (z - 1)(z - A\beta) = 0$$

The two solutions are when $z_t = 1$, and when $z_t = A\beta$. Note that the latter solution implies that

$$\frac{k_{t+1}}{k_t^\alpha + (1 - \delta)k_t} = -\frac{\beta^2(1 - \alpha)k_t^\alpha}{k_{t+1}} \implies k_{t+1} = \sqrt{-\beta^2(1 - \alpha)k_t^\alpha(k_t^\alpha + (1 - \delta)k_t)}$$

which is a non-real complex number. Thus the only viable solution is when $z_t = 1$, which is when

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

and substituting in our parameters, we get that this is met when

$$k^\star = (k^\star)^{0.3} + 0.25k^\star \implies k^\star \approx 1.508$$

We will thus discretize the space around $k = 1.508$. Specifically, we will take 10,000 values between $0.25k^\star$ and $1.75k^\star$.

I used two methods to code this. For my own edification, I manually wrote up the code in Julia, and I also modified the code Ekaterina provided in MATLAB. I produced the following figures with Julia, with an error tolerance of $1 \cdot 10^{-8}$:

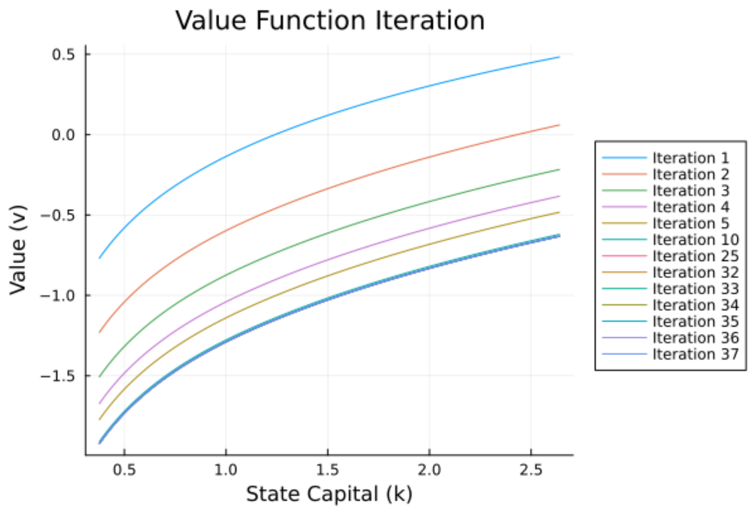

Figure 1: Value Function Iteration (Julia)

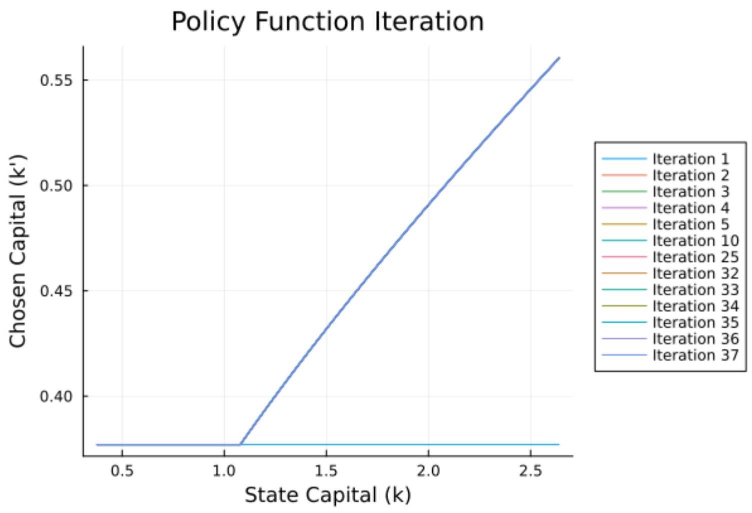

Figure 2: Policy Function Iteration (Julia)

The Julia code to perform the value function iteration is:

```
# First, define the capital space and number of values for k
```

```
k_ss = 1.508
N = 10000

K = collect(range(start = 0.25 * k_ss, stop = 1.75 * k_ss, length =  N))

# Define the previous value function, current value function, policy
   function, and parameters
v_prev = zeros(N)
v = zeros(N)
g = zeros(N)
alpha = 0.3
beta = 0.6
delta = 0.75
tolerance = 1e-8
dist = 1.0

# Define the iterations for the value and policy functions
value_fns = []
policy_fns = []

# Next, iterate until you reach a certain tolerance between iterations
while dist > tolerance
    global dist, v_prev, v, g
    for i in 1:N
        # Calculate possible capital choices
        capital_choices = K[i] ^ alpha + (1 - delta) * K[i] .- K
        valid_choices = findall(x -> x > 0, capital_choices)

        if !isempty(valid_choices)
            # Only evaluate the log on valid choices
            log_choices = log.(capital_choices[valid_choices]) .+ beta *
               v_prev[valid_choices]
            optimal_index = argmax(log_choices)
            g[i] = K[valid_choices[optimal_index]]
            v[i] = log(K[i] ^ alpha + (1 - delta) * K[i] - g[i]) + beta*
               v_prev[findfirst(==(g[i]), K)]
        else
            # If no valid choices, set to a default value
            g[i] = 0
            v[i] = -Inf # Log of zero / negative undefined, so set it to -
               Inf
        end
    end

    dist = maximum(abs.(v .- v_prev))
    v_prev = copy(v)

    push!(value_fns, copy(v))
    push!(policy_fns, copy(g))
end
```

```
using Plots

iterations_to_plot = [1, 2, 3, 4, 5, 10, 25, length(policy_fns) - 5,
   length(policy_fns) - 4, length(policy_fns) - 3, length(policy_fns) -
   2, length(policy_fns) - 1, length(policy_fns)]

p = plot()
for i in iterations_to_plot
    plot!(p, K, value_fns[i], label = "Iteration $i", title = "Value
       Function Iteration", xlabel = "State Capital (k)", ylabel = "Value
        (v)")
end

plot!(p, legend=:outerright)
display(p)
savefig(p,"pset3_value_iteration.png")

p2 = plot()
for i in iterations_to_plot
    plot!(p2, K, policy_fns[i], label = "Iteration $i", title = "Policy
       Function Iteration", xlabel = "State Capital (k)", ylabel = "
       Chosen Capital (k')")
end

plot!(p2, legend=:outerright)
display(p2)
savefig(p2,"pset3_policy_iteration.png")
```

For the MATLAB code, I was having severe runtime issues using the same parameters as in Julia. I reduced the discretization to 1,000 values and increased the error tolerance to $1 \cdot 10^{-6}$. I got the following figures (just of the final functions, not the entire iterations:

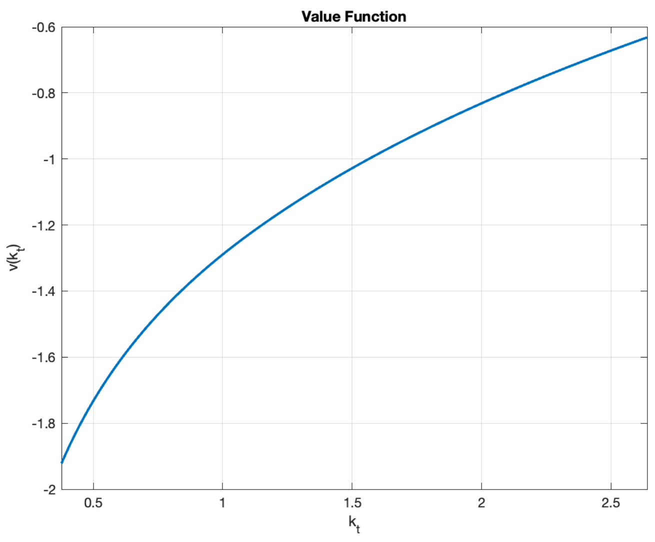

Figure 3: Value Function (MATLAB)

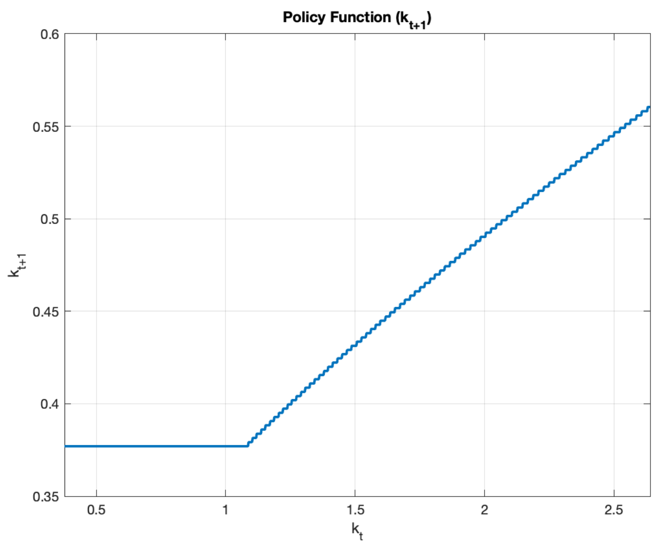

Figure 4: Policy Function (MATLAB)

Allowing for the differences in scale, these plots are the exact same as in Julia. I used the code:

```
clc;
clear all;
close all;
format compact

tic

% Set parameters:
alpha = 0.3;
beta = 0.6;
delta = 0.75;
diff = 1;
tol = 1e-6;
its = 1;

% Utility function:
u = @(c) (c>0).*log(c)+(c<=0).*(-1000);

kss = 1.508;
N = 1000;
kmin = 0.25 * kss;
kmax = 1.75 * kss;
kgrid = linspace(kmin, kmax, N);

val_fun = zeros(1, N);
pol_fun_idx = zeros(1, N);

% Iterate:
while diff > tol
```

```
    for i = 1:N
        c = (kgrid(i)^alpha + (1 - delta)*kgrid(i)) - kgrid;
        [val_new(i), pol_fun_idx(i)] = max(u(c) + beta * val_fun);
    end
    diff = max(abs((val_new - val_fun)));
    val_fun = val_new;
    its = its + 1;
end

pol_fun = kgrid(pol_fun_idx);
cons = (kgrid .^ alpha) - pol_fun;

toc

% Plots:
figure(1)
plot(kgrid, pol_fun,'linewidth',1.8); title('Policy Function (k_{t+1})');
   ...
    xlabel('k_t'); ylabel('k_{t+1}'); grid on ; hold on; ...
    xlim([kmin kmax]); saveas(gcf,'pset3_policy_function.png')

figure(2)
plot(kgrid, val_fun,'linewidth',1.8); title('Value Function'); ...
    xlabel('k_t'); ylabel('v(k_t)'); grid on ; hold on;  ...
    xlim([kmin kmax]); saveas(gcf,'pset3_value_function.png')
```