Dynamic Programming Notes

Richard W. Evans, University of Chicago
https://sites.google.com/site/rickecon/

January 2020

k_{t+1} = k_{t} + \delta * k_{t} + i_{t}
\delta: deprecation rate
i_{t}: investment related with firm decisions
k_{t+1} = (k_{t}, i_{t} | \theta)

dynamic decision: looking for functions to solve problems

1 Introduction

It is hard to think of any decision that is not dynamic and does not involve some trade-off between the current period and some future period. Examples of dynamic decisions include the classical consumption-savings decision, marriage, eduction, labor-leisure. Dynamic decision making is ubiquitous. In fact, static models are often simply an approximation of the more realistic dynamic setting.

The term "Dynamic programming" initially was the name for the program of research that studied how to solve dynamic models (see Dreyfus, 2002). Dynamic programming has since come to be associated with the particular dynamic solution method of value function iteration pioneered by Richard Bellman.

Although dynamic modeling is very rich and realistic, it is also very hard. Take, for example, the law of motion for the capital stock.

$$k_{t+1} = k_t - \delta k_t + i_t \tag{1}$$

This equation says that the value of the capital stock tomorrow k_{t+1} is equal to the value of the capital stock today k_t minus the percent of the capital stock that depreciates between today and tomorrow δk_t plus how ever much value (investment) you put back into the capital stock today i_t. This equation is dynamic because it has current period variables that have a t subscript and next period variables that have a $t + 1$ subscript.

How many sequences of capital stock values and investment values $\{k_t, i_t\}_{t=0}^{\infty}$ satisfy the law of motion for capital (1)? One potential sequence is to have the initial capital stock be $k_0 > 0$ and have investment be $i_t = 0$ for all t. This would make the capital stock get smaller and smaller every period $k_{t+1} < k_t$ until the capital stock gets very close to zero $\lim_{t\to\infty} k_t = 0$. A different sequence of capital stocks and investment amounts that satisfies (1) is the investment amount that exactly offsets the depreciation, thereby keeping the capital stock constant $i_t = \delta k_t$ for all t. This is termed the steady state. You could also come up with infinitely many other sequences that satisfy (1), where capital k_t and investment i_t are fluctuating. More structure on the model—like household optimization or firm profit maximization conditions—helps pin down the set of possible sequences satisfying (1).

第一种满足条件的可能

第二种满足条件的可能

In this chapter, we are going to look at a very particular class of dynamic problems with "nice" properties. These will be models that can be written in recursive form and that can be transformed into a contraction mapping.[1] We will learn to solve them using a particular solution method: value function iteration. But, as you already know, there are many solution methods to dynamic models. The best one depends on what model you are using and what your research question is. Value function iteration is a nonlinear solution method for dynamic models.

A general form of the Bellman equation of a recursive model is the following,

$$V(x, y) = \max_{y'} \sigma(x, y, y') + \beta E\left[V(x', y')\right] \tag{2}$$

where x is the exogenous state (set of exogenous variables), y is the endogenous state (set of choice variables), y' is the control or choice variable, $\sigma(\cdot)$ is the period objective function, and $V(x, y)$ is the value function. The value function tells you the value to the agent of showing up in a given period with state (x, y), and the value function accounts for all expected future benefits in addition to the current period benefits.

period utility

discounted expected continuation value

[1] Good references for how to solve these problems are Stokey et al. (1989) and Adda and Cooper (2003).

2 The Sequence Problem

Assume individuals have perfect foresight (no uncertainty) and that remaining lifetime utility U of an agent who lives T periods is given by the following equation,

$$U = \sum_{t=1}^{T} \beta^{t-1} u(c_t)$$

\beta^{t-1}

where $\beta \in (0,1)$ is the discount factor and $u(\cdot)$ is a period utility function that is increasing, continuous, continuously differentiable, and concave $(u' > 0, u'' < 0)$.

Using the notation and example of Adda and Cooper (2003), we assume that individuals are choosing how much to consume each period c_t of a cake that starts out with size $W_1 > 0$. So the law of motion for the size of the cake at the end of period t is the following.

$$W_{t+1} = W_t - c_t \tag{3}$$

W_{t}: cake size in period t
budget constraint: W_{t+1} = W_{t} - c_{t}

You can think of the size of the cake at the beginning of a period W_t as being given.

The optimization problem for the individual is to choose consumption in each period c_t in order to maximize lifetime utility U_t subject to the constraint on the size of the cake (law of motion).

$$\max_{c_t \in [0, W_t]} \sum_{t=1}^{T} \beta^{t-1} u(c_t) \quad \text{s.t} \quad W_{t+1} = W_t - c_t$$

This problem can be rewritten in the following way by substituting the law of motion for the cake size into the utility function.

$$\max_{W_{t+1} \in [0, W_t]} \sum_{t=1}^{T} \beta^{t-1} u(W_t - W_{t+1})$$

meaning the change in the cakesize
c_{t} >= 0
0 <= W_{t-1} <= W_{t}

Exercise 1. If the individual lives for one period $T = 1$, what is the condition that characterizes the optimal amount of cake to eat in period 1? Write the problem in the equivalent way of showing what the condition is for the optimal amount of cake to save for the next period W_{T+1} or W_2.

T = 1: max(W_{T+1}) u(W_{t} - W_{t+1})
W_{t+1} = min(W_{t}) = 0 for any W_{t}
U = u(W_{t})

T = 2: max(W_{2}, W_{3}) \sum^{T=2}_{t=1} \beta^{t−1} * u(W_{t} − W_{t+1})
max(W_2, W_3) u(W_1−W_2) + \beta * u(W2_W3)

trade−off of the W2: increase utility in the period 2 but reduces utility in the period 1

functional solutions! set W_{3} = min(W3) = 0 for any W_{1}, W_{2}

F.O.C.s W_{2} = dU/dW_{2} = 0 => dU(c_{1}) / dc_{1} = \beta * dU(c_{2}) / dc_{2} condition

u(c) = ln(c)
1/c_{1} = \beta * 1 /c_{2}
i / (w_{1} - w_{2}) = \beta * 1/ w_{2}
w_{2} = \beta * (w_{1} - w_{2})
w_{2} = \beta \ (1 + \beta) * w_{1}
cake size in the period 1 goes up by 1 unit, cake size in the period 2 increase by less than 1 unit

Exercise 2. If the individual lives for two periods $T = 2$, what is the condition that characterizes the optimal amount of cake to leave for the next period W_3 in period 2? What is the condition that characterizes the optimal amount of cake leave for the next period W_2 in period 1?

Exercise 3. If the individual lives for three periods $T = 3$ what are the conditions that characterize the optimal amount of cake to leave for the next period in each period $\{W_2, W_3, W_4\}$? Now assume that the initial cake size is $W_1 = 1$, the discount factor is $\beta = 0.9$, and the period utility function is $\ln(c_t)$. Show how $\{c_t\}_{t=1}^{3}$ and $\{W_t\}_{t=1}^{4}$ evolve over the three periods.

3 The Recursive Problem, Finite Horizon

Now we want to define a general function, called a value function $V_T(W_T)$, that represents the value to an individual of entering the last period of his life T with a cake of size W_T.[2]

assume an individual lives for T periods

$$V_T(W_T) \equiv \max_{W_{T+1}} u(W_T - W_{T+1}) \tag{4}$$

The solution to this problem is a policy function for W_{T+1} that is a function of the state $W_{T+1} = \psi_T(W_T)$ that maximizes the value of entering the period with state W_T, and the corresponding value function $V_T(W_T)$. The state refers to all the variables that are known by the individual at the time of the decision and that are relevant to the decision (W_T in this case). So one way to think of this problem is as a policy function $W_{T+1} = \psi_T(W_T)$ that satisfies a value function condition.

So the value function for the last period of an individual's life can be rewritten as the utility of choosing the optimal amount of cake to save for the next period,

$$V_T(W_T) = u\Big(W_T - \psi_T(W_T)\Big) \tag{5}$$

[2]Note that the notation for the value function V_t and policy function ψ_t with a time subscript is different from the notation in chapter 1 of Adda and Cooper (2003). Here, the time subscript denotes a different function for each period of time, whereas the subscript T in Adda and Cooper (2003) denotes a function in which the total number of periods is T.

where $W_{T+1} = \psi_T(W_T) = \min\{W_T\} = 0$ maximizes the period-T value function. This solution is equivalent to the one-period problem from Exercise 1. Again, this means that the optimal cake-saving policy in the last period of one's life is to save no cake. So the value to an individual of entering the last period of his life with cake of size W_T is equal to the utility of eating W_T of cake.

$$V_T(W_T) = u(W_T) \tag{6}$$

The problem of the individual in period $T-1$ becomes more interesting. One way to write the value function of entering period $T-1$ with a cake of size W_{T-1} is to characterize it as a function of the discounted sum of utilities in which W_T and W_{T+1} are chosen optimally. ????

$$V_{T-1}(W_{T-1}) \equiv \max_{W_T, W_{T+1}} u(W_{T-1} - W_T) + \beta u(W_T - W_{T+1})$$

W_{T+1} = 0

If we make an envelope theorem assumption that W_{T+1} will be chosen optimally in period T according to (5), we can rewrite the problem in the following way,

$$V_{T-1}(W_{T-1}) \equiv \max_{W_T} u(W_{T-1} - W_T) + \beta V_T(W_T) \tag{7}$$

value functions and decision functions are different in different periods.

where $V_T(W_T)$ is defined in (4). This is the finite horizon version of the famous recursive workhorse, the Bellman equation.[3] The assumption that future choices will be made optimally is called the principle of optimality.

Exercise 4. Using the envelope theorem that says W_{T+1} will be chosen optimally in the next period, show the condition that characterizes the optimal choice (the policy function) in period $T-1$ for $W_T = \psi_{T-1}(W_{T-1})$. Show the value function V_{T-1} in terms of $\psi_{T-1}(W_{T-1})$.

Exercise 5. Let $u(c) = \ln(c)$. Show that $V_{T-1}(\bar{W})$ does not equal $V_T(\bar{W})$ and that $\psi_{T-1}(\bar{W})$ does not equal $\psi_T(\bar{W})$ for a cake size of $\bar{W}$ when $T < \infty$ represents the last period of an individual's life.

[3] Dreyfus (2002) is a good reference for the origin of the Bellman equation.

Exercise 6. Using $u(c) = \ln(c)$, write the finite horizon Bellman equation for the value function at time $T-2$. Characterize the solution for the period $T-2$ policy function for how much cake to save for the next period $W_{T-1} = \psi_{T-2}(W_{T-2})$ using the envelope theorem (the principle of optimality) and write its analytical solution. Also, write the analytical solution for V_{T-2}.

Exercise 7. Using $u(c) = \ln(c)$ and the answers to Exercises 5 and 6, write down the expressions for the analytical solutions for $\psi_{T-s}(W_{T-s})$ and $V_{T-s}(W_{T-s})$ for the general integer $s \geq 1$ using induction. Show that $\lim_{s\to\infty} V_{T-s}(W_{T-s}) = V(W_{T-s})$ and that $\lim_{s\to\infty} \psi_{T-s}(W_{T-s}) = \psi(W_{T-s})$. That is, as the horizon becomes further and further away (infinite), the value function and policy function become independent of time. Another way of saying this is the following. The value of entering a period t with a certain amount of cake when the end of your life is far enough away only depends on how much cake there is W_t, not in what period you have that amount of cake.

Exercise 8. Write the Bellman equation for the cake eating problem with a general utility function $u(c)$ when the horizon is infinite (i.e., either $T = \infty$ or $s = \infty$).

4 The Recursive Problem, Infinite Horizon

We can add \pho as the mortality rate to modify the weights of future values

You showed in Exercise 7 in Section 3 that the value function and policy function in the Bellman equation for the infinite horizon problem are independent of time. So everything can now be written in terms of variables today and variables tomorrow. We will denote variables tomorrow with a "$'$".

same value function taking different parameters as values

$$V(W) = \max_{W' \in [0,W]} u(W - W') + \beta V(W') \tag{8}$$

Note that the value function V on the left-hand-side of (8) and on the right-hand-side are the same function. This is what you showed in Exercise 7.

Because the problem now has an infinite horizon, the nature of the solution is a little different. The solution to (8) is a policy function $W' = \psi(W)$ that creates a fixed

point in V. In other words, the solution is a policy function $\psi(W)$ that makes the function V on the left-hand-side of (8) equal the function V on the right-hand-side.

Another way of thinking of the problem is that the Bellman equation is one equation with two unknowns—the value function V and the policy function ψ. The condition that renders this problem identified is that the Bellman equation must be a contraction mapping. In a sense, the contraction mapping condition pins down the value function V.

Define C as an operator on any value function $V_t(W)$. Let C perform the following operation.[4]

$$C\Big(V_t(W)\Big) \equiv \max_{W' \in [0,W]} u(W - W') + \beta V_t(W') \tag{9}$$

Note that the value function on the right-hand-side of (9) and on the left-hand-side are the same function V_t, but have a different value of the size of the cake—W versus W'. For a reason that will become apparent in a moment, define the resulting function from the C operator as V_{t-1}.

$$V_{t-1}(W) \equiv C\Big(V_t(W)\Big) \tag{10}$$

The value function that results from the C operation V_{t-1} is not necessarily the same as the value function that the system began with V_t. The solution, then, is the fixed point in V.

$$C\Big(V_t(W)\Big) = V_{t-1}(W) = V_t(W) = V(W)$$

C: contraction operator

Definition 1 (Contraction mapping). Let (S, ρ) be a metric space and $C : S \to S$ be a function mapping S onto itself. C is a contraction mapping (with modulus β) if for some $\beta \in (0,1)$, $\rho(Cx, Cy) \leq \beta\rho(x,y)$, for all $x, y \in S$.

模

The operator $C(\cdot)$ is called a contraction mapping if applying it over and over again to an arbitrary value function V_t converges to a fixed point. One way to characterize

[4] I use a subscript t here to denote the iteration number. I have the contraction operator $C(\cdot)$ advance the iteration number backward $t-1$ to maintain the backwards induction analogy from the previous exercises of solving for value functions from some terminal period T.

a contraction mapping is:

$$\lim_{s\to\infty} C^s\Big(V_t\left(W\right)\Big) = V\left(W\right)$$

Theorem 1 (Contraction mapping theorem). If (S, ρ) is a complete metric space and $C : S \to S$ is a contraction mapping with modulus β, then

- C has exactly one fixed point v in S, and
- for any $v_0 \in S$, $\rho(C^n v_0, v) \leq \beta^n \rho(v_0, v)$ for $n = 0, 1, 2, ...$

A set of sufficient conditions for $C(\cdot)$ to be a contraction mapping are due to Blackwell (1965) and are that C be monotonic and that it have the property of discounting.[5] Adda and Cooper (2003) outline one set of sufficient conditions that ensure our problem is a contraction mapping and therefore has a unique solution. These conditions are that the period utility function $u(\cdot)$ must be real-valued, continuous, and bounded, that $\beta \in (0, 1)$, and that the constraint set $W' \in [0, W]$ be nonempty, compact-valued, and continuous.

Theorem 2 (Blackwell's sufficient conditions for a contraction). Let $X \subseteq \mathbb{R}^l$, and let $B(X)$ be a space of bounded functions $f : X \to \mathbb{R}$, with the sup norm. Let $C : B(X) \to B(X)$ be an operator satisfying the following two conditions.

- (monotonicity) $f, g \in B(X)$ and $f(x) \leq g(x)$ for all $x \in X$, implies $(Cf)(x) \leq (Cg)(x)$, for all $x \in X$;
- (discounting) there exists some $\beta \in (0, 1)$ such that

$$\left[C(f + a)\right](x) \leq (Cf)(x) + \beta a \quad \text{for all} \quad f \in B(X), \quad a \geq 0, \quad x \in X$$

[Here $(f + a)(x)$ is the function defined by $(f + a)(x) = f(x) + a$.] Then C is a contraction with modulus β.

[5] A more formal definition of a contraction mapping and the corresponding sufficient conditions is given in Stokey et al. (1989, pp. 49-55). The conditions for a contraction mapping are due to Blackwell (1965) and are often called "Blackwell's sufficient conditions for a contraction."

Before moving on to the exercises in which you will solve for the value function V and the policy function ψ by utilizing the contraction mapping theorem, it is important to communicate one last important reason why it works. In the recursive finite horizon problem from Section 3, you could always solve for V and ψ. The Bellman equation was a second-order difference equation in W_t. This is easily seen in the period $T-2$ problem from Exercise 6.

$$V_{T-2}(W_{T-2}) = \max_{W_{T-1}} u\big(W_{T-2} - W_{T-1}\big) + \beta u\big(W_{T-1} - W_T\big) + \beta^2 u\big(W_T\big) \tag{11}$$

This is a difference equation with cakes sizes in three different periods, W_{T-2}, W_{T-1}, and W_T. We could solve for the value function V and policy function ψ in this second order difference equation because we had an initial condition W_{T-2}, an Euler equation, and an ending condition $W_{T+1} = 0$.[6]

But how do we solve for these objects V and ψ in the infinite horizon in which we do not have an ending condition? The answer is that we do have an ending condition in the infinite horizon problem, and it is called the transversality condition.
横向

$$\lim_{t\to\infty} \beta^t E_0 \left[W_t u'(W_t)\right] = 0 \tag{12}$$

The transversality condition simply states that the present value of the state W_t goes to zero far off into the future (ending condition). This ensures that people don't save cake forever, thereby consuming zero in any period. In value function iteration, this is analogous to starting with $V_0 = 0$. All recursive (infinite horizon) problems have a transversality condition in the background.

Exercise 9. Let the maximum size of the cake be $W_{max} = 1$. Approximate the continuum of possible cake sizes by a column vector called W that ranges between a number very close to 0 to 1.[7] Let the number of possible cake values be $N = 100$ so

[6] For a first order difference equation, you just need an initial condition or an ending condition and an Euler equation.

[7] We use a number close to zero rather than zero because the log utility function that we will use in the rest of the problems is undefined at zero.

that the increment between each value is 0.01. So $W_{min} = 0.01$.

Exercise 10. As in the previous problem sets, assume that period T is the final period of an individual's life. So $V_{T+1}(W')$ for entering period $T+1$ with a cake of size W' is a column vector of zeros of length N, where $V_{T+1} : \mathbb{R}^N \to \mathbb{R}^N$ and $W, W' \in [W_{min}, W_{max}]$. Assume that the period utility function has the logarithmic functional form $u(c) = \ln(c)$, and that the discount factor is $\beta = 0.9$. What is the resulting policy function $W' = \psi_T(W)$ and value function $V_T(W)$ when V_T is defined as the contraction in equations (9) and (10)?[8] See Appendix 9 for a fast way to compute this exercise.

Exercise 11. Generate a norm $\delta_T = \|V_T(W) - V_{T+1}(W')\|$ that measures the distance between the two value functions. Define the distance metric as the sum of the squared differences,

$$\delta_T \equiv \|V_T(W) - V_{T+1}(W')\| = (V_T - V_{T+1})' * (V_T - V_{T+1}) \tag{13}$$

where $(V_T - V_{T+1})'$ is the transpose of the difference of the two vectors. Defined in this way, $\delta_T \in \mathbb{R}_+$.

Exercise 12. Take the resulting V_T from Exercise 10, and perform the same contraction on it to generate V_{T-1} and ψ_{T-1}. That is, generate,

$$V_{T-1}(W) = C\Big(V_T(W)\Big) = \max_{W' \in [0,W]} u(W - W') + \beta V_T(W')$$

and the accompanying policy function $W' = \psi_{T-1}(W)$. Calculate the accompanying distance measure for δ_{T-1} using the formula from (13) with the updated period subscripts. Compare δ_{T-1} with δ_T from Exercise 11.

Exercise 13. Repeat Exercise 12 and generate V_{T-2} and ψ_{T-2} by performing the contraction on V_{T-1}. Compare δ_{T-2} to δ_{T-1} and δ_T.

[8] HINT: The policy function should be a vector of length N of optimal future values of the cake W' given the current value of the cake W, and V_T should be an N-length vector representing the value of entering a period with cake size W.

Exercise 14. Write a while loop in Python that performs the contraction operation from Exercises 10, 12, and 13 iteratively until the distance measure is very small $\delta_{T-s} < 10^{-9}$. How many iterations did it take $(s+1)$? Congratulations, you've just completed your first solution by value function iteration. The distance measure δ_{T-s} being arbitrarily close to zero means you have converged to the fixed point $V_t = V_{t-1} = V$. (For fun, you can show that the policy function converges to the same function regardless of what you put in for your initial policy function value.)

Exercise 15. Using the `matplotlib` library, plot the policy function for the converged problem $W' = \psi_{T-s}(W) = \psi(W)$ which gives the value of the cake tomorrow (y-axis) as a function of the cake today (x-axis).

5 Infinite Horizon, Stochastic, i.i.d.

\epsilon stands for how hungry the individual is

Now assume that the individuals preferences fluctuate each period according to some i.i.d. shock ε. The Bellman equation can be easily rewritten in the following way to incorporate the uncertainty,

$$V(W,\varepsilon) = \max_{W' \in [0,W]} \varepsilon u(W - W') + \beta E_{\varepsilon'}\left[V(W',\varepsilon')\right] \quad \text{where} \quad \varepsilon \sim \mathrm{N}(\mu, \sigma^2)$$

where E is the unconditional expectations operator over all values in the support of ε, μ is the mean of ε, and σ^2 is the variance of ε.

Exercise 16. Approximate the support of ε by generating a row vector of possible values for ε. Let the maximum value be three standard deviations above the mean $\varepsilon_{max} = \mu + 3\sigma$, and let the minimum value be three standard deviations below $\varepsilon_{min} = \mu - 3\sigma$, where $\sigma^2 = 0.25$ and $\mu = 4\sigma$.[9] And let there be $M = 7$ (make m an odd number so the mean is included in the support) equally spaced points in the support so that ε is an M-length row vector. Generate the probability distribution over $\Gamma(\varepsilon)$ such that $\varepsilon \sim N(\mu, \sigma^2)$. Thus, $\Gamma(\varepsilon)$ represents the probability of a particular realization

[9] I set $\mu = 4\sigma$ so that the entire support of ε is positive. This is important because the utility function is any number in the range $u(c) = log(c) \in (-\infty, W]$. A negative shock would give a premium to negative utility.

$\Pr(\varepsilon = \varepsilon_m)$. (Hint: This is essentially the Newton-Cotes method of approximating an integral that did this in your numerical integration labs.)

Exercise 17. As in Exercise 9 from Section 4, assume that the vector of possible cake sizes is W with $N = 100$ equally spaced values between 0.01 and 1. As in Exercise 10 from Section 4, assume a value function $V_{T+1}(W', \varepsilon')$ for entering the period after the last period of life with cake size W' and taste shock realization ε'. This value function will be a matrix with each row corresponding to different values of W' and each column corresponding to different values of ε'. So each element in the matrix is $V_{T+1}(W'_n, \varepsilon'_m)$. Let your initial guess for the value function V_{T+1} be a matrix of zeros with N rows and M columns. Assume that the period utility function has the logarithmic functional form $u(c) = \log(c)$, and that the discount factor is $\beta = 0.9$. What is the resulting policy function $W' = \psi_T(W, \varepsilon)$ and value function $V_T(W, \varepsilon)$ when V_T is defined as in (14) below? See Appendix 10 for a fast way to compute this exercise.

$$V_{t-1}(W, \varepsilon) \equiv C\Big(V_t(W, \varepsilon)\Big) \equiv \max_{W' \in [0, W]} \varepsilon u(W - W') + \beta E_{\varepsilon'}\Big[V_t(W', \varepsilon')\Big] \tag{14}$$

Exercise 18. Generate a norm $\delta_T = \|V_T(W, \varepsilon) - V_{T+1}(W', \varepsilon')\|$ that measures the distance between the two value functions. Define the distance metric as the sum of the squares of each corresponding element in the two value functions,

$$\delta_T \equiv \|V_T(W, \varepsilon) - V_{T+1}(W'\varepsilon')\| \equiv \text{vec}(V_T - V_{T+1})' * \text{vec}(V_T - V_{T+1}) \tag{15}$$

where $\text{vec}(V_T - V_{T+1})'$ is the transpose of the column vectorized version of $V_T - V_{T+1}$. Defined in this way, $\delta_T \in \mathbb{R}_+$.

Exercise 19. Take the resulting V_T from Exercise 17, and perform the same contraction on it to generate V_{T-1} and ψ_{T-1}. That is, generate,

$$V_{T-1}(W, \varepsilon) = C\Big(V_T(W, \varepsilon)\Big) = \max_{W' \in [0, W]} \varepsilon u(W - W') + \beta E_{\varepsilon'}\Big[V_T(W', \varepsilon')\Big]$$

and the accompanying policy function $W' = \psi_{T-1}(W, \varepsilon)$. Calculate the accompany-

ing distance measure for δ_{T-1} using the formula from (15) with the updated period subscripts. Compare δ_{T-1} with δ_T from Exercise 18.

Exercise 20. Repeat Exercise 19 and generate V_{T-2} and ψ_{T-2} by performing the contraction on V_{T-1}. Compare δ_{T-2} to δ_{T-1} and δ_T.

Exercise 21. Write a while loop in Python that performs the contraction operation from Exercises 17, 19, and 20 iteratively until the distance measure is very small $\delta_{T-s} < 10^{-9}$. How many iterations did it take $(s+1)$? Congratulations, you've just completed your first solution to a stochastic problem by value function iteration. The distance measure δ_{T-s} being arbitrarily close to zero means you have converged to the fixed point $V_t = V_{t-1} = V$.

Exercise 22. Use Python's `matplotlib` library to make a 3-D surface plot of the policy function for the converged problem $W' = \psi_{T-s}(W, \varepsilon) = \psi(W, \varepsilon)$ which gives the value of the cake tomorrow (y-axis) as a function of the cake today ($x1$-axis) and the taste shock today ($x2$-axis).

6 Infinite Horizon, Stochastic, AR(1)

Now assume that the taste shock is persistent. Let the persistence be characterized by the following AR(1) process.

$$\varepsilon' = (1-\rho)\mu + \rho\varepsilon + \nu' \quad \text{where} \quad \rho \in (0,1) \quad \text{and} \quad \nu \sim N(0, \sigma^2) \tag{16}$$

Then the Bellman equation becomes the following, in which the only change from the problems in Section 5 is that the expectations operator is now a conditional expectation because of the persistent shock process,

$$V(W, \varepsilon) = \max_{W' \in [0, W]} \varepsilon u(W - W') + \beta E_{\varepsilon'|\varepsilon}\left[V(W', \varepsilon')\right]$$

where ε' is distributed according to (16). Let $\Gamma(\varepsilon'|\varepsilon) = \Pr\left(\varepsilon'_j|\varepsilon_i\right)$ where ε'_j is the shock in the next period and ε_i is the value of the shock in the current period.

Exercise 23. Use the method described by Tauchen and Hussey (1991) to approximate the AR(1) process for ε from (16) as a first order Markov process. The Python function file "tauchenhussey.py" will produce a vector of length M for the support of ε and an $M \times M$ transition matrix $\Gamma(\varepsilon'|\varepsilon) = \Pr\left(\varepsilon'_j|\varepsilon_i\right)$ where each element in row i and column j represents the probability of ε'_j tomorrow given ε_i today. As inputs, let $M = 7$, the mean of the process $\mu = 4\sigma$, $\rho = 1/2$, $\sigma = \sqrt{\sigma^2} = 1/2$, and $basesigma = (0.5 + \frac{\rho}{4})\sigma + (0.5 - \frac{\rho}{4}) * \frac{\sigma}{\sqrt{1-\rho^2}}$.

Exercise 24. As in Exercise 17 from Section 5, assume that the vector of possible cake sizes is W with $N = 100$ equally spaced values between 0.01 and 1 and assume a value function $V_{T+1}(W', \varepsilon')$ for entering the period after the last period of life with cake size W' and taste shock realization ε'. This value function will be a matrix with each row corresponding to different values of W' and each column corresponding to different values of ε'. So each element in the matrix is $V_{T+1}(W'_n, \varepsilon'_m)$. Let your initial guess for the value function V_{T+1} be a matrix of zeros with N rows and M columns. Assume that the period utility function has the logarithmic functional form $u(c) = \log(c)$, and that the discount factor is $\beta = 0.9$. What is the resulting policy function $W' = \psi_T(W, \varepsilon)$ and value function $V_T(W, \varepsilon)$ when V_T is defined as in (17) below? See Appendix 11 for a fast way to compute this exercise.

$$V_{t-1}(W, \varepsilon) \equiv C\Big(V_t(W, \varepsilon)\Big) \equiv \max_{W' \in [0, W]} \varepsilon u(W - W') + \beta E_{\varepsilon'|\varepsilon}\Big[V_t(W', \varepsilon')\Big] \tag{17}$$

Exercise 25. Generate a norm $\delta_T = \|V_T(W, \varepsilon) - V_{T+1}(W', \varepsilon')\|$ that measures the distance between the two value functions. Define the distance metric as the sum of the squares of each corresponding element in the two value functions,

$$\delta_T \equiv \|V_T(W, \varepsilon) - V_{T+1}(W'\varepsilon')\| \equiv \text{vec}(V_T - V_{T+1})' * \text{vec}(V_T - V_{T+1}) \tag{18}$$

where $\text{vec}(V_T - V_{T+1})'$ is the transpose of the column vectorized version of $V_T - V_{T+1}$. Defined in this way, $\delta_T \in \mathbb{R}_+$.

Exercise 26. Take the resulting V_T from Exercise 24, and perform the same contrac-

tion on it to generate V_{T-1} and ψ_{T-1}. That is, generate,

$$V_{T-1}(W,\varepsilon) = C\Big(V_T(W,\varepsilon)\Big) = \max_{W'\in[0,W]} \varepsilon u(W-W') + \beta E_{\varepsilon'|\varepsilon}\Big[V_T(W',\varepsilon')\Big]$$

and the accompanying policy function $W' = \psi_{T-1}(W,\varepsilon)$. Calculate the accompanying distance measure for δ_{T-1} using the formula from (18) with the updated period subscripts. Compare δ_{T-1} with δ_T from Exercise 25.

Exercise 27. Repeat Exercise 26 and generate V_{T-2} and ψ_{T-2} by performing the contraction on V_{T-1}. Compare δ_{T-2} to δ_{T-1} and δ_T.

Exercise 28. Write a `while` loop in Python that performs the contraction operation from Exercises 24, 26, and 27 iteratively until the distance measure is very small $\delta_{T-s} < 10^{-9}$. How many iterations did it take $(s+1)$? Congratulations, you've just completed your first solution to a stochastic AR(1) problem by value function iteration. The distance measure δ_{T-s} being arbitrarily close to zero means you have converged to the fixed point $V_t = V_{t-1} = V$.

Exercise 29. Make a 3-D surface plot of the policy function for the converged problem $W' = \psi_{T-s}(W,\varepsilon) = \psi(W,\varepsilon)$ which gives the value of the cake tomorrow (y-axis) as a function of the cake today ($x1$-axis) and the taste shock today ($x2$-axis).

7 Policy Function Iteration versus Value Function Iteration

The solution method for the recursive models in Sections 4 through 6 is called value function iteration (VFI). It is a nonlinear functional analysis solution technique for finding the functions $V(\cdot)$ and $\psi(\cdot)$ that solve the Bellman equation of the general form (2). As you have seen in the previous exercises, VFI solves for $V(\cdot)$ and $\psi(\cdot)$ by guessing an initial value function V_0 and applying the operator C from (9) to V_0 to create a new value function V_1 until the new value functions do not change. This

fixed point $V_{i+1} \approx V_i$ where $V_{i+1} = C(V_i)$ is the solution to the Bellman equation along with the policy function $\psi(\cdot)$ that corresponds to that final value function.

Policy function iteration (PFI) is another nonlinear solution technique to the general recursive problem (2) that is closely related to VFI. But policy function iteration is almost always faster and more efficient to compute. As described in the previous paragraph, VFI involves guessing an initial value function and iteratively operating on it with C until convergence. PFI guesses an initial policy function ψ_0 and operates on it with an operator D to get a new policy function ψ_1 until the policy function converges $\psi_{i+1} \approx \psi_i$ where $\psi_{i+1} = D(\psi_i)$. PFI differs from VFI in two main ways. First, it solves for the fixed point in the policy function ψ rather than in the value function V. The second difference is that the operator D uses the Euler equation implied by the optimization problem in each period rather than the sum of lifetime utilities implied by the VFI operator C. It is this last characteristic that provides the increase efficiency and computational speedup of the algorithm. Something about using the Euler equations each iteration makes each successive solution $\psi_{i+1} = D(\psi_i)$ get closer to the fixed point solution.

One additional advantage of PFI over VFI is subtle but very important. VFI uses the C operator described in (9) to solve for the fixed point in V, which maximizes the sum of remaining lifetime utilities. VFI solves for the value function and policy function that maximizes lifetime utility. In contrast, PFI uses the Euler equations between each period. The VFI solution can be called the centralized solution or the planner's solution. This VFI solution disregards any frictions or adverse incentives the household might face and solves for a value function and policy function that optimally offsets the underlying frictions. However, in economics, we are often also interested in the decentralized or competitive solution that accounts for the effects of underlying frictions in the economy. Only PFI, which uses the individual Euler equations, gives the decentralized competitive solution.

An example is a model in which households face an income tax. [TODO: flesh out this example.]

PFI is easily motivated following the approach in Section 3 of starting with a

finitely lived agent problem and moving toward an infinitely lived agent problem. Suppose an individual in the cake eating problem lives for T periods. The problem in the final period of this individual's life is given in equation (4).

$$V_T(W_T) \equiv \max_{W_{T+1}} u(W_T - W_{T+1}) \tag{4}$$

As you showed in Exercise 1, the policy function for $W_{T+1} = \psi_T(W_T)$ does not involve an Euler equation. It is just to choose the minimum possible amount of cake to leave for the period after the individual dies $W_{T+1} = \psi_T(W_T) = 0$ for all W_T. Once you know the policy function in the last period of life $\psi_T(W_T)$, you can solve for the value function in the last period of life by just plugging the policy function in to equation (4), thereby getting rid of the max operator. We leave the policy function $\psi_T(W_T)$ in the solution even though it is trivially zero in order illustrate the pattern of PFI.

$$V_T(W_T) = u\Big(W_T - \psi_T(W_T)\Big) \tag{5}$$

Now we can really see how PFI differs from VFI by moving to the second-to-last period of life. The Bellman equation is the following.

$$V_{T-1}(W_{T-1}) \equiv \max_{W_T} u(W_{T-1} - W_T) + \beta V_T(W_T) \tag{7}$$

The policy function for $W_T = \psi_{T-1}(W_{T-1})$ is characterized by taking the derivative of $V_{T-1}(W_{T-1})$ in (7) with respect to W_T and setting it equal to zero. Using the known solution to $V_T(W_T)$ from (5) on the right-hand-side of (7), the dynamic first order condition or Euler equation that characterizes the optimal choice of $W_T = \psi_{T-1}(W_{T-1})$ in the second-to-last period is the following,

$$u'(W_{T-1} - W_T) = \beta u'\Big(W_T - \psi_T(W_T)\Big) \tag{19}$$

where we already know that $\psi_T(W_T) = 0$ for all W_T on the right-hand-side of (19). In other words, in the second-to-last period of life, if the individual knows the size of

the cake at the beginning of that period W_{T-1} and he knows the policy function for the decision in the next period $\psi_T(W_T)$, he can optimally choose the size of the cake to save for the next period $W_T = \psi_{T-1}(W_{T-1})$ using the Euler equation (19).

Note that the Euler equation (19) is a second order difference equation in W. That is, equation (19) has W_{T-1}, W_T, and W_{T+1}. To solve that equation for the optimal choice of W_T, you must also know an initial condition W_{T-1} and the ending condition $W_{T+1} = \psi(W_T)$. So given our policy function $W_{T+1} = \psi_T(W_T)$ and an initial condition W_{T-1}, we can solve for the policy function for the second-to-last period of life $W_T = \psi_{T-1}(W_{T-1})$ from (19). Once we know the policy function $W_T = \psi_{T-1}(W_{T-1})$, we can plug it into (7), thereby getting rid of the max operator, and delivering a solution for the value function $V_{T-1}(W_{T-1})$.

The last step to fully motivate PFI is to show the third-to-last period problem. The Bellman equation is the following.

$$V_{T-2}\left(W_{T-2}\right) \equiv \max_{W_{T-1}} u\left(W_{T-2} - W_{T-1}\right) + \beta V_{T-1}\left(W_{T-1}\right) \tag{20}$$

The derivative of (20) with respect to W_{T-1} set equal to zero starts out a little bit uglier than expected, but then simplifies nicely with a beautiful intuition. Substituting in the known value function $V_{T-1}(W_{T-1})$ starts out as the following

$$V_{T-2}\left(W_{T-2}\right) \equiv \max_{W_{T-1}} u\left(W_{T-2} - W_{T-1}\right) + \beta u\Big(W_{T-1} - W_T\Big) + \beta^2 u\Big(W_T - W_{T+1}\Big) \tag{21}$$

We already know that $W_{T+1} = \psi_T(W_T) = 0$ and that $W_T = \psi_{T-1}(W_{T-1})$ is characterized by the Euler equation (19). Plugging the known policy functions into (21) gives the following.

$$V_{T-2}\left(W_{T-2}\right) \equiv \max_{W_{T-1}} u\left(W_{T-2} - W_{T-1}\right) + \beta u\Big(W_{T-1} - \psi_{T-1}(W_{T-1})\Big) + \beta^2 u\Big(\psi_{T-1}(W_{T-1}) - \psi_T(W_T)\Big) \tag{22}$$

Now we can take the derivative with respect to W_{T-1} and set it equal to zero.

$$u'\left(W_{T-2}-W_{T-1}\right)=\beta u'\Big(W_{T-1}-\psi_{T-1}(W_{T-1})\Big)-\beta u'\Big(W_{T-1}-\psi_{T-1}(W_{T-1})\Big)\frac{\partial\psi_{T-1}(W_{T-1})}{\partial W_{T-1}}$$
$$...+\beta^2 u'\Big(\psi_{T-1}(W_{T-1})-\psi_T(W_T)\Big)\frac{\partial\psi_{T-1}(W_{T-1})}{\partial W_{T-1}} \tag{23}$$

If we factor out from the last two terms of (23) a negative β times the derivative of the policy function ψ_{T-1} with respect to W_{T-1}, the Euler equation becomes the following.

$$u'\left(W_{T-2}-W_{T-1}\right)=\beta u'\Big(W_{T-1}-\psi_{T-1}(W_{T-1})\Big)$$
$$...-\beta\frac{\partial\psi_{T-1}(W_{T-1})}{\partial W_{T-1}}\left[u'\Big(W_{T-1}-\psi_{T-1}(W_{T-1})\Big)-\beta u'\Big(\psi_{T-1}(W_{T-1})-\psi_T(W_T)\Big)\right] \tag{24}$$

Notice that the optimal decision in the second-to-last period of life $\psi_{T-1}(W_{T-1})$ is characterized by the Euler equation in (19). That Euler equation implies that the difference in the brackets in the third term in (24) equals zero. The Euler equation characterizing the period $T-2$ decision simplifies to the following.

$$u'\left(W_{T-2}-W_{T-1}\right)=\beta u'\Big(W_{T-1}-\psi_{T-1}(W_{T-1})\Big) \tag{25}$$

This dropping out of the term in brackets in (24) is the envelope condition. The intuition is that the agent knows that he will make his period $T-1$ decision W_T optimally. For this reason, the effect of W_{T-1} on that decision does not matter for choosing W_{T-1} in period $T-2$. This is the principle of optimality. It is the foundation of the ability to solve a recursive problem by VFI or PFI.

Comparing the period $T-2$ Euler equation (25) to the period $T-1$ Euler equation, it is easy to see that the general formulation of the period t Euler equation is the

following second order difference equation in W_t, W_{t+1}, and W_{t+2},

$$u'\left(W_t - W_{t+1}\right) = \beta u'\Big(W_{t+1} - \psi_{t+1}(W_{t+1})\Big) \tag{26}$$

where $\psi_{t-1}(W_{t-1})$ is the policy function for W_{t+2}, W_t is the initial value of the cake, and W_{t+1} is the amount of cake to be saved for next period.

PFI starts with an initial guess for the policy function $\psi_0(W)$ and then performs the functional operator D on the policy function guess, where the operator D is defined in the following way.

$$\psi_{i+1}(W) = D\Big(\psi_i(W)\Big) = \arg\min_{W'} \left\| u'(W - W') - \beta u'\big(W' - \psi_i(W')\big) \right\| \tag{27}$$

If D is a contraction mapping according to Definition 1, then we can start with any initial guess of the policy function and apply the functional operator D defined in (27) a finite number of times in order to find the fixed point policy function that solves the infinite horizon Bellman equation. This is perfectly analogous to our motivation of VFI starting with a finitely lived agent and moving backward in time.

Exercise 30. Let discretized support of the size of the cake have the same minima and maxima as in Exercise 9, with $W_{max} = 1$, $W_{min} = 0.01$. But let the number of points in the vector be $N = 1,000$ evenly spaced cake sizes between and including the min and the max. Assume that the period utility function is logarithmic $u(c) = \ln(c)$ and that the discount factor is $\beta = 0.9$. Let your initial guess for the policy function be a vector of 100 zeros $\psi_0(W) = 0$ for all W. Notice that this is the same as imposing the last period policy function $\psi_T(W_T)$. Solve for the new policy function $\psi_1(W)$ that you get by using the functional operator D on $\psi_0(W)$. Plot your resulting policy function $\psi_1(W)$.

Exercise 31. Now write a program that starts with $\psi_0(W)$ from Exercise 30 and operates on it with D until the policy function converges to where $\|\psi_{i+1}(W) - \psi_i(W)\| < 10^{-9}$. Plot the resulting policy function. It should be very similar to the one from Exercise 14, except that your policy function from policy function iteration has 10

times more points in it.

Exercise 32. Import Python's `time` library (e.g., `import time as tm`). Time the `while` loop in your policy function iteration. Go back to your code for value function iteration in Exercise 14, set the size of the cake support vector to $N = 1,000$, and time the `while` loop in that code. How many iterations does policy function iteration take to converge? How many iterations does value function iteration take to converge? How much computation time does policy function iteration take to converge? How much computation time does value function iteration take to converge?

8 Discrete Choice (Threshold) Problems

One powerful application of dynamic programming that illustrates its versatility as a dynamic solution method is to models that have both an extensive and intensive margin. Adda and Cooper (2003) refer to these models as discrete choice problems or optimal stopping problems. They are also sometimes called threshold problems because the discrete choice policy function is determined by the state variable being above or below certain threshold values. Examples include models of employment that involve both the choice of whether to work and how much to work, models of firm entry and exit that involve the choice of both whether to produce and how much to produce, and models of marriage that involve the choice of whether to date (get married or keep dating) and how much to date. [TODO: Include Diamond-Mortensen-Pissarides reference.]

In this problem set, we follow a simple version of a standard job search model.[10] Assume that workers are infinitely lived. Let the value of entering a period with most recent wage w, current job offer wage w', and employment status s be given by the following value function,

$$V(w, w', s) = \begin{cases} V^E(w) & \text{if} \quad s = E \\ V^U(w, w') & \text{if} \quad s = U \end{cases} \tag{28}$$

[10] See Rogerson et al. (2005) and Adda and Cooper (2003, pp. 257-263).

where employment status $s = \{E, U\}$ can either be employed or unemployed.

If an individual's job status is employed $s = E$ in a given period, net present value of expected utility is the period utility of consumption plus the discounted expected value of the entering the next period with wage w, job offer wage w'', and employment status s'.

$$V^E(w) = u(w) + \beta E_{w'',s'} V(w, w'', s') \tag{29}$$

The period utility function is $u(w)$, and the argument w implies a simplified budget constraint $c = w$ that abstracts from any ability to borrow or save. The discount factor is β, the expectations operator $E_{w'',s'}$ is over the job offer wage, and employment status in the next period, and next period's value function is simply (28) with the future value of employment status s'.

The joint probability distribution over w'' and s' is characterized in the following simple way. If the worker stays employed in the next period $s' = E$, then next period's wage equals the current period's wage. If the worker becomes unemployed in the next period $s' = U$, then the worker's unemployment benefits will be a percentage of his current wage αw. Any worker who is unemployed will receive one wage offer per period w', which that worker will receive in the following period, drawn from the cumulative density function $F(w')$ or probability density function $f(w')$, which is independent of the worker's previous wage (for simplicity). Lastly, let γ represent the probability that an employed worker becomes unemployed in the next period. So (29) can be rewritten in the following way.

$$V^E(w) = u(w) + \beta\Big[(1-\gamma)V^E(w) + \gamma E_{w''} V^U(w, w'')\Big] \tag{30}$$

The value of being unemployed in a given period is a function of both the wage at the most recent job w as well as the wage of the current job offer w',

$$V^U(w, w') = u(\alpha w) + \beta \max_{s' \in \{E,U\}} \Big\{ V^E(w'), E_{w''}\left[V^U(w, w'')\right] \Big\} \tag{31}$$

where $\alpha \in (0, 1)$ is the fraction of the worker's previous wage paid in unemployment

insurance benefits. It is only in the unemployed state $s = U$ in which the worker makes a decision. Once the job offer is received w' which is drawn from the independent cumulative probability distribution $F(w')$ or the probability density function $f(w')$, the worker can choose whether to accept or reject the offer. The expectation in (31) is, therefore, not over w' but over the possible job offers in the following period w'' if the worker chooses to reject the current job offer $s' = U$.

The policy function for the decision of the unemployed worker whether to accept a job $s' = E$ or whether to reject a job $s' = U$ will be a function of both the amount of the most recent wage w and the amount the the current job offer: $s' = \psi(w, w')$. These discrete choice problems are often called threshold problems because the policy choice depends on whether the state variable is greater than or less than some threshold level. In the labor search model, the threshold level is called the "reservation wage" w'_R. The reservation wage w'_R is defined as the wage offer such that the worker is indifferent between accepting the job $s' = E$ and staying unemployed $s' = U$.

$$w'_R \equiv w' : V^E(w') = E_{w''}\left[V^U(w, w'')\right] \tag{32}$$

Note that the reservation wage w'_R is a function of the wage at the most recent job w. The policy function will then take the form of accepting the job if $w' \geq w'_R$ or rejecting the job offer and stay unemployed if $w' < w'_R$.

$$s' = \psi(w, w') = \begin{cases} E & \text{if} \quad w' \geq w'_R \\ U & \text{if} \quad w' < w'_R \end{cases} \tag{33}$$

In summary, the labor search discrete choice problem is characterized by the value functions (28), (30), and (31), the reservation wage (32), and the policy function (33). Because wage offers are distributed according to the cdf $F(w')$ and because the policy function takes the form of (33), the probability that the unemployed worker receives a wage offer that he will reject is $F(w'_R)$ and the probability that he receives a wage offer that he will accept is $1 - F(w'_R)$. Just like the continuous choice cake eating problems in problem sets 1 through 5, this problem can be solved by value function

iteration, which is similar to starting at the "final" period of an individual's life and solving for the an infinite series of solutions by backward induction.

1. Assume that workers only live a finite number of periods T and assume that the utility of consumption is log utility $u(c) = \log(c)$. The value of entering the last period of life with most recent wage w and employment status s is the following.

$$V_T(w, w', s) = \begin{cases} V_T^E(w) = \log(w) & \text{if} \quad s = E \\ V_T^U(w, w') = \log(\alpha w) & \text{if} \quad s = U \end{cases}$$

   Solve analytically for the value of entering the second-to-last period of life with most recent wage, current job offer, and employment status $V_{T-1}(w, w', s)$ (which includes $V_{T-1}^E(w)$ and $V_{T-1}^U(w, w')$), the reservation wage $w'_{R,T-1}$, and the policy function $s' = \psi_{T-1}(w, w')$.

2. Given the solutions for the V_{T-1}, $w'_{R,T-1}$, and $s' = \psi_{T-1}(w')$ from the previous exercise, solve analytically for the value of entering the third-to-last period of life with most recent wage, current job offer, and employment status $V_{T-2}(w, w', s)$ (which includes $V_{T-2}^E(w)$ and $V_{T-2}^U(w, w')$), the reservation wage $w'_{R,T-2}$, and the policy function $s' = \psi_{T-2}(w, w')$. [NOTE: This operation of solving for the new value function $V_t(w, s)$ is a contraction.]

The value function iteration solution method for the equilibrium in the labor search problem is analogous to the value function iteration we did in problem sets 3, 4, and 5. The only difference is that two value functions must converge to a fixed point in this problem instead of just one value function converging in the previous problems.

For the following exercises, you will use Python. Assume that the probability of becoming unemployed in a given period is $\gamma = 0.10$, the fraction of wages paid in unemployment benefits is $\alpha = 0.5$, and the discount factor is $\beta = 0.9$. Assume that wage offers to unemployed workers are distributed lognormally $w' \sim \text{LogN}(\mu, \sigma)$ where $m = 20$ is the mean wage, $v = 400$ is the variance of the wage, μ is the mean

of $\log(w')$ and σ is the standard deviation of $\log(w')$. Denote the cdf of the lognormal distribution as $F(w')$ and the pdf of the distribution as $f(w')$.

[The following exercises require Python.]

3. Approximate the support of $w \in (0, \infty)$ by generating a column vector of possible values for w. Let the maximum value be $w_{max} = 100$, let the minimum value be $w_{min} = 0.2$, and let the number of equally spaced points in the vector be $N = 500$ (an increment value of 0.2). Let the wage of a job offer in any period be lognormally distributed $w' \sim \text{LogN}(\mu, \sigma)$, where $\mu = E[\log(w')]$ and $\sigma = \sqrt{\text{var}[\log(w')]}$. So if the mean job offer wage w' is $m = 20$ and the variance of job offer wages w' is $v = 200$, the the corresponding mean μ and standard deviation σ for the lognormal distribution are $\mu = \log\left(m^2/\sqrt{v + m^2}\right)$ and $\sigma = \sqrt{\log\left((v/m^2) + 1\right)}$. Generate the discrete approximation of the lognormal probability density function $f(w')$ such that $w' \sim \text{LogN}(\mu, \sigma)$. Thus, $f(w')$ represents the probability of a particular realization $\Pr(w' = w_n)$. (Hint: This problem is very easy if you use the MatLab function *discretelognorm* in the function file discretelognorm.py available upon request. You're welcome.)

4. Write Python code that solves for the equilibrium optimal policy function $s' = \psi(w, w')$, the reservation wage w'_R as a function of the current wage w, and the value functions $V^E(w)$ and $V^U(w, w')$ using value function iteration.

5. Plot the equilibrium reservation wage w'_R of the converged problem as a function of the current wage w with the current wage on the x-axis and the reservation wage w'_R on the y-axis. This is the most common way to plot discrete choice policy functions. The reservation wage represents the wage that makes the unemployed worker indifferent between taking a job offer and rejecting it. So any wage above the reservation wage line represents $s' = E$ and any wage below the reservation wage line represents $s' = U$.

9 Computation of the value function and policy function using discretized state and control space with perfect foresight

In Exercise 10 from Section 4, the finite horizon Bellman equation is:

$$V_T(W) = \max_{W' \in [W_{min}, W_{max}]} u(W - W') + \beta V_{T+1}(W') \tag{34}$$

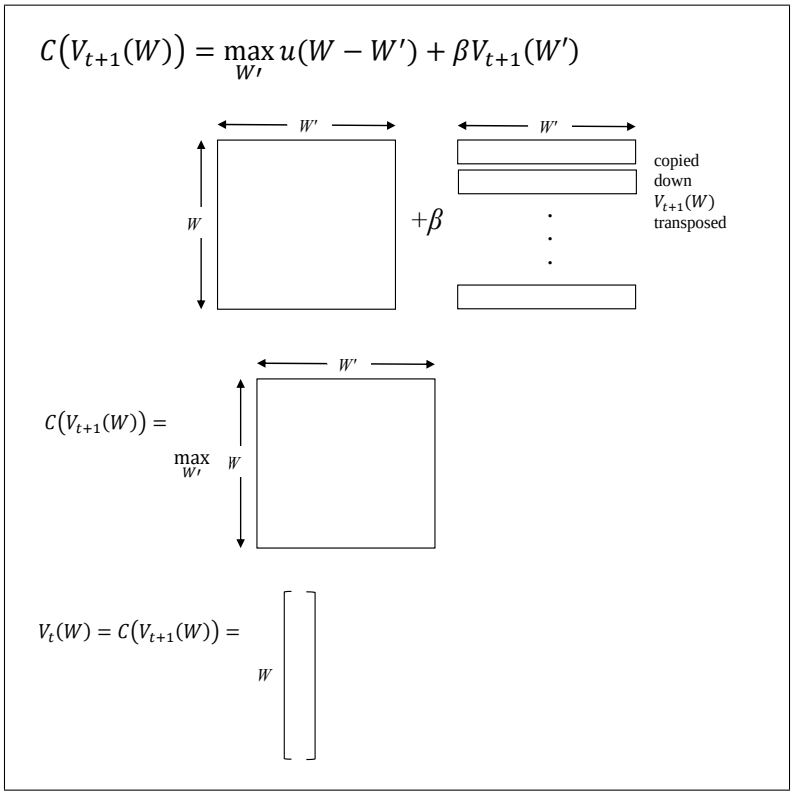
A simple approach to take in calculating these kinds of problems is to put it into the computer without the $\max_{W'}$ operator using the entire set of utility values and value function values for each possible point in the state space W and each possible point in the control space W'. In this case, I put values associated with W on the rows (1st dimension) and values associated with W' in the columns (2nd dimension). The computational geometry is shown below.

It is straightforward to calculate $u(W - W')$ for every value of W and W'. You make sure the constraint of $c > 0$ is satisfied by replacing the negative entries in $W - W'$ with a number that is very close to zero (e.g., 10^{-10}) because negative entries do not satisfy the constraint that $W' < W$. I also then replace any entries in the value function $V_{t+1}(W')$ that correspond to negative $W - W'$ with a very big negative number (e.g., -10^{10}).

The $V_{T+1}(W')$ function is a column vector (similar to V_T). But in (34) it is only a function of W' which is measured along the column dimension (2nd dimension). So I simply take the transpose of V_{T+1} so that it is a row vector and then copy it down n rows. This copying represents the fact that V_{T+1} is not a function of W. As mentioned in the previous paragraph, you'll need to replace all the entries of the V_{T+1} matrix that correspond to values for which $W' > W$ with a very large negative number (e.g., -10^{10} or even -1000) so that those values of W' will not be picked in the maximization.

Now you just add your $u(W - W')$ matrix to your $N \times N$ βV_{T+1} matrix and you

Figure 1: Computational geometry of perfect foresight value function iteration



have an $N \times N$ matrix $V_T(W, W')$ representing the period-T value function for any W and any W'. The last step is to maximize over the W' dimension (2nd dimension). The policy function $\psi_T(W)$ will be an $N \times 1$ column vector that represents the W' value that maximizes the value function for a given W.

10 Computation of the value function and policy function using discretized state and control space with i.i.d. shock

In Exercise 17 from Section 5, the finite horizon Bellman equation is:

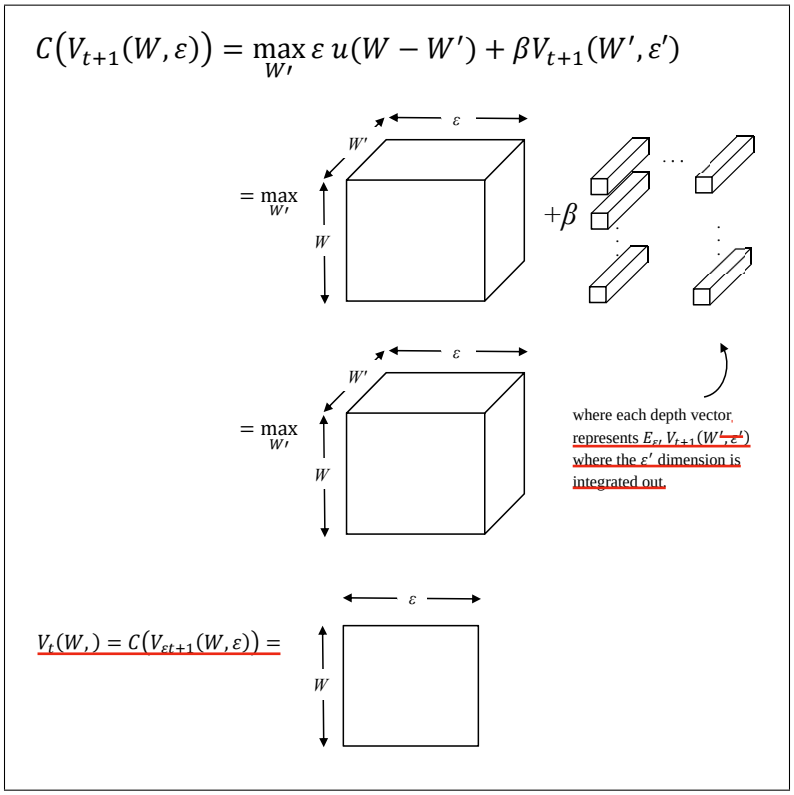
$$V_T(W,\varepsilon) = \max_{W' \in [W_{min}, W]} \varepsilon u(W - W') + \beta E_{\varepsilon'} V_{T+1}(W', \varepsilon') \tag{1}$$

The approach I take in calculating these kinds of problems is to put it into the computer without the $\max_{W'}$ operator using the entire set of utility values and value function values for each possible point in the state space (W, ε) and each possible point in the control space W'. In this case, I put values associated with W on the rows (1st dimension), the values associated with ε on the columns (2nd dimension), and values associated with W' in the depth (3rd dimension). The computational geometry is shown below.

It is straightforward to calculate $\varepsilon u(W - W')$ for every value of W, ε, and W'. I simplify this process by replacing the negative entries in $W - W'$ with a number that is very close to zero (e.g., 10^{-10}) because negative entries do not satisfy the constraint that $W' \leq W$. I also then replace any entries in the value function $V_{t+1}(W', \varepsilon')$ that correspond to negative $W - W'$ with a very big negative number (e.g., -10^{10}).

The $V_{T+1}(W', \varepsilon')$ function is an $N \times M$ matrix (similar to V_T). But in (1) it is only a function of W' because the expectations operator integrates out the ε' dimension, and W' is measured along the depth dimension (3rd dimension). So I simply reshape the vector $E_{\varepsilon'} V_{T+1}$ so that it is $(1 \times 1 \times N)$-dimensional array and then copy it down N rows and M columns. This copying represents the fact that V_{T+1} is not a function of W or ε. As in the previous paragraph, you'll need to replace all the entries of the V_{T+1} matrix that correspond to values for which $W' > W$ with a very large negative number (e.g., -10^{10} or even -100) so that those values of W' will not be picked in the maximization.

Figure 2: Computational geometry of stochastic i.i.d. value function iteration



Now you just add your $\varepsilon u(W - W')$ array to your $N \times M \times N$ $\beta E_{\varepsilon'} V_{T+1}$ array, and you have an $N \times M \times N$ matrix $V_T(W, \varepsilon, W')$ representing the period-T value function for any W, ε, and W'. The last step is to maximize over the W' dimension. The policy function $\psi_T(W)$ will be an $n \times M$ matrix that represents the W' value that maximizes the value function for a given W and ε.

Here is a summary of the process.

1. Take the expectation of $E_{\varepsilon'}[V_{T+1}(W', \varepsilon')]$ by integrating out the ε' dimension of the value function using the probability distribution of the taste shock $\Gamma(\varepsilon)$ from Exercise 16. Do this even though the answer is trivially a vector of zeros for the case in Exercise 17. It will not be trivial in future cases. So $E_{\varepsilon'}[V_{T+1}(W', \varepsilon')]$ becomes a column vector of length N that is only a function of W'.

2. Then change the shape of $E_{\varepsilon'}[V_{T+1}(W', \varepsilon')]$ so that it is a $1 \times 1 \times N$-dimensional

array.

3. Then copy the reshaped $E_{\varepsilon'}\left[V_{T+1}\left(W', \varepsilon'\right)\right]$ to N rows and M columns so that you have an array that has dimension $N \times M \times N$ that is only a function of W' which is represented in the third dimension of the array.

4. Then create an array that represents all the possible values of $\varepsilon u\left(W - W'\right)$ in which the (row, column, depth) dimensions correspond to the values of (W, ε, W').

5. Lastly, the new value function is obtained by adding the two three-dimensional arrays together (multiplying the second array by the discount factor) and maximizing over the third dimension W'. A *max* command along the 3rd dimension of the array can return a matrix of index numbers that represent the optimal value of W', from which you can create the policy function matrix $\psi\left(W, \varepsilon\right)$.

11 Computation of the value function and policy function using discretized state and control space with AR(1) shock

In Exercise 24 from Section 6, the finite horizon Bellman equation is:

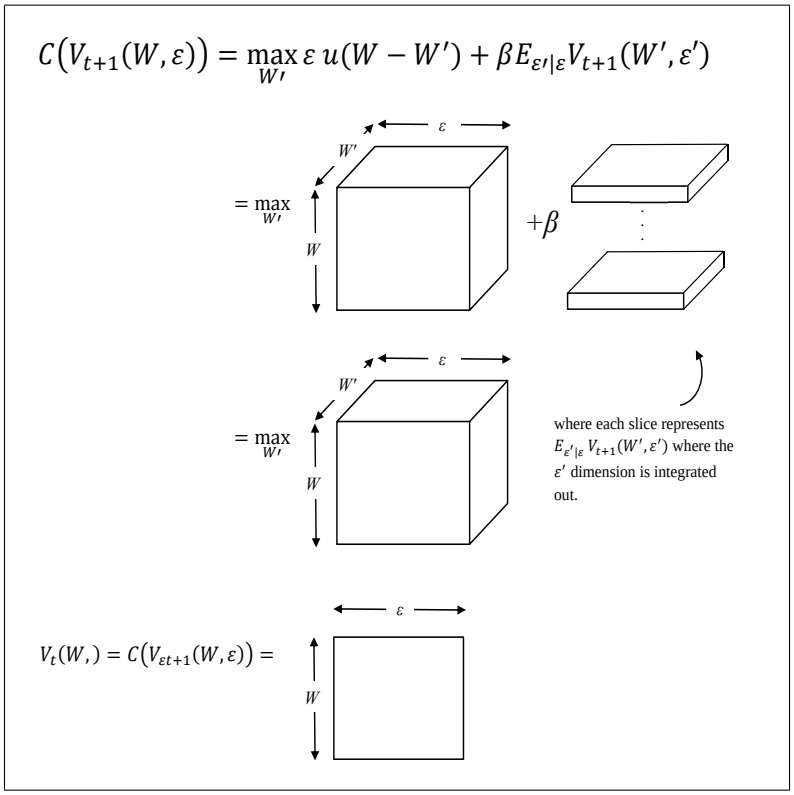
$$V_T(W,\varepsilon) = \max_{W' \in [W_{min}, W]} \varepsilon u(W - W') + \beta E_{\varepsilon'|\varepsilon} V_{T+1}(W', \varepsilon') \tag{1}$$

A simple approach to take in calculating these kinds of problems is to put it into the computer without the $\max_{W'}$ operator using the entire set of utility values and value function values for each possible point in the state space (W, ε) and each possible point in the control space W'. In this case, I put values associated with W on the rows (1st dimension), the values associated with ε on the columns (2nd dimension), and values associated with W' in the depth (3rd dimension). The computational geometry is shown below.

It is straightforward to calculate $\varepsilon u(W - W')$ for every value of W, ε, and W'. I simplify this process by replacing the negative entries in $W - W'$ with a number that is very close to zero (e.g., 10^{-10}) because negative entries do not satisfy the constraint that $W' \leq W$. I also then replace any entries in the value function $V_{t+1}(W', \varepsilon')$ that correspond to negative $W - W'$ with a very big negative number (e.g., -10^{10}).

The $V_{T+1}(W', \varepsilon')$ function is an $N \times M$ matrix (similar to V_T). But in (1) it is rather a function of W' and ε because of the conditional expectations operator. The ε' dimension can be integrated out by matrix multiplying $V_{T+1}(W', \varepsilon')\,\Gamma(\varepsilon'|\varepsilon)'$ where values of W' correspond to the rows (1st dimension) and values of ε correspond to the columns (2nd dimension). The matrix must then be reshaped into a $1 \times M \times N$-array so that values of W' correspond to the depth of the array (3rd dimension). If the original $N \times M$ matrix were called "VTp1", it could be reshaped in MatLab in the way described by writing the following code: "EVTp1array = reshape(VTp1′, [1, M, N])" where you make sure to use the transpose "VTp1′" in the reshape command.

Figure 3: Computational geometry of stochastic AR(1) value function iteration



Then copy the $1 \times M \times N$-array $E_{\varepsilon'|\varepsilon} V_{T+1}(W', \varepsilon')$ down N rows. This copying can be easily done using the "repmat" command in MatLab and represents the fact that V_{T+1} is not a function of W. As with the $\varepsilon u(W - W')$ array, you will need to replace all the entries of the V_{T+1} matrix that correspond to values for which $W' \geq W$ with a very large negative number (e.g., -10^{10} or even -100) so that those values of W' will not be picked in the maximization.

Now you just add your $\varepsilon u(W - W')$ array to your $N \times M \times N$ $\beta E_{\varepsilon'|\varepsilon} V_{T+1}$ array, and you have an $N \times M \times N$ matrix $V_T(W, \varepsilon, W')$ representing the period-T value function for any W, ε, and W'. The last step is to maximize over the W' dimension. The policy function $\psi_T(W)$ will be an $N \times M$ matrix that represents the W' value that maximizes the value function for a given W and ε.

Here is a summary of the process.

1. Take the expectation of $E_{\varepsilon'|\varepsilon}[V_{T+1}(W', \varepsilon')]$ by integrating out the ε' dimen-

sion of the value function using Markov transition matrix for the taste shock $\Gamma(\varepsilon'|\varepsilon)$ from Exercise 23. Do this even though the answer is trivially a vector of zeros for the case in Exercise 24. It will not be trivial in future cases. So $E_{\varepsilon'|\varepsilon}\left[V_{T+1}\left(W',\varepsilon'\right)\right]$ becomes an $N \times M$ matrix that is now a function of W' and ε.

2. Then change the shape of $E_{\varepsilon'|\varepsilon}\left[V_{T+1}\left(W',\varepsilon'\right)\right]$ so that it is a $1 \times M \times N$-dimensional array.

3. Then copy the reshaped $E_{\varepsilon'|\varepsilon}\left[V_{T+1}\left(W',\varepsilon'\right)\right]$ to N rows so that you have an array that has dimension $N \times M \times N$ that is only a function of W' (represented in the third dimension of the array) and ε (represented on the second dimension of the array).

4. Then create an array that represents all the possible values of $\varepsilon u\left(W - W'\right)$ in which the (row, column, depth) dimensions correspond to the values of (W, ε, W').

5. Lastly, the new value function is obtained by adding the two three-dimensional arrays together (multiplying the second array by the discount factor) and maximizing over the third dimension W'. A *max* command along the 3rd dimension of the array can return a matrix of index numbers that represent the optimal value of W', from which you can create the policy function matrix $\psi\left(W,\varepsilon\right)$.

References

Adda, Jérôme and Russell Cooper, *Dynamic Economics: Quantitative Methods and Applications*, MIT Press, 2003.

Blackwell, David, "Discounted Dynamic Programming," *Annals of Mathematical Statistics*, February 1965, *36* (1), 226–235.

Dreyfus, Stuart, "Richard Bellman on the Birth of Dynamic Programming," *Operations Research*, January-February 2002, *50* (1), 48–51.

Rogerson, Richard, Robert Shimer, and Randall Wright, "Search-Theoretic Models of the Labor Market: A Survey," *Journal of Economic Literature*, December 2005, *43* (4), 959–988.

Stokey, Nancy L., Edward C. Prescott, and Robert E. Lucas, *Recursive methods in economic dynamics*, Cambridge, Massachusetts: Harvard University Press, 1989.

Tauchen, George and Robert Hussey, "Quadrature-based Methods for Obtaining Approximate Solutions to Nonlinear Asset Pricing Models," *Econometrica*, March 1991, *59* (2), pp. 371–396.