

1

Basic Analysis of Forward-Looking Decision Making

Individuals and families make the key decisions that determine the future of the economy. The decisions involve balancing current sacrifice against future benefits. People decide how much to invest in health care, exercise, and good diet, and so determine their longevity and future satisfaction. They make choices about jobs that determine employment and unemployment levels. Their investment decisions are at the heart of some issues of financial stability.

1.1 The Dynamic Program

Economists have gravitated to the dynamic program as the workhorse model of the way that people balance the present against the future. The dynamic program is one of the two tools economists tend to reach for when solving problems of optimization over time. The other is the Euler equation. The two approaches are perfectly equivalent: if a problem is susceptible to solution by a dynamic program, it is susceptible to an Euler equation solution, and vice versa. The class

of models suited to either method have the property that the trade-off between this year and next year—the marginal rate of substitution—depends only on consumption this year and next year. Utility functions with this property are *additively separable*. They have the form,

$$\sum_t \beta^t u(x_t), \quad (1.1)$$

where x_t is a vector of things the family cares about.

In words, the idea of a dynamic program is to summarize the entire future in a *value function*, which shows how much lifetime utility the family will enjoy starting next year based on the resources for future spending left after this year. People make choices about this year by balancing the immediate marginal benefits from using resources this year against the marginal value of the remaining resources as of next year, according to the value function. A dynamic program uses backward recursion. Start with a known or assumed value function for some distant year. Find the value function in the previous year by solving the year-to-year balancing problem for all possible levels of resources in that year. Keep moving backward in time until you reach a year from now. Finally, solve this year's optimization by taking actual resources and solving the balancing problem for this year's utility and next year's value function. The dynamic programming approach is conceptually simple and numerically robust. A famous book by Stokey and Lucas (1989) helped persuade economists of the virtues of dynamic programming for recursive problems.

The Euler equation approach considers a possible choice this year and then uses the marginal rate of substitution to map it into a choice next year. Apply

this as a forward recursion, to see where the immediate choice leads to at some distant future date. Again, as in the dynamic program, you need to have some concept of a distant target—for example, the family cannot be deeply in debt at that time. Keep trying out initial choices to find the one where the family meets its distant target. Though this approach has some appeal in explaining dynamic models, it fails completely as a way to solve models numerically. The Euler equation is numerically unstable. Good methods exist for dealing with the instability, but are rarely used in modern dynamic economics because the dynamic program approach works so well. The instability of the Euler equation creates conceptual confusion as well, because one might fall into the trap of thinking that the aberrant paths that do not reach the target might describe actual behavior.

All the models in this book rest on dynamic programming. At the beginning of the year, the family inherits some *state variables* as a result of choices made last year and events during that year. These could be wealth, health status, employment status, or debt. The family picks values of *choice variables*: consumption, health spending, job search, or borrowing. A *law of motion* shows how the inherited values of the state variables, the values of the choice variables, and events during the year map into next year's values of the state variables. The events include financial returns, health outcomes, job loss, and fluctuations in earnings.

The *Bellman equation* describes the condition for recursive optimization:

$$V_t(s_t) = \max_{x_t} (u_t(s_t, x_t) + \beta \mathbb{E}_t V_{t+1}(s_{t+1})). \quad (1.2)$$

Here $V(s)$ is the value function, which assigns lifetime expected utility based on the state variables, s . The choice variables are in the vector x_t . The function $u(s, x)$ is the *period utility* that describes the flow of satisfaction that the family receives when choosing x given s . The general law of motion is

$$s_{t+1} = f_t(s_t, x_t, \epsilon_t). \quad (1.3)$$

Here ϵ_t describes the random events that occur during year t . Anything about year t that is known in advance can be built into the function f_t .

As a simple example, consider the standard lifecycle saving model. The single state variable is wealth, W_t . The only choice variable is current consumption, c_t . The Bellman equation is

$$U_t(W_t) = \max_{c_t} (u(c_t) + \beta \mathbb{E}_{\epsilon_t} U_{t+1}(W_{t+1})) \quad (1.4)$$

and the law of motion for wealth is

$$W_{t+1} = (1 + r_t)(W_t - c_t + y_t + \epsilon_t). \quad (1.5)$$

Here r_t is the known, time-varying return to savings, y_t is the known part of income, and ϵ_t is the random part of income. Although it would be easy to write down the first-order condition for the maximization in the Bellman equation, the condition usually does not add much to understanding. The best approach is often just to leave the maximization to software (Matlab).

As I noted earlier, to solve for the family's optimal choice this year, x_t , start with a known value function in some distant year T , $V_T(s_T)$, iterate backwards to find V_{t+1} , and then solve the Bellman equation for the optimal x_t . Sometimes (but never in this book), the value functions have known functional forms. For

the great majority of interesting problems, the functions need to be represented as well-chosen approximations. Judd (1998) discusses this topic at an advanced level. The state variables may be discrete or continuous. Discrete variables might tell whether a person was employed or unemployed, or well or sick. Discrete variables may also describe random outside events, even some, such as financial returns, that might be considered continuous. Endogenous state variables such as wealth, that result from choices that are continuous, need to be treated as continuous. One should avoid the temptation to convert them to discrete variables, as the resulting approximation is hard to manage and gives unreliable results.

Handling discrete state variables is straightforward: just subscript the value function by the discrete state. Thus the Bellman equation when s_t is discrete is

$$V_{s_t, t} = \max_{x_t} (u_t(s_t, x_t) + \beta \mathbb{E}_t V_{s_{t+1}, t+1}). \quad (1.6)$$

1.2 Approximation

Many interesting models have only a single continuous state variable, including all the models in this book. A useful family of approximations in that case is a weighted sum of known functions of s . Let $\phi_i(s)$ denote these known functions—typically there are several hundred of them. It is convenient to normalize them so that they equal 1 at a given value s_i and 0 at the points corresponding to other of the functions:

$$\phi_i(\tilde{s}_i) = 1 \quad \text{and} \quad \phi_j(\tilde{s}_i) = 0, \quad j \neq i. \quad (1.7)$$

The approximation is

$$V_t(s) = \sum_i \phi_i(s) V_{i,t}. \quad (1.8)$$

Under the normalization of the ϕ s, the value $V_{i,t}$ is the value of $V_t(s)$ at $s = s_i$. The approximation interpolates between the $V_{i,t}$ points for intermediate values of s . The point defined by an s_i and V_i pair is called a *knot*.

A convenient choice for the interpolation functions is a tent:

$$\begin{aligned} \phi_i(s) &= 0 && \text{if } s \leq s_{i-1} \\ &= \frac{s - s_{i-1}}{s_i - s_{i-1}} && \text{if } s_{i-1} < s \leq s_i \\ &= \frac{s_{i+1} - s}{s_{i+1} - s_i} && \text{if } s_i < s \leq s_{i+1} \\ &= 0 && \text{if } s \geq s_{i+1}. \end{aligned} \quad (1.9)$$

The function $V_t(s)$ is then the linear interpolation between the knots.

Given a set of interpolation functions, the backward recursion to find the current knot values is

$$V_{i,t} = \max_{x_t} (u_t(s_i, x_t) + \beta \mathbb{E}_{\epsilon_t} U_{t+1}(f_t(s_i, x_t, \epsilon_t))) \quad (1.10)$$

For the models in this book, with twenty years of backward recursion and 500 knots in the approximation to the value function, a standard PC, vintage 2008, takes around half a minute to calculate the value functions.

When calculating the value functions, it is usually a good idea to store away the choice functions, represented as values $x_{i,t}$ of the optimal choice at time t given state variable value s_i (these are also called *policy functions*).

1.3 Stationary Case

Sometimes the stationary value function is interesting. Suppose that the decision maker is embedded

1.4. Markov Representation

in an unchanging environment with the random ϵ s drawn from an unchanging distribution. Suppose further that the horizon is infinite. Then the value function becomes stationary, in the sense that it loses its time subscript. The stationary approximating Bellman equation is

$$V_i = \max_x (u(s_i, x) + \beta \mathbb{E}_{\epsilon} V(f(s_i, x, \epsilon))). \quad (1.11)$$

To find the values V_i that solve this equation, we can treat the Bellman equation as a big system of nonlinear equations to be solved for the unknowns, V_i . This method is called *projection* and can be tricky, but when it works it is usually fast. An alternative, foolproof method is *value function iteration*. Start with arbitrary V_i , put them on the right-hand side of the Bellman equation, and calculate a new set. Iterate this operation to convergence, which is guaranteed (see Judd 1998, p. 412). This approach can be slow for bigger problems (Judd discusses a number of ways to speed it up).

1.4 Markov Representation

The solution to the dynamic program describes the way a decision maker responds to an uncertain environment. From the stochastic driving force ϵ , the model generates the decision maker's stochastic response, in the sense of the joint distribution of the endogenous variables of the model. Most researchers describe the joint distribution by simulation. They start at given values of the state variables s , evaluate the choice functions $x(s)$, draw a random ϵ from the appropriate distribution, compute the new state vector from the law of motion, and continue for many

simulated years. But simulation is extremely inefficient: to drive down the sampling errors from simulation, which cause the joint distribution of the simulated data to differ from the true joint distribution generated by the model, to acceptable low values, you may need to simulate for days. Some (but not all) of the aspects of the joint distribution are available with essentially perfect accuracy by direct calculation rather than simulation.

A recursive model is a Markov process. For given current values of the state variables s , the choice functions and the law of motion generate a probability distribution across states in the coming year. If the model is stationary, the Markov process has constant probabilities; otherwise, they vary with time. A Markov process is fully defined by its transition matrix. Interesting aspects of the joint distribution can be calculated by standard matrix operations applied to the matrix. For example, transition probabilities over more than one year are powers of the transition matrix and the stationary probabilities of a stationary model can be calculated in no time by matrix inversion.

For a continuous state variable, the true transition matrix is infinitely big, so again we need to use an approximation. I treat the model as assuming that the state variables originate from only the grid of points used in the earlier approximation, s_i . Then I calculate the transition probability from state s_i this year to s'_i next year as the probability that a person starting from the exact point s_i this year winds up in an interval containing s'_i next year. The interval runs from halfway between s'_{i-1} and s'_i to halfway between s'_i and s'_{i+1} . I denote the transition probability as $\Pi_{i,i'}$ and calculate

it as

$$\Pi_{i,i'} = \text{Prob} \left[\frac{s'_{i-1} + \bar{s}'_i}{2} \leq f(s_i, x_i, \epsilon) < \frac{s'_i + s'_{i+1}}{2} \right]. \quad (1.12)$$

Solve the linear system $\pi \Pi = \pi$ and $\sum_i \pi_i = 1$ to find stationary probabilities π_i .

1.5 Distribution of the Stochastic Driving Force

The calculation of the Bellman equation requires the evaluation of an expectation over the distribution of the random ϵ . One could imagine assuming a continuous distribution of the disturbance with a known functional form and performing an analytic or numerical integration to form the expectation. But it is rare to know that the disturbance has a particular functional form and often impossible to do the integrative analytically and challenging to do it numerically. It is usually better to use a purely empirical distribution. For example, if the disturbance is productivity, one can take 50 realizations of actual productivity. The integration is replaced by adding up the value function at 50 values and dividing by 50. Chapter 6 takes this approach with 14,000 values of startup companies.