

Economics 719
November 20, 2007

For the remainder of the semester we will discuss structural estimation. Before considering specific papers will cover techniques today.

Dynamic Programming with Discrete Time and Discrete State. You have seen the problem formulation many times. I mention it for completeness but want to focus on estimation problems. I start with the discrete-time and discrete-state representation as it is the simplest problem.

Denote the set of states as $\{1, 2, \dots, S\}$, and the decision space as $\{1, 2, \dots, D\}$ and the constraint set as $\{1, \dots, D(s)\}$. $S, D, D(s)$ are positive integers.

A feasible stationary decision rule δ is an S -dimensional vector, $\delta(s) \in \{1, \dots, D(s)\}$, $s = 1, \dots, S$. Given $\delta(s)$ define a vector $u_\delta \in \mathbb{R}^S$ with the i^{th} component, $u(i, \delta(i))$ and an $S \times S$ transition probability matrix, E_δ , the i, j element:

$$p(i|j, \delta(j)) = \Pr(s_{t+1} = i | s_t = j, d_t = \delta(j)).$$

Then Bellman's equation is

$$\Gamma(V)(s) = \max_{1 \leq d \leq D(s)} \left[u(s, d) + \beta \sum_{s'=1}^S V(s') p(s'|s, d) \right]$$

Let V^* be the value function for the problem. It is the value of following the optimal policy, δ^* . The optimal policy also satisfies the optimality condition:

$$\delta^*(s) \in \arg \max_{\delta} [u(s, \delta) + \beta E_\delta V_\delta] \quad s = 1, \dots, S.$$

Consider the map T that takes next period's value function, V^+ , creates the current value function V and is defined by

$$V(s) = \max_{\delta} \left[u(s, \delta) + \beta \sum_{s'=1}^S V^+(s') p(s'|s, \delta) \right] \equiv (TV^+)(s) \quad s = 1, \dots, S$$

And we can define a selection for today's control given next period's value function

$$\delta(s) \in \arg \max_{\delta} \left[u(s, \delta) + \beta \sum_{s'=1}^S V^+(s') p(s'|s, \delta) \right] \equiv (\mathcal{U}V^+)(s) \quad s = 1, \dots, S$$

These maps have useful interpretations. If V^+ is tomorrow's value function, then $V = TV^+$ is today's value function, and $\mathcal{U}V^+$ is today's policy function. A variety of computation methods are defined in terms of these maps.

Value Function Iteration

The simplest solution procedure is **value function iteration**. Value function iteration computes the sequence,

$$V^{k+1} = TV^k, \quad k = 0, 1, 2, \dots$$

By the contraction mapping theorem, V^k will converge to the infinite horizon value function for any initial guess V^0 . The sequence of control rules, δ^k , defined by $\delta^{k+1} = \mathcal{U}V^k$ will also converge to the optimal control rule.

In practice one picks a convergence criterion and stops when successive iterations improves the value function by less than the criterion.

Policy Function Algorithm This procedure iterates on the policy function. One can either guess at the initial value function, V^0 or make an initial guess on the policy function, δ^1 . Assuming that we have initialized the value function, V^0 then we apply the map \mathcal{U} , $\delta^{k+1} = \mathcal{U}V^k$ for iteration $k = 0$. (See the maps defined earlier.) Calculate the current period return as

$$R^{k+1}(s) = u(s, \delta^{k+1}), \quad s = 1, \dots, S$$

The new value function is

$$V^{k+1} = [I - \beta E_{\delta^{k+1}}]^{-1} R^{k+1}$$

And as before we continue until improvements in the valuation function are less than the required tolerance.

Notice in the third step calculating the value function of following policy δ^{k+1} it is as if we follow the policy for all periods in the future. In contrast, value function iteration assumes that the new policy is used only for one period. Not surprisingly, convergence in policy function algorithm converges in a few steps, many, many fewer than required for value function convergence.

The difference in the computation is in the evaluation of policy step improvement, $V^{k+1} = [I - \beta E_{\delta^{k+1}}]^{-1} R^{k+1}$ the difficulty is in the inversion of the matrix $E_{\delta^{k+1}}$ which is of dimension $S \times S$, for small state spaces this is manageable, but becomes prohibitive as S increases. This is true even using sparse matrix routines. For example, in the work with Kennan our state is for a given age is $(151 \times 153 \times 9 \times 50 = 207,927$, for each of 50 home locations and 40 ages). I think this is the difference between labor and public applications that use value function iteration and many IO applications where policy function iteration is common.

See Judd for additional details.

Continuous States — consumption problems. I am going to focus on discrete action policies, but mention in passing continuous decision variable models. These are common in wealth accumulation in which the decision variable is continuous — how much to save or consume. If decisions are all interior solutions then solutions are characterized by the Euler equation,

$$\begin{aligned} u'(c_t) &= \beta(1+r)E[u'(c_{t+1})] \\ u((1+r)A_t + y_t - A_{t+1}) &= \beta(1+r)E[u'((1+r)A_{t+1} + y_{t+1} - A_{t+2})] \end{aligned}$$

Thus, it is possible to write the problem in terms of the original one-period utility flow and eliminate the value function entirely. This is a second order difference equation. This form of analysis dominates models of savings, life cycle consumption, in both the micro and macro literatures.

It is a workhorse in macro and in many areas of micro economics. For a constructive hands-on of computation see Chris Carroll's web page for his write up of computational approaches (with MatLab code) for life cycle consumption with and without liquidity constraints.

Projection Methods I am not going to discuss projection methods, also popularized by Judd and Sargent and Ljungqvist (See Recursive Macroeconomics, 2nd edition.) Dynamic programming problems are functional in that we are solving for the optimal policy which is a function of the state space; the optimal action given state s , $\delta(s)$. The basic idea in projection methods, we solve for the approximate solution. We know by the Weierstrass theorem, for example, that any continuous function is well-approximated by large sums of polynomial terms. Approximate the function $\delta(s)$ as

$$\delta(s) \approx \hat{\delta}(s; a) = 1 + \sum_{j=1}^n a_j s^j.$$

We find the coefficients \mathbf{a} that “nearly solve” $R(s; a) = V(\hat{\delta}) - T(V)(\hat{\delta})$, where the residual function is the deviation from zero, the target value. We must define a metric or distance function by which to evaluate discrepancies. Applications differ by choice of approximating functions (different sets of orthogonal polynomials) and weighting or distance functions.

These solution methods are elegant and used by several in the department. Project methods are useful once you know a lot about the solution and especially about its domain. I am going to discuss discrete approximation, because of their wide use and simplicity. Discrete approximations always give a solution, and we know how to improve the quality of the solution (i.e., increase the dimension of the state space). The difficulty with discrete approximations is the so-called curse of dimensionality: the number of states is exponentially increasing. If we need N cells for one continuous state variable, we will need N^k for k , and since N needs to be large, the approach becomes infeasible for relatively small k .

Notice in projection methods as described there is no *statistical* estimation. The residual function not identically zero because of approximation error, not sampling error. In the same way there will be an error by using the discrete approximation. For all states within an approximating cell, $sinS_m$ we will define $\hat{\delta}(\bar{s})$, where \bar{s} is the assigned value within the cell. For example, if the approximating cell is an interval, it is common to use set \bar{s} equal to the midpoint. It is clear that as we increase the number of approximating cells, the approximation error will decline. Projection methods improve the fit by increasing the degree of the polynomial.

The point is to understand that errors at this stage are approximation error. There is no approximation if the state space and decision space are each discrete. For example, Wolpin (1984) paper on fertility in the *JPE* has a discrete state space, the number of surviving children. However, once the model admits (financial) assets or human capital the state space is continuous. The discrete approximation is that the continuous state variable is divided into

discrete cells. The value function and optimal policy are calculated then for each discrete state.

Discrete Space. Let there be K possible alternatives. I will assume we face a finite horizon, T . (Infinite horizon problems solved by letting $T \rightarrow \infty$.) Let $d_k(t) = 1$ denote that alternative k selected in time period t . The indicated variables are collectively exhaustive, so $\sum_k d_k(t) = 1$.

Let the current reward function be $R_k(t)$ - it is known at time t but is seen as random for periods prior to t . The lifetime problem is then at time $t = 0, 1, \dots, T$

$$\max E \left[\sum_{t'=t}^T \delta^{t'-t} \sum_{k \in K} R_k(t) d_k(t) | S(t) \right]$$

The expectations use information at time t , δ is the discount factor, and $S(t)$ is the state space at time t (includes all factors known to the individual at t that affects current returns or the probability distribution of future returns). To solve the problem we have to determine the **optimal** sequence of control variables $\{d_k(t); k \in K; t = 0, 1, \dots, T\}$.

Define maximal expected value of the discounted lifetime reward at t as

$$V(s(t), t) = E \left[\sum_{t'=t}^T \delta^{t'-t} \sum_{k=1}^K d_k(t') R_k(t') | S(t) \right]$$

That is,

$$V(s(t), t) = \max_k \{V_k(s(t), t)\}$$

where the V_k is the choice-specific lifetime reward — the expected lifetime utility in state s at time t when the decision rule selects action k .

The choice-specific value functions follow Bellman's equation:

$$\begin{aligned} V_k(s(t), t) &= R_k(s(t), t) + \delta E[V(s(t+1), t+1) | S(t), d_k(t) = 1], \quad t \leq T-1 \\ V_k(S(T), T) &= R_k(S(T), T), \quad t = T \end{aligned}$$

The expectation is taken over the distribution of $S(t+1)$ conditional on the state and action taken at time t , $s(t), d_k(t) = 1$. The conditional density is denoted as

$$p_{kt}(s(t+1) | s(t), d_k(t) = 1)$$

Randomness in rewards arises from state variables at $t+1$ observable to the agent at time $t+1$ but are unobservable at time t and before. Can think of the randomness to income, wages/prices, health or other environmental states [e.g., married]). In this framework think of rewards as $R_k(s^0(t), \epsilon_k(t))$ where, $s^0(t)$ is the deterministic state (e.g., age) and $\epsilon_k(t)$ is the innovation or shock to return k which is uncertain at time $t-1$ but known to the agent at time t .

I will assume that $\{\epsilon_k(t)\}_{k=1}^K$ are **serially uncorrelated**. (Of course that is a simplification; need not be the case.) So the joint distribution is $\prod_{t=0}^T f(\epsilon_{1t}, \dots, \epsilon_{kt}; \eta)$ where η is a vector of parameters defining density f .

Let's take a closer look at the expectation in the choice-specific value functions.

$$E[V(s(t+1), t+1)|s(t), d_k(t) = 1] = E\left[\max_k\{V_k(s(t+1), t)|s(t), d_k(t) = 1\}\right]$$

Thus, at time $T - 1$ must calculate

$$\begin{aligned} & Emax(R_1(T), R_2(T), \dots, R_K(T)|s(T-1), d_k(T-1)) = \\ & = \int_{\epsilon_{KT}} \cdots \int_{\epsilon_{1T}} \max(R_1(T), \dots, R_K(T)|s(T-1), d_k(T-1)) f(\epsilon_{1T}, \dots, \epsilon_{KT}) d\epsilon_{1T} \cdots d\epsilon_{KT} \end{aligned}$$

So that, even if we assume that the ϵ s are serially independent without out additional assumptions have a multivariate integral to evaluate.

Moreover, because conditional distribution and future payoffs depend on current choice, must calculate evaluate the Emax for each choice, or K times for each period.

In the stochastic representation, the agent sees the current ϵ but we do not. So deterministic choices of the agent are stochastic to us. Thus, the policy function $\delta(s)$ are truly probabilities, $\Pr(d_k(t)|s(t))$. It is these **conditional choice probabilities** that summarize the decision process.

Letting the unknown parameters be θ then the likelihood function for a sample of N individuals each observed T periods is:

$$L(\theta) = \prod_{a=1}^N \prod_{t=1}^T \Pr(d_t^a | s_t^a, \theta) p(s_t^a | s_{t-1}^a, d_{t-1}^a, \theta)$$

Evaluation of the *Emax* function and the conditional choice probabilities are the two computationally demanding tasks in structural estimation. And of course, the functions must be evaluated for each trial parameter value θ_ℓ . One way to understand procedures is to understand how they simplify the calculation of these two functions.

It is important to see that structural estimation requires **two** maximization problems. The first, is the solution to the agent's problem; the calculation of the optimal decision rule (for a given θ). Rust calls this the "inner" optimization problem. Then we optimize some statistical objective function to estimate the unknown parameters, θ . Rust labels this as the "outer" optimization problem.

I wrote the objective function as the likelihood function, so that we want to estimate parameters to maximize the sample likelihood. It may be more intuitive to think of minimum distance estimators. We observe the choices made by the individuals (agents) in our sample, $\{d_t^a, a = 1, \dots, N, t = 1, \dots, T\}$ Our structural model determines the conditional choice probabilities of the observed choices, $\Pr(d_t^a | s_t^a, \theta)$. Instead of a ML routine, a minimum distance estimator would be to find parameters θ that minimize the distance between observed choices and predicted probabilities of those choices,

$$Q(\theta) = \sum_{a=1}^N \sum_{t=1}^T [d_t^a - \Pr(d_t^a | s_t^a, \theta)]^2.$$

Two general approaches, one characterized by Wolpin/Keane and the other by Rust. There is less difference now than before as they share and combine methods. But to a large extent, the difference is between probits and logits. Wolpin/Keane typically adopt error structures for

the ϵ that are multivariate normal. They have the flexibility of the multivariate normal. They also have the complexity of evaluating high dimensional integrals. Rust lives in an extreme value world, in which the conditional choice probabilities have a MNL. His approach also yields a closed form expression for the *Emax* term. Common now to combine elements of both.

There are also differences on where the approximation comes in. In a sense all solutions are approximate, unless the action and state spaces are each discrete. Keane and Wolpin advocate an approximation approach, the formal properties to my understanding have not been shown, but which apparently works well in practice. Essentially, they fix a grid over the state space and solve the dynamic program exactly at those points. Then use regression techniques to project the value functions onto the elements of the state vector. This gives a smoothed value of the value function. They use the smooth value or predicted value in their calculations (via interpolation). It is a form of the projection method, though Judd has some critical remarks. See Keane and Wolpin (1994) RE Stat paper.

Key assumption of the Rust paper

- (1) The current payoff function is additively separable.
- (2) Conditional independence. The law of motion of the state variables depends only on the observable state variables and choices made. Conditional on observed state variables, unobservables in $t + 1$ are independent of the unobservables in t . The force of this assumption is that it rules out time persistent unobservables in the ϵ .
- (3) The unobservables epsilons are iid type 1 extreme value random variables.

According to AM(2002), the expectation of a type 1 extreme value random variable is $E(\epsilon(a)|x, a) = \gamma - \ln(\Pr(a|x))$, where γ is Euler's constant.