Single Agent Dynamics: Dynamic Discrete Choice Models

Part I: Theory and Solution Methods

Günter J. Hitsch
The University of Chicago Booth School of Business

2013

Overview: Goals of this lecture

1. Discuss the structure and theory of dynamic discrete choice models
2. Survey numerical solution methods
3. Discuss goals of statistical inference: Reduced vs structural form of the model
4. Model estimation
   - Nested fixed point (NFP) estimators
   - MPEC approach
   - Heterogeneity and unobserved state variables

Overview

Overview

Review: Infinite Horizon, Discounted Markov Decision Processes

The Storable Goods Demand Problem

Dynamic Discrete Choice Models

The Storable Goods Demand Problem (Cont.)

The Durable Goods Adoption Problem

Numerical Solution of a Dynamic Decision Process

- Approximation and Interpolation
- Numerical Integration

The Durable Goods Adoption Problem (Cont.)

References

Elements of a Markov Decision Process

- Time: $t = 0, 1, \ldots$ (infinite horizon)
- A set of states, $s \in \mathbb{S}$
  - $\mathbb{S}$ is the state space
- A set of actions, $a \in \mathcal{A}(s)$
  - Set of feasible actions may depend on the current state, although this is more general than what we typically need in dynamic discrete choice

Elements of a Markov Decision Process

- A decision maker chooses some action $a \in \mathcal{A}(s)$ in period t, given the current state s, and then receives the utility $U(s, a)$
- The state evolves according to a probability distribution $p(\cdot|s, a)$
  - If $\mathbb{S}$ is discrete, then $\Pr\{s_{t+1} = j|s_t, a_t\} = p(j|s_t, a_t)$ for $j \in \mathbb{S}$
  - If $\mathbb{S}$ is continuous, then $p(\cdot|s_t, a_t)$ is the conditional density of s_{t+1}

Elements of a Markov Decision Process

- The decision process is called "Markov" because the utility function and transition probabilities depend only on the current state and actions, not on the whole history of the process
- Utilities, $U(s, a)$, and transition probabilities, $p(\cdot|s, a)$ are *stationary*, i.e., do not depend on the time period t when an action is taken
  - Note: this does not imply that the time series process $\{s_t\}_{t=0}^{\infty}$ is stationary, and vice versa, stationarity of $\{s_t\}_{t=0}^{\infty}$ is not required in this class of models
  - Example: A decision make gets tired of a product as time progresses. Could this be captured by a Markov Decision Process? — Answer: yes
  - $U_t(s, a) = -t$ is not allowed
  - Instead let $\mathbb{S} = \{0, 1, 2, \ldots\}$, and for any $s \in \mathbb{S}$ let $p(s + 1|s, a) = 1$, i.e. $s_{t+1} = s_t + 1$. Then define

$$U(s, a) = -s$$

Actions, decision rules, and policies

- We examine decision problems where the decision maker uses current (and possibly past) information to choose actions in order to maximize some overall objective function
- Decision rules: $d_t : \mathbb{S} \to \bigcup_{s \in \mathbb{S}} \mathcal{A}(s)$, $d_t(s) \in \mathcal{A}(s)$
  - Decision rules associate a current state with a feasible action
- Policies, strategies, or plans: $\pi = (d_0, d_1, \ldots) = (d_t)_{t=0}^{\infty}$
  - A policy is a sequence of decision rules—a plan on how to choose actions at each point in time, given the state at that point in time, s_t

Expected total discounted reward

- A given policy $\pi = (d_t)_{t=0}^{\infty}$ induces a probability distribution on the sequence of states and actions, $(s_t, a_t)_{t=0}^{\infty}$
- For a given policy π, the expected total discounted reward (utility) is given by
$$v^{\pi}(s_0) = \mathbb{E}^{\pi}\left[\sum_{t=0}^{\infty} \beta^t U(s_t, d_t(s_t))|s_0\right]$$
- Future utilities are discounted based on the discount factor $0 \leq \beta < 1$ (in finite horizon problems we could allow for $\beta = 1$)

Expected total discounted reward

- We will assume that v^{π} is well-defined
- One necessary condition if the state space is discrete: The series inside the expectation converges
  - Ensured if U is bounded, because $0 \leq \beta < 1$
  - Note, however, that boundedness is not a necessary condition for convergence
- If the state space is continuous, certain technical conditions need to hold for the expectation to exist (see Stokey and Lucas 1989)

Optimal policies

- Let Π be the set of all policies. $\pi^* \in \Pi$ is optimal if

$$v^{\pi^*}(s) \geq v^{\pi}(s), \qquad s \in \mathbb{S}$$

  for all $\pi \in \Pi$
- The value of the Markov decision problem is given by

$$v^*(s) = \sup_{\pi \in \Pi} \{v^{\pi}(s)\}$$

- We call v^* the (optimal) value function
- An optimal policy π^* is a policy such that

$$v^{\pi^*}(s) = v^*(s) = \max_{\pi \in \Pi} \{v^{\pi}(s)\}$$

Bellman Equation

- Intuition:
  - Due to the stationarity of the utilities and transition probabilities, and given that there is always an infinite decision horizon, the (optimal) value function should only depend on the current state, but not on t
- Hence, for any current state s, the expected total discounted reward under optimal decision making should satisfy the relationship

$$v^*(s) = \sup_{a \in \mathcal{A}(s)} \left\{ U(s,a) + \beta \int p(s'|s,a) v^*(s') ds' \right\}$$

Bellman Equation

- The equation above is called a *Bellman equation*
- Note that the Bellman equation is a functional equation with potential solutions of the form $v : \mathbb{S} \to \mathbb{R}$
- More formally, let $\mathcal{B} = \{f : \mathbb{S} \to \mathbb{R} : \|v\| < \infty\}$ be the metric space of real-valued, bounded functions on $\mathbb{S}$. $\|f\| = \sup_{s \in \mathbb{S}} |f(s)|$ is the supremum-norm. Define the operator $L : \mathcal{B} \to \mathcal{B}$,

$$Lv(s) \equiv \sup_{a \in \mathcal{A}(s)} \left\{ U(s,a) + \beta \int p(s'|s,a) v(s') ds' \right\}$$

- A solution of the Bellman equation is a fixed point of L, $v = Lv$

Questions to be addressed

1. Is a solution of the Bellman equation the optimal value function?
2. Is there a solution of the Bellman equation, and how many solutions exist in general?

The main theorem

Theorem

If v is a solution of the Bellman equation, $v = Lv$, then $v = v^*$. Furthermore, under the model assumptions stated above a (unique) solution of the Bellman equation always exists.

- The proof of this theorem is based on the property that L is a contraction mapping. According to the Banach fixed-point theorem, a contraction mapping (defined on a complete metric space) has a unique fixed point. The contraction mapping property crucially depends on the additive separability of the total discounted reward function, and in particular on discounting, $\beta < 1$

Optimal policies

- Assume that the supremum in the Bellman equation can be attained, and define the decision rule

$$d^*(s) \in \arg\max_{a \in \mathcal{A}(s)} \left\{ U(s,a) + \beta \int p(s'|s,a) v^*(s') \right\}$$

- Let $\pi^* = (d^*, d^*, \dots)$ be a corresponding policy

Theorem

Let π^* be as defined above. Then π^* is an optimal policy, such that $v^{\pi^*} = v^*$

Optimal policies

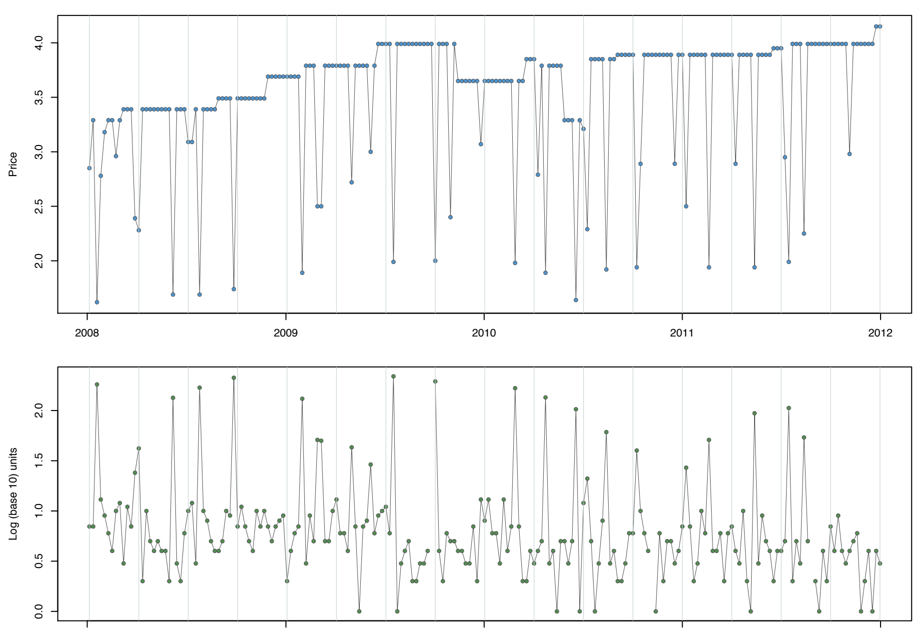
- Note that the theorem on the previous slide shows us how to find an optimal policy
- In particular, note that the theorem shows that if the supremum in the Bellman equation can be attained, then there is a *stationary* optimal policy

Optimal Policies: Notes

- A sufficient condition for the existence of an optimal policy is
  - $\mathcal{A}(s)$ is finite
  - Satisfied (by definition) in dynamic discrete choice problems
- Decision rules that depend on the past history of states and actions do not improve the total expected reward
  - Hence, basing decisions only on current, not past information, is not restrictive in Markov decision processes

Overview

Kellogg's Raisin Bran, 20 oz pack size



Overview: Sales promotions

- Many CPG products are sold at both regular (base) prices and at promotional prices
- We often observe sales spikes during a promotion
  - Could be due to brand switching or increased consumption
  - Could be due to stockpiling if the good is storable
- Develop a simple model of stockpiling—storable goods demand
  - Goal: predict how demand is affected by the distribution of prices at which the storable good can be bought

Model outline

- At the beginning of each period a consumer goes to a store and chooses among K different pack sizes of a storable good
  - Pack size k contains n_k units of the product ($n_k \in \{1, 2, \dots\}$)
  - No purchase: $n_0 = 0$
- The choice in period t is $a_t \in \mathcal{A} = \{0, 1, \dots, K\}$
- i_t is the number of units in the consumer's inventory at the beginning of period t
- After the shopping trip, if $a_t = k$ the consumer has $i_t + n_k$ units of the product at her disposal
- The consumer has a consumption need of one unit in each period
  - Will consume one unit if $i_t + n_k \geq 1$, and will not consume otherwise

Evolution of inventory

- Consumer can store at most I units
- Conditional on i_t and $a_t = k$, inventory evolves as

$$i_{t+1} = \begin{cases} 0 & \text{if } i_t + n_k = 0 \\ i_t + n_k - 1 & \text{if } 1 \leq i_t + n_k \leq I \\ I & \text{if } i_t + n_k > I \end{cases}$$

  - Note: Units in excess of $I + 1$ can neither be consumed nor stored
- The evolution of inventory is deterministic, and we can write $i_{t+1} = \phi(i_t, a_t)$

Utilities and disutilities

- The purchase utility (rather: disutility) from buying pack size k is $-\alpha P_{kt} - \tau(n_k)$
  - P_{kt} is the price of pack size k and α is the price sensitivity parameter
  - $\tau(n_k)$ is the hassle cost of purchasing (or transporting) pack size k
  - Outside option: $P_{0t} = 0$, $\tau(n_0) = \tau(0) = 0$
- The consumption utility, given that there is at least one unit to consume $(i_t + n_k \geq 1)$ is δ, and 0 otherwise
- Inventory holding cost: $c(i_{t+1})$
  - $c(0) = 0$

Notation

- Define $x_t \equiv (i_t, P_t)$, where $P_t = (P_{1t}, \dots, P_{Kt})$
- Define utility as a function of x_t and as a function of the chosen pack size $a_t = k$:

$$u_k(x_t) = \begin{cases} -\alpha P_{kt} - \tau(n_k) + \delta - c(i_{t+1}) & \text{if } k \neq 0 \\ \delta - c(i_{t+1}) & \text{if } k = 0 \text{ and } i_t \geq 1 \\ 0 & \text{if } k = 0 \text{ and } i_t = 0 \end{cases}$$

Distribution of prices

- The price vectors P_t are drawn from a discrete distribution
- P_t can take one of L values $\{P^{(1)}, \dots, P^{(L)}\}$ with probability $\Pr\{P_t = P^{(l)}\} = \pi_l$
- In this specification prices are i.i.d. over time
  - Generalization: Prices be drawn from a Markov process where current prices depend on past price realizations

State vector and utility function

- A random utility term enters the consumer's preferences over each pack size choice
- $\epsilon_t = (\epsilon_{0t}, \ldots, \epsilon_{Kt})$, where ϵ_{kt} is *iid* Type I Extreme Value distributed
  - $g(\epsilon)$ is the pdf of ϵ_t
- The state of this decision process is $s_t = (x_t, \epsilon_t)$
- The (full) utility function of the consumer:

$$U(s_t, a_t = k) = u_k(x_t) + \epsilon_{kt}$$

State evolution

- The evolution of x_t is given by the pmf f :

$$\begin{aligned} f(i, P^{(l)}|x_t, a_t) &\equiv \Pr\{i_{t+1} = i, P_{t+1} = P^{(l)}|x_t, a_t\} \\ &= \mathbb{I}\{i = \phi(i_t, a_t)\} \cdot \pi_l \end{aligned}$$

  - Defined for $0 \leq i \leq I, 1 \leq l \leq L$
- Then the evolution of the state, $s = (x, \epsilon)$, is given by

$$p(s_{t+1}|s_t, a_t) = f(x_{t+1}|x_t, a_t) \cdot g(\epsilon_{t+1})$$

Discussion

- The storable goods demand model has a specific structure due to the way the random utility terms ϵ_{kt} enter the problem:
  - The state has two separate components, x_t and ϵ_t
  - The utility function has the additive form

$$U(s_t, a_t = k) = u_k(x_t) + \epsilon_{kt}$$

  - The transition probability factors into two separate components:

$$p(s_{t+1}|s_t, a_t) = f(x_{t+1}|x_t, a_t) \cdot g(\epsilon_{t+1})$$

- In the next section we will discuss Markov decision processes with this special structure

Overview

Structure of the decision problem

- A decision maker (typically a consumer or household) chooses among $K+1$ alternatives (goods or services), $a \in \mathcal{A} = \{0, 1, \ldots, K\}$, in each period $t = 0, 1, \ldots$
  - Alternative 0 typically denotes the no-purchase option
- The state vector, $s \in \mathbb{S}$, has the following structure:

$$s = (x, \epsilon), \quad \text{where } x \in \mathbb{X},\ \epsilon \in \mathbb{R}^{K+1}$$

- The transition probability is $p(s_{t+1}|s_t, a_t)$

Main assumptions

1. *Additive separability.* The utility from choosing action $a_t = j$ in state $s_t = (x_t, \epsilon_t)$ has the following additive structure:

$$U(s_t, a_t) = u_j(x_t) + \epsilon_{jt}$$

2. *Conditional independence*. Given x_t and a_t, current realizations of ϵ_t do not influence the realizations of future states x_{t+1}. Hence, the transition probability of x can be written as $f(x_{t+1}|x_t, a_t)$
3. *iid* ϵ_{jt}. ϵ_{jt} is *iid* across actions and time periods.
   - $g(\epsilon)$ is the pdf of $\epsilon = (\epsilon_0, \ldots, \epsilon_K)$, and we typically assume that the support of ϵ_j is $\mathbb{R}$
   - We could allow for $g(\epsilon|x)$, although this more general specification is rarely used in practice

Discussion of model structure

- While x can contain both observed and unobserved components, ϵ is assumed to be unobservable to the researcher
  - ϵ_j is an unobserved state variable
- Additive separability and the assumptions on ϵ_j will allow us to rationalize any observed choice, and thus ensure that the likelihood function is positive on the whole parameter space
- Given the conditional independence and *iid* assumptions, the transition probability can be written as follows:

$$p(s_{t+1}|s_t, a_t) = f(x_{t+1}|x_t, a_t) \cdot g(\epsilon_{t+1})$$

Decision rules and rewards

- In each period, the decision maker chooses an action according to a decision rule $d : \mathbb{X} \times \mathbb{R}^{K+1} \to \mathcal{A}$, $a_t = d(x_t, \epsilon_t)$.
- The expected present discounted value of utilities under the policy $\pi = (d, d, \dots)$ is

$$v^\pi(x_0, \epsilon_0) = \mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t U(x_t, \epsilon_t, d(x_t, \epsilon_t))|x_0, \epsilon_0\right].$$

Optimal behavior and the Bellman Equation

- Given that $\mathcal{A}$ is finite, there is an optimal decision rule $d^*(x, \epsilon)$
- The associated value function satisfies

$$\begin{aligned} v(x,\epsilon) &= \max_{j\in\mathcal{A}} \{u_j(x) + \epsilon_j + \beta\mathbb{E}\left[v(x',\epsilon')|x, a=j\right]\} \\ &= \max_{j\in\mathcal{A}} \left\{u_j(x) + \epsilon_j + \beta \int v(x',\epsilon') f(x'|x,j) g(\epsilon')\, dx' d\epsilon'\right\} \end{aligned}$$

The expected value function

- Goal:
  - Characterize optimality condition as a function of x only
  - $\Rightarrow$ faster computation of model solution
- Define the *expected* or *integrated value function*,

$$w(x) = \int v(x,\epsilon) g(\epsilon) d\epsilon$$

- $w(x)$ is the value that the decision maker expects to receive before the unobserved states, ϵ, are realized
- Using the definition of w, we can re-write the value function:

$$v(x,\epsilon) = \max_{j\in\mathcal{A}} \left\{u_j(x) + \epsilon_j + \beta \int w(x') f(x'|x,j)\, dx'\right\}$$

The expected value function

- Taking the expectation on both sides of the previous (last slide) equation with respect to ϵ, we obtain the *integrated Bellman equation*

$$w(x) = \int \max_{j\in\mathcal{A}} \left\{ u_j(x) + \epsilon_j + \beta \int w(x') f(x'|x,j)\, dx' \right\} g(\epsilon) d\epsilon$$

- The right-hand side of this equation defines a contraction mapping, $\Gamma : \mathcal{B} \to \mathcal{B}$, on the space of bounded functions, $\mathcal{B}$
- Thus, the equation has a unique solution which must equal the expected value function, $w = \Gamma(w)$
- Note the role of the conditional independence and *iid* ϵ assumptions in the derivation of this equation

Blackwell's sufficient conditions for a contraction

Theorem

Let $\mathcal{B} = \{f : \mathbb{S} \to \mathbb{R} : \|f\| < \infty\}$ be the metric space of real-valued, bounded functions on $\mathbb{S}$. Suppose the operator $L : \mathcal{B} \to \mathcal{B}$ satisfies the following two conditions:

(1) Monotonicity: If $f, g \in \mathcal{B}$ and $f(x) \leq g(x)$ for all $x \in \mathbb{S}$, then $(Lf)(x) \leq (Lg)(x)$ for all $x \in \mathbb{S}$.

(2) Discounting: There is a number $\beta \in (0, 1)$ such that $(L(f + a))(x) \leq (Lf)(x) + \beta a$ for all $f \in \mathcal{B}$, $a > 0$, and $x \in \mathbb{S}$.

Then L is a contraction mapping with modulus β.

Γ is a contraction mapping

Proof.

If $g \leq h$, then

$$\int g(x')f(x'|x,j)\,dx' \leq \int h(x')f(x'|x,j)\,dx'$$

for all $x \in \mathbb{X}$ and $j \in \mathcal{A}$. It follows that $\Gamma f \leq \Gamma h$. Let $a \geq 0$. Then

$$\beta \int (g(x') + a)f(x'|x,j)\,dx' = \beta \int g(x')f(x'|x,j)\,dx' + \beta a.$$

Hence

$$(\Gamma(g + a))(x) = (\Gamma g)(x) + \beta a.$$

□

Choice-specific value functions

- Using the definition of the expected value function, $w(x)$, define the choice-specific value functions

$$v_j(x) = u_j(x) + \beta \int w(x')f(x'|x,j)\,dx'$$

- The full expected PDV of utilities from choosing j is $v_j(x) + \epsilon_j$

Choice-Specific value functions

- The choice-specific value functions characterize the model solution
- The value function can be expressed in terms of these choice-specific value functions,
$$v(x, \epsilon) = \max_{j \in \mathcal{A}} \{v_j(x) + \epsilon_j\}$$
  - Therefore, $w(x) = \int v(x, \epsilon) g(\epsilon) d\epsilon$ is sometimes also called the *EMAX function:*
$$w(x) = \int v(x, \epsilon) g(\epsilon) d\epsilon = \int \max_{j \in \mathcal{A}} \{v_j(x) + \epsilon_j\} g(\epsilon) d\epsilon$$
- Furthermore, the optimal decision rule satisfies
$$d^*(x, \epsilon) = k \Leftrightarrow v_k(x) + \epsilon_k \geq v_j(x) + \epsilon_j \quad \text{for all } j \in \mathcal{A},\ j \neq k$$

Recap

- The previous slides show how the solution of this class of dynamic decision problems can be simplified:
  1. Solve for the expected value function, $w(x)$
  2. Calculate the choice-specific value functions, $v_j(x)$
  3. Calculate the value function and optimal policy from the choice-specific value functions
- The key step here is 1: By "integrating out" ϵ, we can reduce the dimensionality of the problem by $K + 1$. This saves huge amounts of computing time!

Conditional choice probabilities (CCP's)

- The additive separability assumption on the utility function $U(s, a)$ implies that the expected PDV of choosing j is additively separable in $v_j(x)$ and the unobserved state, ϵ_j
- Thus, once $v_j(x)$ is known, the model predictions—choice probabilities—can be derived just as the model predictions of a static discrete choice model:

$$\Pr\{a = k|x\} = \Pr\{v_k(x) + \epsilon_k \geq v_j(x) + \epsilon_j \quad \forall j \neq k\}$$

  - We call these the conditional (on the state x) choice probabilities (CCP's)
- Frequently, we assume that ϵ_j has the Type I Extreme Value distribution. Then, choice probabilities have the logit form

$$\Pr\{a = k|x\} = \frac{\exp(v_k(x))}{\sum_{j=0}^{K} \exp(v_j(x))}$$

Technical note

- Let $\delta_1, \ldots, \delta_N$ be some numbers and let each of the N random variables ϵ_j be i.i.d. Type I Extreme Value (Gumbel) distributed
  - The cdf of each ϵ_j is $G(\epsilon_j) = \exp(-\exp(-(\epsilon_j - \mu)))$
  - $\mathbb{E}(\epsilon_j) = \mu + \gamma$, where $\gamma \approx 0.57722$ is Euler's constant
  - In many applications, ϵ_j is assumed to be drawn from the standard Gumbel distribution, $\mu = 0$
- If $g(\epsilon) = \prod_{j=1}^{N} g(\epsilon_j)$, we obtain the closed-form expression

$$\int \left(\max_{j=1,\ldots,N} \{\delta_j + \epsilon_j\} \right) g(\epsilon) d\epsilon = \mu + \gamma + \log\left(\sum_{j=1}^{N} \exp(\delta_j) \right)$$

Technical note

- Hence if ϵ_j is i.i.d. Type I Extreme value distributed, then the expected value function can be expressed as

$$w(x) = \mu + \gamma + \log \left(\sum_{j \in \mathcal{A}} \exp \left(u_j(x) + \beta \int w(x') f(x'|x, j)\, dx' \right) \right)$$

- Hence, the integration over ϵ can be performed analytically
- To get rid of the constant term $\mu + \gamma$ in the equation above, simply assume $\mu = -\gamma$, in which case ϵ_j is centered at 0

Computing the expected value function w

- Use a form of value function iteration
- The algorithm works because of the contraction mapping property of the operator Γ
- Note:
  - Policy iteration or modified policy iteration are better algorithms to obtain the value function corresponding to a Markov decision process
  - However, these algorithms are not applicable here because we calculate the expected value function, w, not the value function v

Computing w : The algorithm

1. Start with some $w^{(0)} \in \mathcal{B}$
2. For each $x \in \mathbb{X}$, calculate

$$w^{(n+1)}(x) = \int \max_{j \in \mathcal{A}} \left\{ u_j(x) + \epsilon_j + \beta \int w^{(n)}(x') f(x'|x,j)\, dx' \right\} g(\epsilon) d\epsilon$$

3. If $\left\| w^{(n+1)} - w^{(n)} \right\| < \varepsilon_w$, then proceed to 4. Otherwise, return to 2.
4. Calculate the (approximate) choice-specific value functions

$$v_j^{(n+1)}(x) = u_j(x) + \beta \int w^{(n+1)}(x') f(x'|x,j)\, dx'$$

Computing w : Speed of convergence

- For any starting value $w^{(0)} \in \mathcal{B}$, $w^{(n)} \to w$ in the sup-norm ($\|\cdot\|$)
- The rate of convergence is $O(\beta^n)$:

$$\limsup_{n \to \infty} \frac{\left\| w^{(n)} - w \right\|}{\beta^n} < \infty$$

- That is, $w^{(n)}$ converges geometrically at the rate β
  - Correspondingly, in marketing applications problems with short time intervals (daily or weekly decision making) the algorithm requires a larger number of iterations to find a solution with a given accuracy, because β will be close to 1

Literature background

- The model structure discussed above closely follows John Rust's (1987) "Optimal Replacement of GMC Bus Engines: An Empirical Model of Harold Zurcher" paper
  - Harold Zurcher was the superintendent of maintenance at the Madison Metropolitan Bus Company
- Other early key contributions
  - Choice of occupations — Miller (1984)
  - Fertility choice (number and timing of children) — Wolpin (1984)
  - Option value of patents — Pakes (1986)

Overview

Model solution code

- Remember that $x_t \equiv (i_t, P_t)$
  - $i_t \in \{0, 1, \ldots, I\}$
  - The price vector P_t can take one of L values, $\{P^{(1)}, \ldots, P^{(L)}\}$
- Hence, the state space $\mathbb{X}$ is discrete and finite, and we can associate each state x with two indices, (i, l), where $i \in \{0, \ldots, I\}$ and $l \in \{1, \ldots, L\}$
  - Allows us to store the expected value function, w, in an array with dimensions $(I + 1) \times L$

Code documentation

- The example is coded in MATLAB
- Run the code by calling Main.m
  - Defines the model settings, such as the size of the state space and the model parameters
  - Calls the value function iteration algorithm, simulates the model predictions, and creates graphs displaying the CCP's for each state
- The main model information is contained in three MATLAB structures:
  - `settings` — main model settings
  - `param` — model parameters
  - `price` — price distribution

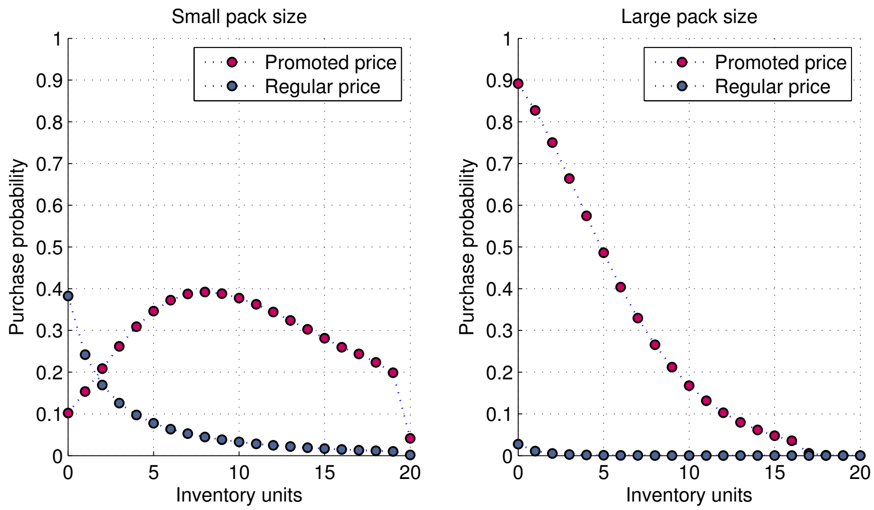
Code documentation

- `value_function_iteration.m`: Algorithm to solve for the expected value function, w
- `Bellman_operator.m`: Calculates the right-hand side of the expected value function equation given a guess of w
- `Model_simulation.m`: Simulates product choices and inventories for given choice-specific value functions (`v_choice`) and model settings

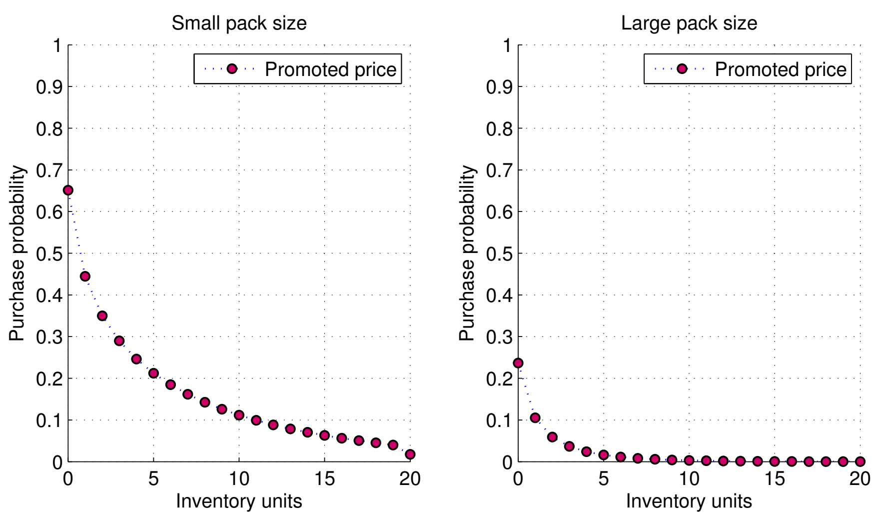
Example

- Number of different pack sizes: $K = 2$
- Units contained in each pack size: $n_1 = 2$, $n_2 = 5$
- Prices: $P_{low} = (1.2, 3.0)$, $P_{high} = (2.0, 5.0)$. $\Pr\{P = P_{low}\} = 0.16$
- State dimensions: $I = 20$, $L = 2$ — $21 \cdot 2 = 42$ states
- Consumption utility: $\delta = 4$
- Price sensitivity: $\alpha = 4$
- Inventory holding cost: $c(i) = 0.05 \cdot i$
- Purchase cost: $\tau(n_k) \equiv 0$

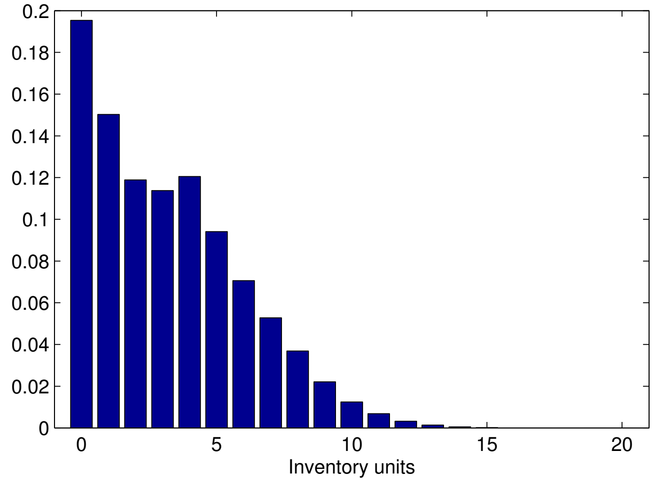
CCP's — pricing with promotions



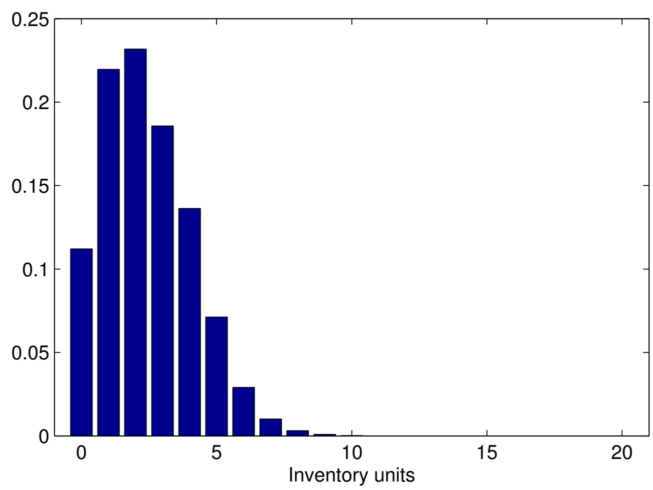
CCP's — only low price



Inventory distribution — pricing with promotions



Inventory distribution — only low price



Simulated purchase frequencies and sales levels

	Purchase frequency			Average sales units		
	Average price	P_{low}	P_{high}	Average price	P_{low}	P_{high}
Promotions						
Small	0.187	0.251	0.175	0.900	3.782	0.358
Large	0.105	0.656	0.001			
No purchase	0.708	0.093	0.823			
Only low price						
Small		0.374			0.991	
Large		0.049				
No purchase		0.577				

Overview

Overview

- Durable goods adoption decisions are inherently dynamic if consumers consider the trade-off between adopting a product now or in future
  - Prices of consumer durables typically fall over time
  - Product qualities improve
  - Uncertainty, for example about standards, is resolved over time

Model outline

- In each period $t = 0, 1, \ldots,$ a consumer decides whether to adopt one of K products
  - Choice $k = 0$: no purchase/waiting
- State vector: $x_t = (P_{1t}, \ldots, P_{kt}, z_t)$
  - P_{kt} is the price of product k
  - $z_t = 1$ indicates that the consumer already adopted the product in some period prior to t ($z_t = 0$ otherwise)
- Prices follow the log-normal process

$$\log(P_{jt}) = \gamma_j \log(P_{j,t-1}) + \eta_{jt},$$
$$\eta_t = (\eta_{1t}, \ldots, \eta_{Kt}) \sim N(0, \Sigma)$$

Model outline

- Utility from adopting k is

$$u_k(x_t) = \delta_k + \alpha P_{kt} \qquad \text{if } z_t = 0$$

- δ_k includes the present discounted utility from the product, hence we can assume that payoffs are $\equiv 0$ after product adoption $(z_t = 1)$
    - Hence, we focus exclusively on the case $z_t = 0$ and drop z_t from the state vector
- Utility from waiting: $u_0(x_t) = 0$
- The total payoff from choice k also includes an *iid* Type I Extreme Value distributed random utility component, ϵ_{kt}
- $\Rightarrow$ model fits into the dynamic discrete choice framework discussed before

State space

- The state vector, $x_t = (P_{1t}, \ldots, P_{kt})$, is an element of $\mathbb{X} = \mathbb{R}_+^K$
- $\mathbb{X}$ is continuous
- How can we calculate the expected value function w and the choice-specific value functions v_k on a computer if $\mathbb{X}$ is not discrete (finite)?

Overview

Numerical solution of v or w

- A value function is a mapping $v : \mathbb{S} \to \mathbb{R}$
- In the storable goods demand problem the state space was discrete (finite), but in the durable goods adoption problem the state space is continuous
- Numerical issues that we need to address:
  1. How do we store v, an infinite-dimensional object, in computer memory?
  2. How do we evaluate $v(x)$?
  3. How do we calculate the integral $\int v(x')p(s'|s,a)ds'$?
- These issues are addressed in Numerical Analysis
- 1. and 2. are issues of approximation and interpolation, 3. deals with numerical integration
- Note: Issues 1.-3. are the same with regard to the expected value function, $w : \mathbb{X} \to \mathbb{R}$

Discretization

- One strategy is to discretize the state space into a finite number of points
- For example, for a D-dimensional state space, we can choose a grid for each axis, $\mathcal{G}_i = \{x_{1i}, \ldots, x_{N_i,i}\}$, $x_{ki} < x_{k+1,i}$, and approximate $\mathbb{S}$ by $\mathcal{G} = \mathcal{G}_1 \times \cdots \times \mathcal{G}_D = \{s_1, \ldots, s_L\}$ (note that $L = |\mathbb{S}| = N_1 \cdot N_2 \cdots N_D$)
- v then becomes a finite-dimensional object, $v(s_i) = v_i$, and can be stored in computer memory using L floating-point numbers

Discretization

- One way to proceed is to also discretize the transition probability:

$$p_i(s,a) = \Pr\{s' = s_i | s, a\} = \int_{B_i} p(s'|s,a)ds',$$

or:
$$p_i(s,a) = \frac{p(s_i|s,a)}{\sum_{l=1}^{L} p(s_l|s,a)}$$

- Here, B_i is an appropriately chosen partition of the state space and $s_i \in B_i$
- For a given a, the evolution of the state can now be described by a stochastic matrix, P^a with (k,i)-element $p_i(s_k, a)$
- One can also discretize the set of actions, $\bigcup_{s \in \mathbb{S}} A(s)$

Discussion

- Although conceptually straightforward, the discretization discussed on the previous slide may neither deliver a fast algorithm nor an accurate solution
- Note that taking the expectation can involve up to $L = |\mathbb{S}|$ multiplications, which can be much larger than the number of multiplications using alternative methods, such as quadrature and interpolation

Interpolation

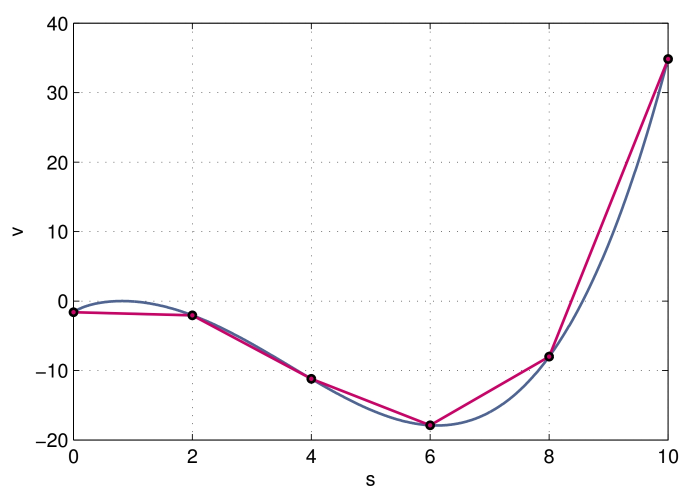
- Interpolation methods record the function values at a finite number of points (the table values), and use some algorithm to evaluate the function for any point that is not in the table:
  1. Discretize the state space into a grid $\mathcal{G}$, as before, and record the values of v at each point $s_i \in \mathcal{G}$
  2. Instead of assuming that the state can only take on values in $\mathcal{G}$, we now try to find a method to evaluate $v(s)$ at some arbitrary point $s \in \mathbb{S}$

Interpolation

- Linear interpolation in one dimension $(D = 1)$:
  1. Find the index i such that $x_i \leq s < x_{i+1}$
  2. Calculate $\omega = (s - x_i)/(x_{i+1} - x_i)$
  3. The interpolated value is

$$\widehat{v}(s) = v_i + \omega(v_{i+1} - v_i)$$

Interpolation: Example



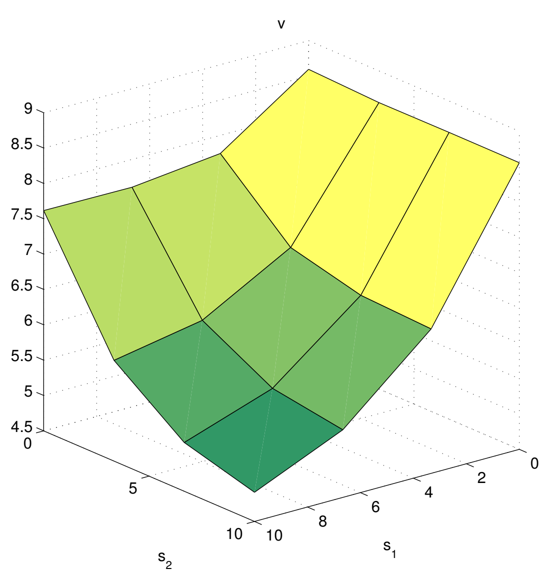
Bilinear interpolation

- We can extend the linear interpolation algorithm to more than one dimension
    1. In two dimensions, $s = (s_1, s_2)$, find the indices i and j such that $x_{i1} \leq s_1 < x_{i+1,1}$ and $x_{j2} \leq s_2 < x_{j+1,2}$
    2. Calculate $\omega_1 = (s_1 - x_{i1})/(x_{i+1,1} - x_{i1})$ and $\omega_2 = (s_2 - x_{j2})/(x_{j+1,2} - x_{j2})$
    3. The interpolated value is

$$\begin{aligned}\widehat{v}(s) &= y_0 + \omega_1(y_1 - y_0)\\ y_0 &= v_{ij} + \omega_2(v_{i,j+1} - v_{ij})\\ y_1 &= v_{i+1,j} + \omega_2(v_{i+1,j+1} - v_{i+1,j})\end{aligned}$$

- This procedure is called *bilinear interpolation*
- Note that bilinear interpolation does not yield a linear function surface!

Bilinear interpolation: Example



Other interpolation methods: Simplicial interpolation

- Partition the state space into triangles
- Interpolate using linear surfaces
- Example for $D = 2$
  1. Partition the state space into rectangles as before
  2. Further subdivide each rectangle into two triangles
  3. Interpolate using the unique linear function that coincides with v_i at each of the three vertices of the triangle

Other interpolation methods: Splines

- A spline of order N is a function ψ that is a polynomial of degree $N-1$ on each interval (or rectangle) defined by the grid
- Furthermore, we require that ψ is C^{N-2} on all of the grid, not just within each interval or rectangle
- That is, we require that ψ is smooth not only within, but also at the boundary of each interval or rectangle
- Example: A cubic spline is of order $N = 4$. It has continuous first and second derivatives everywhere. For $D = 1$, it can be represented as

$$\psi(s) = a_{0i} + a_{1i}s + a_{2i}s^2 + a_{3i}s^3$$

  within each interval $[x_i, x_{i+1}]$
- The requirement that ψ is C^{N-2} imposes coefficient restrictions for the polynomials across intervals/rectangles

Splines

- Advantage of splines: allows for continuous or smooth derivatives
- This can be important in games with continuous controls to ensure continuity of best-reply functions and thus existence of a pure-strategy equilibrium

Approximation using orthogonal polynomials

- Instead of first discretizing the state space and then approximating $v(s)$ using interpolation, we try to approximate $v(s)$ globally using a linear combination of easy-to-evaluate functions
- We start with a set of basis functions, $\phi_i : \mathbb{S} \to \mathbb{R}$, $i = 1, \ldots, K$. Here, $\mathbb{S} \subset \mathbb{R}^D$
- Each ϕ_i is a polynomial (polynomials are easy to evaluate)
- To simplify the exposition, let $D = 1$ and consider approximations on the interval $[-1, 1]$

Approximation using orthogonal polynomials

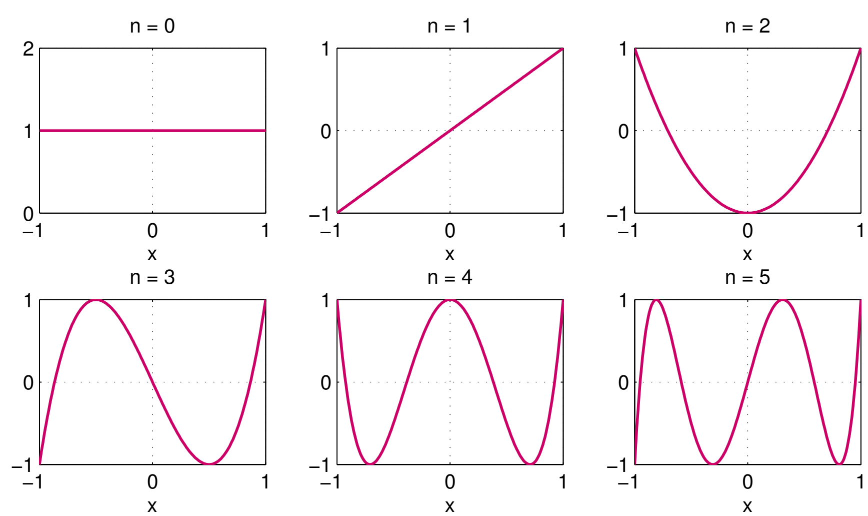
- We will only discuss Chebyshev polynomials, which are particularly useful in many applications
- The Chebyshev polynomial of degree n is defined as

$$T_n(x) = \cos(n \cdot \cos^{-1}(x))$$

- Even though this doesn't look like a polynomial, it is. An easier way to calculate $T_n(x)$ is to set $T_0(x) = 1$, $T_1(x) = x$, and then for $n \geq 2$ recursively calculate

$$T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x)$$

Chebyshev polynomials, degree $n = 0, \ldots, 5$



Approximation using orthogonal polynomials

- Define the inner product $\langle f, g \rangle$ between two functions f and g on $[-1, 1]$ as

$$\langle f, g \rangle = \int_{-1}^{1} \frac{f(x)g(x)}{\sqrt{1-x^2}} dx$$

- We say that f and g are orthogonal to each other if $\langle f, g \rangle = 0$
- One can verify that Chebyshev polynomials are mutually orthogonal, $\langle T_n, T_m \rangle = 0$ for $n \neq m$
- Goal: Choose a linear combination of Chebyshev polynomials to approximate some arbitrary function,

$$v(s) \approx \sum_{i=0}^{K} \theta_i T_i(s)$$

Chebyshev regression algorithm

- An algorithm to find a good approximation for f on some interval $[a, b]$ is as follows:
  1. Choose the degree N of the polynomial and the number of interpolation nodes $M \geq N + 1$
  2. Compute each interpolation node:

$$\nu_i = -\cos\left(\frac{2i-1}{2M}\pi\right), \quad i = 1, \ldots, M$$

  3. Adjust these nodes to the interval $[a, b]$:

$$\xi_i = (\nu_i + 1)\frac{(b-a)}{2} + a$$

  4. Calculate the Chebyshev coefficients:

$$\theta_k = \frac{\sum_{i=1}^{M} f(\xi_i) T_k(\nu_i)}{\sum_{i=1}^{M} T_k(\nu_i)^2} \quad k = 0, \ldots, N$$

Regression with orthogonal regressors

- Consider the regression $y = X\beta + \epsilon$, where x_j denotes the j^{th} column of X
- Suppose the regressors are orthogonal:

$$\langle x_j, x_k \rangle = x_j' x_k = 0$$

- Then

$$X^T X = \begin{bmatrix} x_1'x_1 & 0 & \cdots & 0 \\ 0 & x_2'x_2 & & \\ \vdots & & \ddots & \vdots \\ 0 & & \cdots & x_N'x_N \end{bmatrix}$$

Regression with orthogonal regressors

- Let $b = (X^T X)^{-1} X^T y$
- Then

$$b_k = \frac{x_k' y}{x_k' x_k} = \frac{\sum_i x_{ik} y_i}{\sum_i x_{ik} x_{ik}}$$

- Note the form of the Chebyshev coefficients:

$$\theta_k = \frac{\sum_i v(\xi_i) T_k(\nu_i)}{\sum_i T_k(\nu_i)^2}$$

- Hence the name "Chebyshev regression"!
- Difference to regular regression: the Chebyshev regression algorithm chooses the values of the regressors/nodes (in a smart fashion, to minimize approximation error)

Approximation using orthogonal polynomials

- Given the Chebyshev coefficients calculated in the Chebyshev regression algorithm, we can approximate $v(x)$, $x \in [a, b]$ by

$$\hat{v}(x) = \sum_{k=0}^{N} \theta_k T_k \left(2\frac{x-a}{b-a} - 1 \right)$$

- If v is C^k, $k \geq 1$, and $\hat{v}_N(x)$ is a degree N Chebyshev approximation, then

$$\lim_{N \to \infty} \sup_x |\hat{v}_N(x) - v(x)| = 0$$

  - $\hat{v}_N$ converges uniformly to v if v is (at least) continuously differentiable
  - For non-differentiable v's, Chebyshev regression will generally not provide a good approximation

Multidimensional approximation

- Suppose we have basis functions $T_{id}(x_d)$, $i = 1, \ldots K_d$, for each of $d = 1, \ldots, D$ dimensions
- Goal: construct a basis for functions of $(x_1, \ldots, x_D)$
- Collect all functions of the form

$$\Psi(x_1, \ldots, x_D) = \prod_{d=1}^{D} T_{i_d d}(x_d), \qquad 1 \leq i_d \leq K_d$$

- The collection of all these $N = \prod_{d=1}^{D} K_d$ functions is called the D*-fold tensor product basis*
- Important application: D-dimensional Chebyshev approximation

Discussion

- What are the relative merits of approximations using orthogonal polynomials versus interpolation methods such as bilinear interpolation?
  - Chebyshev coefficients are easy to compute if there is only a small number of nodes M, while finding values for each grid point $s_i \in \mathcal{G}$ can be very time-intensive
  - If the true function v is not smooth or displays a lot of curvature, then a Chebyshev approximation will be poor unless N and M are large, in which case there is no more speed advantage over interpolation
- Note that approximation using grids and interpolation is essentially a non-parametric method, while approximation using polynomials of fixed degree N is a parametric method

Discussion: The curse of dimensionality

- In practice, the biggest problem when we try to find a solution of a Markov decision problem involves computation time
- If we discretize each axis of the space into M points, then the total number of grid points is M^D
  - Hence, the number of points at which the value function has to be calculated rises exponentially in D
  - Even worse, the number of floating point operations to evaluate V also rises exponentially for many methods

Newton-Cotes formulas

- Easy to understand, but ineffective compared to Gaussian quadrature or Monte Carlo methods
- Example: Midpoint rule
  - Consider the interval $[a, b]$, choose N and define the step-size $h = (b - a)/N$
  - Let $x_i = a + ih - h/2$. x_i is the midpoint in the interval $[a + (i - 1)h, a + ih]$
  - Approximate the integral of f :

$$\int_a^b f(x)dx \approx h \sum_{i=1}^{N} f(x_i)$$

- In practice, we can typically calculate the integral with much higher accuracy given the same number of operations using alternative methods

Gaussian quadrature

- Gaussian quadrature rules are formulas for integrals of the function $W(x)f(x)$ for some specific $W(x)$:

$$\int_a^b W(x)f(x)dx \approx \sum_{i=1}^{N} w_i f(x_i)$$

- The w_i's are weights, and the x_i's are called nodes or abscissas
- This looks exactly like a Newton-Cotes formula, however, the main idea of Gaussian quadrature is to choose the weights and nodes such that the approximation is very close to the true integral even if N is small
  - Weights and nodes are chosen so that the approximation is exact for low-order polynomials

Gaussian quadrature

- Given a function $W(x)$ and a choice of N, there are algorithms to compute the quadrature nodes x_i and the corresponding weights w_i
- There are different quadrature rules, distinguished by the choice of the function $W(x)$
- Each rule makes use of the fact that the integrand can be factored as $W(x)f(x)$ with $W(x)$ known

Gaussian quadrature

- Consider the important case of Gauss-Hermite quadrature:

$$W(x) = \exp\left(-x^2\right)$$

- For the corresponding nodes x_i and weights w_i,

$$\int_{-\infty}^{\infty} \exp\left(-x^2\right) f(x)dx \approx \sum_{i=1}^{N} w_i f(x_i)$$

- Hence,

$$\int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right) f(x)dx \approx \frac{1}{\sqrt{\pi}} \sum_{i=1}^{N} w_i f(\mu + \sqrt{2}\sigma x_i)$$

Gaussian quadrature

- Gauss-Hermite quadrature yields a very accurate approximation to integrals involving a normal pdf for a small number of quadrature nodes N
- Gaussian quadrature can be extended to higher dimensions, $D > 1$
- Unfortunately, there is a curse of dimensionality in D
- For large D, Monte Carlo integration is the only feasible integration method

Monte Carlo integration

- Basic idea: Let $(X_n)_{n\geq 0}$ be a sequence of random variables with density $p(x)$
- Then we can simulate the integral

$$\int f(x)p(x)dx \approx \frac{1}{N}\sum_{n=1}^{N} f(X_i)$$

- In fact,

$$\frac{1}{N}\sum_{n=1}^{N} f(X_i) \xrightarrow{a.s.} \int f(x)p(x)dx$$

- Methods such as importance sampling improve on the efficiency of this crude simulator

Overview

Code documentation

- The solution algorithm is coded in MATLAB
- Run `Main.m` to compute and display the solution of the durable goods adoption problem
- Solutions are obtained using both bilinear interpolation and Chebyshev regression
- All information (grid points, table values, Chebyshev coefficients, ...) is contained in MATLAB structures (`w_interp`, w_cheb, etc.)

Main.m

1. Sets values for the model parameters
2. Calls `Gauss_Hermite_multidim` to calculate Gauss-Hermite quadrature weights and nodes
3. Defines and initializes the bilinear and Chebyshev approximation routines
4. Calls `value_function_iteration` to solve the model using both methods
5. Calls `calculate_choice_probabilities` to predict the CCP's
6. Defines a grid on which to graph the solutions and calls `transfer_representation` to evaluate the model solutions on this grid
7. Plots the expected value functions, conditional choice probabilities, and differences in the CCP's predicted by the interpolation and Chebyshev approximation solutions

value_function_iteration

- Uses value function iteration to calculate the expected value function w
- Works for both bilinear interpolation and Chebyshev approximation

Bellman_operator

1. Bilinear interpolation:
   - Iterates through the state space and calculates the expected value function for each state
   - The expected future value is obtained using quadrature and bilinear interpolation
2. Chebyshev approximation:
   - Algorithm can be efficiently expressed using matrix operations
   - Utilize linearity of quadrature and Chebyshev approximation to pre-compute many terms needed to update the expected value function
   - See details below

calculate_choice_probabilities and Gauss_Hermite_multidim

- `calculate_choice_probabilities`
  - Input: Array of bilinear interpolation structures, `Pr_list`, and a structure containing the solution of w
  - Output: `Pr_list`, containing the CCP's
- `Gauss_Hermite_multidim`
  - Routine to calculate Gauss-Hermite quadrature weights and nodes for a given dimension and number quadrature nodes
  - Needs to be called only once

Bilinear interpolation utilities

- The interpolation structure is initialized using `initialize_interpolator_2D`
- `interpolate_2D` uses bilinear interpolation to find a function value for a list (array) of state vectors
- `display_interpolator_2D` graphs the interpolated function

Chebyshev approximation utilities

- The Chebyshev approximation structure is initialized using `initialize_Chebyshev_2D`
- `calculate_Chebyshev_coefficients_2D` calculates the Chebyshev regression coefficients θ
- `evaluate_Chebyshev_2D` uses the current Chebyshev coefficients to evaluate the function approximation
- `return_evaluation_T_Chebyshev_2D` computes and returns the polynomial terms involved in evaluating the Chebyshev approximation
- `calculate_Chebyshev_polynomials` is a utility used by some of the previous routines

Details on the Chebyshev regression algorithm

- Goal: Approximate w on the rectangle $[a_1, b_1] \times [a_2, b_2]$
- Preliminary steps and definitions:
  1. Choose the degree N of the polynomial and the number of interpolation nodes $M \geq N + 1$
  2. Compute each interpolation node:
  $$\nu_i = -\cos\left(\frac{2i-1}{2M}\pi\right), \quad i = 1, \ldots, M$$
  3. Adjust these nodes to the interval $[a_k, b_k]$, $k = 1, 2$:
  $$\xi_{ik} = (\nu_i + 1)\frac{(b_k - a_k)}{2} + a_k$$
  4. Let $\mathcal{B} = \{f : [-1, 1]^2 \to \mathbb{R} : f(x_1, x_2) \equiv T_{i_1}(x_1)T_{i_2}(x_2)$ for $0 \leq i_1, i_2 \leq N\}$ be the two-fold tensor product basis. We denote the $J = (N + 1)^2$ elements of $\mathcal{B}$ by ϕ_j, $\mathcal{B} = \{\phi_1, \ldots, \phi_J\}$
  5. Let $\mathcal{Z} = \{z = (x_1, x_2) : x_k = \xi_{ik}$ for $i = 1, \ldots, M$ and $k = 1, 2\}$. $\mathcal{Z}$ has $L = M^2$ elements, $\mathcal{Z} = \{z_1, \ldots, z_L\}$

Details on the Chebyshev regression algorithm

- To find the Chebyshev regression approximation of some function $f : [a_1, b_1] \times [a_2, b_2] \to \mathbb{R}$:
  1. Calculate $y = (y_1, \ldots, y_L)$, where $y_l = f(z_l)$, $z_l \in \mathcal{Z}$
  2. Calculate the $L \times J$ matrix X, where element $X_{lj} = \phi_j(z_l)$
  3. Find the Chebyshev coefficient vector θ from the regression model
  $$y = X\theta,$$
  $$\theta = (X^T X)^{-1} X^T y = Ay,\ A = (X^T X)^{-1} X^T$$
- Note that X and hence A can be pre-computed before running the value function iteration algorithm
  - Hence, given y, finding the Chebyshev regression coefficients only requires the matrix multiplication Ay

Details on the Chebyshev regression algorithm

- To calculate y_l we need to compute the choice-specific value functions

$$v_k(z_l) = u_k(z_l) + \beta \int w(x')f(x'|z_l, k)\, dx'$$

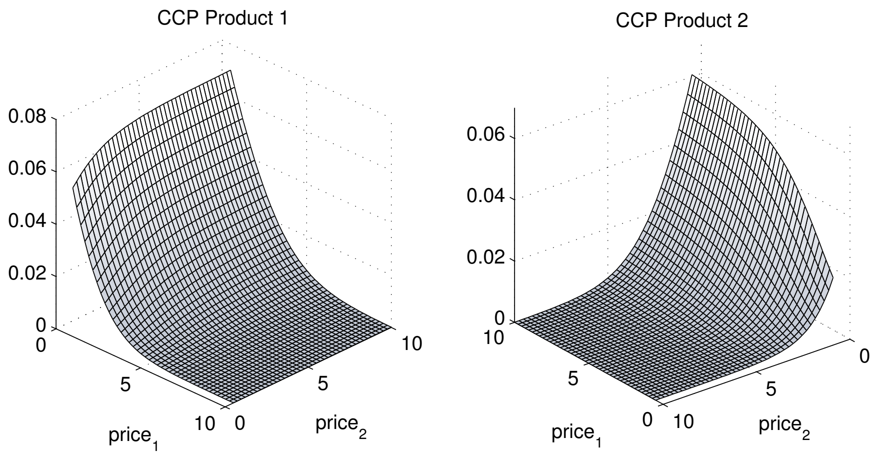
- The expectation in the expression above can be calculated using Chebyshev approximation and quadrature (x_{ql} and ω_q denote the respective quadrature nodes and weights):

$$\begin{aligned}
\int w(x')f(x'|z_l, k)\, dx' &\approx \sum_{q=1}^{Q} \omega_q \left(\sum_{j=1}^{J} \theta_j \phi_j(x_{ql}) \right) \\
&= \sum_{j=1}^{J} \theta_j \left(\sum_{q=1}^{Q} \omega_q \phi_j(x_{ql}) \right) \\
&\approx \sum_{j=1}^{J} \theta_j \int \phi_j(x')f(x'|z_l, k)\, dx'
\end{aligned}$$

Details on the Chebyshev regression algorithm

- Define a $L \times J$ matrix T with elements $T_{lj} = \sum_{q=1}^{Q} \omega_q \phi_j(x_{ql})$
- T can be pre-computed before running the value function algorithm
- Calculate $e = T\theta$
  - Then $e_l \approx \int w(x')f(x'|z_l, k)\, dx'$
- Note
  - The calculation of T exploits the linearity in quadrature and Chebyshev approximation (hence, a similar method does not work for non-linear approximation or interpolation methods, such as bilinear interpolation)
  - Pre-computing T saves enormous amounts of computing time
  - Pre-computing T would not be possible for continuous actions a_t
- Also, note that in the durable goods adoption example the transition density does not depend on k

Example: CCP's from model solution



Overview

Theory of dynamic decision processes

- Bertsekas, D. P. (2005): Dynamic Programming and Optimal Control: 2nd Edition (Volumes 1 and 2). Athena Scientific
- Puterman, M. L. (1994): Markov Decision Processes. Wiley-Interscience
- Rust, J. (1996): "Numerical Dynamic Programming in Economics," in Handbook of Computational Economics, ed. by H. M. Amman, D. A. Kendrick, and J. Rust: Elsevier Science B.V., 619-729
- Stokey, N. L., R. E. Lucas, and E. C. Prescott (1989): Recursive Methods In Economic Dynamics. Harvard University Press

Dynamic discrete choice models

- Aguirregabiria, V. and Mira, P. (2010): "Dynamic discrete choice structural models: A survey," Journal of Econometrics, 156, 38-67
- Miller, R. A. (1984): "Job Matching and Occupational Choice," Journal of Political Economy, 92(6), 1086-1120
- Pakes, A. (1986): "Patents as Options: Some Estimates of the Value of Holding European Patent Stocks," Econometrica, 54(4), 755-784
- Rust, J. (1987): "Optimal Replacement of GMC Bus Engines: An Empirical Model of Harold Zurcher," Econometrica, 55(5), 999-1033
- Rust, J. (1994): "Structural Estimation of Markov Decision Processes," in Handbook of Econometrics, ed. by R. F. Engle, and D. L. McFadden. London and New York: Elsevier North-Holland, 3081-3143
- Wolpin, K. I. (1984): "An Estimable Dynamic Stochastic Model of Fertility and Child Mortality," Journal of Political Economy, 92(5), 852-874

Numerical analysis

- Judd, K. L. (1998): Numerical Methods in Economics. Cambridge: MIT Press
- Press, W. H., S. A. Teukolsky, W. T. Vetterling, And B. P. Flannery (2007): Numerical Recipes: The Art of Scientific Computing. Cambridge University Press
- Rust, J. (1996): "Numerical Dynamic Programming in Economics," in Handbook of Computational Economics, ed. by H. M. Amman, D. A. Kendrick, and J. Rust: Elsevier Science B.V., 619-729
- Ueberhuber, C. W. (1997): Numerical Computation: Methods, Software, and Analysis (Volumes 1 and 2). Springer Verlag
- Benítez-Silva, H., G. Hall, G. J. Hitsch, G. Pauletto, and J. Rust (2001): "A Comparison of Discrete And Parametric Approximation Methods For Continuous-State Dynamic Programming Problems," manuscript

The Storable Goods Demand Problem

- Erdem, T., Imai, S., and Keane, M. P. (2003): "Brand and Quantity Choice Dynamics Under Price Uncertainty," Quantitative Marketing and Economics, 1, 5-64
- Hartmann, W. R. and Nair, H. S. (2010): "Retail Competition and the Dynamics of Demand for Tied Goods," Marketing Science, 29(2), 366-386
- Hendel, I. and Nevo, A. (2006): "Measuring the Implications of Sales and Consumer Inventory Behavior," Econometrica, 74(6), 1637-1673

The Durable Goods Adoption Problem

- Dubé, J.-P., Hitsch, G. J., and Chintagunta, P. (2010): "Tipping and Concentration in Markets with Indirect Network Effects," Marketing Science, 29(2), 216-249
- Goettler, R. and Gordon, B. (2011: "Does AMD Spur Intel to Innovate More?," Journal of Political Economy, 119(6), 1141-1200
- Gordon, B. R. (2009): "A Dynamic Model of Consumer Replacement Cycles in the PC Processor Industry," Marketing Science, 28(5), 846-867
- Gowrisankaran, G. and Rysman, M. (2012): "Dynamics of Consumer Demand for New Durable Goods," Journal of Political Economy, 120(6), 1173-1219
- Nair, H. S. (2007): "Intertemporal Price Discrimination with Forward-Looking Consumers: Application to the US Market for Console Video-Games," Quantitative Marketing and Economics, 5, 239-292
- Melnikov, O. (2013): "Demand for Differentiated Durable Products: The Case of the U.S. Computer Printer Market," Economic Inquiry, 51(2), 1277-1298
- Song, I. and Chintagunta, P. K. (2003): "A Micromodel of New Product Adoption with Heterogeneous and Forward-Looking Consumers: Application to the Digital Camera Category," Quantitative Marketing and Economics, 1, 371-407