

Approximation Methods

General Objective:

Given some data about a function $f(x)$ which is difficult to compute, construct a simpler function $g(x)$ which is a good approximation of $f(x)$.

Questions:

- What data should be produced and used?
- What family of “simpler” functions should be used?
- How good is approximation?
- How simple can a good approximation be?

Comparisons with statistical regression

- Both have some unknown function which is approximated
- Both use a finite amount of data
- Statistical data is noisy; we assume here that errors are small
- Nature produces data in statistics; we produce the data here
- Approximation methods are like experimental design with very small experimental error

Regression

- **Data:** $(x_i, y_i), i = 1, \dots, n$.
- **Objective:** Find a function $f(x; \beta)$ with $\beta \in R^m, m \leq n$, with $y_i \doteq f(x_i), i = 1, \dots, n$.
- **Least Squares regression:**

$$\min_{\beta \in R^m} \sum (y_i - f(x_i; \beta))^2$$

- **Problem:** If $f(x; \beta) = \sum_{j=0}^m \beta_j x^j$, the equations defining β are ill-conditioned since the x^j functions are highly correlated - multicollinearity

Challenge

Find approximation methods

- which are reliable as we get more data
- avoid ill-conditioned formulations.

Interpolation

- **Interpolation Problem:** find a function from a family which exactly fits some data

- **Lagrange polynomial interpolation**

- **Data:** $(x_i, y_i), i = 1, \dots, n.$

- **Objective:** Find a polynomial of degree $n - 1$, $p_n(x)$, which agrees with the data, i.e.,

$$y_i = f(x_i), i = 1, \dots, n$$

- **Result:** If the x_i are distinct, there is a unique interpolating polynomial

- **Evaluation of interpolation**

- Advantage: use few data points

- Disadvantage: data must be computed with small error

- **Question:** Does $p_n(x)$ converge to $f(x)$ as we use more points (and higher-order p_n)?

- **Convergence Counterexample**

– Suppose

$$f(x) = \frac{1}{1+x^2}, \quad x_i : \text{uniform on } [-5, 5]$$

– Degree 11 Result:

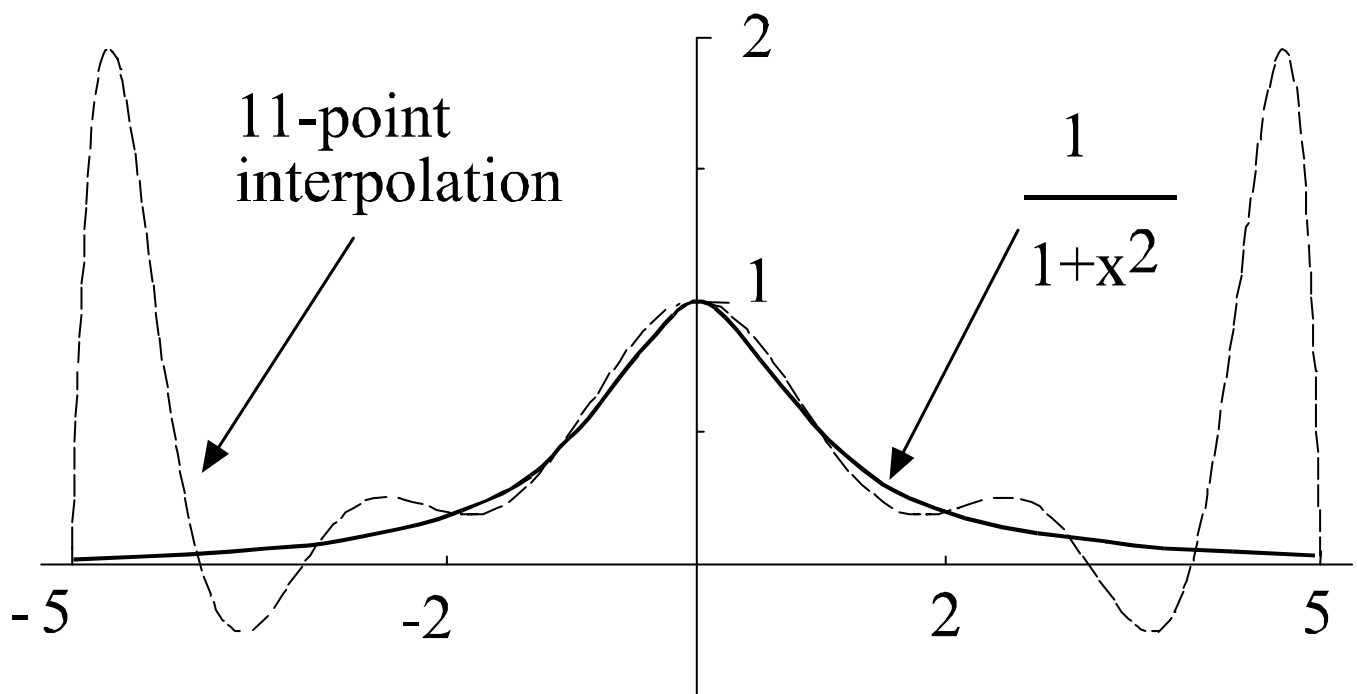


Figure 1:

- **Conclusion:** simple interpolation is not reliable

Orthogonal polynomials

General orthogonal polynomials

- Space: polynomials over domain D
- weighting function: $w(x) > 0$
- Inner product: $\langle f, g \rangle = \int_D f(x)g(x)w(x)dx$

Usefulness of orthogonal polynomials

- Regressing against orthogonal polynomials is numerically stable - regressors are orthogonal
- Reliable interpolation formulas are generated by orthogonal polynomials.

Chebyshev polynomials

- $[a, b] = [-1, 1]$
- $w(x) = 1$
- $T_n(x) = \cos(n \cos^{-1} x)$
- Recurrence formula:

$$T_0(x) = 1$$

$$T_1(x) = x$$

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

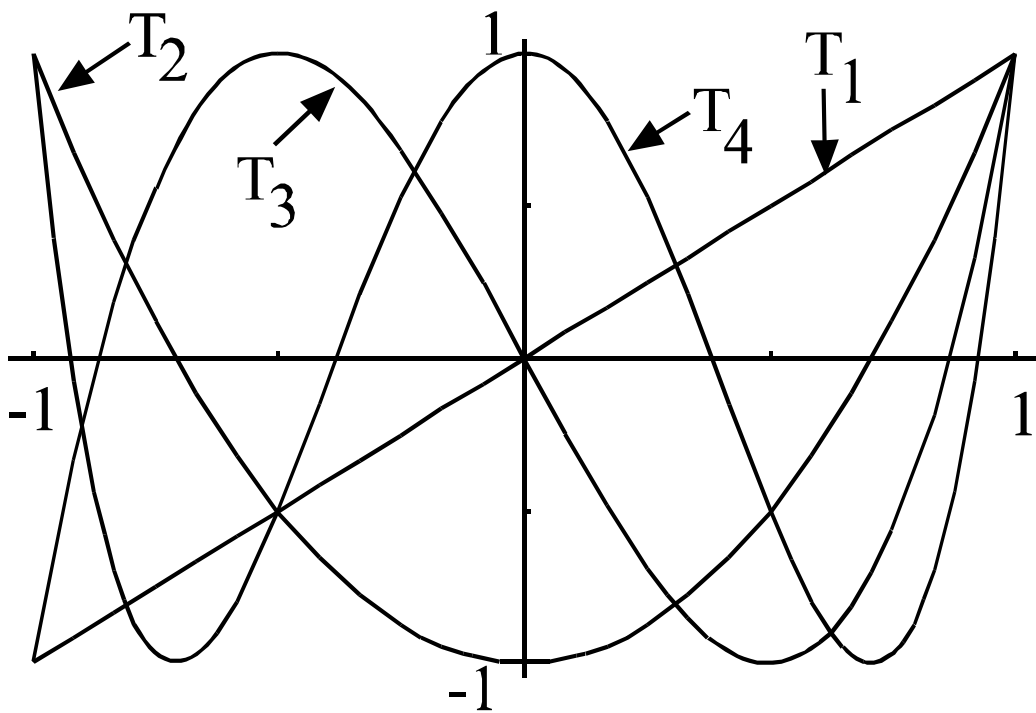


Figure 2:

Legendre polynomials

- $[a, b] = [-1, 1]$
- $w(x) = 1$
- $P_n(x) = \frac{(-1)^n}{2^n n!} \frac{d^n}{dx^n} [(1 - x^2)^n]$
- Recurrence formula:

$$\begin{aligned}P_0(x) &= 1 \\P_1(x) &= x \\P_{n+1}(x) &= \frac{2n+1}{n+1} x P_n(x) - \frac{n}{n+1} P_{n-1}(x),\end{aligned}$$

Laguerre polynomials

- $[a, b] = [0, \infty]$
- $w(x) = e^{-x}$
- $L_n(x) = \frac{e^x}{n!} \frac{d^n}{dx^n} (x^n e^{-x})$
- Recurrence formula:

$$\begin{aligned}L_0(x) &= 1 \\L_1(x) &= 1 - x \\L_{n+1}(x) &= \frac{1}{n+1} (2n+1-x) L_n(x) - \frac{n}{n+1} L_{n-1}(x),\end{aligned}$$

Hermite polynomials

- $[a, b] = [-\infty, \infty]$
- $w(x) = e^{-x^2}$
- $H_n(x) = (-1)^n e^{x^2} \frac{d^n}{dx^n} (e^{-x^2})$
- Recurrence formula:

$$H_0(x) = 1$$

$$H_1(x) = 2x$$

$$H_{n+1}(x) = 2x H_n(x) - 2n H_{n-1}(x).$$

Trigonometric polynomials and Fourier series

- If f is continuous on $[-\pi, \pi]$ and $f(-\pi) = f(\pi)$, then

$$f(\theta) = \frac{1}{2}a_0 + \sum_{n=1}^{\infty} a_n \cos(n\theta) + \sum_{n=1}^{\infty} b_n \sin(n\theta)$$

where the *Fourier coefficients* are defined by

$$a_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(\theta) \cos(n\theta) d\theta$$
$$b_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(\theta) \sin(n\theta) d\theta,$$

- Convergence is uniform.
- Since the sine and cosine functions are orthogonal on $[-\pi, \pi]$.
- A *trigonometric polynomial* is any function of the form in (6.4.4).
- Fourier approximation is excellent for approximating a smooth *periodic function*, that is, a function $f : R \rightarrow R$ such that for some ω , $f(x) = f(x + \omega)$.
- However, convergence is not uniform, and many terms are needed for good fits of nonperiodic functions.

Chebyshev Approximation

- **Chebyshev Interpolation:**

- Let x_i be the n zeroes of $T_n(x)$ (adapted to $[a, b]$)

$$x_i = \cos\left(\frac{2i-1}{2n}\pi\right), \quad i = 1, \dots, n.$$

- Coefficients of the $T_k(x)$ are easily computed; coefficient of $T_k(x)$ is essentially covariance of x_i with $T_k(x_i)$.

- **Chebyshev interpolation results**

- converges in L^∞ as we increase n : number of points and degree of polynomial
- easy to compute
- does *not* approximate both f and f'

- **Chebyshev regression:**

- Let x_i be the n zeroes of $T_n(x)$ (adapted to $[a, b]$)

$$x_i = \cos\left(\frac{2i-1}{2n}\pi\right), \quad i = 1, \dots, n.$$

- Regress $m > n$ values of y_i against $(T_0(x_i), T_1(x_i), \dots, T_{n-1}(x_i))$
- Coefficients of the $T_k(x)$ are easily computed; coefficient of $T_k(x)$ is essentially covariance of x_i with $T_k(x_i)$.
- Result is smoother than interpolation
- Does a better job of approximating $f'(x)$

Other approximation schemes

- Splines

- Split $[a, b]$ into several intervals
- Use a different polynomial on each interval
- Make sure that the different polynomials meet properly

- Rational functions

$$f(x) = \frac{a_0 + a_1x + a_2x^2 + \dots + a_nx^n}{b_0 + b_1x + b_2x^2 + \dots + b_mx^m}$$

- Neural networks

- Single-layer neural network is a function

$$F(x; \beta) \equiv h \left(\sum_{i=1}^n \beta_i g(x_i) \right) \quad (6.14.1)$$

where $g(x)$ and $h(x)$ are monotone increasing scalar functions

- Single hidden-layer feedforward network has the form

$$F(x; \beta, \gamma) \equiv f \left(\sum_{j=1}^m \gamma_j h \left(\sum_{i=1}^n \beta_i^j g(x_i) \right) \right), \quad (6.14.2)$$

where $f(x)$, $g(x)$, and $h(x)$ are monotone increasing scalar functions

Shape Problems

- Concave (monotone) data may lead to nonconcave (nonmonotone) approximations by polynomials, splines, etc..
- Example

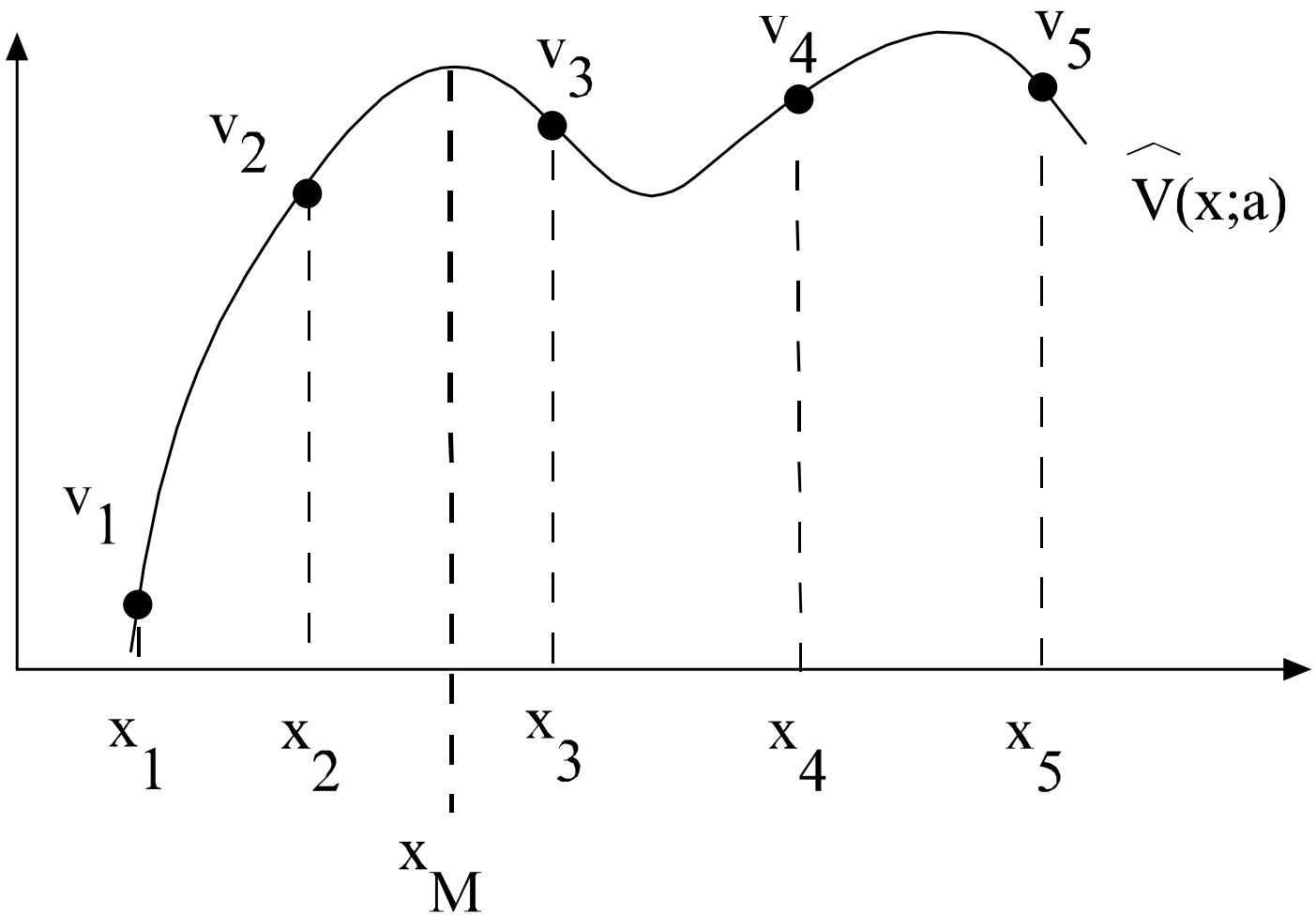


Figure 3:

- There are procedures which can preserve shape
- Schumaker Procedure:
 1. Take level (and maybe slope) data at nodes x_i
 2. Add intermediate nodes $z_i^+ \in [x_i, x_{i+1}]$
 3. Run quadratic spline with nodes at the x and z nodes which interpolate data and preserves shape.
 4. Schumaker formulas tell one how to choose the z and spline coefficients.
- Many other procedures exist for one-dimensional problems
- Some procedures exist for two-dimensional problems
- Higher dimensions are difficult, but many questions are open.

Approximation Methods: Summary

Interpolation versus regression

- Lagrange data uses level information only
- Hermite data also uses slope information
- Regression uses more points than coefficients

One-dimensional problems

- Smooth approximations
 - Orthogonal polynomial methods for nonperiodic functions
 - Fourier approximations for periodic functions
- Less smooth approximations
 - Splines
 - Shape-preserving splines

Multidimensional data

- Can use direct, dimension-by-dimension generalizations of orthogonal polynomials and splines
- Neural networks become useful

Approximation versus Statistics

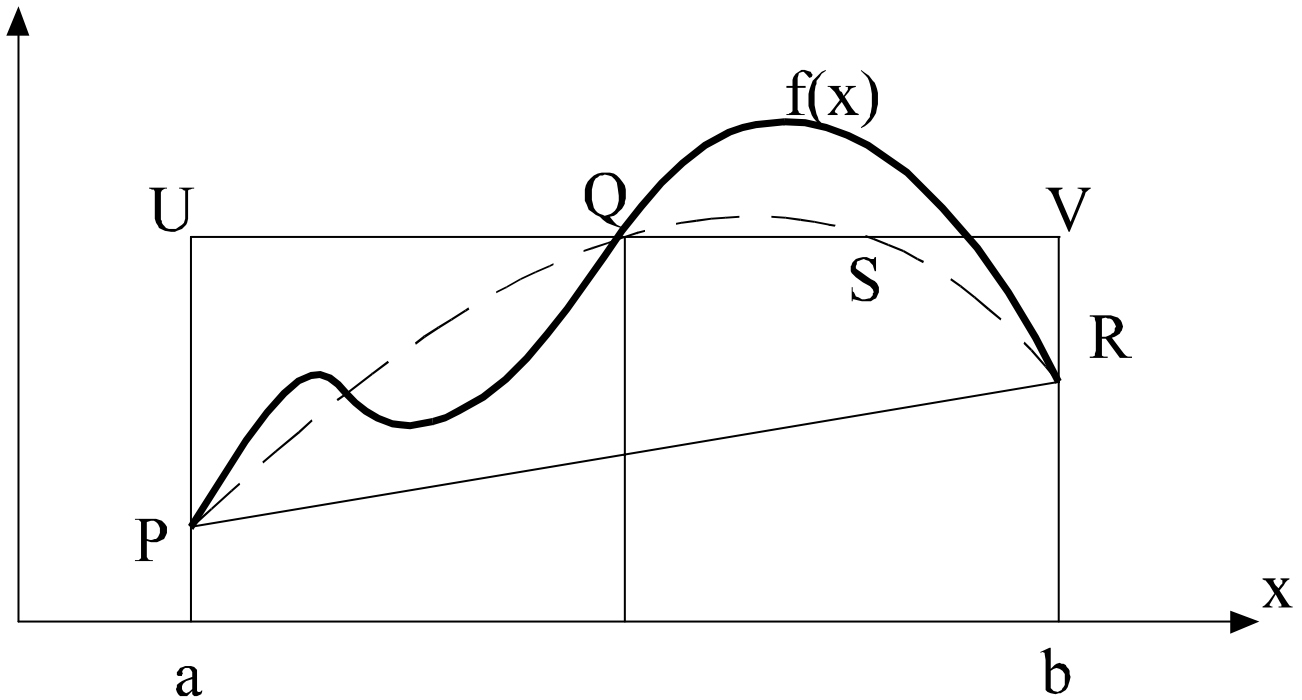
- Similarities:
 - both approximate unknown functions
 - both use finite amount of data
- Differences
 - approximation uses error-free data, not noisy data
 - approximation generates data, not constrained by observations

Integration

- Integrals frequently arise in economics
 - Expected utility
 - Discounted utility and profits over a long horizon
 - Bayesian posterior
 - Likelihood functions
- Most integrals cannot be evaluated analytically

Newton-Cotes Formulas

- Idea: Approximate function with low order polynomials and then integrate approximation



- Step function approximation:
 - approximate $f(x)$ with constant equal to $f(x)$ at midpoint of $[a, b]$
 - Integral approximation is $aUQVb$ box
- Linear function approximation:
 - approximate $f(x)$ with linear function between a and b
 - Integral approximation is trapezoid $aPRb$
- Parabolic function approximation:
 - approximate $f(x)$ with parabola through a , b , and $(a + b)/2$
 - Integral approximation is area of $aPQRb$

- Trapezoid Rule: piecewise linear approximation

– Simple rule: for some $\xi \in [a, b]$

$$\int_a^b f(x) dx = \frac{b-a}{2} [f(a) + f(b)] - \frac{(b-a)^3}{12} f''(\xi)$$

– Composite trapezoid rule:

* nodes: $x_j = a + (j - \frac{1}{2})h, j = 1, 2, \dots, n, h = (b - a)/n$

* for some $\xi \in [a, b]$

$$\int_a^b f(x) dx = \frac{h}{2} [f_0 + 2f_1 + \dots + 2f_{n-1} + f_n] - \frac{h^2(b-a)}{12} f''(\xi)$$

- Simpson's Rule: piecewise quadratic approximation

– for some $\xi \in [a, b]$

$$\int_a^b f(x) dx = \left(\frac{b-a}{6}\right) \left[f(a) + 4f\left(\frac{a+b}{2}\right) + f(b) \right] - \frac{(b-a)^5}{2880} f^{(4)}(\xi)$$

– Composite Simpson's rule: for some $\xi \in [a, b]$

$$\int_a^b f(x) dx = \frac{h}{3} [f_0 + 4f_1 + 2f_2 + 4f_3 + \dots + 4f_{n-1} + f_n] - \frac{h^4(b-a)}{180} f^{(4)}(\xi)$$

- Obscure rules for degree 3, 4, etc. approximations.

Gaussian Formulas

- All integration formulas are of form

$$\int_a^b f(x) w(x) dx \doteq \sum_{i=1}^n \omega_i f(x_i) \quad (7.2.1)$$

for some *quadrature nodes* $x_i \in [a, b]$ and *quadrature weights* ω_i .

- Newton-Cotes use arbitrary x_i
 - Gaussian quadrature uses good choices of x_i nodes and ω_i weights
 - Note: $\omega_i \neq w(x_i)$
- Exact quadrature formulas:
 - Let \mathcal{F}_k be the space of degree k polynomials
 - A quadrature formula is exact of degree k if it correctly integrates each function in \mathcal{F}_k
 - Gaussian quadrature formulas use n points and are exact of degree $2n - 1$

Gauss-Chebyshev Quadrature

- Domain: $[-1, 1]$
- Weight: $(1 - x^2)^{-1/2}$
- Formula:

$$\int_{-1}^1 f(x)(1 - x^2)^{-1/2} dx = \frac{\pi}{n} \sum_{i=1}^n f(x_i) + \frac{\pi}{2^{2n-1}} \frac{f^{(2n)}(\xi)}{(2n)!} \quad (7.2.4)$$

for some $\xi \in [-1, 1]$, with quadrature nodes

$$x_i = \cos\left(\frac{2i-1}{2n}\pi\right), \quad i = 1, \dots, n. \quad (7.2.5)$$

Arbitrary Domains

- Want to approximate $\int_a^b f(x) dx$

- Different range, no weight function
- Linear change of variables $x = -1 + 2(y - a)(b - a)$
- Multiply the integrand by $(1 - x^2)^{-1/2}/(1 - x^2)^{-1/2}$.
- C.O.V. formula

$$\int_a^b f(y) dy = \frac{b - a}{2} \int_{-1}^1 f\left(\frac{(x + 1)(b - a)}{2} + a\right) \frac{(1 - x^2)^{1/2}}{(1 - x^2)^{1/2}} dx$$

- Gauss-Chebyshev quadrature produces

$$\int_a^b f(y) dy \doteq \frac{\pi(b - a)}{2n} \sum_{i=1}^n f\left(\frac{(x_i + 1)(b - a)}{2} + a\right) (1 - x_i^2)^{1/2}$$

where the x_i are Gauss-Chebyshev nodes over $[-1, 1]$.

Gauss-Hermite Quadrature

- Domain: $[-\infty, \infty]$
- Weight: e^{-x^2}
- Formula:

$$\int_{-\infty}^{\infty} f(x)e^{-x^2} dx = \sum_{i=1}^n \omega_i f(x_i) + \frac{n!\sqrt{\pi}}{2^n} \cdot \frac{f^{(2n)}(\xi)}{(2n)!}$$

for some $\xi \in (-\infty, \infty)$.

Normal Random Variables

- Y is distributed $N(\mu, \sigma^2)$
- Expectation is integration:

$$E\{f(Y)\} = (2\pi\sigma^2)^{-1/2} \int_{-\infty}^{\infty} f(y) e^{-\frac{(y-\mu)^2}{2\sigma^2}} dy$$

- Use Gauss-Hermite quadrature
 - linear COV $x = (y - \mu)/\sqrt{2} \sigma$

– COV formula:

$$\int_{-\infty}^{\infty} f(y) e^{-(y-\mu)^2/(2\sigma^2)} dy = \int_{-\infty}^{\infty} f(\sqrt{2} \sigma x + \mu) e^{-x^2} \sqrt{2} \sigma dx$$

– COV quadrature formula:

$$E\{f(Y)\} \doteq \pi^{-\frac{1}{2}} \sum_{i=1}^n \omega_i f(\sqrt{2} \sigma x_i + \mu)$$

where the ω_i and x_i are the Gauss-Hermite quadrature weights and nodes over $[-\infty, \infty]$.

Gauss-Laguerre Quadrature

- Domain: $[0, \infty]$
- Weight: e^{-x}
- Formula:

$$\int_0^{\infty} f(x)e^{-x} dx = \sum_{i=1}^n \omega_i f(x_i) + (n!)^2 \frac{f^{(2n)}(\xi)}{(2n)!}$$

for some $\xi \in [0, \infty)$.

- General integral
 - Linear COV $x = r(y - a)$
 - COV formula

$$\int_a^{\infty} e^{-ry} f(y) dy \doteq \frac{e^{-ra}}{r} \sum_{i=1}^n \omega_i f\left(\frac{x_i}{r} + a\right)$$

where the ω_i and x_i are the Gauss-Laguerre quadrature weights and nodes over $[0, \infty]$.

Multidimensional Integration

- Most economic problems have several dimensions
 - Multiple assets
 - Multiple error terms
- Multidimensional integrals are much more difficult
 - Simple methods suffer from curse of dimensionality
 - There are methods which avoid curse of dimensionality

Numerical Dynamic Programming

Discrete-Time, Infinite-Horizon Problem:

- Objective:

$$\max_{u_t} E \left\{ \sum_{t=1}^{\infty} \beta^t \pi(x_t, u_t) \right\} \quad (12.1.1)$$

- X : set of states
 - \mathcal{D} : the set of controls
 - $D(x) \subseteq \mathcal{D}$: controls which are feasible in state x .
 - $\pi(x, u)$ payoff in period t , for $x \in X$ at the beginning of period t , and control $u \in \mathcal{D}$ is applied in period t .
 - $F(A; x, u)$: probability that $x^+ \in A \subset X$ conditional on current control u and current state x .
- Value function: if $\mathcal{U}(x)$ is set of all feasible strategies starting at x .

$$V(x) \equiv \sup_{\mathcal{U}(x)} E \left\{ \sum_{t=0}^{\infty} \beta^t \pi(x_t, u_t) \mid x_0 = x \right\}, \quad (12.1.8)$$

- Bellman equation

$$V(x) = \sup_{u \in D(x)} \pi(x, u) + \beta E \{V(x^+) | x, u\} \equiv (TV)(x), \quad (12.1.9)$$

- Optimal policy function, $U(x)$:

$$U(x) \in \arg \max_{u \in D(x)} \pi(x, u) + \beta E \{V(x^+) | x, u\}.$$

Standard existence theorem:

Theorem 1 *If X is compact, $\beta < 1$, and π is bounded above and below, then the map*

$$TV = \sup_{u \in D(x)} \pi(x, u) + \beta E \{V(x^+) | x, u\} \quad (12.1.10)$$

is monotone in V , is a contraction mapping with modulus β in the space of bounded functions, and has a unique fixed point.

- Finite-horizon problems can be similarly expressed where value function depends on number of periods from end.

Deterministic Growth Example

$$\begin{aligned} V(k_0) &= \max_{c_t} \sum_{t=0}^{\infty} \beta^t u(c_t), \\ k_{t+1} &= F(k_t) - c_t \\ k_0 &\text{ given} \end{aligned} \tag{12.1.12}$$

- Euler equation:

$$u'(c_t) = \beta u'(c_{t+1}) F'(k_{t+1})$$

- Bellman equation

$$V(k) = \max_c u(c) + \beta V(F(k) - c). \tag{12.1.13}$$

- The solution to (12.1.13) is a policy function $C(k)$ and a value function $V(k)$ satisfying

$$0 = u'(C(k)) F'(k) - V'(k) \tag{12.1.15}$$

$$V(k) = u(C(k)) + \beta V(F(k) - C(k)) \tag{12.1.16}$$

- (12.1.16) defines the value of an arbitrary policy function $C(k)$, not just for the optimal $C(k)$.
- The pair (12.1.15) and (12.1.16) combines
 - the definition of the value function of a policy (12.1.16), and
 - a first-order condition for optimality (12.1.15).

Stochastic Growth Accumulation

$$\begin{aligned}
 V(k, \theta) &= \max_{c_t, \ell_t} E \left\{ \sum_{t=0}^{\infty} \beta^t u(c_t) \right\} \\
 k_{t+1} &= F(k_t, \theta_t) - c_t \\
 \theta_{t+1} &= g(\theta_t, \epsilon_t) \\
 \epsilon_t &: \text{i.i.d. random variable} \\
 k_0 &= k, \theta_0 = \theta.
 \end{aligned}$$

- State variables:
 - k : productive capital stock, endogenous
 - θ : productivity state, exogenous
- The dynamic programming formulation is

$$\begin{aligned}
 V(k, \theta) &= \max_{c, \ell} u(c, \ell) + \beta E \{ V(F(k, \theta) - c, \theta^+) | \theta \} \quad (12.1.21) \\
 \theta^+ &= g(\theta, \epsilon)
 \end{aligned}$$

- The control law $c = C(k, \theta)$ satisfies the first-order conditions

$$0 = u_c(C(k, \theta)) - \beta E \{ u_c(C(k^+, \theta^+)) F_k(k^+, \theta^+) | \theta \}, \quad (12.1.23)$$

where

$$k^+ \equiv F(k, L(k, \theta), \theta) - C(k, \theta),$$

Discrete State Space Problems

- State space $X = \{x_i, i = 1, \dots, n\}$
- Controls $\mathcal{D} = \{u_i | i = 1, \dots, m\}$
- $q_{ij}(u) = \Pr(x_{t+1} = x_j | x_t = x_i, u_t = u)$
- $Q(u) = (q_{ij}(u))_{i,j}$: Markov transition matrix at t if $u_t = u$.

Value Function iteration

- Initial guess

$$V_i^0, \quad i = 1, \dots, n.$$

- Bellman equation: iteration j from backwards iteration

$$V_i^{k+1} = \max_u [\pi(x_i, u) + \beta \sum_{j=1}^n q_{ij}(u) V_j^k], \quad i = 1, \dots, n$$

- Bellman equation can be directly computed

– Called *value function iteration*

– Infinite-horizon problems

* Can use value function iteration with arbitrary V_i^0 and iterate $k \rightarrow \infty$.

* Error is given by contraction mapping property:

$$\|V^k - V^*\| \leq \frac{1}{1 - \beta} \|V^{k+1} - V^k\|$$

Continuous states: discretization

- Method:

- “Replace” continuous X with a finite

$$X^* = \{x_i, i = 1, \dots, n\} \subset X$$

- Proceed with a finite-state method.

- Problems:

- Sometimes need to alter space of controls to assure landing on an x in X .
- A fine discretization often necessary to get accurate approximations

Continuous Methods for Continuous-State Problems

- Basic Bellman equation:

$$V(x) = \max_{u \in D(x)} \pi(u, x) + \beta E\{V(x^+) | x, u\} \equiv (TV)(x). \quad (12.7.1)$$

- Discretization essentially approximates V with a step function
- Approximation theory provides better methods to approximate continuous functions.

Approximating $V(x)$

- Choose a finite-dimensional parameterization

$$V(x) \doteq \hat{V}(x; a), \quad a \in R^m \quad (12.7.2)$$

and a finite number of states

$$X = \{x_1, x_2, \dots, x_n\}, \quad (12.7.3)$$

- polynomials with coefficients a and collocation points X
 - splines with coefficients a with uniform nodes X
 - rational function with parameters a and nodes X
 - neural network
 - specially designed functional form
- Objective: find coefficients $a \in R^m$ such that $\hat{V}(x; a)$ “approximately” satisfies the Bellman equation.

Approximating T

For each x_j , $(TV)(x_j)$ is defined by

$$v_j = (TV)(x_j) = \max_{u \in D(x_j)} \pi(u, x_j) + \beta \int \hat{V}(x^+; a) dF(x^+ | x_j, u) \quad (12.7.5)$$

In practice, we compute the approximation \hat{T}

$$v_j = (\hat{T}V)(x_j) \doteq (TV)(x_j)$$

- Integration step: for ω_j and x_j for some numerical quadrature formula

$$\begin{aligned} E\{V(x^+; a) | x_j, u\} &= \int \hat{V}(x^+; a) dF(x^+ | x_j, u) \\ &= \int \hat{V}(g(x_j, u, \epsilon); a) dF(\epsilon) \\ &\doteq \sum_{\ell} \omega_{\ell} \hat{V}(g(x_j, u, \epsilon_{\ell}); a) \end{aligned}$$

- Maximization step: for $x_i \in X$, evaluate

$$v_i = (T\hat{V})(x_i)$$

- Fitting step:

- Data: (v_i, x_i) , $i = 1, \dots, n$
- Objective: find an $a \in R^m$ such that $\hat{V}(x; a)$ best fits the data
- Methods: determined by $\hat{V}(x; a)$

Value Function Iteration

$$\begin{aligned} \text{guess } a &\longrightarrow \hat{V}(x; a) \\ &\longrightarrow (v_i, x_i), \quad i = 1, \dots, n \\ &\longrightarrow \text{new } a \end{aligned}$$

- Comparison with discretization
 - This procedure examines only a finite number of points, but does *not* assume that future points lie in same finite set.
 - Our choices for the x_i are guided by systematic numerical considerations.
- Synergies
 - Smooth interpolation schemes allow us to use Newton's method in the maximization step.
 - They also make it easier to evaluate the integral in (12.7.5).

- Convergence
 - T is a contraction mapping
 - \hat{T} may be neither monotonic nor a contraction

- Shape problems
 - An Instructive Example

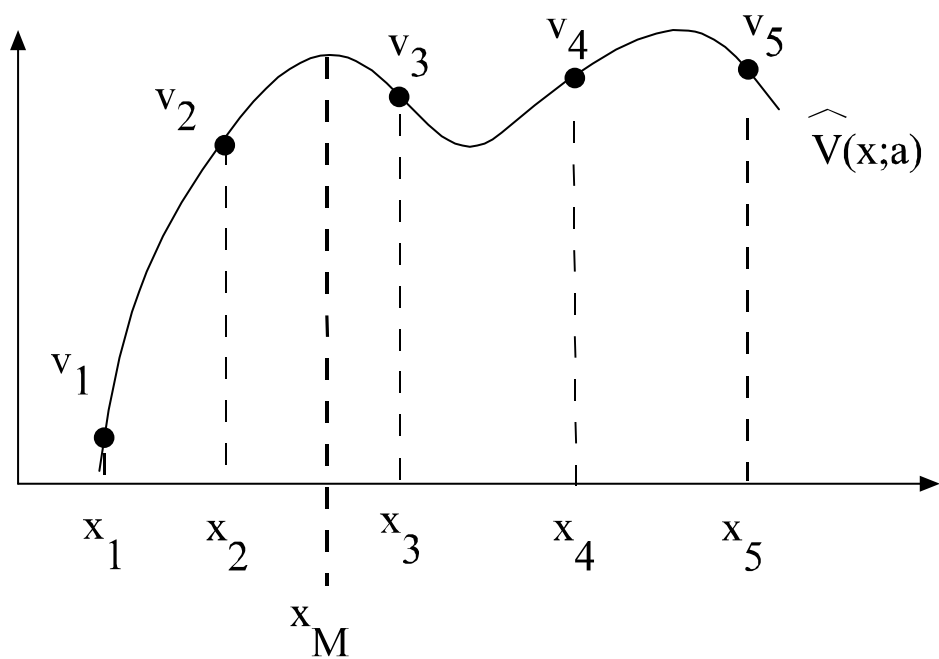


Figure 4:

- Shape problems may become worse with value function iteration
- Shape-preserving approximation implies monotonicity

Comparisons

We apply various methods to the deterministic growth model

N	Relative L2 Errors over $[0.7,1.3]$					
	$(\beta, \gamma) :$					
	$(.95,-10.)$	$(.95,-2.)$	$(.95,-.5)$	$(.99,-10.)$	$(.99,-2.)$	$(.99,-.5)$
	Discrete model					
12	7.6e-02	2.8e-03	5.3e-03	7.9e-01	1.8e-01	1.1e-02
120	1.2e-03	2.2e-04	5.3e-04	4.5e-02	6.6e-02	1.3e-03
1200	1.0e-04	2.1e-05	5.4e-05	2.9e-03	5.4e-03	1.3e-04
	Linear Interpolation					
4	7.9e-03	4.1e-03	2.4e-03	8.0e-03	4.1e-03	2.4e-03
12	1.5e-03	9.8e-04	5.6e-04	1.5e-03	1.0e-03	6.3e-04
40	3.9e-04	2.3e-04	9.8e-05	4.2e-04	2.8e-04	1.6e-04
120	1.1e-04	3.7e-05	1.3e-05	1.4e-04	8.4e-05	4.2e-05
	Cubic Spline					
4	6.6e-03	5.0e-04	1.3e-04	7.1e-03	5.7e-04	1.8e-04
12	8.7e-05	1.5e-06	1.8e-07	1.3e-04	4.9e-06	1.1e-06
40	7.2e-08	1.8e-08	5.5e-09	7.6e-07	8.8e-09	4.9e-09
120	5.3e-09	5.6e-10	1.3e-10	4.2e-07	4.1e-09	1.5e-09
	Polynomial					
4	DNC	5.4e-04	1.6e-04	1.4e-02	5.6e-04	1.7e-04
12	3.0e-07	2.0e-09	4.3e-10	5.8e-07	4.5e-09	1.5e-09
	Shape Preserving Quadratic Interpolation					
4	1.1e-02	3.8e-03	1.2e-03	2.2e-02	7.3e-03	2.2e-03
12	6.7e-04	1.1e-04	3.1e-05	1.2e-03	2.1e-04	5.7e-05
40	3.5e-05	5.8e-06	8.3e-07	4.5e-05	9.3e-06	3.0e-06
120	2.5e-06	1.5e-07	2.2e-08	4.3e-06	8.5e-07	1.9e-07

Summary:

- Discretization methods
 - Easy to implement
 - Numerically stable
 - Poor approximation to continuous problems
 - Curse of dimensionality
- Continuous approximation methods
 - Can exploit smoothness in problems
 - Possible numerical instabilities
 - Better able to handle multidimensional problems