

Numerical Methods in Economics

Kenneth L. Judd

Value function iteration p. 102 Mac.

The MIT Press
Cambridge, Massachusetts
London, England

9

Quasi-Monte Carlo Methods

We next discuss *quasi-Monte Carlo* methods for integration and simulation. In contrast to Monte Carlo ideas, they rely on ideas from number theory and Fourier analysis, and they are often far more powerful than Monte Carlo methods. In this chapter we present the basic quasi-Monte Carlo ideas and methods.¹ Before we get to the details, we first make some general points.

To understand the basic idea behind quasi-Monte Carlo methods, it is useful to realize what Monte Carlo methods really are in practice. As we pointed out in chapter 8, pseudorandom sequences are generally used in practice instead of truly “random” (whatever that means) sequences. Despite their similarities, pseudorandom sequences are deterministic, not random, sequences. Mathematicians have long known this; according to John von Neumann, anyone who believes otherwise “is living in a state of sin.” Since they are not probabilistically independent, neither the law of large numbers nor the central limit theorem² apply to pseudorandom sequences. The fact that neither linear statistical techniques nor the human eye can distinguish between random numbers and pseudorandom number sequences is of no logical relevance and has no impact on mathematical rates of convergence.

These issues are often confusingly discussed in the Monte Carlo literature. For example, Kloek and van Dijk (1978) contrast the Monte Carlo convergence rate of $N^{-1/2}$ with Bahravov’s theorem showing that the convergence rate for deterministic schemes for d -dimensional C^1 integrable functions is $N^{-1/d}$ at best. However, they then go on to use pseudorandom number generators in their Monte Carlo experiments. This is typical of the Monte Carlo literature where probability theorems are often used to analyze the properties of algorithms that use *deterministic* sequences. Strictly speaking, procedures that use pseudorandom numbers are not Monte Carlo schemes; when it is desirable to make the distinction, we will refer to Monte Carlo schemes using pseudorandom sequences as *pseudo-Monte Carlo* (pMC) methods.

Even though experimental evidence of pseudo-Monte Carlo methods shows that they work as predicted by the Monte Carlo theory, this does not validate the invalid use of probabilistic arguments. Pseudorandom methods are instead validated by a substantial literature (Niederreiter 1992 and Traub and Wozniakowski 1992 are recent good examples) which explicitly recognizes the determinism of pseudorandom sequences. These analyses rely on the properties of the deterministic pseudorandom

1. In this chapter we frequently use Fourier expansions wherein the letter i will denote the square root of -1 . To avoid confusion, we do not use it as an index of summation.

2. I here refer to the law of large numbers as stated in Chung (1974, theorem 5.2.2), and the central limit theorem as stated in Chung (1974, theorem 6.4.4). These theorems rely on independence properties of random variables, not just on zero correlation. There may exist similar theorems that rely solely on the serial correlation and runs properties of pseudorandom sequences, but I don’t know of any.

generators, not on probability theory. More important, since pseudorandom sequences are deterministic, their success tells us that there are deterministic sequences that are good for numerical integration. Furthermore the random versus deterministic distinction is of no practical importance in numerical analysis. In fact it is damaging, for it may keep a reader from considering other deterministic schemes. Therefore the real questions for our purposes are, How can *deterministic* sequences yield good approximations to integrals, which *deterministic* sequences yield the most rapidly convergent quadrature schemes, and how can good *deterministic* sequences be constructed? This perspective leads to a new set of concepts, and also to *quasi-Monte Carlo* (qMC) methods.

These observations still raise a logical problem: Why do the *deterministic* schemes used in Monte Carlo applications satisfy the $N^{-1/2}$ prediction of *probability* theorems, and avoid the $N^{-1/d}$ convergence rate of *deterministic* schemes? There is no problem once one knows the norms that are being used in these comparisons. Monte Carlo methods have an *expected* error proportional to $N^{-1/2}$, whereas the *worst-case* error for deterministic schemes is $N^{-1/d}$. Furthermore, if we confine our attention to functions of bounded variation, as is usually the case for applications of Monte Carlo methods, then there exist deterministic schemes with convergence rate N^{-1} . In this chapter we present a unified analysis of error bounds for Monte Carlo, pseudo-Monte Carlo, and quasi-Monte Carlo schemes and find that this confusion disappears when one uses the *same* metric and focus on relevant spaces of functions to compare alternative methods.

When we dispense with the irrelevant random versus deterministic distinction, we find that quasi-Monte Carlo methods schemes do far better asymptotically than any Monte Carlo method for many problems. For example, in September 1995, IBM announced that it had developed software for pricing financial derivatives based on quasi-Monte Carlo methods that outperformed Monte Carlo methods by a factor of 1,000.

Before proceeding, we make two distinctions. First, a *sampling* method is any method that “samples” the domain of integration and uses an (usually unweighted) average of the integrand’s values at the sample as an estimate of an integral’s value. The Monte Carlo method is a sampling method. Second, we define quasi-Monte Carlo methods in a negative way: *Quasi-Monte Carlo methods* are sampling methods that do not rely on probabilistic ideas and pseudorandom sequences for constructing the sample and analyzing the estimate. Quasi-Monte Carlo methods are frequently called *number theoretic methods* because they often explicitly rely on number theory ideas. The term “quasi-Monte Carlo” is misleading because the methods covered under that umbrella have nothing to do with probability theory; on the other hand,

the use of the prefix “pseudo”—which, according to *Webster’s Third New International Dictionary*, means “false, feigned, fake, counterfeit, spurious, illusory”—to describe pseudorandom numbers is totally appropriate. The key idea is sampling, and the various methods, pseudo- and quasi-, are distinguished by the particular deterministic scheme used to generate the sample.

9.1 Equidistributed Sequences

Up to nonlinear changes of variables, Monte Carlo integration methods ultimately rely on the properties of sequences of uniformly distributed i.i.d. random variables. The ex ante uniformity of the samples make the Monte Carlo estimate of an integral unbiased. However, probability theory does not apply to pseudorandom sequences. In order to analyze deterministic sampling schemes, including pseudorandom numbers, we need to develop concepts for analyzing errors and convergence.

We next introduce the concept for deterministic sequences which (up to measure zero) generalizes the key properties of Monte Carlo samples.

DEFINITION A sequence $\{x_j\}_{j=1}^{\infty} \subset R$ is *equidistributed* over $[a, b]$ iff

$$\lim_{n \rightarrow \infty} \frac{b-a}{n} \sum_{j=1}^n f(x_j) = \int_a^b f(x) dx \quad (9.1.1)$$

for all Riemann-integrable $f(x)$. More generally, a sequence $\{x^j\}_{j=1}^{\infty} \subset D \subset R^d$ is *equidistributed* over D iff

$$\lim_{n \rightarrow \infty} \frac{\mu(D)}{n} \sum_{j=1}^n f(x^j) = \int_D f(x) dx \quad (9.1.2)$$

for all Riemann-integrable $f(x) : R^d \rightarrow R$, where $\mu(D)$ is the Lebesgue measure of D .

Equidistribution formalizes the idea of a uniformly distributed sequence. One key property is that an equidistributed sequence is ex post uniformly distributed, not just uniform in expectation. Equidistribution is exactly the property that a sequence will give us an asymptotically valid approximation to any integral. Here we focus directly on constructing sequences that do well at integration.

Unfortunately, it is not easy to construct equidistributed sequences. For example, one might think that the sequence $0, \frac{1}{2}, 1, \frac{1}{4}, \frac{3}{4}, \frac{1}{8}, \frac{3}{8}, \frac{5}{8}, \frac{7}{8}$, etc., would be equidistributed over $[0, 1]$. This is not the case, since the finite sums $\frac{1}{n} \sum_{j=1}^n f(x_j)$ for, say, $f(x) = x$, do not converge to $\int_0^1 x dx = \frac{1}{2}$, nor to anything else. Certain subsequences of the finite sums will converge properly; in particular, if we average over the first 5 terms,

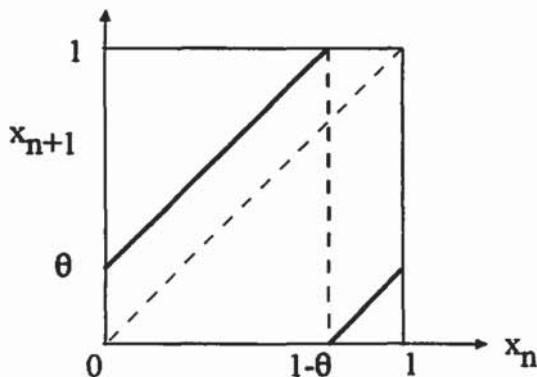


Figure 9.1
Graph of a Weyl rule

then the first 9, then 17, 33, ..., $2^k + 1$, etc., the averages will converge to $\frac{1}{2}$. Hence the true integral is a limit point. However, if we average over the first 7, then 13, 25, etc., the limit is $\frac{3}{8}$. This is because the first 7 terms contain two terms greater than 0.5 and four below 0.5, and a similar two-to-one ratio holds for the first 13, 25, 49, etc. The definition of equidistributed sequences, however, requires convergence of the sequence *and all subsequences* of finite sums to the integral, not just a subsequence. The failure of this simple sequence to be equidistributed shows that equidistribution is a strong property.

The simplest example of a sequence equidistributed over $[0, 1]$ is a *Weyl sequence*

$$x_n = \{n\theta\}, \quad n = 1, 2, \dots, \quad (9.1.3)$$

for θ irrational, where the expression $\{x\}$ is called the *fractional part of x* and is defined by $\{x\} \equiv x - \max\{k \in \mathbb{Z} \mid k \leq x\}$. The $\{x\}$ notation is a bit odd since $\{\cdot\}$ has other uses. In this chapter we try to avoid confusion by limiting use of $\{\cdot\}$ to this fractional part function. Figure 9.1 displays the graph of $x_{n+1} = f(x_n)$ implied by (9.1.3). Note that the slope is unity and there is only one discontinuity. This function is notably different from the typical LCM generator in figure 8.1.

One should be careful when calculating $\{x\}$. This calculation is determined by the digits of x of lowest significance, making it very sensitive to roundoff error. For example, suppose that we have a machine with six-digit precision. To six digits, $\sqrt{2} = 1.41421$. On such a machine, $\{100\sqrt{2}\} = 0.421$, which is only the first three significant digits of the six-digit accurate value 0.421356. In general, if $\theta \in [0, 1]$, $\{n\theta\}$ will have $\log_{10} n$ fewer significant digits than machine precision. Such sequences will have successively less precision as n increases. If this is a concern, then $\{n\theta\}$ should be calculated using quadruple precision or multiple precision software.

Table 9.1
Equidistributed sequences in R^d

Name	Formula for x^n
Weyl	$(\{np_1^{1/2}\}, \dots, np_d^{1/2})$
Haber	$\left(\left\{\frac{n(n+1)}{2} p_1^{1/2}\right\}, \dots, \left\{\frac{n(n+1)}{2} p_d^{1/2}\right\}\right)$
Niederreiter	$(\{n2^{1/(d+1)}\}, \dots, \{n2^{d/(d+1)}\})$
Baker	$(\{ne^{r_1}\}, \dots, \{ne^{r_d}\}), r_j$ rational and distinct

This example can be generalized to construct multidimensional equidistributed sequences. If θ_j , $j = 1, \dots, d$, are linearly independent irrational numbers over the rationals, that is, if $\forall \alpha \in Z^d$ ($\sum_{j=1}^d \alpha_j \theta_j \neq 0$), then the points

$$x^n = (\{n\theta_1\}, \dots, \{n\theta_d\}), \quad n = 1, 2, \dots,$$

are equidistributed over $[0, 1]^d$.

Many equidistributed sequences rely on the properties of primes; let p_1, p_2, \dots , denote the sequence of prime numbers 2, 3, 5, Table 9.1 contains a number of examples of equidistributed sequences.

The connections between equidistributed sequences and Monte Carlo ideas should be noted. First, the law of large numbers essentially says that a sequence of i.i.d. draws is almost surely equidistributed. However, pseudorandom sequences, such as those of the linear congruential form, are not equidistributed since they eventually cycle. Second, the sequence (9.1.3) can be generated by the iteration $x_{n+1} = (x_n + \theta) \bmod 1$, a form similar to the schemes used to generate pseudorandom numbers.

The more relevant distinction between pseudorandom sequences and equidistributed sequences is that there is no attempt to make equidistributed sequences "look like" random numbers. In fact they generally display substantial nonzero serial correlation. We begin to see why quasi-Monte Carlo methods may outperform Monte Carlo methods when it comes to integration. Equidistributed sequences are designed from the outset to do integration accurately, and they are not encumbered by other requirements associated with "randomness." The only purpose served by the "random" character of pseudorandom sequences is that it tempts us to invoke the law of large numbers in convergence arguments. If there is some other way of showing (9.1.1) or (9.1.2), then we can jettison "randomness," which is needed for applications of LLN and CLT. In fact we will see that by attacking the integration problem directly we will do much better than Monte Carlo.

9.2 Low-Discrepancy Methods

In Monte Carlo methods the ex post sample is not uniformly distributed. The deviations from uniformity lead to errors proportional to $N^{-1/2}$, in accord with the CLT. If deterministic pseudo-Monte Carlo methods have a similar error, then something other than the CLT is at work. We saw in the previous section that equidistribution was the key to a valid quadrature method. In this section we introduce the concepts that are used to analyze the properties of all deterministic sequences, pseudorandom and quasi-Monte Carlo, for integration. These concepts allow us to rank various equidistributed sequences and compare them to Monte Carlo sequences.

Discrepancy

We need to define a measure of how dispersed a collection of points is; this leads us to concepts of *discrepancy*. Let $x_j \in I \equiv [0, 1]$, $j = 1, \dots, N$, be a sequence of scalars. If $S \subset I$, define the cardinality of a set X in a set S , $\text{card}(S \cap X)$, to be the number of elements of X which are also in S . We define two notions of discrepancy for finite sets.³

DEFINITION The *discrepancy* D_N of the set $X \equiv \{x_1, \dots, x_N\} \subset [0, 1]$ is

$$D_N(X) = \sup_{0 \leq a < b \leq 1} \left| \frac{\text{card}([a, b] \cap X)}{N} - (b - a) \right|.$$

DEFINITION The *star discrepancy* D_N^* of the set $X \equiv \{x_1, \dots, x_N\} \subset [0, 1]$ is

$$D_N^*(X) = \sup_{0 \leq t \leq 1} \left| \frac{\text{card}([0, t] \cap X)}{N} - t \right|$$

We can extend these concepts to sequences.

DEFINITION If X is a sequence $x_1, x_2, \dots \subset [0, 1]$, then $D_N(X)(D_N^*(X))$ is $D_N(X^N)$ ($D_N^*(X^N)$) where $X^N = \{x_j \in X | j = 1, \dots, N\}$.

These definitions allow us to measure the deviations from uniformity of both sets and initial segments of sequences. We focus on the behavior of D_N and D_N^* as N increases. The difference between D_N and D_N^* is that D_N examines all subintervals, whereas D_N^* examines only initial segments, $[0, t]$. Even though a continuum of

3. Authors differ in their definitions of D and D^* ; we use the definitions in Niedereitter (1992). Despite the possible confusion, all results concerning rates of convergence are the same for D^* and D .

intervals is used in the definitions, we need only to check open intervals of the form (x_l, x_j) , $1 \leq l, j \leq N$, for D_N , and $(0, x_l)$, $1 \leq l \leq N$, for D_N^* . The star discrepancy is far easier to compute, but both are useful concepts measuring dispersion. The smallest possible value for D_N and D_N^* is $1/(N+1)$, which is the discrepancy of $\{1/(N+1), 2/(N+1), \dots, N/(N+1)\}$, the most dispersed set of N points possible in $[0, 1]$. The dimension one theory of discrepancy is trivial. We next consider multi-dimensional versions of these concepts.

DEFINITION The star discrepancy D_N^* of the set $X = \{x_1, \dots, x_N\} \subset I^d$ is

$$D_N^*(X) = \sup_{0 \leq t_1, \dots, t_d \leq 1} \left| \frac{\text{card}([0, t_1] \times \dots \times [0, t_d] \cap X)}{N} - \prod_{j=1}^d t_j \right|$$

and $D_N^*(X)$ for sequences in I^d is defined as in the case $d = 1$. $D_N(X)$ is generalized similarly from $d = 1$.

A small discrepancy says that the sequence evenly fills up the hypercube I^d . As N increases, the discrepancy should go down. Not all equidistributed sequences have the same discrepancy. We want to find equidistributed sequences with initial segments having small discrepancy, a property that will lead to smaller errors in integration applications.

One set of particular interest is the uniformly distributed lattice points. More specifically, if we define

$$U_{d,m} = \left\{ \left(\frac{2m_1 - 1}{2m}, \dots, \frac{2m_d - 1}{2m} \right) \middle| 1 \leq m_j \leq m, j = 1, \dots, d \right\}$$

to be the lattice with m points in each of the d directions in I^d , then $U_{d,m}$ has $N = m^d$ points and $D_N^*(U_{d,m}) = \mathcal{O}(N^{-1/d}) = \mathcal{O}(m^{-1})$. (See Hua and Wang 1981.) For the uniform lattice the discrepancy falls very slowly as the number of points increases, and more slowly as d increases.

Random sequences have smaller discrepancies than $U_{d,m}$ on average. Let x^j , $j = 1, \dots, N$, be an i.i.d. sequence uniformly and independently distributed over I^d . Chung (1949) and Kiefer (1961) prove that the star discrepancy of such a collection of N points is almost surely $\mathcal{O}(N^{-1/2}(\log \log N)^{1/2})$, where the proportionality constant depends on dimension. Note that this is essentially $\mathcal{O}(N^{-1/2})$, the familiar Monte Carlo rate of convergence.

There are also results that put lower limits on D_N^* . Roth (1954) and Kuipers and Niederreiter (1974) showed that for any collection of N points in I^d ,

$$D_N^* > 2^{-4d}((d-1)\log 2)^{(1-d)/2} N^{-1} (\log N)^{(d-1)/2}. \quad (9.2.1)$$

Notice that this lower bound is considerably below the Chung-Kiefer result on randomly generated point sets. This bound applies to all sets, even to those generated “randomly.”

No one has yet constructed a sequence that achieves the lower bound in (9.2.1). Some good sequences are constructed using the *radical-inverse function*. Suppose that p is a prime number. Any positive integer n has a base p expansion $n = \sum_{j=0}^k a_j p^j$. Given the coefficients, a_j , the radical-inverse function, $\varphi_p(n)$, is defined by

$$\varphi_p(n) = \sum_{j=0}^k a_j p^{-j-1}. \quad (9.2.2)$$

The *van der Corput sequence* is the sequence $\varphi_2(n)$ for $n = 1, 2, \dots$, and has discrepancy $D_N^* < (\frac{1}{3} \log_2 N + 1)/N$.

In d dimensions, $d \geq 2$, the *Hammersley set* of N points is the set

$$\left\{ \left(\frac{n}{N}, \varphi_{p_1}(n), \dots, \varphi_{p_{d-1}}(n) \right) \middle| n = 1, \dots, N \right\},$$

where p_1, \dots, p_{d-1} are $d - 1$ relatively prime integers. Because it is finite, the Hammersley set is not an equidistributed sequence. The *Halton sequence* in I^d is

$$\{(\varphi_{p_1}(n), \varphi_{p_2}(n), \dots, \varphi_{p_d}(n))\}_{n=1}^\infty, \quad (9.2.3)$$

where p_1, \dots, p_d are d distinct primes, and has discrepancy

$$ND_N < \frac{d}{N} + \prod_{j=1}^d \left(\frac{p_j - 1}{2 \log p_j} \log N + \frac{p_j + 1}{2} \right). \quad (9.2.4)$$

The discrepancy bounds in (9.2.1), (9.2.4) hold for all N , but is useful only for their asymptotic information since $(\log N)^d$ can be quite large. In fact, for a fixed value of d , $(\log N)^d$ will grow faster than N for moderate N , implying that D_N can be small for only very large N . Therefore, while the asymptotic properties of these bounds are good, they kick in at only impractically large N if the dimension d is not small.

Figure 9.2 displays a two-dimensional set of equidistributed points. Figure 9.2 is a plot of the first 1,500 Weyl points with $d = 2$. Note that this sample is more uniform than the set of pseudorandom points seen earlier in figure 8.2. This illustrates the point that quasi-random points aim to construct a uniform set of points, not one that looks random.

The weakness of these simple equidistributed sequences lies in their poor small sample properties. Of course some pseudo-Monte Carlo methods have similar

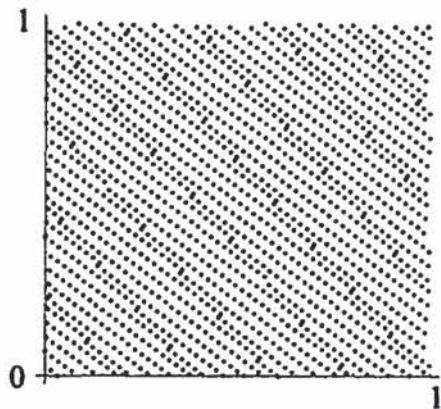


Figure 9.2
Initial 1,500 Weyl points

problems. It may appear easier to avoid small sample problems with pseudo-Monte Carlo schemes, but new quasi-Monte Carlo methods, cited in section 9.7, produce good small samples.

Variation and Integration

To apply the notions of discrepancy to integration, we need a measure of how variable a function is. This is easy in the one-dimensional case.

DEFINITION The *total variation* of f , $V(f)$, on $[0, 1]$ is

$$V(f) = \sup_n \sup_{0 \leq x_0 < x_1 < \dots < x_n \leq 1} \sum_{j=1}^n |f(x_j) - f(x_{j-1})|.$$

The next theorem ties together the total variation and the discrepancy concepts to compute an upper bound on a quadrature formula.

THEOREM 9.2.1 (Koksma) If f has bounded total variation, namely $V(f) < \infty$, on I , and the sequence $x_j \in I$, $j = 1, \dots, N$, has discrepancy D_N^* , then

$$\left| N^{-1} \sum_{j=1}^N f(x_j) - \int_0^1 f(x) dx \right| \leq D_N^* V(f) \quad (9.2.5)$$

The quadrature approximation $N^{-1} \sum_{j=1}^N f(x_j)$ in theorem 9.2.1 is essentially the composite midpoint rule except for the fact that here the x_j nodes need not be evenly distributed. However, (9.2.5), tells us that the best choice of $\{x_1, \dots, x_N\}$ would be the uniform collection since that set has the smallest D_N^* . Therefore the one-dimensional case leads us essentially back to the midpoint rule.

Note the kind of the quadrature error analysis implicit in Koksma's theorem. Monte Carlo methods use the variance of a function to get probabilistic estimates of error. Quasi-Monte Carlo methods use variation measures to generate worst-case error bounds. Since Monte Carlo methods also use quadrature approximations of the kind in (9.2.5), the worst-case error bound in (9.2.5) also applies to Monte Carlo errors. This observation gives us more information about Monte Carlo integration. Koksma's theorem combined with the Chung-Kiefer bound on D_N^* of a random set of points in R tells us that the worst-case error for Monte Carlo integration is almost surely $\mathcal{O}(N^{-1/2}(\log \log N)^{1/2})$. Of course the absolute worst-case error is $\mathcal{O}(1)$, since there are sequences that won't converge.

While these results are aesthetically interesting and provide insight as to how these methods work, they have little practical value since the convergence rate is the same as for the midpoint rule. No one seriously uses Monte Carlo or quasi-Monte Carlo methods for one-dimensional integrals of smooth functions.

The real interest lies in multidimensional applications. To extend these ideas to R^d , we need a multidimensional version of total variation. Suppose that $H = [a_1^1, a_2^1] \times [a_1^2, a_2^2] \times \dots \times [a_1^d, a_2^d]$ is a subset of I^d . A measure of how much f varies over H is

$$\Delta(f; H) \equiv \sum_{j_1=1}^2 \dots \sum_{j_d=1}^2 (-1)^{j_1+\dots+j_d} f(a_{j_1}^1, \dots, a_{j_d}^d).$$

A partition \mathcal{P} of I^d is any collection of hyper-rectangles that is pairwise disjoint but covers I^d . Our first measure of total variation is

$$V^V(f) = \sup_{\mathcal{P}} \sum_{H \in \mathcal{P}} |\Delta(f; H)|.$$

If f has enough derivatives, then $V^V(f) = \int_{I^d} |\partial^d f / \partial x_1 \dots \partial x_d| dx$. If $V^V(f)$ is finite, then f is of bounded variation on I^d in the sense of Vitali.

From the derivative version, we immediately see a problem with $V^V(f)$. The d th cross-partial of f for $d > 2$ will be zero even for functions that vary substantially, such as $\sum_{j=1}^{d-1} x_j x_{j+1}$. The Vitali concept of variation ignores variation on lower-dimensional subspaces. To address this problem, we measure $V^V(f)$ on those lower-dimensional subspaces and include it in the measure of total variation. Suppose that $J \subset \{1, \dots, d\}$. Define f^J to be the function equal to f but restricted to $I^J \equiv \{x \in I \mid x_j = 1, j \in J\}$. If J is the empty set, f^J is f ; otherwise, f^J is the restriction of f to a $(d - |J|)$ -dimensional subspace of I^d , and $V^V(f^J)$ is the variation of f on that subspace. The variation in the sense of Hardy and Krause is defined by the formula

$$V^{HK}(f) = \sum_{J \subset \{1, \dots, d\}} V^V(f^J).$$

The following theorem generalizes Koksma to multiple dimensions.

THEOREM 9.2.2 (Hlawka) If $V^{HK}(f)$ is finite and $\{x^j\}_{j=1}^N \subset I^d$ has discrepancy D_N^* , then

$$\left| \frac{1}{N} \sum_{j=1}^N f(x^j) - \int_{I^d} f(x) dx \right| \leq V^{HK}(f) D_N^*.$$

The Hlawka theorem is a much more useful result. Again it tells us to use sets with small D_N^* , but that does not point to any simple rule we saw before. The product trapezoid rule essentially uses the uniform grid point set, $U_{d,m}$, with star-discrepancy $\mathcal{O}(N^{-1/d})$. Instead, we should use one of the sequences that had much smaller discrepancies, such as the Halton sequence or one of those in table 9.1.

Monte Carlo versus Quasi-Monte Carlo

We next compare “stochastic” sampling methods and quasi-Monte Carlo methods of sampling. In terms of asymptotic performance for smooth functions of bounded variation, there is no contest. For all dimensions d we know of sequences such that the rate of convergence of the worst-case error for quasi-Monte Carlo methods is of order $N^{-1}(\log N)^d$, whereas the *expected* error of Monte Carlo integration methods is of order $N^{-1/2}$.

Furthermore, even though the worst-case measures used to evaluate quasi-Monte Carlo methods differ in spirit from the average-case measure implicitly used in Monte Carlo analyses, there is no difference for this problem. Wozniakowski (1991) showed that the average convergence rate for the optimal deterministic sampling method is the same as the worst-case rate, that being $N^{-1}(\log N)^{(d-1)/2}$.

We have examined the theoretical issue of rate of convergence of Monte Carlo and quasi-Monte Carlo sampling schemes. We will next show that these asymptotic results do appear at reasonable sample sizes and are of practical importance.

Tables 9.2 and 9.3 concretely illustrate our asymptotic results. These tables examine two integrals, displaying the errors of various quadrature formulas for versions of each in various dimensions. In table 9.2 we examine an integral that has no interaction across variables but does contain kinks. First note that for $d = 10$, the errors appear to be following the predicted asymptotic behavior, with all procedures converging at roughly rate N^{-1} . At both dimensions we see that the Monte Carlo procedure (we invoked a library Fortran random number generator) was inferior to

Table 9.2
Integration errors for $\int_{I^d} d^{-1} \sum_{j=1}^d |4x_j - 2| dx$

N	MC	Weyl	Haber	Niederreiter
$d = 10$				
10^3	1(-3)	3(-4)	4(-4)	4(-4)
10^4	2(-4)	6(-5)	1(-3)	3(-5)
10^5	1(-3)	7(-6)	2(-4)	2(-6)
10^6	4(-5)	6(-7)	2(-4)	2(-7)
$d = 40$				
10^3	3(-3)	4(-4)	3(-3)	2(-4)
10^4	3(-4)	6(-5)	1(-3)	2(-6)
10^5	4(-6)	5(-6)	3(-4)	9(-6)
10^6	1(-4)	6(-7)	1(-5)	4(-7)

Table 9.3
Integration errors for $\int_{I^d} \prod_{j=1}^d \left(\frac{\pi}{2} \sin \pi x_j\right) dx$

N	MC	Weyl	Haber	Niederreiter
$d = 10$				
10^3	1(-2)	6(-2)	8(-2)	9(-3)
10^4	3(-2)	8(-3)	5(-3)	5(-4)
10^5	9(-3)	2(-3)	1(-3)	6(-4)
10^6	2(-3)	3(-5)	6(-3)	2(-4)
$d = 40$				
10^3	4(-1)	5(-1)	5(-2)	7(-1)
10^4	2(-1)	4(-1)	4(-1)	8(-2)
10^5	1(-2)	2(-1)	3(-3)	5(-2)
10^6	3(-2)	2(-1)	3(-2)	4(-3)

two of the three quasi-random sequences. All procedures did less well for the high-dimension case, but the Monte Carlo procedure seems to be the most sensitive to the dimensionality problems, whereas the Haber and Niederreiter quasi-random sequences have small errors converging at the theoretical rate even for $d = 40$.

Table 9.3 displays results for integrals with more interactions and higher total variation. For $d = 10$ the errors are small and falling, but for $d = 40$ it appears that the asymptotic bounds are not operating except for the largest sample sizes. Monte Carlo seems to be the best behaved in terms of avoiding very large errors but begins to be dominated by quasi-random sequences at large samples. The Niederreiter sequence appears to be the most reliable one, almost always dominating Monte Carlo for large samples.

Tables 9.2 and 9.3 apply equidistributional sequences to integration problems. They can also be used for optimization problems. Section 8.3 discussed using pseudorandom sequences to solve optimization problems by search. Equidistributional sequences can be substituted for the pseudorandom sequences and optimization implemented by *quasi-random search*; see Niederreiter and McCurley (1979) for a discussion.

These experiments are consistent with the theory in that Monte Carlo schemes are dominated by quasi-Monte Carlo schemes asymptotically but may enjoy finite sample advantages. The finite sample advantages of Monte Carlo methods are greater as the dimension increases. Therefore the choice of method depends on the dimension of the problem and the sample size used. In the past, such as when Davis and Rabinowitz (1984) wrote, sensible sample sizes were small. As computing speeds increase, the typical sample size will increase, likely increasing the dominance of quasi-random sequences over Monte Carlo methods. Also, as we develop equidistributional schemes better than the simple ones used in tables 9.2 and 9.3, even these small sample advantages of Monte Carlo schemes will be reduced and likely disappear.

9.3 Fourier Analytic Methods

Quasi-Monte Carlo integration methods which rely on low-discrepancy sequences can be used for any integrable function of finite variation. Despite these weak restrictions the results of the previous section show that small discrepancy quasi-Monte Carlo methods are able to construct integration methods with $\mathcal{O}(N^{-1+\varepsilon})$ convergence rates for any positive ε .

However, the methods of the previous section are often not the best for our purposes. In particular, their finite sample behavior is not acceptable for large dimensions. Fortunately we can do better by exploiting a function's properties. Most of the functions we use in economics are much better behaved than just being of bounded variation. Bringing this good behavior to our analysis can dramatically improve sampling approaches to quadrature just as it did in Gaussian quadrature. In this section we will see that we can do much better when we apply Fourier methods to periodic integrands.

The power of Fourier methods is illustrated by the integral $\int_0^1 \cos 2\pi x dx = 0$ and its finite-sum approximation $N^{-1} \sum_{n=1}^N \cos 2\pi x_n$, where $x_n = \{n\alpha\}$. The periodicity of $\cos x$ implies that $\cos 2\pi\{n\alpha\} = \cos 2\pi n\alpha$ and allows us to drop the fractional part

notation from the finite-sum expressions. Periodicity of $\cos 2\pi x$ also implies the existence of a convergent Fourier series representation; from de Moivre's theorem, the Fourier series for $\cos 2\pi x$ is $\cos 2\pi x = \frac{1}{2}(e^{2\pi i x} + e^{-2\pi i x})$. Since the integral $\int_0^1 \cos 2\pi x \, dx$ equals zero, the magnitude of any finite-sum approximation is the error and equals

$$\begin{aligned} \left| \frac{1}{N} \sum_{n=1}^N \frac{1}{2}(e^{2\pi i n \alpha} + e^{-2\pi i n \alpha}) \right| &\leq \frac{1}{2N} \left(\left| \frac{e^{2\pi i N \alpha} - 1}{e^{2\pi i \alpha} - 1} \right| + \left| \frac{e^{-2\pi i N \alpha} - 1}{e^{-2\pi i \alpha} - 1} \right| \right) \\ &\leq \frac{1}{2N} \left| \frac{2}{|e^{2\pi i \alpha} - 1|} + \frac{2}{|e^{-2\pi i \alpha} - 1|} \right| \leq \frac{C}{N} \end{aligned} \quad (9.3.1)$$

for a finite C as long as $e^{2\pi i \alpha} \neq 1$, which is true for any irrational α . Since the error is at most C/N , the convergence rate is N^{-1} . This is much better than the $N^{-1/2}$ promised by Monte Carlo methods, and also better than the $N^{-1} \ln N$ rate of using any of the sequences in table 9.1. The calculation in (9.3.1) is valid for any $f(x)$ which is a finite sum of $e^{2\pi i k x}$ terms. It can also be generalized to functions with arbitrary Fourier expansions.

We next want to apply this method to periodic functions over arbitrary dimension. To do this, we need to use some notation from Fourier analysis.

DEFINITION Suppose that $f: [0, 1]^d \rightarrow R$ is bounded. f is *periodic* if

$$f(x_1, \dots, x_{j-1}, 0, x_{j+1}, \dots, x_d) = f(x_1, \dots, x_{j-1}, 1, x_{j+1}, \dots, x_d)$$

for $j = 1, \dots, d$, and for all $x \in I^d$. For $h \in Z^d$ and $x \in I^d$, let $h \cdot x$ denote $\sum_{l=1}^d h_l x_l$, and define.

$$\hat{f}(h) \equiv \int_{[0,1]^d} f(x) e^{-2\pi i h \cdot x} \, dx \quad (9.3.2)$$

to be the *order h Fourier coefficient* of the periodic function f . Define

$$r(h) = \prod_{j=1}^d \max(1, |h_j|) \quad (9.3.3)$$

for $h \in Z^d$. For a scalar $C > 0$, define $\mathcal{E}^k(C)$ to be the set of all periodic functions f such that its Fourier series representation

$$f(x) = \sum_{h \in Z^d} \hat{f}(h) e^{2\pi i h \cdot x} \quad (9.3.4)$$

is absolutely convergent and that for all $h \neq 0$ the coefficients $\hat{f}(h)$ in (9.3.2) satisfy $|\hat{f}(h)| < Cr(h)^{-k}$. \mathcal{E}^k is the set of functions which belong to $\mathcal{E}^k(C)$ for some $C > 0$.

The growth condition for the Fourier coefficients of \mathcal{E}^k functions is certainly satisfied for any k by any finite Fourier series. We would like to have an easy way to determine if a function is a member of \mathcal{E}^k . Korobov proved the following important theorem (reported in Davis and Rabinowitz):

THEOREM 9.3.1 (Korobov) Suppose, for some integer k , that $f: [0, 1]^d \rightarrow R$ satisfies the following two conditions:

1. All partial derivatives

$$\frac{\partial^{m_1 + \dots + m_d} f}{\partial x_1^{m_1} \dots \partial x_d^{m_d}}, \quad 0 \leq m_j \leq k - 1, \quad 1 \leq j \leq d,$$

exist and are of bounded variation in the sense of Hardy and Krause.

2. All partial derivatives

$$\frac{\partial^{m_1 + \dots + m_d} f}{\partial x_1^{m_1} \dots \partial x_d^{m_d}}, \quad 0 \leq m_j \leq k - 2, \quad 1 \leq j \leq d,$$

are periodic on $[0, 1]^d$.

Then $f \in \mathcal{E}^k$.

Korobov's theorem is useful. It is impractical to compute Fourier series coefficients of multidimensional functions. Korobov tells us that we don't need to do this. All we need to check are smoothness, periodicity, and bounded variation; that is, if a function and enough of its derivatives are bounded and periodic, then its Fourier coefficients disappear at rate $r(h)^{-k}$. These derivative conditions are generally easy to check, and therefore Korobov's theorem makes it practical to use the \mathcal{E}^k property.

The importance of periodicity is illustrated in Richtmeyer's (1952) theorem.

THEOREM 9.3.2 (Richtmeyer) Suppose that $f: [0, 1]^d \rightarrow R$ is periodic and $\theta_1, \dots, \theta_d$, are d linearly independent irrational numbers in $[0, 1]$. If f is C^1 with ∇f being bounded and of bounded variation in the sense of Hardy and Krause, then

$$\left| \int_{[0,1]^d} f dx - \frac{1}{N} \sum_{n=1}^N f(\{n\theta_1\}, \dots, \{n\theta_d\}) \right| \leq \frac{A}{N}$$

for some $A \in R$.

This is a significant improvement, since we have eliminated the $(\log N)^d$ terms which appear in even the best versions of Hlawka's theorem. This theorem shows that we can get rapid convergence with moderate-size samples as long as f is periodic and smooth.

Periodization

At first theorems 9.3.1 and 9.3.2 may seem of little use because they assume that f is periodic. This can be remedied at relatively small cost. To use these rules for nonperiodic functions, we replace $f(x)$ with a periodic function, $\tilde{f}(x)$ such that $\int_{[0,1]^d} f = \int_{[0,1]^d} \tilde{f}$. One simple example of such a transformation is

$$\tilde{f}(x_1, \dots, x_d) = f(1 - 2|x_1 - \frac{1}{2}|, \dots, 1 - 2|x_d - \frac{1}{2}|).$$

Since this transformation will not preserve differentiability, \tilde{f} cannot belong to any C^k .

If we want Richtmeyer's theorem to apply, we must preserve differentiability. Suppose that we want to compute $\int_0^1 f(x)dx$ where $f(x)$ is C^∞ but not periodic. To accomplish this, we use a change of variables. Let $x = g(u)$ be a C^∞ nonlinear change of variables with $g(0) = 0$ and $g(1) = 1$. Then

$$\int_0^1 f(x)dx = \int_0^1 f(g(u))g'(u)du \equiv \int_0^1 h(u)du,$$

where $h(u) \equiv f(g(u))g'(u)$. If $g'(0) = g'(1) = 0$, then $h(0) = h(1) = 0$ and $h(u)$ is continuous and periodic. Furthermore, if $g''(0) = g''(1) = 0$, then we can show that $h'(u)$ is also continuous and periodic. In these cases g has transformed $\int_0^1 f(x)dx$ into an integral with a periodic integrand.

The following family of transformations accomplish periodization:

$$\tilde{f}_m(x_1, \dots, x_d) = f(\psi_m(x_1), \dots, \psi_m(x_d)) \psi'_m(x_1) \cdots \psi'_m(x_d),$$

where

$$\psi_m(u) \equiv (2m-1) \binom{2m-2}{m-1} \int_0^u t^{m-1} (1-t)^{m-1} dt.$$

In general, if f is C^m on a compact domain, then \tilde{f}_m will be in \mathcal{E}^{m+2} . Each transformation can be easily computed. In particular, $\psi_2(u) = 3u^2 - 2u^3$ and $\psi_3(u) = 10u^3 - 15u^4 + 6u^5$.

This is again the application of the change of variables idea. In this case we want to use formulas designed for periodic functions; therefore we find a change of variables

such that the new integrand is periodic. This periodization technique is also useful for the next class of rules we examine.

9.4 Method of Good Lattice Points

The procedures outlined in sections 9.2 and 9.3 use infinite sequences that are equidistributed. The advantage of such methods is that one has flexibility concerning the size of the sample to be taken. However, they have finite sample problems. Instead, the *method of good lattice points* first fixes the sample size, and then chooses a good collection of points of that size. This method is due to Korobov (1959, 1960) and makes powerful use of the \mathcal{E}^k property to construct quadrature formulas that converge at rate $N^{-k+\varepsilon}$ for $f \in \mathcal{E}^k$.

The method of good lattice points begins with an integer N and a vector $g \in \{0, 1, \dots, N-1\}^d$, forms the finite collection of points

$$x_l = \left\{ \frac{l}{N} g \right\}, \quad l = 1, \dots, N, \quad (9.4.1)$$

and computes the quasi-Monte Carlo approximation

$$\int_{I^d} f(x) dx \doteq \frac{1}{N} \sum_{l=1}^N f(x_l). \quad (9.4.2)$$

This approach uses a sample of fixed size N ; extending (9.4.1) beyond N is pointless, since $x_{N+l} = x_l$.

The task is to find combinations of N and g such that the approximation in (9.4.2) is good. To do this, we apply Fourier series methods. Suppose that f is periodic on I^d , with an absolutely convergent Fourier series (9.3.4). Next note that $\hat{f}(0) = \int_{I^d} f(x) dx$. Using the Fourier series (9.3.4), the error in the approximation (9.4.2) can be written

$$\frac{1}{N} \sum_{l=1}^N \sum_{h \in \mathbb{Z}^d} \hat{f}(h) e^{(2\pi i l/N) h \cdot g} - \hat{f}(0). \quad (9.4.3)$$

In (9.4.3) we note that $e^{2\pi i x}$ is periodic on I^d and drop the fractional part notation from $\{(l/N)g\}$. By changing the order of summation, we can rewrite (9.4.3) as

$$\frac{1}{N} \sum_{h \in \mathbb{Z}^d} \hat{f}(h) \sum_{l=1}^N e^{(2\pi i l/N) h \cdot g} - \hat{f}(0). \quad (9.4.4)$$

The key insight revolves around the inner sum, $\sum_{l=1}^N e^{(2\pi i l/N)(h \cdot g)}$. If $h \cdot g$ is a multiple of N , then $e^{(2\pi i l/N)(h \cdot g)} = 1$ for all l , and $\sum_{l=1}^N e^{(2\pi i l/N)(h \cdot g)} = N$. If $h \cdot g$ is not a multiple of N , then $h \cdot g = m \bmod N$ for some $m \neq 0$. The inner sum in (9.4.4) reduces to $\sum_{l=1}^N e^{2\pi i l m/N}$, which equals 0 when $m \neq 0$. Hence the error (9.4.3) reduces to

$$\sum_{\substack{0 \neq h \in \mathbb{Z}^d \\ h \cdot g = 0 \bmod N}} \hat{f}(h). \quad (9.4.5)$$

which is a sum of some of the Fourier coefficients of $f(x)$. This error is bounded above by $\sum_{\substack{h \neq 0 \\ h \cdot g = 0 \bmod N}} |\hat{f}(h)|$ which is finite whenever the Fourier series is absolutely convergent. Since the low-order terms of (9.3.4) make the major contribution to (9.4.5), good choices of N and g will make $h \cdot g = 0 \bmod N$ for low-order choices of h . Furthermore, if the Fourier coefficients decline rapidly, then this error bound will also decline rapidly as we choose g and N that leave fewer of the low-order terms in (9.4.5). Therefore good choices for g and N will result in quadrature formulas that converge rapidly as N increases.

These ideas lead to informal notions of what are “good lattice points.” Since $r(h)$ is a measure of the degree (“complexity”) of e^{ihx} , we use it to create a sense of order to the Fourier coefficients. Hence we want $r(h)$ to be large if $h \cdot g \equiv 0 \bmod N$. In this case the error bound for the integration approximation is small.

Computation of Good Lattice Points

The computation of good lattice points is a difficult task; see Hua and Wang (1981) for some derivations. We include some examples of good lattice points, g , in table 9.4. In table 9.4, N is the sample size corresponding to a good lattice point. The good lattice points in table 9.4 have the form $g^d = (1, g_2^d, \dots, g_d^d) \in R^d$ for $d = 3, 4, 5, 6$.

Unfortunately, the tables in existing books and papers are limited in their coverage. A serious user of lattice point methods would want to have a way to compute good lattice points for any sample size and dimension. A strategy pursued by Korobov and others is to examine lattice points that are simply generated and evaluate their performance in integrating certain test functions with known integrals. Two test functions that are particularly valuable are

$$F_1(x) = \prod_{j=1}^d (1 - 2 \ln(2 \sin \pi x_j)),$$

$$F_2(x) = \prod_{j=1}^d \left(1 - \frac{\pi^2}{6} + \frac{\pi^2}{2} (1 - 2\{x_j\})^2 \right),$$

Table 9.4
Good lattice points

N	g_2^3	g_3^3	g_2^4	g_3^4	g_4^4
101	40	85			
1,069	136	323	71	765	865
10,007	544	5,733	1,206	3,421	2,842
100,063	53,584	37,334	92,313	24,700	95,582
N	g_2^5	g_3^5	g_4^5	g_5^5	
1,069	63	762	970	177	
10,007	198	9,183	6,967	8,507	
33,139	32,133	17,866	21,281	32,247	
10,063	90,036	77,477	27,253	6,222	
N	g_2^6	g_3^6	g_4^6	g_5^6	g_6^6
3,001	233	271	122	1,417	51
10,007	2,240	4,093	1,908	931	3,984
33,139	18,236	1,831	19,143	5,522	22,910
100,063	43,307	15,440	39,114	43,534	39,955

which are both defined on I^d and integrate to 1. Their Fourier coefficients are $\hat{F}_1(h) = r(h)^{-1}$ and $\hat{F}_2(h) = r(h)^{-2}$. Therefore F_k is the function in \mathcal{E}^k , $k = 1, 2$, which Fourier coefficients converge at the slowest possible rate.

The first procedure is due to Korobov (1960). We choose a prime, p , to be the sample size N in (9.4.1), and choose the integer $a \in \{1, \dots, p-1\}$ that minimizes

$$H_1(a) \equiv \frac{3^d}{p} \sum_{k=1}^p \prod_{j=1}^d \left(1 - 2 \left\{ k \frac{a^{j-1}}{p} \right\} \right)^2.$$

$H_1(a)$ is the lattice integration formula (9.4.2) applied to $F_2(x)$ where we construct the lattice point g from the powers of $a \bmod p$ by $g_j = a^{j-1} \bmod p$, $j = 1, \dots, d$. Since the Fourier coefficients of F_2 are all positive, the error is always positive. Therefore the Korobov method is to find a lattice point consisting of powers of $a \bmod p$ that minimizes $H_1(a)$, one of the worst-behaved functions in \mathcal{E}^2 . We call such a point a *Korobov good lattice point*.

We can do this for each prime p to get a p -point lattice quadrature formula. There is no assurance that the formulas are monotonically better as we increase p , the number of points. Therefore, in constructing a sequence of lattice formulas, one should keep only those formulas that do better in integrating F_2 than formulas with fewer points.

The Korobov procedure for finding lattice points can be rather inefficient, requiring up to $O(dp^2)$ operations. Keast (1973) offers a more general procedure with lower computational cost. He first chooses J distinct primes, p_j , $j = 1, \dots, J$, and lets their product p be the sample size N . He then chooses a sequence of integers a_j , $j = 1, \dots, J$. First, a_1 is chosen to minimize

$$\frac{3^d}{p_1} \sum_{k=1}^{p_1} \prod_{j=1}^d \left(1 - 2 \left\{ k \frac{a_j^{j-1}}{p} \right\} \right)^2$$

over $a \in \{1, \dots, p_1 - 1\}$. More generally, for $l = 1, \dots, J$, a_l minimizes

$$H_l(a) \equiv \frac{3^d}{p_1 \cdots p_l} \sum_{k=1}^{p_1 \cdots p_l} \prod_{j=1}^d \left(1 - 2 \left\{ k \left(\frac{a_1^{j-1}}{p_1} + \cdots + \frac{a_{l-1}^{j-1}}{p_{l-1}} + \frac{a_l^{j-1}}{p} \right) \right\} \right)^2$$

for $a \in \{1, \dots, p_l - 1\}$. The *Keast good lattice point* g is then defined to be

$$g_j = \sum_{l=1}^J \frac{p}{p_l} a_l^{j-1}, \quad j = 1, \dots, d.$$

This approach has the advantage of requiring at most $\mathcal{O}(d(p_1 \cdot p_1 + (p_1 p_2) \cdot p_2 + \cdots + p p_J))$ operations. Again $H_J(g)$ serves as a performance index to rank various lattice point rules. The Korobov and Keast procedures are capable of generating lattice integration rules with a wide variety of accuracy, size, and dimensions.

We next state precisely how good “good lattice points” are.

THEOREM 9.4.1 (Korobov, Keast) For each d there is a constant, α , such that the following is true for all N : The error in integrating $f \in \mathcal{E}^k(C)$ with a Korobov or Keast good lattice point with sample size N is bounded above by $\alpha C N^{-k} (\ln N)^{kd}$.

Note the very rapid asymptotic convergence rate as the sample size, N , increases; in fact we do better than $N^{-k+\varepsilon}$ convergence for every $\varepsilon > 0$. The presence of the logarithms and the dimensional factor may indicate that the asymptotics take longer to bite at higher dimensions and in the less smooth classes of functions, or it may just indicate that these error bounds are not good for small N . The practical value of these rules depends on whether these error bounds are tight, an issue best examined by numerical experimentation.

The method of good lattice points requires $f(x)$ to be periodic. The periodization methods discussed in section 9.3 can be used here to apply lattice methods to non-periodic $f(x)$.

9.5 Estimating Quasi-Monte Carlo Errors

One purported advantage of Monte Carlo quadrature rules is the joint computation of an estimate for an integral and an estimate for the variance of such estimates, resulting in a confidence interval for the estimate. We now show that one can similarly estimate the error in a quasi-Monte Carlo approximation.

The basic idea is to add a small degree of randomness to provide an error estimate. This is referred to as the *randomization of a quasi-Monte Carlo rule*, and originated with Cranley and Patterson (1976). The general idea is to take a quadrature rule, $Q(f)$, and construct a continuum of related rules, $Q(f; \beta)$, parameterized by β , such that each rule is of “equal quality.” While the equal quality requirement is ambiguous, the essential necessity is that the error of the quadrature rule $Q(f; \beta)$ is not systematically related to β and that these β rules are accurate “on average.” More precisely, we require that

$$I(f) \equiv \int_D f(x) dx = E\{Q(f; \beta)\} \quad (9.5.1)$$

for some distribution of β .

One simple case is where (9.5.1) holds when β is uniformly distributed on $[0, 1]$. Then, if we take a sample, β_1, \dots, β_m , the estimate

$$\hat{I} \equiv \frac{1}{m} \sum_{j=1}^m Q(f; \beta_j) \quad (9.5.2)$$

is an unbiased estimator of $I(f)$. Furthermore the standard deviation of \hat{I} , $\sigma_{\hat{I}}$, is approximated by

$$\hat{\sigma}_{\hat{I}}^2 \equiv \frac{\sum_{j=1}^m (Q(f; \beta_j) - \hat{I})^2}{m-1} \quad (9.5.3)$$

From the Chebyshev inequality $Pr\{|\hat{I} - I(f)| < k\sigma_{\hat{I}}\} \geq 1 - 1/k^2$, we can estimate confidence intervals. Here, as in Monte Carlo integration, the crucial step is to formulate the problem in such a fashion that we have an estimate of the standard deviation.

We can apply this to many quasi-Monte Carlo methods. Recall that number theoretic methods approximate $\int_D f(x) dx$ with rules of the form $Q^g(f) \equiv (1/N) \sum_{n=1}^N f(\{(n/N)g\})$ which has an error (9.4.5). Similarly the β shifted approximation $Q^g(f; \beta) \equiv 1/n \sum_{n=1}^N f(\{(n/N)g + \beta\})$ has error

$$\sum_{\substack{0 \neq h \in \mathbb{Z}^d \\ h \cdot g = 0 \bmod N}} \hat{f}(h) e^{2\pi i \beta}, \quad (9.5.4)$$

which has the same absolute convergence properties as (9.4.5), since the formulas are identical except that each $\hat{f}(h)$ in (9.5.4) is multiplied by a complex number of norm one. Obviously the condition (9.5.1) holds when β is uniform. We can also use this procedure for any quasi-Monte Carlo integration scheme that relies on an equidistributed sequence, since if x_j is equidistributed on $[0, 1]$, then so is $x_j + \beta$.

9.6 Acceleration Methods and qMC Schemes

In the previous chapter we saw how we could improve the performance of Monte Carlo methods through a variety of acceleration methods. We can also use the same methods to improve quasi-Monte Carlo methods. The point of the acceleration methods was to replace $f(x)$ with a function that had the same integral but reduced variance. Since qMC methods do better with functions with reduced variation, acceleration schemes can also help qMC schemes.

For example, consider antithetic variates applied to $\int_0^1 x^2$. The total variation of $f(x) = x^2$ on $[0, 1]$ is 1, whereas the total variation of $\frac{1}{2}(x^2 + (1-x)^2) = x^2 - x + \frac{1}{2}$ is $\frac{1}{2}$. Therefore applying a sampling scheme to $\frac{1}{2}(x^2 + (1-x)^2)$ instead of x^2 will reduce the Koksma error bound, (9.2.5), by half. Similarly control variates are applicable whether we use MC or qMC to integrate the residual $f - \varphi$. We can also incorporate “importance sampling” into qMC schemes because it is just a change of variables scheme. We proceed just as in importance sampling for Monte Carlo integration except now we use a quasi-Monte Carlo sequence instead of a Monte Carlo scheme.

This observation holds for each of the acceleration methods discussed in chapter 8. Despite their probabilistic motivations, each acceleration method is basically a way to reduce total variation, which will reduce the maximum error of any deterministic sampling scheme, pseudo- or quasi-Monte Carlo.

9.7 Further Reading and Summary

This chapter has described sampling methods for integration and optimization that are based on number theoretic and Fourier analytic ideas. These methods realize an asymptotic convergence rate of N^{-1} , far faster than Monte Carlo methods. However, these asymptotic properties may not apply at reasonable sample sizes, and Monte

Carlo methods may dominate existing quasi-Monte Carlo methods for small sample sizes.

We have focused on the simpler quasi-Monte Carlo methods. The Sobol sequence, as implemented in Bratley and Fox (1988), is better than the equidistributed sequences displayed in tables 9.1. The (t, m, s) sequences described in Niederreiter (1992) dominate the Fourier analytic methods we discussed. This is an active area of ongoing research that is continuing to produce even better quasi-Monte Carlo methods.

The past three chapters have focused on integration methods. There is no one method that is always best. Gaussian methods dominate for low-dimensional integrals, say two to four, with smooth integrands. Monomial rules are likely to be competitive for integrals of moderate dimension, five to eight, with polynomial-like integrands. For smooth integrands and moderate dimension, quasi-Monte Carlo methods will also do well. If the dimension is large, the integrand not smooth, and the sample size is limited to a relatively small size (e.g., under a million), then Monte Carlo methods are likely the only ones that can give even a rough estimate.

Quasi-Monte Carlo methods have not been extensively used in economic analysis or econometrics. They have, however, received substantial attention in the financial literature, even the financial press, due to Paskov (1994) and Paskov and Traub (1995). Two web pages on quasi-Monte Carlo methods are <http://www.mat.sbg.ac.at/schmidw/links.html/> and <http://www.math.hkbu.edu.hk/qmc/qmc.html>.

Exercises

1. Write programs that compute the Weyl, Haber, and Niederreiter sequences. Write them so as to keep down round-off errors. Compute serial correlation properties for the first 10^5 points.
2. Redo the example solving (8.5.4) in the preceding chapter, but use equidistributed sequences. Are the errors less?
3. Write programs implementing Korobov's and Keast's method for generating good lattice points. How well do they do on the integrals in tables 9.2, and 9.3?
4. Write a program to compute the star discrepancy of a set of points from I^n . Generate several random sets of 1,000 points in I^n , for $n = 3, 4, 5$, and compute their star discrepancy.
5. Devise a way to compute a set of points from I^n with low discrepancy. Use it to find a set of 100 points with $n = 2, 3, 5, 10, 20$. Do your sets have lower discrepancy than 100 random points? than 500 random points? Repeat this for 1,000 points.
6. Solve exercise 8.5 using Monte Carlo and quasi-Monte Carlo methods. Compare performance.
7. Redo exercise 8.9 using quasi-Monte Carlo samples.
8. Redo exercise 8.4 with equidistributional sequences. Compare the performance of Monte Carlo and quasi-Monte Carlo methods for this problem.
9. Repeat Exercise 8.10 using quasi-Monte Carlo methods.

III NUMERICAL METHODS FOR FUNCTIONAL PROBLEMS

10 Finite-Difference Methods

A wide variety of economic problems lead to differential, difference, and integral equations. Ordinary differential equations appear in models of economic dynamics. Integral equations appear in dynamic programming problems and asset pricing models. Discrete-time dynamic problems lead to difference equations. These examples are all examples of dynamic problems. However, other problems also reduce to differential equations, such as equilibria of signaling models. Furthermore differential and difference equations are just examples of what is more generally called functional equations. Many functional equations that arise naturally in economics lie outside the usual families. As economic analysis of dynamic problems becomes more advanced, even more complex functional equations will appear. In this and the next chapter we examine basic ways to solve functional equations.

In this chapter we examine basic methods for solving ordinary differential equations and linear Fredholm integral equations. The key common feature is that they discretize the independent variable. We begin with finite-difference methods for ordinary differential equations, and discuss applications to continuous-time optimal control models.

We first describe basic methods for solving initial value problems and apply them to a signaling problem. We then discuss shooting methods, which combine initial value problem methods with nonlinear equation methods to solve boundary value problems. Optimal control problems are boundary value problems, which we illustrate with a life-cycle example. Simple shooting is often numerically unstable in economic applications. We introduce *reverse shooting* and show how it can do very well in optimal control problems by replacing “shooting” *for* a steady state with “shooting” *from* a steady state. We then examine integral equation solution methods that discretize the state space.

10.1 Classification of Ordinary Differential Equations

A *first-order ordinary differential equation* (ODE) has the form

$$\frac{dy}{dx} = f(y, x), \tag{10.1.1}$$

where $f: R^{n+1} \rightarrow R^n$ and the unknown is the function $y(x): [a, b] \subset R \rightarrow R^n$. When $n = 1$, we have a single differential equation, whereas if $n > 1$, we call (10.1.1) a *system of differential equations*.

In addition to (10.1.1) we need side conditions to tie down the unknown function $y(x)$. If we impose the condition $y(x_0) = y_0$ for some x_0 , we have an *initial value problem* (IVP). A simple condition is $y(a) = y_0$; that is, we fix the value of y at the

initial point, $x = a$. The IVP classification is also appropriate if we impose the terminal condition $y(b) = y_0$; the key fact is that $y(x)$ is pinned down at one $x_0 \in [a, b]$.

If we have a single differential equation, we can fix y at only one x ; hence a first-order differential equation in one variable is an IVP by default. When $n > 1$, the auxiliary conditions could fix different components of y at various x 's. A *two-point boundary value problem* (2PBVP) imposes n conditions on y of the form

$$\begin{aligned} g_i(y(a)) &= 0, & i &= 1, \dots, n', \\ g_i(y(b)) &= 0, & i &= n' + 1, \dots, n, \end{aligned} \tag{10.1.2}$$

where $g: R^n \rightarrow R^n$. More generally, a BVP imposes

$$g_i(y(x_i)) = 0 \tag{10.1.3}$$

for a set of points, x_i , $a \leq x_i \leq b$, $1 \leq i \leq n$. We often take $b = \infty$ in which case we impose some condition on $\lim_{x \rightarrow \infty} y(x)$. In both (10.1.2) and (10.1.3) we implicitly assume that the possibly nonlinear functions g do constitute n independent conditions. Despite the use of the words “initial” and “boundary,” the critical difference between IVPs and BVPs is whether the auxiliary conditions concern the solution at one point, as in IVPs, or whether the auxiliary conditions concern the solution at several points, as in BVPs.

A simple transformation makes all these definitions apply even when higher derivatives of y are used. For example, when presented with the second-order differential equation

$$\frac{d^2y}{dx^2} = g\left(\frac{dy}{dx}, y, x\right)$$

for $x, y \in R$, we define $z = dy/dx$ and study the system

$$\frac{dy}{dx} = z, \quad \frac{dz}{dx} = g(z, y, x),$$

of two first-order differential equations. Similarly we can transform an n th-order differential equation to n first-order equations.

The same descriptions of IVP and BVP apply to discrete-time systems, and we can focus on first-order systems. For example, we can convert the second-order system $x_{t+1} = g(x_t, x_{t-1}, t)$ into the first-order system

$$z_{t+1} = \begin{pmatrix} x_{t+1} \\ x_t \end{pmatrix} = \begin{pmatrix} g(x_t, x_{t-1}, t) \\ x_t \end{pmatrix} = G(z_t).$$

This stacking trick allows first-order notation to denote arbitrary dynamic systems.

10.2 Solution of Linear Dynamic Systems

We review basic results from linear system theory. These are special cases but cases that arise frequently and are important.

One-Dimensional Linear Problems

The general linear differential equation in R is

$$\dot{x} + a(t)x = b(t), \quad x(0) = x_0, \quad (10.2.1)$$

and has the solution

$$x(t) = x_0 e^{-\int_0^t a(s)ds} + \int_0^t e^{-\int_s^t a(z)dz} b(s) ds. \quad (10.2.2)$$

We can also solve linear finite-difference equations. The equation

$$x_{t+1} = a_t x_t + b_t \quad (10.2.3)$$

with x_0 given has the solution

$$x_{t+1} = \left(\prod_{s=0}^t a_s \right) x_0 + \sum_{s=0}^t \left(\prod_{j=s}^{t-1} a_j \right) b_s. \quad (10.2.4)$$

Linear Systems with Constant Coefficients

The special case of linear systems of ordinary differential equations with constant coefficients is also important. We will need to know the basic methods in later chapters. Suppose that $x \in R^n$ and

$$\dot{x} = Ax, \quad x(0) = x_0. \quad (10.2.5)$$

The solution to (10.2.5) is

$$x(t) = e^{At} x_0. \quad (10.2.6)$$

A useful representation of the solution (10.2.6) is to take the Jordan canonical form of A , $A = N^{-1}DN$, and write the solution (10.2.6) as

$$x(t) = e^{At} x_0 = e^{N^{-1}DNt} x_0 = N^{-1} e^{Dt} Nx_0,$$

which, if the eigenvalues of A are distinct, expresses the solution $x(t)$ as a weighted sum of the fundamental solutions, $e^{\lambda_i t}$ for $\lambda_i \in \sigma(A)$. This form shows that the

solution (10.2.6) is bounded only if all eigenvalues of A have negative real part. If the eigenvalues are not distinct, the Jordan canonical form is the same and the solution in (10.2.6) is unchanged; the only alteration is that e^{Dt} is not simply the diagonal matrix of $e^{\lambda_i t}$ terms.¹

Discrete-time systems can be similarly solved. Suppose that $x \in R^n$, and we want to solve²

$$x_{t+1} = Ax_t. \quad (10.2.7)$$

The solution to (10.2.7) is

$$x_t = A^t x_0. \quad (10.2.8)$$

Again we take the Jordan canonical form of A and rewrite the solution as

$$x_t = A^t x_0 = (N^{-1}DN)^t x_0 = N^{-1}D^t Nx_0,$$

which, in the case of distinct eigenvalues, expresses the solution x_t as a weighted sum of the fundamental solutions, λ_i^t where $\lambda_i \in \sigma(A)$. Here the solution is stable only if all $\lambda \in \sigma(A)$ has modulus less than one.

Equations (10.2.5) and (10.2.7) are examples of initial value problems. Linear boundary value problems also arise frequently in economics. Specifically we often need to solve problems of the form

$$\dot{x} = Ax,$$

$$x \equiv \begin{pmatrix} y \\ z \end{pmatrix}, \quad (10.2.9)$$

$$y(0) = y_0, \quad \lim_{t \rightarrow \infty} |z(t)| < \infty,$$

where y are called the *predetermined variables* and the z are free variables. To solve such problems, we replace A with its Jordan form, resulting in the system

$$\frac{d}{dt} \begin{pmatrix} y \\ z \end{pmatrix} = N^{-1}DN \begin{pmatrix} y \\ z \end{pmatrix},$$

which we rewrite as

1. For a discussion of linear differential equations, see Hirsch and Smale (1974).
2. We will use the conventional notation in dynamic analysis and let subscripts denote different vectors.

$$\frac{d}{dt} \left(N \begin{pmatrix} y \\ z \end{pmatrix} \right) = DN \begin{pmatrix} y \\ z \end{pmatrix}. \quad (10.2.10)$$

Without loss of generality, we can assume that $D \equiv \begin{pmatrix} D_1 & 0 \\ 0 & D_2 \end{pmatrix}$, where D_1 has the eigenvalues of D with negative real parts and D_2 has the eigenvalues of D with positive real parts.³

At this point we need to discuss determininacy issues. We want (10.2.9) to have a unique solution. That holds if and only if D_1 is square with the same size as the length of y , which is the number of predetermined variables, and the size of the square matrix D_2 equals the number of free variables. If we define

$$N \equiv \begin{pmatrix} N_{11} & N_{12} \\ N_{21} & N_{22} \end{pmatrix}, \quad \bar{x} = \begin{pmatrix} \bar{y} \\ \bar{z} \end{pmatrix} \equiv N \begin{pmatrix} y \\ z \end{pmatrix},$$

the system (10.2.10) reduces to $d\bar{x}/dt = D\bar{x}$, which implies the solution

$$\bar{x} = \begin{pmatrix} \bar{y} \\ \bar{z} \end{pmatrix} = \begin{pmatrix} e^{D_1 t} \\ e^{D_2 t} \end{pmatrix} = e^{Dt}. \quad (10.2.11)$$

We must choose $z(0)$ so that $z(t)$ is asymptotically bounded. Since the eigenvalues of D_2 are unstable, \bar{x} can be stable only if $0 = \bar{z}(0) = N_{21}y(0) + N_{22}z(0)$, which implies the solution

$$z(0) = -N_{22}^{-1}N_{21}y(0). \quad (10.2.12)$$

Therefore, to solve (10.2.9), we compute the matrix of eigenvectors, N , decompose it, and compute $z(0)$. Since $\bar{z}(t) = 0$ for all t , we actually have $z(t) = -N_{22}^{-1}N_{21}y(t)$ and $y(t) = (N_{11} - N_{12}N_{22}^{-1}N_{21})^{-1}e^{D_1 t}$.

Discrete-time boundary value problems are also handled in this fashion. Suppose that we have the problem

$$x_{t+1} = Ax_t,$$

$$x_t \equiv \begin{pmatrix} y_t \\ z_t \end{pmatrix}, \quad (10.2.13)$$

$$y_0 \text{ given, } \lim_{t \rightarrow \infty} |z_t| < \infty.$$

3. We are ignoring the more complicated case of eigenvalues with zero real part.

We again replace A with its Jordan form and examine the system

$$N \begin{pmatrix} y_{t+1} \\ z_{t+1} \end{pmatrix} = DN \begin{pmatrix} y_t \\ z_t \end{pmatrix}.$$

Without loss of generality, we can assume that $D \equiv \begin{pmatrix} D_1 & 0 \\ 0 & D_2 \end{pmatrix}$, where now D_1 has the eigenvalues of D which have modulus less than one and D_2 has the other eigenvalues. Notice that the decomposition of D differs in the discrete-time case because the notion of a stable eigenvalue has changed.

We again have to deal with determinacy issues. If we have a unique solution to (10.2.13), then D_1 has the same size as the length of y , the number of predetermined variables, and the size of the square matrix D_2 equals the number of free variables. If we define $N \equiv \begin{pmatrix} N_{11} & N_{12} \\ N_{21} & N_{22} \end{pmatrix}$, and $\bar{x} = \begin{pmatrix} \bar{y} \\ \bar{z} \end{pmatrix} \equiv N \begin{pmatrix} y \\ z \end{pmatrix}$, the system (10.2.13) reduces to $\bar{x}_{t+1} = D\bar{x}_t$, which has the solution

$$\bar{x} = \begin{pmatrix} \bar{y} \\ \bar{z} \end{pmatrix} = \begin{pmatrix} D_1^t \bar{y}_0 \\ D_2^t \bar{z}_0 \end{pmatrix}. \quad (10.2.14)$$

Since z_t is asymptotically bounded and the eigenvalues of D_2 are unstable, \bar{x} can be stable only if $0 = \bar{z}_0 = N_{21}y_0 + N_{22}z_0$, which in turn implies the solution

$$z_0 = -N_{22}^{-1}N_{21}y_0. \quad (10.2.15)$$

Therefore, to solve (10.2.13), we compute the matrix of eigenvectors, N , decompose it, and compute z_0 . Since $\bar{z}_t = 0$ for all t , we have $z_t = -N_{22}^{-1}N_{21}y_t$ and $y_t = (N_{11} - N_{12}N_{22}^{-1}N_{21})^{-1}D_1^t$ for all t .

10.3 Finite-Difference Methods for Initial Value Problems

Finite-difference methods are frequently used to solve the IVP

$$y' = f(x, y), \quad y(x_0) = y_0. \quad (10.3.1)$$

A finite-difference method first specifies a grid for x , $x_0 < x_1 < \dots < x_i < \dots$. In this chapter we assume that the grid has the form $x_i = x_0 + ih$, $i = 0, 1, \dots, N$ where h is the step size. The intent is to find for each i a value Y_i which approximates $y(x_i)$. Therefore we construct a difference equation on the grid, such as $Y_{i+1} = F(Y_i, Y_{i-1}, \dots, x_{i+1}, x_i, \dots)$, similar to the differential equation, and solve the difference equation for Y_1, \dots, Y_i, \dots , in sequence, where Y_0 is fixed by the initial con-

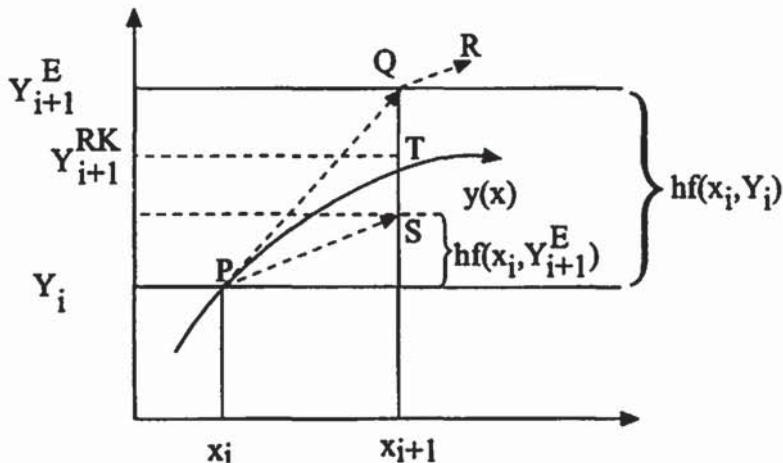


Figure 10.1
Euler and Runge-Kutta rules

dition, $Y_0 = y(x_0) = y_0$. Finite-difference methods approximate the solution at the grid points. To approximate the solution at other points, one could use any of a variety of interpolation methods.

Euler's Method

The simplest finite-difference scheme for (10.3.1) is the *Euler method*. Euler's method is the difference equation

$$Y_{i+1} = Y_i + hf(x_i, Y_i), \quad (10.3.2)$$

where Y_0 is fixed by the initial condition, $Y_0 = y(x_0) = y_0$. Figure 10.1 displays the geometric content of (10.3.2). Suppose that the current iterate is $P = (x_i, Y_i)$ and $y(x)$ is the true solution. At P , $y'(x_i)$ is the tangent vector \vec{PQ} . Euler's method follows that direction until $x = x_{i+1}$ at Q . The Euler estimate of $y(x_{i+1})$ is then Y_{i+1}^E .

The Euler scheme can be motivated by a Taylor series argument. Suppose that $y(x)$ is the true solution. Then expanding $y(x)$ around x_i shows that

$$y(x_{i+1}) = y(x_i) + hy'(x_i) + \frac{h^2}{2} y''(\xi)$$

for some $\xi \in [x_i, x_{i+1}]$. If we drop the h^2 term, assume $y'(x_i) = f(x_i, Y_i)$ and $Y_i = y(x_i)$, we get the Euler formula. If h is small, $y(x)$ should approximately solve this truncated Taylor expansion, allowing us to consider the Y_i generated by (10.3.2) to be a good approximation of $y(x_i)$. This derivation of the Euler scheme implicitly approximates $y(x)$ on the interval $[x_i, x_{i+1}]$ with a linear function with slope $f(x_i, Y_i)$.

We could also motivate the Euler scheme with a simple integration argument. The fundamental theorem of calculus implies that

$$y(x_{i+1}) = y(x_i) + \int_{x_i}^{x_{i+1}} f(t, y(t)) dt. \quad (10.3.3)$$

If we approximate the integral in (10.3.3) with $hf(x_i, y(x_i))$, that is, we approximate the integral with the box having width h and height $f(x_i, y(x_i))$, then (10.3.3) reduces to $y(x_{i+1}) \doteq y(x_i) + hf(x_i, y(x_i))$ which implies (10.3.2) if $Y_i \doteq y(x_i)$. This derivation of the Euler scheme also approximates $y(x)$ with a linear function $[x_i, x_{i+1}]$ of slope $f(x_i, Y_i)$.

As the step size h decreases, one hopes that the approximate solutions produced by the Euler method become better. To understand how the error of the Euler procedure depends on h , consider the differential equation $y'(x) = y(x)$, $y(0) = 1$. The solution is $y(x) = e^x$. The Euler method reduces to the finite-difference equation $Y_{i+1} = Y_i + h Y_i = (1 + h) Y_i$ which has the solution $Y_i = (1 + h)^i$ and implies the approximation $Y(x) = (1 + h)^{x/h}$. The relative error is

$$\ln \left| \frac{Y(x)}{y(x)} \right| = \frac{x}{h} \ln(1 + h) - x = \frac{x}{h} (h - h^2 + \dots) - x = -xh + \dots,$$

where the excluded terms are of higher order than h . This exercise shows that the relative error in the Euler approximation to $y(x)$ is proportional to h . Clearly, as h goes to zero, $Y(x)$ converges to the true solution $y(x)$.

The last example was a particular example, but it does indicate the general result. The next theorem states that, in general, the error of the Euler method is proportional to h , the step size, displaying *linear convergence*.

THEOREM 10.3.1 Suppose that the solution to $y'(x) = f(x, y(x))$, $y(0) = y_0$, is C^3 on $[a, b]$, that f is C^2 , and that f_y and f_{yy} are bounded for all y and $a \leq x \leq b$. Then the error of the Euler scheme with step size h is $\mathcal{O}(h)$; that is, it can be expressed as

$$y(x_i) - Y_i = D(x_i)h + \mathcal{O}(h^2)$$

where $D(x)$ is bounded on $[a, b]$ and solves the differential equation

$$D'(x) = f_y(x, y(x)) D(x) + \frac{1}{2} y''(x), \quad D(x_0) = 0$$

Proof See Atkinson (1989, pp. 352–53). ■

Theorem 10.3.1 shows that in the limit as the step size h goes to zero, the Euler scheme will produce $y(x)$, since the function $D(x)$ is finite and independent of h .

We generally need to make a choice between the rate of convergence and the amount of calculation per step. A higher rate of convergence will allow a larger choice of h and fewer total iterations. For example, if the error target is 0.0001, and the error equals h , then h must be 0.0001 or less, but if the error is h^2 , then $h = 0.01$ is adequate, reducing the number of iterations by a factor of 100. Quadratically convergent methods generally use more calculations per step, but it is clear that more rapidly schemes will be preferred unless the extra calculation burden is large.

The Euler scheme is an example of an *explicit method*. Explicit schemes are those that calculate Y_{i+1} explicitly in terms of x_i and Y_i . Explicit schemes are easy to program but often have stability problems unless h is small.

Implicit Euler Method

An alternative to the Euler scheme is the *implicit Euler method*. One way to derive the Euler scheme was to compute Y_{i+1} by using the Taylor expansion of y around x_i . Suppose instead that we expand around x_{i+1} . This yields the approximation

$$y(x_i) \doteq y(x_{i+1}) - hy'(x_{i+1}) = y(x_{i+1}) - hf(x_{i+1}, y(x_{i+1})),$$

which motivates the implicit Euler method

$$Y_{i+1} = Y_i + hf(x_{i+1}, Y_{i+1}). \quad (10.3.4)$$

This is a more difficult problem, for now Y_{i+1} is defined only implicitly in terms of x_i and Y_i . Each step of the implicit Euler scheme will need to solve a nonlinear equation in the unknown Y_{i+1} . This may appear to make the implicit Euler scheme inferior. On the other hand, the approximation scheme underlying the implicit Euler scheme is better in that it is more global in nature. The value of Y_{i+1} depends not only on Y_i and x_{i+1} but also on the behavior of f at (x_{i+1}, Y_{i+1}) , whereas the explicit Euler's method depended only on f at (x_i, Y_i) . Experience shows that the approximation considerations are often important and that we may do *much* better with the implicit Euler scheme. The extra computing time used in the nonlinear equation solving is often compensated by the ability to use larger h .

The primary difficulty with (10.3.4) is that it is a nontrivial numerical problem. To solve for Y_{i+1} in the nonlinear equation (10.3.4), we begin with $Y_{i+1}^0 = Y_i$, and construct a sequence, Y_{i+1}^j , of approximations to Y_{i+1} iteratively by using either fixed-point iteration,

$$Y_{i+1}^{j+1} = Y_i^j + hf(x_{i+1}, Y_{i+1}^j). \quad (10.3.5)$$

or Newton's method,

$$Y_{i+1}^{j+1} = Y_{i+1}^j - \frac{Y_{i+1}^j - Y_i^j - hf(x_{i+1}, Y_{i+1}^j)}{1 - hf_y(x_{i+1}, Y_{i+1}^j)}, \quad (10.3.6)$$

and stop when successive guesses are close. Fixed-point iteration is convenient to program but requires $|hf_y| < 1$ for convergence. Newton's method may also have convergence problems but should do better for *stiff* problems, that is, where f_y is large.

Trapezoid Rule

Another improvement on the Euler scheme is to use the integral approximation approach with a better quadrature formula. The fundamental theorem of calculus says that

$$\begin{aligned} y(x_{i+1}) &= y(x_i) + \int_{x_i}^{x_{i+1}} f(t, y(t)) dt \\ &\doteq y(x_i) + \frac{h}{2} [f(x_i, y(x_i)) + f(x_{i+1}, y(x_{i+1}))] - \frac{h^3}{12} y'''(\xi) \end{aligned} \quad (10.3.7)$$

for some $\xi \in [x_i, x_{i+1}]$. This motivates the difference equation

$$Y_{i+1} = Y_i + \frac{h}{2} [f(x_i, Y_i) + f(x_{i+1}, Y_{i+1})], \quad (10.3.8)$$

which is called the *trapezoid method*, has quadratic convergence, and is an implicit single-step method.

Runge-Kutta Methods

The Runge-Kutta method is one of many that first use a simple formula to look where the solution is going, but then check it and implement a correction when needed. In figure 10.1 we illustrate the typical iteration. Y_i and x_i are given by previous calculations, and we next want to compute Y_{i+1} . We first calculate the Euler step, $hf(x_i, Y_i)$, and the Euler approximation, $Y_{i+1}^E = Y_i + hf(x_i, Y_i)$. We then check the slope of the vector field at the point (x_i, Y_{i+1}^E) , which is $f(x_i, Y_{i+1}^E)$. The Euler approximation implicitly assumes that the slope at (x_i, Y_{i+1}^E) is the same as the slope at (x_i, Y_i) , a bad assumption in general. For example, if $y(x)$ is concave, this estimate of Y_{i+1} will overshoot $y(x_{i+1})$. If instead we use the slope at (x_i, Y_{i+1}^E) to estimate Y_{i+1} , we would end up at $Y_i + h f(x_i, Y_{i+1}^E)$, point S in figure 10.1. That estimate implicitly assumes that the slope of the vector field is always $f(x_i, Y_{i+1}^E)$ between x_i and x_{i+1} , an equally bad assumption, but in the opposite direction. For concave $y(x)$

this estimate of Y_{i+1} will undershoot $y(x_{i+1})$. The *first-order Runge-Kutta formula* (RK1) takes the average of these two estimates to arrive at a superior estimate displayed as T in figure 10.1. This geometrical argument yields the formula

$$Y_{i+1} = Y_i + \frac{h}{2} [f(x_i, Y_i) + f(x_{i+1}, Y_i + hf(x_i, Y_i))]. \quad (10.3.9)$$

The Runge-Kutta method in (10.3.9) converges quadratically to the true solution as $h \rightarrow 0$, using only two evaluations of f per step. Compared to the Euler method, Runge-Kutta achieves a higher order of convergence with only twice as much calculation per step.

A refinement of this approach will yield the *fourth-order Runge-Kutta method* (RK4), a more accurate Runge-Kutta scheme. RK4 executes the following iteration:

$$\begin{aligned} z_1 &= f(x_i, Y_i), \\ z_2 &= f\left(x_i + \frac{1}{2}h, Y_i + \frac{1}{2}hz_1\right), \\ z_3 &= f\left(x_i + \frac{1}{2}h, Y_i + \frac{1}{2}hz_2\right), \\ z_4 &= f(x_i + h, Y_i + hz_3), \\ Y_{i+1} &= Y_i + \frac{h}{6}[z_1 + 2z_2 + 2z_3 + z_4]. \end{aligned} \quad (10.3.10)$$

This scheme converges at the rate h^4 , and evaluates f only four times per step. The extra computation per step is amply rewarded by the higher rate of convergence.

Systems of Differential Equations

Many problems reduce to a system of differential equations of the form

$$\begin{aligned} y'_1(x) &= f_1(x, y_1, y_2, \dots, y_n), \\ &\vdots \\ y'_n(x) &= f_n(x, y_1, y_2, \dots, y_n). \end{aligned} \quad (10.3.11)$$

To solve a system of differential equations, one can just apply a single-equation method to each equation. For example, the explicit Euler method for this system is

$$Y_l^{i+1} = Y_l^i + hf_l(x_i, Y_1^i, \dots, Y_n^i), \quad l = 1, \dots, n. \quad (10.3.12)$$

Again it yields error proportional to h asymptotically. The implicit Euler method produces the system

$$Y_l^{i+1} = Y_l^i + h f_l(x_{i+1}, Y_1^{i+1}, \dots, Y_n^{i+1}), \quad l = 1, \dots, n, \quad (10.3.13)$$

which would make each iteration an n -dimensional system of nonlinear equations.

Runge-Kutta methods can also be adapted to solve systems. RK1 applied to (10.3.11) results in the vector difference system

$$Y^{i+1} = Y^i + \frac{h}{2} [f(x_i, Y^i) + f(x_{i+1}, Y^i + hf(x_i, Y^i))]. \quad (10.3.14)$$

RK4 applied to (10.3.11) results in the difference system

$$\begin{aligned} z^1 &= f(x_i, Y^i), \\ z^2 &= f\left(x_i + \frac{1}{2}h, Y^i + \frac{1}{2}hz^1\right), \\ z^3 &= f\left(x_i + \frac{1}{2}h, Y^i + \frac{1}{2}hz^2\right), \\ z^4 &= f(x_i + h, Y^i + hz^3), \\ Y^{i+1} &= Y^i + \frac{h}{6}[z^1 + 2z^2 + 2z^3 + z^4]. \end{aligned} \quad (10.3.15)$$

Systems of differential equations lead to stability problems, which don't arise in one-dimensional problems. As with single equations, implicit methods have superior stability properties, and one can use larger h . While the formulas are similar for systems, stability problems are more likely, since it only takes one explosive dimension to destabilize the entire solution. Stability problems are particularly severe in *stiff systems*. A system $dy/dx = f(y, x)$ is stiff if, along the solution path, the Jacobian of f , f_y , has negative eigenvalues that are much larger in magnitude than the other eigenvalues; such conditions can arise in optimal control problems. We do not discuss these problems here for two reasons. First, they do not appear to have been a major problem in economic problems so far, nor in the examples we discuss. Second, the methods discussed in the next chapter will also solve differential equations and do avoid these difficulties.

10.4 Economic Examples of IVPs

We will now examine a few examples of IVPs that arise in economic models. The first is Spence's signaling model. In the second example we return to homotopy continuation methods for solving nonlinear equations.

Signaling Equilibrium

One of the simplest initial value problems in economics is the education signaling model of Spence (1974). In this section we will describe the Spence model and numerically solve it with various methods.

The Spence model assumes that workers vary in their ability, each worker knowing that his ability is n for some $n \in [n_m, n_M]$, but cannot be directly observed by anyone else. Workers acquire y years of education, paying a total cost of $C(y, n)$; they then get a job. The output of a worker, $S(y, n)$, is a function of a worker's skill level, n , and his education, y , where $S_n, S_y, C_y > 0 > C_n$, and C is (weakly) concave in y . The critical assumption is that the employer sees only y , not output nor ability. Therefore wages are a function only of the education that a worker obtains. The problem is to compute the equilibrium wage schedule $w(y)$. Let $N(y)$ be the worker type that chooses education level y . Spence showed that in equilibrium $N(y)$ satisfies the first-order nonlinear differential equation

$$N'(y) = \frac{C_y(y, N(y)) - S_y(y, N(y))}{S_n(y, N(y))}. \quad (10.4.1)$$

Note that in this formulation of the problem, we take the education y as the independent variable and the type, n , as the dependent variable. While this may seem odd since y is endogenous and n is exogenous, it is nonetheless an equivalent formalization of the problem, since the only important feature is the presence of a monotonic functional relation between n and y , not which is endogenous and which is exogenous.

We assume that the lowest-ability individuals choose the socially efficient level of education; hence $N(y_m) = n_m$, where y_m , is fixed by the efficiency condition

$$S_y(y_m, n_m) = C_y(y_m, n_m). \quad (10.4.2)$$

The condition $N(y_m) = n_m$ gives us the initial condition we need to solve the Spence signaling equation (10.4.1).

In the case $S(y, n) = ny^\alpha$ and $C(y, n) = y/n$ the differential equation (10.4.1) reduces to

$$N'(y) = \frac{N(y)^{-1} - \alpha N(y)y^{\alpha-1}}{y^\alpha}, \quad (10.4.3)$$

and (10.4.2) implies that

$$N(y_m) = n_m, \quad (10.4.4)$$

Table 10.1
Signaling model results

h												
$y - y_m$	Euler				RK				RK4			
	0.1	0.01	0.001	0.0001	0.1	0.01	0.001	0.0001	0.1	0.01	0.001	
0.1	4(-1)	3(-2)	1(-3)	1(-4)	4(-1)	1(-3)	1(-4)	1(-6)	3(-2)	3(-3)	2(-6)	
0.2	1(0)	2(-2)	1(-3)	1(-4)	3(-1)	1(-3)	5(-4)	1(-6)	2(-2)	2(-3)	1(-6)	
0.4	7(-1)	1(-2)	7(-4)	7(-5)	2(-1)	4(-4)	3(-4)	4(-7)	1(-2)	1(-3)	6(-7)	
1.0	4(-1)	6(-3)	4(-4)	4(-5)	8(-2)	2(-4)	1(-4)	2(-7)	4(-3)	4(-4)	3(-7)	
2.0	2(-1)	4(-3)	3(-4)	3(-5)	4(-2)	1(-4)	7(-5)	1(-7)	2(-3)	2(-4)	1(-7)	
10.0	6(-2)	1(-3)	1(-4)	1(-5)	1(-2)	2(-5)	2(-6)	0(-7)	6(-4)	6(-5)	0(-7)	
Time	0.01	0.11	1.15	9.17	0.02	0.16	1.49	14.4	0.02	0.27	2.91	

h												
$y - y_m$	Implicit Euler-fixed-point iteration				Implicit Euler-Newton				Trapezoid			
	0.1	0.01	0.001	0.0001	0.1	0.01	0.001	0.0001	0.1	0.01	0.001	
0.1	2(0)	1(-1)	1(-3)	1(-4)	1(-1)	1(-2)	1(-3)	1(-4)	3(-3)	1(-4)	2(-6)	
0.2	2(0)	5(-2)	1(-3)	1(-4)	1(-1)	1(-2)	1(-3)	1(-4)	2(-3)	7(-5)	8(-7)	
0.4	2(0)	3(-2)	7(-2)	7(-5)	7(-2)	8(-3)	7(-4)	7(-5)	1(-3)	4(-5)	5(-7)	
1.0	3(0)	1(-2)	4(-3)	4(-5)	5(-2)	5(-3)	5(-4)	4(-5)	4(-4)	2(-5)	2(-7)	
2.0	3(0)	6(-3)	3(-3)	3(-5)	3(-2)	3(-3)	3(-4)	3(-5)	2(-4)	9(-6)	1(-7)	
10.0	6(0)	1(-3)	1(-4)	1(-5)	1(-2)	1(-3)	1(-4)	1(-5)	6(-5)	2(-6)	0(-7)	
Time	0.02	0.22	2.14	15.87	0.11	0.50	4.12	45.1	0.44	3.02	22.91	

where $y_m = (n_m^2 \alpha)^{1/(1-\alpha)}$ is the efficient level of education for type n_m individuals. The closed-form solution is

$$N(y) = y^{-\alpha} \left(\frac{2(y^{1+\alpha} + D)}{1 + \alpha} \right)^{1/2}, \quad (10.4.5)$$

where $D = ((1 + \alpha)/2)(n_m/y_m^{-\alpha})^2 - y_m^{1+\alpha}$.

Since we have a closed-form solution for this case, we can compare the true solution to the solutions produced by our discretization methods. In table 10.1 we display the results for $\alpha = 0.25$. We choose $n_m = 0.1$, which fixes $y_m = 0.00034$. The first column lists a variety of education levels; specifically it expresses $y - y_m$ for various values of y . We then display the magnitude of the error, that is, the difference between $N(y)$ and the computed approximation to the solution of (10.4.1), for several choices of method and h . We report the absolute error; since $n_m = 0.1 = N(y_m)$ is the smallest ability level, the relative errors are generally more but by one

order of magnitude at most. The first collection of four columns indicate the results when the Euler method was used and $h = 0.1, 0.01, 0.001$, and 0.0001 . The errors are initially nontrivial but decline roughly in proportion to h . The simple Runge-Kutta method has smaller errors and appears to be converging more rapidly. RK4 does even better.

Implicit methods are also considered. The implicit Euler method using Newton's method to solve the nonlinear equation appears to dominate the fixed-point iteration implementation for $h = 0.1, 0.01$, and 0.001 but is slower with the same error for $h = 0.0001$. At $h = 0.0001$, fixed-point iteration converges quickly, and its simplicity makes it faster than Newton's method. The trapezoid method is very good for $h \leq 0.01$. In this example, the best method was RK4 with $h = 0.0001$, since it achieved six-digit accuracy in under three seconds. The times in seconds are for a 33 MHz 486 computer; however, the important and robust feature is the relative time across methods.

Table 10.1 gives us an idea as to how small h needs to be to get good solutions. We find that good solutions can be computed for moderate values of h and that they need rather little computer time. We also see that the solutions do settle down quickly as h becomes smaller; apparently the convergence properties that the theory gives us begin to take effect at reasonable values of h . While numerical methods are unnecessary for the special case examined here, they would be needed for more general choices of S and C . While table 10.1 does not directly apply to these more general cases, it gives us assurances that these methods should do well for the more general cases.

Homotopy Solution Methods

In chapter 5 we saw that homotopy methods of solving nonlinear systems of equations naturally lead to systems of differential equations. Recall that the basic differential equation (5.9.4) is

$$\frac{dy_i}{ds} = (-1)^i \det\left(\frac{\partial H}{\partial y}(y)_{-i}\right), \quad i = 1, \dots, n+1, \quad (10.4.6)$$

where $(\cdot)_{-i}$ means we remove the i th column, $y = (x, t)$, and $H: R^n \times [0, 1] \rightarrow R^n$ is a homotopy with $H(0, 0) = 0$. With the initial conditions $t(0) = 0$ and $H(x(0), 0) = 0$, the solution to (10.4.6) is a parametric path $(x(s), t(s))$ which passes through a solution to $H(x, 1) = 0$, that is, for some $s > 0$, $t(s) = 1$ and $H(x(s), 1) = 0$.

In chapter 5 we used the Euler equation method in an example of the homotopy solution method. In that example we used a small step size and attained three-digit accurate answer to the solution of $H(x, 1) = 0$. More typically we would use a better differential equation solver. See Garcia and Zangwill (1981), and Allgower (1990) for extensive discussions of such methods.

10.5 Boundary Value Problems for ODEs: Shooting

Initial value problems are relatively easy to solve because the solution at each point depends only on local conditions. This allows us to use methods that are also local in nature, as are both the explicit and implicit procedures. In contrast, boundary value problems impose conditions on the solution at multiple points. We lose the local nature of the problem, and the numerical methods we use must become correspondingly global in nature. In this section we will discuss basic methods for solving BVPs.

Suppose that we have the two-point boundary value problem:

$$\begin{aligned}\dot{x} &= f(x, y, t), \\ \dot{y} &= g(x, y, t), \\ x(0) &= x^0, \quad y(T) = y^T,\end{aligned}\tag{10.5.1}$$

where $x \in R^n$, $y \in R^m$, and \dot{x} and \dot{y} are dx/dt and dy/dt .

A basic method for solving two-point boundary value problems is *shooting*. We know only $x(0)$, not $y(0)$. If we knew $y(0)$, then we could solve the equation by an initial value method. The idea of shooting is to guess the value of $y(0)$ and use an initial value method to see what that guess implies about $y(T)$. Usually the resulting value of $y(T)$ will not be consistent with the terminal condition, $y(T) = y^T$. So we make new guesses for $y(0)$ until we find a value for $y(0)$ that is consistent with the terminal condition.

When we consider the details, we see that the shooting algorithm has two basic pieces. First, for any guess $y(0) = y^0$, we solve the IVP differential equation in (10.5.2)

$$\begin{aligned}\dot{x} &= f(x, y, t), \\ \dot{y} &= g(x, y, t), \\ x(0) &= x^0, \quad y(0) = y^0,\end{aligned}\tag{10.5.2}$$

to get the corresponding value of $y(T)$; call that value $Y(T, y^0)$ to make explicit its dependence on y^0 . Therefore the first piece of any BVP method is a method for solving the IVP (10.5.2) that arises when we make a guess for $y(0)$. This can be done by using finite-difference methods.

The second piece of a BVP method involves finding the correct y^0 . Most guesses for y^0 will be wrong in that the terminal value $Y(T, y^0)$ won't equal y^T . We are interested in finding the y^0 such that $y^T = Y(T, y^0)$. This is a nonlinear equation in

the unknown y^0 . Therefore the second piece of a BVP method is a method for solving nonlinear equations.

Algorithm 10.1 Generic Shooting Algorithm

Objective: Solve two-point BVP (10.5.1).

Initialization. Guess $y^{0,i}$. Choose stopping criterion $\varepsilon > 0$.

Step 1. Solve (5.2) for $(x(T), y(T))$ given initial condition $y^0 = y^{0,i}$.

Step 2. If $\|y(T) - y^T\| < \varepsilon$, STOP. Else choose $y^{0,i+1}$ based on $y^{0,i}, y^{0,i-1}$, etc., and go to step 1.

This generic shooting method is an example of a two-layer algorithm. The inner layer, represented here in step 1, uses an IVP method that solves $Y(T, y^0)$ for any y^0 . This could be the Euler method, a Runge-Kutta method, or any of the other IVP methods. The accuracy of any of these methods depends on the choice of the step size h and will have errors substantially greater than machine zero. At the outer layer, step 2 here, we solve the nonlinear equation $Y(T, y^0) = y^T$. One can use any nonlinear equation method to choose the next iterate, $y^{0,i+1}$, based on previous iterates, $y^{0,i}, y^{0,i-1}, \dots$, and/or derivatives of $Y(T, y^0)$. Therefore we implement this in practice by defining a subroutine to compute $Y(T, y^0) - y^T$ as a function of y^0 and then send that subroutine to a zero-finding program. Our admonitions in chapter 5 about two-layer algorithms apply here; since the inner procedure is numerical, its error may require one to use a loose stopping criterion in the outer procedures.

10.6 Finite-Horizon Optimal Control Problems

Optimal control problems generally lead to boundary value problems and will be extensively studied in future chapters. We will review them now, focusing on the important computational aspects. The canonical problem is⁴

$$\begin{aligned} \max_u \quad & \int_0^T e^{-\rho t} \pi(x, u, t) dt + W(x(T)) \\ \text{s.t.} \quad & \dot{x} = f(x, u, t), \\ & x(0) = x_0, \end{aligned} \tag{10.6.1}$$

4. There are several variations of this problem. See Kamien and Schwartz (1981) for a catalogue of optimal control problems and the associated necessary conditions.

where $x \in R^n$, $u \in R^m$; that is, we have n state variables and m controls, ρ is the rate of discount, $\pi: R^n \times R^m \times R \rightarrow R$ is the flow rate of payoff, $W(x)$ is the value of the terminal state, and $f: R^n \times R^m \rightarrow R^n$ is the law of motion. To examine this problem, we form the current-value Hamiltonian

$$H(x, u, \lambda, t) = \pi(x, u, t) + \lambda^T f(x, u, t), \quad (10.6.2)$$

where $\lambda \in R^n$ is the vector of shadow prices for x . The costate equations are

$$\dot{\lambda} = \rho\lambda - (\pi_x + \lambda^T f_x). \quad (10.6.3)$$

and the maximum principle implies that

$$u(t) \in \arg \max_u H(x, u, \lambda, t). \quad (10.6.4)$$

The resulting collection of equations constitute a differential equation problem once we see how to use the maximum principle. The maximum principle produces an algebraic relation among the state, costate, and control variables which allows us to express the control variables as functions of the state and costate variables. If H is C^2 and concave in u , there is a unique selection in (10.6.4), $U(x, \lambda, t)$, defined by the first-order condition

$$0 = H_u(x, U(x, \lambda, t), \lambda, t). \quad (10.6.5)$$

Using the substitution $u(t) = U(x, \lambda, t)$, the solution to (10.6.1) satisfies (10.6.6):

$$\begin{aligned} \dot{x} &= f(x, U(x, \lambda, t), t), \\ \dot{\lambda} &= \rho\lambda - (\pi_x(x, U(x, \lambda, t), t) + \lambda^T f_x(x, U(x, \lambda, t), t)), \end{aligned} \quad (10.6.6)$$

plus boundary conditions, which for this problem are the initial condition on the state

$$x(0) = x_0 \quad (10.6.7)$$

plus the transversality condition

$$\lambda(T) = W'(x(T)). \quad (10.6.8)$$

Another common problem sets $W(x(T)) = 0$ in (10.6.1) and imposes the terminal condition $x(T) = x^T$. In these problems the state and costate equations are the same as for (10.6.1) but the terminal condition replaces (10.6.8) with

$$x(T) = x^T. \quad (10.6.9)$$

The typical optimal control problem leads to the boundary value problem system of equations consisting of (10.6.6)–(10.6.7) and either (10.6.8) or (10.6.9). Numerical solution of the resulting system requires methods to deal with differential equations obviously but also needs to deal with the function $U(x, \lambda, t)$. Sometimes U can be determined analytically but more frequently one will have to solve (10.6.5) numerically. The direct approach is to solve (10.6.4) for u at the values of x and λ that arise at each step of solving (10.6.6).

Life-Cycle Model of Consumption and Labor Supply

The life-cycle model is a simple example of an important BVP in economics. A simple case of this is the problem

$$\begin{aligned} & \max_c \int_0^T e^{-\rho t} u(c) dt \\ \text{s.t. } & \dot{A} = f(A) + w(t) - c(t), \\ & A(0) = A(T) = 0. \end{aligned} \tag{10.6.10}$$

$u(c)$ is the concave utility function over consumption c , $w(t)$ is the wage rate at time t , $A(t)$ is assets at time t , and $f(A)$ is the return on invested assets. We assume assets are initially zero and terminally zero.

The Hamiltonian of the agent's problem is $H = u(c) + \lambda(f(A) + w(t) - c)$. The costate equation is $\dot{\lambda} = \rho\lambda - \lambda f'(A)$. The maximum principle implies the first-order condition $0 = u'(c) - \lambda$, which implies consumption function $c = C(\lambda)$. The final system of differential equations describing life-cycle consumption is

$$\begin{aligned} \dot{A} &= f(A) + w - C(\lambda), \\ \dot{\lambda} &= \lambda(\rho - f'(A)), \end{aligned} \tag{10.6.11}$$

with the boundary conditions

$$A(0) = A(T) = 0. \tag{10.6.12}$$

It is often convenient to convert a state-costate system into one that consists of observable variables, such as assets and consumption. The relation $u'(c) = \lambda$ implies that (10.6.11) can be replaced by

$$\begin{aligned} \dot{c} &= -\frac{u'(c)}{u''(c)}(f'(A) - \rho), \\ \dot{A} &= f(A) + w - c, \end{aligned} \tag{10.6.13}$$

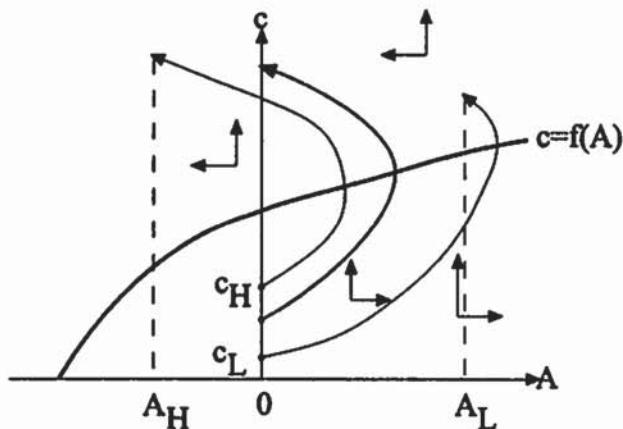


Figure 10.2
Shooting in a life-cycle problem

with boundary conditions (10.6.12), which is a system for the observables c and A .

Figure 10.2 gives a phase diagram representation of (10.6.10) assuming that $f'(A) > \rho$ for all A . The consumption path must obey the vector field defined by the system (10.6.10). The terminal condition is represented by the requirement that the consumption path begins and ends on the $A = 0$ line. We know neither the initial nor terminal λ values.

To solve this problem, the shooting method is a natural one to use. Figure 10.2 displays the implications of alternative guesses for $c(0)$. If $A(T) < 0$ when we guess $c(0) = c_H$, but $A(T) > 0$ when we guess $c(0) = c_L$, then we know that the correct $c(0)$ lies between c_L and c_H . We can find the true $c(0)$ by using the bisection method presented in algorithm 5.1. We formalize these steps in algorithm 10.2.

Algorithm 10.2 Life-Cycle Shooting

Objective: Solve (10.6.11) and (10.6.12) for $c(t)$ and $A(t)$ paths.

Initialization. Find some c_H such that $c(0) = c_H$ implies $A(T) < 0$ and c_L such that $c(0) = c_L$ implies $A(T) > 0$; choose a stopping criterion $\varepsilon > 0$. Set $c_0 = (c_L + c_H)/2$.

Step 1. Solve (using Runge-Kutta, Euler, or any other IVP scheme) the IVP consisting of (10.6.11) with the initial conditions $c(0) = c_0$, $A(0) = 0$ to compute $c(T)$, $A(T)$.

Step 2. If $|A(T)| < \varepsilon$, STOP. If $A(T) > \varepsilon$, then set $c_L = c_0$; else set $c_H = c_0$; set $c_0 = (c_L + c_H)/2$ and go to step 1.

10.7 Infinite-Horizon Optimal Control and Reverse Shooting

The canonical infinite-horizon autonomous optimal control problem is

$$\begin{aligned} \max_u & \int_0^\infty e^{-\rho t} \pi(x, u) dt \\ \text{s.t. } & \dot{x} = f(x, u), \\ & x(0) = x_0, \end{aligned} \tag{10.7.1}$$

which again will imply conditions (10.6.3)–(10.6.6). The difference lies in the boundary conditions. We still have $x(0) = x_0$, but we no longer have (10.6.8). In the examples we use in this book, (10.6.8) will be replaced by the transversality condition at infinity (TVC_∞)

$$\lim_{t \rightarrow 0} e^{-\rho t} |\lambda(t)^\top x(t)| < \infty.$$

Shooting methods have only limited value in solving infinite-horizon optimal control problems. It is particularly difficult to solve long-horizon models with shooting methods, since they involve integrating the differential equations over long time periods, and $x(T)$ is very sensitive to $\lambda(0)$ when T is large. To deal with these problems we must develop a better approach.

Let us go back to the basic problem. The difficulty with simple shooting is that the terminal state is excessively sensitive to the initial guess. However, this implies the reverse result that the initial state corresponding to any terminal state is relatively insensitive to the terminal state. Therefore, instead of making a guess as to the value of the unspecified initial conditions and integrating forward, we will make a guess as to the value of the unspecified terminal conditions and integrate *backward*; we will call that *reverse shooting*. We will illustrate it in some problems related to infinite-horizon control.

Optimal Growth

The simplest infinite-horizon economic problem is the continuous-time optimal growth model with one good and one capital stock. It is

$$\begin{aligned} \max_c & \int_0^\infty e^{-\rho t} u(c) dt \\ \text{s.t. } & \dot{k} = f(k) - c, \\ & k(0) = k_0, \end{aligned} \tag{10.7.2}$$

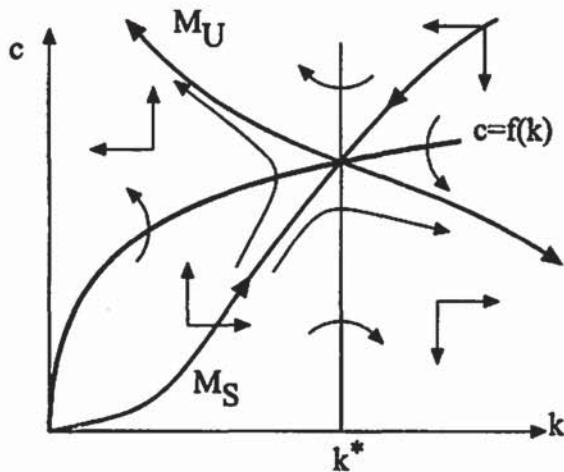


Figure 10.3
Shooting in a saddle-point problem

where k is the capital stock, c consumption, and $f(k)$ the aggregate net production function. The differential equations governing $c(t)$ and $k(t)$ reduce to

$$\dot{c} = \frac{u'(c)}{u''(c)} (\rho - f'(k)), \quad (10.7.3)$$

$$\dot{k} = f(k) - c,$$

and the boundary conditions are

$$k(0) = k_0, \quad 0 < \lim_{t \rightarrow \infty} |k(t)| < \infty.$$

Figure 10.3 is a phase diagram of this system when we assume that u and f are concave. The steady state is $k = k^*$, where $f'(k^*) = \rho$, and $c^* = f(k^*)$. There is a stable manifold, M_S , and an unstable manifold, M_U , making the steady state *saddle point stable*. Both M_U and M_S are *invariant manifolds* because, if the system begins on a point on M_S (M_U), the resulting path will remain on M_S (M_U). M_S is “stable,” since any path beginning on M_S will converge to the steady state, and M_U is “unstable” because any path beginning on M_U will diverge from the steady state.

We first try to use shooting to compute the stable manifold. We must adjust shooting to accomodate the infinite horizon. We want k and c to equal their steady state values at $t = \infty$, but that is impossible to accomplish. Instead, we try to find a $c(0)$ that produces a $(c(t), k(t))$ path that comes close to the steady state. This idea leads to algorithm 10.3 for $k_0 < k^*$. Observe that if $c(0)$ is too large, then the path crosses the $\dot{k} = 0$ curve and ultimately implies a falling capital stock, but that a defi-

cient $c(0)$ results in a path that crosses the $\dot{c} = 0$ line and implies a falling consumption level. These observations produce algorithm 10.3.

Algorithm 10.3 Infinite-Horizon Forward Shooting

Objective: Solve (10.7.3) for $c(t)$ and $k(t)$ paths for $t \in [0, T]$ and $k_0 < k^*$.

Initialization. Set c_H equal to $f(k_0)$, and set $c_L = 0$; choose a stopping criterion $\varepsilon > 0$.

Step 1. Set $c_0 = (c_L + c_H)/2$.

Step 2. Solve (using Runge-Kutta, Euler, or any other IVP scheme) the IVP consisting of (10.7.3) with initial conditions $c(0) = c_0$, $k(0) = k_0$; stop the IVP algorithm at the first t when $\dot{c} < 0$ or $\dot{k} < 0$, and denote it T .

Step 3. If $|c(T) - c^*| < \varepsilon$, STOP. If $\dot{c} < 0$, then set $c_L = c_0$; else set $c_H = c_0$; go to step 1.

From the phase diagram it is clear why numerical solutions will have difficulty computing the path corresponding to the stable manifold. We see that any small deviation from M_S is magnified and results in a path that increasingly departs from M_S . However, suppose that we wanted to compute a path on M_U ; note that the flow actually pushes points toward M_U , and small deviations are squashed. For example, if we solve the differential equation system beginning at a point near the steady state, the solution will move toward M_U . Therefore, if we were interested in computing a path that lies on M_U , we would just choose a point near the steady state for the initial condition and integrate the system.

Even though we don't want to compute the unstable manifold, we can use this observation to compute the stable manifold. The idea is to change the system so that the stable manifold becomes the unstable manifold. This is accomplished easily by reversing the direction of time, yielding the system

$$\begin{aligned}\dot{c} &= -\frac{u'(c)}{u''(c)} (\rho - f'(k)), \\ \dot{k} &= -(f(k) - c).\end{aligned}\tag{10.7.4}$$

The phase diagram of this system is the same as figure 10.3 but *with the arrows turned 180 degrees*. Therefore the unstable manifold of (10.7.4) is M_U in figure 10.4.

The stable manifold of (10.7.3) has a particularly important interpretation. Since calendar time plays no role in this problem, the choice of consumption at time t depends solely on the current capital stock. We express this by defining a policy function, $C(k)$, such that $c(t) = C(k(t))$. Since optimality requires the capital stock to converge to the steady state, the only optimal (c, k) pairs are those on the stable

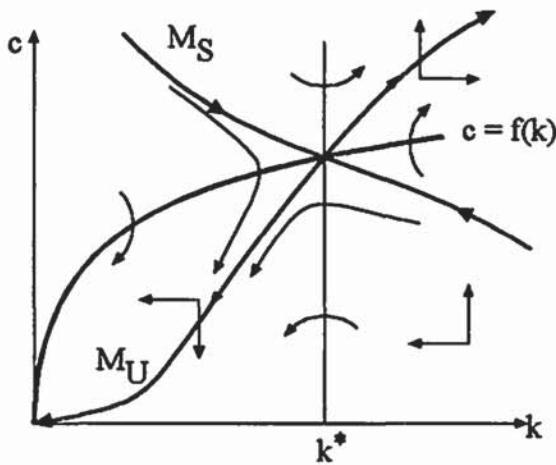


Figure 10.4
Reverse shooting in a saddle-point problem

manifold. Hence the stable manifold is the consumption policy function, $C(k)$. Furthermore (10.7.3) implies that $C(k)$ satisfies the differential equation

$$C'(k) = \frac{\dot{c}}{\dot{k}} = \frac{u'(C(k))}{u''(C(k))} \frac{\rho - f'(k)}{f(k) - C(k)}, \quad (10.7.5)$$

which can also be written

$$C'(k)(f(k) - C(k)) - \frac{u'(C(k))}{u''(C(k))}(\rho - f'(k)) = 0. \quad (10.7.6)$$

One application of reverse shooting is to compute the stable manifold of (10.7.3). We first compute $C'(k)$ at k^* . At the steady state, (10.7.5) reduces to 0/0. By l'Hôpital's rule

$$C'(k^*) = -\frac{f''(k^*)}{f'(k^*) - C'(k^*)} \frac{u'(C(k^*))}{u''(C(k^*))},$$

which is a quadratic equation in $C'(k^*)$. Since consumption is increasing in k , we choose the positive root:

$$C'(k^*) = \frac{f'(k^*)}{2} \left(1 + \sqrt{1 + 4 \frac{u'(c^*)}{u''(c^*)} \frac{f''(k^*)}{f'(k^*)f'(k^*)}} \right). \quad (10.7.7)$$

With reverse shooting, $C(k)$ can be calculated by solving two IVPs. For $k > k^*$ we begin with $k = k^* + h$, using the approximate initial condition

Table 10.2
Optimal growth with reverse shooting

k	c	Errors		
		$h = 0.1$	$h = 0.01$	$h = 0.001$
0.2	0.10272	0.00034	3.1(-8)	3.1(-12)
0.5	0.1478	0.000025	3.5(-9)	4.1(-13)
0.9	0.19069	-0.001	-3.5(-8)	1.8(-13)
1.	0.2	0	0	0
1.1	0.20893	-0.00086	-5.3(-8)	-1.2(-12)
1.5	0.24179	-0.000034	-1.8(-9)	-2.1(-14)
2.	0.2784	-9.8(-6)	-5.0(-10)	3.6(-15)
2.5	0.31178	-5.0(-6)	-2.6(-10)	-1.3(-14)
2.8	0.33068	-3.8(-6)	-1.9(-10)	-1.2(-14)

$$C(k^* + h) \doteq c^* + hC'(k^*),$$

and then proceed via some IVP method applied to (10.7.5) to compute $C(k)$ for $k = k^* + nh$, $n = 2, 3, 4, \dots$. For $k < k^*$ the initial condition is

$$C(k^* - h) \doteq c^* - hC'(k^*),$$

and we proceed via some IVP method applied to (10.7.5) to compute $C(k)$ for $k = k^* - nh$, $n = 2, 3, 4, \dots$. The total result is an approximation to $C(k)$ for k above and below k^* .

Table 10.2 illustrates results of reverse shooting applied to (10.7.6) for the case $\rho = 0.05$, $f(k) = 0.2k^{0.25}$, and $u(c) = -c^{-1}$. The first column is capital stock k , and the second column is the computed consumption for k when $h = 10^{-5}$, a step size so small that we take the answer as the true solution. Columns 3 through 5 display the errors when we use RK4 with $h = 0.1$, 0.01, and 0.001. Note that the consumption values appear to be converging at the rate h^4 to the $h = 10^{-5}$ solution, the convergence predicted by theory. It is not surprising that the solution is so accurate for k near 1; table 10.2 also shows that the solution is quite accurate even for the extreme values of $k = 0.2$ and $k = 2.8$. The uniformity of the error illustrates the stability of the reverse shooting approach.

We often want to express the policy function, $C(k)$ in a simple form. The points $\{(c_i, k_i) | i = -N, \dots, N\}$ generated by reverse shooting are all supposedly on or near the graph of $c = C(k)$. Therefore we could use the least squares approach

$$\min_{\beta} \sum_{i=-N}^N \left(c_i - \sum_{j=0}^m \beta_j \varphi_j(k_i) \right)^2$$

to approximate $C(k)$ with $\hat{C}(k) \equiv \sum_{j=0}^m \beta_j \varphi_j(k)$ for some basis functions $\varphi_j(k)$.

Multidimensional Reverse Shooting

To further illustrate reverse shooting, we next consider the multidimensional profit maximization problem

$$\max \int_0^\infty e^{-rt} \left(\pi(k) - \sum_{i=1}^n I_i - \sum_{i=1}^n \frac{\gamma_i I_i^2}{2} \right) dt,$$

$$\dot{k}_i = I_i, \quad i = 1, \dots, n,$$

$$k(0) = k^0,$$

where $k \in R^n$ represents n capital stocks, $\pi(k)$ is the profit flow, r the interest rate, and I_i the net investment rate of stock i . The adjustment costs for stock i equal $\gamma_i I_i^2/2$. The costate equation is $\dot{\lambda}_i = r\lambda_i - \pi_i$, $i = 1, \dots, n$. The maximum principle yields $0 = -(1 + \gamma_i I_i) + \lambda_i$, $i = 1, \dots, n$, which implies the investment rules $I_i = (\lambda_i - 1)/\gamma_i$, $i = 1, \dots, n$. The resulting system is

$$\dot{k}_i = \frac{\lambda_i - 1}{\gamma_i}, \quad i = 1, \dots, n, \tag{10.7.9}$$

$$\dot{\lambda}_i = r\lambda_i - \pi_i(k), \quad i = 1, \dots, n,$$

with the initial condition $k(0) = k^0$. The steady state is defined by $\dot{k} = \dot{\lambda} = 0$, which implies that $\lambda_i = 1$ and $\pi_i = r$ in the steady state for $i = 1, \dots, n$. We assume that π is strictly concave and that there exists a unique steady state at $k = k^*$. This is a simple system, but we use it to illustrate the basic ideas.

We first need to deal with the problem of having an infinite horizon. One way to deal with the infinite horizon is to replace it with a long, finite horizon and solve the problem

$$\max \int_0^T e^{-rt} \left(\pi(k) - \sum_{i=1}^n I_i - \sum_{i=1}^n \frac{\gamma_i I_i^2}{2} \right) dt,$$

$$\dot{k}_i = I_i, \quad i = 1, \dots, n,$$

$$k(0) = k^0, \quad k(T) = k^*,$$

for some large T . The terminal conditions on $k(T)$ make sure that we approximate a path to the steady state. The system (10.7.9) still describes the optimal path.

The standard shooting procedure would guess λ^0 and integrate the system (10.7.9) to compute $k(T)$, which is a function, say $K(T, \lambda^0)$, of λ^0 . It then solves the equation $K(T, \lambda^0) - k^* = 0$ to determine the correct λ^0 . Since small changes in λ^0 will produce large changes in $k(T)$, this approach is poor. In reverse shooting, we also take $k(T) = k^*$ as fixed and try to choose a $\lambda^T \in R^n$ consistent with $k(0) = k_0$. To do this, we guess a value for λ^T , then compute $k(0)$ by integrating the time-reversed system of (10.7.9), expressed as

$$\begin{aligned}\dot{k}_i &= -(\lambda_i - 1)\gamma_i^{-1}, & i &= 1, \dots, n, \\ \dot{\lambda}_i &= -r\lambda_i + \pi_i(k), & i &= 1, \dots, n,\end{aligned}\tag{10.7.10}$$

together with the initial conditions⁵ $k(T) = k^*$ and $\lambda(T) = \lambda^T$. In this case it is $k(0)$ which is a function of λ^T , say $K(0, \lambda^T)$, and our task is to compute a solution to the equation

$$K(0, \lambda^T) - k^0 = 0.\tag{10.7.11}$$

This is more likely to work, since small changes in λ^T will, hopefully, generate only small changes in the implied value of $k(0)$.

We must be flexible with the terminal time. If T is too small, we will find that the λ^T that solves (10.7.11) will differ substantially from λ^* . Since we want λ^T be close to λ^* , we increase T until the λ^T solution to (10.7.11) is close to λ^* .

We could use the result of reverse shooting to compute policy functions in the multidimensional case. To generate the necessary data, we would have to apply reverse shooting to several choices of $k(0)$. This will generate points on several threads on the stable manifold, points that can be used in a least squares approximation of the policy function.

In this problem the reverse shooting method is generally far more stable than the forward shooting method. There is no guarantee that it will always be better for arbitrary optimal control problems; this will depend on the character of steady state convergence. We offer it as an alternative to the more common forward shooting method.

5. We need an initial guess for $\lambda(T)$. Ideally we would like to use $\lambda(T) = \lambda^*$ as the initial guess, but then the system (10.7.9) would go nowhere. Instead we could use, for example, $\lambda(T) = 0.99\lambda^*$.

10.8 Integral Equations

Integral equations arise naturally in dynamic economic analyses. The general integral equation can be written

$$0 = \varphi \left(g(x), f(x), \int_a^b H(x, z, f(z)) dz \right), \quad (10.8.1)$$

where $f(x)$ is the unknown function, and $g(x)$ and $H(x, z, y)$ are known functions.

This general formulation has some special forms of particular interest. A *linear Fredholm equation of the first kind* has the form

$$f(x) = \int_a^b K(x, z) f(z) dz. \quad (10.8.2)$$

The function $K(x, z)$ is called the *kernel* of the integral equation. This is a linear equation because the RHS is linear in the unknown function f . A *linear Fredholm equation of the second kind* has the form

$$f(x) = g(x) + \int_a^b K(x, z) f(z) dz. \quad (10.8.3)$$

Volterra equations of the first and second kind are special cases of Fredholm equations. Suppose that $K(x, z) = 0$ for $x < z$; then $\int_a^b K(x, z) f(z) dz = \int_a^x K(x, z) f(z) dz$, and the linear Fredholm equations of first and second kind can be written

$$f(x) = \int_a^x K(x, z) f(z) dz \quad (10.8.4)$$

and

$$f(x) = g(x) + \int_a^x K(x, z) f(z) dz, \quad (10.8.5)$$

which are *linear Volterra equations of first and second kind*. Volterra integral equations and ordinary differential equations are closely related, since the differential equation $f'(x) = K(x, f(x))$ for $x \in [a, b]$ is equivalent to the nonlinear Volterra integral equation

$$f(x) = f(a) + \int_a^x K(z, f(z)) dz,$$

where $f(a)$ is the initial condition.

Linear Fredholm Equations

A general approach to solving linear Fredholm integral equations of the second kind arise naturally from quadrature methods. In equations like (10.8.2) and (10.8.3), the item needing approximation is the integral, $\int_a^b K(x, z)f(z) dz$. For each x this is an integral over z . Furthermore, if $K(x, z)$ is continuous in z for each x , we can replace the integral with a good quadrature rule. Any such rule has a weighting function, $w(x)$, (which is $w(x) = 1$ in the case of Newton-Cotes formulas), nodes $x_i \in [a, b]$, $i = 1, \dots, n$, and weights, $\omega_i > 0$, $i = 1, \dots, n$, and, at each x , yields the approximation

$$\int_a^b K(x, z)f(z) dz \doteq \sum_{j=1}^n \frac{\omega_j K(x, x_j)f(x_j)}{w(x_j)}. \quad (10.8.6)$$

This sum approximates $\int_a^b K(x, z)f(z) dz$ for any x , in particular, for the quadrature nodes x_i , $i = 1, \dots, n$. If we make the substitutions $x = x_i$ for $i = 1, \dots, n$, in the Fredholm equation and apply (10.8.6), we arrive at the set of n linear equations for the n values $f(x_i)$:

$$f(x_i) = g(x_i) + \sum_{j=1}^n \frac{\omega_j K(x_i, x_j)f(x_j)}{w(x_j)}, \quad i = 1, \dots, n, \quad (10.8.7)$$

which is linear in the $f(x_i)$. This use of a quadrature formula in the integral equation reduces the problem of estimating the function f to that of estimating f at the quadrature nodes.

Integral equation methods differ in their choice of the weights and nodes. One choice is to set $w(x) = 1$, $x_i = a + (i - 1)(b - a)/(n - 1)$, and choosing the ω_i according to a Newton-Cotes formula; this leads to the linear system

$$f(x_i) = g(x_i) + \sum_{j=1}^n \omega_j K(x_i, x_j)f(x_j), \quad i = 1, \dots, n. \quad (10.8.8)$$

An alternative is the Gauss-Legendre set of nodes and weights, which also assumes $w(x) = 1$ and leads to a similar formula.

Sometimes an efficient choice will suggest itself. For example, if $K(x, z) = H(x, z)e^{-z^2}$, $-a = b = \infty$, and $H(x, z)$ is polynomial-like in z , then a Gauss-Hermite approximation of the integral is suggested. This leads to the linear system

$$f(x_i) = g(x_i) + \sum_{j=1}^n \omega_j H(x_i, x_j)f(x_j), \quad i = 1, \dots, n, \quad (10.8.9)$$

where the x_i and ω_i are the Gauss-Hermite nodes and weights. One can apply any Gaussian quadrature formula. In applying Gaussian quadrature corresponding to the weighting function $w(z)$, one uses the corresponding Gaussian rule's weights, ω_i and nodes, x_i and solve the system (10.8.8).

The solution to (10.8.9) produces values for $f(x)$ at the $x_i, i = 1, \dots, n$. We really want a function that approximates the solution to (10.8.3) at all $x \in [a, b]$. To construct this approximation, we form $\hat{f}(x)$, where

$$\hat{f}(x) = g(x) + \sum_{j=1}^n \frac{\omega_j K(x, x_j) f(x_j)}{w(x_j)}. \quad (10.8.10)$$

This is called the *Nystrom extension*. It is an approximation to the unknown function, not just an approximation to a finite set of values.

Markov Chains: Approximations and Ergodic Distributions

One example of integral equation is the computation of an ergodic distribution of a discrete-time, continuous-state Markov process. Suppose that a random variable obeyed the law $\Pr\{x_{t+1} \leq x | x_t\} = F(x|x_t)$ where F has a density $f(x|x_t)$. We are often interested in the ergodic distribution

$$H(x) = \lim_{T \rightarrow \infty} P\{x_T \leq x | x_0\}.$$

If $H(x)$ has a density $h(x)$, then $h(x)$ satisfies the integral equation

$$h(x) = \int h(z) f(x|z) dz. \quad (10.8.11)$$

Since $H(\cdot)$ is a distribution, $h(\cdot)$ also must satisfy $\int h(z) dz = 1$. To solve (10.8.11), we pick a weighting function $w(x)$, and an appropriate quadrature rule with weights ω_i and nodes x_i , and form the linear system of equations

$$h(x_i) = \sum_{j=1}^n \frac{\omega_j h(x_j) f(x_i|x_j)}{w(x_j)}. \quad (10.8.12)$$

This system is linear and homogeneous in the unknown $h(x_i)$ values. To eliminate the indeterminacy arising from the homogeneity, we replace one of the equations with

$$\sum_{j=1}^n \frac{\omega_j h(x_j)}{w(x_j)} = 1, \quad (10.8.13)$$

which states that $h(x)$ is a probability density.

The linear problem (10.8.12) has a suggestive interpretation. Equation (10.8.12) examines only the value of h on the x_i grid. Tauchen and Hussey (1991) show how we can approximate the continuous-state Markov chain with a finite-state chain. They begin as we do with a grid of x_i points and a weight function. They then define

$$\pi(x_k|x) = \frac{f(x_k|x)}{s(x)w(x_k)} \omega_k$$

where

$$s(x) = \sum_{i=1}^n \frac{f(x_i|x)}{w(x_i)} \omega_i$$

For each x , $\pi(x_k|x)$ is a probability measure over the x_k , constructed by replacing the integral with a sum to compute the relative probabilities and then normalizing to attain true probabilities. These probabilities have the advantage of being approximations to the conditional expectation operator, since

$$E\{g(x, y)|x\} = \int g(x, y)f(y|x)dy \doteq \sum_{k=1}^n g(x, x_k)\pi(x_k|x).$$

Furthermore, when we restrict $\pi(x_k|x)$ to the x_i grid, we have a Markov chain on the x_i . This should serve as a good approximation to the continuous-space chain as long as it is constructed to be a good approximation to all the conditional expectations operators. As a result many authors use this finite-state approximation to model continuous-state processes.

10.9 Further Reading and Summary

Finite difference methods are powerful techniques to solve differential equations and have been extensively developed. Lick (1989) is a recent presentation of basic finite-difference methods. We did not discuss partial differential equations because most economists are unaware of them and special methods dominate in those areas where PDEs do arise. The next chapter presents simpler, more intuitive, and more robust numerical methods for PDEs and applies them in continuous-time dynamic stochastic contexts. A reader interested in finite-difference methods for PDEs is referred to Ames (1977), Botha and Pinder (1983), and Lapidus and Pinder (1982). PDEs are used in option pricing models; see Hull (1989) for the key references.

Since optimal control is so pervasive in the analysis of deterministic dynamic economic systems, the boundary value problems discussed here arise frequently.

Shooting methods have been used; in particular, see Lipton et al. (1982) for a discussion of multiple shooting. Other methods have also been utilized. Ascher et al. (1988), Kubicek and Hlavacek (1983), and Roberts and Shipman (1971) are the standard references for boundary value problems.

Exercises

1. Consider the Solow growth model $\dot{k} = sf(k) - \delta k$ with $s = 0.2$, $\delta = 0.1$, $f(k) = k^\alpha$, $\alpha = 0.25$, and $k(0) = 0.5$. Compute the numerical solutions using the Euler method, Runge-Kutta, and RK4 with $h = 1, 0.1, 0.01$. Compare these solutions with the exact solution.
2. Consider the life-cycle problem (10.6.10). Suppose that $u(c) = c^{r+1}/(y+1)$. Suppose that the lending rate is r but that the borrowing rate of interest is $r(1 + \theta A^2)$ expressing the costs and limitations of borrowing. Suppose that wages follow the pattern $w(t) = 1$ for $M \leq t \leq R$ and zero otherwise. Compute solutions for $c(t)$ and $A(t)$ for $\rho = 0.04$, $r \in \{0.05, 0.06, 0.08, 0.10\}$, $\theta \in \{0, 0.04, 0.2, 1.0\}$, $T = 55$, $M = 5, 10$, $R = 40, 45$, and $h = 0.01, 0.1, 1.0$.
3. Use the path-following homotopy method to solve the general equilibrium problem posed in equation (5.2.6). Which solution do you get to? Can you find other homotopies that find the other equilibrium? Apply homotopy path-following to solve instances of exercise 5.3. How large can the step sizes be before you have problems? Does RK4 do better than the Euler finite-difference method?
4. Using the approach exposited before theorem 10.3.1, compute the error of applying the implicit Euler, RK, RK4, and trapezoid methods to the differential equation $y'(x) = y(x)$, $y(0) = 1$.
5. Consider the optimal growth problem with adjustment costs

$$\max_I \int_0^\infty e^{-\rho t} u(f(k) - I - \gamma I^2) dt$$

$$\text{s.t. } \dot{k} = I,$$

for $\rho = 0.05$, $u(c) = \log c$, $f(k) = 0.2k^{0.25}$, $\gamma = 100$. Compute both the time path with $k(0) = 0.5$, and the optimal policy function.

6. Consider the learning curve problem

$$\max_Q \int_0^\infty e^{-\rho t} \pi(q, Q) dt$$

$$\text{s.t. } \dot{Q} = q - \frac{Q}{100},$$

where $\pi(q, Q) = q - q^2/2 - (Q+1)^{-0.2}q$. Compute both the optimal time path with $Q(0) = 0$ and the optimal policy function.

7. Write programs using shooting and reverse shooting to solve the multidimensional growth problem

$$\max_{I \in R^n} \int_0^\infty e^{-\rho t} u \left(\sum_{i=1}^n f_i(k_i) - \sum_{i=1}^n (I_i + \gamma_i I_i^2) \right) dt$$

$$\text{s.t. } \dot{k}_i = I_i, \quad i = 1, \dots, n,$$

for $\rho = 0.05$, $u(c) = \log c$, and $f_i(k_i) = \rho k_i^{\alpha_i}/\alpha_i$. Solve the problem for $n = 2, 3, 5, 10$ and for several combinations of $\alpha_i \in \{0.25, 0.4\}$, and $\gamma_i \in \{10, 50, 100, 250\}$, and $k_i(0) \in \{0.5, 0.8, 1.0, 1.2, 1.5\}$. Compute the optimal policy functions for several cases.

8. Solve the two-sector optimal growth problem

$$\max_{I_c, I_I, l_c, l_I} \int_0^\infty e^{-\rho t} u(f_c(k_c, l_c), l_c + l_I) dt$$

s.t. $\dot{k}_c = I_c$,

$\dot{k}_I = I_I$,

$$0 = I_c + \gamma_c I_c^2 + I_I + \frac{\gamma_I I_I^2}{100} - f_I(k_I, l_I),$$

where $f_i(k_i, l_i)$ is the output of sector i with capital input k_i and labor input l_i , $i = c$ denotes the consumption sector, and $i = I$ denotes the investment sector. Make Cobb-Douglas choices for the f functions and CRRA choices for u . Compute the steady state. Compute the optimal path with initial conditions equal to ± 5 and ± 10 percent of the steady state. Plot the optimal k_c and k_I paths in (k_c, k_I) space from these various initial conditions.

9. Repeat Exercise 8.10 using integral equation methods.

11 Projection Methods for Functional Equations

Finite-difference methods have been useful for solving ordinary differential equations in economics; they can be used to solve partial differential equations and a wide variety of other equations. However, many problems in economics lead to functional equations that are not of a standard type. For example, the functional equations that arise in discrete-time growth models involve composition of functions, a feature that cannot be handled directly by finite-difference methods. The complexity of the equations that arise in economics forces us to use more sophisticated methods from numerical functional analysis.

In this chapter we study *projection methods*. The underlying approach of projection methods is substantially more general than finite-difference methods, and this more general approach will allow us to solve problems for which there are no easy finite-difference methods. The advantage of the projection approach is that it is so intuitive; it gives a general way to solve functional equations of almost any description. In fact the typical economist's acquaintance with regression will make these methods feel natural. Moreover many of the ad hoc solution methods that have been developed by economists to solve economic problems turn out to be particular implementations of the projection method. By learning the general projection method, we will understand better how these methods work, how to improve on existing methods, and how to apply the projection method to more complex problems.

In this chapter we will first display the projection method applied to linear ordinary and partial differential equations. We then give a general description of projection methods in terms of the functional analytic basics. From this description, it is obvious that projection methods are potentially useful in solving almost any functional equation in economics. We demonstrate their flexibility in this chapter by showing how to use projection methods to solve boundary value problems, life-cycle and optimal growth problems, implicit function problems, and conditional expectations. Projection methods will also be used extensively in later chapters on solving dynamic economic models.

11.1 An Ordinary Differential Equation Example

The projection method is a general method using approximation methods and finite-dimensional methods to solve problems with unknown functions. In section 11.3 we will define it abstractly in functional analytic terms. Since that description will be rather abstract, we first present examples of projection methods applied to a simple ordinary differential equation and to a simple partial differential equation. These examples will make the later abstract discussion more understandable.

Consider the differential equation

$$y' - y = 0, \quad y(0) = 1 \quad (11.1.1)$$

which has the solution $y = e^x$. We will use projection methods to solve it over the interval $0 \leq x \leq 3$.

The first point is that projection methods view the problem differently than finite-difference methods. Finite-difference methods focus on approximating the value of y at a finite number of specified points; that is, at a prespecified collection of x_i , we compute approximations Y_i . Finite-difference methods proceed by solving a finite set of equations linking these x_i and Y_i values. In contrast, projection methods take a functional analytic view of a problem, focusing on finding a *function* that approximately solves the *functional* equation expressed in (11.1.1). While the differences appear to be slight at this point, the change in focus leads to a much more flexible approach.

To formulate (11.1.1) in a functional way, we define L

$$Ly \equiv y' - y. \quad (11.1.2)$$

L is an operator, a function that maps functions to functions. The domain of L includes all C^1 functions, and its range is C^0 . The differential equation $y' = y$ combined with the initial condition $y(0) = 1$ can be viewed as the problem of finding a C^1 function y that satisfies the initial condition $y(0) = 1$ and is mapped by L to the zero function. Another way of viewing it is to define the set $Y = \{y(x) \mid y \in C^1, y(0) = 1\}$; that is, Y is the set of C^1 functions that satisfies the initial condition $y(0) = 1$. Then (11.1.1) is the problem of finding a $y \in Y$ such that $Ly = 0$.

Since the focus in projection methods is on finding an approximate function, not just a sequence of values, we must find a way to represent functions in the computer. One way is to use polynomials. We know from the Weierstrass theorem that any C^1 function can be approximated well by large sums of polynomial terms. We cannot represent infinite series on the computer, but we can represent finite sums. We will let the finite list $a_i, i = 1, \dots, n$, represent the polynomial $\sum_{j=1}^n a_j x^j$. For each choice for the values of the a_i , we define the function

$$\hat{y}(x; a) = 1 + \sum_{j=1}^n a_j x^j. \quad (11.1.3)$$

The set of such functions created by letting the a_j coefficients range over R creates an affine subset of the vector space of polynomials. Note that $\hat{y}(0; a) = 1$ for any choice of a , implying that \hat{y} satisfies the boundary condition of (11.1.1) no matter what a is chosen. Hence $\hat{y}(\cdot; a) \in Y$ for any a .

We will try to choose a so that $\hat{y}(x; a)$ “nearly” solves the differential equation (1.1). To determine how well \hat{y} does this, we compute the *residual function*

$$R(x; a) \equiv L\hat{y} = -1 + \sum_{j=1}^n a_j(jx^{j-1} - x^j) \quad (11.1.4)$$

The residual function is the deviation of $L\hat{y}$ from zero, the target value. A projection method adjusts a until it finds a “good” a that makes $R(x; a)$ “nearly” the zero function. Different projection methods are generated by varying how one operationalizes the notions “good” and “nearly.” We will now review the basic implementations.

For this example we initially take $n = 3$. We need to calculate the three coefficients, a_1 , a_2 , and a_3 . A natural procedure for economists is the *least squares* method. In this method we find a that minimizes the total squared residual over the interval of interest; that is, it solves

$$\min_a \int_0^3 R(x; a)^2 dx. \quad (11.1.5)$$

The objective is quadratic in the a 's, and the first-order conditions reduce to the linear problem

$$\begin{pmatrix} 6 & \frac{9}{2} & -\frac{54}{5} \\ \frac{9}{2} & \frac{36}{5} & 0 \\ \frac{54}{5} & 0 & 41\frac{23}{35} \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} -3 \\ 0 \\ \frac{27}{2} \end{pmatrix}. \quad (11.1.6)$$

The solution is displayed on the first row of table 11.1.

Table 11.1
Solutions for coefficients in (11.1.3)

Scheme	a_1	a_2	a_3
Least squares	1.290	-0.806	0.659
Galerkin	2.286	-1.429	0.952
Subdomain	2.500	-1.500	1.000
Chebyshev collocation	1.692	-1.231	0.821
Uniform collocation	1.000	-1.000	0.667
Power series	1.000	0.500	0.167
Optimal L^2	1.754	-0.838	0.779

Another concept of $R(\cdot; a)$ being nearly zero is that it is zero on average. Since we have three unknowns, we need to find three averages. One way is to decompose $[0, 3]$ into three intervals

$$[0, 3] = [0, 1] \cup [1, 2] \cup [2, 3] \equiv D_1 \cup D_2 \cup D_3.$$

We then use the three conditions

$$0 = \int_{D_i} R(x; a) dx, \quad i = 1, 2, 3, \quad (11.1.7)$$

to fix a . This yields the linear conditions

$$\begin{pmatrix} \frac{1}{2} & \frac{2}{3} & \frac{3}{4} \\ -\frac{1}{2} & \frac{2}{3} & \frac{13}{4} \\ -\frac{3}{2} & -\frac{4}{3} & \frac{11}{4} \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}. \quad (11.1.8)$$

This procedure is called the *subdomain* method. The solution to (11.1.8) appears on the row in table 11.1 labeled “subdomain.”

If $R(x; a)$ were the zero function, then the integral of its product with any other function would also be zero. This observation leads us to the *method of moments* which fixes a by using powers of x to generate projection conditions. Since we have three unknowns, we use the first three powers:

$$0 = \int_0^3 R(x; a) x^j dx, \quad j = 0, 1, 2. \quad (11.1.9)$$

Continuing for $j = 1, 2$, we end up with a linear system in a :

$$\begin{pmatrix} -\frac{3}{2} & 0 & \frac{27}{4} \\ -\frac{9}{2} & -\frac{9}{4} & \frac{243}{20} \\ -\frac{45}{4} & \frac{81}{10} & \frac{243}{10} \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} 3 \\ \frac{9}{2} \\ 6 \end{pmatrix}. \quad (11.1.10)$$

Our approximation is built up from weighted sums of x , x^2 , and x^3 . If $R(x; a)$ were the zero function, then the integral of its product with each of these functions would

be zero. This observation leads us to the *Galerkin method* which fixes a by using the functions used in the approximation to impose the three moment conditions

$$0 = \int_0^3 R(x; a) x^j dx, \quad j = 1, 2, 3.$$

This is the same as the method of moments except that here we use x , x^2 , and x^3 instead of 1, x , and x^2 . However, the motivation is different; in the method of moments we use x^k because it is a moment, but in the Galerkin we use x^k because it is used in the construction of the approximation. In general, there will be substantial differences between these methods.

The preceding methods all involve integrals of the residual function. In this problem these integrals are easy to compute directly; that is not true generally. In contrast, *collocation* methods proceed by choosing a so that $R(x; a)$ is zero at a finite set of x values. Since we have three unknowns, we will choose three collocation points. If we choose the three points 0, $3/2$, 3, and set $R(x; a)$ equal to zero at these values for x , the linear conditions on a are

$$R(0; a) = 0 = -1 + a_1$$

$$R(1.5; a) = 0 = -1 - \frac{1}{2}a_1 + \frac{3}{4}a_2 + \frac{27}{8}a_3 \tag{11.1.11}$$

$$R(3; a) = 0 = -1 - 2a_1 - 3a_2$$

Although a uniform grid is natural, our discussion of interpolation in chapter 6 indicates that it is not a good choice. If we were told that $R(x_i; a) = 0$, $i = 1, 2, 3$, then the quadratic interpolant for $R(x; a)$ given these data would be the zero function. Collocation implicitly assumes (hopes) that if a is such that $R(x_i; a) = 0$, $i = 1, 2, 3$, then $R(x; a) = 0$ for all $0 \leq x \leq 3$. In order for this interpolation argument to be as reliable as possible, what x_i 's should be chosen? In chapter 6 we saw that a uniform grid may be a poor choice, in that there are functions with modest curvature that are zero on the uniform grid but vary substantially otherwise.

However, we saw that if a smooth, well-behaved function were zero at the zeros of some Chebyshev function on $[-1, 1]$, then the function is close to zero on all of $[-1, 1]$. When we are on intervals other than $[-1, 1]$, we must make an appropriate linear change of variable, discussed in section 6.4. In this context, this implies that the x_i should be the zeros of $T_3(x)$ adapted to $[0, 3]$, which is the set

$$\left\{ \frac{3}{2} \left(\cos \frac{\pi}{6} + 1 \right), \frac{3}{2}, \frac{3}{2} \left(\cos \frac{5\pi}{6} + 1 \right) \right\}.$$

Collocation at zeros of Chebyshev polynomials adapted to the interval of interest called *Chebyshev collocation*.

Another way to compute an approximate solution is the *power series* method. The power series method uses information about y and its derivatives at $t = 0$ to compute a . Since $y' - y = 0$ at $t = 0$, we know that $y'(0) = 1$. But $\hat{y}' = a_1 + 2a_2x + 3a_3x^2$ and $\hat{y}'(0) = a_1$. Hence $a_1 = 1$. We can also fix a_2 and a_3 in this way. Assuming a smooth solution, we can differentiate the differential equation to find $y'' - y' = 0$. Since this holds at $t = 0$, $y''(0) = y'(0) = 1$. However, $\hat{y}'' = 2a_2 + 6a_3x$ and $\hat{y}''(0) = 2a_2$. Hence $a_2 = 1/2$. Similarly $y''' - y'' = 0$, $\hat{y}'''(0) = 1 = 6a_3$, implying that $a_3 = 1/6$. The final approximation is

$$\hat{y} \approx 1 + x + \frac{x^2}{2} + \frac{x^3}{6},$$

which indeed are the first four terms of the Taylor series expansion of e^x .

The power series method will produce an approximation that is very good near 0. In fact, since the error is $\mathcal{O}(x^4)$, this cubic approximation is the best possible cubic approximation near 0. As we move away from 0 the quality of the approximation will decay rapidly because the error is $\mathcal{O}(x^4)$. We discuss the power series method here because it is also a well-known method for solving ordinary differential equations which produces an approximating function.

The rows in table 11.1 show that the coefficients generated by these procedures differ substantially from each other. They often differ from the coefficients of the least squares fit of a cubic polynomial satisfying the boundary condition to the function e^x . These coefficients are displayed in the row labeled “optimal L^2 .”

The L^2 errors for these methods for $n = 3, 4, \dots, 10$, are summarized in table 11.2. table 11.2 demonstrates several aspects of these methods. First, they all do better as

Table 11.2
Errors of projection methods applied to (11.1.2)

n	Uniform collocation	Chebyshev collocation	Least squares	Galerkin	Subdomain	Taylor series	Optimal L^2
3	5.3(0)	2.2(0)	3.2(0)	5.3(-1)	1.4(0)	3.8(0)	1.7(-1)
4	1.3(0)	2.9(-1)	1.5(-1)	3.6(-2)	1.4(-1)	1.8(0)	2.4(-2)
5	1.5(-1)	2.5(-2)	4.9(-3)	4.1(-3)	2.2(-2)	7.7(-1)	2.9(-3)
6	2.0(-2)	1.9(-3)	4.2(-4)	4.2(-4)	2.0(-3)	2.9(-1)	3.0(-4)
7	2.2(-3)	1.4(-4)	3.8(-5)	3.9(-5)	2.7(-4)	9.8(-2)	2.8(-5)
8	2.4(-4)	9.9(-6)	3.2(-6)	3.2(-6)	2.1(-5)	3.0(-2)	2.3(-6)
9	2.2(-5)	6.6(-7)	2.3(-7)	2.4(-7)	2.3(-6)	8.2(-3)	1.7(-7)
10	2.1(-6)	4.0(-8)	1.6(-8)	1.6(-8)	1.6(-7)	2.1(-3)	1.2(-8)

one uses more terms. Since both collocation methods are based on interpolation ideas, we cannot be assured of this. The Taylor series method is generally the worst. Its poor performance is not too surprising, for it is a local method. Uniform collocation is inferior to the integral methods, an expected result since collocation uses information at so few points. Least squares and Galerkin do best and yield similar results. Their solutions are almost as good as the polynomial that minimizes the sum of squared errors over $[0, 3]$. Chebyshev collocation does nearly as well as Galerkin and better than subdomain. This is strong evidence of how good Chebyshev collocation can be.

11.2 A Partial Differential Equation Example

We will now apply projection methods to a simple partial differential equation to illustrate projection methods in a different context. Readers who are unfamiliar with partial differential equations should not despair; the point of this example is to show how one can use projection methods even when one is not expert about the kind of equation being considered.

Consider the simple heat equation

$$\theta_t - \theta_{xx} = 0 \quad (11.2.1)$$

on the domain $0 \leq x \leq 1$ and $0 \leq t < \infty$. We assume the initial condition

$$\theta(x, 0) = \sin \pi x \quad (11.2.2)$$

and impose the boundary conditions

$$\theta(0, t) = 0, \quad \theta(1, t) = 0, \quad 0 \leq t < \infty. \quad (11.2.3)$$

The problem is to find a function on $[0, 1] \times [0, \infty)$ that satisfies the partial derivative conditions in (11.2.1) and also satisfy (11.2.2) for $t = 0$ and (11.2.3) at $x = 0, 1$. There is a unique solution, $\theta(x, t) = e^{-\pi^2 t} \sin \pi x$.

We will take a projection approach to solve this problem. We begin by forming the polynomial approximation

$$\hat{\theta}(x, t) = \theta_0(x) + \sum_{j=1}^n a_j(t) (x - x^j), \quad (11.2.4)$$

where

$$\theta_0(x) = \sin \pi x$$

is the initial condition. Note how we handled the boundary condition, (11.2.3). The approximation in (11.2.4) consists of weighted sums of $x - x^j$ terms. Since $x - x^j$ is zero at $x = 0, 1$, $\hat{\theta}$ will satisfy the boundary conditions in (11.2.3) for any $a_j(t)$ functions. Initial conditions for initial value ODEs are easy to specify, since they just involve the value of a function at a point. Modeling boundary conditions for PDE's generally requires more ingenuity.

When we substitute $\hat{\theta}$ into $\theta_t - \theta_{xx} = 0$, we get the residual function:

$$R(x, t) = -\theta_{0xx}(x, t) + \sum_{j=1}^n (\dot{a}_j(t)(x - x^j) - a_j(t)(-j)(j-1)x^{j-2}). \quad (11.2.5)$$

We have n unknown functions, $a_j(t)$. Let $\langle \cdot, \cdot \rangle$ be the inner product $\langle u, v \rangle = \int_0^1 uv dx$. As in the previous example, we solve out for the unknown $a_j(t)$ functions by imposing, at each t , n projection conditions of the form

$$\langle R(x, t), \psi_j(x) \rangle = 0, \quad j = 1, \dots, N, \quad (11.2.6)$$

for some collection of functions $\psi_j(x)$. This will result in a system of linear differential equations of the form

$$M\dot{A} + BA + C = 0, \quad (11.2.7)$$

where

$$A(t) = \begin{pmatrix} a_1(t) \\ \vdots \\ a_n(t) \end{pmatrix},$$

$$M_{jk} = \langle x - x^j, \psi_k \rangle,$$

$$B_{jk} = -\langle j(j-1)x^{j-2}, \psi_k \rangle,$$

$$C_k = -\langle \theta_{0xx}, \psi_k \rangle.$$

Since $\theta(x, 0) = \theta_0(x)$, $A(0) = 0$ is the initial condition. Rewrite (11.2.7) as $\dot{A} = -M^{-1}BA - M^{-1}C$. Since M , B , and C are constant with respect to time, the result is a linear ordinary differential equation in A with constant coefficients and with initial condition $A = 0$. This could be solved for arbitrary t once we calculate the eigenvalues of $M^{-1}B$.

We have not yet specified the ψ_j functions. The ψ_j could be the $x - x^j$ functions, but the rows of M and B would be somewhat collinear, creating the possibility of

$M^{-1}B$ being poorly conditioned. A better choice may be the Legendre polynomials adapted to the interval $[0, 1]$. The main requirement is that the ψ_j be chosen so that the ψ_j can be easily computed and the linear systems to be solved be well-conditioned.

Note that the projection conditions reduced the partial differential equation to a system of ordinary differential equations in the coefficient functions $a_j(t)$. These equations could be solved using finite-difference methods. We could go one step further and eliminate all derivatives. Suppose that we want to solve $\theta(x, t)$ over the compact domain $0 \leq x \leq 1$ and $0 \leq t \leq 1$. We can approximate θ in both the x and t dimensions with a finite polynomial sum:

$$\theta(x, t) = \theta_0(x) + \sum_{i=1}^n \sum_{j=1}^m a_{ij} (x - x^i) t^j. \quad (11.2.8)$$

Then the residual function is a function of both space and time, equaling

$$R(x, t) = -\theta_{0xx}(x) + \sum_{i=1}^n \sum_{j=1}^m (a_{ij} (x - x^i) jt^{j-1} - a_{ij} (-i)(i-1)x^{i-2} t^j). \quad (11.2.9)$$

We now have only scalar unknowns, the a_{ij} . To determine those nm scalar constants, we impose the nm projection conditions:

$$\langle R(x, t), \psi_{ij}(x, t) \rangle = 0, \quad i = 1, \dots, n, j = 1, \dots, m, \quad (11.2.10)$$

where $\psi_{ij}(x, t) = (x - x^i)t^j$ is a collection of nm basis functions. Equations (11.2.10) form a system of linear algebraic equations in the unknown coefficients a_{ij} .

We will not go further with this example. These manipulations show how to convert a linear partial differential equation into a linear ordinary differential equation or even into a linear algebraic system. Such transformations can be used generally to construct powerful solution methods. We next describe the projection method for general problems.

11.3 General Projection Method

In the previous two sections we applied projection ideas to familiar problems. We now describe the projection method in a more generally applicable fashion. The ability to do this is a strength of projection methods. We give a step-by-step overview of the approach; then we highlight the critical issues for each step and discuss how the pieces interact.

General Projection Approach

Suppose that we want a solution to the operator equation

$$\mathcal{N}(f) = 0, \quad (11.3.1)$$

where $\mathcal{N}: B_1 \rightarrow B_2$, with B_1 and B_2 complete normed vector spaces of functions $f: D \subset R^n \rightarrow R^m$, and where \mathcal{N} is a continuous map. In our simple ordinary differential equation example (11.1.1), $D = [0, T]$, $f: D \rightarrow R$, $\mathcal{N} = d/dt - I$ where I is the identity operator, B_1 is the space of C^1 functions, and B_2 is the space of C^0 functions. In this example, B_2 contained B_1 , and both are contained in the space of measurable functions. More generally, f is a list of functions in the definition of equilibrium, such as decision rules, price functions, value functions, and conditional expectations functions, and the \mathcal{N} operator expresses equilibrium conditions such as market clearing, Euler equations, Bellman and HJB equations, and rational expectations.

Since we are focused on describing the computational method, we will specify the topological details only to the extent implicit in the computational details. For example, we do not say exactly what norms and inner products are being implicitly used in the ODE and PDE examples in the previous sections. While these topological details are important, they lie far beyond the scope of this book. This may make some readers uncomfortable, particularly when we note that the topological aspects of many of our applications are not well-understood. Some comfort can be taken in the fact that we always check the validity of our solutions, and those checks may keep us from accepting projection method solutions in cases where the underlying functional analytic structure does not support the use of such methods. The reader can also see Zeidler (1986) and Krasnosel'ski and Zabreiko (1984) for serious discussions of those issues, and is encouraged to apply the methods there to economic problems.

The first step is to decide how to represent approximate solutions to $\mathcal{N}(f) = 0$. One general way is to assume that our approximation, \hat{f} , is built up as a linear¹ combination of simple functions from B_1 . We will also need concepts of when two functions are close or far apart. Therefore, the first step is to choose bases and concepts of distance:

STEP 1 Choose a basis over B_1 , $\Phi = \{\varphi_i\}_{i=1}^\infty$ and a norm, $\|\cdot\|$. Similarly choose a basis over B_2 , $\Psi = \{\psi_i\}_{i=1}^\infty$ and an inner product, $\langle \cdot, \cdot \rangle_2$ over B_2 .

1. Nonlinear combinations are also possible, but we stay with linear combinations here, since linear approximation theory is a much more developed theory than nonlinear approximation theory.

When B_1 and B_2 are subsets of another space, we will often use a basis of the larger space as the basis of B_1 and an inner product norm of the larger space as the norm on B_2 . Our approximation to the solution of $\mathcal{N}(f) = 0$ will be denoted \hat{f} ; we next decide how many of these basis elements we will use.

STEP 2 Choose a degree of approximation, n , and define² $\hat{f} \equiv \sum_{i=1}^n a_i \varphi_i(x)$.

Step 1 lays down the structure of our approximation, and step 2 fixes the flexibility of the approximation. Once we have made these basic decisions, we begin our search for an approximate solution to (11.3.1). Since the only unknown part of the approximation is the vector a , we have reduced the original infinite-dimensional problem to a finite-dimensional one. If our diagnostic tests leave us dissatisfied with that approximation, we can return to step 2 and increase n in hopes of getting an improved approximation. If that fails, we can return to step 1 and begin again with a different basis.

Since the true solution f satisfies $\mathcal{N}(f) = 0$, we will choose as our approximation some \hat{f} that makes $\mathcal{N}(\hat{f})$ nearly equal to the zero function, where by near we refer to properties defined by the norm in B_2 , $\|\cdot\|_2$, which corresponds to the inner product $\langle \cdot, \cdot \rangle_2$. Since \hat{f} is parameterized by a , the problem reduces to finding an a which makes $\mathcal{N}(\hat{f})$ nearly zero. In many cases computing $\mathcal{N}(\hat{f})$ is also challenging, such as when $\mathcal{N}(f)$ involves integration of f ; in those cases we need to approximate the \mathcal{N} operator.

STEP 3 Construct a computable approximation, $\hat{\mathcal{N}}$, to \mathcal{N} , and define the residual function

$$R(x; a) \equiv (\hat{\mathcal{N}}(\hat{f}(\cdot; a))(x)).$$

Steps 2 and 3 transform an operation in an infinite-dimensional space into a computable finite-dimensional one. We need next to specify our notion of \hat{f} nearly solving $\mathcal{N}(f) = 0$.

STEP 4 Either compute the norm of $R(\cdot; a)$, $\|R(\cdot; a)\| \equiv \langle R(\cdot; a), R(\cdot; a) \rangle$, or choose a collection of l test functions in B_2 , $p_i: D \rightarrow \mathbb{R}^m$, $i = 1, \dots, l$, and for each guess of a compute the l projections, $P_i(\cdot) \equiv \langle R(\cdot; a), p_i(\cdot) \rangle$.

Step 4 creates the projections we will use. The choices made in step 4 generally give the projection method its name. Projection methods are also called “weighted

2. The convention is that the φ_i increase in “complexity” and “nonlinearity” as i increases, and that the first n elements are used. In the case of standard families of orthogonal polynomials, φ_i is the degree $i - 1$ polynomial.

residual methods,” since the criteria in step 4 weigh the residuals. Once we have chosen our criterion, we can determine the value of the unknown coefficients, a .

STEP 5 Find $a \in R^n$ that either minimizes $\|R(\cdot; a)\|$ or solves $P(a) = 0$.

When we have our solution, \hat{f} , we are not done. It is only a candidate solution, and we must test the quality of our solution before accepting it.

STEP 6 Verify the quality of the candidate solution by approximating $\mathcal{N}(\hat{f})$.

We accomplish step 6 by computing the norm $\|\mathcal{N}(\hat{f})\|$ and/or projections of $\mathcal{N}(\hat{f})$ against test functions not used in step 4. If \mathcal{N} must be approximated, we should, if feasible, use a better approximation in step 6 than that constructed in step 3. Ideally all of these quantities will be zero; in practice, we choose target quantities for these diagnostics that imply that the deviations from zero are not economically significant.

This general algorithm breaks the numerical problem into several distinct steps. It points out the many distinct techniques of numerical analysis which are important. First, in steps 1 and 2 we choose the finite-dimensional space wherein we look for approximate solutions, hoping that within this set there is something “close” to the real solution. These steps require us to think seriously about approximation theory methods. Second, step 4 will involve numerical integration if we cannot explicitly compute the integrals that define the projections, and step 3 frequently involves numerical integration in economics applications. Third, step 5 is a third distinct numerical problem, involving the solution of a nonlinear set of simultaneous equations or the solution of a minimization problem. We will now consider each of these numerical problems in isolation.

Choice of Basis and Approximation Degree

There are many criteria that the basis and inner product should satisfy. The full basis Φ for the space of candidate solutions should be “rich,” flexible enough to approximate any function relevant to the problem. The best choice of n cannot be determined a priori. Generally, the only correct choice is $n = \infty$. If the choice of the basis is good, then larger n will yield better approximations. We are most interested, however, in the smallest n that yields an acceptable approximation. We initially begin with small n and increase n until some diagnostic indicates little is gained by continuing. Computational considerations also play a role in choosing a basis. The φ_i should be simple to compute, and all be similar in size to avoid scaling problems.

Of course, the number of basis elements needed will depend greatly on the basis being used; using a basis that is well-suited to a problem can greatly improve performance. Approximation theory, discussed in chapter 6, can be used to evaluate

alternative bases because ultimately we are trying to approximate the solution f with a finite combination of simple known functions. The basis elements should look something like the solution so that only a few elements can give a good approximation. While asymptotic results such as the Stone-Weierstrass theorem may lull one into accepting polynomial approximations, practical success requires a basis where only a few elements will do the job. The individual terms should also be “different”; ideally they should be orthogonal with respect to the inner product $\langle \cdot, \cdot \rangle$. The reasons are essentially the same as why one wants uncorrelated explanatory variables in a regression. Nonorthogonal bases will reduce numerical accuracy just as multicollinear regressors enlarge confidence intervals. Algorithms that solve for the unknown coefficients a solve several linear systems of equations, and the accuracy of these solutions depends on the rows and columns not being collinear. Orthogonal bases will help avoid ill-conditioned matrices in these intermediate steps.

From chapter 6 we know that there are several possible bases. First, let us consider the ordinary polynomials, $\{1, x, x^2, x^3, \dots\}$. If B_1 is the set of bounded measurable functions on a compact set then the Stone-Weierstrass theorem assures us of their completeness in the L_1 norm. However, they may not be a good choice since they are too similar. For example, they are all monotonically increasing and positive on R^+ , and they will not be orthogonal in any natural inner product on R^+ . They will also vary a great deal in size over most intervals D . The ordinary polynomials will sometimes be adequate, as they were in our simple examples, because we needed few terms to get a good solution.

These considerations do not mean that we cannot use ordinary polynomials, just that it is preferable to use polynomial bases that are orthonormal with respect to the inner product. A generally useful choice are systems of Chebyshev polynomials, which were discussed in chapter 6. Nonpolynomial alternatives include various sequences of trigonometric and exponential functions. The choice depends on the range, D , computational demands, and the expected shape of a solution. In physics, trigonometric bases such as $\{1, \sin x, \sin 2x, \sin 3x, \dots\}$ are often used, since solutions are often periodic, allowing for Fourier series techniques. In economic problems, however, it is better to use nontrigonometric bases, since solutions are generally not periodic and periodic approximations to nonperiodic functions require many terms.

We can use the full array of approximation methods discussed in chapter 6. Some families of projection methods are known by their method of approximation. *Spectral methods* use bases where each element is nonzero almost everywhere, such as in trigonometric bases and orthogonal polynomials. *Finite element* methods use bases where each element has small support, as discussed in section 6.13.

Most interesting problems in economics involve more than one state variable—physical versus human capital, capital stocks of oligopolistic competitors, wealth distribution across investor groups, and so on. The tensor product methods discussed in Chapter 6 build up multidimensional basis functions from simple one-dimensional basis functions. The curse of dimensionality, a problem that arises with tensor product bases, can be avoided by using the complete polynomials as a basis instead.

We are not limited to the conventional approaches described in chapter 6. If we have some reason to believe that a solution will look like some nonconventional functions, then we should try them. We may want to orthogonalize the family, and we may need to develop the corresponding Gaussian quadrature formulas, but all that is just a straightforward application of the methods discussed in chapters 6 and 7. This may be quite important in multidimensional problems because we will need to economize on basis elements. In chapter 15 we will discuss formal ways to generate good problem-specific bases. Even though we will focus here on using standard approximation methods, the ideas of projection methods generalize directly to more idiosyncratic choices.

Choice of Projection Conditions

As we have seen in our examples, projection techniques include a variety of special methods. In general, we specify some inner product, $\langle \cdot, \cdot \rangle$, of B_2 , and use $\langle \cdot, \cdot \rangle$ to measure the “size” of the residual function, R , or its projection against the test functions. We can use inner products of the form

$$\langle f(x), g(x) \rangle \equiv \int_D f(x)g(x)w(x) dx$$

for some weighting function $w(x)$, but there is no reason why we are limited to them. In choosing the norm, one should consider exactly what kind of error should be small and find a norm that will be sensitive to the important errors. There are several ways to proceed.

The general *least squares projection method* computes the L^2 norm of the residual function, namely $\langle R(x; a), R(x; a) \rangle$, and chooses a so as to minimize the “sum of squared residuals”:

$$\min_a \langle R(x; a), R(x; a) \rangle.$$

We have thereby reduced the problem of solving a functional equation to solving a nonlinear minimization problem in R^n , a more tractable problem. Of course, the standard difficulties will arise. For example, there may be local minima which are not

global solutions. However, there is no reason for these problems to arise more often here than in any other context, such as maximum likelihood estimation, where minimization problems are solved numerically.

The least squares method is a direct implementation of the idea to make small the error of the approximation. In general, one could develop alternative implementations by using different norms. However, most projection techniques find a good-fitting approximation in less direct fashions. For these techniques the basic idea is that the true solution would have a zero residual error function; in particular, its projection in all directions is zero. Therefore one way to find the n components of a is to fix n projections and choose a so that the projection of the resulting residual function in each of those n directions is zero. Formally these methods find a such that $\langle R, p_i \rangle = 0$ for some specified collection of test functions, p_i . Different choices of the p_i defines different implementations of the projection method.

It is clear that the least squares and alternative implementations of projection ideas are similar since one way to solve the least squares approach is to solve the nonlinear set of equations generated by its first-order conditions, $\langle R, \partial R / \partial a_i \rangle = 0$. Seeing the least squares method expressed as a system of projection equations gives us some indication why other methods may be better. The projection directions in the least squares case, the gradients of the residual function, could be highly correlated. Furthermore the projection directions depend on the guess for a . This lack of control over the implicit projection directions is not a good feature. Also in economic problems we may have a preference for approximations that have zero projections in certain directions, such as the average error in an Euler equation. Many of the alternative techniques will naturally include that condition.

One such alternative technique is the *Galerkin* method, also known as the *Bubnov-Galerkin* or *Galerkin-Petrov* method. In the Galerkin method we use the first n elements of the basis for the projection directions, where we are making the weak assumption that Φ , our basis of B_1 , lies in B_2 . Therefore a is chosen to solve the following set of equations:

$$P_i(a) \equiv \langle R(x; a), \varphi_i(x) \rangle = 0, \quad i = 1, \dots, n.$$

Notice that here we have reduced the problem of solving a differential equation to one of solving a set of nonlinear equations. In some cases the Galerkin projection equations are the first-order conditions to some minimization problem, as is often the case in linear problems from physics. When we have such an equivalence, the Galerkin method is also called the *Rayleigh-Ritz* method. This is not as likely to happen in economics problems because of nonlinearities.

The method of moments, subdomain, and collocation procedures can be applied to the general setting. If $D \subset R$, then the *method of moments* chooses the first n polynomials for the projection directions; that is, we find a that solves the system

$$P_i(a) \equiv \langle R(x; a), x^{i-1} \rangle = 0, \quad i = 1, \dots, n.$$

If D is of higher dimension, then we project R against a sufficient number of low-order multivariate monomials. In the *subdomain method* the idea is to find an approximation that is good on average on a collection of subsets that cover the whole domain. More specifically, we choose a so that

$$P_i(a) \equiv \langle R(x; a), I_{D_i} \rangle = 0, \quad i = 1, \dots, n,$$

where $\{D_i\}_{i=1}^n$ is a sequence of intervals covering D , and I_{D_i} is the indicator function for D_i .

The *collocation method* chooses a so that the functional equation holds exactly at n fixed points. That is, we choose a to solve

$$R(x_i; a) = 0, \quad i = 1, \dots, n,$$

where $\{x_i\}_{i=1}^n$ are n fixed points from D . This is a special case of the projection approach, since $R(x_i; a)$ equals $\langle R(x; a), \delta(x - x_i) \rangle$, the projection of $R(x; a)$ against the Dirac delta function at x_i .

Orthogonal collocation is the method where the x_i are the n zeros of the n th orthogonal polynomial basis element and the basis elements are orthogonal with respect to the inner product. It is a particularly powerful application of projection ideas when used with a Chebyshev polynomial basis. This is not a surprise in light of the Chebyshev interpolation theorem. Suppose that $D = [-1, 1]$ and $R(z_i^n; a) = 0$, $i = 1, \dots, n$, where the z_i^n are the n zeros of T_n . As long as $R(x; a)$ is smooth in x , the Chebyshev interpolation theorem says that these zero conditions force $R(x; a)$ to be close to zero for all $x \in [-1, 1]$. The optimality of Chebyshev interpolation also says that if one is going to use collocation, these are the best possible points to use. Even after absorbing these considerations, it is not certain that even Chebyshev collocation is a reliable method. We will see below that its performance is surprisingly good.

Collocation can be used for bases other than orthogonal polynomials. *Spline collocation* methods use spline bases. The collocation points could be the spline nodes themselves or some other set of points, such as the midpoints of the spline mesh. The key objective is keeping the Jacobian of the collocation equation system well-conditioned.

Evaluation of Projections

The meat of the problem is step 4 whose the major computational task is the computation of those projections. The collocation method is fastest in this regard because it only uses the value of R at n points. More generally, the projections will involve integration. In some cases one may be able to explicitly perform the integration. This is generally possible for linear problems, and possible for special nonlinear problems. However, our experience with the economic applications below is that this will generally be impossible for nonlinear economic problems. We instead need to use quadrature techniques to compute the integrals associated with the evaluation of $\langle \cdot, \cdot \rangle$. A typical quadrature formula approximates $\int_a^b f(x) w(x) dx$ with a finite sum $\sum_{i=1}^n \omega_i f(x_i)$ where the x_i are the quadrature nodes and the ω_i are the weights. Since these formulas also evaluate R at just a finite number of points, quadrature-based projection techniques are essentially weighted collocation methods. The advantage of quadrature formulas is that information at more points is used to compute a more accurate approximation of the projections.

Finding the Solution

Step 5 uses either a minimization algorithm or a nonlinear algebraic equation solver. If the system $P(a) = 0$ is overidentified or if we are minimizing $\|R(\cdot; a)\|$, we may invoke a nonlinear least squares algorithm. The nonlinear equations associated with Galerkin and other inner product methods can be solved by the variety of methods discussed in chapter 5. While fixed-point iteration appears to be popular in economics, Newton's method and its refinements have often been successful. Homotopy methods can also be used if one has no good initial guesses.

Initial Guesses

Good initial guesses are important since projection methods involve either a system of nonlinear equations or optimizing a nonlinear, possibly multimodal, objective. Fortunately this is generally not a big problem. Often there are degenerate cases for which we can find the solution, which in turn will be a good guess for the problem we want to solve. The perturbation methods discussed in chapters 13 and 14 often generate good initial guesses. In some problems there are problem-specific ways of generating good initial guesses.

There is one general approach which is often useful. The least squares approach may not be a good one to use for high-quality approximations. However, it may yield low-quality approximations relatively quickly, and, since the least squares method is an optimization method, convergence to a local extrema is ensured even if one has

no good initial guess. Furthermore, by adding terms to the least squares objective, one can impose sensible restrictions on the coefficients to eliminate economically nonsensical extrema. These facts motivate a two-stage approach. First, one uses a least squares approach with a loose convergence criterion to quickly compute a low-quality approximation. Second, one uses this approximation as the initial guess for a projection method attempting to compute a higher-order approximation. With some luck the least squares solution will be a good initial guess for the second computation. If it is difficult to find good initial guesses, then one can use homotopy methods that are globally convergent.

Coordination among Steps 1–5

We now see what is needed for efficiency. The key is to choose elements for the separate steps that work well together. We need basis functions that are easy to evaluate because they will be frequently evaluated. Any integration in steps 3 and 4 must be accurate but fast. Therefore we should use quadrature formulas that work well with the basis. The nonlinear equation solver in step 5 needs to be efficient and should be able to use all the information arising from step 4 calculations. Step 5 will typically use gradient information about the integrals of step 4. It is therefore important to do those gradient calculations quickly, doing them analytically when practical.

A particularly important interaction is that between the choice of a basis and the solution of the nonlinear problem in R^n . Most methods for solving the system $P(a) = 0$ will use its Jacobian, $P_a(a) = 0$. If this matrix is nearly singular near the solution, accuracy will be poor due to round-off error and convergence will be slow. Choosing an orthogonal basis, or nearly orthogonal basis (as is the case with B -splines), will substantially reduce the likelihood of a poorly conditioned Jacobian, even in nonlinear problems.

Most methods used in numerical analysis of economic models fall within the general description for projection methods. We will see these connections below when we compare how various methods attack a common problem. The key fact is that the methods differ in their choices of basis, fitting criterion, and integration technique.

Evaluating a Solution

As with operator equation methods in general, the projection algorithm does not automatically evaluate the quality of the candidate approximate solution. One of the advantages of the projection approach is the ease with which one can do the desired evaluation. The key observation is that we typically use an approximation to \mathcal{N} , $\hat{\mathcal{N}}$, when searching for the approximate solution, \hat{f} . For example, we use numerical integration methods to compute conditional expectations and to compute projec-

tions. To economize on computer time, we use the least amount of information possible to compute \hat{f} .

This is a risky but acceptable strategy, since we accept \hat{f} only after we strenuously test the candidate \hat{f} . Therefore we use better quadrature rules here to evaluate \hat{f} , and we check if $0 = (\hat{\mathcal{N}}(\hat{f}))(x)$ at many points that were not used in the derivation of \hat{f} . We also use a finite number of test functions in constructing \hat{f} , leaving an infinite number of test functions that we can use to evaluate \hat{f} . While this discussion is abstract here, the actual implementation will be quite intuitive in actual applications.

Existence Problems

Projection methods are useful ways to transform infinite-dimensional problems into finite-dimensional problems. However, these transformations present problems that we have not faced in previous methods—existence. In previous chapters we investigated methods that were more similar to the problem being analyzed. For example, in chapter 5 we examined nonlinear equations, where there is a solution to the numerical problem if and only if there is a solution to the original, pure problem. This can be different here. Sometimes the finite-dimensional problem generated by a projection method will not have a solution even when the original, pure problem does have a solution. Therefore, if one is having difficulty solving a projection equation system for a particular basis and particular n , the problem may go away just by trying another n or another basis. For well-behaved problems, choosing a sufficiently large n will work.

One way to ensure existence of a solution but gain some of the advantages of the projection methods is to construct a least squares objective from the projections. For example, in the case of the Galerkin method, one could solve the least squares problem

$$\min_a \sum_{i=1}^n \langle R(x; a), \varphi_i(x) \rangle^2.$$

One could use an *overidentification* approach and solve the problem

$$\min_a \sum_{i=1}^m \langle R(x; a), \varphi_i(x) \rangle^2$$

for some $m > n$. These combinations of least squares and projection ideas are compromises. Existence of the finite-dimensional approximation is assured as long as the objectives are continuous, and optimization methods can be reliably used to find solutions.

In beginning a numerical analysis of a model, it is important to maintain flexibility as to which method to use. Since the projection methods are so similar, it is easy to change from one to another. Experimentation with several methods is the only way to find out which will be best.

Consistency

When using numerical procedures, it is desirable to know something concerning the error of the solution. As we discussed in chapter 2, an important focus of theoretical numerical analysis is deriving error bounds and proving that methods are asymptotically valid. For example, we would like to know that the errors of a projection method go to zero as we enlarge the basis; this property is called *consistency*. There has been little work on proving that the algorithms used by economists are asymptotically valid.

The absence of convergence theorems does not invalidate the projection approach. The compute and verify approach will help avoid bad approximations. Even if we had a consistent method, we would have to develop a stopping criterion for n , the degree of the approximation, which itself would be a verification procedure. This is in keeping with our philosophy that a convergence theorem is not necessary for a procedure to be useful nor do convergence theorems make any particular result more reliable. In practice, we must use a method which stops at some finite point, and any candidate solution produced by any method must be tested before it is accepted.

11.4 Boundary Value Problems

Boundary value problems are commonly solved by projection methods. In this section we will apply projection methods to solve continuous-time boundary value problems arising in life-cycle problems.

Consider the two-point boundary value problem for $x, y \in R$:

$$\begin{aligned}\dot{x} &= f(x, y, t), \\ \dot{y} &= g(x, y, t), \\ x(0) &= x_0, \quad y(T) = y_T.\end{aligned}\tag{11.4.1}$$

To solve this, we approximate $x(t)$ and $y(t)$ with polynomials

$$\hat{x}(t) = \sum_{i=0}^n a_i t^i, \quad \hat{y}(t) = \sum_{i=0}^n b_i t^i.\tag{11.4.2}$$

To fix a and b , we first impose the boundary conditions

$$x_0 = \hat{x}(0) = a_0,$$

$$y_T = \hat{y}(T) = \sum_{i=0}^n b_i T^i. \quad (11.4.3)$$

Since a and b contain $2n + 2$ unknown parameters, we also impose the $2n$ projection conditions

$$\int_0^T \left(\frac{d}{dt} (\hat{x}(t)) - f(\hat{x}(t), \hat{y}(t), t) \right) \varphi_i(t) w(t) dt = 0, \quad i = 1, \dots, n, \quad (11.4.4)$$

$$\int_0^T \left(\frac{d}{dt} (\hat{y}(t)) - g(\hat{x}(t), \hat{y}(t), t) \right) \varphi_i(t) w(t) dt = 0, \quad i = 1, \dots, n,$$

for some functions $\varphi_i(t)$ and some weighting functions $w(t)$. Equations (11.4.3) and (11.4.4) give us a total of $2n + 2$ conditions that hopefully pin down the $2n + 2$ parameters.

There are many kind of BVP's which can be treated in the same way. For example, the boundary conditions could be $x(0) = x_0$ and $x(T) = x_T$; in this case (11.4.3) becomes $x_0 = \hat{x}(0) = a_0$ and $x_T = \hat{x}(T) = \sum_{i=0}^n a_i T^i$. Whatever the boundary conditions in (11.4.1), simple substitution can be used to produce the numerical conditions in (11.4.3).

Continuous-Time Life-Cycle Consumption Models

We next use simple projection methods to solve a simple life-cycle version of the life-cycle problem (10.6.10) with utility function $u(c) = c^{1+\gamma}/(1+\gamma)$ and asset return function $f(A) = rA$. After some manipulation the optimal c and A paths are characterized by the boundary value problem

$$\begin{aligned} \dot{c} &= \gamma^{-1} c(\rho - r), \\ \dot{A} &= rA + w - c, \\ A(0) &= 0 = A(T). \end{aligned} \quad (11.4.5)$$

Projection methods are often used to solve boundary value problems. We next turn to a particular case of (11.4.5) with $\rho = 0.05$, $r = 0.10$, $\gamma = -2$, $w(t) = 0.5 + t/10 - 4(t/50)^2$, and $T = 50$. We will use degree 10 approximations for both $c(t)$ and $A(T)$:

$$\begin{aligned} A(t) &= \sum_{i=0}^{10} a_i T_i \left(\frac{t-25}{25} \right), \\ c(t) &= \sum_{i=0}^{10} c_i T_i \left(\frac{t-25}{25} \right), \end{aligned} \tag{11.4.6}$$

where $T_i(x)$ is the degree i Chebyshev polynomial. We use a linear transformation in the argument of the T_i functions that maps $t \in [0, 50]$ into $[-1, 1]$, the domain of the Chebyshev polynomials. The functions $c(t)$ and $A(t)$ must approximately solve the two point BVP

$$\begin{aligned} \dot{c}(t) &= -\frac{1}{2} c(t)(0.05 - 0.10), \\ \dot{A}(t) &= 0.1A(t) + w(t) - c(t), \\ A(0) &= A(T) = 0. \end{aligned} \tag{11.4.7}$$

We define the two residual functions

$$\begin{aligned} R_1(t) &= \dot{c}(t) - 0.025c(t) \\ R_2(t) &= \dot{A}(t) - \left(0.1A(t) + \left(0.5 + \frac{t}{10} - 4\left(\frac{t}{50}\right)^2 \right) - c(t) \right). \end{aligned} \tag{11.4.8}$$

We want to choose coefficients a_i , $i = 0, 1, \dots, 10$, and c_i , $i = 0, 1, \dots, 10$, so that the residual functions are nearly zero and so that the boundary conditions $A(0) = A(T) = 0$ are satisfied. Since the boundary conditions impose two conditions, we need 20 more conditions to determine the 22 unknown coefficients. We next determine the collocation points for our residuals. Since we need 20 conditions for the two equations, we need only 10 collocation points. We will therefore use the 10 zeros of $T_{10}((t-25)/25)$, which are $\mathcal{C} \equiv \{0.31, 2.72, 7.32, 13.65, 21.09, 28.91, 36.35, 42.68, 47.28, 49.69\}$. We then choose the a_i and c_i coefficients, which solve

$$\begin{aligned} R_1(t_i) &= 0, \quad t_i \in \mathcal{C}, \quad i = 1, \dots, 10, \\ R_2(t_i) &= 0, \quad t_i \in \mathcal{C}, \quad i = 1, \dots, 10, \\ A(0) &= \sum_{i=1}^{10} a_i (-1)^i = 0, \\ A(50) &= \sum_{i=1}^{10} a_i = 0. \end{aligned} \tag{11.4.9}$$

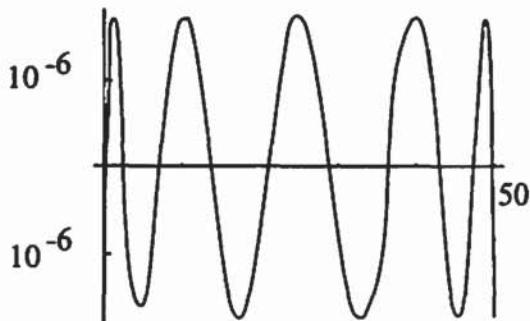


Figure 11.1
Residuals of Chebyshev collocation applied to (11.4.7): c equation

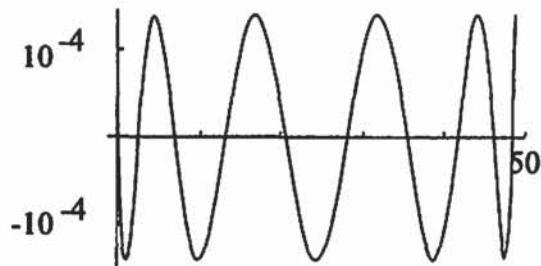


Figure 11.2
Residuals of Chebyshev collocation applied to (11.4.7): k equation

These are 22 linear equations in 22 unknowns. The system is nonsingular; therefore there is a unique solution. The true solution to the system (11.4.7) can be solved since it is a linear problem. Figures 11.1 to 11.4 illustrate the accuracy of the collocation procedure. Figures 11.1 and 11.2 display the residuals for both the consumption and asset equations. Note the near-equioscillation property of the residuals. This is almost expected because we are using Chebyshev ideas to approximate the residuals directly. The fact that the residuals nearly satisfy equioscillation implies that the implicit polynomial approximation is nearly optimal in the L^∞ sense. Figure 11.3 illustrates the relative errors in the consumption solution, and figure 11.4 displays the errors of the asset solution relative to the maximum asset level in the true solution. The consumption error is less than 0.05 of 1 percent. The asset error is similarly small. The fact that the errors in the asset and consumption paths also nearly satisfy equioscillation implies that their approximations are also close to being optimal. The fact that the amplitudes are not equal says that they are not the best possible, but that is not critical. The fact that we have the desired amount of oscillation is most critical. If instead the errors were linear without any oscillation, then we would know that we

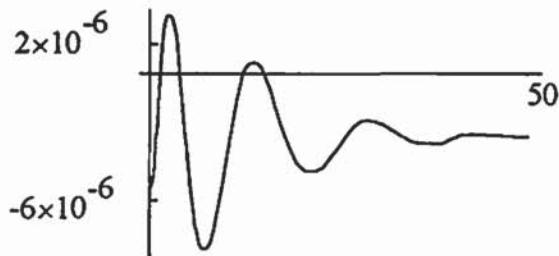


Figure 11.3
Relative error of Chebyshev collocation consumption solution of (11.4.7)

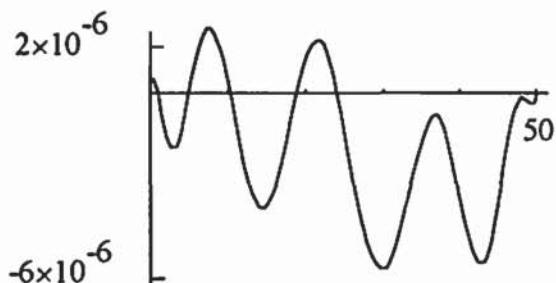


Figure 11.4
Relative error of Chebyshev collocation asset solution of (11.4.7)

could do much better. These graphs illustrate the importance and usefulness of the equioscillation theorem of chapter 6.

11.5 Continuous-Time Growth Model

We again consider the canonical continuous-time optimal growth problem for an economy with one good and one capital stock:

$$\max_c \int_0^\infty e^{-\rho t} u(c) dt$$

$$\text{s.t. } \dot{k} = f(k) - c.$$

In chapter 10 we noted that the optimal policy function, $C(k)$, satisfied the differential equation

$$0 = C'(k)(f(k) - C(k)) - \frac{u'(C(k))}{u''(C(k))} (\rho - f'(k)) \equiv \mathcal{N}(C) \quad (11.5.1)$$

together with the boundary condition that $C(k^*) = f(k^*)$, where k^* is the steady state capital stock and is fixed by the condition $f'(k^*) = \rho$.

We use (11.5.1) to solve for $C(k)$. We assume that $f(k) = \rho k^\alpha / \alpha$ and that $u(c) = c^{1+\gamma} / (1 + \gamma)$ with $\rho = 0.04$, $\alpha = 0.25$ and $\gamma = -2$; this implies a steady state $k^* = 1$. We use a basis of Chebyshev polynomials to approximate $C(k)$, and compute $C(k)$ for $k \in [0.25, 1.75]$. With the approximation $\hat{C}(k; a) \equiv \sum_{i=1}^n a_i T_i((k-1)/0.75)$ and the definition $R(k; a) = \mathcal{N}(\hat{C}(\cdot; a))(k)$ a collocation method can solve for a , by solving the system of equations

$$R(k_i; a) = 0, \quad i = 1, \dots, n,$$

where the k_i are the n zeros of $T_n((k-1)/0.75)$.

The performance of the algorithm is very good, independent of the details of the implementation. The only critical element is the choice of the initial guess. However, the crude approximation consisting of the linear policy that consumes nothing at a zero capital stock and consumes all output at the (easily computed) steady state serves as a satisfactory initial guess. Tenth-order polynomial approximations are easily found in a few seconds and all diagnostics (e.g., the location of the steady state and slope at the steady state) all indicated a very good approximation. This algorithm easily outperformed the more commonly used shooting approach to the problem.

In table 11.3 we display the solution for various polynomial degrees and capital stocks. $\hat{C}^j(k)$ is the degree j collocation solution and $\hat{E}^j(k)$ equals the residual function of the degree j solution divided by $\rho C^j(k)$ to take out the units. Table 11.3 shows that the degree 12 solution has residuals little larger than machine zero, and that the degree 5 and 8 solutions are also very good. We did not use the fact that we know $C(k^*) = 0.2$. Also we do not use the point $k = k^*$ as a collocation point when computing the fifth-degree approximation. Note that the degree 8 and 12 solutions agree to six digits on the points used in table 11.3. All of these diagnostics indicate that the approximation is quite good. In fact all three solutions do an excellent job at getting $C(k^*)$ right when we compare the projection method results with the reverse shooting results.

11.6 Computing Conditional Expectations

Many problems in economics require us to compute conditional expectation functions. If Y and X are random variables, the *conditional expectation of Y given X* , denoted $E\{Y|X\}$, is a function of X , $\psi(X)$, such that

$$E\{(Y - \psi(X))g(X)\} = 0 \tag{11.6.1}$$

Table 11.3
Projection methods applied to (11.5.1)

k	$\hat{C}^2(k)$	$\hat{E}^2(k)$	$\hat{C}^5(k)$	$\hat{E}^5(k)$	$\hat{C}^8(k)$	$\hat{E}^8(k)$	$\hat{C}^{12}(k)$	$\hat{E}^{12}(k)$
0.6	0.158905	-9(-3)	0.159640	-2(-3)	0.159638	4(-6)	0.159638	-9(-9)
0.8	0.180273	-2(-2)	0.180924	-2(-4)	0.180922	-2(-6)	0.180922	-1(-8)
1.0	0.200000	5(-16)	0.199991	-2(-4)	0.200000	-5(-16)	0.200000	5(-16)
1.2	0.218086	1(-2)	0.217543	1(-4)	0.217543	1(-6)	0.217543	7(-9)
1.4	0.234531	4(-3)	0.233944	-9(-5)	0.233941	-2(-6)	0.233941	7(-9)

for all continuous functions g . This says that the error $Y - \psi(X)$ is uncorrelated with all functions of X .

We seek a function $\hat{\psi}(X)$ which approximates $E\{Y|X\}$. To do so, we construct

$$\hat{\psi}(X; a) = \sum_{i=0}^n a_i \varphi_i(X), \quad (11.6.2)$$

where $\{\varphi_i(X)\}_{i=0}^\infty$ is a basis for continuous functions of X . This reduces the problem to finding the a coefficients in $\hat{\psi}$. Suppose that there is a random variable Z such that $Y = g(Z)$ and $X = h(Z)$. Then the least squares coefficients a solve

$$\min_a E\{(\hat{\psi}(h(Z); a) - g(Z))^2\}. \quad (11.6.3)$$

There are two ways to compute the unknown coefficients. First, the Monte Carlo approach generates a sample of (Y, X) pairs, then regresses the values of Y on X . More precisely, we construct a random number generator tuned to model the joint distribution (Y, X) and use it to produce the set of pairs, $\{(y_i, x_i) \mid i = 1, \dots, N\}$. We then fix a by solving the least squares problem

$$\min_a \sum_i (\hat{\psi}(x_i; a) - y_i)^2. \quad (11.6.4)$$

The second approach uses projection ideas instead of probabilistic ideas. Since $\psi(X) = E\{Y|X\}$, then the projection condition $E\{(g(Z) - \hat{\psi}(h(Z))) \varphi_i(h(Z))\} = 0$ holds for all i . To fix the coefficients a in an approximation $\hat{\psi}(\cdot; a)$, we impose the $n+1$ projection conditions

$$E\{(g(Z) - \hat{\psi}(h(Z); a)) \varphi_i(h(Z))\} = 0, \quad i = 0, \dots, n. \quad (11.6.5)$$

The system (11.6.5) is a linear equation in the a coefficients. Each of the equations in (11.6.5) involves an integral; we use deterministic quadrature methods to evaluate them.

The two methods are related. The least-squares problem in (11.6.3) leads to first-order conditions similar to (11.6.5) and the integrals in (11.6.5) could be approximated by pseudo-Monte Carlo methods. Of course quasi-Monte Carlo methods could also be used to compute the integrals in (11.6.5).

To indicate the relative quality of these procedures, we apply them to a simple example. Let $Y, W \sim U[0, 1]$ and $X = \varphi(Y, W) = (Y + W + 1)^2$; then $E\{Y|X\} = (X^{1/2} - 1)/2$. We let $N = 1,000$ and produce 1,000 (y, w) pairs. We compute the 1,000 values of $x = \varphi(y, w)$. We then regress y on $1, x, x^2, x^3$, and x^4 , producing

$$\hat{\psi}_{MC}(x) = -0.1760 + 0.2114x - 0.0075x^2 - 0.0012x^3 + 0.0001x^4. \quad (11.6.6)$$

Straightforward calculation shows that the L^2 norm of $\hat{\psi}_{MC} - \psi$ is 0.0431.

We next try a projection approach. We project the prediction error $\hat{\psi}(\varphi(y, w); a) - y$ against the polynomials $1, x, x^2, x^3, x^4$, and x^5 by forming the five equations

$$\int_0^1 \int_0^1 (\hat{\psi}(\varphi(y, w); a) - y) \varphi(y, w)^k dw dy = 0 \quad (11.6.7)$$

for $k = 0, 1, 2, 3, 4$. Since $\hat{\psi}$ is linear in a , (11.6.7) is a linear system of equations in the unknown coefficients a . Moreover the integrals in (11.6.7) are easy to compute analytically; even if they weren't analytically integrable, the integrals in (11.6.7) can be efficiently approximated by deterministic methods since φ is smooth. The solution to (11.6.7) implies the approximate conditional expectation function

$$\hat{\psi}_P = -0.2471 + 0.2878x - 0.0370x^2 + 0.0035x^3 - 0.0001x^4.$$

The L^2 norm of $\hat{\psi}_P - \psi$ is 0.0039, ten times less than the L^2 error of the $\hat{\psi}_{MC}$ approximation displayed in (11.6.6). Furthermore $\hat{\psi}_P$ is faster to compute than $\hat{\psi}_{MC}$.

We see that the projection method is far more efficient than Monte Carlo sampling in computing the conditional expectation $E(Y|X)$. The reason is that the conditional expectation is a smooth function of X , and can be well-approximated by a polynomial, in which case we know from chapters 6 and 11 that we only need a few well-chosen points to determine the unknown function. The Monte Carlo approach (or a pseudo-Monte Carlo approach) generates (y, x) pairs in a disorganized and inefficient way, and leads to an inferior approximation of $E\{Y|X\}$.

11.7 Further Reading and Summary

Projection methods are powerful alternatives to discretization methods. The basic idea is to choose a parameterization of the critical functions and then use either

optimization or nonlinear equation solution methods to fix the parameterization's coefficients. Some implementations resemble nonlinear least squares procedures, but we can often do much better. We have seen that the basic ideas can be applied to a variety of mathematical and economic problems. Their flexibility and the ease with which they can be applied to new problems make them particularly valuable in economics where many problems do not fit into the standard differential and integral equation classifications.

Boyd (1989) is a reader-friendly introduction to spectral projection methods, whereas Canuto et al. (1988), Fletcher (1984), and Gottlieb and Orszag (1977) are more formal. We have focused on spectral methods. Finite element methods are the focus of Akin (1982), Burnett (1987), and White (1985). Judd (1992) presents the basic ideas in the context of a few simple dynamic growth problems, and Gaspar and Judd (1997) examine some multidimensional problems.

Exercises

1. Solve the Spence signaling example in equation (10.4.5) and table 6.1 using projection methods.
2. Solve the problem (11.2.1–3) using the method in (11.2.10) for Chebyshev, Legendre, and ordinary polynomial bases.
3. Solve the life-cycle problem

$$\max_{c,l} \int_0^T e^{-\rho t} (u(c) - v(l)) dt$$

$$\text{s.t. } \dot{A} = rA + wl - c,$$

$$A(0) = 0 = A(T),$$

where

$$w(t) = 1 - \left(\frac{t - T/2}{T/2}\right)^2, \quad u(c) = \frac{c^{1+\gamma}}{\gamma+1}, \quad v(l) = \frac{l^{\eta+1}}{\eta+1}.$$

Assume that $\rho = 0.04$, $r = 0.08$, $T = 50$, $\gamma \in \{-0.5, -1.1, -2.0, -5.0\}$, and $\eta \in \{0.1, 1, 2\}$.

- a. Compute average wealth over the life-cycle.
- b. Suppose interest income is taxed at rate τ . Compute resulting utility and present value of revenue. Plot the trade-off between utility and revenue.
- c. Repeat b for a proportional wage tax.
- d. Compare the welfare cost of wage income taxation and interest income taxation.
4. Use the projection approach to solve the discrete-time life-cycle problem

$$\max_c \sum_0^T \beta^t u(c_t)$$

s.t. $A_{t+1} = (1+r)A_t + w_t - c_t,$

$$A(0) = A(T) = 0,$$

with $u(c) = c^{1+\gamma}/(1+\gamma).$

5. Solve exercise 10.5 using projection methods.
6. Solve exercise 10.6 using projection methods.
7. Solve exercise 10.7 using projection methods.
8. Solve exercise 10.8 using projection methods.

12 Numerical Dynamic Programming

Dynamic programming is a fundamental tool of dynamic economic analysis. Theoretical properties are well-understood (see Bertsekas 1976, 1995, Bertsekas and Shreve 1978; Puterman 1995). Stokey and Lucas (1989) develop the theory of dynamic programming and present some applications of dynamic programming in economic analysis.

Various numerical techniques have been developed in the operations research literature. This chapter examines the two basic numerical approaches to dynamic programming problems. We first discuss finite-state problems. These problems are straightforward, with much of the work reducing to basic matrix operations. We describe the value function iteration method for finite-state problems and various accelerations. Finite-state problems can be used to approximate continuous-state problems via discretization, but this approach is often inefficient. This motivates the second approach of using continuous methods, building on the approximation methods of chapter 6, for dynamic programming problems with continuous state spaces.

12.1 Discrete-Time Dynamic Programming Problems

In dynamic programming we are concerned with controlling a dynamic process that takes on several possible states, can be influenced by the application of controls, and yields a stream of state- and control-dependent payoffs. We first specify the problem in each period. Let X be the set of states, and \mathcal{D} the set of controls. The flow of payoffs to the controller in period t is $\pi(x, u, t)$ if $x \in X$ is the state at the beginning of period t , and the control $u \in \mathcal{D}$ is applied in period t . There may be time- and state-contingent constraints on the controls; let $D(x, t) \subseteq \mathcal{D}$ be the nonempty set of controls which are feasible in state x at time t . We let x_t and u_t denote the state and control in period t .

We next specify the dynamics of the process. If x is the state at the beginning of period t and u is the control, then let $F(A; x, u, t)$ be the probability that $x_{t+1} \in A \subseteq X$. We assume that $F(\cdot; x, u, t)$ describes a probability measure over the states X for each state x , control u , and time t .

Finite-Horizon Problems

In the finite-horizon case we assume that the objective of the controller is to maximize expected total returns over a $T + 1$ period horizon,

$$E \left\{ \sum_{t=1}^T \pi(x_t, u_t, t) + W(x_{T+1}) \right\}, \quad (12.1.1)$$

where $W(x)$ is the terminal valuation and x_1 , the initial state, is given. The analysis of a dynamic programming problem revolves around the *value function*, $V(x, t)$, which is defined to be the greatest feasible total expected payoff from time t forward if the time t state is x . More precisely, let $\mathcal{U}(x, t)$ be the set of all feasible rules for choosing u at time s , $s \geq t$, beginning from state x in period t . Then V is defined by

$$V(x, t) \equiv \sup_{\mathcal{U}(x, t)} E \left\{ \sum_{s=t}^T \pi(x_s, u_s, s) + W(x_{T+1}) \mid x_t = x \right\}. \quad (12.1.2)$$

The fundamental result is that the optimal policy and value function are *memoryless*; that is, the value and optimal policy at time s depends only on x_s , and the value function satisfies the Bellman equation

$$V(x, t) = \sup_{u \in D(x, t)} \pi(x, u, t) + E\{V(x_{t+1}, t+1) \mid x_t = x, u_t = u\}. \quad (12.1.3)$$

with terminal condition

$$V(x, T+1) = W(x). \quad (12.1.4)$$

The value function always exists by backward induction: the terminal valuation is given by (12.1.4), and for any $V(x, t)$, $V(x, t-1)$ exists because the supremum in the right-hand side of (12.1.3) exists as long as $D(x, t)$ is nonempty.

The value function is the least upper bound on the possible payoff. We also want to know how to achieve this value. To that end, we attempt to compute the *optimal policy function*, $U(x, t)$, which is an optimal choice of control in state x at time t ; if it exists, it solves

$$U(x, t) \in \arg \max_{u \in D(x, t)} \pi(x, u, t) + E\{V(x_{t+1}, t+1) \mid x_t = x, u_t = u\}.$$

The existence of the optimal policy function in state x at time t depends on the existence of a control which achieves the supremum in (12.1.3). Even when there is no such control, we can construct nearly optimal controls. This is because for any ε there is a policy function that comes within ε of achieving the value function; this fact follows from the definition of supremum.

The concept of value function in (12.1.2) is a bit limited. Note that the value function defined in (12.1.2) is the total utility of following the optimal policy. Sometimes we will want to examine nonoptimal policies and their values. In general, if we are at the state (x, t) and we are going to use the policy U , the resulting expected total utility equals $V^U(x, t)$, which is defined recursively by

$$V^U(x, t) = \pi(x, U(x, t), t) + E\{V^U(x_{t+1}, t+1) | x_t = x, u_t = U(x, t)\} \quad (12.1.5)$$

Equation (12.1.5) is the general equation defining the value of a policy U and is linear in the unknown function V^U . The usual value function defined in (12.1.2) is the value of the optimal policy.

The notion of state variable is flexible since there are often many variables in a problem which can serve as the state variable. The only requirement is that the state vector must be a sufficient description of the problem at some moment during a period. In particular, it is sometimes useful to focus on an end-of-period state variable instead of the beginning-of-period specification. Suppose that y is a beginning-of-period state variable, u a control that is implemented in the middle of the period, $\pi(y, u, t)$ the payoff, and $x \equiv g(y, u)$ an end-of-period state variable. One could let (y, u) be the end-of-period state variable; fortunately there is often a more parsimonious end-of-period state variable. Let $F(\cdot; x_t, t)$ be the probability measure over y_{t+1} conditional on x_t . The end-of-period value function $V(x, t)$ is

$$V(x, t) = E \left\{ \sup_{u \in \mathcal{U}(y_{t+1}, t+1)} \sum_{s=t+1}^T \pi(y_s, u_s, s) + W(y_{T+1}) | x_t = x \right\}, \quad (12.1.6)$$

and it satisfies the Bellman equation

$$V(x, t) = E \left\{ \sup_{u \in D(y_{t+1}, t+1)} \pi(y_{t+1}, u, t+1) + V(x_{t+1}, t+1) | x_t = x \right\}, \quad (12.1.7)$$

together with the terminal condition $V(x, T) = E\{W(y_{T+1}) | x_T = x\}$.

From a theoretical perspective there is no difference between a beginning-of-period state variable formulation of the value function and an alternative end-of-period formulation. However, there may be substantial computational differences that argue for one formulation over the other. We will see this below in the commodity storage problem.

Infinite-Horizon Problems

We frequently use infinite-horizon, discounted, time-separable, autonomous dynamic programming problems to analyze long-run economic problems. The calendar time t does not enter into the problem directly; therefore the payoff function is $\pi(x, u)$, and $D(x)$ is the nonempty set of feasible controls for the current state x . The state in the “next” period will be denoted x^+ , and it depends on the current state and action in a possibly stochastic, but autonomous, fashion: Let $F(A; x, u)$ be the time-independent probability that $x^+ \in A$ if the current state is x and the current control is u .

For time-separable, autonomous problems the objective of the controller is to maximize discounted expected returns $E\{\sum_{t=0}^{\infty} \beta^t \pi(x_t, u_t)\}$ given a fixed initial value, x_0 , and $\beta < 1$, the discount factor. The value function is defined by

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

where $\mathcal{U}(x)$ is the set of all feasible strategies starting at x . The value function satisfies the Bellman equation

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

and the policy function, $U(x)$, solves

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

The key theorem in infinite horizon dynamic programming is the contraction mapping theorem applied to the Bellman equation. We first need two definitions.

DEFINITION A map $T: Y \rightarrow Z$ on ordered spaces Y and Z is *monotone* if and only if $y_1 \geq y_2$ implies $Ty_1 \geq Ty_2$.

DEFINITION A map $T: Y \rightarrow Y$ on a metric space Y is a *contraction with modulus $\beta < 1$* if and only if $\|Ty_1 - Ty_2\| \leq \beta \|y_1 - y_2\|$.

The critical theorem, due to Denardo (1967), is the contraction mapping theorem for dynamic programming.

THEOREM 12.1.1 If X is compact, $\beta < 1$, and π is bounded above and below, then the map

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

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

The boundedness and compactness assumptions in theorem 12.1.1 may appear limiting but not for us, since numerically feasible dynamic programming problems can generally examine only compact state spaces. The task we address here is the computation of $V(x)$ and $U(x)$.

The formulation in (12.1.9) assumes that x is the beginning-of-period state. Infinite-horizon problems can also be formulated in terms of an end-of-period state. If y is the beginning-of-period state, $\pi(y, u)$ the payoff function, $x = g(y, u)$ the end-of-period state, and y^+ is the next period's beginning-of-period state then the end-of-period value function, $V(x)$, is defined to be

$$V(x) = E \left\{ \sup_{\mathcal{U}(y_1)} \sum_{s=1}^{\infty} \beta^s \pi(y_s, u_s) \mid x_0 = x \right\},$$

and Bellman's equation is

$$V(x) = \beta E \left\{ \sup_{u \in D(y)} \pi(y^+, u) + V(g(y^+, u)) \mid x \right\}. \quad (12.1.11)$$

Theorem 12.1.1 applies also to problems of this kind. We now present a few examples of dynamic programming applied to familiar problems.

Wealth Accumulation Examples

The most common examples of dynamic programming models are simple wealth accumulation models. We describe them here and use them in examples.

The simplest wealth accumulation model is the deterministic growth model. The rate of consumption is denoted by c , and $f(k)$ is the net-of-depreciation output when capital stock is k , implying that capital evolves according to $k_{t+1} = k_t + f(k_t) - c_t$. We let $F(k)$ denote $f(k) + k$. We assume the time-separable utility function $\sum_{t=0}^{\infty} \beta^t u(c_t)$ where $\beta < 1$ is the discount factor. The discrete-time optimal growth problem becomes

$$\max_{c_t} \sum_{t=0}^{\infty} \beta^t u(c_t) \quad (12.1.12)$$

$$\text{s.t. } k_{t+1} = F(k_t) - c_t,$$

where k_0 is given. If c_t and k_t are consumption and capital at time t , the first-order conditions for the optimal path are $u'(c_t) = \beta u'(c_{t+1}) F'(k_{t+1})$ for all t . The dynamic programming version of (12.1.12) is

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

The optimal policy function, also called the control law, $C(k)$, obeys the first-order conditions

$$0 = u'(C(k)) - \beta V'(F(k) - C(k)). \quad (12.1.14)$$

The envelope theorem applied to (12.1.13) implies that $V'(k) = \beta V'(F(k) - c)F'(k)$, which together with (12.1.14) implies that

$$0 = u'(C(k))F'(k) - V'(k). \quad (12.1.15)$$

Therefore the solution to (12.1.13) is a policy function $C(k)$ and a value function $V(k)$ that satisfy (12.1.15) and

$$V(k) = u(C(k)) + \beta V(F(k) - C(k)). \quad (12.1.16)$$

Note that (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) with a first-order condition for optimality (12.1.15).

When labor, l , is supplied elastically, the net production function is $f(k, l)$, $F(k, l) = k + f(k, l)$, and the utility function becomes $u(c, l)$. The discrete-time dynamic programming problem becomes

$$V(k) = \max_{c, l} u(c, l) + \beta V(F(k, l) - c). \quad (12.1.17)$$

Combining the first-order conditions and the envelope theorem, we find that the control laws $C(k)$ and $L(k)$ satisfy

$$\begin{aligned} 0 &= u_c(C(k), L(k))F_k(k, L(k)) - V'(k), \\ 0 &= u_l(C(k), L(k)) + F_l(k, L(k))u_c(C(k), L(k)). \end{aligned} \quad (12.1.18)$$

The solution to (12.1.17) are functions $C(k)$ and $V(k)$ that satisfy (12.1.18) and

$$V(k) = u(C(k, L(k))) + \beta V(F(k, L(k)) - C(k)). \quad (12.1.19)$$

While this example is motivated by an optimal growth model, that is only one interpretation. For example, (12.1.17) can also be interpreted as the dynamic programming problem of a monopolistic firm. In the monopolistic firm application, c is dividends, $u(c, l) = c$, l is the amount of labor hired, k is both the capital stock and total assets, and $F(k, l)$ is the profit flow if the monopolist has k units capital and hires l units of labor. We can also interpret (12.1.17) as a dynamic problem of an individual, where u is the utility function, k is individual wealth, and $F(k, l)$ is asset plus labor income. It would be most natural to use the finite horizon version in this case, make F and u depend on age, and require $k \geq 0$ at death. The problem in (12.1.17) is quite general. There is an endogenous state variable, k , and controls, c

and l , that enter into the payoff and the law of motion. Our interpretations are not as general as possible, since the payoff function does not depend on the state. However, the use of growth model terms and notation does not imply that we are limited to studying those problems since we will not utilize any of their special features.

Stochastic Accumulation Problems

We next consider a stochastic model of accumulation. Let θ denote the current productivity level and $f(k, l, \theta)$ denote net income; define $F(k, l, \theta) = k + f(k, l, \theta)$, and assume that θ follows $\theta_{t+1} = g(\theta_t, \varepsilon_t)$ where the ε_t are i.i.d. disturbances. The infinite-horizon discrete-time optimization problem becomes

$$\begin{aligned} V(k, \theta) &= \max_{c_t, l_t} E \left\{ \sum_{t=0}^{\infty} \beta^t u(c_t, l_t) \right\} \\ \text{s.t. } k_{t+1} &= F(k_t, l_t, \theta_t) - c_t, \\ \theta_{t+1} &= g(\theta_t, \varepsilon_t), \\ k_0 &= k, \theta_0 = \theta. \end{aligned} \tag{12.1.20}$$

The Euler equation for (12.1.20) is

$$u_c(c_t, l_t) = \beta E \{ u_c(c_{t+1}, l_{t+1}) F_k(k_{t+1}, l_{t+1}, \theta_{t+1}) | k_t, \theta_t \}.$$

In (12.1.20), k is no longer a sufficient description of the problem because the value of today's θ provides us information about tomorrow's value of θ . Therefore both the beginning-of-period capital stock and the current value of θ are needed for a sufficient description of the state. The dynamic programming formulation is

$$V(k, \theta) = \max_{c, l} u(c, l) + \beta E \{ V(F(k, l, \theta) - c, \theta^+) | \theta \}, \tag{12.1.21}$$

where θ^+ is next period's θ realization. The control laws $c = C(k, \theta)$ and $l = L(k, \theta)$ satisfy the first-order conditions

$$\begin{aligned} 0 &= u_c(C(k, \theta), L(k, \theta)) F_k(k, L(k, \theta), \theta) - V_k(k, \theta), \\ 0 &= u_l(C(k, \theta), L(k, \theta)) + F_l(k, \theta) u_c(C(k, \theta), L(k, \theta)). \end{aligned} \tag{12.1.22}$$

The Euler equation implies that the control laws also satisfy

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

where

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

$$l^+ \equiv L(k^+, \theta^+),$$

represent the next period's capital stock and labor supply.

Again this example has many economic interpretations. In the life-cycle interpretation, θ is a state variable that may affect either asset income, labor income, or both. In the monopolist interpretation, θ may reflect shocks to costs, demand, or both.

Stochastic Transition Problems

In the preceding examples the current control fixes the next period's state exactly. For example, by an appropriate choice of consumption, one can fix the end-of-period capital stock in (12.1.20). In such cases we get a simple Euler equation for the policy function. We will see that such Euler equations allow us to compute the policy function directly without any reference to the value function. When the controls can't fix the state exactly, we need the value function. We present one simple example of that here.

Suppose that a firm has profits $\pi(x)$ that depend on productivity x , where productivity can have several possible values, x_i , $i = 1, \dots, n$. Suppose that an investment expenditure of I will control the transition probability of the productivity state according to $\text{Prob}\{x^+ = x_i | x = x_j\} = q(I, i, j)$ where $0 \leq q \leq 1$ and $\sum_i q(I, i, j) = 1$ for all I and j . In productivity state x gross profits will be $\pi(x)$. The resulting dynamic programming problem will be

$$V(x_j) = \max_I \pi(x_j) - I + \beta \sum_i q(I, i, j) V(x_i), \quad (12.1.24)$$

which implies the first-order condition

$$0 = -1 + \beta \sum_i q_I(I, i, j) V(x_i), \quad (12.1.25)$$

Note that in this problem the value function is an essential component of the first-order condition, a feature absent in (12.1.23). This is an important difference for computational purposes.

12.2 Continuous-Time Dynamic Programming Problems

We can also formulate dynamic programming problems in continuous time. Here we must deal specifically with the stochastic structure. Most problems in econo-

mics assume a Brownian motion or jump process, or a combination, to model the stochastic elements. We will consider only Brownian motion examples here; see Malliaris and Brock (1982) for a more complete discussion of stochastic control.

The canonical deterministic discounted optimal control problem is

$$\begin{aligned} V(x_0, t) &= \max_u \int_t^T e^{-\rho(s-t)} \pi(x, u, s) ds + W(x(T), T) \\ \text{s.t. } & \dot{x} = f(x, u, s), \\ & x(t) = x_0, \end{aligned} \tag{12.2.1}$$

where $x \in R^n$ and $u \in R^m$. The Bellman equation for this problem is

$$\rho V(x, t) - V_t(x, t) = \max_u \pi(x, u, t) + \sum_{i=1}^n V_{x_i}(x, t) f^i(x, u, t), \tag{12.2.2}$$

with the terminal condition

$$V(x, T) = W(x, T). \tag{12.2.3}$$

If π depends only on x and u , and $T = \infty$, we have the autonomous case, wherein $V(x, t) = V(x)$ and the Bellman equation

$$\rho V(x) = \max_u \pi(x, u) + \sum_{i=1}^n V_{x_i}(x) f^i(x, u) \tag{12.2.4}$$

characterizes the current value function.

Wealth Accumulation Examples

The continuous-time optimal accumulation problem is

$$\max_c \int_0^\infty e^{-\rho t} u(c) dt \tag{12.2.5}$$

$$\text{s.t. } \dot{k} = f(k) - c,$$

which was studied in chapter 10. The dynamic programming formulation of (12.2.5) is

$$0 = \max_c u(c) + V'(k)(f(k) - c) - \rho V(k), \tag{12.2.6}$$

and the control law $c = C(k)$ obeys the first-order condition

$$0 = u'(C(k)) - V'(k). \quad (12.2.7)$$

The continuous-time dynamic programming problem with elastic labor supply becomes

$$0 = \max_{c, l} u(c, l) + V'(k)(f(k, l) - c) - \rho V(k), \quad (12.2.8)$$

and the control laws $C(k)$ and $L(k)$ satisfy the first-order conditions

$$\begin{aligned} 0 &= u_c(C(k), L(k)) - V'(k), \\ 0 &= u_l(C(k), L(k)) + f_l(k, l)V'(k). \end{aligned} \quad (12.2.9)$$

The solution to (12.2.8) are functions $C(k)$, $L(k)$, and $V(k)$ that satisfy (12.2.9) and

$$0 = u(C(k), L(k)) + V'(k)(f(k, L(k)) - C(k)) - \rho V(k). \quad (12.2.10)$$

Stochastic Dynamic Programming

We next examine the continuous-time case with white noise disturbances. Suppose that $x \in R^n$, $u \in R^m$ are the state and controls, $u \in D(x, t)$ describes the control constraint, and the law of motion is

$$dx = f(x, u, t) dt + \sigma(x, u, t) dz, \quad (12.2.11)$$

where $f(x, u, t) \in R^n$ is the instantaneous drift, dz is “white noise,” and $\sigma(x, u, t)$ is the $n \times n$ matrix function representing the instantaneous standard deviation of the x process at (x_t, u_t) ; we let $\sigma^2(x, u, t)$ denote the corresponding variance-covariance matrix. The value function is defined

$$V(x, t) = \max_{u \in \mathcal{U}(x, t)} E_t \left\{ \int_t^T e^{-\rho(s-t)} \pi(x, u, s) ds + W(x_T) \right\}, \quad (12.2.12)$$

where $\mathcal{U}(x, t)$ is the set of feasible feedback controls if the state is x at time t . If $x \in R$, then the Hamilton-Jacobi-Bellman (HJB) equation for V is

$$\rho V(x, t) - V_t(x, t) = \max_{u \in D(x, t)} \pi(x, u, t) + V_x(x, t)f(x, u, t) + \frac{1}{2} V_{xx}(x, t)\sigma^2(x, u, t), \quad (12.2.13)$$

with the terminal condition

$$V(x, T) = W(x). \quad (12.2.14)$$

If $x \in R^n$, then the HJB equation is

$$\begin{aligned} \rho V(x, t) - V_t(x, t) &= \max_{u \in D(x, t)} \pi(x, u, t) + \sum_{i=1}^n V_{x_i}(x, t) f^i(x, u, t) \\ &\quad + \frac{1}{2} \text{Tr}[\sigma(x, u, t) \sigma(x, u, t)^T V_{xx}(x, t)], \end{aligned} \quad (12.2.15)$$

where $\text{Tr}[A]$ is the trace of the matrix A . In the autonomous infinite-horizon case, tastes and technology do not depend on t ; hence the value function depends solely on x and (12.2.13) reduces to

$$\rho V(x) = \max_{u \in D(x, t)} \pi(x, u) + V'(x) f(x, u) + \frac{1}{2} \sigma^2(x, u) V''(x) \quad (12.2.16)$$

and (12.2.15) reduces to

$$\rho V(x) = \max_{u \in D(x, t)} \pi(x, u) + \sum_{i=1}^n V_{x_i}(x) f^i(x, u) + \frac{1}{2} \text{Tr}[\sigma(x, u) \sigma(x, u)^T V_{xx}(x)] \quad (12.2.17)$$

12.3 Finite-State Methods

The simplest dynamic programming problems have a finite number of states. We discuss these problems for three reasons. First, some problems are naturally discrete. Second, discrete approximations are often used in the economics literature. Third, discrete problems allow us to clearly illustrate important aspects of the common numerical dynamic programming methods.

Finite-Horizon Problems

Let X be the set x_i , $i = 1, \dots, n$, of n states, and $\mathcal{D} = \{u_i \mid i = 1, \dots, m\}$ the set of m controls for a finite-state problem. Each control induces a Markov transition rule for the states, which can be summarized by the probabilistic law $q_{ij}^t(u)$; that is, for each control u , $q_{ij}^t(u)$ is the probability of a transition to state x_j if the state is x_i and control is u in period t . The collection of probabilities specifies a Markov transition matrix at time t for each u , which we will denote $Q^t(u)$. For control u and state x_i , row i in $Q^t(u)$ is the probability distribution of the state at time $t+1$ if the time t state is x_i and the controller applies control u .

This formulation is general, including a wide variety of problems. For example, deterministic transition laws can be modeled by permitting only zero and unit entries in the transition matrix, $Q^t(u)$, for each control u . State-dependent constraints on controls can be modeled by creating an extra absorbing state with a constant payoff of $-\infty$ and making the process move to that state whenever an inappropriate

control is chosen. Hence, when we deal with finite-state, finite-control problems, we will drop the control constraints, assuming that any such constraints are expressed in the payoff function.

The analysis of a dynamic programming problem generally concentrates on the value function. In the finite-horizon case we inductively use (12.1.3) to compute the value function at each time t , beginning with the terminal value function, $V(x, T + 1) = W(x)$. Since we have just a finite number of states, the value function $V(x, t)$ at time t is really just a finite list of values. If we define $V_i^t \equiv V(x_i, t)$, the Bellman (12.1.3) equation becomes

$$V_i^t = \max_u \left[\pi(x_i, u, t) + \beta \sum_{j=1}^n q_{ij}^t(u) V_j^{t+1} \right], \quad i = 1, \dots, n, \quad (12.3.1)$$

with the terminal condition

$$V_i^{T+1} = W(x_i), \quad i = 1, \dots, n. \quad (12.3.2)$$

Therefore finite-horizon dynamic programming reduces to (12.3.2) combined with the induction step (12.3.1), where each induction step in (12.3.1) consists of several optimization problems. At this point it is clear that the most demanding task is solving the maximization problems in (12.3.1). If there is no useful structure and there are only a finite number of controls, the only way to solve each of the maximization problems in (12.3.1) is to check out each possible control at each state. In any case each step in (12.3.1) is finitistic, and this procedure will terminate in a finite amount of time.

If u is a continuous variable, then the maximization problems in (12.3.1) must be solved using an appropriate optimization method. Define

$$h(u; x_i, t) \equiv \pi(x_i, u, t) + \beta \sum_{j=1}^n q_{ij}^t(u) V_j^{t+1} \quad (12.3.3)$$

to be the specific optimization problem at state x_i in period t with period $t + 1$ value function V^{t+1} . The proper choice of optimization algorithm depends on how h depends on u . If $h(u; x_i, t)$ is a smooth function of u for each x_i and t , then we may use Newton's method or some variant; however, if h is not smooth in u , then we may need to use a comparison method such as the polytope method. If $h(u; x_i, t)$ is multimodal, then we must use global optimization methods. It is this maximization step that will often consume most of the computing time and will be a focus of efforts below to create efficient algorithms.

Infinite-Horizon Problems

In infinite-horizon problems with a finite number of states, the infinite-horizon Bellman equation (12.1.9) still imposes definitive conditions on the value function. Since we have just a finite number of states, the value function is really just a finite list of values, a value vector. If we define $V_i \equiv V(x_i)$ to be the vector of values then the Bellman equation (12.1.9) becomes a system of nonlinear equations:

$$V_i = \max_u \left[\pi(x_i, u) + \beta \sum_{j=1}^n q_{ij}(u) V_j \right], \quad i = 1, \dots, n. \quad (12.3.4)$$

Let V^* denote the solution to (12.3.4).

By the contraction theorem, this equation does fix a unique solution, V^* . Unfortunately, it is a nonlinear equation because of the maximization step. However, we can decompose the problem into a linear problem and a sequence of optimization problems. First we note that we can easily compute the value of any specific policy. Suppose that in state x_i , the control is $U_i \in \mathcal{D}$, $i = 1, \dots, n$; then the vector $U \equiv (U_1, \dots, U_n)$ is the policy function and the return in state i is $P_i^U \equiv \pi(x_i, U_i)$. The induced transition rule for the states, Q^U , is defined by $Q_{ij}^U = q_{ij}(U_i)$; that is, the row i column j element of Q^U is $q_{ij}(u)$ if $U_i = u$. In terms of P^U and Q^U , the equation defining the value of following the policy U forever reduces to the vector equation $V^U = P^U + \beta Q^U V^U$ which has the solution

$$V^U = (I - \beta Q^U)^{-1} P^U. \quad (12.3.5)$$

Since Q^U is a probability matrix, $I - \beta Q^U$ is invertible, and V^U is well-defined in (12.3.5). Therefore, to compute the value of the control law U , one constructs the induced Markov chain, Q^U , and the induced state-contingent payoff vector, P^U , and computes (12.3.5), a linear algebra calculation.

If V^* is the value function for the problem, then it is the value of following the optimal policy, U^* . The optimal policy also satisfies the optimality condition

$$U_i^* \in \arg \max_u \left[\pi(x_i, u) + \beta \sum_j q_{ij}(u) V_j \right], \quad i = 1, \dots, n. \quad (12.3.6)$$

The optimal control policy and the value function for the problem are the U and V that jointly satisfy (12.3.5) and (12.3.6). If there are only a finite number of possible control policies, one could find V by computing the value of all possible policies. However, that is an impractical approach, and usually avoidable.

Before considering various schemes for calculating V and U , we will define two useful maps. The map T takes next period's value function, V^+ , creates the current value function, V , and is defined by

$$V_i = \max_u \left[\pi(x_i, u) + \beta \sum_{j=1}^n q_{ij}(u) V_j^+ \right] \equiv (TV^+)_i, \quad i = 1, \dots, n, \quad (12.3.7)$$

We also define a selection for today's control given next period's value

$$U_i \in \arg \max_u \left[\pi(x_i, u) + \beta \sum_{j=1}^n q_{ij}(u) V_j^+ \right] \equiv (\mathcal{U}V^+)_i, \quad i = 1, \dots, n. \quad (12.3.8)$$

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.¹ With the mappings T and \mathcal{U} we can describe a variety of computation methods. We next turn to two iterative schemes for solving (12.3.4).

Value Function Iteration

The simplest numerical procedure for finding V^* , *value function iteration*, is motivated by the contraction properties of the Bellman equation. Value function iteration computes the sequence

$$V^{l+1} = TV^l, \quad l = 0, 1, 2, \dots \quad (12.3.9)$$

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

The iterative scheme (12.3.9) will converge to the solution V^∞ only asymptotically. In practice, one iterates until the successive V^l change little, and the successive U^l do not change. When one believes that the U^l series has converged, one should compute the final approximation of V^* , $V^{U^l} = (I - \beta Q^{U^l})^{-1} P^{U^l}$. This will ensure that the computed value function is the value corresponding to the computed policy, and it is often a faster way to compute V^* than continued iteration of the value function equation. If V^{U^l} satisfies $V = TV$, then one is done. Otherwise, the iteration (12.3.9) should continue from V^{U^l} . Since the contraction theorem applies and there

1. Since $\mathcal{U}V$ is uniquely defined for generic V and for many common economic problems, we will not explicitly consider the multiplicity problem. Furthermore, if there are multiple optimal policies, they lead to the same value function and the planner is indifferent among them.

are only a finite number of possible control laws, this process will converge in finite time.

It is also advisable to choose a V^0 which is close to and similar to the final solution. For example, if one knows some bounds on V^* , then V^0 should also satisfy those bounds. Also, if V_i is increasing in i , then a good choice of V_i^0 would also be increasing in i . It is well worth the effort to think about the initial guess; the better the initial guess, the fewer iterations needed for convergence. In summary, the value function algorithm is presented in algorithm 12.1.

Algorithm 12.1 Value Function Iteration Algorithm

[Value function iteration](#)

Objective: Solve the Bellman equation, (12.3.4).

Initialization. Make initial guess V^0 ; choose stopping criterion $\varepsilon > 0$.

Step 1. For $i = 1, \dots, n$, compute

$$V_i^{l+1} = \max_{u \in D} \pi(x_i, u) + \beta \sum_{j=1}^n q_{ij}(u) V_j^l.$$

Step 2. If $\|V^{l+1} - V^l\| < \varepsilon$, then go to step 3; else go to step 1.

Step 3. Compute the final solution, setting

$$U^* = \mathcal{U}V^{l+1},$$

$$P_i^* = \pi(x_i, U_i^*), \quad i = 1, \dots, n,$$

$$V^* = (I - \beta Q^{U^*})^{-1} P^*,$$

and STOP.

Error Bounds

Once we compute an approximate solution to a dynamic programming problem, we would like to have some idea as to the possible error. The special structure of the Bellman equation implies that we can compute error bounds on (12.3.9). The contraction property implies that the iterates of value function iteration satisfy the inequality

$$\|V^* - V^k\|_\infty \leq \frac{1}{1-\beta} \|V^{k+1} - V^k\|_\infty. \quad (12.3.10)$$

Therefore, if we want to be sure that we have a value function V^k which is within ε^V of the solution V^* , we stop the value function iteration process at the first iterate V^k such that

$$\|V^{k+1} - V^k\|_\infty \leq \varepsilon^V(1 - \beta). \quad (12.3.11)$$

Inequality (12.3.11) becomes the convergence rule given our ε^V goal, implying that we set $\varepsilon = \varepsilon^V(1 - \beta)$ in the initialization step of algorithm 12.1.

This is one of the few cases where our convergence criterion also gives us an upper bound on how far we are from the true solution. Note that this bound applies only to the value function, not the policy function. In some cases the inequality (12.3.11) also gives us information about the error in the policy function, but that requires problem-specific analysis.

Exploiting Concavity and Monotonicity

The maximization step in value iteration is a costly step. Anything that can economize on this step will help convergence. In some cases we have a priori information that can be exploited. We here explore two such examples.

For example, if F and u in (12.1.12) are concave then $V(k)$ is also concave. In such cases we should begin with a concave initial guess, since then each value iteration will also be concave. We consider methods to exploit this in the general case. Define

$$h(u; x) \equiv \pi(x, u) + \beta E\{V(x^+) | x, u\}$$

to be the objective function in the maximization step at state x , and suppose that we know that V is concave in x and $h(u; x)$ is concave in u . Suppose that D is finite; then each maximization problem in the maximization step would normally be solved by trying all possible $u \in D$. However, if the objective, $h(u; x)$ is concave in u , then we need not try all such u . Instead, we need only compute $h(u_1; x)$, $h(u_2; x)$, $h(u_3; x), \dots$, for an increasing sequence of u 's until the $h(u_i; x)$ values begin to decrease. As soon as $h(u_{j+1}; x) < h(u_j; x)$, we know that u_j is the optimal solution since $h(u, x)$ is concave in u . We will call this the *concave stopping rule*, since it gives us a useful stopping rule for concave problems. If u is a continuous control, then Newton's method is likely to be a good choice for problems where h is concave and smooth in u .

In some problems, such as concave versions of (12.1.13), theory tells us that the optimal control is monotone increasing in the state and that we only need to examine monotone increasing U ; we can also exploit this to economize on comparisons in step 1 of value function iteration. If u_j maximizes $h(u; x_i)$, then when we analyze $x_l > x_i$, we know that $u_j < U_l$. The monotonicity of U then tells us we only need to consider $h(u_j; x_l)$, $h(u_{j+1}; x_l), \dots$. This is an example of the solution of one optimization problem being a good initial guess to another optimization problem. Clearly the order in which we solve the various $\max_u h(u; x)$ problems is important in exploiting these properties.

This discussion of concavity and monotonicity makes the general point that anything that helps us solve the maximization step will help the value iteration algorithm. When solving dynamic programming problems, there often is special structure that can be exploited to reduce running time.

Adaptive Control Choices and Optimization Rules

We next discuss two more ideas that can help for infinite-horizon problems. The basic idea is that since initial iterates of the value iteration algorithm are only interim approximations, they need not be computed with great accuracy. Only as we approach the final solution do we really need to make the effort to exactly solve the maximization problem.

One way to implement this idea is to initially examine only a subset of the controls in D ; that is, solve the optimization problem for $u \in D' \subset D$ in the early iterations and then solve it for all $u \in D$ in later iterations. Another idea is to initially solve the problem for a subset of the states, and solve the problem for all states only in the final iterations.

The value function iteration method implicitly assumes that the maximization step (12.3.7) can be solved exactly. If \mathcal{D} is finite, then (12.3.7) can be solved by checking all possibilities, but if \mathcal{D} is infinite (which usually means continuous), then one must use an appropriate optimization method. We are implicitly assuming that (12.3.7) can be solved. Since it is the only nontrivial calculation in (12.3.9), it is critical that (12.3.7) be solved efficiently.

We should also note that these ideas to economize on the optimization step also work if u is a continuous variable. In particular, use a loose convergence criterion for the maximization step in the early iterations when accuracy is not as important, and tighten the convergence criterion in later iterations. Also, after solving the maximization problem for state x_i , one can use that solution for u along with any gradient and Hessian information to produce a hot start for solving the maximization step for state x_{i+1} . Of course, in order to make this effective, the states should be ordered so that x_i is close to x_{i+1} for most i .

12.4 Acceleration Methods for Infinite-Horizon Problems

Value function iteration is a slow procedure for infinite-horizon problems since V^l converges only linearly at the rate β , and at each iteration we solve as many optimization problems as there are states. In the finite-horizon case, V^0 in the initialization step of value function iteration is just the terminal value function, and the later

iterates produce the value and policy functions for each period before the terminal time. In finite-horizon problems, all of this is a necessary part of the final solution, and we can do no better than value function iteration. However, in infinite-horizon problems, we want only the stationary optimal policy and value function. Since the intermediate iterates computed by value function iteration are of no intrinsic value, we would like to economize on their computation. In this section we consider acceleration methods that solve infinite-horizon problems much faster than value function iteration.

Policy Function Iteration

The first acceleration method is the *policy function iteration* method. Policy iteration economizes by making more use of each new computed policy function. More specifically, each time a new policy function is computed, policy function iteration computes the value function which would occur if the new policy function were used forever; in contrast, value function iteration assumes that the new policy is used only for one period. Then, with this new value function, we compute a new optimal policy function and continue the iteration. Using our definitions of T and \mathcal{U} , the iteration scheme is expressed in algorithm 12.2.

Algorithm 12.2 Policy Function Algorithm

Objective: Solve the Bellman equation, (12.3.4).

Initialization. Choose stopping criterion $\varepsilon > 0$. EITHER make initial guess, V^0 , for the value function and go to step 1, OR make initial guess, U^1 , for the policy function and go to step 2.

$$\text{Step 1. } U^{l+1} = \mathcal{U} V^l$$

$$\text{Step 2. } P_i^{l+1} = \pi(x_i, U_i^{l+1}), \quad i = 1, \dots, n$$

$$\text{Step 3. } V^{l+1} = (I - \beta Q^{U^{l+1}})^{-1} P^{l+1}$$

Step 4. If $\|V^{l+1} - V^l\| < \varepsilon$, STOP; else go to step 1.

Note that in the initialization step we are given a choice: We may begin either with an initial guess for the policy or value function. Sometimes we may have a good guess for V but not U , but other times we may have a good guess for U but not V . Having a good initial guess is important; we choose between these alternatives depending on the relative quality of our initial guesses for V and U .

Policy iteration often takes only a few iterations. Unfortunately, the computation of $(I - \beta Q^{U^{l+1}})^{-1}$ may be expensive, making each iteration costly. A second acceleration procedure, *modified policy iteration with k steps*, implements the basic idea of

policy iteration without computing $(I - \beta Q^{U^{l+1}})^{-1}$. Modified policy iteration replaces step 3 of policy function iteration with

$$V^{l+1} = \sum_{t=0}^k \beta^t (Q^{U^{l+1}})^t P^{l+1} + \beta^{k+1} (Q^{U^{l+1}})^{k+1} V^l. \quad (12.4.1)$$

The expression in (12.4.1) is actually implemented by the steps

$$\begin{aligned} W^0 &= V^l, \\ W^{j+1} &= P^{l+1} + \beta Q^{U^{l+1}} W^j, \quad j = 0, \dots, k, \\ V^{l+1} &= W^{k+1}. \end{aligned} \quad (12.4.2)$$

The modified policy iterate V^{l+1} defined in (12.4.2) is the value of following the policy U^{l+1} for $k+1$ periods after which the problem has value V^l . Policy iteration is the limiting case of modified policy iteration as k becomes infinite. The following theorem summarizes what we know about convergence rates of modified and unmodified policy iteration:

THEOREM 4.1 (Puterman and Shin) The successive iterates of modified policy iteration with k steps, (12.4.1), satisfy the error bound

$$\frac{\|V^* - V^{l+1}\|}{\|V^* - V^l\|} \leq \min \left[\beta, \frac{\beta(1 - \beta^k)}{1 - \beta} \|U^l - U^*\| + \beta^{k+1} \right]. \quad (12.4.3)$$

Theorem 4.1 points out that as the policy function gets close to U^* , the linear rate of convergence approaches β^{k+1} . Hence convergence accelerates as the iterates converge.

Gaussian Iteration Methods

The value and policy function iteration methods have strong dynamic intuitions. The value function iteration is exactly the limit of backward iteration. However, recall that the problem being solved,

$$V_i = \max_u \pi(x_i, u) + \beta \sum_{j=1}^n q_{ij}(u) V_j, \quad i = 1, \dots, n, \quad (12.4.4)$$

is just a nonlinear system of equations in the unknowns V_i . We next examine iteration methods that are motivated by standard nonlinear equation solution methods.

Value function iteration is also called *pre-Gauss-Jacobi* since its definition

$$V_i^{l+1} = \max_u \pi(x_i, u) + \beta \sum_{j=1}^n q_{ij}(u) V_j^l \quad (12.4.5)$$

is similar to the Gauss-Jacobi iteration examined in chapters 3 and 5. When we apply the full nonlinear Gauss-Jacobi method to (12.4.4), we have the *Gauss-Jacobi* iteration method

$$V_i^{l+1} = \max_u \frac{\pi(x_i, u) + \beta \sum_{j \neq i} q_{ij}(u) V_j^l}{1 - \beta q_{ii}(u)}. \quad (12.4.6)$$

The difference between (12.4.5) and (12.4.6) is the $1 - \beta q_{ii}(u)$ term, which is essentially 1 unless there is significant chance that the optimal control results in no change in state.

In both pre-Gauss-Jacobi and Gauss-Jacobi, we do not use the new state i value, V_i^{l+1} , until we have computed all components of V^{l+1} . In the *pre-Gauss-Seidel* method we use the new value immediately but in a value function iteration fashion:

$$V_i^{l+1} = \max_u \pi(x_i, u) + \beta \sum_{j < i} q_{ij}(u) V_j^{l+1} + \beta \sum_{j \geq i} q_{ij}(u) V_j^l, \quad i = 1, 2, \dots \quad (12.4.7)$$

This is the same as value function iteration except that we use the new approximation V_i^{l+1} immediately when computing V_j^{l+1} for $j > i$. The *Gauss-Seidel* method is

$$V_i^{l+1} = \max_u \frac{\pi(x_i, u) + \beta \sum_{j < i} q_{ij}(u) V_j^{l+1} + \beta \sum_{j > i} q_{ij}(u) V_j^l}{1 - \beta q_{ii}(u)}, \quad i = 1, 2, \dots \quad (12.4.8)$$

We would like to know that these methods converge. All of the procedures are obviously monotone in the sense that the solutions for the V_i^{l+1} in (12.4.7) and (12.4.8) are increasing in the V_i^l values. Therefore, if the initial guess for the value function is uniformly less than (uniformly greater than) $\min_{x,u} \pi(x, u)$ ($\max_{x,u} \pi(x, u)$), then the iterates of each of these methods will converge monotonically to the true value function. Since we have $\beta < 1$, the Gauss-Jacobi and Gauss-Seidel iterations are contraction mappings.

Upwind Gauss-Seidel

The methods discussed in the previous section accelerate convergence by exploiting ideas from nonlinear equation solving. The Gauss-Seidel methods in (12.4.7) and

(12.4.8) can be further refined since they depend on the ordering of the states. In this section we will describe Gauss-Seidel methods in which the ordering is endogenous, chosen to enhance convergence.

To illustrate the basic idea, we consider a trivial two-state model. Suppose that there are two states, x_1 and x_2 , and two controls, u_1 and u_2 , where control u_i causes the state to move to x_i , $i = 1, 2$, and

$$\begin{aligned}\pi(x_1, u_1) &= -1, \quad \pi(x_1, u_2) = 0, \\ \pi(x_2, u_1) &= 0, \quad \pi(x_2, u_2) = 1.\end{aligned}\tag{12.4.9}$$

We assume that $\beta = 0.9$. Since the only time when the payoff is positive is when the state is x_2 and the control is u_2 , and since u_2 keeps the state in state x_2 , it is clear that the optimal policy is to choose control u_2 in both states. This implies the solution

$$V(x_1) = 9, \quad V(x_2) = 10.$$

Under the optimal policy, x_2 is the unique stable steady state; that is, once we get there the optimal policy keeps us there, and if we are in any other state we use a control which takes us to the steady state.

We will now illustrate value iteration and policy function iteration for (12.4.9). If we use value iteration with $V^0(x_1) = V^0(x_2) = 0$ the first three iterates are

$$\begin{aligned}V^1(x_1) &= 0, \quad V^1(x_2) = 1, \quad U^1(x_1) = 2, \quad U^1(x_2) = 2, \\ V^2(x_1) &= 0.9, \quad V^2(x_2) = 1.9, \quad U^2(x_1) = 2, \quad U^2(x_2) = 2, \\ V^3(x_1) &= 1.71, \quad V^3(x_2) = 2.71, \quad U^3(x_1) = 2, \quad U^3(x_2) = 2,\end{aligned}$$

and further iterates will converge at the linear rate prescribed by theory. The first iteration produces the optimal policy function and the successive iterates are necessary only to get V right.

Policy iteration applied to (12.4.9) produces the sequence

$$\begin{aligned}V^1(x_1) &= 0, \quad V^1(x_2) = 1, \quad U^1(x_1) = 2, \quad U^1(x_2) = 2, \\ V^2(x_1) &= 9, \quad V^2(x_2) = 10, \quad U^2(x_1) = 2, \quad U^2(x_2) = 2,\end{aligned}$$

converging after two iterations. This rapid convergence arises because the first iterate produces the optimal policy function. Since the policy iteration method then immediately computes the value of that policy, we converge at the end of the second iterate. In this case we execute the maximization step in value function iteration only

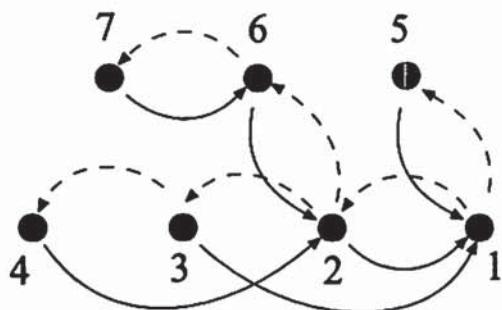


Figure 12.1
Downwind and upwind directions

twice and get the exact answer, whereas value function iteration requires much more computation to get answers that are only approximately accurate.

The key observation to make about this problem is that we have a very good idea about the value at x_2 , since it is an absorbing state under the optimal policy. The value function at absorbing states is trivial to compute: If s is an absorbing state when the optimal policy constantly applies control u , then $V(s) = \pi(s, u)/(1 - \beta)$. Once we know the value of an absorbing state we can easily compute the value of any state that sends the system to the absorbing state under the optimal policy. In our example, we know that x_2 is an absorbing state under the optimal policy; then $V(x_2) = 10$. When we examine state x_1 , we find that the optimum is to choose u_2 . Therefore the system will be in state x_1 for one period when control u_2 is applied, sending the system to state x_2 . This implies that $V(x_1) = \beta V(x_2) = 9$. We have therefore computed the infinite horizon value function *in just one pass* through the state space.

Figure 12.1 illustrates the general idea. Suppose we have a seven-state, infinite horizon dynamic programming problem, and suppose that a policy induces the motion indicated by the solid arrows. State 1 is the absorbing state for all initial conditions. The value of the policy can be easily computed at state 1, since it is the steady state. The value at states 2, 3, and 5 can be directly computed once we know the value at state 1, after which we can compute the value at states 4 and 6, and then finally state 7 can be valued. We see here that if the state flows in the directions of the solid arrows, the proper direction for us to construct the value function is in the opposite, “upwind,” direction.² Another way to express this is that information nat-

2. We use the term “upwind” because it expresses the critical role played by the flow of the state. Also the same idea is behind the “upwind differencing” finite-difference method often used in fluid mechanics problems.

urally flows in the upwind direction. For example, information about state 1 tells us much about states 2 and 5, but we can evaluate state 1 without knowing anything about states 2 and 5. Again, a single pass through the states will determine the value of the indicated policy.

While the speed achieved in these simple examples is unusual, it does suggest a more general approach. In general, once we ascertain the value of the problem at the stable steady states, we can determine the value at the remaining states. To accomplish this, the *upwind Gauss-Seidel (UGS)* algorithm makes a policy guess, constructs an upwind sequence of states, and then updates the value of those states in the reverse order.

Algorithm 12.3 Upwind Gauss-Seidel (UGS)

Objective: Solve the Bellman equation (12.3.4).

Initialization. Make initial guess for policy, U^0 , and set $V^0 = 0$; choose stopping criterion $\varepsilon > 0$.

Step 1. Reorder the states to produce a new indexing, x_1, x_2, \dots , such that $q_{i,i+1}(U_i^k) \geq q_{i+1,i}(U_{i+1}^k)$.

Step 2. Construct V^{k+1} : Set $W = V^k$. Then build W by iterating

$$W_i = \max_{u \in D} \pi(x_i, u) + \beta \sum_{j=1}^n q_{ij}(u) W_j, \quad i = n, n-1, \dots, 1,$$

where, at each i , we set U_i^{k+1} equal to a maximizing choice of u .

Step 3. Set $V^{k+1} = W$. If $\|V^k - V^{k+1}\| < \varepsilon$, STOP. Otherwise, go to step 1.

The key details are the reordering in step 1 and the construction of W in step 2. The reordering requirement in step 1 just requires that it is more likely that we move from state x_i to x_j , than vice versa, if $i > j$. In step 2 note that new values of W_i are used to compute the new values of W_j for $j < i$. It is the combination of the reordering of the states and the construction of W which implements the upwind idea. The contraction mapping theorem shows that V^k will converge to V^* no matter what order is chosen in step 1.

However, it may be difficult in step 1 of the UGS method to find an appropriate order for traversing the states. We next explore two ideas that attempt to deal with this problem. First, we describe the *alternating sweep* method. Formally, we first fix an ordering of the states, x_1, x_2 , etc. The alternating sweep Gauss-Seidel method then traverses the states first in increasing order and then in decreasing order. Step 1 in UGS would be eliminated, and step 2 in UGS would be replaced by

$$W = V^k,$$

$$W_i^+ = \max_u \pi(x_i, u) + \beta \sum_{j=1}^{i-1} q_{ij}(u) W_j^+ + \beta \sum_{j=i}^n q_{ij}(u) W_j, \quad i = 1, 2, 3, \dots,$$

$$W_i^{++} = \max_u \pi(x_i, u) + \beta \sum_{j=1}^i q_{ij}(u) W_j^+ + \beta \sum_{j=i+1}^n q_{ij}(u) W_j^{++}, \quad i = n, n-1, \dots,$$

$$V^{k+1} = W^{++}.$$

The alternating sweep method has the advantage of trying two opposite orders in alternation; at each state, one of the orders will, with any luck, be useful. In one-dimensional problems, this is likely but in multidimensional cases the effectiveness will depend critically on the indexing.

The key idea in upwind updating is to find a sequence of communicating states and traverse them in an upwind fashion. One way is to accomplish this is by simulation. In this approach, called *simulated upwind Gauss-Seidel*, we start with a state, x_{i_1} , then simulate the dynamic process of the dynamic programming problem under the current policy rule, U^l , to generate a sequence of states, $x_{i_2}, x_{i_3}, x_{i_4}$, etc., such that in the simulation the process moves from x_{i_j} to $x_{i_{j+1}}$. We simulate the process for T periods. We take this simulation as evidence that $q_{i_1 i_2}(U_{i_1}^l)$ exceeds $q_{i_2 i_1}(U_{i_2}^l)$, $q_{i_2 i_3}(U_{i_2}^l)$ exceeds $q_{i_3 i_2}(U_{i_3}^l)$, etc., the desired feature of step 1 of UGS. Then we apply Gauss-Seidel updating to state x_{i_T} , next to state $x_{i_{T-1}}$, then $x_{i_{T-2}}$, etc., updating x_{i_1} last; that is, we update the states in the reverse order. The resulting procedure is displayed in algorithm 12.4.

In our simple example, (12.4.9), SUGS does much better than value function iteration. After an iteration of value function iteration, we arrive at the optimal policy, $U^1(x_1) = 2$, $U^1(x_2) = 2$. If we begin our simulation with x_1 and simulate a five-period process, we arrive at the sequence $\{x_1, x_2, x_2, x_2, x_2, x_2\}$. We then apply value function iteration to x_2 five times, followed by one value function iteration applied to x_1 . This produces a second iterate of

$$V^2(x_1) = 4.68, \quad V^2(x_2) = 4.21$$

which is exactly what the sixth iterate of value function iteration would achieve but only after applying value function at x_2 five times instead of once. This clearly shows how computing the downwind states first may result in substantial savings in computing the upwind states.

Algorithm 12.4 Simulated Upwind Gauss-Seidel Algorithm (SUGS)

Objective: Solve the Bellman equation (12.3.4).

Initialization. Make initial guess for policy, U , and value, V^0 , functions; choose stopping criterion $\varepsilon > 0$.

Step 1. Set $I = \emptyset$, $W = V^k$.

Step 2. Simulate the problem under U for T periods beginning with some $x_{i_1} \notin I$ to produce a sequence $x_{i_2}, x_{i_3}, \dots, x_{i_T}$. Add the realized states to the set I .

Step 3. Recompute value function at the realized states in the upwind Gauss-Seidel direction:

$$W_l = \max_{u \in D} \pi(x_l, u) + \beta \sum_{j=1}^n q_{lj}(u) W_j, \quad l = i_T, i_{T-1}, \dots, i_1.$$

At each l set U_l equal to the maximizing choice of u .

Step 4. If $I \neq X$, go to step 2; otherwise, go to step 5.

Step 5. Set $V^{k+1} = W$. If $\|V^{k+1} - V^k\| < \varepsilon$, go to step 6; else go to step 2.

Step 6. Compute the final solutions:

$$U^* = \mathcal{U} V^{k+1},$$

$$P_i^* = v(x_i, U_i^*), \quad i = 1, \dots, n,$$

$$V^* = (I - \beta Q^{U^*})^{-1} P^*.$$

and STOP.

Linear Programming Approach

Another approach to finite-state, finite-control dynamic programming is to reformulate it as a linear programming problem. It is clear the (12.3.4) is equivalent to the linear program

$$\begin{aligned} & \min_{V_i} \sum_{i=1}^n V_i \\ \text{s.t. } & V_i \geq \pi(x_i, u) + \beta \sum_{j=1}^n q_{ij}(u) V_j, \quad \forall i, u \in \mathcal{D}, \end{aligned} \tag{12.4.10}$$

where \mathcal{D} is finite. This linear program finds the smallest value function that satisfies the inequalities implied by (12.3.4) which are here the constraints. While (12.4.10) may be a large problem, the critical fact is that all components are linear in the V_i .

values. This transformation allows us to use linear programming to solve dynamic programming.

This approach has not been viewed favorably in the operations research literature. However, it may have value for some problems we see in economics. Once we have reformulated the problem in this fashion, we can utilize acceleration methods for linear programming problems; Trick and Zin (1997) pursued one such approach with success.

12.5 Discretization Methods for Continuous-State Problems

In many economic applications, both the state and control variables are naturally continuous. One way to approximately solve such problems is to specify a finite-state problem that is “similar” to the continuous problem, and solve the finite problem using finite-state methods. In this section we will examine the details of this approach.

Single-State Optimal Growth

We examine the deterministic growth model exposed in (12.1.12) with the Bellman equation

$$V(k) = \max_c u(c) + \beta V(F(k) - c). \quad (12.5.1)$$

Discretizing this problem is fairly easy but not trivial. We first replace the continuous-state variable, k , with a finite set, $K = \{k_1, \dots, k_n\}$, of permissible values. We next need to choose a collection of values for c , the control variable, that are consistent with the discretization of the state variable. Specifically, since V will be defined only for $k \in K$, those c values must be such that they keep the capital stock on that grid. Therefore the set of permissible c choices varies with the state k . To deal with this difficulty, we rewrite this problem as

$$\max_{I_t} \sum_{t=0}^{\infty} \beta^t u(F(k_t) - I_t) \quad (12.5.2)$$

$$\text{s.t. } k_{t+1} = I_t,$$

where now the control is the next period’s state. This kind of transformation is not always possible, but when possible, it is convenient to have the control variable be the next period’s state.

The Bellman equation for (12.5.2) becomes

$$V(k) = \max_{k^+} u(F(k) - k^+) + \beta V(k^+) \quad (12.5.3)$$

where tomorrow's capital stock, k^+ , is the control. The consumption choice equals $F(k) - k^+$. In this form it is now easy to create a discrete version of (12.5.3) since the future state is the control. We let K be both the set of states, X , and the set of permissible controls, D , and solve the finite-state, finite-control problem

$$V(k_i) = \max_{k_j \in K} u(F(k_i) - k_j) + \beta V(k_j), \quad i = 1, \dots, n. \quad (12.5.4)$$

Some care is necessary in choosing K . The finiteness of K implies that there is a minimum, k^m , and a maximum, k^M , value in K . Therefore we are really discretizing the problem

$$\begin{aligned} & \max_{c_t} \sum_{t=0}^{\infty} \beta^t u(c_t) \\ \text{s.t. } & k_{t+1} = F(k_t) - c_t, \\ & k^m \leq k_t \leq k^M. \end{aligned} \quad (12.5.5)$$

We want to choose values in K such that the problems (12.5.1) and (12.5.5) are nearly the same. Theory tells us that if $u(c)$ and $F(k)$ are concave the solution to (12.5.1) will converge to a steady state capital stock, k^{ss} , defined by $\beta F'(k^{ss}) = \beta(1 + f'(k^{ss})) = 1$. The constraints $k^m \leq k_t \leq k^M$ will not bind as long as $k_0, k^{ss} \in [k^m, k^M]$; hence we should choose k^m and k^M so that $k^m < k_0, k^{ss} < k^M$. In general, one should choose a range $[k^m, k^M]$ such that the optimal solution to (12.5.1) begins and stays in $[k^m, k^M]$. Otherwise, the constraints implicit in the discretization will cause the solution to (12.5.1) to differ from the solution to (12.5.5).

We will now examine the performance of various methods when applied to (12.5.2). Specifically, we choose $u(c) = c^{\gamma+1}/(\gamma+1)$, and $F(k) = k + f(k) \equiv k + ((1-\beta)/\alpha\beta)k^\alpha$. This implies a steady state at $k = 1$. We will focus on the interval $[0.8, 1.2]$. We choose $\gamma = -2$, $\alpha = 0.25$, and $\beta = 0.96$. Our discretization in general will be $k_i = 0.8 + (i-1)\kappa$, for some κ . We first choose $\kappa = 0.001$, which implies 401 capital stocks between $k = 0.8$ and $k = 1.2$.

We first explore value iteration. Table 12.1 displays the results at a few capital stocks. The first two columns, labeled V^* and U^* , display the optimal value and policy for the $\kappa = 0.001$ case. In this problem there is a good initial guess. One of the feasible policies here is zero net saving, that is, $k^+ = k$. Under this policy the utility flow at state k is $u(f(k))$ forever, yielding a discounted value of $u(f(k))/(1-\beta)$. Therefore we know that the function $V^*(k) \equiv u(f(k))/(1-\beta)$ is a feasible value function and hence less than the value function of the optimal policy. However, since savings is probably small at any particular k , $V^*(k)$ is probably not too far away

Table 12.1
Dynamic programming solutions to (12.52)—Value iteration

<i>k</i>	Solution		Case 1					Case 2				
	V^*	U^*	V^0	U^1	V^1	U^{10}	V^{10}	V^0	U^1	V^1	U^{10}	V^{10}
0.8	-23.7454	0.816	-23.7908	0.826	-23.7772	0.817	-23.7475	0	0.8	-2.38	.811	-15.5
0.9	-23.0906	0.908	-23.1005	0.913	-23.0976	0.908	-23.0911	0	0.8	-1.88	.901	-14.8
1.0	-22.5000	1.000	-22.5000	1.000	-22.5000	1.000	-22.5000	0	0.8	-1.55	.991	-14.3
1.1	-21.9623	1.092	-21.9702	1.097	-21.9681	1.092	-21.9628	0	0.8	-1.32	1.081	-13.8
1.2	-21.4689	1.184	-21.4945	1.174	-21.4899	1.183	-21.4706	0	0.8	-1.16	1.170	-13.3

Table 12.2
Dynamic programming solutions to (12.5.2)—Policy iteration

<i>k</i>	Case 1					Case 2				
	V^0	U^1	$V^{1,50}$	U^2	$V^{2,50}$	V^0	U^1	V^1	$V^{1,50}$	U^5
0.8	-23.79	0.826	-23.75	0.820	-23.7459	0	0.8	-2.38	-23.68	0.818
0.9	-23.10	0.913	-23.09	0.908	-23.0907	0	0.8	-1.88	-23.17	0.917
1.0	-22.50	1.000	-22.50	1.000	-22.5000	0	0.8	-1.55	-22.85	0.993
1.1	-21.97	1.087	-21.96	1.092	-21.9624	0	0.8	-1.32	-22.63	1.091
1.2	-21.50	1.174	-21.47	1.185	-21.4691	0	0.8	-1.16	-22.46	1.194

from the true value function. Hence let $V^0(k) = V^c(k)$ be the initial guess. We display the results in case 1 of table 12.1. This initial guess is close to the solution, and the subsequent iterates are quite good. Also, even the first policy iterate U^1 is quite good.

We also tried the initial guess $V^0 = 0$. This is a general purpose initial guess but seldom a good one. We display the results in case 2 of table 12.1. The initial guess $V^0 = 0$ implies the constant policy $U^1 = 0.8$, choosing 0.8, the minimal capital stock, for next period's capital stock no matter what the current k is. This policy is a poor one; a larger grid would have produced an even worse U^1 , since U^1 will always choose maximal consumption if $V^0 = 0$. Even the tenth iterates are rather poor. Comparisons of the two initial guesses used in table 12.1 shows the value of a good initial guess.

We next examine policy and modified policy iteration in table 12.2. The value function $V^{1,50}$ is the value function at the end of the first iteration in (12.4.1) when we take 50 policy iterations in (12.4.1). We will again use the initial guesses $V^0 = V^c$ and $V^0 = 0$. The results are displayed in table 12.2. The combination of a good initial guess and policy iteration is impressive; in fact U^2 is correct to 0.001 in case 1 where

$V^0 = V^c$. The $V^0 = 0$ initial guess again leads to slow progress in case 2, but far better than the value iteration scheme.

We now use the upwind Gauss-Seidel procedure to solve (12.5.2) rapidly and illustrate convergence properties as we take finer discretizations. Theory tells us that both $C(k)$ and $F(k) - C(k)$ are monotonically increasing. These properties imply that the solution is stable with the optimal policy moving the capital stock toward the steady state capital stock, whose value we know. Therefore from any capital stock we know the downwind direction in which the state will move, telling us the upwind direction in which we should compute the solution.

This means that the growth problem (12.5.1) has the same structure as the example in (12.4.9), and that we can develop a one-pass algorithm for computing the value and policy functions. We choose F and β so that $k^* = 1$. Since the steady state is an absorbing state, we know $V(k^*) = u(f(k^*))/(1 - \beta)$. Next consider the capital stock just below the steady state, $1 - \kappa$. Since it is less than the steady state, the optimal policy will either choose zero net saving, in which case the capital stock will stay at $1 - \kappa$ forever, or it will choose investment equal to κ and move the capital stock up to the steady state level, where it will stay forever. This exhausts all of the possibilities for the optimal policy since, due to the monotonicity of C and $F - C$, it is not optimal to overshoot the steady state and later retreat to it, nor is it optimal to reduce the capital stock and later climb to the steady state. We can compute the value of these two choices and know that the correct value is the greater of the two. Hence from the steady state we can calculate the value function at the next lower capital stock. Proceeding in this way, we can calculate the value and policy at $1 - i\kappa, i = 2, 3, \dots$, down to $k = 0.1$. Symmetric considerations show that we can also compute the value function at capital stocks in excess of the steady state, starting at $k = 1.0$ and proceeding to $k = 1.9$ in steps of κ . This produces the value function in just one pass.

Table 12.3 displays the results for the case $\alpha = 0.33333$, $\gamma = -2$, and $\beta = 0.95$ and we choose $\kappa = 10^{-4}, 10^{-5}$, and 10^{-6} , implying 18,001 to 1,800,001 capital stocks in $[0.1, 1.9]$. The results show that the computed policy functions converge smoothly. In fact, for most capital stocks, the computed optimal policy at $\kappa = 10^{-4}$ and $\kappa = 10^{-5}$ equals the optimal policy for $\kappa = 10^{-6}$ rounded to the nearest point in the coarser discretizations. For example, the optimal saving policy at $k = 0.9$ is 0.003036 if $\kappa = 10^{-6}$, 0.00304 if $\kappa = 10^{-5}$, and 0.0030 if $\kappa = 10^{-4}$. Therefore it appears that the computed savings function is as accurate as possible given the discretization κ .

All of the policy functions in table 12.3 were computed in a few minutes on a 90-MHz Pentium, including the case with 1,800,001 capital stocks. This efficiency is due to the one-pass nature of the UGS scheme we used, which in turn relied on our knowing the steady state. This is not essential. If we begin with the initial guess of

Table 12.3
UGS solution to (12.5.2) with $\alpha = 1/3$, $\gamma = -2$

k	κ		
	10^{-4}	10^{-5}	10^{-6}
1.9	-0.028700	-0.028690	-0.028692
1.5	-0.015800	-0.015750	-0.015752
1.1	-0.003100	-0.003090	-0.003086
1.0	0	0	0
0.9	0.003000	0.003040	0.003036
0.5	0.014300	0.014260	0.014261
0.1	0.019200	0.019240	0.019241

zero net saving, then the optimal value of any steady state is computed correctly. If we then apply the alternating sweep Gauss-Seidel procedure we will correctly compute the optimal value function after visiting each capital stock only twice. Hence, even if we did not know the steady state, we could solve the problem in at most twice the time of a single-pass scheme.

The final issue we will analyze is the size of the error due to discretization. The value function computed in table 12.3 is clearly a lower bound since the policy function it uses would continue to be feasible in the continuous k case. We would like an upper bound for V . To do that, we examine the dynamic programming problem

$$V^u(k) = \max_{k^+} u(F(k) - k^+ + \kappa) + \beta V^u(k^+) \quad (12.5.6)$$

where, recall, κ is the difference between successive k 's in our discretization. The problem (12.5.6) allows an individual to consume an extra κ units of consumption in each period. Suppose that k were allowed to be continuous and $k^+ = I(k)$ were the optimal policy. An individual would attain greater utility if, in the continuous k case, he chose k^+ instead to be the next highest value in K above $I(k)$, say k_j , and consumed $f(k) + k - k_j + \kappa$; but that is now a policy which keeps k on the discrete grid in (12.5.2), and feasible in (12.5.6). Therefore V^u exceeds the value function of the continuous state problem, but can be solved exactly since it is a discrete problem. By combining the error estimates for the solution of the discrete problems, we can compute upper and lower bounds for the true value function.

Commodity Storage Problems

We next describe the optimal commodity stockpiling problem which agricultural economists have studied intensively. This problem goes back to the analysis of

Gustafson (1958), to which we will return later. Here we focus on the dynamic programming analysis.

The commodity storage model assumes that a crop is harvested each year, with the yield allocated between current consumption and changes in inventory. The state variable is the inventory, which can never be negative. The social planner also chooses inputs for the next year's production. We include *convenience value*, that is, the value to having a positive level of inventory. Specifically $u(c_t)$ is the per-period utility function, expressed in dollars, over consumption of grain in time t , c_t , $L(y_t)$ is the dollar cost of input choice (e.g., labor effort) y_t at the end of period t , and I_t is the nonnegative beginning-of-period- t inventory. We assume that output in period $t + 1$, q_{t+1} , equals $y_t \theta_{t+1}$, where θ_{t+1} is a serially uncorrelated productivity shock. We first let the total amount of grain available *after* harvest, including inventories, be the state variable; let $x_t \equiv I_t + q_t$ denote the state. After harvest, the grain supply is allocated between consumption and inventories, $x_t = c_t + I_{t+1}$, and input, y_t , is chosen. The convenience value in period t of the inventory I_{t+1} is $g(I_{t+1})$. This implies the dynamic programming problem

$$V(x) = \max_{0 \leq y, I \leq x} u(x - I) + g(I) - L(y) + \beta E\{V(I + \theta y)\}. \quad (12.5.7)$$

This problem is novel because of the nonnegativity constraint on I . In this form it is also an example of where we can discretize neither the state nor control space because we have no guarantee that $I + \theta y$ will lie on the prescribed grid. This difficulty can easily arise in stochastic problems.

To apply discretization methods, one must develop a reformulation that allows discretization. In this case we may proceed by changing the definition of state variable and focusing on a function other than $V(x)$. Consider the inelastic supply case with no convenience value, modeled here as $y = 1$ and $g(I) = 0$. In this case we proceed as follows: First, consider the dynamic programming problem which uses I , the *end-of-period* stock, as both the state and control variable:

$$W(I) = \beta E \left\{ \max_{0 \leq I^+ \leq I + \theta} u(I + \theta - I^+) + W(I^+) \right\},$$

where I^+ is the next period's inventory chosen after θ is observed. To keep matters simple, we will assume that θ is uniformly distributed over $[\theta_m, \theta_M]$. If we choose a grid of I values, $I_i, i = 1, \dots, n_I$, and a uniform grid of θ values on $[\theta_m, \theta_M]$, $\theta_j, j = 1, \dots, n_\theta$, we arrive at the discrete-state problem

$$W(I_i) = \frac{1}{n_\theta} \beta \sum_{j=1}^{n_\theta} \left\{ \max_{0 \leq I^+ \leq I_i + \theta_j} u(I_i + \theta_j - I^+) + W(I^+) \right\}.$$

In this discretization, for each I_i , we cycle through the θ_j values, where for each θ_j we solve

$$\max_{0 \leq I^+ \leq I_i + \theta_j} u(I_i + \theta_j - I^+) + W(I^+)$$

and then take the average of these maximal values over the θ values to arrive at our approximation to $W(I_i)$. When we have done this for each I_i , we have our new approximation for the value function W . We can apply policy iteration to accelerate convergence as long as we store the policy function approximations, $\mathcal{J}^+(I_i, \theta_j)$, along the way.

We can also use this procedure for the elastic supply case; here the end-of-period state vector is (I, y) , and Bellman's equation is

$$W(I, y) = \beta E \left\{ \max_{0 \leq y^+, I^+ \leq I + \theta} u(I + y\theta - I^+) - L(y^+) + W(I^+, y^+) \right\}.$$

Again we can discretize I , y , and θ to arrive at a discrete-state approximation of $W(I, y)$.

This example shows why we must be careful in formulating dynamic programming problems when developing computational strategies. While all formulations of a dynamic programming problem are theoretically equivalent, they may differ substantially in their numerical properties. Here we find that when we use a different state variable and rewrite the problem, we can go from a formulation where discretization is not possible to one where discretization can be applied.

Limits of Discretization Methods

Discretization procedures are of limited value even if one fully uses a priori information about the solution. They are often inefficient since, in order to have any hope of realistically approximating an agent's flexibility and the stochastic law of motion, a large number of points are required. Multidimensional problems are practically impossible since the "curse of dimensionality" is particularly vexing for this method; if N points are used for a one-dimensional problem, then it would be natural to use N^k points for a k -dimensional problem. Since N must be large, N^k is infeasible for even small k . This problem would be reduced by using more efficient grids motivated by multidimensional quadrature methods, such as monomial formulas and low-discrepancy sets of points. For multidimensional problems with continuous state variables and smooth value functions, there are alternatives to discretization. We will now turn to those methods.

12.6 Methods for Solving Linear-Quadratic Problems

A special case of dynamic programming is the class of linear-quadratic problems. This class is used extensively in dynamic economic analyses because it is relatively tractable. We also study it here because it is a simple example of a different approach to solving dynamic programs. When we solve any dynamic programming problem, we focus on solving the decision rule and/or the value function. In the case of linear-quadratic problems, we know that the value function is a quadratic form and the decision rule is linear in the state. Therefore, to solve for these unknown functions, we solve for the unknown coefficients of the quadratic value function or linear decision rule. This is an obvious approach to take in the linear-quadratic case since we know the form of the solution. We will see that it is also a natural approach in general, providing an alternative to discretization methods.

The discrete-time linear-quadratic optimal control problem assumes a payoff flow function $\pi(x, u, t) = \frac{1}{2}x^T Q_t x + u^T R_t x + \frac{1}{2}u^T S_t u$, a discount factor of $\beta \in [0, 1)$, a law of motion $x_{t+1} = A_t x_t + B_t u_t$, and a terminal valuation $\frac{1}{2}x^T W_{T+1} x$. Without loss of generality, we can assume that Q_t , S_t , and W_{T+1} are symmetric; we will also assume here that $\pi(x, u, t)$ is concave in (x, u) , implying that we have a well-behaved concave dynamic optimization problem. This is a general representation once we adopt the convention that the first component in x is the constant 1. With these specifications the problem is

$$\max_{u_t} \sum_{t=0}^T \beta^t (\frac{1}{2}x_t^T Q_t x_t + u_t^T R_t x_t + \frac{1}{2}u_t^T S_t u_t) + \frac{1}{2}x_{T+1}^T W_{T+1} x_{T+1} \quad (12.6.1)$$

$$\text{s.t. } x_{t+1} = A_t x_t + B_t u_t,$$

and the Bellman equation becomes

$$V(x, t) = \max_{u_t} \frac{1}{2}x^T Q_t x + u_t^T R_t x + \frac{1}{2}u_t^T S_t u_t + \beta V(A_t x + B_t u_t, t + 1). \quad (12.6.2)$$

If we make the guess that $V(x, t) = \frac{1}{2}x^T W_t x$, then (12.6.2) implies that

$$0 = S_t u_t + R_t x + \beta B_t^T W_{t+1} (A_t x + B_t u_t),$$

which in turn implies the control law

$$u_t = -(S_t + \beta B_t^T W_{t+1} B_t)^{-1} (R_t + \beta B_t^T W_{t+1} A_t) x \equiv U_t x. \quad (12.6.3)$$

After some manipulation we find that

$$W_t = Q_t + \beta A_t^\top W_{t+1} A_t + (\beta A_t^\top W_{t+1} B_t + R_t^\top) U_t. \quad (12.6.4)$$

Equation (12.6.4) is referred to as the *Riccati equation*.

Solving the finite-horizon problem is easy. Since we know W_{T+1} , we can iteratively apply (12.6.4) to solve for the W_t ; under the assumption of strict concavity of π and W_{T+1} , the necessary inverses in (12.6.4) exist. The iteration in (12.6.4) is essentially the basic value function iteration method.

The infinite-horizon case is a bit more difficult. Suppose that $T = \infty$, $R_t = R$, $Q_t = Q$, $S_t = S$, $A_t = A$, and $B_t = B$ in (12.6.1). Then the value function becomes autonomous; in particular, $V(x) \equiv \frac{1}{2}x^\top Wx$ for some W which is a solution to the *algebraic Riccati equation*

$$W = Q + \beta A^\top WA - (\beta A^\top WB + R^\top)(S + \beta B^\top WB)^{-1}(\beta B^\top WA + R). \quad (12.6.5)$$

Solving (12.6.5) for W is not simple because it is a quadratic matrix equation. Our dynamic programming intuition for the concave case does produce a method. The value function iteration approach to solving (12.6.5) is to choose some initial negative definite guess W_0 and compute the sequence

$$W_{k+1} = Q + \beta A^\top W_k A - (\beta A^\top W_k B + R^\top)(S + \beta B^\top W_k B)^{-1}(\beta B^\top W_k A + R). \quad (12.6.6)$$

The W_k iterates will converge to the solution, W , of (12.6.5).

We can also use policy iteration to help solve (12.6.5). Suppose that we begin with our initial guess, W_0 , and perform one value function iteration; a by-product of such is a guess of the policy function,

$$U = -(S + \beta B^\top W_0 B)^{-1}(R + \beta B^\top W_0 A).$$

Given the terminal value function W_0 and the policy rule U , the Bellman equation implies that the current value function is

$$\frac{1}{2}x^\top W_1 x = \frac{1}{2}x^\top Qx + (Ux)^\top Rx + \frac{1}{2}(Ux)^\top SUx + \beta \frac{1}{2}x^\top W_0 x.$$

If we followed the policy U forever, the resulting value function would be $\frac{1}{2}x^\top Wx$ with

$$W = \frac{\frac{1}{2}Q + \frac{1}{2}U^\top SU + U^\top R}{1 - \beta}.$$

Hence policy iteration for linear-quadratic problems is summarized by the iteration

$$U_{i+1} = -(S + \beta B^\top W_i B)^{-1} (R + \beta B^\top W_i A),$$

$$W_{i+1} = \frac{\frac{1}{2}Q + \frac{1}{2}U_{i+1}^\top S U_{i+1} + U_{i+1}^\top R}{1 - \beta},$$

which continues until $\|W_{i+1} - W_i\|$ is small for some norm $\|\cdot\|$.

12.7 Continuous Methods for Continuous-State Problems

Most of the methods examined so far either assume that there were only a finite number of states, or approximate a continuous state with a finite set of values. These methods are reliable in that they will solve the problem, or will approach the solution as the discretization is made finer. However, discretization becomes impractical as one moves to larger problems. In this section we introduce a parametric approach due to Bellman et al. (1963).

We don't use discretization to solve linear-quadratic problems because we know the functional form of the solution. This allows us to focus on finding the appropriate coefficients. Linear-quadratic problems do not suffer a curse of dimensionality since the number of unknown coefficients do not grow exponentially in the dimension. Even though most dynamic programming problems do not have a useful functional form, we can still use the functional form approach by exploiting approximation ideas from chapter 6.

Recall the basic Bellman equation:

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

The unknown here is the function V . Discretization methods essentially approximate V with a step function, since it implicitly treats any state between x_i and x_{i+1} as either x_i or x_{i+1} . Chapter 6 presented better methods to approximate continuous functions. We apply those ideas here.

We assume that the payoff, motion, and value functions are all continuous functions of their arguments. The basic idea is that it should be better to approximate the continuous-value function with continuous functions and put no restrictions on the states and controls other than those mandated by the problem. Since the computer cannot model the entire space of continuous functions, we focus on a finitely parameterizable collection of functions,

$$V(x) \doteq \hat{V}(x; a). \quad (12.7.2)$$

The functional form \hat{V} may be a linear combination of polynomials, or it may represent a rational function or neural network representation with parameters $a \in R^m$, or it may be some other parameterization specially designed for the problem.

Once we fix the functional form, we focus on finding coefficients $a \in R^m$ such that $\hat{V}(x; a)$ “approximately” satisfies the Bellman equation. Solving the Bellman equation, (12.7.1), means finding the fixed point of T , but that is the pure mathematical problem. The basic task for a numerical procedure is to replace T , an operator mapping continuous functions to continuous functions, with a finite-dimensional approximation, \hat{T} , which maps the set of functions of the form \hat{V} into itself. If done properly, the fixed point of \hat{T} should be close to the fixed point of T .

The construction of \hat{T} relies on three critical steps. First, we choose some parameterization scheme $\hat{V}(x; a)$ with $a \in R^m$, and n points in the state space,

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

where $n \geq m$. Second, we then evaluate $v_i = (T\hat{V})(x_i)$ at each $x_i \in X$; we refer to this as the *maximization step*. The maximization step gives us values v_i which are points on the function $T\hat{V}$. We next use the information about $T\hat{V}$ contained in the $v_i, i = 1, \dots, m$, to find an $a \in R^m$ such that $\hat{V}(x; a)$ best fits the $(v_i, x_i), i = 1, \dots, n$, data; we call this the *fitting step*. The fitting step can be an unweighted nonlinear least squares procedure as in

$$\min_{a \in R^m} \sum_{i=1}^n (\hat{V}(x_i; a) - v_i)^2$$

or any other appropriate approximation scheme. This fitting step produces a new value function defined on all states x . The *parametric dynamic programming algorithm* is outlined in algorithm 12.5.

Algorithm 12.5 Parametric Dynamic Programming with Value Function Iteration

Objective: Solve the Bellman equation, (12.7.1).

Initialization. Choose functional form for $\hat{V}(x; a)$, and choose the approximation grid, $X = \{x_1, \dots, x_n\}$. Make initial guess $\hat{V}(x; a^0)$, and choose stopping criterion $\varepsilon > 0$.

Step 1. Maximization step: Compute $v_j = (T\hat{V}(\cdot; a^i))(x_j)$ for $x_j \in X$.

Step 2. Fitting step: Using the appropriate approximation method, compute the $a^{i+1} \in R^m$ such that $\hat{V}(x; a^{i+1})$ approximates the (v_i, x_i) data.

Step 3. If $\|\hat{V}(x; a^i) - \hat{V}(x; a^{i+1})\| < \varepsilon$, STOP; else go to step 1.

Steps 1 and 2 in algorithm 12.5 constitute a mapping \hat{T} taking \hat{V} , corresponding to a parameter vector a , to another function $\hat{T}\hat{V}$ corresponding to another coefficient vector, a' . \hat{T} is therefore a mapping in the space of coefficients, a subspace of R^m .

Algorithm 12.5 presents the general idea; we now examine the numerical details. Rewrite (12.7.1) as

$$V(x) = \max_{u \in D(x)} \pi(u, x) + \beta \int V(x^+) dF(x^+ | x, u), \quad (12.7.4)$$

where $F(x^+ | x, u)$ is the distribution of the future state x^+ conditional on the current state and control. The maximization steps compute

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

With the v_j values for the grid X , we next compute a new value function approximation $\hat{V}(x; a)$ which fits the new v_j values at the $x_j \in X$.

These expressions make clear the three kinds of problems we need to handle. The first type of numerical problem is evaluating the integral $\int \hat{V}(x^+) dF(x^+ | x, u)$. This would usually require numerical quadrature. If the integrand is smooth in x^+ for fixed x and u , a Gaussian approach is suggested. Less well-behaved integrals would require low-order Newton-Cotes formulas. High-dimensional integrands would require a monomial, Monte Carlo, or quasi-Monte Carlo method.

The second type of numerical problem appearing in (12.7.4) is the optimization problem, (12.7.5). If the objective in (12.7.5) is smooth in u , we could use faster methods such as Newton's method. It is important to choose an approximation scheme \hat{V} which preserves any smoothness properties of the problem.

Third, given the v_i estimates of the $(T\hat{V})(x_i)$ values, we need to compute the new coefficients in $\hat{V}(x; a)$. The appropriate approximation procedure depends on the nature of V . If we expect V to be a smooth function, then orthogonal polynomial procedures may be appropriate. Otherwise, splines may be advisable. We will see below that these considerations are particularly important here.

There is substantial interaction across the three problems. 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). Since the integral in (12.7.5) is costly to evaluate, we may want to use different rules when computing the gradients and Hessian, using the observation that low-quality gradient and Hessian approximations often suffice.

While algorithm 12.5 looks sensible, there are several questions concerning its value. First, convergence is not assured since \hat{T} may not be a contraction map; in

fact, we will see that it may be quite ill-behaved. The key detail is the choice of the approximation scheme incorporated in $\hat{V}(x; a)$ and the grid X . The discussion below will focus on the behavior of this algorithm for various choices of these elements.

We should emphasize that we can still use some of the techniques developed for the finite-state case. Algorithm 12.5 presents only value iteration, but we could also implement policy iteration. Many other procedures are not so easily adapted for this approach. The Gauss-Seidel ideas, particularly the upwind methods, do not have obvious counterparts consistent with the parametric approach in general. This leaves open the possibility that the finite-state approach may dominate the functional approximation approach for some problems because of the applicability of Gauss-Seidel acceleration ideas.

12.8 Parametric Approximations and Simulation Methods

The main idea of parametric dynamic programming methods is to parameterize the critical functions and find some parameter choice which generates a good approximation. One direct and simple implementation of that idea is to parameterize the control law, $\hat{U}(x; a)$, and through simulation find that coefficient choice, a , which generates the greatest value. In this section we will discuss a simple application of that approach.

Consider again the stochastic growth problem:

$$V(k) = \max_c u(c) + \beta E\{V(k - c + \theta f(k - c))|k, c\}, \quad (12.8.1)$$

where the θ are i.i.d. productivity shocks affecting the net output function f ; this is a special case of (12.1.20). For smooth concave problems we know that the true policy and value functions, $C(k)$ and $V(k)$, are smooth functions, increasing in k . In this section we will use a simple simulation approach to solve the stochastic growth problem (12.8.1).

Instead of parameterizing $C(k)$, we parameterize the savings function, $S(k) \equiv k - C(k)$. We know that S is increasing but that $S(k_1) - S(k_2) \leq k_1 - k_2$ for $k_1 \geq k_2$; these properties allow us to examine a simple class of rules. We will examine linear rules; hence $\hat{S}(k) = a + bk$ for coefficients a and b where $b \in (0, 1)$. We will use simulation to approximate the value of a savings rule. Suppose that $\theta_t, t = 1, \dots, T$ is a sequence of productivity shocks. Then, for a given initial value of k_0 , the resulting paths for c_t and k_t are given by $c_t = k_t - \hat{S}(k_t)$ and $k_{t+1} = \hat{S}(k_t) + \theta_t f(\hat{S}(k_t))$, and the realized discounted utility is

$$W(\theta; \hat{S}) = \sum_{t=0}^T \beta^t u(c_t). \quad (12.8.2)$$

We can do this for several θ_t sequences. Let θ^i be the i th sequence drawn and let c_t^i be the resulting consumption when we compute (12.8.2). The value of a rule $\hat{S}(k)$ beginning at k_0 is $V(k_0; \hat{S}) = E\{W(\theta; \hat{S})\}$, which can be approximated by the sum

$$\frac{1}{N} \sum_{j=1}^N W(\theta^j; \hat{S}) = \frac{1}{N} \sum_{j=1}^N \sum_{t=0}^T \beta^t u(c_t^j). \quad (12.8.3)$$

Note that (12.8.3) is essentially an integral over the space of θ series. A literal “simulation” approach would construct each θ by a sequence of i.i.d. draws.

This value of $W(\theta; S)$ depends on the initial capital stock k_0 , the particular realization of θ_t , $t = 1, \dots, T$, and the choices for a and b . The use of several θ realizations makes the average in (12.8.3) less sensitive to the particular realizations used.

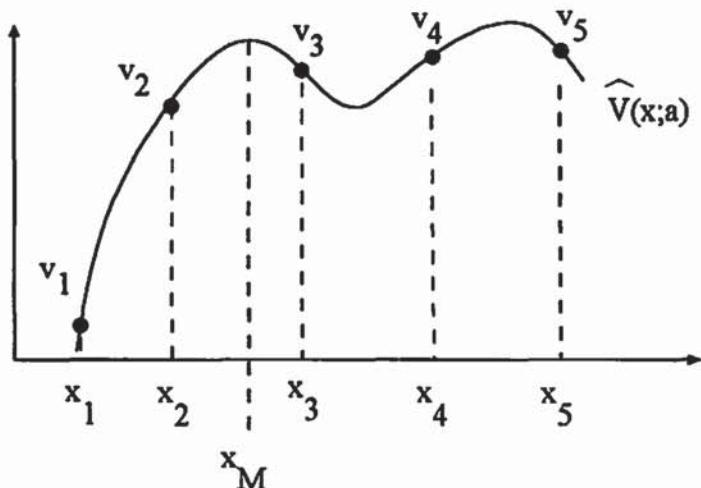
Once we have a way to approximate $V(k_0; \hat{S})$ for a linear rule \hat{S} parameterized by a and b , we optimize over a and b to approximate the optimal linear $\hat{S}(k)$ rule. Since k_0 is the initial capital stock in our definition of $V(k_0; \hat{S})$, this approximation depends on k_0 , whereas the optimal rule does not. To reduce the sensitivity of the chosen S rule to k_0 , we should choose k_0 to be close to the “average” capital stock.

This sounds easy but one can run into problems. Our procedure does not impose any restriction on $\hat{S}(k)$. In particular, $\hat{S}(k)$ could be negative at some k , or consumption $k - \hat{S}(k)$ could be negative, with either possibility causing problems because f and u are usually defined only for positive arguments. To deal with this problem, one should constrain the linear approximation, implying that $\hat{S}(k) = \min[\max[0, a + bk], k]$, or choose some transformation that avoids this possibility. The true rule may not be linear; in that case we could use a more flexible specification for \hat{S} .

The simulation approach is not efficient for smooth problems like (12.8.1) where we can exploit the continuity and concavity properties of the solution. However, more complex dynamic programming problems involving constraints, several dimensions, and/or integer variables may be approximately solved by simulation strategies. The key idea is to parameterize a family of control laws, use simulation to evaluate them, and choose the best. See Smith (1990) for an application of these ideas.

12.9 Shape-Preserving Methods

The parametric approach to dynamic programming is promising but can fail if we are not careful. To illustrate this, consider the problem (12.5.1) with a very concave

**Figure 12.2**

Dynamic programming and shape of value function interpolant

utility function. In figure 12.2, we display the results of a typical value function iteration. Suppose that we have chosen $x_i, i = 1, \dots, 5$, for the nodes of the approximation and that we computed v_1, v_2, v_3, v_4 , and v_5 as in figure 12.2. These five points appear to be consistent with an increasing and concave value function. However, applying interpolation to these data to fit $\hat{V}(x; a)$ may produce a curve, neither concave nor monotone increasing, such as $\hat{V}(x; a)$ in figure 12.2. Even worse is that the maximum of $\hat{V}(x; a)$ is a point between x_2 and x_3 , and even exceeds the maximum v_i values.

While these internodal fluctuations are consistent with the approximation theory of chapter 6, they can wreck havoc with dynamic programming. For example, suppose that the true V is increasing and concave but that at some iteration \hat{V} looks like \hat{V} in figure 12.2. In the next iteration, x_M will be considered a very desirable state, and controls will be chosen to push the state towards x_M . The artificially high value of $\hat{V}(x_M)$ will lead to artificially high values for $\hat{V}(x_2)$ and $\hat{V}(x_3)$ in the next maximization step. The errors at x_2 and x_3 could interact with the values computed elsewhere to produce even worse internodal oscillations at the next fitting stage. Once this process begins, it can feed on itself and destabilize the value iteration procedure.

The problem here is the absence of shape-preservation in the algorithm. Shape-preservation is a valuable property, particularly in concave problems. If \hat{V} and $\pi(x, u)$ are concave in x and u , the maximization step is a concave problem; hence the global maximum is the unique local maximum and easy to find. Furthermore T is a shape-preserving operator; that is, if $\pi(x, u)$ and the conditional expectation is concave in (x, u) for concave V , then TV is concave if V is concave. Therefore, if \hat{V} is concave

in x , then the $(T\hat{V})(x_i)$ points will be concave, and a shape-preserving scheme will cause $(\hat{T}\hat{V})(x)$ to be a concave function of x . Approximation methods should match the shape properties of the approximated objects.

This does not mean that we can't use the polynomial methods. In some instances these problems do not arise. For example, if $V(x)$ is C^∞ with well-behaved high-order derivatives, then orthogonal polynomial approximation is a good approximation choice, and there is less chance of these problems arising.

However, we still want to find more reliable procedures. Following is a discussion of methods that avoid these problems by design. The key idea is the use of shape-preserving approximation methods. Discretization methods will preserve shape and avoid these problems. More promising are the shape-preserving methods discussed in chapter 6.

Linear Interpolation

For one-dimensional problems disruptive internodal oscillations can be avoided by the simplest of all interpolation schemes—linear interpolation. Furthermore, as discussed in chapter 6, linear interpolation is shape-preserving. Therefore, if the v_i points are increasing and concave, so will be the interpolating function.

The problem with linear interpolation is that it makes the maximization step less efficient. The kinks in a linear interpolant will generally produce kinks in the objective of the maximization step, forcing us to use slower optimization algorithms. Kinks in the value function may cause the approximate policy function to be discontinuous, an unappealing property if the true policy function is continuous. Using linear interpolation is a costly way of preserving shape.

Multilinear Interpolation Methods

In multidimensional problems there are a couple of easy well-behaved approximation schemes which one can use. First, one could use the multilinear or simplicial interpolation methods discussed in chapter 6. The DYGAM package discussed in Dantzig et al. (1974) used multilinear interpolation. These methods eliminates problematic internodal oscillations since \hat{V} at each point in the interior of a box is a convex combination of the values at the vertices. The fact that the interpolation scheme is monotone in the interpolation data means that \hat{T} is monotone. Furthermore \hat{T} inherits the contraction properties of T . Multilinear interpolation is costly to compute; a less costly alternative is multidimensional simplicial interpolation, as discussed in chapter 6.

Unfortunately, multilinear and simplicial interpolation have the same problems of one-dimensional linear interpolation and more. First, they preserve positivity and

monotonicity but not concavity. Second, they also produce kinks in the value function and discontinuities in the policy function. These problems can be ameliorated by cutting the state space into small boxes, but only at substantial cost.

Schumaker Shape-Preserving Splines

For one-dimensional dynamic programming problems with smooth concave payoff and transition functions and concave C^1 solutions, the Schumaker quadratic shape-preserving spline procedure will produce C^1 approximations of the value function and continuous policy functions. The objective in the maximization step will always be concave and C^1 , allowing us to use a rapid scheme such as Newton's method. We can also use a small number of approximation nodes since we do not need to worry about disruptive internodal fluctuations.

Judd and Solnick (1994) apply Shumaker's quadratic splines to the single good, deterministic optimal growth problem, (12.5.1). They found that the resulting approximations were very good, substantially dominating other methods. For example, the shape-preserving method using 12 nodes did as well as linear interpolation using 120 nodes and the discrete-state approximation using 1,200 points.

Shape-preserving Hermite interpolation is particularly valuable. In this approach, after one computes the value function at a node, one also uses the envelope theorem to compute the slope of the value function at essentially zero computational cost. This slope information can be used along with the level information in the Schumaker shape preserving scheme to arrive at an even better, but still stable, approximation of value function iteration. Examples in Judd and Solnick indicate that shape-preserving Hermite interpolation will produce highly accurate solutions using few approximation nodes.

12.10 Continuous-Time Problems

Value function iteration is an important method for solving discrete-time dynamic programming problems. Unfortunately, the structure of value function iteration is itself tied to the discrete nature of time in those models, and cannot be used for continuous-time problems, since there is no today-tomorrow distinction in continuous time.

One approach is to replace the continuous-time structure with a discrete-time structure with short periods and then use discrete-time methods. Using short periods will make the discount factor, β , close to unity, and imply very slow convergence.

Policy iteration can be implemented in autonomous infinite horizon problems. Suppose that $T = \infty$ in (12.2.1) and the payoff depends only on x and u . The Bell-

man equation for this problem is

$$\rho V(x) = \max_u \pi(x, u) + \sum_{i=1}^n V_{x_i}(x) f^i(x, u). \quad (12.10.1)$$

Suppose that the current guess for the value function is $V^i(x)$. The RHS of (12.10.1) produces the control law, $U^{i+1}(x)$ implied by the value function V^i . Substituting $U^{i+1}(x)$ into (12.2.4) produces the equation

$$\rho V^{i+1}(x) = \pi(x, U^{i+1}(x)) + \sum_{i=1}^n V_{x_i}^{i+1}(x) f^i(x, U^{i+1}(x)) \quad (12.10.2)$$

for the value V^{i+1} of the policy U^{i+1} . Equation (12.10.2) is a linear differential equation in the unknown V^{i+1} . If there is a steady state x^* , then $V(x^*)$ is known, and its value can serve as a boundary condition. The solution of (12.10.2) provides a new guess for the value function. This process produces a sequence of value and policy functions, and would continue until convergence has been reached.

Projection methods are natural for continuous-time problems and could be used to solve (12.10.2). Instead of implementing a policy function iteration, we will proceed in a more direct fashion working simultaneously with the Bellman equation and first-order conditions. We illustrate the idea with the problem stated in (12.2.5). Suppose that $u(c) = -c^{-1}$, $f(k) = 0.2k^{0.25}$, and $\rho = 0.05$, the example studied in table 6.2. The general idea is to parameterize both the value function $V(k)$ and the policy function $C(k)$, and use projection methods to solve the functional equations

$$0 = -\rho V(k) + u(C(k)) + V'(k)(f(k) - C(k)), \quad (12.10.3)$$

$$0 = u'(C(k)) - V'(k). \quad (12.10.4)$$

We approximate V and C with polynomials $\hat{V}(k) = \sum_{l=0}^6 a_l k^l$ and $\hat{C}(k) = \sum_{l=0}^6 b_l k^l$. The steady state is $k^* = 1$; we aim for a solution for $k \in [0.9, 1.1]$. We have fourteen unknown coefficients and two functional equations that we want to solve. We choose seven points, the Chebyshev zeros of T_7 adapted to the interval $[0.9, 1.1]$; these are $X = \{0.9025, 0.9218, 0.9566, 1.0000, 1.0434, 1.0782, 1.0975\}$. We then choose a and b so that both (12.10.1) and (12.10.2) are zero at $k \in X$. We use Newton's method to solve for a and b , which finds a candidate solution. We next need to check if these degree six polynomials are acceptable solutions to the system (12.10.3) and (12.10.4). To do this, we evaluate (12.10.3) and (12.10.4) over the interval $[0.9, 1.1]$. To eliminate units, we divide (12.10.3) by $\rho V(k^*)$, and we divide (12.10.4) by $u'(f(k^*))$. The result had normalized residuals all less than 10^{-8} . Also the policy

function computed here differs by less than one part in 10^{-6} from that computed by reverse shooting applied to the optimal control formulation of this problem. Again we see that projection methods provide a powerful tool for solving a dynamic problem.

12.11 Further Reading and Summary

This chapter presented several approaches to solving finite- and infinite-horizon dynamic programming problems. Finite-horizon problems must use value function iteration. There are many methods available for infinite-horizon autonomous problems. The contraction nature of the basic dynamic programming operator leads us to reliable algorithms. We have seen how this idea can be carried out in value function iteration with both finite- and continuous-state spaces once we adopt appropriate approximation schemes. Christopeit (1983) and Kushner and Dupuis (1992) presents methods for solving stochastic control problems using discretization ideas.

Unfortunately, contraction-based value function iteration schemes are slow. Other methods rely on other properties of dynamic programming. First, policy function iteration results in substantial acceleration. Second, when we focused only on the nonlinear equation structure of the problem, we found that Gauss-Seidel acceleration methods from the nonlinear equation literature could be applied here. Upwind Gauss-Seidel combines the Gauss-Seidel acceleration method with dynamic properties of the dynamic programming problem.

Finite-state problems are easy to understand but time-consuming to solve. Since many problems have continuous states, we presented parametric methods that combine value and policy iteration ideas with approximation ideas. Parametric ideas can greatly improve convergence on simple problems. Johnson et al. (1993) examines the parametric approach for three- and four-dimensional problems. They can also make it possible to attack very large problems. The literature on neurodynamic programming (see Bertsekas 1996) solves very large problems using ideas similar to a combination of the parametric approach and downwind Gauss-Seidel iteration.

Some approaches to solving the linear-quadratic problem ignore the dynamic programming origins of the problem and deal with the Riccati equation as an algebraic problem. As such, the unknown elements of W in (12.6.5) could be fixed using nonlinear equation methods applied to the algebraic Riccati equation. The difficulty with this approach is the existence of extraneous solutions. This makes the choice of initial guess important. However, if one has a good initial guess, then a rapidly convergent nonlinear equation scheme may do better than the value or policy iter-

tion schemes. Amman (1996) explores this idea. Other methods have been developed that exploit the specific algebraic structure of the Riccati equation. Anderson et al. (1996) is a recent review of this literature.

The recent Bertsekas books on dynamic programming contains a good introduction to numerical methods. The Rust (1996) survey focuses on numerical issues and presents the state of the art. Dynamic programming is a key step in some econometric procedures; see Keane and Wolpin (1994), Rust (1996), and Pakes (1996) for reviews of this literature.

Exercises

1. Consider (12.1.17) for $\beta = 0.8$, $u(c, l) = \log c + \log(1 - l)$, and $F(k, l) = k + \beta^{-1}(1 - \beta)k^{0.25}l^{0.75}$. Compute the steady state, k^* . Compute the value and optimal policy functions for $k \in [0.1k^*, 2k^*]$ using discretization methods, piecewise-linear interpolation, Chebyshev polynomial parameterizations (using both Chebyshev regression and Chebyshev interpolation), and Schumaker shape-preserving interpolation. Use both value and policy function methods, and develop Gauss-Seidel approaches. Compare the time and accuracy performance of the various methods. Repeat for $\beta = 0.96$. Then repeat for $\beta = 0.99$.
2. Suppose that a worker earns $w_t = 2 + t/20 - (t/20 - 1)^2$ in period t for $1 \leq t \leq 40$ and is retired for $41 \leq t \leq 55$. Suppose also that there is a safe asset paying interest at rate $r = 0.02$ and a risky asset with price equal to one and dividend equal to $Z \sim U[-0.1, 0.2]$ i.i.d. in each period. Assume a dynamic utility function $E\{\sum_{t=0}^{55} \beta^t u(c)\}$. Write a program to solve this life-cycle asset allocation problem for the cases $\beta = 0.96$ and $u(c) = \log c$, $(c + 0.5)^{0.5}$, and $-e^{-c}$. First, solve for the case where borrowing is not allowed. Second, solve for the case where the borrowing interest rate is $r = 0.1$.
3. Suppose that a monopolist faces a demand curve $D(p)$ and has a cost function of $C(q, Q)$ where q is current output and Q is an exponential lagged sum of past output, $Q_{t+1} = \lambda q_t + (1 - \lambda)Q_t$. Formulate the dynamic programming problem, and compute the value and policy functions for various values of $0 < \lambda < 1$, linear demand curves, and cost curves of the form $C(q, Q) = a + (b - c(Q + 1)^{-0.2})q$ where a, b , and c are positive. Use value iteration, policy iteration, and various gaussian acceleration methods.
4. Solve (12.1.12) with $\beta = 0.95$, $u(c) = \log c$, and $F(k) = k + 0.5(2 + \sin 2\pi k)k^{0.25}$. Discretize k , and use value function iteration, policy function iteration, and upwind Gauss-Seidel approaches to solve for $V(k)$, $k \in [0.1, 2]$.
5. Consider the following multidimensional growth problem. Assume there are two sectors where total output of the single good is

$$y = \sum_{i=1}^2 f^i(k_i, l_i),$$

where k_i, l_i is the capital and labor used in sector i , and $f^i(k_i, l_i)$ is output in sector i of the single good. In each period output is allocated between consumption, c , and net investment in stock i , I_i , where consumption is

$$c = y - \sum_{i=1}^2 I_i - \sum_{i=1}^2 \gamma_i I_i^2$$

and where the γ_i are adjustment cost parameters. The problem reduces to

$$\max \sum_{t=0}^{\infty} \beta^t u\left(c_t, \sum_{i=1}^2 l_{i,t}\right)$$

$$\text{s.t. } k_{i,t+1} = k_{i,t} + I_{i,t},$$

$$c_t = \sum_{i=1}^2 f^i(k_{i,t}, l_{i,t}) - \sum_{i=1}^2 I_{i,t} - \sum_{i=1}^2 \gamma_i I_{i,t}^2.$$

Solve this problem for capital stocks within 50 percent of the steady state for various choices of tastes and technology. First, take a discrete-state approach, using value function iteration, policy function iteration, upwind Gauss-Seidel, and alternative sweep methods. Second, take a parametric approach using regression on the grid consisting of Chebyshev zeros in each dimension. Third, use regression with a grid consisting of a low-discrepancy set of points. Compare the effectiveness of these methods.

6. Suppose that a worker earns a wage w_t that follows a Markov process on a finite set of wages, can earn a safe interest rate of r on savings, cannot borrow, and has the dynamic utility function $\sum_{t=0}^{\infty} \beta^t u(c_t)$ where $\beta(1+r) < 1$ and u is concave. Write a program that will solve the corresponding dynamic programming problem for the case of two wages, $w_t \in \{1, 5\}$, where the chance of a change in wage is 0.1 each period. Compare value function iteration, policy function iteration, and various Gauss-Seidel schemes for discretization schemes. Next try polynomial and spline approximation methods. Compute the ergodic distribution of savings. Next compute the impact of a 25 percent income tax on utility, the present value of tax revenues, and the ergodic distribution of savings.

7. The simplest stochastic growth model with adjustment costs is

$$\max_c E \left\{ \int_0^{\infty} e^{-pt} u(c) dt \right\}$$

$$\text{s.t. } dk = \varphi(f(k) - c) dt + \sigma(k) dz,$$

where φ is a concave function, $\varphi'(0) = 1$. Derive its HJB equation and a computational scheme to compute the optimal consumption policy function.

8. Solve (12.5.7) with values as follow:

- a. $u(c) = 5c - c^2/2$, $L(y) = 1$, $g(I) = \sqrt{I}$, $\beta = 0.9$, and $\theta \sim U[0, 2]$.
- b. $u(c) = 5c - c^2/2$, $L(y) = 1$, $g(I) = 0$, $\beta = 0.9$, and $\theta \sim U[0, 2]$.
- c. $u(c) = 5c - c^2/2$, $L(y) = y^2$, $g(I) = 0$, $\beta = 0.9$, and $\theta \sim U[0, 2]$.

Use both discretization and parametric approaches.

IV

PERTURBATION METHODS

13 Regular Perturbations of Simple Systems

In this chapter and the next two we examine *perturbation*, or *asymptotic*, methods of approximation. The basic idea of asymptotic methods is to formulate a general problem, find a particular case that has a known solution, and then use that particular case and its solution as a starting point for computing approximate solutions to nearby problems. This approach relies on implicit function theorems, Taylor series expansions, and techniques from bifurcation and singularity theory. These methods are widely used in mathematical physics, particularly in quantum mechanics and general relativity theory, with much success.

Economists have often used special versions of perturbation techniques, such as asymptotic methods in statistics and linearizing around a steady state. Unfortunately, many have eschewed rigorous perturbation methods based on mathematics, using instead intuitive, ad hoc procedures. While these ad hoc procedures are sometimes valid, little care is taken to state or check the implicit assumptions, and the ability to apply these ad hoc methods beyond simple problems is limited by the lack of a precise mathematical foundation.

More generally, the economics literature has not exploited the full range and power of asymptotic techniques. The advantage of proceeding formally is that problems that cannot be handled by intuitive informal approaches can be addressed easily within the formal framework. In this and the two following chapters we follow the mathematics literature, discussing when useful the connections to the procedures used in the economics literature.

This chapter presents the elementary structure of regular perturbation methods. We first state the implicit function theorem, which together with Taylor's theorem forms the foundation for regular perturbation methods in Euclidean spaces. We discuss comparative statics problems, the envelope theorem, and comparative dynamics problems, showing that they are all examples of the same regular perturbation ideas.

We then focus on dynamic recursive models, developing notation and methods that produce useful local approximations. We use these techniques to approximate the policy functions of dynamically stable stochastic control models near the steady state of their deterministic counterparts. We use local information to calculate linear and higher-order approximations of the solution to the deterministic problem near the steady state. We then show how to use such local information to approximate the solutions of stochastic problems. In this chapter we focus on systems with a single state variable and a single control variable; the next chapter discusses general systems.

The result of regular perturbation methods is a polynomial or similar function which approximates the true solution in a neighborhood of a special point. There are two reasons why these approximations may be valuable. First, if the expansions are asymptotically valid, they provide us with proofs of a function's local properties; that

is the domain of comparative statics. That is not the main objective of this chapter. Instead, we aim to use these methods instead to construct good *numerical* approximations of *nonlocal* features. Even if the resulting series is not good by itself, the computed approximation may be a valuable input into more standard numerical procedures.

A *warning* to the reader should be made at this point. Many of the perturbation computations below are strictly formal. We do not always discuss the conditions under which these expansions are valid, since such problems are beyond the scope of this book. Instead, we describe diagnostics which check whether a formal expansion does well as an approximate solution, using the compute and verify approach to numerical error discussed in chapter 2. Even if we had all the asymptotic theory we would like, we would still need to use these diagnostics in any application to check if an expansion is reasonably accurate. These diagnostics show that the perturbation-based approximations do far better than one would expect of purely local methods.

13.1 Mathematics of Regular Perturbation Methods

Asymptotic methods depend on a few basic theorems. The critical theorems for regular perturbation theory are Taylor's theorem and the implicit function theorem for R^n . Taylor's theorems were theorems 2.6.1 and 2.6.2. We now state the implicit function theorem.

THEOREM 13.1.1 (Implicit function theorem) If $H(x, y): R^n \times R^m \rightarrow R^m$ is C^k , $H(x_0, y_0) = 0$, and $H_y(x_0, y_0)$ is not singular, then there is a unique C^0 function $h: R^n \rightarrow R^m$ such that $y_0 = h(x_0)$ and for x near x_0 , $H(x, h(x)) = 0$. Furthermore, if H is C^k , then h is C^k , and its derivatives can be computed by implicit differentiation of the identity $H(x, h(x)) = 0$.

Our techniques below will use Taylor's theorem and the implicit function theorem to examine functions implicitly defined by nonlinear equations in R^n . Together they allow us to implicitly compute the derivatives of h with respect to x at x_0 . We will also use similar ideas to examine functions defined implicitly by differential and other functional equations. The formal foundation for those applications relies on generalizing Taylor's theorem and the implicit function theorem to Banach spaces. We will proceed intuitively, since these generalizations are beyond the scope of this text.

Meaning of “Approximation”

We often use the phrase “ $f(x)$ approximates $g(x)$ for x near x_0 ,” but the meaning of this phrase is seldom made clear. One trivial sense of the term is that

$$f(x_0) = g(x_0). \quad (13.1.1)$$

While this is certainly a necessary condition for an approximation, it is generally not by itself a useful concept because it says nothing about f at any x other than x_0 . Approximating one function with another at a point x_0 usually means at least that

$$f'(x_0) = g'(x_0) \quad (13.1.2)$$

holds as well as (13.1.1). In this case we say that “ f is a first-order approximation to g at $x = x_0$,” or that “ f is a linear approximation of g at $x = x_0$.” Note the details: a *linear* approximation *at* x_0 . The less specific nonlocal statement “ f is a linear approximation of g ” has no meaning here; instead, it is more appropriate when referring to the global types of approximation, such as least squares, studied in chapter 6. In general, we say that “ f is an n th order approximation of g at $x = x_0$ ” if

$$\lim_{x \rightarrow x_0} \frac{\|f(x) - g(x)\|}{\|x - x_0\|^n} = 0, \quad (13.1.3)$$

which, for C^n f and g , is true iff $f^{(k)}(x_0) = g^{(k)}(x_0)$ for $k = 0, \dots, n$.

While this seems rather obvious, these definitions are not always used in the economics literature. Some so-called linear approximations don’t even satisfy (13.1.2). In these chapters, we use the mathematical meaning of approximation, and use other terms as needed when discussing ad hoc procedures.

Regular Perturbation: The Basic Idea

The basic idea behind regular perturbations is quite simple. Suppose that a problem reduces to solving

$$f(x, \varepsilon) = 0 \quad (13.1.4)$$

for x where ε is a parameter. We assume that for each value of ε the equation in (13.1.4) has a (possibly not unique) solution for x ; in this way (13.1.4) is a collection of equations in x parameterized by ε . Let $x = x(\varepsilon)$ denote a smooth function such that $f(x(\varepsilon), \varepsilon) = 0$. In general, we cannot solve (13.1.4) for arbitrary ε , but there may be special values of ε for which (13.1.4) can be solved; these will serve as benchmarks.

To see how to proceed, consider the special case of differentiable f , scalar x and ε , $x(\varepsilon)$ unique, and $x(0)$ known. We can apply implicit differentiation to (13.1.4) to find

$$f_x(x(\varepsilon), \varepsilon)x'(\varepsilon) + f_\varepsilon(x(\varepsilon), \varepsilon) = 0, \quad (13.1.5)$$

which implicitly defines the derivative $x'(\varepsilon)$. Equation (13.1.5) is generally useless since it depends on the unknown function $x(\varepsilon)$. However, at $\varepsilon = 0$ (13.1.5) becomes

linear equation in $x'(0)$ with solution $x'(0) = -f_\varepsilon(x(0), 0)/f_x(x(0), 0)$ which is known if $x(0)$ is known. This is well-defined only if $f_x \neq 0$, a condition which can be checked at $x = x(0)$. The linear approximation of $x(\varepsilon)$ for ε near zero is

$$x(\varepsilon) \doteq x^L(\varepsilon) \equiv x(0) - \frac{f_\varepsilon(x(0), 0)}{f_x(x(0), 0)} \varepsilon. \quad (13.1.6)$$

We can continue to differentiate to find higher-order derivatives of $x(\varepsilon)$. We differentiate (13.1.5) to find

$$f_x x'' + f_{xx}(x')^2 + 2f_{xe}x' + f_{ee} = 0. \quad (13.1.7)$$

At $\varepsilon = 0$, (13.1.7) implies that

$$x''(0) = -\frac{f_{xx}(x(0), 0)(x'(0))^2 + 2f_{xe}(x(0), 0)x'(0) + f_{ee}(x(0), 0)}{f_x(x(0), 0)}.$$

With this we can compute the quadratic approximation

$$x(\varepsilon) \doteq x^Q(\varepsilon) \equiv x(0) + \varepsilon x'(0) + \frac{1}{2}\varepsilon^2 x''(0). \quad (13.1.8)$$

We see some patterns developing from these computations of $x'(0)$ and $x''(0)$. The critical equation defining $x''(0)$ is a linear equation, as was the case when we computed $x'(0)$. Also the solution for $x''(0)$ is well-defined as long as $f_x(x(0), 0) \neq 0$, just as was the case for $x'(0)$. Each time we differentiate (13.1.4) we get a new unknown derivative, but its value at $\varepsilon = 0$ is the unique solution to some linear equation. Since the solvability of each such linear problem depends solely on $f_x(x(0), 0)$, if $f_x(x(0), 0) \neq 0$, we can continue this process of constructing a Taylor expansion for $x(\varepsilon)$ as long as $f(x, \varepsilon)$ has the necessary derivatives.

Checking a Perturbation Approximation

Once we have a partial expansion for some function $x(\varepsilon)$ in terms of powers of ε , we would like to know if it is good for any interesting value of ε . Suppose that we are really interested in the case $\varepsilon = 1$. One way to check if the linear approximation $x^L(1)$ is a good approximation for $x(1)$ is to compute the residual $r \equiv f(x^L(1), 1)$; if r is small, then we may decide that $x^L(1)$ is a good approximation to $x(1)$. Of course we must make f and r unit free if this test is to be substantive. We could also do this to evaluate the quality of the quadratic approximation $x^Q(1)$. Even if we could construct a convergent sequence z_k of successively higher-order approximations that converge to $x(1)$, we would still need to check the residual, and stop only when

$f(z_k, 1)$ and/or some other diagnostic test is small. Essentially we should apply the same diagnostic tests to $x^L(1)$ and $x^Q(1)$ as we would to determine when to stop a convergent sequence.

This is all rather obvious for differentiable functions of real variables, but this same idea applies for x 's in general Banach spaces. We will not go into the formalism behind the methods we use below, but the idea is robust. First, express your problem as a continuum of problems parameterized by ε with the $\varepsilon = 0$ case known. Second, “differentiate” the continuum of problems with respect to ε . Third, “solve” the resulting equation for the implicitly defined derivatives at $\varepsilon = 0$. While the differentiation and solution steps are nontrivial problems in general, we will confine our analyses to cases where intuition and basic calculus produces the correct answers.

13.2 Comparative Statics

Although the terms “perturbation” and “asymptotic” may not be familiar, the ideas are common in economics. One form of perturbation analysis familiar to economists is comparative statics. To illustrate the connection between regular perturbation methods and comparative statics, we consider a familiar tax exercise.

Suppose that the demand for apples is $D(p)$ and the supply is $S(p - \tau)$ where p is the price of an apple paid by consumers and τ is the tax per apple paid by producers. The equilibrium equation for p given the tax rate τ is $0 = D(p) - S(p - \tau)$; denote this relation as $p = P(\tau)$. Therefore $P(\tau)$ is defined by the implicit relation

$$0 = D(P(\tau)) - S(P(\tau) - \tau). \quad (13.2.1)$$

We often want to know how a change in the tax rate will affect equilibrium price. To do this, we compute $P'(\tau)$ by differentiating (13.2.1) with respect to τ , yielding

$$0 = D'P' - S'(P' - 1) \Rightarrow P' = \frac{S'}{S' - D'} = \frac{\eta_S}{\eta_S - \eta_D}, \quad (13.2.2)$$

where η_S is the elasticity of supply, pS'/S , and η_D is the elasticity of demand, pD'/D , both of which depend on $p = P(\tau)$. In (13.2.2) we wrote the final answer in terms of elasticities at the equilibrium price given τ ; one should generally write the results of perturbation methods in unit-free terms such as elasticities and shares.

Equation (13.2.2) expresses the local change in price relative to a small change in τ at any τ . In economic applications it is natural to expand $P(\tau)$ around $\tau = 0$, the competitive, tax-free case. This yields the local approximation

$$P(\tau) \doteq P(0) + \frac{\eta_S}{\eta_S - \eta_D} \tau. \quad (13.2.3)$$

We can use this approximation to compute other quantities of interest. Suppose that we want to compute the welfare consequences of the tax. To do so, we first define social surplus

$$SS(\tau) \equiv \int_0^{D(P(\tau))} (D^{-1}(p) - S^{-1}(p)) dp + \tau D(P(\tau)). \quad (13.2.4)$$

We next compute the linear approximation of $SS(\tau)$. Differentiating (13.2.4) implies that $SS'(0) = 0$. Does this imply that there is no welfare loss from a small tax? That would be misleading. To avoid such misleading conclusions, we next compute $SS''(\tau)$ and computing the second-order expansion

$$SS(\tau) \doteq SS(0) + 0 \cdot \tau + \frac{1}{2} D(P(0)) \frac{\eta_D \eta_S}{\eta_S - \eta_D} \tau^2. \quad (13.2.5)$$

This is the common rule-of-thumb that the welfare cost of a tax is proportional to τ^2 . The approximation in (13.2.5) is careful to use prices, quantities, and elasticities instead of derivatives; this is desirable because economic intuition is better expressed in terms of elasticities, which are unit-free, than in terms of derivatives.

Note that the linear term in the expansion of SS is zero and the first nontrivial term, also called the *dominant term*, is τ^2 . While the dominant term is second-order, it is not true that it is negligible. Of course we could define a new variable $\psi = \tau^2$ and then the expansion of SS in terms of ψ begins with a term linear in ψ . On the other hand, we could define $\psi = \sqrt{\tau}$ and find that the first nontrivial term is fourth-order in ψ . Since the notion of n th order is not invariant to these nonlinear changes in variables, we focus instead on the dominant term of an expansion instead of the first-order term. Therefore one should continue the Taylor expansion of a function until a nonzero term is reached. We will use the common language of first-order, second-order, and so on, but it should be understood that any expansion should continue until a nonzero term appears.

Proceeding with successive differentiation of (13.2.1), we can compute several terms of a series expansion of P for τ near 0. It is common to stop at the first nontrivial term, since we generally use these methods to compute qualitative expressions. For numerical purposes it may be useful to compute higher-order terms. In this tax case we could compute arbitrarily high-order terms for $P(\tau)$ as long as $D(p)$ and $S(p)$ have sufficiently many derivatives.

Envelope Theorem

One of the more useful and familiar perturbation results is the *envelope theorem*. We discuss it here because we will frequently invoke it when doing perturbations.

THEOREM 13.2.1 (Envelope theorem): Let

$$\begin{aligned} F(\varepsilon) &= \max_x f(x, \varepsilon) \\ \text{s.t. } g(x, \varepsilon) &= 0 \end{aligned} \tag{13.2.6}$$

define a parameterized collection of problems. Suppose that the parameterized Lagrangian is $\mathcal{L}(x, \lambda, \varepsilon) = f(x, \varepsilon) + \lambda^T g(x, \varepsilon)$, that $x^*(\varepsilon)$ is the solution function for (13.2.6), and that the corresponding shadow price vector is $\lambda^*(\varepsilon)$. Then

$$\frac{dF}{d\varepsilon} = \frac{d}{d\varepsilon} (\mathcal{L}(x^*(\varepsilon), \lambda^*(\varepsilon), \varepsilon)) = \frac{\partial f}{\partial \varepsilon}(x^*(\varepsilon), \varepsilon) + \lambda^*(\varepsilon)^T \frac{\partial g}{\partial \varepsilon}(x^*(\varepsilon), \varepsilon).$$

The basic point of the envelope theorem is that the $\partial x^*/\partial \varepsilon$ term need not be computed to compute $dF/d\varepsilon$.

These simple examples in familiar contexts help us get an idea of how to think about perturbations. While the tax example is simple, it displays exactly how to compute a regular perturbation of a problem. Below we apply these ideas to more complex problems. However, the basic idea remains the same: A regular perturbation is nothing more than computing a derivative of an implicitly defined function.

13.3 Perturbing an IVP

We will often want to perturb dynamic systems. In this section we consider initial value problems; in the next section we consider boundary value problems.

Consider the continuum of IVP's indexed by the parameter ε :

$$\dot{x} \equiv \frac{\partial x(t, \varepsilon)}{\partial t} = g(x, t; \varepsilon), \tag{13.3.1}$$

$$x(0, \varepsilon) = x_0(\varepsilon).$$

For each value of ε (13.3.1) is an ordinary differential equation; in particular, it is an IVP. All terms are assumed to be C^∞ in the parameter ε and we presume that the function $x(t, \varepsilon)$ is smooth in both t and ε ; see Hartman (1964) for a discussion of these issues. Suppose that we can easily solve the $\varepsilon = 0$ problem but not problems where

$\varepsilon \neq 0$. For example, consider the problem

$$\dot{x} = x + \varepsilon h(x), \quad x(0; \varepsilon) = 1. \quad (13.3.2)$$

At $\varepsilon = 0$, (13.3.2) is a linear IVP, and the solution is, $x = e^t$. However, for $\varepsilon \neq 0$ the system is nonlinear, and generally difficult if not impossible to solve analytically.

Perturbation methods study problems of the form in (13.3.1) where $\varepsilon \neq 0$ but is “close enough” to the $\varepsilon = 0$ case, and construct an approximation of the form

$$x(t, \varepsilon) \doteq \sum_{k=0}^n a_k(t) \frac{\varepsilon^k}{k!}, \quad (13.3.3)$$

where $a_k(t) = \partial^k x(t, 0)/\partial \varepsilon^k$. Notice that (13.3.3) is a power series in ε where the coefficients are functions of time.

The perturbation approach constructs conditions on the $a_i(t)$ functions that allow us to solve for them. The first step in constructing (13.3.3) is to compute $a_0(t)$, which is just the solution to (13.3.1) with $\varepsilon = 0$. We assume that this is available. For the example, (13.3.2), the coefficient of ε^0 in (13.3.3) is $a_0(t) = e^t$, since $x(t, 0) = e^t$. The next step is to compute $a_1(t) = (\partial x/\partial \varepsilon)(t, 0)$, which represents the impact on $x(t, \varepsilon)$ as ε increases from $\varepsilon = 0$. To find $(\partial x/\partial \varepsilon)(t, 0)$, we differentiate (13.3.1) with respect to ε and evaluate the result at $\varepsilon = 0$; the result is

$$\dot{a}_1(t) = g_x(a_0(t), t; 0) a_1(t) + g_\varepsilon(a_0(t), t; 0), \quad a_1(0) = x'_0(0). \quad (13.3.4)$$

The key fact is that (13.3.4) is a linear IVP in the unknown function $a_1(t)$ with the initial condition $a_1(0) = x'_0(0)$. The existence theory for linear IVP's applies; in particular, if $|g_x(a(t), t, 0)|$ and $|g_\varepsilon(a_0(t), t, 0)|$ are bounded for $t \in [0, T]$ then a solution exists. Furthermore (13.3.4) has the closed-form solution

$$a_1(t) = a_1(0) e^{\int_0^t g_x(a_0(s), s; 0) ds} + \int_0^t e^{\int_s^t g_x(a_0(z), z; 0) dz} g_\varepsilon(a_0(s), s; 0) ds.$$

Just as the perturbation of a nonlinear equation in R^n produces a linear equation which we can solve, (13.3.4) shows that the perturbation of a nonlinear IVP produces a solvable linear IVP. For the example in (13.3.2), (13.3.4) implies $a_1(0) = x'_0(0)$, $a_0(t) = e^t$, $g_x(x, t; 0) = 1$, $g_\varepsilon(x, t; 0) = h(e^t)$, and the first-order approximation

$$x(t; \varepsilon) \doteq e^t + \varepsilon \left(x'_0(0) e^t + \int_0^t e^{\int_s^t e^z dz} h(e^s) ds \right)$$

which can be evaluated by quadrature for any t .

Taking the second derivative of (13.3.1) with respect to ε and evaluating the result at $\varepsilon = 0$ yields the first-order IVP

$$\dot{a}_2 = g_x a_2 + g_{xx} a_1^2 + g_{ex} a_1 + g_{ee}, \quad a_2(0) = x_0''(0), \quad (13.3.5)$$

where the arguments here of g and its derivatives are the same as in (13.3.4). Since we have solutions for a_0 and a_1 , the only unknown function in (13.3.5) is $a_2(t)$; hence (13.3.5) is again a linear IVP, but now $a_2(t)$ is the unknown function. The initial condition for (13.3.1), $x(0, \varepsilon) = x_0(\varepsilon)$, implies that $a_2(0) = x_0''(0)$, the initial condition in (13.3.5). This IVP then fixes $a_2(t)$. The only assumptions we have used is that g and $x_0(\varepsilon)$ have the derivatives necessary for (13.3.4) and (13.3.5) to be well-defined.

We can apply this to the Solow growth model with taxes. Suppose that the capital stock grows according to $\dot{k} = sR(k)k - \delta k$, where $R(k)$ is the aftertax return on savings, $s > 0$ is the savings rate, and $\delta > 0$ is the depreciation rate. If capital income is taxed at the rate τ , then $R(k) = (1 - \tau)f'(k)$ if $f(k)$ is the aggregate production function. The steady state is defined by $s(1 - \tau)f'(k^*)k^* = \delta k^*$, and is stable and unique if $f(k)$ is concave. The steady state depends on τ ; let $k^*(\tau)$ be the steady state capital stock for tax rate τ .

We examine the effect of a change in τ . To do this, we construct the continuum of problems where τ is now a smooth function, $\tau(\varepsilon)$, of ε

$$\begin{aligned} \dot{k}(t, \varepsilon) &= s(1 - \tau(\varepsilon))kf'(k(t, \varepsilon)) - \delta k(t, \varepsilon), \\ k(0, \varepsilon) &= k^*(\tau(0)). \end{aligned} \quad (13.3.6)$$

If $\varepsilon = 0$, $k(t, 0) = k^*(\tau(0))$ for all t ; that is, k begins and remains at the steady state corresponding to the tax when $\varepsilon = 0$. A change in ε will cause τ to change. Differentiation of (13.3.6) shows

$$\dot{k}_\varepsilon(t, \varepsilon) = -s\tau'(\varepsilon)f'k + s(1 - \tau)f''k_\varepsilon k + (s(1 - \tau)f' - \delta)k_\varepsilon. \quad (13.3.7)$$

At $\varepsilon = 0$, $s(1 - \tau)f' = \delta$ and (13.3.7) can be written as

$$\frac{d}{dt}(k_\varepsilon(t, 0) - k_\varepsilon^\infty) = -\lambda(k_\varepsilon(t, 0) - k_\varepsilon^\infty),$$

where

$$k_\varepsilon^\infty = \frac{\tau'(0)\delta}{s(1 - \tau)^2 f''(k^*)},$$

$$\lambda = -s(1 - \tau)k^*f''(k^*) > 0.$$

This is a stock adjustment form, telling us that $k_\varepsilon(t, 0)$ converges to k_ε^∞ , the change in the steady state k as ε increases, and does so at rate λ . The solution is $k_\varepsilon(t, 0) = (1 - e^{-\lambda t})k_\varepsilon^\infty$.

The use of continuous time here is inessential. The same procedures can be used to perturb discrete-time initial value problems. For example, the discrete-time version of the tax example is $k(t+1) = k(t) + sR(k(t))k(t) - \delta k(t)$, and the steady state is defined by $s(1 - \tau)f'(k)k = \delta k$. We again let $\tau(\varepsilon)$ be a parameterization of the tax rate. The discrete-time version of (13.3.6) is the continuum of difference equations

$$\begin{aligned} k(t+1, \varepsilon) &= k(t, \varepsilon) + s(1 - \tau(\varepsilon))k(t, \varepsilon)f'(k(t, \varepsilon)) - \delta k(t, \varepsilon), \\ k(0, \varepsilon) &= k^*(\tau(0)). \end{aligned} \tag{13.3.8}$$

If $\varepsilon = 0$, $k(t, 0) = k^*(\tau(0))$ for all t ; that is, k remains at the steady state. If we differentiate (13.3.8) with respect to ε and set $\varepsilon = 0$, we obtain a linear difference equation for $k_\varepsilon(t, 0)$, which in turn can be solved by standard means.

13.4 Perturbing a BVP: Comparative Perfect Foresight Dynamics

We next apply these methods to autonomous perturbations of boundary value problems. Suppose that we have a continuum of BVP's

$$\begin{aligned} \dot{x} &= f(x, y, \varepsilon), \\ \dot{y} &= g(x, y, \varepsilon), \\ x(0) &= x_0(\varepsilon), \quad y(T) = y_T(\varepsilon). \end{aligned} \tag{13.4.1}$$

All of the problems in (13.4.1) indexed by ε are autonomous since t never enters explicitly into (13.4.1). We generally construct (13.4.1) so that we know the solution to (13.4.1) for $\varepsilon = 0$, and we want to know the nature of the solution for positive ε . Let $x(t, \varepsilon), y(t, \varepsilon)$ be the solutions for (13.4.1) for $t \in [0, T]$ and small ε . Therefore we assume that the system (13.4.1) has a solution for all ε in some neighborhood of zero and that the solutions are arbitrarily differentiable in ε for small ε .

Since they are smooth functions of t and ε , the dependence of $x(t, \cdot)$ and $y(t, \cdot)$ on ε is linearly approximated at $\varepsilon = 0$ by their derivatives:

$$\begin{aligned} x(t, \varepsilon) &= x(t, 0) + \varepsilon x_\varepsilon(t, 0) + \mathcal{O}(\varepsilon^2), \\ y(t, \varepsilon) &= y(t, 0) + \varepsilon y_\varepsilon(t, 0) + \mathcal{O}(\varepsilon^2). \end{aligned}$$

Therefore we are interested in computing the functions $x_\varepsilon(t, 0)$ and $y_\varepsilon(t, 0)$.

Given the critical role played by $x_\varepsilon(t, 0)$ and $y_\varepsilon(t, 0)$, we need to be clear as to their nature. First, for each t , they are the derivatives of $x(t, \varepsilon)$ ($y(t, \varepsilon)$) with respect to ε when $\varepsilon = 0$. Second, with $\varepsilon = 0$ fixed, we view both $x_\varepsilon(t, 0)$ and $y_\varepsilon(t, 0)$ as functions of t . It is this functional nature that we often exploit when we actually solve for the derivative functions $x_\varepsilon(t, 0)$ and $y_\varepsilon(t, 0)$. In the computations below, we will switch frequently between these two views.

Our smoothness assumptions allow us to differentiate (13.4.1) with respect to ε , which shows us that $x_\varepsilon(t, 0)$ and $y_\varepsilon(t, 0)$ are solutions to the perturbed system

$$\begin{aligned}\dot{x}_\varepsilon &= f_x x_\varepsilon + f_y y_\varepsilon + f_\varepsilon, \\ \dot{y}_\varepsilon &= g_x x_\varepsilon + g_y y_\varepsilon + g_\varepsilon,\end{aligned}\tag{13.4.2}$$

$$x_\varepsilon(0, 0) = \frac{dx_0}{d\varepsilon}(0), \quad y_\varepsilon(T, 0) = \frac{dy_T}{d\varepsilon}(0),$$

where $f_x, f_y, f_\varepsilon, g_x$, and g_y are all evaluated at $(x(t, 0), y(t, 0), 0)$. When we rearrange terms, (13.4.2) becomes the linear BVP

$$\frac{d}{dt} \begin{pmatrix} x_\varepsilon(t, 0) \\ y_\varepsilon(t, 0) \end{pmatrix} = \begin{pmatrix} f_x & f_y \\ g_x & g_y \end{pmatrix} \begin{pmatrix} x_\varepsilon(t, 0) \\ y_\varepsilon(t, 0) \end{pmatrix} + \begin{pmatrix} f_\varepsilon \\ g_\varepsilon \end{pmatrix},\tag{13.4.3}$$

$$x_\varepsilon(0, 0) = \frac{dx_0}{d\varepsilon}(0), \quad y_\varepsilon(T, 0) = \frac{dy_T}{d\varepsilon}(0).$$

Being a linear BVP (with possibly time-varying coefficients) we know that there is a solution to (13.4.3); see chapter 10 for methods to solve (13.4.3) for $x_\varepsilon(t, 0)$ and $y_\varepsilon(t, 0)$. We see that the perturbation of (13.4.1), a nonlinear BVP, produces (13.4.3), a linear BVP, which is a more tractable problem. To compute higher-order approximations of x and y , we compute higher-order derivatives with respect to ε , producing at each stage a linear BVP. In the next section we will solve a broad class of problems encountered in economics.

Nonautonomous Perturbations of a Perfect Foresight Model

The most common use of perturbation analysis arises in the study of perfect foresight economic models, and in the equilibrium response to “shocks” near a steady state. We will now consider nonautonomous perturbations of an autonomous equation. Many infinite-horizon, perfect foresight economic models reduce to the differential equations,

$$\dot{\lambda} = g^1(\lambda, k, \varepsilon h(t)),\tag{13.4.4a}$$

$$\dot{k} = g^2(\lambda, k, \varepsilon h(t)),\tag{13.4.4b}$$

with the boundary conditions,

$$\lim_{t \rightarrow \infty} |k(t)| < \infty, \quad k(0) = k_0(\varepsilon), \quad (13.4.5)$$

where λ, k are economic variables (both taken to be scalars in this example), ε is a scalar parameter, and $h(t)$ is bounded and eventually constant. The boundary conditions in (13.4.5) tell us that k is a predetermined variable and λ is an endogenous variable. Since the perturbation $h(t)$ is assumed to be eventually constant, the system (13.4.4) is autonomous after some point; we assume that the solution converges to the new steady state, telling us how the system behaves asymptotically. We assume that there is a unique solution to (13.4.4) for each ε . For each value of ε we have a different solution; denote the solutions $\lambda(t, \varepsilon)$ and $k(t, \varepsilon)$, making explicit the dependence on t and ε .

We are often interested in the induced change in a dynamic evaluation function

$$W(\varepsilon) = \int_0^\infty e^{-\rho t} v(\lambda(t, \varepsilon), k(t, \varepsilon)) dt,$$

where $v(\lambda, k)$ is the utility or profit flow expressed as a function of λ and k . Therefore we want to know $dW/d\varepsilon$ in the neighborhood of the $\varepsilon = 0$ paths. $W(\varepsilon)$ may represent the present value of profits, utility, or tax revenue, for example, associated with different ε changes. The change in W due to an infinitesimal change in ε is

$$\begin{aligned} \frac{dW}{d\varepsilon} &= \int_0^\infty e^{-\rho t} (v_\lambda \lambda_\varepsilon(t; 0) + v_k k_\varepsilon(t; 0)) dt, \\ &= \begin{pmatrix} v_\lambda(\lambda_0, k_0) \\ v_k(\lambda_0, k_0) \end{pmatrix}^\top \int_0^\infty e^{-\rho t} \begin{pmatrix} \lambda_\varepsilon(t, 0) \\ k_\varepsilon(t, 0) \end{pmatrix} dt, \end{aligned} \quad (13.4.6)$$

where $\lambda_\varepsilon(t, \varepsilon), k_\varepsilon(t, \varepsilon)$ are the partial derivatives of $\lambda(t, \varepsilon)$ and $k(t, \varepsilon)$ with respect to ε . Since we are differentiating the functions at $\varepsilon = 0$, these derivatives are evaluated at $\varepsilon = 0$ in (13.4.6). Note that the final integral in (13.4.6) is actually the Laplace transform of $(\lambda_\varepsilon, k_\varepsilon)^\top$, which we denote by $(\Lambda_\varepsilon(s), K_\varepsilon(s))^\top$, evaluated at $s = \rho$.

Differentiation of the system (13.4.4) with respect to ε yields

$$\frac{d}{dt} \begin{pmatrix} \lambda_\varepsilon \\ k_\varepsilon \end{pmatrix} = J \begin{pmatrix} \lambda_\varepsilon \\ k_\varepsilon \end{pmatrix} + \begin{pmatrix} g_3^1(\lambda_0, k_0, 0) h(t) \\ g_3^2(\lambda_0, k_0, 0) h(t) \end{pmatrix}, \quad (13.4.7)$$

where J is the Jacobian of the vector function $G: R^2 \rightarrow R^2$ evaluated at (λ_0, k_0) where $G(\lambda, k) = (g^1(\lambda, k, 0), g^2(\lambda, k, 0))^\top$. The Laplace transform of (13.4.7) yields an

algebraic equation in the transforms Λ_ε and K_ε ; specifically, at each s we have

$$s \begin{pmatrix} \Lambda_\varepsilon(s) \\ K_\varepsilon(s) \end{pmatrix} = J \begin{pmatrix} \Lambda_\varepsilon(s) \\ K_\varepsilon(s) \end{pmatrix} + \begin{pmatrix} \lambda_\varepsilon(0, 0) + g_3^1(\lambda_0, k_0, 0)H(s) \\ k_\varepsilon(0, 0) + g_3^2(\lambda_0, k_0, 0)H(s) \end{pmatrix}, \quad (13.4.8)$$

where $\lambda_\varepsilon(0, 0)$ is the change in λ at $t = 0$ induced by ε and $H(s)$ is the Laplace transform of $h(t)$. $\lambda_\varepsilon(0, 0)$ is an unknown at this point, but the initial value of k is fixed, and fixed at $k_0(\varepsilon)$. This fact yields an initial condition for (13.4.7), $k_\varepsilon(0, 0) = k'_0(0)$. The basic equation in the transform variables solves easily since for each s it is linear in the unknown $\lambda_\varepsilon(0, 0)$, $\Lambda_\varepsilon(s)$, $K_\varepsilon(s)$. Hence

$$\begin{pmatrix} \Lambda_\varepsilon(s) \\ K_\varepsilon(s) \end{pmatrix} = (sI - J)^{-1} \begin{pmatrix} \lambda_\varepsilon(0, 0) + g_3^1 H(s) \\ k'_0(0) + g_3^2 H(s) \end{pmatrix} \quad (13.4.9)$$

gives the solution for $\Lambda_\varepsilon(s)$ and $K_\varepsilon(s)$ in terms of $\lambda_\varepsilon(0, 0)$.

To pin down $\lambda_\varepsilon(0, 0)$, we need another boundary condition for (13.4.7). In order for this problem to be well-defined, we may need some information on the nature of the linearized system; in particular, to ensure the uniqueness of $\lambda_\varepsilon(0, 0)$, we will assume that the eigenvalues of the linearized system, J , are distinct, real, and of opposite signs. This is the case in the anticipated economic applications, such as the optimal growth example cited above.¹ Let μ be the positive eigenvalue and ξ the negative eigenvalue.

In many applications, it can be proven that $k_\varepsilon(t, 0)$ is bounded; we proceed under this assumption. In this case $K_\varepsilon(s)$ must be finite for all $s > 0$, since $K_\varepsilon(s)$ is $\int_0^\infty e^{-st} k_\varepsilon(t, 0) dt$. In particular, $K_\varepsilon(\mu)$ must be finite for any positive eigenvalue, μ . However, if we try to evaluate (4.9) at $s = \mu$, we have a singularity problem. By the definition of μ being an eigenvalue $\mu I - J$ is a singular matrix. The matrix $(sI - J)^{-1}$ is

$$\frac{\begin{pmatrix} s - J_{22} & J_{12} \\ J_{21} & s - J_{11} \end{pmatrix}}{(s - \mu)(s - \xi)}.$$

In particular, the denominator is zero when $s = \mu$. Therefore the only way for $K_\varepsilon(\mu)$ in (13.4.9) to be finite is for

1. If the eigenvalues do not split in this fashion, we have either too many solution paths or too few, and perturbation methods are inapplicable because the equilibrium either fails to exist or is indeterminate.

$$\begin{pmatrix} \mu - J_{22} & J_{12} \\ J_{21} & \mu - J_{11} \end{pmatrix} \begin{pmatrix} \lambda_\varepsilon(0, 0) + g_3^1 H(\mu) \\ k'_0(0) + g_3^2 H(\mu) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

which implies two conditions for $\lambda_\varepsilon(0, 0)$; however, since μ is an eigenvalue, these conditions are not independent so we have a unique $\lambda_\varepsilon(0, 0)$. Therefore

$$\begin{aligned} \lambda_\varepsilon(0, 0) &= -\frac{(\mu - J_{11})(k'_0(0) + g_3^2 H(\mu))}{J_{21}} - g_3^1 H(\mu) \\ &= -\frac{J_{12}(k'_0(0) + g_3^2 H(\mu))}{\mu - J_{22}} - g_3^1 H(\mu). \end{aligned} \quad (13.4.10)$$

Once we have $\lambda_\varepsilon(0, 0)$, (13.4.7) becomes a nonautonomous linear IVP which can be solved by standard methods, yielding a solution for both $k_\varepsilon(t, 0)$ and $\lambda_\varepsilon(t, 0)$. Substituting (13.4.10) into (13.4.9), and substituting the results into (13.4.6) yields

$$\frac{dW}{d\varepsilon} = \begin{pmatrix} v_\lambda \\ v_k \end{pmatrix}^\top (\rho I - J)^{-1} \begin{pmatrix} g_3^1(H(\rho) - H(\mu)) - \frac{\mu - J_{11}}{J_{21}}(g_3^2 H(\mu) + k'_0(0)) \\ k'_0(0) + g_3^2 H(\rho) \end{pmatrix}. \quad (13.4.11)$$

Application to the Simple Growth Model

We saw in chapter 10 that the simple optimal growth problem, (10.7.2), reduces to solving the BVP

$$\begin{aligned} \dot{c} &= \frac{u'(c)}{u''(c)} (\rho - f'(k)), \\ \dot{k} &= f(k) - c, \\ k(0) &= k_0, \quad 0 < \lim_{t \rightarrow \infty} |k(t)| < \infty. \end{aligned} \quad (13.4.12)$$

We also saw that the steady-state capital stock, k^* , satisfied $f'(k^*) = \rho$, and the steady state consumption, c^* , satisfied $c^* = f(k^*)$. We can express consumption by a policy function, $c(t) = C(k(t))$. Hence $c^* = C(k^*)$. Moreover, since $c(0) = C(k(0))$, we can trace out $C(k)$ by resolving (13.4.12) with different values for $k(0)$.

Suppose that we want to know $C'(k^*)$. $C'(k^*)$ is an important quantity because it tells us how consumption changes as we change the capital stock in the neighborhood of the steady state. We already used a special method to compute $C'(k^*)$ in section 10.7 which led to the solution in (10.7.7). We repeat this problem here to illustrate a

more general approach to determining such derivatives. Consider the continuum of problems,

$$\begin{aligned}\dot{c} &= (\rho - f'(k))u'(c)/u''(c), \\ \dot{k} &= f(k) - c, \\ k_0 &= k^* + \varepsilon, \quad 0 < \lim_{t \rightarrow \infty} |k(t)| < \infty.\end{aligned}\tag{13.4.13}$$

Let $c(t, \varepsilon)$ and $k(t, \varepsilon)$ denote the consumption and capital stocks at time t in problem ε . Since $c(0, \varepsilon) = C(k^* + \varepsilon)$, $C'(k^*) = c_\varepsilon(0, 0)$. Applying the linearization (13.4.2) to (13.4.13) at $k = k^*$ and $c = c^*$ produces the linear system

$$\begin{pmatrix} \dot{c}_\varepsilon \\ \dot{k}_\varepsilon \end{pmatrix} = \begin{pmatrix} 0 & -f''u'(c)/u''(c) \\ -1 & \rho \end{pmatrix} \begin{pmatrix} c_\varepsilon \\ k_\varepsilon \end{pmatrix},$$

$$k_\varepsilon(0, 0) = k'_0(0) = 1.$$

When we apply (13.4.10), we find that $c_\varepsilon(0, 0) = \mu$. Therefore $C'(k^*) = \mu$, the positive eigenvalue which is expressed in (10.7.7). This is a particularly useful application of the perturbation approach, one to which we will return below.

There are many deviations we can examine. For example, we could parameterize the production function as $(1 + \varepsilon h(t))f(k)$. The function $h(t) = 1$ represents an immediate and permanent productivity shock. When $h(t) = 1$, $g^1(c, k, \varepsilon) = (u'(c)/u''(c))(\rho - f'(k)(1 + \varepsilon))$ and $g^2(c, k, \varepsilon) = (1 + \varepsilon)f(k) - c$. Suppose that we are initially in a steady state $\varepsilon = 0$ of the system, namely $k_0 = k^*$ and $c_0 = c^*$ where $g^1(c^*, k^*, 0) = g^2(c^*, k^*, 0) = 0$ define the steady state values, c^* and k^* . Then the solutions are $c(t, 0) = c^*$ and $k(t, 0) = k^*$. Next consider a problem with a small ε . A change in ε away from zero would represent the perturbation of the economy away from the steady state due to an unanticipated immediate output-augmenting change in the production function. The new paths are $c(t, \varepsilon)$ and $k(t, \varepsilon)$. If $h(t)$ were non-constant, then a change in ε represents the introduction of a time-varying change in productivity; if $h(t) = \sin 2\pi t$, we would be modeling predictable, seasonal changes in productivity. In general, the appropriate choice of $h(t)$ and its place in g can represent a large variety of phenomena.

Perturbing Discrete-Time BVP's

As with IVP's, there is no essential difference between continuous and discrete time. The discrete-time optimal growth problem, (12.1.12), leads to the discrete-time BVP

implicitly defined by

$$\begin{aligned} u'(c(t)) &= \beta u'(c(t+1))F'(k(t+1)), \\ k(t+1) &= F(k(t)) - c(t), \\ k_0 = k^*, \quad 0 < \lim_{t \rightarrow \infty} |k(t)| &< \infty. \end{aligned} \tag{13.4.14}$$

This is the discrete-time analogue to (13.4.12). Again we set up a continuum of problems parameterized by ε :

$$\begin{aligned} u'(c(t, \varepsilon)) &= \beta u'(c(t+1, \varepsilon))F'(k(t+1, \varepsilon)), \\ k(t+1, \varepsilon) &= F(k(t, \varepsilon)) - c(t, \varepsilon) \\ k_0 = k(0, \varepsilon) = k^* + \varepsilon, \quad 0 < \lim_{t \rightarrow \infty} |k(t, \varepsilon)| &< \infty. \end{aligned} \tag{13.4.15}$$

At $\varepsilon = 0$ we have the steady state solution $k(t, 0) = k^*$ and $c(t, 0) = F(k^*) - k^*$. Differentiation of (13.4.15) with respect to ε and evaluating the result at $\varepsilon = 0$ produces a linear discrete-time BVP for the unknown discrete-time derivative functions $k_\varepsilon(t, 0)$ and $c_\varepsilon(t, 0)$, which can be solved using methods in section 10.2 or, more generally, methods discussed in the linear rational expectations literature; for a review of such methods, see Anderson et al. (1996).

13.5 Continuous-Time Deterministic Control

Many dynamic economic problems are expressed in dynamic programming terms. We first use basic perturbation ideas to compute a Taylor series approximation for the solution to the canonical single-state single-control continuous-time dynamic programming problem. We then illustrate some perturbation ideas by applying them to a simple growth problem.

The general single-state, single-control continuous-time dynamic programming problem is

$$\begin{aligned} V(x_0) &= \max_u \int_0^\infty e^{-rt} \pi(x, u) dt \\ \text{s.t. } \dot{x} &= f(x, u), \\ x(0) &= x_0. \end{aligned} \tag{13.5.1}$$

In chapter 12 we saw that this dynamic programming problem implies the Bellman

system for $U(x)$, the optimal control, and $V(x)$:

$$rV(x) = \pi(x, U(x)) + V'(x)f(x, U(x)), \quad (13.5.2a)$$

$$0 = \pi_u(x, U(x)) + V'(x)f_u(x, U(x)). \quad (13.5.2b)$$

Differentiation of (13.5.2a) with respect to x and application of (13.5.2b) yields

$$rV'(x) = \pi_x(x, U(x)) + V''(x)f(x, U(x)) + V'(x)f_x(x, U(x)). \quad (13.5.3)$$

Suppose that there is a steady state to the problem, that is, a value x^* such that $\dot{x} = f(x^*, U(x^*)) = 0$. A steady state is a solution to the nonlinear system

$$0 = f(x^*, u^*),$$

$$0 = \pi_u(x^*, u^*) + V'(x^*)f_u(x^*, u^*), \quad (13.5.4)$$

$$rV'(x^*) = \pi_x(x^*, u^*) + V'(x^*)f_x(x^*, u^*),$$

where the three unknowns are u^* , x^* , and $V'(x^*)$. We assume that (13.5.4) has a solution.

To proceed with the perturbation analysis below, we need to assume differentiability and stability of the problem its solution. We further assume that $V(x)$ and $U(x)$ are C^∞ in some neighborhood of x^* and that the solution $U(x)$ causes $\dot{x} = f(x, U(x))$ to be locally stable near the steady state x^* . These differentiability assumptions are clearly excessive but not unrealistic, since many applications assume that π and f are C^∞ . Also we are assuming only local smoothness and stability. If these assumptions are significantly violated, our computations will likely be able to detect that. For example, local stability can be verified by examining the eigenvalues of the linearized system. We want the result to be a good solution over a nontrivial neighborhood of the steady state; if that is not true, the approximation will likely fail the diagnostics we discuss in section 13.9. We now proceed with the formal expansion.

With a steady state in hand, we aim to construct a local approximation to $V(x)$ and $U(x)$ near x^* . That is, we aim to compute the Taylor series expansions

$$\begin{aligned} V(x) &= V(x^*) + V'(x^*)(x - x^*) + \frac{1}{2}V''(x^*)(x - x^*)^2 + \dots, \\ U(x) &= U(x^*) + U'(x^*)(x - x^*) + \frac{1}{2}U''(x^*)(x - x^*)^2 + \dots. \end{aligned} \quad (13.5.5)$$

We have x^* , $u^* = U(x^*)$, and $V'(x^*)$ from (13.5.4). We next move to calculate $U'(x^*)$ and $V''(x^*)$. This will give us a linear approximation to the policy function,

$U(x)$, near x^* . To do this we differentiate (13.5.3) and (13.5.2b) with respect to x to form

$$rV'' = \pi_{xx} + \pi_{ux}U' + V'''f + V''(2f_x + f_u U') + V'(f_{xx} + f_{ux}U'), \quad (13.5.6a)$$

$$0 = \pi_{ux} + \pi_{uu}U' + V''f_u + V(f_{ux} + f_{uu}U'). \quad (13.5.6b)$$

The system (13.5.6) holds at all x ; since we don't know $V(x)$ nor $U(x)$ for $x \neq x^*$, (13.5.6) cannot tell us anything about their derivatives. However, we need only $U'(x^*)$ and $V'(x^*)$ to construct (13.5.5). To determine these values, we next let $x = x^*$ and impose the steady state conditions on (13.5.6). The result is a quadratic system of two equations,

$$\begin{aligned} rV'' &= \pi_{xx} + \pi_{ux}U' + V''(2f_x + f_u U') + V'(f_{xx} + f_{ux}U'), \\ 0 &= \pi_{ux} + \pi_{uu}U' + V''f_u + V'(f_{ux} + f_{uu}U'). \end{aligned} \quad (13.5.7)$$

where now the arguments of all functions in (13.5.7) are the steady state values of u and x , (u^*, x^*) .

The key fact is that in the steady state system (13.5.7), we know x^* , $U(x^*)$, $V'(x^*)$, π and its derivatives, and f and its derivatives. This leaves only two unknowns, $U'(x^*)$ and $V''(x^*)$. The system (13.5.7) will have two solution pairs, but only one will make economic sense. In particular, the maximum principle implies not only the first-order condition (13.5.2b), but it also implies that V is concave at x^* . Hence $V''(x^*) < 0$. This concavity condition will often determine the desired solution to (13.5.7). If both solutions to (13.5.7) are consistent with $V''(x^*) < 0$, we cannot continue because that indicates indeterminacy in the underlying problem. We proceed from here under the assumption that there is a unique solution to (13.5.7) with $V''(x^*) < 0$.

At this point we should note that we have another way to compute $U'(x^*)$. In example (13.4.13) we showed how to compute the change in the control as we move the state away from its steady state value; that change is $U'(x^*)$. We could use the result of this BVP approach to compute $U'(x^*)$ and then use (13.5.6a) to compute $V''(x^*)$. This is in fact a little easier because we get $U'(x^*)$ directly when we linearize the corresponding optimal control problem and solve linear differential equations. In contrast, there are multiple solutions in (13.5.7), a quadratic system. In general, it is easier to compute $U'(x^*)$ as in (13.4.13) instead of analyzing the quadratic equation in (13.5.7). The gain here is slight but will be much more important in the multi-dimensional case.

Even though the dynamic programming formulation is not the best way to proceed for the linearization problem, it is advantageous for producing higher-order terms. We now turn to the necessary computations for those higher-order terms. With $U'(x^*)$ and $V''(x^*)$ in hand, we continue this process to get higher-order terms of the Taylor expansion, (13.5.5). Differentiating (13.5.6) with respect to x produces

$$\begin{aligned} 0 = & V'''f + V'''(f_x - r + 3(f_x + f_u U')) + U''(\pi_{ux} + 2V''f_u + f_{ux}) \\ & + \pi_{xxx} + 2\pi_{xxx}U' + V''(3f_{xx} + 3f_{xu}U' + f_{uu}U'U') \\ & + V'(f_{xxx} + f_{xu}U' + r_{xux}U' + f_{xuu}U'U'), \end{aligned} \quad (13.5.8)$$

$$\begin{aligned} 0 = & V'''f_u + U''(\pi_{uu} + V'f_{uu}) + \pi_{uxx} + 2\pi_{uux}U' + V''(2f_{uu}U' + 2f_{ux}) \\ & + V'(f_{uxx} + 2f_{uxu}U' + f_{uuu}U'U'). \end{aligned}$$

Again, (13.5.8) is true everywhere. Since we don't know U and V everywhere, (13.5.8) is not useful for $x \neq x^*$. However, at the steady state (13.5.8) implies the linear system

$$\begin{pmatrix} f_x - r + 3(f_x + f_u U') & \pi_{ux} + 2V''f_u + V'f_{ux} \\ f_u & \pi_{uu} + V'f_{uu} \end{pmatrix} \begin{pmatrix} V''' \\ U'' \end{pmatrix} = \begin{pmatrix} A \\ B \end{pmatrix}, \quad (13.5.9)$$

where A and B involve neither $U''(x^*)$ nor $V'''(x^*)$. The matrix in (13.5.9) is a known real matrix at the steady state since $U'(x^*)$ and $V''(x^*)$ are known at this point as is the RHS of (13.5.9). Therefore (13.5.9) implies a unique solution for $U''(x^*)$ and $V'''(x^*)$ as long as the matrix is nonsingular, a condition that can be checked directly in any application.

Note what we have computed. Since we now have the first two derivatives of U at the steady state $x = x^*$, we can use them to compute a quadratic approximation of $U(x)$ based at x^* , which consists of the terms expressed in the second line of (13.5.5). Since we have the first three derivatives of V at $x = x^*$, we can use them to compute a cubic approximation of $V(x)$ based at x^* . We have therefore gone one degree beyond the result of linearizing the system near the steady state.

These manipulations can be continued to get even higher-order terms and poses several questions. Can we replicate this for multidimensional problems to construct multivariate Taylor expansions of policy and value functions? Can this procedure be completely mechanized? How valuable are the further expansions? Is this power series approximation approach an alternative to the numerical techniques explored

before? We will address some of these issues below, but first we consider a simple problem where we can see clearly how to compute an arbitrarily long expansion.

Application to the Single Sector Growth Model

To demonstrate these perturbation methods, we will again look at the single-sector, single-good, continuous-time optimal growth problem

$$\begin{aligned} \max_c & \int_0^\infty e^{-\rho t} u(c) dt \\ \text{s.t. } & \dot{k} = f(k) - c, \\ & k(0) = k_0. \end{aligned} \tag{13.5.10}$$

We now take a dynamic programming view instead of the optimal control perspective of (13.4.12). In (13.4.12), we focused on perturbations to the time path of capital and consumption. We will here compute approximations for the policy function.

To simplify the presentation, we will assume that $u(c) = c^{\gamma+1}/(\gamma+1)$. Equation (10.7.5) shows that $C(k)$, the optimal policy function for the solution to (13.5.10), satisfies the differential equation

$$\gamma C'(k)(f(k) - C(k)) - C(k)(\rho - f'(k)) = 0. \tag{13.5.11}$$

We will use (13.5.11) to compute a Taylor series expansion for $C(k)$. Differentiation of (13.5.11) with respect to k yields

$$\gamma C''(f - C) + \gamma C'(f' - C') - C'(\rho - f') + Cf'' = 0 \tag{13.5.12}$$

Equation (13.5.12) holds at all k ; at $k = k^*$ we have $\gamma C'(f' - C') + Cf'' = 0$, which is a quadratic equation in the only unknown, $C'(k^*)$, and which implies that

$$C'(k^*) = \frac{\rho}{2} \left(1 \pm \sqrt{1 + \frac{4f''f}{\gamma f'f'}} \right), \tag{13.5.13}$$

where we use the fact that $\rho = f'(k^*)$ and $f(k^*) = C(k^*)$. Since $\gamma < 0$, (13.5.13) has two solutions, but one is negative. Since we know $C' > 0$, we choose the positive root.

We can do this for an arbitrary number of derivatives. If we take (13.5.11) and differentiate it m times with respect to k we have for all k the identity

$$\begin{aligned}
0 &= \gamma \frac{d^m}{dk^m} (C'(f - C)) - \frac{d^m}{dk^m} (C(\rho - f')) \\
&= \gamma \left(C'(f^{(m)} - C^{(m)}) + \sum_{j=1}^{m-2} \frac{m!}{j!(m-j)!} C^{(j+1)}(f^{(m-j)} - C^{(m-j)}) \right. \\
&\quad \left. + mC^{(m)}(f' - C') + C^{(m+1)}(f - C) \right) \\
&\quad - \left(\sum_{j=0}^{m-1} \frac{m!}{j!(m-j)!} C^{(j)}(-f^{(m-j+1)}) + C^{(m)}(\rho - f') \right). \tag{13.5.14}
\end{aligned}$$

At the steady state k^* , $f(k^*) = C(k^*)$ and $\rho = f'(k^*)$; therefore at the steady state (13.5.14) reduces to the equality

$$\begin{aligned}
0 &= \gamma \left(C'(f^{(m)} - C^{(m)}) + \sum_{j=1}^{m-2} \frac{m!}{j!(m-j)!} C^{(j+1)}(f^{(m-j)} - C^{(m-j)}) \right. \\
&\quad \left. + mC^{(m)}(f' - C') \right) - \left(\sum_{j=0}^{m-1} \frac{m!}{j!(m-j)!} C^{(j)}(-f^{(m-j+1)}) \right), \tag{13.5.15}
\end{aligned}$$

where now f and C are evaluated at $k = k^*$. Equation (13.5.15) implies that $C^{(m)}(k^*)$ is

$$\begin{aligned}
C^{(m)}(k^*) &= \left(\gamma \left(C'(k^*)f^{(m)}(k^*) + \sum_{j=1}^{m-2} \frac{m!}{j!(m-j)!} C^{(j+1)}(k^*)(f^{(m-j)}(k^*) - C^{(m-j)}(k^*)) \right) \right. \\
&\quad \left. - \sum_{j=0}^{m-1} \frac{m!}{j!(m-j)!} C^{(j)}(k^*)(-f^{(m-j+1)}(k^*)) \right) \\
&\quad \times (\gamma(C'(k^*) - m(f'(k^*) - C'(k^*))))^{-1}. \tag{13.5.6}
\end{aligned}$$

If $C(k) = \sum_{n=0}^{\infty} (c_n/n!)(k - k^*)^n$, and $f(k) = \sum_{n=0}^{\infty} (f_n/n!)(k - k^*)^n$, are Taylor series expansions, then (13.5.16) implies the iterative scheme

$$\begin{aligned}
c_m &= \left(\gamma \left(c_1 f_m + \sum_{j=1}^{m-2} \frac{m!}{j!(m-j)!} c_{j+1}(f_{m-j} - c_{m-j}) \right) \right. \\
&\quad \left. - \sum_{j=0}^{m-1} \frac{m!}{j!(m-j)!} c_j(-f_{m-j+1}) \right) (\gamma(c_1 - m(\rho - c_1)))^{-1}. \tag{13.5.17}
\end{aligned}$$

Table 13.1
Errors of the (13.5.17) solution to (13.5.10)

k	c	E_{10}	E_{120}
0.01	0.03782	2.0(−2)	2.2(−3)
0.05	0.06288	6.7(−3)	1.3(−3)
0.20	0.10272	3.0(−4)	2.2(−4)
0.50	0.14780	1.1(−5)	1.2(−5)
0.80	0.18092	1.7(−6)	1.7(−7)
0.90	0.19069	8.8(−9)	8.8(−9)
0.98	0.19817	1.2(−11)	1.2(−11)
1.00	0.20000	0	0
1.02	0.20182	1.2(−11)	1.8(−11)
1.10	0.20893	6.5(−9)	6.5(−9)
1.20	0.21754	9.1(−8)	9.1(−8)
1.40	0.23394	1.2(−6)	1.1(−6)
2.00	0.27840	9.1(−4)	6.3(−4)
2.09	0.28461	2.2(−3)	1.2(0)

These calculations compute the Taylor series for $C(k)$. The series defined in (13.5.17) may have poor convergence properties. In particular, it may have a singularity at $k = 0$ because $f(k)$ has infinite derivatives at $k = 0$. Since $C(k)$ is related to $f(k)$, it would not be surprising if the Taylor series for $C(k)$ were similarly behaved.

We illustrate the accuracy of the Taylor series defined in (13.5.7) by comparing them to the approximation from reverse shooting. Table 13.1 considers the case of $f(k) = 0.2k^{0.25}$, $u(c) = c^{-1}$, and $\rho = 0.05$. The columns labeled E_{10} and E_{120} display the magnitude of the errors of the degree 10 and degree 120 Taylor series approximations of $C(k)$. The “truth” is taken to be the result of a reverse shooting computation of $C(k)$ using RK4 and a step size of 10^{-5} . Both do very well for a large range of k , but the degree 120 expansion does about ten times better for k near 0. For $k > 2$ both expansions break down rapidly. This is expected, since the singularity in f at $k = 0$ makes it unlikely that any meaningful expansion around $k = 1$ has a radius of convergence greater than 1.

The ability to compute such high-order expansions is unusual, and one would seldom compute them even when possible. While it is an exaggeration, this example does illustrate the general structure of perturbation approaches. It also highlights the fact that the high-order terms can be computed, and that they can substantially improve the accuracy of the approximation. The results in table 13.1 also show that the approximation is far from being just locally good.

General Perturbations

So far we have based our approximation on already knowing the solution at a particular point, the steady state. Not all problems have such points. The perturbation idea may still be valuable if we have some examples where we know the entire solution and use these to approximate solutions to “nearby” problems where we initially have no information.

The general idea really is the same when we think in terms of general spaces. Suppose that we are trying to solve a functional equation, $\mathcal{F}(f) = 0$, for f in a function space. Suppose that we know of a similar functional equation \mathcal{G} , where we know the solution g^* such that $\mathcal{G}(g^*) = 0$. We then create a parameterized system, $\mathcal{H}(h, \varepsilon) = \varepsilon\mathcal{F}(h) + (1 - \varepsilon)\mathcal{G}(h)$. Let $h(\varepsilon)$ be implicitly defined by $\mathcal{H}(h(\varepsilon), \varepsilon) = 0$. In this form, forming the parameterized continuum of problems, is similar to the homotopy method discussed in chapter 5. To approximate a solution to \mathcal{G} , we use the known solution g^* to the equation $\mathcal{H}(\cdot, 0) = 0$ and then perturb $\mathcal{H}(h(\varepsilon), \varepsilon) = 0$ around $h(0) = g^*$ to compute $h_\varepsilon(0)$, $h_{\varepsilon\varepsilon}(0)$, ... and form an approximation of $h(1) \doteq h(0) + h_\varepsilon(0) + h_{\varepsilon\varepsilon}(0) + \dots$ which approximately solves $\mathcal{F}(f) = 0$. This all sounds very abstract but can be illustrated by some simple perturbations of (13.5.11).

Again suppose we are trying to solve (13.5.11). To pin down the equation, we impose the “boundary condition” $C(k^*) = f(k^*)$; that is, we assume that for any choice of utility and production functions, $C(k^*) = f(k^*)$ where k^* is defined by $\rho = f'(k^*)$. Let ε parameterize tastes and/or technology so that we can examine the continuum of problems $\gamma(\varepsilon)$; for example, we assume that the utility function is $u(c) = c^{1+\gamma(\varepsilon)} / (1 + \gamma(\varepsilon))$ and that output is $y = f(k; \varepsilon)$. Equation (13.5.11) then implies the continuum of problems

$$C_k(k, \varepsilon)(f(k, \varepsilon) - C(k, \varepsilon)) + \gamma(\varepsilon)^{-1}C(k, \varepsilon)(f_k(k, \varepsilon) - \rho) = 0 \quad (13.5.18)$$

with the boundary conditions $C(k^*(\varepsilon), \varepsilon) = f(k^*(\varepsilon), \varepsilon)$. The steady state condition $\rho = f_k(k^*(\varepsilon), \varepsilon)$ defines the function $k^*(\varepsilon)$ which expresses the steady state for each ε . As long as we know the solution to the $\varepsilon = 0$ case, we can attempt to construct an approximation of $C(k, \varepsilon)$ of the form

$$\sum_{i=0}^n \frac{\varepsilon^i}{i!} \frac{\partial^i C}{\partial \varepsilon^i}(k, 0).$$

There are several instances where we know the solution to (13.5.18). Suppose that we choose $\varepsilon \equiv -\gamma^{-1}$ to be the perturbation parameter, the differential equation, (13.5.18), for the consumption policy function, $C(k, \varepsilon)$, reduces to

$$C_k(k, \varepsilon)(f(k) - C(k, \varepsilon)) - \varepsilon C(k, \varepsilon)(f'(k) - \rho) = 0. \quad (13.5.19)$$

When $\varepsilon = 0$, the utility function displays “infinite” curvature and the solution is $C(k, 0) = f(k)$. If we differentiate (13.5.19) with respect to ε , we can compute $C_\varepsilon(k, 0)$, which expresses how $C(k, \varepsilon)$ deviates from $f(k)$ as curvature retreats from infinity. Differentiation of (13.5.19) with respect to ε implies that

$$C_{k\varepsilon}(f - C) + C_k(-C_\varepsilon) - C(f' - \rho) - \varepsilon C_\varepsilon(f' - \rho) = 0. \quad (13.5.20)$$

At $\varepsilon = 0$, $C(k, 0) = f(k)$ and $C_k(k, 0) = f'(k)$. Therefore at all k ,

$$C_\varepsilon(k, 0) = f\left(\frac{\rho}{f'} - 1\right) \quad (13.5.21)$$

Further differentiation will yield $C_{\varepsilon\varepsilon}(k, 0)$, $C_{\varepsilon\varepsilon\varepsilon}(k, 0)$, and so on.

For $f(k) = \rho k^\alpha / \alpha$, we have

$$\begin{aligned} C_{\varepsilon\varepsilon}(k) &= \alpha^{-2} 2(\alpha - 1) \rho k^{1-\alpha} (k - k^\alpha), \\ C_{\varepsilon\varepsilon\varepsilon}(k) &= \alpha^{-3} 6(\alpha - 1) \rho k^{1-2\alpha} (k - k^\alpha) (3k - 2\alpha k - k^\alpha). \end{aligned} \quad (13.5.22)$$

We should emphasize the fact that all of these manipulations are formal. We have not presented the mathematical foundation necessary to prove that we are constructing asymptotically valid approximations. This is particularly apparent in the last case where any $\varepsilon < 0$ case implies a convex utility function.

Another example of (13.5.18) begins by noting that if $f(k) = \rho k$, then $C(k) = f(k) = \rho k$. This is the degenerate case where the marginal product of capital is constant and equals the constant pure rate of time preference. Suppose that $f(k)$ is instead a concave production function and $u'(c) = c^\gamma$. Consider the continuum of problems indexed by $\varepsilon \in [0, 1]$:

$$C_k(k, \varepsilon)[(1 - \varepsilon)\rho k + \varepsilon f(k) - C(k, \varepsilon)] + \gamma^{-1} C(k, \varepsilon)(\rho(1 - \varepsilon) + \varepsilon f' - \rho) = 0. \quad (13.5.23)$$

At $\varepsilon = 0$ we have the linear production function ρk and $C(k, 0) = \rho k$. At $\varepsilon = 1$ we have the general production function $f(k)$. By computing $C(k, 0)$, $C_\varepsilon(k, 0)$, $C_{\varepsilon\varepsilon}(k, 0)$, etc., we are computing the change in the consumption function as the production function evolves from the special case of ρk to $f(k)$. Straightforward calculations show that for this perturbation problem

$$C_\varepsilon(k, 0) = \gamma^{-1} (f' - \rho) k + f - \rho k \quad (13.5.24)$$

Further differentiation will yield higher-order derivatives.

A third example of (13.5.18) can be based on Chang's (1988) result that when $f(k) = \rho k^\alpha / \alpha$, $u(c) = c^{1+\gamma} / (1 + \gamma)$, and $\gamma = -\alpha$, equation (13.5.11) has the solution

$C(k) = \theta k$, where $\theta = -\rho/\gamma = \rho/\alpha$. This is another kind of special case around which we can perturb the problem. Suppose that we have $\gamma(\varepsilon) = \varepsilon - \alpha$ and want to find an approximate solution for γ near $-\alpha$. The system (13.5.18) becomes

$$(-\alpha + \varepsilon)C_k(k, \varepsilon)(f(k) - C(k, \varepsilon)) + C(k, \varepsilon)(f'(k) - \rho) = 0. \quad (13.5.25)$$

Perturbing (13.5.25) at $\varepsilon = 0$ yields

$$C_k(f - C) - \alpha(C_{k\varepsilon}(f - C) - C_k C_\varepsilon) + C_\varepsilon(f' - \rho) = 0, \quad (13.5.26)$$

which is the linear ordinary differential equation

$$\alpha C_{k\varepsilon}(f - C) - C_\varepsilon(\alpha C_k + (f' - \rho)) = C_k(f - C) \quad (13.5.27)$$

in the unknown function $C_\varepsilon(k, 0)$. Furthermore we know that $C(k, 0) = \theta k$ and $f(k) = \rho k^\alpha/\alpha$; this reduces (13.5.27) to

$$\psi'(x)(\rho k^\alpha - \rho k) - \psi(k)\rho k^{\alpha-1} = \rho^2 \alpha^{-2}(k^\alpha - k), \quad (13.5.28)$$

where $\psi(k) = C_\varepsilon(k, 0)$. We also know that the steady state is unaffected by such changes; hence we have the boundary condition $\psi(k^*) = C_\varepsilon(k^*, \varepsilon) = 0$ for all ε . This boundary condition guarantees a bounded solution for the linear differential equation in (13.5.28).

This perturbation is much more complex than the previous two but more typical of the outcome of perturbing a dynamic system. It corresponds with common sense: Differentiating a nonlinear system of real equations generally produces a linear set of real equations, and differentiating a system of nonlinear differential equations generally produces a system of linear differential equations. The first examples in this section were unusual in that perturbing nonlinear differential equations produced algebraic equations that can be viewed as degenerate linear differential equations. The linear differential equation (13.5.28) may itself be difficult to solve. However, it is a linear problem, and reliable numerical methods are available even if there is no closed-form solution. There is no guarantee that the problems which must be solved as part of a regular perturbation procedure are analytically soluble; they just happen to be in our simple examples. However, these problems will be more tractable than the original nonlinear problem being approximated, since the perturbation equations are generally linear.

13.6 Stochastic Control

We next add uncertainty to the infinite-horizon, single-state, single-control problem. The canonical problem is

$$\begin{aligned}
 V(x_0) &= \max_u E \left\{ \int_0^\infty e^{-pt} \pi(x, u) dt \mid x(0) = x_0 \right\} \\
 \text{s.t. } &dx = f(x, u) dt + \sqrt{2\varepsilon\sigma(x, u)} dz, \\
 &x(0) = x_0.
 \end{aligned} \tag{13.6.1}$$

While dependence of (13.6.1) on ε is normally suppressed in the notation, we will add ε as a parameter and express the value function as $V(x, \varepsilon)$, emphasizing the fact that we are using the $\varepsilon = 0$ case as the basis of our approximation. If $V(x, \varepsilon)$ is C^2 in x , then stochastic optimal control theory (see equation 12.2.13) implies that for each ε , the value function $V(x, \varepsilon)$ solves the Bellman equation:

$$\rho V(x, \varepsilon) = \max_u \pi(x, u) + V_x(x, \varepsilon)f(x, u) + \varepsilon\sigma(x, u)V_{xx}(x, \varepsilon). \tag{13.6.2}$$

Proceeding as in the deterministic case, we let $u = U(x, \varepsilon)$ denote the control rule and find that (13.6.2) implies the system

$$\rho V = \pi + V_x f + \varepsilon\sigma V_{xx}, \tag{13.6.3}$$

$$0 = \pi_u + V_x f_u + \varepsilon\sigma_u V_{xx}. \tag{13.6.4}$$

Of course we drop arguments of functions whenever they can be understood from context.

Formally we are again looking for the terms of the Taylor expansions of C and V , which are here expressed as

$$\begin{aligned}
 U(x, \varepsilon) &\doteq U(x^*, 0) + U_x(x^*, 0)(x - x^*) + U_\varepsilon(x^*, 0)\varepsilon \\
 &\quad + \frac{U_{xx}(x^*, 0)(x - x^*)^2}{2} + U_{ex}(x^*, 0)\varepsilon(x - x^*) + \frac{U_{e\varepsilon}(x^*, 0)\varepsilon^2}{2} + \dots,
 \end{aligned} \tag{13.6.5}$$

$$\begin{aligned}
 V(x, \varepsilon) &\doteq V(x^*, 0) + V_x(x^*, 0)(x - x^*) + V_\varepsilon(x^*, 0)\varepsilon \\
 &\quad + \frac{V_{xx}(x^*, 0)(x - x^*)^2}{2} + V_{ex}(x^*, 0)\varepsilon(x - x^*) + \frac{V_{e\varepsilon}(x^*, 0)\varepsilon^2}{2} + \dots.
 \end{aligned} \tag{13.6.6}$$

The validity of these simple methods in this case is surprising. Equations (13.6.3)–(13.6.3)–(13.6.4) are second-order differential equations when $\varepsilon \neq 0$, but they degenerate to first-order differential equations when $\varepsilon = 0$. Changing ε from zero to a nonzero value is said to induce a *singular perturbation* in the problem because of this change of order. Normally much more subtle and sophisticated techniques are required to use the $\varepsilon = 0$ case as a basis of approximation for nonzero ε . The

remarkable feature of stochastic control problems, proved by Fleming (1971), is that perturbations of ε can be analyzed as a regular perturbation in ε . Furthermore we must emphasize the fact that $\varepsilon\sigma$ is the instantaneous variance, not the standard deviation. While it may appear that the perturbation is really second order, recall that risk premia are locally linear in variance. A simple regular perturbation analysis may have tried to derive a series in terms of the standard deviation. The important result of Fleming is that for this problem we can proceed in a regular perturbation fashion, but in the variance parameter.

At $\varepsilon = 0$ we have the deterministic solution. Our task now is to determine what happens as ε increases. To do this we differentiate (13.6.3)–(13.6.4) with respect to ε , and apply (13.6.4). This produces

$$rV_\varepsilon = V_{x\varepsilon}f + \sigma V_{xx} + \varepsilon\sigma V_{xx\varepsilon}, \quad (13.6.7)$$

$$0 = \pi_{uu}U_\varepsilon + V_{x\varepsilon}f_u + V_xf_{uu}U_\varepsilon + \sigma_u V_{xx} + \varepsilon\sigma_{uu}U_\varepsilon V_{xx} + \varepsilon\sigma_u V_{xx\varepsilon}. \quad (13.6.8)$$

When we impose $\varepsilon = 0$ and the steady state conditions for the deterministic problem, (13.6.7)–(13.6.8) reduce to

$$rV_\varepsilon = \sigma V_{xx}, \quad (13.6.9)$$

$$0 = \pi_{uu}U_\varepsilon + V_{x\varepsilon}f_u + V_xf_{uu}U_\varepsilon + \sigma_u V_{xx}, \quad (13.6.10)$$

at $\varepsilon = 0$ and $x = x^*$. This is a linear system in the unknowns V_ε and U_ε . Equation (13.6.9) implies that

$$V_\varepsilon = \frac{\sigma}{r} V_{xx}. \quad (13.6.11)$$

This is an intuitive result because it shows that the utility loss due to uncertainty is proportional to the variance and to the curvature of the value function.

To solve for U_ε and $V_{x\varepsilon}$, we differentiate (13.6.7) with respect to x , yielding

$$\begin{aligned} rV_{x\varepsilon} &= V_{xx\varepsilon}f + V_{x\varepsilon}(f_x + f_u U_x) + (\sigma_x + \sigma_u U_x)V_{xx} + \sigma V_{xxx} \\ &\quad + \varepsilon(\sigma_x + \sigma_u U_x)V_{xxx} + \varepsilon\sigma V_{xx\varepsilon} \end{aligned} \quad (13.6.12)$$

which at $\varepsilon = 0$ and $x = x^*$ reduces to

$$0 = -rV_{x\varepsilon} + V_{x\varepsilon}(f_x + f_u U_x) + (\sigma_x + \sigma_u U_x)V_{xx} + \sigma V_{xxx}. \quad (13.6.13)$$

Combining (13.6.10) and (13.6.13), we find that $V_{x\varepsilon}(x^*, 0)$ and $U_\varepsilon(x^*, 0)$ are determined by the linear system

$$\begin{pmatrix} \pi_{uu} + V_x f_{uu} & f_u \\ 0 & f_x + f_u U_x - r \end{pmatrix} \begin{pmatrix} U_\varepsilon \\ V_{xe} \end{pmatrix} = \begin{pmatrix} -\sigma_u V_{xx} \\ -(\sigma_x + \sigma_u U_x) V_{xx} - \sigma V_{xxx} \end{pmatrix}, \quad (13.6.14)$$

where all functions are evaluated at the steady state value of x and $\varepsilon = 0$.

The results in (13.6.14) are interesting because of the information they require. Note that in order to compute the certainty nonequivalence correction term U_ε , we need to compute first the value of V_{xxx} at the deterministic steady state. Here we see an important application of the higher-order perturbation terms we computed for the deterministic model.

This can be continued to compute even higher-order approximations. Judd and Guu (1993) and Gaspar and Judd (1997) discuss this. These methods are not easy if they are done by hand. However, they can be automated; in fact Judd and Guu provides Mathematica programs and Gaspar and Judd provide Fortran programs to compute these expansions.

13.7 Perturbing Discrete-Time Systems

Since perturbation ideas are often used in discrete-time economic models, we will next show how to apply these ideas to discrete-time problems. Given the differences between continuous- and discrete-time, this attention to discrete-time models is not redundant. In particular, we see that we need to be careful in specifying the perturbation in the stochastic version of this analysis.

Deterministic Discrete-Time Control

We next show how to apply regular perturbation procedures to discrete-time control problems. Consider the problem

$$\max_{u_t} \sum_{t=0}^{\infty} \beta^t \pi(x_t, u_t) \quad (13.7.1)$$

$$\text{s.t. } x_{t+1} = f(x_t, u_t),$$

where x and u are scalars. The Bellman equation is

$$V(x) = \max_u \pi(x, u) + \beta V(f(x, u)) \quad (13.7.2)$$

with the first-order condition

$$0 = \pi_u(x, u) + \beta V'(f(x, u)) f_u(x, u). \quad (13.7.3)$$

The solution is a value function, $V(x)$, and a control law, $U(x)$. The defining pair of equations is

$$V(x) = \pi(x, U(x)) + \beta V(f(x, U(x))), \quad (13.7.4)$$

$$0 = \pi_u(x, U(x)) + \beta V'(f(x, U(x)))f_u(x, U(x)). \quad (13.7.5)$$

To derive a Taylor expansion of $V(x)$ and $U(x)$ near x^* , the steady state, we examine the system

$$0 = \pi_u(x, U(x)) + \beta V'(f(x, U(x)))f_u(x, U(x)), \quad (13.7.6)$$

$$V'(x) = \pi_x(x, U(x)) + \beta V'(f(x, U(x)))f_x(x, U(x)). \quad (13.7.7)$$

The first equation, (13.7.6), is a repetition of (13.7.5). The second condition, (13.7.7), is derived from differentiating (13.7.2) with respect to x and using (13.7.3).

A steady state is a pair (x^*, u^*) such that

$$\begin{aligned} x^* &= f(x^*, u^*), \\ 0 &= \pi_u(x^*, u^*) + \beta V'(x^*)f_u(x^*, u^*), \\ V(x^*) &= \pi(x^*, u^*) + \beta V(x^*), \\ V'(x^*) &= \pi_x(x^*, u^*) + \beta V'(x^*)f_x(x^*, u^*). \end{aligned} \quad (13.7.8)$$

We will assume that these four equations have solutions for the four steady state quantities x^* , u^* , $V(x^*)$, and $V'(x^*)$. As in the continuous-time case we need only assume local uniqueness and stability.

Differentiating (13.7.6)–(13.7.7) with respect to x yields

$$0 = \pi_{ux} + \pi_{uu}U' + \beta V''(f)[f_x + f_u U']f_u + \beta V'(f)[f_{ux} + f_{uu}U'], \quad (13.7.9)$$

$$V'' = \pi_{xx} + \pi_{xu}U' + \beta V''(f)[f_x + f_u U']f_x + \beta V'(f)[f_{xx} + f_{xu}U']. \quad (13.7.10)$$

At the steady state, $x^* = f(x^*, u^*)$. Therefore the steady state version of the system (13.7.9)–(13.7.10) is

$$\begin{aligned} 0 &= \pi_{ux}(x^*, u^*) + \pi_{uu}(x^*, u^*)U'(x^*) \\ &\quad + \beta V''(x^*)[f_x(x^*, u^*) + f_u(x^*, u^*)U'(x^*)]f_u(x^*, u^*) \\ &\quad + \beta V'(x^*)[f_{ux}(x^*, u^*) + f_{uu}(x^*, u^*)U'(x^*)], \end{aligned} \quad (13.7.11)$$

$$\begin{aligned}
V''(x^*) &= \pi_{xx}(x^*, u^*) + \pi_{xu}(x^*, u^*) U'(x^*) \\
&\quad + \beta V''(x^*) [f_x(x^*, u^*) + f_u(x^*, u^*) U'(x^*)] f_x(x^*, u^*) \\
&\quad + \beta V'(x^*) [f_{xx}(x^*, u^*) + f_{xu}(x^*, u^*) U'(x^*)].
\end{aligned} \tag{13.7.12}$$

These equations define a quadratic system for the unknowns $V''(x^*)$ and $U'(x^*)$. Again we use concavity to pick the correct solution, or we could compute these quantities by solving the corresponding discrete-time BVP.

Deterministic Growth

Let us take once more the simple optimal growth problem

$$\max_{c_t} \sum_{t=0}^{\infty} \beta^t u(c_t) \tag{13.7.13}$$

$$\text{s.t. } k_{t+1} = F(k_t) - c_t.$$

We could proceed as in the previous section. Instead, we will take a different approach, similar to that taken in Christiano (1990). The solution can be expressed as a policy function, $C(k)$, satisfying the Euler equation

$$u'(C(k)) = \beta u'(C(F(k) - C(k))) F'(F(k) - C(k)). \tag{13.7.14}$$

At the steady state, k^* , we have $F(k^*) - C(k^*) = k^*$, where (13.7.14) implies that $u'(C(k^*)) = \beta u'(C(k^*)) F'(k^*)$, which in turn implies the steady state condition $1 = \beta F'(k^*)$ which uniquely determines k^* . Furthermore $k^* = F(k^*) - C(k^*)$ determines $C(k^*)$.

Taking the derivative of (13.7.14) with respect to k implies that

$$\begin{aligned}
u''(C) C' &= \beta u''(C(F - C)) C'(F - C) [F' - C'] F'(F - C) \\
&\quad + \beta u'(C(F - C)) F''(F - C) [F' - C'].
\end{aligned} \tag{13.7.15}$$

At $k = k^*$, (13.7.15) reduces to (we will now drop all arguments)

$$u'' C' = u'' C' [F' - C'] + \beta u' F'' [F' - C']. \tag{13.7.16}$$

In (13.7.16) we know the value of all the terms at $k = k^*$ except $C'(k^*)$. Equation (13.7.16) is a quadratic equation in $C'(k^*)$ with the solution

$$C' = \frac{1}{2} \left(1 - F' + \beta \frac{u'}{u''} F'' + \sqrt{\left(-1 + F' - \beta \frac{u'}{u''} F'' \right)^2 + 4 \frac{u'}{u''} F''} \right). \tag{13.7.17}$$

Most applications end with the computation of $C'(k^*)$, but we next compute higher-order terms of the Taylor expansion of $C(k)$ at $k = k^*$. If we take another derivative of (13.7.15) and set $k = k^*$, we find that $C''(k^*)$ must satisfy

$$\begin{aligned} u''C'' + u'''C'C' \\ = \beta u'''(C'F'(1 - C'))^2 F' + \beta u''C''(F'(1 - C'))^2 F' \\ + 2\beta u''C'F'(1 - C')^2 F'' + \beta u'F'''(1 - C')^2 + \beta u'F''(-C''). \end{aligned} \quad (13.7.18)$$

The key fact is that (13.7.18) is a linear equation in the unknown $C''(k^*)$. This analysis can continue to compute higher-order terms. While it is clear that the discrete-time case has greater algebraic complexity than the continuous-time case, symbolic software can easily handle the problems; see Judd and Guu (1993) for Mathematica programs to solve this problem.

Stochastic Growth

We can also apply perturbation ideas to approximate solutions to discrete-time stochastic growth problem, albeit with somewhat greater difficulty to illustrate the ideas, we examine the problem

$$\max_{c_t} E \left\{ \sum_{t=0}^{\infty} \beta^t u(c_t) \right\} \quad (13.7.19)$$

$$\text{s.t. } k_{t+1} = (1 + \varepsilon z)F(k_t - c_t).$$

In this formulation we are assuming the k_t is the capital stock in hand at the beginning of period t and that out of it must come today's consumption, c_t , with the remaining capital, $k_t - c_t$, used in production and with the resulting output, $(1 + \varepsilon z)F(k_t - c_t)$, serving as the beginning-of-period capital stock in period $t + 1$. This formulation may seem awkward, but it keeps the analysis one-dimensional. If we assume, as is implicit in (13.7.13), that consumption choices are made after output is realized, then consumption depends on $(1 + \varepsilon z)F(k_t)$ which is a function of both z and k_t . We instead want to define the state k so that consumption depends only on k ; this is example of how changing the state variable may affect the computational complexity of a problem. Therefore we assume that consumption is chosen and consumed at the beginning of the period, before output has been realized, effectively reducing the available capital stock. So we let ε be a scalar parameter, and assume that the random output in period t is $(1 + \varepsilon z)F(k_t - c_t)$ where z is a mean-zero random variable with unit variance. This analysis will also be different from the

continuous-time case because here the perturbation parameter is ε , the standard deviation, not the variance.

Since we have a new state variable, we must redo the deterministic perturbation analysis. The solution of the deterministic case, $\varepsilon = 0$, can be expressed as a policy function, $C(k)$, satisfying the Euler equation

$$u'(C(k)) = \beta u'(C(F(k - C(k))))F'(k - C(k)). \quad (13.7.20)$$

At the steady state, k^* , $F(k^* - C(k^*)) = k^*$, and $1 = \beta F'(k^* - C(k^*))$, conditions which uniquely determine k^* and $C(k^*)$. Taking the derivative of (13.7.20) with respect to k implies that

$$\begin{aligned} u''(C(k))C'(k) &= \beta u''(C(F(k - C(k))))C'(F(k - C(k))) \\ &\quad \times F'(k - C(k))[1 - C'(k)]F'(k - C(k)) \\ &\quad + \beta u'(C(F(k - C(k))))F'''(k - C(k))[1 - C'(k)]. \end{aligned} \quad (13.7.21)$$

At $k = k^*$, (13.7.21) reduces to (we now drop all arguments of all functions)

$$u''C' = \beta u''C'F'[1 - C']F' + \beta u'F''[1 - C']. \quad (13.7.22)$$

Equation (13.7.22) is a quadratic equation with the solution

$$C' = \frac{1}{2} \left(1 - \beta - \beta^2 \frac{u'}{u''} F'' + \sqrt{\left(1 - \beta - \beta^2 \frac{u'}{u''} F'' \right)^2 + 4 \frac{u'}{u''} \beta^2 F''} \right).$$

If we take another derivative of (13.7.21) and set $k = k^*$, we find that

$$\begin{aligned} u''C'' + u'''C'C' &= \beta u'''(C'F'(1 - C'))^2F' + \beta u''C''(F'(1 - C'))^2F' \\ &\quad + 2\beta u''C'F'(1 - C')^2F'' + \beta u'F'''(1 - C')^2 \\ &\quad + \beta u'F''(-C''), \end{aligned}$$

which is a linear equation in the unknown $C''(k^*)$.

We now examine the fully stochastic problem (13.7.20). Again we express the consumption function in terms of the beginning-of-period capital stock. Note that the current productivity shock does not enter into the consumption function because it is not known at the time consumption is chosen, and the previous period's productivity shock does not affect current consumption beyond its impact on the current capital stock because productivity shocks are assumed to be i.i.d. With uncertainty, the Euler equation is

$$u'(C(k)) = \beta E\{u'(g(\varepsilon, k, z))R(\varepsilon, k, z)\}, \quad (13.7.23)$$

where

$$\begin{aligned} g(\varepsilon, k, z) &\equiv C((1 + \varepsilon z)F(k - C(k))), \\ R(\varepsilon, k, z) &= (1 + \varepsilon z)F'(k - C(k)). \end{aligned} \quad (13.7.24)$$

Differentiating (13.7.24) with respect to ε yields (we drop arguments of F and C)

$$\begin{aligned} g_\varepsilon &= C_\varepsilon + C'(zF - (1 + \varepsilon z)F'C_\varepsilon), \\ g_{\varepsilon\varepsilon} &= C_{\varepsilon\varepsilon} + 2C'_\varepsilon(zF - (1 + \varepsilon z)F'C_\varepsilon) + C''(zF - (1 + \varepsilon z)F'C_\varepsilon)^2, \\ &\quad + C'(-zF'C_\varepsilon^2 + (1 + \varepsilon z)F''(C_\varepsilon^2 - (1 + \varepsilon z)F'(C_{\varepsilon\varepsilon}))). \end{aligned} \quad (13.7.25)$$

At $\varepsilon = 0$, (13.7.25) implies that

$$\begin{aligned} g_\varepsilon &= C_\varepsilon + C'(zF - F'C_\varepsilon), \\ g_{\varepsilon\varepsilon} &= C_{\varepsilon\varepsilon} + 2C'_\varepsilon(zF - F'C_\varepsilon) + C''(zF - F'C_\varepsilon)^2 \\ &\quad + C'(-2zF'C_\varepsilon + F''C_\varepsilon^2 - F'C_{\varepsilon\varepsilon}). \end{aligned} \quad (13.7.26)$$

Differentiating (13.7.23) with respect to ε shows that

$$u''C_\varepsilon = \beta E\{u''g_\varepsilon(1 + \varepsilon z)F' + u'F'z - u'(1 + \varepsilon z)F''C_\varepsilon\}, \quad (13.7.27)$$

$$\begin{aligned} u'''C_\varepsilon^2 + u''C_{\varepsilon\varepsilon} &= \beta E\{u'''g_\varepsilon^2(1 + \varepsilon z)F' + 2u''g_\varepsilon F'z \\ &\quad - 2u''g_\varepsilon(1 + \varepsilon z)F''C_\varepsilon + u''g_{\varepsilon\varepsilon}(1 + \varepsilon z)F' \\ &\quad - 2u'zF''C_\varepsilon + u'(1 + \varepsilon z)F'''C_\varepsilon^2 - u'(1 + \varepsilon z)F''C_{\varepsilon\varepsilon}\}. \end{aligned} \quad (13.7.28)$$

Since $E\{z\} = 0$, we conclude from (13.7.27) that $C_\varepsilon = 0$, which in turn implies that

$$g_\varepsilon = C'zF,$$

$$g_{\varepsilon\varepsilon} = C_{\varepsilon\varepsilon} + 2C'_\varepsilon zF + C''(zF)^2 - C'F'C_{\varepsilon\varepsilon}.$$

Using the second-order terms in (13.7.28), we find that at $\varepsilon = 0$,

$$\begin{aligned} u'''C_\varepsilon^2 + u''C_{\varepsilon\varepsilon} &= \beta E\{u'''g_\varepsilon^2F' + 2u''g_\varepsilon F'z - 2u''g_\varepsilon F''C_\varepsilon + u''g_{\varepsilon\varepsilon}F' \\ &\quad - 2u'zF''C_\varepsilon + u'F'''C_\varepsilon^2 - u'F''C_{\varepsilon\varepsilon}\}. \end{aligned}$$

Using the normalization $E\{z^2\} = 1$, we find that

$$u''C_{ee} = \beta[u'''C'C'F^2F' + 2u''C'FF' + u''(C_{ee} + C''F^2 - C'F'C_{ee})F' - u'F''C_{ee}].$$

Solving for C_{ee} yields

$$C_{ee} = \frac{u'''C'C'F^2 + 2u''C'F + u''C''F^2}{u''C'F' + \beta u'F''}.$$

This exercise demonstrates that perturbation methods can also be applied to the discrete-time stochastic growth model, albeit at somewhat greater cost.

13.8 Perturbing Jump Process Control Problems

Our examples have concentrated on economic growth problems. In this section we use perturbation methods to examine a “patent race” problem, which is a simple jump process control problem. We do this because such problems have a structure very different from those studied above.

In this problem the controller is trying to reach some state at which he stops and receives a payment; we call it a patent race because it is similar to the problem a research and development program faces. The example below assumes a monopolist; Judd (1985) discusses the duopoly case. In this problem, the state is x , the distance from the objective; specifically, $x \leq 0$ and the prize is P and won when $x = 0$. The control is u , which is the instantaneous probability that progress will be made, and its cost is $\alpha u^2/2$. If the current state is x when a jump occurs and $s > x$ is the point it lands on, the probability density of where it lands is $f(s; x)$; we let $F(x)$ denote the probability that a jump from x will send the state to $x = 0$. If ρ is the rate of utility discount, the Bellman equation for the profit-maximizing control problem is

$$\begin{aligned} M(x) = \max_u & \left\{ -\frac{\alpha u^2}{2} dt + M(x)(1 - \rho dt)(1 - u dt) \right. \\ & \left. + (1 - \rho dt) \left(\int_x^0 M(s) f(s, x) ds \right) u dt + P F(x) u dt \right\}. \end{aligned} \quad (13.8.1)$$

The implied control law is

$$\alpha u = \int_x^0 M(s) f(s, x) ds + P F(x) - M(x). \quad (13.8.2)$$

When we substitute the control law into (13.8.1), we arrive at

$$0 = \frac{\left(\int_x^0 M(s)f(s, x) ds + PF(x) - M(x) \right)^2}{2\alpha} - \rho M(x). \quad (13.8.3)$$

We will compute an approximation of the value function, $M(x)$, valid for an open set of parameter choices. We will take advantage of this example to show how we need to construct a parameter which is small in a meaningful sense. We will solve problems for which the prize, P , is small, since, if $P = 0$, then we know that the solution is $M(x) = 0$. But what does small P mean? If we express P in dollars instead of, say, Japanese yen, we automatically reduce P by two orders of magnitude. But such a change in units has no economic substance. The rule in perturbation methods is that the perturbation parameter should be dimensionless; only then will the concept of "small" not depend on the choice of units. To proceed in this fashion, one should examine dimensionless versions of a problem.

Define $m \equiv M/P$ to be the value of problem (13.8.1) relative to the prize. m is a dimensionless quantity representing the value of the game which will yield a substantive concept of small. Rewritten in terms of m , (13.8.3) becomes

$$m(x) = p \left(\int_x^0 m(s)f(s, x) ds + F(x) - m(x) \right)^2, \quad (13.8.4)$$

where $p \equiv P/2\alpha\rho$ is the size of the prize relative to the marginal cost of innovation and the cost of capital. Since the dimension of ρ is $(\text{time})^{-1}$ and that of α is $(\text{dollars}) \times (\text{time})$, p is dimensionless and will be our measure of the prize. Since m , p , f , and F are all dimensionless, (13.8.4) is a dimensionless representation of (13.8.3). When p is zero, (13.8.4) yields the obvious solution, $m(x) = 0$. When expressed in this fashion, we see that p may be zero either because P is zero or because $\alpha\rho$, the "costs," are infinite. The case of $\rho = \infty$ is the case where the controller discounts the future at an infinite rate.

We see that the trivial case of $p = 0$ corresponds to several situations. We are really assuming the the prize is small compared to the slope of the marginal cost curve and the value of the future to the controller. This will imply that the prize is to the first order equal to the costs and that the net profits of an innovator are small relative to the prize. The interpretation that the prize just covers the opportunity costs of innovative activity makes our focus on small p more plausible.

Once we transform (13.8.3) into a dimensionless equation, we also must transform other variables of interest; in particular, the control variable, u . However, u is not dimensionless because it measures effort per unit of time, has dimension time^{-1} , and depends on the time unit. We can rewrite (13.8.2) into the dimensionless form:

$$\tilde{u} \equiv \frac{u}{\rho} = 2p \left(\int_x^0 m(s)f(s, x) ds + F(x) - m(x) \right), \quad (13.8.5)$$

where \tilde{u} is the dimensionless rate of effort per normalized unit of time.

We now illustrate computing a local solution to (13.8.4). If $p = 0$, then $m = 0$. We approximate $m(x; p)$ for small p up to $\mathcal{O}(p^n)$:

$$m(x; p) \approx pm_p(x; 0) + p^2 m_{pp}(x; 0)/2 + \cdots + p^n \frac{1}{n!} \frac{\partial^n m}{\partial p^n}(x, 0). \quad (13.8.6)$$

Differentiating (13.8.4) with respect to p and evaluating at $p = 0$ shows that

$$m_p(x; 0) = F(x)^2. \quad (13.8.7)$$

Taking a second derivative of (13.8.4) with respect to p , evaluating it at $p = 0$, and using the fact that $\partial m / \partial p (x; 0) = F(x)^2$, we find that

$$m_{pp}(x; 0) = 2F(x) \left(\int_x^0 F(s)^2 f(s, x) ds - F(x)^2 \right) \quad (13.8.8)$$

Continuing in this fashion, we can recursively compute each derivative of $m(x; 0)$ with respect to p justified by the known smoothness of m in terms of p . Note that no smoothness of m in x need be assumed.

From these expressions we may infer several obvious properties of the optimal control for small p . For example, if p is small, effort increases as one is closer to the finish. This follows from the observation that the $pF(x)$ term dominates in (13.8.6), implying that u rises as $F(x)$ and x rise. Also u falls as α and ρ rise, an intuitive result, since both represent costs.

13.9 Global Quality Test for Asymptotic Approximations

The ideas underlying the foregoing asymptotic methods validate only a local concept of approximation. In our growth examples, we can be confident about the asymptotic expansion of $C(k)$ around the steady state only for a sufficiently small neighborhood of the steady state. We saw in table 13.1 that the perturbation approximations do quite well even far from the steady state for a simple growth problem, but to do so, we had to find another way to compute the true solution. This approach to accuracy checking is desirable but generally not feasible; if we had the true answer from another method, why would we do a perturbation analysis?

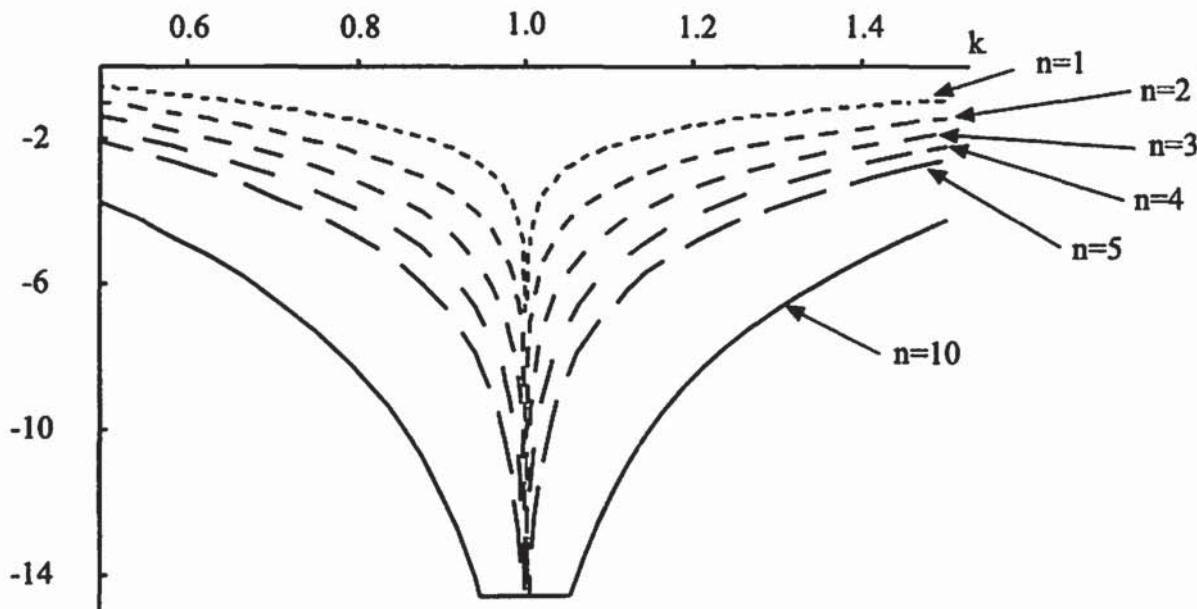


Figure 13.1
Euler equation residuals for Taylor series solutions of (13.5.10)

We next evaluate a simple Euler equation test of the global quality of the asymptotic approximations given in (13.5.17) by computing how well approximations for $C(k)$ do in solving the defining differential equation (13.5.11). More precisely, if $\hat{C}(k)$ is the asymptotic expansion of the consumption policy for (13.5.10), we compute

$$E(k) = \frac{\gamma C'(k)(f(k) - C(k)) - C(k)(\rho - f''(k))}{\rho C(k^*)}. \quad (13.9.1)$$

The definition of $E(k)$ includes a division to create a unit-free expression of the continuous-time Euler equation.

We consider the case of Cobb-Douglas production function with capital share 0.25, $\rho = 0.05$, and utility function c^{-1} . Figure 13.1 plots $\max[\log_{10} E(k), -15]$ for the degree $n = 1, 2, 3, 4, 5$, and 10 expansions; we put a floor at -15 , since the computation of $E(k)$ has at most a precision of 10^{-15} . For degree 1 expansions, the log error quickly rises to -1 . As we increase n , the log error falls uniformly over the interval $[0.5, 1.5]$. At $n = 5$, the log error is -2 over the region, and much better on $[0.8, 1.2]$. At $n = 10$, the log error even better; being -5 over $[0.6, 1.4]$ and displaying essentially zero Euler equation error over $[0.9, 1.1]$. The quality of the $n = 10$ approximation indicated by our $E(k)$ measure is quite similar to the actual errors indicated in table 13.1 for the same problem.

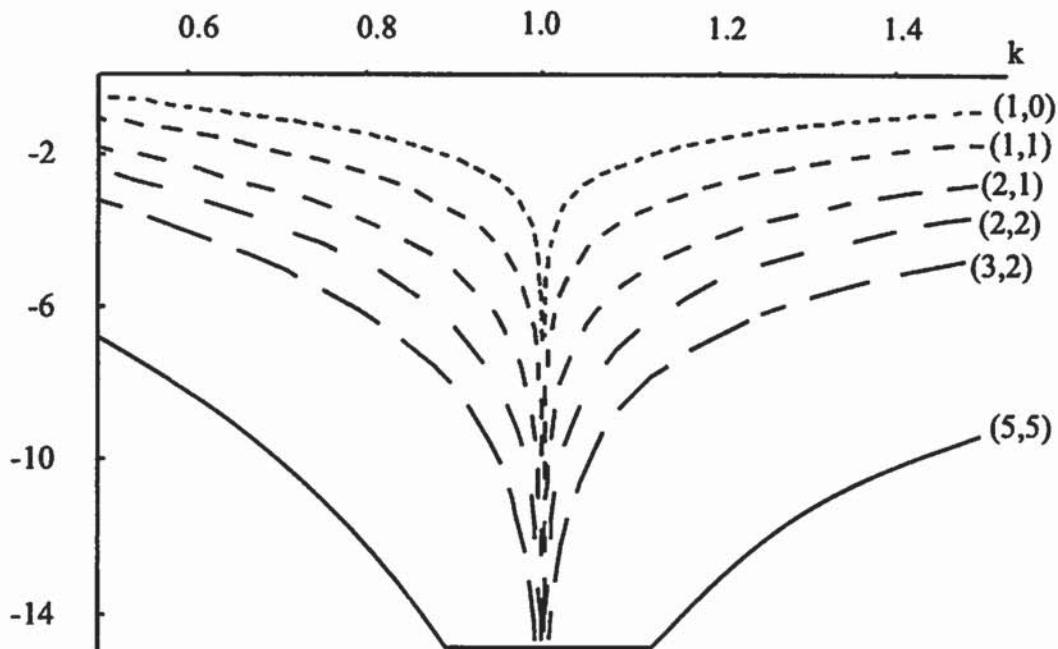


Figure 13.2
Euler equation residuals for Padé solutions of (13.5.10)

The perturbation procedure produces derivatives of $C(k)$ at $k = k^*$. We have so far used them to compute a Taylor expansion for $C(k)$. We could also use the derivative information to produce a Padé expansion. Figure 13.2 displays both the Euler equation errors for the degree 1 Taylor expansion and the $(1, 1)$, $(2, 1)$, $(2, 2)$, $(3, 2)$, and $(5, 5)$ Padé expansions; these choices use the same information as the degree 1, 2, 3, 4, 5, and 10 Taylor expansions. The Padé expansion does better than the Taylor expansion using the same information, particularly for capital stocks far from the steady state.

This example shows that high-order asymptotic expansions can be very accurate away from the central point of the expansion, particularly when Padé expansions are computed. We have also presented a diagnostic test that can be applied to any perturbation approximation and appears to indicate its quality. These expansions and the diagnostic tests can be computed quickly once we know the steady state.

Exercises

- Consider the optimal growth problem in equation (13.5.10) with $u(c) = c^{0.2}$, $\rho = 0.05$, and Cobb-Douglas $f(k)$ with capital share 0.25 and steady state capital stock equal to 1. Compute the degree 5 Taylor expansion of the optimal consumption rule around the steady state. Compare it to the reverse shooting result.

2. Compute the Taylor expansion around the steady state of degree 4 in k and degree 2 in ε for (13.7.19) with $\beta = 0.9$, $u(c) = \log c$, $\log z \sim N(0, 1)$, and $F(k) = k + 0.2k^{0.25}$. Can you do lengthen this to degree 4 in ε .
3. Compute the degree 1, 2, 3, and 4 approximations of solutions to exercise 10.5 around the steady state. Compare the accuracy of the Taylor series approximations to the answers computed there. Suppose that we change the laws of motion to $dk_t = I_t dt + \sigma dz_t$ where the dz_t are i.i.d. white noises and formulate the problem as in (13.6.1). Compute the Taylor series approximation that is degree 4 in k and degree 2 in σ . Compare the result to a projection method solution with small Euler equation methods.
4. Compute the degree 1, 2, 3, and 4 approximations of the solution to exercise 10.6 around the steady state. Compare the accuracy of the Taylor series approximations to the answer computed there. Suppose that we change the laws of motion to $dQ = (q - Q/100) dt + \sigma dz$ where the dz is white noise. Compute the initial terms in the Taylor series expansion of the policy function with respect to (Q, σ) based at the deterministic steady state.
5. Suppose that the production and utility functions in the continuous-time optimal growth problem (10.7.2) depends on time as in $f(k, t) = (1 + Ah(t))k^\alpha$ and $u(c, t) = (1 + Bg(t))c^{1+\gamma}/(1+\gamma)$, where h and g are periodic with period one; that is, $h(t+1) = h(t)$ for all t , and similarly for g . Linearize the resulting dynamic system to determine how the seasonal factors $h(t)$ and $g(t)$ affect the seasonality of output, savings, and interest rates.
6. Consider the stochastic growth model with adjustment costs in exercise 12.7. Derive its HJB equation, and compute some low-order Taylor expansions for the optimal consumption policy function around the deterministic steady state in terms of k and σ .

14 Regular Perturbations in Multidimensional Systems

We continue our study of perturbation methods, generalizing the methods of the previous chapter to the case of several dimensions. This extension is necessary to handle important questions with multiple capital stocks, multiple states, and heterogeneous agents. This results in a substantial increase in complexity, but the basic ideas remain unchanged.

We first review the multivariate version of the implicit function theorem. When we attempt to move beyond the simple linear term, we run into substantial notational problems since the objects being calculated are *tensors*, which are generalizations of matrices and vectors. We introduce Einstein summation notation which allows us to compactly express the nonlinear terms in a multivariate Taylor series expansion. This tensor notation approach also helps us write computer procedures to do these calculations, since tensor manipulations are easily implemented by symbolic manipulation programming languages such as MACSYMA, Maple, and Mathematica.

We then illustrate how to perturb multidimensional boundary value problems similar to those that arise in dynamic control problems. We first extend the Einstein summation notation to make it appropriate for optimal control problems, and then compute multivariate, nonlinear Taylor series expansions of solutions to both continuous- and discrete-time deterministic multidimensional control problems, and for stochastic continuous-time multidimensional control problems.

14.1 Multidimensional Comparative Statics and Tensor Notation

We first review the simple mechanics of comparative statics and the implicit function theorem. We then develop the tensor notation which is necessary to efficiently express the critical formulas for multidimensional analysis.

Multivariate Linear Approximation

Theorem 13.1.1 stated the general implicit function theorem for R^n . Comparative statics for multidimensional systems are constructed using this theorem and implicit differentiation. Let

$$H(x, h(x)) = 0 \tag{14.1.1}$$

implicitly define $h: R^n \rightarrow R^m$, where $H: R^n \times R^m \rightarrow R^m$. Suppose that we know $H(x_0, y_0) = 0$; then $h(x_0) = y_0$. Implicit differentiation of (14.1.1) with respect to x and evaluating the result at $x = x_0$ shows that the Jacobian h_x at $x = x_0$ is

$$h_x(x_0) = -H_y(x_0, y_0)^{-1} H_x(x_0, y_0), \tag{14.1.2}$$

which is well-defined as long as $H_y(x_0, y_0)$ is nonsingular. Equation (14.1.2) is the comparative static vector which expresses how $h(x)$ changes as we move x away from x_0 .

The linear approximation of $h(x)$ based at $x = x_0$ is

$$h(x) \doteq h(x_0) + h_x(x_0)(x - x_0). \quad (14.1.3)$$

Equation (14.1.3) is just the beginning of the multivariate Taylor series approximation of $h(x)$ based at x_0 . One may wonder why we would want to go beyond the first-order expression of (14.1.3) and construct higher-order terms of the Taylor series expansion, equation (6.11.3). While such higher-order terms may not be useful in *the qualitative* comparative static expressions we often see in economic analysis, they may be useful as a *numerical* approximation of $h(x)$ for x not too far from x_0 . To do this in a clean and compact fashion, we need tensor notation.

Tensor Notation

In multidimensional perturbation problems, we use the multidimensional chain rule. Unfortunately, when applying the chain rule in R^n , we will have many summations, and conventional notation becomes unwieldy. To see this, look back on theorem 6.11.3; the central expression is one cluttered with several \sum and ∂ signs. The *Einstein summation notation* for tensors and its adaptations will give us a natural way to address the notational problems.¹

Tensor notation is just a way of dealing with multidimensional collections of numbers and operations involving them. First, suppose that a_i is a collection of numbers indexed by $i = 1, \dots, n$, and that x^i is a singly indexed collection of real numbers. Then

$$a_i x^i \equiv \sum_i a_i x^i. \quad (14.1.4)$$

In this way we eliminate the \sum symbol in sums. Of course one has to remember that the summation is over n dimensions, since n is not explicitly expressed; the value of n is understood in actual applications. Similarly suppose that a_{ij} is a collection of numbers indexed by $i, j = 1, \dots, n$, and that x^i and y^j are singly indexed collections of real numbers. Then

1. The dubious reader should try to read the formulas in Bensoussan (1988) where conventional notation is used. We should also note that we use here only the notation of tensors, not the machinery of differential geometry which often accompanies the use of tensors.

$$a_{ij} x^i y^j \equiv \sum_i \sum_j a_{ij} x^i y^j. \quad (14.1.5)$$

In this way we again eliminate the summation symbols. The second item to note is the location of the index. We will often pair a superscript indexed collection x^i with a subscript indexed collection y_i and write $x^i y_i$ as shorthand for the sum $\sum_i x^i y_i$. On the other hand, $a_i^j x^i y_j$ means that $\sum_i \sum_j a_i^j x^i y_j$. The general rule is that if, in a multiplicative term, an index appears as both a subscript and a superscript, then we understand that we sum over that index and eliminate the \sum symbols that would otherwise appear.

So far all of this looks familiar, since it is just a way of rewriting standard matrix and vector operations. If we think of a_j^i as a matrix, x_i as a row vector, and y^j as a column vector, then the product $x_i y^j$ represents the inner product of the vectors x and y , and $a_j^i x_i y^j$ is the quadratic form of the matrix a with the vectors x and y . We can also form new indexed collections of numbers from products. For example, the product $a_j^i x_i$ can be thought of as a singly indexed collection of numbers, z_j , which can also be thought of as a row vector. Therefore a_j^i acts as a linear map on x , just as a matrix. Unlike the case with vectors and matrices, the order in which we write the terms has no importance. Therefore $x_i y^j = y^j x_i$, and $a_j^i x_i y^j$, $x_i a_j^i y^j$, and $y^j a_j^i x_i$ express the same quadratic form. This is expected, for all we have done is dropped the \sum symbol and multiplication is commutative. While the analogies with matrices and vectors are useful, we should not focus on them because we will be constructing complex collections of real numbers that are neither vectors nor matrices.

As long as indexes are not the same, arbitrary products are allowed. For example, $x_i y_j$ is the doubly indexed set of numbers, b_{ij} , where the (i, j) term equals the product of x_i and y_j ; $x_i y_j$ is the *outer product* of x_i and y_j . In general, $a_{j_1, j_2, \dots, j_m}^{i_1, i_2, \dots, i_l}$ is a $l - m$ tensor, a set of numbers indexed by l superscripts and m subscripts. It can be thought of as a scalar-valued multilinear map on $(R^n)^l \times (R^n)^m$. This generalizes the idea that matrices are bilinear maps on $(R^n)^2$. The summation convention becomes particularly useful for higher-order tensors. For example, in R^n ,

$$c_{j_3, j_4}^{i_3, i_4} = a_{j_1, j_2, j_3}^{i_1, i_2, i_3} b_{i_1, i_2, i_4}^{j_1, j_2, j_4} \equiv \sum_{i_1=1}^n \sum_{i_2=1}^n \sum_{j_1=1}^n \sum_{j_2=1}^n a_{j_1, j_2, j_3}^{i_1, i_2, i_3} b_{i_1, i_2, i_4}^{j_1, j_2, j_4}.$$

In our use the distinction between superscript and subscripts will not be exploited. Therefore the summation rule applies even if an index appears only as a subscript or only as a superscript. For example, $a^i x^i \equiv \sum_i a^i x^i$ and $a_{ij} x_i y_j \equiv \sum_{i,j} a_{ij} x_i y_j$. Also the “sum over repeated indexes” rule even applies within a single term. For example,

$a_i^i \equiv \sum_i a_i^i$; if we think of a as a matrix, then a_i^i is the trace of a . Also, if a_{ij} is a tensor, then a_{ii} is defined to be $\sum_i a_{ii}$, and is the trace of a .

Tensor-Style Multivariate Calculus

In our applications if $f: R \rightarrow R^m$, then f_j will be the derivative of f with respect to x_j . The following equation expresses the multivariate Taylor expansion in tensor notation:

$$\begin{aligned} f = f(x_0) + f_i(x^i - x_0^i) + \frac{1}{2} f_{ij}(x - x^0)^i (x - x^0)^j \\ + \frac{1}{3!} f_{ijl}(x - x^0)^i (x - x^0)^j (x - x^0)^l + \dots, \end{aligned}$$

where $f_i \equiv (\partial f / \partial x_i)(x_0)$, $f_{ij} \equiv (\partial^2 f / \partial x_i \partial x_j)(x_0)$, and so on. This expression is much more compact than the conventional expression displayed in theorem 1.6.2 but conveys the same information. More generally, if $f: R^n \rightarrow R^m$, then f_j^i will be the derivative of the i th component of f with respect to x_j . Note that we are deviating from our previous conventions stated in chapter 1 concerning sub- and superscripts. Here x^i refers to the i component of the vector x , but f_i is still the i th component of the gradient of f . Since sub- and superscripts both refer to separate components of a single vector, we cannot distinguish different vectors by this distinction. Therefore different vectors and tensors must use different letters.

We will make extensive use of the multivariate chain rule. If $f: R^n \rightarrow R^m$, $g: R^m \rightarrow R^l$ and $h(x) = g(f(x))$, then $h: R^n \rightarrow R^l$, and the Jacobian of h is

$$h_j^i \equiv \frac{\partial h^i}{\partial x_j} = g_l^i f_j^l.$$

Furthermore the 1–2 tensor of second-order derivatives is

$$h_{jk}^i \equiv \frac{\partial^2 h^i}{\partial x_j \partial x_k} = g_{lm}^i f_k^m f_j^l + g_l^i f_{jk}^l.$$

This can be continued to express arbitrary derivatives.

14.2 Linearization of Multidimensional Dynamic Systems

A frequent technique in studying a dynamical system is to first compute its stationary points, namely its steady states, and then examine the response of the system to small deviations from the steady state; this is expressed in impulse response functions. In

the last chapter we showed how to do this for optimal control problems with one state and one control variable. We next develop this technique for problems with several state variables. We begin our discussion of multidimensional systems by reviewing the standard linearization method. In succeeding sections we develop the necessary mathematics for computing higher-order approximations of such systems.

In dynamic economics, multidimensional systems are generally of the form

$$\begin{aligned}\dot{x} &= f(x, \lambda, u), \\ \dot{\lambda} &= g(x, \lambda, u), \\ 0 &= h(x, \lambda, u, \mu), \\ x(0) &= x_0, \quad \lim_{t \rightarrow \infty} |x(t)|, |\lambda(t)|, |\mu(t)| < \infty.\end{aligned}\tag{14.2.1}$$

In optimal control problems, $x \in R^n$ is the state vector, $u \in R^m$ are the controls, $\lambda \in R^n$ is the costate vector, $f: R^n \times R^n \times R^m \rightarrow R^n$, $g: R^n \times R^n \times R^m \rightarrow R^n$, and $h: R^n \times R^n \times R^m \times R^l \rightarrow R^{m+l}$. In other contexts x is the set of predetermined variables the λ are the free dynamic variables; that is, the value of λ at $t = 0$ is not fixed exogenously but rather limited by asymptotic conditions on the system (14.2.1). In general, $\mu \in R^l$ is a collection of nondynamic variables which may arise, say, due to nonlinear constraints on the states and controls. In general dynamic models there is no restriction that x and λ be of equal length; we will stay with the equal length assumption here, but the generalization is direct.

Generally, the vectors u and μ are fixed once x and λ are determined. We therefore assume that the relation $0 = h(x, \lambda, u, \mu)$ can be inverted, yielding functions $\mathcal{U}(x, \lambda)$ and $\mathcal{M}(x, \lambda)$ that satisfy $0 = h(x, \lambda, \mathcal{U}(x, \lambda), \mathcal{M}(x, \lambda))$, and that these expressions can be implicitly differentiated to find the Jacobians \mathcal{U}_x and \mathcal{U}_λ . In this case we can rewrite (14.2.1) as

$$\begin{aligned}\dot{x} &= f(x, \lambda, \mathcal{U}(x, \lambda)), \\ \dot{\lambda} &= g(x, \lambda, \mathcal{U}(x, \lambda)).\end{aligned}\tag{14.2.2}$$

A steady state of (14.2.1) is a triple $(x^*, \lambda^*) \equiv Z^*$, u^* , and μ^* such that $0 = f(Z^*, u^*) = g(Z^*, u^*) = h(Z^*, u^*, \mu^*)$. We will make a crucial existence and uniqueness assumption about (14.2.1). We assume that for each value of $x(0)$ near the steady state, there is a unique value of $\lambda(0)$ near λ^* such that the system (14.2.1) is bounded. This property of local uniqueness and stability needs to be established by theory. Fortunately failures that are local in nature will be revealed in perturbation analysis.

We next consider what happens to the solution as we slightly change the initial value of x away from the steady state, assuming that the solution to (14.2.1) leads back to the steady state. We proceed in the manner described in section 13.1. We define a continuum of problems all with the dynamic structure of (14.2.2) but with the parameterized initial conditions $x(0, \varepsilon) = x^* + \varepsilon \xi_x(0)$. These initial conditions together with stability will imply the parameterized solutions $x(t, \varepsilon)$ for x and $\lambda(t, \varepsilon)$ for λ . We want to compute both $\xi_\lambda(t) \equiv \lambda_\varepsilon(t, 0)$ and $\xi_x(t) \equiv x_\varepsilon(t, 0)$, the initial deviations of λ and x from their steady state values as we change ε . Our objective is to compute the relation between the exogenous perturbation vector $\xi_x(0)$ and the endogenous $\xi_\lambda(t)$ and $\xi_x(t)$ paths.

To accomplish this, we differentiate (14.2.2) and the parameterized initial conditions $x(0, \varepsilon) = x^* + \varepsilon \xi_x(0)$ with respect to ε . The result is a system of linear differential equations and boundary conditions for $\xi_x(t)$ and $\xi_\lambda(t)$:

$$\dot{\xi} = \begin{pmatrix} \dot{\xi}_x \\ \dot{\xi}_\lambda \end{pmatrix} = \begin{pmatrix} f_x + f_u \mathcal{U}_x & f_\lambda + f_u \mathcal{U}_\lambda \\ g_x + g_u \mathcal{U}_x & g_\lambda + g_u \mathcal{U}_\lambda \end{pmatrix} \begin{pmatrix} \xi_x \\ \xi_\lambda \end{pmatrix}, \quad (14.2.3)$$

where the matrix in (14.2.3), denoted A , is evaluated at Z^* , and where $\xi \equiv (\xi_x^\top, \xi_\lambda^\top)^\top$. The solution to (14.2.3) is $\xi = \xi(0)e^{At}$, where $\xi(0)$ is the initial value of ξ . Given a value for $\xi_x(0)$, we fix the value of $\xi_\lambda(0)$ by imposing asymptotic stability on the linear system (14.2.3),

$$\lim_{t \rightarrow \infty} |\xi(0) e^{At}| < \infty. \quad (14.2.4)$$

The combination of the initial condition $\xi_x(0)$ and the asymptotic stability condition (14.2.4) on (14.2.3) creates a boundary value problem. We saw in chapter 10 that we solve this problem by writing A in its Jordan canonical form, $N^{-1}DN$, and rearrange and decompose D and N^{-1} ,

$$D = \begin{pmatrix} D_1 & 0 \\ 0 & D_2 \end{pmatrix}, \quad N^{-1} = \begin{pmatrix} N_{11} & N_{12} \\ N_{21} & N_{22} \end{pmatrix},$$

so that D_1 is stable and D_2 unstable. In this form the solution is

$$\xi_\lambda(0) = -N_{22}^{-1} N_{21} \xi_x(0). \quad (14.2.5)$$

This assumes that N is nonsingular. This is not always satisfied in practice. Even if N is nonsingular, it may be ill-conditioned. If N is singular or ill-conditioned, then one must use another method to solve the linear problem (14.2.3)–(14.2.4). The key goal here is to compute $\xi_\lambda(0)$; how one accomplishes this is not important to the next steps.

In general, the predetermined variables' initial value, $x(0)$, determines the initial value of λ through the stability condition in (14.2.1). Let $\lambda = \Lambda(x)$ be that map for general value of x . Equation (14.2.5) expresses how a small change in x near x^* equal to $\varepsilon\xi_x(0)$ leads to a small change in the predetermined variables equal to $\varepsilon\xi_\lambda(0)$ near the steady state. Equation (14.2.5) tells us that $\Lambda_x(x^*) = -N_{22}^{-1}N_{21}$.

A recursive formulation aims to determine $U(x)$, the functional relation between the control and the state. Our specification (14.2.1) just defines a relation between u and (x, λ) which was denoted $\mathcal{U}(x, \lambda)$. The function $U(x)$ is defined in $U(x) = \mathcal{U}(x, \Lambda(x))$; that is, to find u given x , we first compute $\Lambda(x)$ and then use \mathcal{U} . Since $\Lambda_x(x^*) = -N_{22}^{-1}N_{21}$ this implies that

$$\frac{\partial U}{\partial x}(x^*) = \frac{\partial \mathcal{U}}{\partial x}(x^*, \lambda^*) - \frac{\partial \mathcal{U}}{\partial \lambda}(x^*, \lambda^*) N_{22}^{-1}N_{21}. \quad (14.2.6)$$

Since \mathcal{U}_x and \mathcal{U}_λ can be determined at the steady state, (14.2.6) expresses all the derivatives in $U_x(x^*)$ in terms of the steady state x^* and the derivatives of f , g , and h at the steady state. Equation (14.2.6) expresses the linear approximation of $U(x)$ at $x = x^*$ in terms of the primitives in (14.2.1).

These calculations have produced the relation between the equilibrium value of u and the state x . With this we can determine the dynamic behavior over time of u and x . Substituting $\lambda = \Lambda(x)$ into (14.2.1) produces the equilibrium dynamic law of motion

$$\dot{x} = f(x, \Lambda(x), U(x)), \quad (14.2.7)$$

which is now the equilibrium system expressed solely in terms of x . For x near x^* , (14.2.7) has the linear approximation $\dot{x} = B(x - x^*)$, where

$$B = f_x(x^*, \lambda^*, u^*) + f_\lambda(x^*, \lambda^*, u^*)\Lambda_x(x^*) + f_u(x^*, \lambda^*, u^*)U_x(x^*).$$

Since the linear approximation has the solution e^{Bt} , the *impulse response function* with respect to the state x_i is the i th component of the vector function e^{At} and expresses the impact over time of a “shock” to x at $t = 0$.

Example: Time to Build

We will next linearize a time-to-build model that represents the idea that it takes time for investment expenditures to augment the productive capital stock. We assume two stocks of capital: k_1 , the maturing capital, and k_2 , the productive capital. Production of the one final good is $f(k_2)$ and is split between consumption c and investment. Investment expenditures consist of two types: additions to k_1 and the costs of the

capital maturation process, $\beta_1 k_1$. Productive capital depreciates at the rate δ_2 , and maturing capital becomes productive capital at the rate δ_1 . Let c be consumption. The resulting optimal control problem is

$$\begin{aligned} \max_c & \int_0^\infty e^{-\rho t} u(c) dt \\ \text{s.t. } & \dot{k}_1 = f(k_2) - c - \delta_1 k_1 - \beta_1 k_1, \\ & \dot{k}_2 = \delta_1 k_1 - \delta_2 k_2. \end{aligned} \tag{14.2.8}$$

The Hamiltonian is

$$H = u(c) + \lambda_1(f(k_2) - c - \delta_1 k_1 - \beta_1 k_1) + \lambda_2(\delta_1 k_1 - \delta_2 k_2),$$

and the costate equations are

$$\begin{aligned} \dot{\lambda}_1 &= \rho \lambda_1 - \lambda_1(-\delta_1 - \beta_1) - \lambda_2 \delta_1, \\ \dot{\lambda}_2 &= \rho \lambda_2 - \lambda_1 f'(k_2) - \lambda_2(-\delta_2). \end{aligned}$$

The maximum principle implies that $0 = u'(c) - \lambda_1$, which can be inverted to define consumption as a function of λ_1 , which we will express as $c = C(\lambda_1)$.

The steady state costate conditions are

$$0 = (\rho + \delta_1 + \beta_1) \lambda_1 - \lambda_2 \delta_1,$$

$$0 = -\lambda_1 f'(k_2) + \lambda_2(\rho + \delta_2),$$

which implies that the steady-state stock of productive capital, k_2^* , is the solution to

$$f'(k_2^*) = \frac{(\rho + \delta_1 + \beta_1)(\rho + \delta_2)}{\delta_1}.$$

The linearization matrix in (14.2.3) for (14.2.7) around its steady state is

$$A = \begin{pmatrix} -\delta_1 - \beta_1 & f' & -C' & 0 \\ \delta_1 & -\delta_2 & 0 & 0 \\ 0 & 0 & \rho + \delta_1 + \beta_1 & -\delta_1 \\ 0 & -\lambda_1 f'' & -f' & \rho + \delta_2 \end{pmatrix}.$$

The matrix A has four distinct eigenvalues, two stable and two unstable. To continue the example, we make the specific assumptions $u(c) = \log c$, $f(k_2) = k_2^\alpha$,

$\alpha = 0.25$, $\rho = 0.05$, $\delta_1 = 1.0$, $\delta_2 = 0.15$, and $\beta_1 = 0.2$. The steady state is $k_2^* = 1.0$, $k_1^* = 0.15$, $c^* = 0.82$, $\lambda_1^* = 1.2195$, $\lambda_2^* = 1.5244$, and the linearized system matrix is

$$A = \begin{pmatrix} -1.2 & 0.25 & 0.67 & 0 \\ 1.0 & -0.15 & 0 & 0 \\ 0 & 0 & 1.25 & -1.0 \\ 0 & 0.23 & -0.25 & 0.2 \end{pmatrix}.$$

The two stable eigenvectors and eigenvalues are

$$\mu_1 = -1.3721, \quad v_{\mu_1} = \begin{pmatrix} -0.77 \\ 0.63 \\ -0.04 \\ -0.01 \end{pmatrix}, \quad \mu_2 = -0.2568, \quad v_{\mu_2} = \begin{pmatrix} -0.07 \\ 0.73 \\ -0.38 \\ -0.57 \end{pmatrix},$$

and the unstable components are

$$\xi_1 = 0.3068, \quad v_{\xi_1} = \begin{pmatrix} 0.32 \\ 0.71 \\ 0.46 \\ 0.43 \end{pmatrix}, \quad \xi_2 = 1.4221, \quad v_{\xi_2} = \begin{pmatrix} -0.26 \\ -0.16 \\ -0.94 \\ 0.16 \end{pmatrix},$$

A choice for the matrix N is

$$N = \begin{pmatrix} -1.18 & 0.20 & 0.32 & 0.20 \\ 0.57 & 0.54 & -0.40 & -0.88 \\ 0.47 & 0.71 & -0.01 & 0.90 \\ 0.05 & 0.12 & -0.96 & 0.79 \end{pmatrix}.$$

This implies that

$$\Lambda^* \equiv \Lambda_k(k^*) = -N_{22}^{-1}N_{21} = \begin{pmatrix} -0.41 & -0.57 \\ -0.57 & -0.85 \end{pmatrix}.$$

Since $u'(c) = \lambda_1$, we know that $c = \mathcal{U}(k, \lambda) = \lambda_1^{-1}$, and $C(k) = \mathcal{U}(k, \Lambda(k))$, which implies that

$$\begin{pmatrix} C_{k_1} \\ C_{k_2} \end{pmatrix} = \begin{pmatrix} \mathcal{U}_{k_1} + \mathcal{U}_{\lambda_1} \Lambda_{11}^* + \mathcal{U}_{\lambda_2} \Lambda_{12}^* \\ \mathcal{U}_{k_2} + \mathcal{U}_{\lambda_1} \Lambda_{21}^* + \mathcal{U}_{\lambda_2} \Lambda_{22}^* \end{pmatrix} = \begin{pmatrix} 0.28 \\ 0.38 \end{pmatrix}.$$

Since $k_2^* = 1.0$, $k_1^* = 0.15$, and $c^* = 0.82$. The final conclusion is that a one dollar increase in k_2 , the productive capital stock, will result in a 38-cent increase in consumption and that a one-dollar increase in k_1 , the maturing capital stock, will result in a 28-cent increase in consumption.

Discrete-Time Systems

We can perform the same operations on discrete-time systems. Suppose that $z \in R^m$, $f: R^{2m} \rightarrow R^m$, and we want to solve

$$f(z_t, z_{t+1}) = 0. \quad (14.2.9)$$

In the case of a discrete-time control problem, we should think of z as a stack of state, costate, and control variables. We again begin by computing the steady state equation

$$f(z^*, z^*) = 0 \quad (14.2.10)$$

for, z^* , the steady state value of z . We then linearize (14.2.9) at z^* to arrive at the linear system

$$f_{z_t}(z^*, z^*)(z_t - z^*) + f_{z_{t+1}}(z^*, z^*)(z_{t+1} - z^*) = 0.$$

At this point we have arrived at a linear dynamic model, the kind examined in the linear rational expectations literature. If $f_{z_{t+1}}(z^*, z^*)$ is invertible, we can write

$$z_{t+1} - z^* = -(f_{z_{t+1}}(z^*, z^*))^{-1} f_{z_t}(z^*, z^*)(z_t - z^*)$$

and apply eigenvalue decomposition methods to the discrete-time system. As with the continuous-time case, asymptotic boundedness conditions may tie down the values of the components of z not predetermined. If $f_{z_{t+1}}(z^*, z^*)$ is singular, then special methods are required. The reader is referred to the linear rational expectations literature (e.g., Anderson et al. 1996 for a review) for a discussion of solution methods.

14.3 Locally Asymptotically Stable Multidimensional Control

In section 14.2 we saw how to compute linear and higher-order approximations of near-steady state dynamics. If we were to stay with problems that can be expressed solely in terms of ordinary differential equations, that would suffice. However, many economic problems, particularly those involving uncertainty and strategic behavior, cannot be so simply expressed. We will next develop a multidimensional version of high-order asymptotic techniques for problems in recursive equilibrium form.

The general n -dimensional deterministic control problem with m controls is

$$\max_u \int_0^\infty e^{-\rho t} \pi(x, u) dt \quad (14.3.1)$$

$$\text{s.t. } \dot{x} = f(x, u),$$

where $x \in R^n$ is the state, $u \in R^m$ is the control, and π is a concave return function. By the second welfare theorem, the solution to the general problem is also an equilibrium solution to the complete market dynamic equilibrium problem. Since our general problem includes the possibility of multiple consumption goods (represented by components of u), multiple capital stocks (represented by components of x), and multiple agents whose aggregate utility function maximized by equilibrium represented by $\pi(x, u)$, the computed solutions also describe dynamic competitive general equilibrium in such models.

Extended Tensor Notation Conventions

We will use the conventional tensor notation introduced in section 14.1, but we need to adapt it for optimal control problems. Specifically we adopt additional conventions so that we can distinguish between states and controls in an efficient fashion.

In our augmented tensor notation, superscripts will still refer to different components of a tensor, whereas subscripts will refer to derivatives of those component functions. Furthermore, in order to distinguish between states and controls, we will let $i, j, k, l \dots$, index states and $\alpha, \beta, \gamma, \dots$, index controls. Therefore, if $f(x, u)$ is a vector-valued function of the state variables x and the controls u , then

$$f \equiv \begin{pmatrix} f^1(x, u) \\ f^2(x, u) \\ \vdots \\ f^n(x, u) \end{pmatrix},$$

where f^i is the i th component function. The matrix of derivatives with respect to the state variables is denoted by the tensor

$$f_i^j(x, u) \equiv \frac{\partial f^j}{\partial x_i}(x, u).$$

Note that at any point (x, u) , $f_i^j(x, u)$ is a 1–1 tensor. The derivatives with respect to controls are denoted by the tensor

$$f_\alpha^j(x, u) \equiv \frac{\partial f^j}{\partial u_\alpha}(x, u).$$

and also form a 1–1 tensor. Furthermore the cross derivatives with respect to controls and states are denoted by the tensor

$$f_{\alpha i}^j(x, u) \equiv \frac{\partial^2 f^j}{\partial u_\alpha \partial x_i}(x, u).$$

Continuing in this fashion, we can express any order of derivative of f .

We will frequently run into expressions of the form $f(x, U(x, \varepsilon), \varepsilon)$. In these cases we will need to use the chain rule. For example, if $g(x, \varepsilon) \equiv f(x, U(x, \varepsilon), \varepsilon)$, then

$$\frac{\partial g}{\partial x_i} \equiv g_i = f_i + f_\alpha U_i^\alpha, \quad \frac{\partial g}{\partial \varepsilon} \equiv g_\varepsilon = f_\varepsilon + f_\alpha U_\varepsilon^\alpha.$$

With this extended Einstein notation in hand, we continue our discussion of perturbation methods applied to multidimensional models.

Computing Taylor Series Expansions for General Optimal Control Problems

The general deterministic problem includes an arbitrary complete market dynamic equilibrium with multiple consumption goods (represented by components of u), multiple capital stocks (represented by components of x), and multiple agents whose aggregate utility function maximized by equilibrium represented by $\pi(x, u)$. We will now proceed with a dynamic programming approach to this analysis. The Bellman equation for the value function, $V(x)$, is

$$\rho V(x) = \max_u \pi(x, u) + V_i(x) f^i(x, u).$$

The first-order conditions with respect to the u_α , $\alpha = 1, \dots, m$, are

$$0 = \pi_\alpha(x, u) + V_i(x) f_\alpha^i(x, u). \quad (14.3.2)$$

Equation (14.3.2) looks like one equation, but it is really m equations, one for each α . Since α is not bound by an implicit summation, as i is, it is understood to be free in (14.3.2). The reader must keep this in mind to understand the equations below.

Equations (14.3.2) implicitly define the optimal control, $u = U(x)$, where the policies $u_\alpha = U^\alpha(x)$ must satisfy

$$0 = \pi_\alpha(x, U(x)) + V_i(x) f_\alpha^i(x, U(x)). \quad (14.3.3)$$

In combination, the Bellman equation and (14.3.2) imply the system

$$0 = \pi(x, U(x)) + V_i(x)f^i(x, U(x)) - \rho V(x), \quad (14.3.4)$$

$$0 = \pi_\alpha(x, U(x)) + V_i(x)f_\alpha^i(x, U(x)), \quad (14.3.5)$$

which defines the value function, $V(x)$, and the policy function, $U(x)$.

Our objective here is to solve for both the value function and the policy function. In fact we are going to compute Taylor series expansions

$$\begin{aligned} V(x) &= V(x^0) + V_i(x - x^0)^i + \frac{1}{2}V_{ij}(x - x^0)^i(x - x^0)^j \\ &\quad + \frac{1}{3!}V_{ijl}(x - x^0)^i(x - x^0)^j(x - x^0)^l + \dots, \\ U^\alpha(x) &= U(x^0) + U_i^\alpha(x - x^0)^i + \frac{1}{2}U_{ij}^\alpha(x - x^0)^i(x - x^0)^j \\ &\quad + \frac{1}{3!}U_{ijl}^\alpha(x - x^0)^i(x - x^0)^j(x - x^0)^l + \dots. \end{aligned} \quad (14.3.6)$$

To compute these U_i^α , U_{ij}^α , U_{ijl}^α , V_i , V_{ij} , and V_{ijl} coefficients, we just differentiate the underlying system, (14.3.4)–(14.3.5), with respect to the x_i , and solve for the undetermined coefficients. If we differentiate (14.3.4) with respect to x_j and use the envelope theorem, we find that

$$\rho V_j = \pi_j + V_{ij}f^i + V_i f_j^i. \quad (14.3.7)$$

In order to keep down the clutter, we will not write the arguments of π , V , and U , and their derivatives, since they will be the same as they are in the basic system, (14.3.4)–(14.3.5), and clear from context.

The steady state is determined by the conditions

$$0 = f^i(u, x),$$

$$\rho V(x) = \pi(x, u) + V_i(x)f^i(x, u),$$

$$0 = \pi_\alpha(u, x) + V_i(x)f_\alpha^i(u, x),$$

$$\rho V_j(x) = \pi_j(u, x) + V_{ij}(x)f^i(u, x) + V_i(x)f_j^i(u, x),$$

which yield the steady state quantities u^* , x^* , $V(x^*)$, and $V_j(x^*)$. Note that the $V_j(x^*)$ values are the linear coefficients in the expansion of V in (14.3.6), and knowing the steady state will also yield $V(x^*)$ and $U(x^*)$, two more terms in (14.3.6).

We next compute the V_{ij} and U_j^β terms. If we differentiate (14.3.5) with respect to the x_j , we find that

$$0 = \pi_{\alpha j} + \pi_{\alpha\beta} U_j^\beta + V_{ij} f_\alpha^i + V_i (f_{\alpha j}^i + f_{\alpha\beta}^i U_j^\beta). \quad (14.3.8)$$

Note that (14.3.8) is a system of conditions, one for each αj pair. In this case we can express U_j^β in terms of V and its derivatives

$$U_j^\beta = -Q^{\alpha\beta} (\pi_{\alpha j} + V_{ij} f_\alpha^i + V_i f_{\alpha j}^i),$$

where $Q^{\alpha\beta}$ is the tensor function defined by $(\pi_{\alpha\beta} + V_i f_{\alpha\beta}^i) Q^{\alpha\beta} = \delta_\beta^\alpha$ where $\delta_\beta^\alpha = 1$ if $\alpha = \beta$ and 0 otherwise. Therefore $Q^{\alpha\beta}$ is the tensor (matrix) inverse of $\pi_{\alpha\beta} + V_i f_{\alpha\beta}^i$. Differentiating (14.3.7) with respect to x_l implies that

$$\begin{aligned} \rho V_{jl} &= \pi_{jl} + \pi_{j\gamma} U_l^\gamma + V_{jl} f^i + V_{ij} (f_l^i + f_\gamma^i U_l^\gamma) \\ &\quad + V_{il} f_j^i + V_i (f_{jl}^i + f_{j\gamma}^i U_l^\gamma). \end{aligned} \quad (14.3.9)$$

Substituting our solution for the U_j^β tensor (matrix) into (14.3.9) yields (remember that the U_j^β tensor is the same as the U_l^γ tensor, since the indexes are just dummy variables)

$$\begin{aligned} \rho V_{jl} &= \pi_{jl} + V_{jl} f^i + V_{ij} f_l^i + V_{il} f_j^i + V_i f_{jl}^i \\ &\quad - (\pi_{j\gamma} + V_{ij} f_\gamma^i + V_i f_{j\gamma}^i) Q^{\alpha\gamma} (\pi_{\alpha l} + V_{il} f_\alpha^i + V_i f_{\alpha l}^i). \end{aligned} \quad (14.3.10)$$

The system of equations in (14.3.10) hold at each state x . If we evaluate (14.3.10) at the steady state, then $f^i = 0$ and (14.3.10) becomes the Riccati-like equation

$$\begin{aligned} \rho V_{jl} &= \pi_{jl} + V_{ij} f_l^i + V_{il} f_j^i + V_i f_{jl}^i \\ &\quad - (\pi_{j\gamma} + V_{ij} f_\gamma^i + V_i f_{j\gamma}^i) Q^{\alpha\gamma} (\pi_{\alpha l} + V_{il} f_\alpha^i + V_i f_{\alpha l}^i). \end{aligned} \quad (14.3.11)$$

Solving the Riccati equation at the steady state yields the steady state values of V_{jl} and, through (14.3.8), the steady state values of U_l^γ . But we already have the solutions for the steady state values of U_l^γ because the U_l^γ tensor is nothing but the linear approximation, $U_x(x^*)$ computed in section 14.2 by applying the eigenvalue decomposition method to linearizing the system (14.2.1). Furthermore that approach is a good one to use to compute U_l^γ . While we have not accomplished anything new at this point, the expressions in (14.3.8) and (14.3.9) lay the foundation for the higher-order terms.

We now find out how easy it is to compute the higher-order terms. Differentiating (14.3.9) with respect to x_m implies that

$$\begin{aligned}
\rho V_{jlm} = & \pi_{jlm} + \pi_{jly} U_m^\gamma + \pi_{jym} U_l^\gamma + \pi_{jy\delta} U_m^\delta U_l^\gamma + \pi_{jy} U_{lm}^\gamma \\
& + V_{ijlm} f^i + V_{ijm} (f_l^i + f_y^i U_l^\gamma) + V_{ij} (f_{lm}^i + f_{ym}^i U_l^\gamma + f_y^i U_{lm}^\gamma) \\
& + V_{ilm} f_j^i + V_{il} (f_{jy}^i U_m^\gamma + f_{jm}^i) + V_{im} (f_{jl}^i + f_{jy}^i U_l^\gamma) \\
& + V_i (f_{jlm}^i + f_{jly}^i U_m^\gamma + f_{jym}^i U_l^\gamma + f_{jy\delta}^i U_m^\delta). \tag{14.3.12}
\end{aligned}$$

When we rewrite (14.3.8) as

$$0 = (\pi_{\alpha\beta} + V_i f_{\alpha\beta}^i) U_l^\beta + \pi_{\alpha l} + V_{il} f_\alpha^i + V_i f_{\alpha l}$$

and differentiate this expression with respect to x_m , we find that

$$\begin{aligned}
0 = & (\pi_{\alpha\beta} + V_i f_{\alpha\beta}^i) U_{lm}^\beta \\
& + (\pi_{\alpha\beta\gamma} U_m^\gamma + \pi_{\alpha\beta m} + V_{im} f_{\alpha\beta}^i + V_i f_{\alpha\beta m}^i + V_i f_{\alpha\beta\gamma}^i U_m^\gamma) U_l^\beta \\
& + \pi_{\alpha lm} + \pi_{\alpha ly} U_m^\gamma + V_{ilm} f_\alpha^i + V_{il} (f_{\alpha m}^i + f_{\alpha y}^i U_m^\gamma) \\
& + V_{im} f_{\alpha l}^i + V_i (f_{\alpha lm}^i + f_{\alpha ly}^i U_m^\gamma). \tag{14.3.13}
\end{aligned}$$

An efficient way to compute the steady state solutions for V_{ilm} and U_{lm}^β in (14.3.13) is immediately apparent. Consider (14.3.13) for a fixed lm pair. The $\pi_{\alpha\beta} + V_i f_{\alpha\beta}^i$ tensor appears repeatedly for each lm pair. Hence we can use the steady state value of the $Q^{\alpha\beta}$ tensor (an inversion which is done just once) to express U_{lm}^β linearly in terms of the known steady state values of various steady state derivatives and V_{ilm} . After we gather terms, it takes just two matrix multiplications for each lm pair to determine the coefficients of the affine representation

$$\begin{aligned}
U_{lm}^\beta = & Q^{\alpha\beta} [(\pi_{\alpha\beta\gamma} U_m^\gamma + \pi_{\alpha\beta m} + V_{im} f_{\alpha\beta}^i + V_i f_{\alpha\beta m}^i + V_i f_{\alpha\beta\gamma}^i U_m^\gamma) U_l^\beta \\
& + (\pi_{\alpha lm} + \pi_{\alpha ly} U_m^\gamma + V_{il} (f_{\alpha m}^i + f_{\alpha y}^i U_m^\gamma) + V_{im} f_{\alpha l}^i + V_i (f_{\alpha lm}^i + f_{\alpha ly}^i U_m^\gamma)] \\
& + (\pi_{\alpha\beta} + V_i f_{\alpha\beta}^i)^{-1} f_\alpha^i V_{ilm}.
\end{aligned}$$

These representations of the U_{lm}^β can then be substituted into the steady state value of (14.3.12), the critical fact being that $f^i = 0$ in the steady state version of (14.3.12) which kill off the one instance of V_{ijlm} to produce a system of equations linear in the V_{jlm} .

The final important fact is that we can repeat this to compute the third-order terms of U in a similar sequentially linear fashion. We may ask how big can we go in this fashion. Clearly the greater the dimension, the fewer higher-order terms we can add to the expansion. Gaspar and Judd (1997) discuss some of these feasibility issues.

14.4 Perturbations of Discrete-Time Problems

We will now repeat this perturbation analysis for the discrete-time canonical control problem

$$\begin{aligned} V(x) &= \max_u \sum_{t=0}^{\infty} \beta^t \pi(x_t, u_t) \\ \text{s.t. } x_{t+1} &= F(x_t, u_t), \\ x_0 &= x, \end{aligned} \tag{14.4.1}$$

where $x \in R^n$ is the vector of states, $u \in R^m$ the vector of controls, and $F(x, u)$ the law of motion. Again we begin with the Bellman dynamic programming equation:

$$V(x) = \max_u \pi(x, u) + \beta V(F(x, u)). \tag{14.4.2}$$

If the optimal control law is $u = U(x)$, the first-order condition implies that

$$0 = \pi_\alpha(x, U(x)) + \beta V_i(F(x, U(x)))F_\alpha^i(x, U(x)), \tag{14.4.3}$$

and the envelope theorem applied to the Bellman equation implies that

$$V_j(x) = \pi_j(x, U(x)) + \beta V_i(F(x, U(x)))F_j^i(x, U(x)). \tag{14.4.4}$$

The steady state is defined by values for x , u , and $V_i(x)$, which satisfy

$$\begin{aligned} \pi_\alpha(x, u) + \beta V_i(F(x, u))F_\alpha^i(x, u) &= 0, \\ \pi_j(x, u) + \beta V_i(F(x, u))F_j^i(x, u) &= V_j(x), \\ x - F(x, u) &= 0. \end{aligned} \tag{14.4.5}$$

We assume that we have found a locally unique and stable steady state, with steady state solutions to 14.4.6 equal to (x^*, u^*, V_i^*) .

We next move to computing U_i^α and V_{jl} . To do this, we differentiate both (14.4.3) and (14.4.4) with respect to each x_l , yielding the systems

$$\begin{aligned} V_{jl}(x) &= \pi_{jl} + \pi_{j\alpha} U_l^\alpha + \beta V_{im}(F)[F_l^m + F_\alpha^m U_l^\alpha]F_j^i + \beta V_i(F)[F_{j\alpha}^i U_l^\alpha + F_{jl}^i], \\ 0 &= \pi_{\alpha l} + \pi_{\alpha\phi} U_l^\phi + \beta V_{im}(F)[F_\gamma^m U_l^\gamma + F_l^m]F_\alpha^i + \beta V_i(F)[F_{\gamma\alpha}^i U_l^\alpha + F_{\alpha l}^i], \end{aligned} \tag{14.4.6}$$

where each instance of F , π , and their derivatives is evaluated at $(x, U(x))$, and each instance of U and its derivatives is evaluated at x . The steady state version of (14.4.6)

$$\begin{aligned} V_{jl}(x^*) &= \pi_{jl} + \pi_{j\alpha} U_l^\alpha + \beta V_{im}(x^*)[F_l^m + F_\alpha^m U_l^\alpha]F_j^i + \beta V_i(x^*)[F_{j\alpha}^i U_l^\alpha + F_{jl}^i], \\ 0 &= \pi_{al} + \pi_{a\phi} U_l^\phi + \beta V_{im}(x^*)[F_\gamma^m U_l^\gamma + F_l^m]F_\alpha^i + \beta V_i(x^*)[F_{\gamma\alpha}^i U_l^\alpha + F_{al}^i], \end{aligned} \quad (14.4.7)$$

where now each instance of F , π , and their derivatives is evaluated at the steady state values (x^*, u^*) , and each instance of U and its derivatives is evaluated at x^* . Note that because $x^* = F(x^*, u^*)$, all instances of V and its derivatives in (14.4.7) are evaluated at x^* . This is the Riccati system of equations for the unknown matrices $U_l^\alpha(x^*, u^*)$ and $V_{jl}(x^*, u^*)$, and it can be solved using the standard methods for solving Riccati systems. However, we can also compute the values of U_l^α via the discrete-time analogue of (14.2.6), and with these values in hand use (14.4.7) to solve for the values of V_{jl} , an easy task since (14.4.7) is linear in the V_{jl} values.

The next step is to compute the quadratic-cubic terms V_{ilm} and U_m^α . To do this, we let $V^+ \equiv V(F(x, U(x)))$, $V_i^+ \equiv V_i(F(x, U(x)))$, etc., and differentiate the system (14.4.6) with respect to x_m , yielding

$$\begin{aligned} V_{ilm} &= \pi_{ilm} + \pi_{jl\alpha} U_m^\alpha + [\pi_{jm\phi} + \pi_{j\alpha\phi} U_m^\alpha]U_l^\phi + \pi_{j\phi} U_{lm}^\phi \\ &\quad + \beta V_i^+[F_{ilm}^i + F_{jl\alpha}^i U_m^\alpha + F_{j\phi m}^i U_l^\phi + F_{j\alpha\phi}^i U_m^\alpha U_l^\phi + F_{j\phi}^i U_{lm}^\phi] \\ &\quad + \beta V_{in}^+[F_m^n + F_\alpha^n U_m^\alpha][F_{jl}^i + F_{j\phi}^i U_l^\phi] \\ &\quad + \beta V_{in}^+[F_l^n + F_\phi^n U_l^\phi][F_{jm}^i + F_{jm}^i U_m^\alpha] \\ &\quad + \beta V_{in}^+[F_{lm}^n + F_{l\alpha}^n U_m^\alpha + F_{\phi m}^n U_l^\phi + F_{\alpha\phi}^n U_m^\alpha U_l^\phi + F_\phi^n U_{lm}^\phi]F_j^i \\ &\quad + \beta V_{inp}^+[F_m^p + F_\alpha^p U_m^\alpha][F_l^n + F_\phi^n U_l^\phi]F_j^i, \end{aligned} \quad (14.4.8)$$

$$\begin{aligned} 0 &= \pi_{alm} + \pi_{a\phi l} U_m^\phi + \pi_{a\phi m} U_l^\phi + \pi_{a\phi\gamma} U_m^\phi U_l^\gamma + \pi_{a\phi} U_{lm}^\phi \\ &\quad + \beta V_i^+[F_{alm}^i + F_{\alpha\gamma l}^i U_m^\gamma + F_{\alpha\phi m}^i U_l^\phi + F_{\alpha\phi\gamma}^i U_l^\phi U_m^\gamma + F_{\alpha\phi}^i U_{lm}^\phi] \\ &\quad + \beta V_{in}^+[F_m^n + F_\alpha^n U_m^\alpha][F_{al}^i + F_{\alpha\phi}^i U_l^\phi] \\ &\quad + \beta V_{in}^+[F_l^n + F_\phi^n U_l^\phi][F_{am}^i + F_{\alpha\phi}^i U_m^\alpha] \\ &\quad + \beta V_{in}^+[F_{lm}^m + F_{l\gamma}^m U_m^\gamma + F_{\phi m}^m U_l^\phi + F_{\phi\gamma}^m U_m^\gamma U_l^\phi + F_\phi^m U_{lm}^\phi]F_\alpha^i \\ &\quad + \beta V_{inp}^+[F_m^p + F_\gamma^p U_m^\gamma][F_l^n + F_\phi^n U_l^\phi]F_\alpha^i. \end{aligned} \quad (14.4.9)$$

Equations (14.4.8) and (14.4.9) hold for all x . At x^* we have $F(x^*, U(x^*)) = F(x^*, u^*) = x^*$, and all the V^+ terms and their derivatives become $V(x^*)$ terms. The result is a linear set of equations in the unknown values of $V_{inp}(x^*)$ and $U_{in}^\gamma(x^*)$.

We again see that the higher-order terms can be computed by linear operations. While the discrete-time case is notationally denser, this is not of much concern because it all can be automated. However, the computational burden is substantially greater due to the presence of the composition of functions. Continuous-time formulations are clearly more convenient.

14.5 Multisector Stochastic Growth

For the multidimensional case, the stochastic problem is

$$\max_u E \left\{ \int_0^\infty e^{-\rho t} \pi(x, u) dt | x_0 \right\}$$

$$\text{s.t. } dx = f(x, u) dt + \sqrt{2\varepsilon} I dz,$$

where dz is a vector of i.i.d. white noises of unit variance; to keep the notation simple, we will assume that the variance-covariance matrix is I . The Bellman equation is

$$0 = \max_u \pi(x, u) + V_i f^i + \varepsilon V_{ii} - \rho V, \quad (14.5.1)$$

and the Euler equation (the first-order condition) is

$$0 = \pi_\alpha + V_i f_\alpha^i. \quad (14.5.2)$$

Differentiating (14.5.1) with respect to ε and using the envelope theorem yields

$$0 = V_{ie} f^i + V_{ii} + \varepsilon V_{iie} - \rho V_\varepsilon. \quad (14.5.3)$$

At the steady state of the deterministic case studied in the previous section, (14.5.3) reduces to $0 = V_{ii} - \rho V_\varepsilon$, which implies that

$$V_\varepsilon = \rho^{-1} V_{ii}. \quad (14.5.4)$$

Equation (14.5.4) tells us, to a first order, how an increase in variance affects the value. The term V_{ii} is the trace of the Hessian of V and is a measure of the curvature of the deterministic value function, so it is known or at least is calculable by the methods used above. Equation (14.5.4) has a natural interpretation. It says that the change in value from adding a unit of variance equals the discounted value of the curvature of the deterministic problem.

We next determine how ε affects U . Differentiating (14.5.3) with respect to x_j implies that

$$0 = V_{ij\epsilon} f^i + V_{ie}(f_\alpha^i U_\alpha^\alpha + f_j^i) + V_{ijj} + \epsilon V_{ijj\epsilon} - \rho V_{je}. \quad (14.5.5)$$

When we impose the steady state requirements, we get

$$0 = V_{ie}(f_\alpha^i U_\alpha^\alpha + f_j^i) + V_{ijj} - \rho V_{je}. \quad (14.5.6)$$

The derivative of (14.5.2) with respect to ϵ is

$$0 = \pi_{\alpha\beta} U_\epsilon^\beta + V_{ie} f_\alpha^i + V_i f_{\alpha\beta}^i U_\epsilon^\beta. \quad (14.5.7)$$

Equations (14.5.6) and (14.5.7) produce U_ϵ^β and V_{je} at $x = x^*$ and $\epsilon = 0$.

We next want to see how ϵ affects the slopes of U which is expressed by the tensor U_{je}^β . The derivative of (14.5.5) with respect to x_k is

$$\begin{aligned} 0 = & V_{ijk\epsilon} f^i + V_{ij\epsilon}(f_k^i + f_\alpha^i U_\alpha^\alpha) + V_{ik\epsilon}(f_\alpha^i U_\alpha^\alpha + f_j^i) + V_{ie}(f_{jk}^i + f_{j\alpha}^i U_\alpha^\alpha) \\ & + V_{ie}(f_{ak}^1 U_\alpha^\alpha + f_{\alpha\beta}^1 U_j^\alpha U_k^\beta + f_\alpha^i U_{jk}^\alpha) + V_{ijk} + \epsilon V_{ijk\epsilon} - \rho V_{jke}, \end{aligned} \quad (14.5.8)$$

which has the steady state equation

$$\begin{aligned} 0 = & V_{ij\epsilon}(f_k^i + f_\alpha^i U_\alpha^\alpha) + V_{ik\epsilon}(f_j^i + f_\alpha^i U_\alpha^\alpha) + V_{ie}(f_{jk}^i + f_{j\alpha}^i U_\alpha^\alpha) \\ & + f_{ak}^i U_\alpha^\alpha + f_{\alpha\beta}^i U_j^\alpha U_k^\beta + f_\alpha^i U_{jk}^\alpha) + V_{ijk} - \rho V_{jke}. \end{aligned} \quad (14.5.9)$$

The derivative of (14.5.7) with respect to x_k is

$$\begin{aligned} 0 = & \pi_{\alpha\beta k} U_\epsilon^\beta + \pi_{\alpha\beta\gamma} U_\epsilon^\beta U_k^\gamma + V_{ik\epsilon} f_\alpha^i + V_{ie}(f_{ak}^i + f_{\alpha\beta}^i U_\epsilon^\beta) \\ & + V_{ik} f_{\alpha\beta}^i U_\epsilon^\beta + V_i(f_{\alpha\beta k}^i + f_{\alpha\beta\gamma}^i U_k^\gamma) U_\epsilon^\beta + V_i f_{\alpha\beta}^i U_{ek}^\beta. \end{aligned} \quad (14.5.10)$$

Equations (14.5.9) and (14.5.10) produce U_{ke}^β and V_{jke} , which tell us how the slopes of the policy function are affected by a change in uncertainty.

This approach can be taken to compute any derivative of U and V with respect to ϵ ; however, we do not continue this here because of the obvious algebraic costs. Again we find that we can compute a formal power series for V in multidimensional problems in a sequentially linear fashion.

Alternative Approaches

Regular perturbation takes an implicit differentiation approach to compute a Taylor series approximation of the nonlinear functions U and V . We showed that this approach can be used to compute both linear and nonlinear approximations to U based at the deterministic steady state. In some areas of economics, this is not the

usual approach to finding linear functions that “approximate” optimal and equilibrium policy functions U .

One alternative approach is to find the deterministic steady state but then replace the original nonlinear optimal control problem with a linear-quadratic problem that is similar to the original problem. The linear-quadratic problem can then be solved using standard methods. The resulting linear control law is then used instead of the nonlinear U to analyze the original problem. The strategy of finding a linear-quadratic problem similar to the nonlinear model is an intuitively appealing approach, since we know how to solve linear-quadratic problems.

Magill (1977) applies this approach to the general multivariate problem

$$\max_u \int_0^\infty e^{-\rho t} \pi(x, u) dt \quad (14.5.11)$$

s.t. $dx = f(x, u)dt + \sigma(x) dz.$

He first computes the deterministic steady state values of x , u , and λ , the shadow price of x ; call them x^* , u^* , and λ^* . He constructs the matrices

$$Q = \begin{pmatrix} \pi_{xx} & \pi_{xu} \\ \pi_{ux} & \pi_{uu} \end{pmatrix} + \sum \lambda_i^* \begin{pmatrix} f_{xx}^i & f_{xu}^i \\ f_{ux}^i & f_{uu}^i \end{pmatrix},$$

$$C = f_x, D = f_u,$$

where all functions are evaluated at the steady state. He showed that the solution to

$$\max_{\delta_x, \delta_u} E \left\{ \int_0^\infty e^{-\rho t} \begin{pmatrix} x^* + \delta_x \\ u^* + \delta_u \end{pmatrix}^\top Q \begin{pmatrix} x^* + \delta_x \\ u^* + \delta_u \end{pmatrix} dt \right\}$$

$$\text{s.t. } dx = (C\delta_x + D\delta_u) dt + \sigma(x^*) dz,$$

locally approximates the solution² to (14.5.11) if $\sigma(x) = 0$. Kydland and Prescott (1982) later describes the same procedure but only for the case where f is linear; this linearity restriction implies that the Hessian of $f(x, u)$ is zero, reducing Q to just a linear-quadratic approximation of π near the steady state. In general, however, the quadratic approximation of π is not adequate.

2. Magill further describes how to take this linear approximation, form a linear stochastic model, and compute the implied spectral properties of the linear approximation. He proposed taking these spectral results and comparing them to the spectral properties of actual data. Versions of this proposal, usually limited to variance-covariance comparisons, have been used frequently in business cycle analysis. Kydland and Prescott (1982) is an early example of this strategy.

Magill's method differs from the perturbation method in that the idea here is to replace the nonlinear problem with a linear-quadratic problem, whereas the perturbation approach focuses on computing derivatives of the nonlinear problem. In fact Kydland and Prescott, in their implementation of this approach, do not claim to be computing those derivatives. However, Magill's procedure does produce the same linear approximation as we did in section 14.3. Many "linearization" procedures have been offered for dynamic economic models, with the literature making little effort to sort out the differences. Some are probably equivalent with the perturbation approach of mathematical literature, but some are not. In particular, McGrattan (1990) proposed an alternative approach that differs substantially from perturbation.³ First, she also computes the steady state, (x^*, u^*) of the optimal path. Second, she replaces $\pi(x, u)$ with its quadratic expansion around $x = x^*$ and $u = u^*$, and replaces $f(x, u)$ with its linear expansion around $x = x^*$ and $u = u^*$. She then examines the linear-quadratic problem

$$\begin{aligned} \max_c \int_0^\infty & e^{-\rho t} (\pi(z^*) + \pi_z(z^*)(z - z^*) + \frac{1}{2}(z - z^*)^\top \pi_{zz}(z^*)(z - z^*)) dt \\ \text{s.t. } & \dot{x} = f(z^*) + f_z(z^* - z), \end{aligned} \tag{14.5.12}$$

where $z \equiv (x, u)$. This approach is one interpretation of the idea to approximate a nonlinear problem by "linearizing the law of motion and using a quadratic approximation of the objective function near the steady state."

Despite the similarity, the answers from (14.5.12) are substantially different from the answers computed by perturbation methods and Magill's method. To see this, apply (14.5.12) to the simple growth problem

$$\max_c \int_0^\infty e^{-\rho t} u(c) dt$$

$$\text{s.t. } \dot{k} = f(k) - c.$$

The resulting approximate policy function is $C(k) = f(k^*) + \rho(k - k^*)$, which is not the true linear approximation of C at k^* and implies behavior very different from that implied by the true solution; in particular, the approximation implies that the rate of convergence to the steady state is zero. The problem is that (14.5.12) leaves

3. We discuss the continuous-time version of McGrattan, but there is no mathematically substantive difference between the continuous- and discrete-time versions as far as these issues are concerned.

out the terms in Magill's Q that arise from the curvature of f at the steady state. While this does replace the nonlinear problem with a "similar" linear-quadratic one, it may produce a policy function substantially different from the policy function of the initial nonlinear problem. Also the results are sensitive to changes in the formulation. If, instead of letting c be the control, we choose $I = k$ to be the control and define $c = f(k) - I$, then the results of (14.5.12) will agree with Magill's approach.

This example illustrates the hazards of following an ad hoc approach. The advantage of regular perturbation methods based on an implicit function formulation is that one directly computes the Taylor expansions in terms of whatever variables one wants to use, and that expansion is the best possible asymptotically.

Another important distinction between Magill, McGrattan, and Kydland and Prescott, on the one hand, and the regular perturbation approach, is the way uncertainty is handled. These linear-quadratic approximations ignore the $C_\sigma(k)$ terms computed in section 14.5 and hence are at best only "half-linear" approximations. That is fine for computing the spectrum of consumption, income, and other controls and states (e.g., as is done in Magill), since the C_σ correction disappears in the differencing. However, if one were to use this method to match spectra of, say, asset prices, then there may be problems, for higher-order derivatives of u and f and the value of C_σ may affect the linear behavior of the variables being studied.

Furthermore the motivation behind the quadratic approximation approach is a conceptual dead end if one wants to go beyond this level of approximation. This way of approaching the problem may lead one to think that the way to do a third-order approximation would be to take a third-order polynomial approximation around the deterministic steady state and solving that control problem. Of course there is no exact solution for third-order problems, possibly leading one to think that a third-order approximation is not possible. The approach in this chapter does not lead to any such conceptual blocks because it motivates all calculations from a Taylor series expansion point of view and in fact demonstrates that they are *easier*.

The only assumption made by regular perturbation methods is that the problems are sufficiently differentiable⁴ so that the implicit function theorem (or its Banach space generalization) applies. While it is advisable to check for the applicability of the implicit function theorem, we do not do that here; the reader is, however, assured that the examples in this chapter can be so justified. In the next chapter we will see some cases where the implicit function theorem does not hold.

4. Differentiability is also implicitly assumed in the ad hoc approaches.

14.6 Further Reading and Summary

This and the previous chapter have presented the general approach of regular perturbation methods for approximating implicitly defined functions. These methods are often used in economics for qualitative analysis, but we show that they also have substantial numerical potential. The mathematical foundations for perturbation methods tell us that they produce good approximations locally. Furthermore we can test any resulting expansion to determine the range over which it is reliable, using our basic “compute but verify” approach.

Regular perturbation methods are based on implicit function theorems. See Zeidler (1986), Krasnosel'skii and Zabreiko (1984), and Aubin and Ekeland (1984) for Banach space versions of implicit function theorems.

The applications to optimal control problems is based on the results in Fleming (1971), Fleming and Souganides (1986), and Bensoussan (1988). Specific applications in the economics literature include Albrecht et al. (1991) and Caputo (1990a, 1990b); Anderson (1993) contains several Mathematica programs for perturbation methods. Dotsey and Mao (1992) evaluates some linearization methods. Judd (1996) reviews economic applications of perturbation methods. The local, certainty equivalent, approximation approach developed in Magill (1977) has been used extensively in real business cycle analyses, but the higher-order procedures have not been used much in economics.

Exercises

1. Assume an exchange economy with two agents with CES utility over two goods. Compute the degree 4 Taylor series for equilibrium based at points where the agents have the same utility and endowment. Repeat for three-agent economies. Compare with nonlinear equation solutions.
2. Compute the degree 1, 2, 3, and 4 approximations of solutions to exercise 10.8 around the steady state. Compare the accuracy of the Taylor series approximations to the answers computed there. Suppose that we change the laws of motion to

$$dk_c = I_c dt + \varepsilon \sigma_c dz_c,$$

$$dk_I = I_I dt + \varepsilon \sigma_I dz_I,$$

where dz_c and dz_I are uncorrelated white noise processes. Compute the first certainty non-equivalence term for the investment policy functions.

3. Compute the quadratic expansion of (14.4.1) around the steady state for $\pi(x, u) = u^{1/2} + 0.01x^{1/2}$, $\beta = 0.9$, and $F(x, u) = x^{0.25} - u$

15 Advanced Asymptotic Methods

In this chapter we continue our examination of asymptotic methods of approximation. Regular perturbation methods for constructing a function $y = f(x)$ such that $H(x, f(x)) = 0$ near $y_0 = f(x_0)$ are successful when Taylor series computations and simple versions of the implicit function theorem (IFT) are applicable. These methods are not always successful; in particular, the implicit function theorem fails when the Jacobian matrix $H_y(x_0, y_0)$ is singular, and some functions do not possess simple power series expansions. In some cases special techniques can handle these problems and be used to produce approximations similar to Taylor expansions. This chapter presents elementary examples using some of these methods.

Sometimes the IFT fails because there are multiple solutions to $H(x, f(x))$ at (x_0, y_0) . Section 15.1 presents bifurcation methods that can often handle such conditions. Section 15.2 applies these methods to portfolio problems. Sometimes implicit solutions are unique, but there does not exist a simple power series expansion. However, power series are special in that they use only integral powers. Section 15.3 introduces the general concepts of a *gauge system* and an *asymptotic approximation*, and section 15.4 introduces the method of undetermined gauges and shows how to use it to solve a simple adverse selection problem.

Economic analyses often find it convenient to assume that uncertainty is small, and therefore would naturally like to compute Taylor expansions around the deterministic case; unfortunately, Taylor expansions often do not exist for such integrals. In section 15.5 we introduce an approach that can approximate such “small noise” integrals.

The final portion of this chapter shows how we can combine the projection and perturbation methods to arrive at hybrid procedures. Both projection and perturbation methods produce a series to approximate a function of interest, but they arrive at this similar condition in very different ways. In hybrid methods, one combines the complementary strengths of perturbation and projection methods to arrive at a hybrid series that does better than either projection or perturbation methods do alone. We outline the general idea and discuss a simple example.

15.1 Bifurcation Methods

Suppose that we want to approximate the function $h(\varepsilon)$ implicitly by $H(h(\varepsilon), \varepsilon) = 0$ at $x_0 = h(0)$ but that the conditions of the IFT, theorem 13.1.1, do not hold; in particular, $H_x(x_0, 0)$ may be singular. Such points may be bifurcation points and can be handled by bifurcation methods, to which we now turn.

The key step comes from l'Hospital's rule. If the assumptions in theorem 13.1.1 hold, then differentiation of the implicit expression $H(h(\varepsilon), \varepsilon) = 0$ with respect to ε

produces the equation

$$h'(0) = -\frac{H_\varepsilon(x_0, 0)}{H_x(x_0, 0)}.$$

But, if $H_x(x_0, 0) = 0$, $h'(0)$ has the form $0/0$ at $x = x_0$. l'Hospital's rule then implies

$$h'(0) = -\frac{H_{\varepsilon\varepsilon}(x_0, 0)}{H_{\varepsilon x}(x_0, 0)}.$$

One way to view the process below is that we first find an x_0 such that $H_x(x_0, 0) = 0$, and then apply l'Hospital's rule to compute $h'(0)$.

The formal approach to these calculations uses ideas from bifurcation theory. We first saw the idea of bifurcations in our discussion of branch points in homotopy methods in chapter 5. Here we will define the notion formally.

DEFINITION Suppose that $H: X \times R \rightarrow X$ and $H(x^0, \varepsilon_0) = 0$. Then (x^0, ε_0) is a *bifurcation point* if there exist two sequences (y^n, ε_n) and (z^n, ε_n) such that $H(y^n, \varepsilon_n) = H(z^n, \varepsilon_n) = 0$, $\lim_{n \rightarrow \infty} y^n = \lim_{n \rightarrow \infty} z^n = x^0$ and $\lim_{n \rightarrow \infty} \varepsilon_n = \varepsilon_0$, but $y^n \neq z^n$ for all n .

DEFINITION A function $f: R^n \rightarrow R^m$ is *diffeomorphic* to $g: R^n \rightarrow R^m$ at $z = z_0$ iff there exists a differentiable function $h: R^n \rightarrow R^n$ which is invertible at $z = z_0$ and $f(h(z)) = g(z)$ in some open neighborhood of $z = z_0$.

Theorem 15.1.1 shows how we can extend the implicit function theorem near bifurcation points.

THEOREM 15.1.1 (Bifurcation theorem on R) Suppose that $H: R \times R \rightarrow R$ and $H(x, \varepsilon) = 0$ for all x if $\varepsilon = 0$. Furthermore suppose that

$$H_x(x_0, 0) = 0 = H_\varepsilon(x_0, 0), \quad H_{x\varepsilon}(x_0, 0) \neq 0, \tag{15.1.1}$$

for some $(x_0, 0)$. Then

- i. If $H_{\varepsilon\varepsilon}(x_0, 0) \neq 0$, there is an open neighborhood \mathcal{N} of $(x_0, 0)$ and a function $h(\varepsilon), h(\varepsilon) \neq 0$ for $\varepsilon \neq 0$, such that $H(h(\varepsilon), \varepsilon) = 0$ and locally $H(x, \varepsilon)$ is diffeomorphic to $\varepsilon(\varepsilon - x)$ or $\varepsilon(\varepsilon + x)$ on \mathcal{N} .
- ii. If $H_{\varepsilon\varepsilon}(x_0, 0) = 0 \neq H_{\varepsilon\varepsilon\varepsilon}(x_0, 0)$, then there is an open neighborhood \mathcal{N} of $(x_0, 0)$ and a function $h(\varepsilon), h(\varepsilon) \neq 0$ for $\varepsilon \neq 0$, such that $H(h(\varepsilon), \varepsilon) = 0$ and $H(x, \varepsilon)$ is diffeomorphic to $\varepsilon^3 - xe$ or $\varepsilon^3 + xe$ on \mathcal{N} .
- iii. In both cases, $(x_0, 0)$ is a bifurcation point.

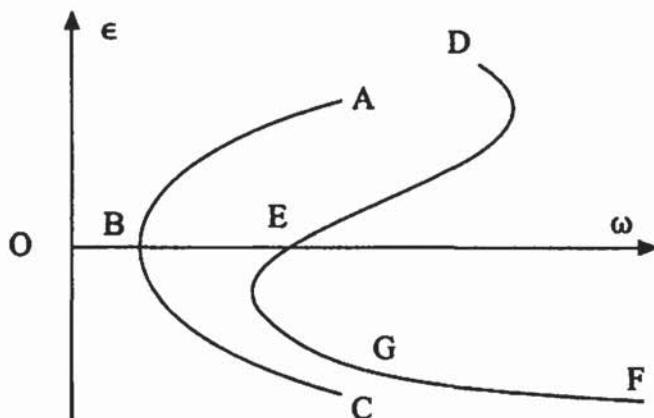


Figure 15.1
Bifurcation points

Figure 15.1 shows both kinds of bifurcations. Suppose that the curve ABC together with the ω axis is the zero set of $H(\omega, \varepsilon)$; then B is a bifurcation point of the $\varepsilon^3 - \omega\varepsilon$ type of bifurcation, called a *pitchfork bifurcation*. Similarly, if the curve $DEGF$ together with the ω axis is a zero set, then E is a bifurcation point of $\varepsilon^2 - \omega\varepsilon$ type of bifurcation, called a *transcritical bifurcation*.

We also have a multidimensional version of theorem 15.1.1. In this case we are searching for a one-dimensional path in (x, y) space that satisfies the implicit relation $H(y, x, \varepsilon) = 0$.

THEOREM 15.1.2 (Bifurcation theorem on R^n) Suppose that $H: R^m \times R^n \times R \rightarrow R^n$ and $H(y^0, x, \varepsilon) = 0$ for all x if $\varepsilon = 0$. Furthermore suppose that for some $(y^0, x^0, 0)$, H is analytic in a neighborhood of $(y^0, x^0, 0)$, $H_x(y^0, x^0, 0) = 0_{n \times n}$, $H_\varepsilon(y^0, x^0, 0) = 0_n$, and $\det(H_{xx}(y^0, x^0, 0)) \neq 0$. Then there is an open neighborhood \mathcal{N} of $(y^0, x^0, 0)$ and a locally analytic function $h(\varepsilon) \equiv (h^y(\varepsilon), h^x(\varepsilon))$, $h(\varepsilon) \neq 0$ for $\varepsilon \neq 0$, such that $H(h^y(\varepsilon), h^x(\varepsilon), \varepsilon) = 0$ on \mathcal{N} .

The proofs of these theorems rely on results in analytic bifurcation theory; see Zeidler (1986). Generalizations of these results for x , y , and ε in Banach spaces exist, and they will surely be useful in future developments but are beyond the scope of this text.

15.2 Portfolio Choices for Small Risks

The basic portfolio problem is a good example where bifurcation methods can be used. Suppose that an investor has W in wealth to invest in two assets. The safe asset yields R per dollar invested and the risky asset yields Z per dollar invested. If

a proportion ω of his wealth is invested in the risky asset, final wealth is $Y = W((1 - \omega)R + \omega Z)$. We assume that he chooses ω to maximize $E\{u(Y)\}$ for some concave, von Neumann-Morgenstern utility function $u(\cdot)$.

We want to “linearize” around the deterministic case of $\varepsilon = 0$. The first problem we encounter is that if we eliminate risk by replacing Z with its mean, \bar{Z} , the resulting problem is unbounded if $R \neq \bar{Z}$ and indeterminate if $R = \bar{Z}$. Since the former case is untenable, we opt for the latter. We create a continuum of portfolio problems by assuming

$$Z = R + \varepsilon z + \varepsilon^2 \pi, \quad (15.2.1)$$

where $E\{z\} = 0$. At $\varepsilon = 0$, Z is degenerate and equal to R . If $\pi > 0$, we model the intuitive case of risky assets paying a premium. Note that we multiply z by ε and π by ε^2 . Since $\text{var}\{\varepsilon z\} = \varepsilon^2 \sigma_z^2$, this models the standard result in finance that risk premia are roughly proportional to variance.

We now investigate the collection of portfolio problems indexed by ε in (15.2.1). The first-order condition for ω , after dividing by εW , is, for all ε , equivalent to

$$0 = E\{u'(WR + \omega W(\varepsilon z + \varepsilon^2 \pi)) (z + \varepsilon \pi)\} \equiv G(\omega, \varepsilon). \quad (15.2.2)$$

We can divide by εW to arrive at (15.2.2), since this condition defines the solution to the decision problem even when $\varepsilon = 0$. We know from concavity of $u(c)$ that there is a unique solution to (15.2.2) for ω if $\varepsilon \neq 0$. However, at $\varepsilon = 0$, ω can be anything since the two assets are perfect substitutes. The indeterminacy of ω at $\varepsilon = 0$ follows from the fact that

$$0 = G(\omega, 0), \quad \forall \omega. \quad (15.2.3)$$

We want to solve for $\omega(\varepsilon)$ as a Taylor series in ε . We don’t know if we can, but suppose that we can. Implicit differentiation implies that

$$0 = G_\omega \omega' + G_\varepsilon. \quad (15.2.4)$$

In (15.2.4) we differentiate G , an integral. We would like to pass the derivative operator through the integral. To do this, we need to recall the following result from integration theory:

LEMMA 15.2.1 Let $f(x, y): R \times R \rightarrow R$ be differentiable in y , and define $F(y) = \int_a^b f(x, y) dx$. If $f_y(x, y)$ is bounded in an open neighborhood of $[a, b] \times y_0$, then $F(y)$ is differentiable at y_0 and

$$\frac{dF}{dy}(y_0) = \int_a^b \frac{df}{dy}(x, y_0) dx.$$

For now, we will assume that lemma 15.2.1 applies. In that case we find

$$G_\varepsilon = E\{u''(Y) W(\omega z + 2\omega\varepsilon\pi)(z + \varepsilon\pi) + u'(Y)\pi\}, \quad (15.2.5)$$

$$G_\omega = E\{u''(Y) W(z + \varepsilon\pi)^2\varepsilon\}. \quad (15.2.6)$$

At $\varepsilon = 0$, $G_\omega = 0$ for all ω . This implies that at no point $(\omega, 0)$ can we apply the implicit function theorem to (15.2.3) to solve for $\omega'(0)$. Moreover we do not know $\lim_{\varepsilon \rightarrow 0} \omega(\varepsilon)$. Therefore, following theorem 15.1.1, we look for a bifurcation point, ω_0 , satisfying

$$0 = G_\varepsilon(\omega_0, 0) \quad (15.2.7)$$

which is one of the conditions in (15.1.1) for theorem 15.1.1. At $\varepsilon = 0$, (15.2.5) reduces to $0 = u''(RW) \omega_0 \sigma_z^2 W + u'(RW)\pi$ which implies that

$$\omega_0 = -\frac{\pi}{\sigma_z^2} \frac{u'(RW)}{Wu''(RW)}. \quad (15.2.8)$$

Now that we have our candidate for a bifurcation point, we need to check the boundedness condition in lemma 15.2.1 for the derivatives in (15.2.5)–(15.2.6). This depends on the nature of u and the density of z . For example, if z were log normal, as long as u' and u'' do not have exponential growth that offsets the normal density's tail behavior, lemma 15.2.1 applies. This is not an innocent assumption, however. For example, suppose that the solution (15.2.8) implies that consumption is negative in some states (as may occur if one shorts the safe asset to increase his position in the risky asset), and that $u'(0) = \infty$. Then the integrands in (15.2.5)–(15.2.6) will not be bounded, and the solution (15.2.8) is not valid. The approach we are using is to proceed as if lemma 15.2.1 holds, and then check its validity in a neighborhood of the candidate point (15.2.8).

An alternative is to assume that z has support $[-\varepsilon, \varepsilon]$ and $u''(RW) \neq \infty$; this is the approach Samuelson (1970) took. However, we will sometimes want to allow z with infinite support. Then we will need to assume that Y avoids singularities of u and its derivatives; it may be necessary to assume that $u''(c)$ is defined for all c , even $c < 0$. We will face the same problem below when we take higher-order derivatives of G , where again, we will have to check that lemma 15.2.1 applies. The basic fact is that we will be assuming that the dominant considerations are the properties of the utility function near RW and the moments of z , and that we can pass differentiation through the integral sign. This is strictly more general than the Samuelson approach.

Now that we have found a candidate bifurcation point, we can continue to derive the Taylor series. The formula (15.2.8) is the simple portfolio rule from

linear-quadratic analysis, indicating that ω is the product of risk tolerance and the risk premium per unit variance. However, ω_0 is not an approximation to the portfolio choice at any particular variance. Instead, ω_0 is the limiting portfolio share as the variance vanishes. If we want the linear and quadratic approximations of $\omega(\varepsilon)$ at $\varepsilon = 0$, we must go further, since the quadratic approximation to $\omega(\varepsilon)$ is $\omega(\varepsilon) \doteq \omega(0) + \varepsilon\omega'(0) + (\varepsilon^2/2)\omega''(0)$.

To calculate $\omega'(0)$ and $\omega''(0)$, we need to do two more rounds of implicit differentiation. If we differentiate (15.2.4) with respect to ε , we find that

$$0 = G_{\omega\omega} \omega' \omega' + 2G_{\omega\varepsilon} \omega' + G_\omega \omega'' + G_{\varepsilon\varepsilon},$$

where (without loss of generality, we assume that $W = 1$)

$$G_{\varepsilon\varepsilon} = E\{u'''(Y)(\omega z + 2\omega\varepsilon\pi)^2(z + \varepsilon\pi) + u''(Y)2\omega\pi(z + \varepsilon\pi) + 2u''(Y)(\omega z + 2\omega\varepsilon\pi)\pi\},$$

$$G_{\omega\omega} = E\{u'''(Y)(z + \varepsilon\pi)^3\varepsilon\},$$

$$G_{\omega\varepsilon} = E\{u'''(Y)(\omega z + 2\omega\varepsilon\pi)(z + \varepsilon\pi)^2\varepsilon + u''(Y)(z + \varepsilon\pi)2\pi\varepsilon + u''(Y)(z + \varepsilon\pi)^2\}.$$

At $\varepsilon = 0$, $G_{\varepsilon\varepsilon} = u'''(R)\omega_0^2 E\{z^3\}$, $G_{\omega\omega} = 0$, and $G_{\omega\varepsilon} = u''(R) E\{z^2\} \neq 0$. Therefore theorem 15.1.1 applies and

$$\omega' = -\frac{1}{2} \frac{u'''(R)}{u''(R)} \frac{E\{z^3\}}{E\{z^2\}} \omega_0^2. \quad (15.2.9)$$

Equation (15.2.9) is a simple formula. It shows that as riskiness increases, the change in ω depends on u'''/u'' and the ratio of skewness to variance. If u is quadratic or z is symmetric, ω does not change to a first order. We could continue this and compute more derivatives of $\omega(\varepsilon)$ as long as u is sufficiently differentiable.

We end the development of this example here. However, it is clear we could do much with this. We computed the asset demand function in terms of ε . We could similarly expand supply functions as well as equilibrium prices and trades. Our asset demand example is just one simple application.

15.3 Gauge Functions and Asymptotic Expansions

In the previous chapter we computed asymptotic expansions of the form $\sum_{i=1}^n a_i \varepsilon^i$. In this chapter we will examine cases where this may not be valid. To deal with these cases, we use a more general formulation of expansions. A system of *gauge functions* is a sequence of functions, $\{\delta_n(\varepsilon)\}_{n=1}^\infty$, such that

$$\lim_{\varepsilon \rightarrow 0} \frac{\delta_{n+1}(\varepsilon)}{\delta_n(\varepsilon)} = 0. \quad (15.3.1)$$

The terminology is appropriate because we will “gauge” the asymptotic behavior of $f(x)$ against the asymptotic behavior of the gauge functions. An *asymptotic expansion* of $f(x)$ near $x = 0$ is any expansion $f(0) + \sum_{i=1}^n a_i \delta_i(x)$ where, for each $k < n$,

$$\lim_{x \rightarrow 0} \frac{f(x) - (f(0) + \sum_{i=1}^k a_i \delta_i(x))}{\delta_k(x)} = 0. \quad (15.3.2)$$

The notation

$$f(x) \sim f(0) + \sum_{i=1}^n a_i \delta_i(x) \quad (15.3.3)$$

expresses the asymptotic relations in (15.3.2).

In regular perturbations and Taylor series expansions using the parameter ε , the sequence of gauge functions is $\delta_k(\varepsilon) = \varepsilon^k$. This is not the only possible set of gauge functions. For example, $\delta_k(x) = x^{k/2}$ is also a gauge system. Some functions have asymptotic expansions which are not Taylor expansions. For example, $e^{x^{1/3}}$ does not have a Taylor expansion around $x = 0$ because its derivative, $x^{-2/3} e^{x^{1/3}}/3$, is not finite at $x = 0$. However, the series

$$e^{x^{1/3}} \sim 1 + x^{1/3} + \frac{1}{2}(x^{1/3})^2 + \frac{1}{6}(x^{1/3})^3 + \dots \quad (15.3.4)$$

is an asymptotic expansion around $x = 0$. The crucial limitation of Taylor series is that they necessarily use the gauge functions x^k for positive integers k . There is no expansion of $e^{x^{1/3}}$ in that gauge, but (15.3.4) is asymptotic in the gauge $\delta_k(x) = x^{k/3}$. The methods behind asymptotic expansions are substantively important generalizations of the basic idea behind Taylor series expansions

15.4 Method of Undetermined Gauges

In our portfolio problem we assumed that the risky return depended on ε according to the form $R + \varepsilon z + \varepsilon^2 \pi$. This is only one way to parameterize a continuous collection of portfolio problems. We could have written $R + \varepsilon z + \varepsilon \pi$ to represent a continuum of portfolio problems. The difference is economically substantive: The choice we made implies a risk premium, $\varepsilon^2 \pi$, that is quadratic in ε and hence proportional to the variance, whereas the $R + \varepsilon z + \varepsilon \pi$ alternative implies a risk premium proportional

to ε and the standard deviation. The reason we chose $R + \varepsilon z + \varepsilon^2 \pi$ was the basic portfolio theory result that the risk premium should be proportional to variance, not standard deviation.

In general, any asymptotic problem involves a choice of gauge functions. We chose the parameterization (15.2.1) and used the gauge family ε^k to find a Taylor series for $\omega(\varepsilon)$. We could use the parameterization $Z = R + \sqrt{\varepsilon}z + \varepsilon\pi$ and use the gauge family $\varepsilon^{k/2}$ for an expansion of $\omega(\varepsilon)$ near $\varepsilon = 0$, since those two combinations are equivalent. We will see below that we cannot use the parameterization $R + \varepsilon z + \varepsilon\pi$ and the gauge family ε^k . We must choose a gauge family that is appropriate for the parameterization.

The portfolio problem was a special case where we had good intuition. In some analyses we may not have such strong guidance in choosing a combination of parameterization and gauge family. We then must determine the appropriate gauge as well as the coefficients. We will first discuss how bad combinations of parameterizations and gauge families would be detected in the portfolio problem. Then we analyze an adverse selection problem, endogenously determining the appropriate gauge as we construct an asymptotic expansion. These examples will show that we can often determine the correct gauge for a parameterization even if we don't begin the analysis with a good idea.

Alternative Gauges in the Portfolio Problem

Say we did not know that risk premia were proportional to variance. Given this lack of information, we might (wisely) assume the more general functional form

$$Z = R + \varepsilon z + \varepsilon^\nu \pi, \quad (15.4.1)$$

where ν is left free and attempt to compute an asymptotic expansion similar to (15.4.1). The general idea is to allow ν to be free until we need to fix it to compute an asymptotic expression. With the parameterization in (15.4.1), the first-order condition becomes (after dividing by ε)

$$0 = E\{u'(R + (\varepsilon z + \varepsilon^\nu \pi)\omega)(z + \varepsilon^{\nu-1}\pi)\} \equiv G(\omega, \varepsilon; \nu) \quad (15.4.2)$$

for all ε . Again $0 = G(\omega, 0; \nu)$ for all ω and $\nu > 0$. Implicit differentiation of (15.4.2) again implies that

$$0 = G_\omega \omega' + G_\varepsilon, \quad (15.4.3)$$

but now (with $Y \equiv R + (\varepsilon z + \varepsilon^\nu \pi)\omega$)

$$G_\varepsilon = E\{u''(Y)(\omega z + \nu \omega \varepsilon^{\nu-1} \pi)(z + \varepsilon^{\nu-1} \pi) + u'(Y) \varepsilon^{\nu-2} (\nu - 1) \pi\}, \quad (15.4.4)$$

$$G_\omega = E\{u''(Y)(z + \varepsilon^{\nu-1} \pi)^2 \varepsilon^{\nu-1}\}. \quad (15.4.5)$$

At a bifurcation point where $\omega = \omega_0$, we must have

$$0 = \lim_{\varepsilon \rightarrow 0} G_\varepsilon(\omega_0, \varepsilon; \nu). \quad (15.4.6)$$

We saw that if $\nu = 2$, we can use the bifurcation theorem to construct an asymptotic expansion for $\omega(\varepsilon)$. We now show what would go wrong if we choose some other ν .

If we choose $\nu = 1$, the implicit function condition (15.4.3) reduces to $0 = (\omega' + \omega) E\{u''(Y)z^2\}$ for all ε . Since $E\{u''(Y)z^2\} < 0$, this implies the differential equation $0 = \omega' + \omega$ for all ε . If $\nu = 1$, (15.4.6) reduces to $0 = \omega_0(\sigma_z^2 + \pi^2)u''(R)$ which implies that $\omega_0 = 0$, which in turn serves as an initial condition to the differential equation $0 = \omega' + \omega$. Hence $\omega(\varepsilon) = 0$ for all ε , a result we know to be false since $\omega = 0$ satisfies the first-order condition (15.4.2) for $\varepsilon > 0$ only if $\pi = 0$. If $1 \neq \nu < 2$, the dominant term in G_ε in (15.4.4) as $\varepsilon \rightarrow 0$ is $\varepsilon^{\nu-2}(\nu - 1)\pi E\{u'(Y)\}$ which diverges as $\varepsilon \rightarrow 0$, and this is inconsistent with (15.4.6). Any $\nu > 2$ will not work because

$$\lim_{\varepsilon \rightarrow 0} G_\varepsilon = E\{u''(Y)\omega z^2\} = 0$$

implies that $\omega_0 = 0$, which, after further differentiation of (15.4.2), will imply an infinite leading coefficient of the expansion.

These calculations show that the only way for the condition $G_\varepsilon(\omega_0, 0; \nu) = 0$ to tie down ω_0 is for $\nu = 2$. These calculations show us that a bad choice for a gauge will often make its inappropriateness apparent. Through this kind of analysis, we can determine the gauges to be used in an asymptotic expansion. The next section pursues this idea in a well-known model of asymmetric information.

An Adverse Selection Example

The portfolio problem is a bit too simple, since we intuitively knew the correct gauge. We next apply the method of undetermined gauges to analyze a simple adverse selection problem, a problem where the correct gauge is not obvious.

Recall the RSW model of adverse selection described in chapter 4. Assume that the high (low) risk type receives observable income of 1 with probability $q(p)$, $q < p$, and zero otherwise. In the Nash equilibrium (if it exists) the high-risk type consumes q in each state, and the low-risk type consumes x in the good state, and y otherwise. The state-contingent consumptions x and y are fixed by the incentive compatibility

condition $u(q) = p u(x) + (1 - p) u(y)$, and the zero-profit condition, $y = p(1 - x)/(1 - p)$. Substitution yields the single equation for x :

$$u(q) = pu(x) + (1 - p) u\left(\frac{p(1 - x)}{1 - p}\right). \quad (15.4.7)$$

If $q = p$, then $x = y = p = q$ is the unique solution. We seek solutions to (15.4.7) for $q = p - \varepsilon$ where $\varepsilon > 0$ is small, modeling situations where the risk differences are small. The solution, x , will depend on ε , and it is implicitly defined by

$$u(p - \varepsilon) = pu(x(\varepsilon)) + (1 - p)u\left(\frac{p(1 - x(\varepsilon))}{1 - p}\right). \quad (15.4.8)$$

We first attempt a standard regular perturbation approach. Suppose that $x(\varepsilon) \sim p + \sum_{i=1}^{\infty} \alpha_i \varepsilon^i$. A Taylor series expansion of (15.4.8) around $\varepsilon = 0$ implies that

$$\begin{aligned} u(p) - u'(p)\varepsilon + \frac{1}{2}u''(p)\varepsilon^2 + \dots \\ = p \left(u(p) + u'(p)(\alpha_1\varepsilon + \alpha_2\varepsilon^2 + \dots) + \frac{1}{2}u''(p)(\alpha_1\varepsilon + \alpha_2\varepsilon^2 + \dots)^2 + \dots \right) \\ + (1 - p) \left(u(p) + u'(p) \left(\frac{-p}{1-p} \right) (\alpha_1\varepsilon + \alpha_2\varepsilon^2 + \dots) \right. \\ \left. + \frac{1}{2} \left(\frac{p}{1-p} \right)^2 u''(p) (\alpha_1\varepsilon + \alpha_2\varepsilon^2 + \dots)^2 + \dots \right). \end{aligned}$$

Combining like terms, we find that

$$\begin{aligned} u(p) - u'(p)\varepsilon + \frac{1}{2}u''(p)\varepsilon^2 + \dots \\ = u(p)(p + (1 - p)) + \varepsilon(p\alpha_1 u'(p) - p\alpha_1 u'(p)) \\ + \varepsilon^2(p\alpha_2 u'(p) + \frac{1}{2}p\alpha_1 u''(p) - p\alpha_2 u'(p) + \dots) + \dots, \end{aligned}$$

which implies that $-u'(p)\varepsilon + \frac{1}{2}u''(p)\varepsilon^2 = 0 + \mathcal{O}(\varepsilon^2)$, but this cannot be true as $\varepsilon \rightarrow 0$, since $u'(p) \neq 0$.

The problem is that regular perturbation *assumes* that there is an asymptotic expansion of x of the form $\sum_{i=1}^{\infty} \alpha_i \varepsilon^i$. Suppose instead that

$$x(\varepsilon) \sim p + \sum_{i=1}^{\infty} \alpha_i \varepsilon^{v_i} \quad (15.4.9)$$

for some increasing sequence of reals, v_i . We will now determine this sequence as part of the asymptotic expansion. We call this the *method of undetermined gauges*. In regular perturbation we know the gauge functions and need to determine only the coefficients. Here we must determine both.

When we substitute (15.4.9) into the Taylor expansion of (15.4.8), we get

$$\begin{aligned} u(p) - u'(p)\varepsilon + \frac{1}{2}u''(p)\varepsilon^2 \\ = p \left(u(p) + u'(p)(\alpha_1\varepsilon^{v_1} + \alpha_2\varepsilon^{v_2} + \dots) + \frac{1}{2}u''(p)(\alpha_1\varepsilon^{v_1} + \alpha_2\varepsilon^{v_2} + \dots)^2 + \dots \right) \\ + (1-p) \left(u(p) + u'(p) \left(-\frac{p}{1-p} \right) (\alpha_1\varepsilon^{v_1} + \alpha_2\varepsilon^{v_2} + \dots) \right. \\ \left. + \frac{1}{2}u''(p) \left(\frac{p}{1-p} \right)^2 (\alpha_1\varepsilon^{v_1} + \alpha_2\varepsilon^{v_2} + \dots)^2 + \dots \right) \end{aligned} \quad (15.4.10)$$

Combining like terms yields

$$\begin{aligned} -u'(p)\varepsilon + \frac{1}{2}u''(p)\varepsilon^2 + \dots \\ = \frac{1}{2}u''(p)p(\alpha_1^2\varepsilon^{2v_1} + 2\alpha_1\alpha_2\varepsilon^{v_1+v_2} + \alpha_2^2\varepsilon^{2v_2} + \dots) \\ + \frac{1}{2}u''(p) \frac{p^2}{1-p} (\alpha_1^2\varepsilon^{2v_1} + 2\alpha_1\alpha_2\varepsilon^{v_1+v_2} + \alpha_2^2\varepsilon^{2v_2} + \dots) \end{aligned} \quad (15.4.11)$$

Since $0 < v_1 < v_2 < v_3 < \dots$, we know that $2v_1 < v_1 + v_2 < 2v_2$ and that $2v_2$ is less than the power of any ε term which is not in (15.4.11). Therefore the dominant term as $\varepsilon \rightarrow 0$ on the RHS is ε^{2v_1} , the next dominant is $\varepsilon^{v_1+v_2}$, and so on. If the two sides of (15.4.11) are to match, $\varepsilon = \varepsilon^{2v_1}$ and $2v_1 < v_1 + v_2$; but $\varepsilon = \varepsilon^{2v_1}$ only if $v_1 = \frac{1}{2}$ and $2v_1 < v_1 + v_2$ is guaranteed, since $v_2 > v_1$. Hence

$$0 = \varepsilon \left(u'(p) + \frac{1}{2}p\alpha_1^2 u''(p) + \frac{1}{2} \frac{p^2}{1-p} \alpha_1^2 u''(p) \right),$$

which implies that

$$\alpha_1 = \sqrt{-2 \frac{u'(p)}{u''(p)} \frac{(1-p)}{p}}. \quad (15.4.12)$$

The next terms are

$$\frac{1}{2} u''(p)\varepsilon^2 = u''(p)p\alpha_1\alpha_2\varepsilon^{v_1+v_2} + u''(p) \frac{p^2}{1-p}\alpha_1\alpha_2\varepsilon^{v_1+v_2}. \quad (15.4.13)$$

Since $v_1 = \frac{1}{2}$ and comparability of these terms requires that $2 = v_1 + v_2$, we conclude that $v_2 = 1\frac{1}{2}$ and that solving for α_2 is a linear problem. We could continue this indefinitely. By flexibly specifying all the parameters of an asymptotic expansion, including the v_i exponents, we can compute asymptotic expressions for a greater variety of problems.

An alternative procedure would be to compute asymptotic expansions of each term in (15.4.8), which here would just be the Taylor expansions. This results in

$$\begin{aligned} u(p) - \varepsilon u'(p) + \frac{\varepsilon^2}{2} u''(p) + \dots \\ = p \left(u(p) + u'(p)(x-p) + \frac{u''(p)(x-p)^2}{2} + \dots \right) \\ + (1-p) \left(u(p) - u'(p) \frac{p}{1-p}(x-p) + u''(p) \left(\frac{p}{1-p} \right)^2 \frac{(x-p)^2}{2} + \dots \right), \end{aligned}$$

which in turn implies that

$$-\varepsilon u'(p) + \dots = \frac{(x-p)^2}{2} u''(p) \frac{p}{1-p} + \dots \quad (15.4.14)$$

If we view the displayed terms in (15.4.14) as a quadratic equation and solve for x in terms of ε , we get $x \sim p + \sqrt{-2(u'(p)/u''(p))((1-p)/p)}\varepsilon^{1/2}$ which is the same as (15.4.12). While this second procedure seems simpler, it is not as clean. Notice that it forces one to solve a nonlinear equation, whereas the perturbation approach involves only linear equations. That is not a problem here, but with more variables it will be. In general, the earlier approach is more robust.

15.5 Asymptotic Expansions of Integrals

Integrals frequently take the form

$$\int_D f(x) e^{-\lambda g(x)} dx, \quad (15.5.1)$$

where λ is a large parameter and $g(x)$ is a positive function. In such cases it is useful to compute asymptotic expansions in λ which approximate (15.5.1) asymptotically as λ goes to infinity. Before considering the proper way to do this, we consider the Taylor series approach. If $\lambda = \infty$, then (15.5.1) is zero for “nice” $g(x)$, such as x^2 . Also, if we define $\varepsilon = \lambda^{-1}$, then $\varepsilon \rightarrow 0$ as $\lambda \rightarrow \infty$. This suggests defining $I(\varepsilon) \equiv \int_D f(x)e^{-g(x)/\varepsilon} dx$, noting that $I(0) = 0$, and applying Taylor’s theorem to $I(\varepsilon)$. The difficulty becomes apparent immediately. Assuming that we can pass differentiation through the integral, $I'(\varepsilon) = \int_0^\infty f(x)(g(x)/\varepsilon^2)e^{-g(x)/\varepsilon} dx$. Taking the limit of $I'(\varepsilon)$ as $\varepsilon \rightarrow 0$ will be difficult because the $e^{-g(x)/\varepsilon}$ term goes to zero, but the $g(x)/\varepsilon^2$ goes to infinity if $g(x) \neq 0$. Therefore the standard regular perturbation approach fails.

The basic idea we pursue here is that when λ is large the major contribution of the integrand in (15.5.1) is in the neighborhood of the point that maximizes $e^{-\lambda g(x)}$, which is the minimum of $g(x)$. Suppose that we can assume that $g(0) = 0$ and $g(x)$ is uniquely minimized at $x = 0$. For large λ , if $x \neq 0$, then $e^{-\lambda g(x)} \ll e^{-\lambda g(0)} = 1$. As long as $f(x)$ does not offset this for values of x away from 0, the integral (15.5.1) is determined largely by $e^{-\lambda g(x)}f(x)$ for x near 0.

Hermite Integrals

We begin our discussion of the asymptotic expansions of integrals by recalling special properties of integrals of the form

$$\int_{-\infty}^{\infty} f(x) e^{-\lambda x^2} dx. \quad (15.5.2)$$

These facts will help us understand why the asymptotic expansions which we examine below do as well as they do.

First, integrals of the form (15.5.2) integrate exactly when $f(x)$ is a polynomial. The general formulas for $\lambda > 0$ are

$$\int_{-\infty}^{\infty} x^{2n} e^{-\lambda x^2} dx = \frac{1 \cdot 3 \cdot 5 \cdots (2n-1)}{2^n \lambda^n} \sqrt{\frac{\pi}{\lambda}}. \quad (15.5.3)$$

Of course $\int_{-\infty}^{\infty} x^{2n+1} e^{-\lambda x^2} dx = 0$ for all integers $n \geq 0$.

Second, even if $f(x)$ is not a polynomial, we may be able to replace it with a Taylor series expansion around $x = 0$; this is the beginning of the Laplace approach. This results in the identity

$$\int_{-\infty}^{\infty} e^{-\lambda x^2} f(x) dx = \int_{-\infty}^{\infty} e^{-\lambda x^2} \left(f(0) + xf'(0) + \frac{x^2}{2} f''(0) + \cdots + R_n(x) \right) dx, \quad (15.5.4)$$

where $R_n(x)$ is the remainder term in the Taylor expansion of f . The polynomial terms in the Taylor series will integrate exactly. The remainder term, $R_n(x)$, will not generally integrate, but $R_n(x)$ is negligible for small x by construction, and for large x the $e^{-\lambda x^2}$ factor will crush $R_n(x)$ as long as $R_n(x)$ does not grow more rapidly than $e^{\lambda x^2}$. Since many reasonable functions satisfy this growth condition, a good approximation is formed from (15.5.4) by dropping $R_n(x)$. This approximation is the basic idea behind Laplace's approximation.

Portfolio Applications

We next discuss a portfolio application of the Laplacian approximation. Suppose that we want to evaluate the value of

$$\frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{\infty} (B + Se^{z+\mu})^\gamma e^{-z^2/2\sigma^2} dz, \quad (15.5.5)$$

which is the expected utility from holding B units of the numéraire and S units of an asset having a lognormal distribution with log mean μ and log variance σ . The integral (15.5.5) is, modulo a linear change of variables, the same kind as (15.5.1). The case of small σ here corresponds to a large λ in (15.5.1). If σ is small then the expectation in (15.5.5) is well-approximated by $(B + S)^\gamma$. In many cases it is reasonable to assume that σ is small and consider using an asymptotic method to compute expected utility. If we define $\lambda = (\sigma^2)^{-1}$, then (15.5.5) takes the form (15.5.1).

Table 15.1 shows that the approximation in (15.5.4) does very well when applied to (15.5.5). We choose $\gamma = -3$ and, for the sake of simplicity, set $B = S = 1$ and $\mu = 0$. We examine degree 1, 3, and 5 Taylor series approximations for $u(1 + e^z)$ and let $\sigma \in \{0.01, 0.1, 0.2, 0.4, 1.0\}$. For $\sigma = 0.01$ even the linear approximation of $u(1 + e^z)$ results in an excellent approximation of the integral. For larger σ the extra terms help. For $\sigma = 1$, even the fifth-order approximation does not produce a good approximation of the integral; of course $\sigma^{-1} = 1$ is a long way from $\sigma^{-1} = \infty$ which is the case where the approximation is exact.

Table 15.1
Errors in the Laplacian approximation of (15.5.5)

n	σ				
	0.01	0.1	0.2	0.4	1.0
1	2.5(-5)	2.5(-3)	9.8(-3)	3.7(-2)	1.7(-1)
3	-1.1(-9)	-1.2(-5)	-1.9(-4)	-2.9(-3)	-7.6(-2)
5	1.3(-10)	8.8(-8)	5.5(-6)	3.2(-4)	4.9(-2)

An Intuitive Derivation of Laplace's Formula

We will next apply these ideas to examine a more general problem. The following procedure illustrates the basic ideas and also generates the coefficients for Laplace's formula. The techniques used here may be useful to compute expansions that don't fit Laplace's method exactly. Also an ambitious reader could use the following procedure to derive multidimensional expansions.

Suppose that we want to approximate

$$I(\lambda) \equiv \int_{-\infty}^{\infty} e^{-\lambda g(x)} f(x) dx, \quad (15.5.6)$$

where $g(x)$ is nonnegative and minimized at $x = 0$, with $g''(0) > 0$. Using Taylor expansions around $x = 0$ for both $g(x)$ and $f(x)$ (we let f_n and g_n denote the n th derivatives of f and g), we approximate the integrand with

$$e^{-\lambda(g_0 + g_2 x^2/2 + g_3 x^3/6 + \dots)} \left(f_0 + f_1 x + \frac{f_2 x^2}{2} + \dots \right),$$

which we factor into three pieces:

$$(e^{-\lambda g_0})(e^{-\lambda g_2 x^2/2}) \left(e^{-\lambda(g_3 x^3/6 + \dots)} \left(f_0 + f_1 x + \frac{f_2 x^2}{2} + \dots \right) \right). \quad (15.5.7)$$

The first piece, $e^{-\lambda g_0}$, is a constant factor. The second is the normal density function with variance $(\lambda g_2)^{-1}$. The third piece, for large λ and small x , can be further approximated and replaced by a low-order Taylor series. We are then left with an integral of the form

$$e^{-\lambda g_0} \int_{-\infty}^{\infty} e^{-\lambda g_2 x^2/2} P(x, \lambda x^3) dx, \quad (15.5.8)$$

where $P(x, \lambda x^3)$ is a low-order polynomial representing the Taylor expansion of the third piece in (15.5.7) in terms of λx^3 and x . At this point we do not specify the order of the polynomial approximation in P ; we will see below how to determine that. The integral in (15.5.8) will be an asymptotically valid approximation of (15.5.6) as $\lambda \rightarrow \infty$ for two reasons. First, for large λ and $|x|$ not small, $e^{-\lambda g_2 x^2/2}$ is trivial and will crush the neglected terms of f . Second, when $|x|$ is small, the integrands in (15.5.8) and (15.5.6) are similar. The formal analysis of Laplace's method makes rigorous these approximation arguments. We will proceed formally.

Having replaced (15.5.6) with (15.5.8), we can just compute (15.5.8), an integral with an analytic solution. After integrating out the x variable we then combine like

powers of λ to compute the coefficients of like powers of λ in the final expansion. However, we should point out a complication due to the presence of λ in the exponential term $e^{-\lambda g_2 x^2/2}$. When we compute (15.5.8), we need to use the change of variables $y = \sqrt{\lambda} x$ to replace (15.5.8) with

$$\frac{e^{-\lambda g_0}}{\sqrt{\lambda}} \int_{-\infty}^{\infty} e^{-g_2 y^2/2} P\left(\frac{y}{\sqrt{\lambda}}, \frac{y^3}{\sqrt{\lambda}}\right) dy. \quad (15.5.9)$$

This shows that each power of λ will arise from two sources: First, each will arise directly in the second component in P , and second, they will less directly appear in the other components of P through the change of variables. Furthermore both the first and second components of P contains powers of $y/\sqrt{\lambda}$. This tells us that if we are to have an accurate expansion in terms of λ and its powers, we will need to include high-order terms in the original construction of P if we are going to get all the appropriate λ terms. Hence, for a two-term expansion, we need all terms which yield either $\lambda^{-1/2}$ or λ^{-1} terms after the change of variables in (15.5.9).

For the case of a two-term expansion, we let

$$P = \left(1 - \frac{\lambda g_3 x^3}{6} - \frac{\lambda g_4 x^4}{24} + \frac{\lambda^2 g_3^2 x^6}{72}\right) \left(f_0 + f_1 x + \frac{f_2 x^2}{2}\right).$$

With the change in variable in (15.5.9), P becomes

$$P = \left(1 - \frac{y^3 \lambda^{-1/2} g_3}{6} - \frac{\lambda^{-1} g_4 y^4}{24} + \frac{\lambda^{-1} g_3^2 y^6}{72}\right) \left(f_0 + f_1 y \lambda^{-1/2} + \frac{f_2 y^2 \lambda^{-1}}{2} + \dots\right),$$

which, when we multiply and keep only constant, $\lambda^{-1/2}$, and λ^{-1} terms, becomes

$$\begin{aligned} P = f_0 &+ \left(f_1 y - \frac{g_3 f_0 y^3}{6}\right) \lambda^{-1/2} \\ &+ \left(\frac{f_2 y^2}{2} - \frac{f_0 g_4 y^4}{24} - f_1 y^4 g_3 + \frac{f_0 g_3^2 y^6}{72}\right) \lambda^{-1} + \dots \end{aligned}$$

We then integrate with respect to y using (15.5.3) and arrive at the two-term Laplace's formula

$$I(\lambda) \sim \sqrt{\frac{2\pi}{\lambda g_2}} e^{-\lambda g_0} \left(f_0 + \frac{1}{\lambda} \left[\frac{f_2}{2g_2} - \frac{f_0 g_4}{8g_2^2} - \frac{f_1 g_3}{2g_2^2} + \frac{5f_0 g_3^2}{24g_2^3} \right] \right). \quad (15.5.10)$$

Most readers will rationally conclude that this is a lot of algebra and be deterred. It is not surprising that these formulas, particularly the higher-order terms, are seldom used in economics. No one wants to spend time keeping track of the Taylor expansions and changes of variables involved. However, all of this can be automated with symbolic software. The Taylor expansions, the change of variables, the Gaussian integration, and the extraction of coefficients are all examples of what symbolic software is built for. The wide availability of symbolic languages now makes these techniques much more accessible.

Stirling's Formula

One important and nonobvious application of Laplace's method is Stirling's formula for $n!$. The factorial function arises in many contexts in economics. Being a function of integers, we generally do not use continuous function tools in such analyses. For example, comparative statics would be difficult, since the derivative of $n!$ is a problematic idea.

Fortunately there is a smooth, monotonic approximation to $n!$, called *Stirling's formula*. It is well known that $n! = \Gamma(n+1)$ where $\Gamma(\lambda) \equiv \int_0^\infty e^{-x} x^{\lambda-1} dx$. If we were to directly apply Laplace's method to $\Gamma(n)$, we would take $g(x) = -\ln x$ and $f(x) = e^{-x} x^{-1}$ in (15.5.1). However, since $\ln x$ has no minimum on $[0, \infty)$, we cannot do this. Fortunately a change of variables will help. If we let $x = y\lambda$, then

$$\Gamma(\lambda) = \lambda^\lambda \int_0^\infty e^{-\lambda(y-\ln y)} y^{-1} dy.$$

Now $g(y) = y - \ln y$, and $f(y) = y^{-1}$. The minimum of $y - \ln y$ occurs at $y = 1$. At $y = 1$,

$$g(y) = y - \ln y \sim 1 + \frac{1}{2}(y-1)^2 - \frac{1}{3}(y-1)^3 + \frac{1}{4}(y-1)^4 + \dots$$

Application of (15.5.10) implies the two-term approximation

$$\Gamma(\lambda) \sim \sqrt{2\pi}\lambda^{\lambda-(1/2)}e^{-\lambda} \left(1 + \frac{1}{12\lambda}\right). \quad (15.5.11)$$

Stirling's formula is just the one-term expansion,

$$n! = \Gamma(n+1) \sim \sqrt{2\pi}(n+1)^{n+(1/2)}e^{-(n+1)}. \quad (15.5.12)$$

The two-term extension is

$$n! \sim \sqrt{2\pi}(n+1)^{n+(1/2)}e^{-(n+1)} \left(1 + \frac{1}{12(n+1)}\right), \quad (15.5.13)$$

Table 15.2Relative error of Laplace approximations of $n!$

n	1	5	10	20
Stirling's formula	0.04	0.01	0.008	0.004
Two-term Laplace	5(-4)	8(-5)	3(-5)	8(-6)
Three-term Laplace	3(-4)	1(-5)	2(-6)	3(-7)

and the three-term extension is

$$n! \sim \sqrt{2\pi}(n+1)^{n+(1/2)} e^{-(n+1)} \left(1 + \frac{1}{12(n+1)} + \frac{1}{288(n+1)^2} \right). \quad (15.5.14)$$

Table 15.2 displays the quality of Stirling's formula and higher-order Laplace expansions.

Note how well Laplace's method does. The theory says only that Stirling's formula does well when n is large. Table 15.2 shows that Stirling's formula does surprisingly well when n is actually small, and that the second and third terms of the Laplace expansion both improve the approximation substantially.

15.6 Hybrid Perturbation-Projection Methods

We saw in chapter 11 that a key step in projection methods of solving operator equations was the choice of a basis. Our earlier discussion focused on general purpose bases such as Chebyshev polynomials. While these bases may be adequate, we use very little information about the solution, primarily differentiability properties, in selecting among them. In this section we use perturbation methods to generate special purpose bases tailored to the problem at hand, adapting the papers of Geer and Andersen (1989, 1990) to a simple economics problem.

The general idea springs from the following observation. Suppose that we want to solve $\mathcal{N}f = 0$ for some operator \mathcal{N} , a type of problem we discussed in chapter 12. The key insight is that both projection and perturbation methods create approximations of the form

$$\hat{f} = \sum a_i \varphi_i(x) \quad (15.6.1)$$

to solve such problems. First, consider perturbation methods. In regular perturbation methods the coefficients are fixed to be $a_i = \varepsilon^i$, and the functions $\varphi_i(x)$ are computed via perturbation methods. The $\varphi_i(x)$ functions produced by perturbation methods are

very good functions for the problem at hand and are the best possible asymptotically. The problem is that fixing $a_i = \varepsilon^i$ may not be optimal when ε is not small.

Second, consider projection methods. In projection methods the basis functions $\varphi_i(x)$ are fixed a priori and the coefficients a_i are computed via projection methods. The coefficients are chosen to be the best possible choices relative to the $\varphi_i(x)$ and relative to some criterion. However, the bases are seldom the best possible for the problem. We usually choose general bases that satisfy some general criteria, such as smoothness and orthogonality conditions. Such bases may be adequate, but not the best. In particular, when we examine multidimensional problems, we often use tensor or complete bases that are quite large. It would be better if we had bases that were tailored for the problem.

The critical difference between perturbation and projection methods lies in what is fixed a priori and what is computed. Furthermore the strengths and weaknesses are complementary. The perturbation method does a good job at producing $\varphi_i(x)$ functions but is limited in its choices of the a_i coefficients, whereas the projection method chooses the a_i coefficients optimally but is limited in its $\varphi_i(x)$ choices. The *hybrid perturbation–projection* method combines the two methods in a way that takes advantage of their complementary strengths by choosing well both the basis functions and their coefficients. Algorithm 15.1 summarizes the procedure. First one formulates a continuum of problems which includes the specific problem we are trying to solve. This continuum should be parameterized by some ε which can be used as a perturbation parameter. Second, we use perturbation methods to construct a finite set of basis functions. Third, one uses projection methods to find a linear combination of the basis functions which approximately solves the problem of interest.

Algorithm 15.1 Hybrid Projection-Perturbation Method

Objective: Solve an equation $\mathcal{N}f = 0$.

Initialization. Construct a continuum of problems $\mathcal{H}(\varepsilon)f = 0$, where $\mathcal{H}(0)f = 0$ has the known solution f_0 , and $\mathcal{H}(1) = \mathcal{N}$.

Step 1. Solve a perturbation problem, arriving at an approximation

$$f(x) \sim \sum_{i=1}^n \delta_i(\varepsilon) \varphi_i(x)$$

for some gauge sequence δ_i .

Step 2. Apply a projection method, using the basis $\varphi_i(x)$, to choose the a_i coefficients that produce a good solution of the form

$$\hat{f}(x) = \sum_{i=1}^n a_i \varphi_i(x).$$

To illustrate the hybrid perturbation–projection method, consider the continuous-time one-sector deterministic growth problem and the perturbations we examined in section 13.5; specifically, we consider the parameterized family

$$C_k(k, \varepsilon)(f(k) - C(k, \varepsilon)) - \varepsilon C(k, \varepsilon)(f'(k) - \rho) = 0, \quad (15.6.2)$$

where $\varepsilon \equiv -\gamma^{-1}$ is the perturbation parameter. When $\varepsilon = 0$, the utility function displays “infinite” curvature and the solution is $C(k, 0) = f(k)$. We found that the first perturbation showed that

$$C_\varepsilon(k, 0) = f\left(\frac{\rho}{f'} - 1\right). \quad (15.6.3)$$

Further differentiation of (15.6.2) yields

$$C_{\varepsilon\varepsilon}(k, 0) = \frac{2\rho(f' - \rho)f''}{(f')^4}. \quad (15.6.4)$$

This is a process that can continue indefinitely. Note the differences between the collection $\{C, C_\varepsilon, C_{\varepsilon\varepsilon}\}$ and the polynomial bases we used in chapter 11. The functions used in C_ε and $C_{\varepsilon\varepsilon}$ involve k^α and other fractional powers of k , functions that are not spanned by finite sets of ordinary polynomials.

We next compare the quality of the perturbation approximation and the hybrid result using the example $\gamma = -2$, $\alpha = 0.25$, $\rho = 0.05$, and $f(k) = 0.2k^{0.25}$. In this case the collection $\{C, C_\varepsilon, C_{\varepsilon\varepsilon}\}$ spans the same functions as does $\{k^{0.25}, k, k^{1.75}\}$. The regular perturbation approach forms the approximation

$$C(k, \varepsilon) \doteq \sum_{i=0}^n \frac{\varepsilon^i}{i!} \frac{d^i C}{d\varepsilon^i}(k, 0).$$

The choice $\gamma = -2$ corresponds to $\varepsilon = 0.5$. In this case the three-term perturbation approximation is

$$\hat{C}_p(k) = 0.1k^{0.25} + 0.25k - 0.15k^{1.75}.$$

\hat{C}_p does a very poor job because it has a maximum error of 30 percent on the interval $k \in [0.5, 1.5]$. In fact it is foolish to expect this perturbation will do well, since we are expanding around $\varepsilon = 0$ and negative values of ε are economically nonsensical.

Regular perturbation imposes $a_i = \varepsilon^i/i!$, which are the correct choices for ε very close to zero. However, as ε moves away from zero, alternative choices may be better. The hybrid perturbation–projection method is to use the functions $C(k, 0)$, $C_\varepsilon(k, 0)$,

$C_{\varepsilon\varepsilon}(k, 0)$, $C_{\varepsilon\varepsilon\varepsilon}(k, 0)$, etc., as basis functions, and let projection conditions determine the $a_i(\varepsilon)$ coefficients in the series

$$\sum_{i=0}^n a_i(\varepsilon) \frac{d^i C}{d\varepsilon^i} (k, 0).$$

If we use the C , C_ε , and $C_{\varepsilon\varepsilon}$ functions as basis elements, and collocate at the three zeros of the cubic Chebyshev polynomial adapted to $[0.5, 1.5]$, we arrive at the approximation

$$\hat{C}_{h,1}(k) = 0.1341 k^{0.25} + 0.0761 k^{1.0} - 0.0101 k^{1.75}.$$

$\hat{C}_{h,1}$ has a maximum relative error of 0.1 percent, much better than that for \hat{C}_p . This should not come as too much of a surprise, since choosing the optimal coefficients should do better than just taking the integral powers of ε . It is somewhat surprising that it does two orders of magnitude better based on so few points. This example also shows that even if the perturbation result does poorly, we can still use the functions to construct a good approximation.

While $\hat{C}_{h,1}$ does well, there may be a natural element to add to the basis. In particular, $\hat{C}_{h,1}$ does not include a constant term. We next add 1 to our hybrid basis, collocate at the degree four Chebyshev zeros, and arrive at the approximation

$$\hat{C}_{h,2}(k) = -0.0266 + 0.1772 k^{0.25} + 0.0527 k^{1.0} - 0.0033 k^{1.75}.$$

$\hat{C}_{h,2}$ has a maximum relative error of 0.03 percent, even better than \hat{C}_p . Figure 15.2 displays the graphs of the Euler equation errors, (15.6.2), normalized by ρC , for $\hat{C}_p(k)$, $\hat{C}_{h,1}(k)$, and $\hat{C}_{h,2}(k)$. We see that the Euler equation errors are rather uniform over the range $k \in [0.5, 1.5]$ and that the Euler equation errors are good indicators for the error of each approximation.

This example illustrates the potential of the hybrid perturbation–projection method. The perturbation phase constructs a basis that is tailored for the problem at hand. With such a tailored basis the projection step should do well with a small number of basis elements. General bases will likely incorporate basis elements that are of little value and unlikely to incorporate individual elements that efficiently capture the true shape of the solution. Furthermore all the advantages which arise with general bases can be achieved with the bases constructed by perturbation methods. For example, many general bases are orthogonal, a fact that assists the projection method in computing the coefficients. The bases constructed by perturbation methods are unlikely to be orthogonal, but one can construct an equivalent orthogonal basis using the Gram-Schmidt process. The resulting orthogonal

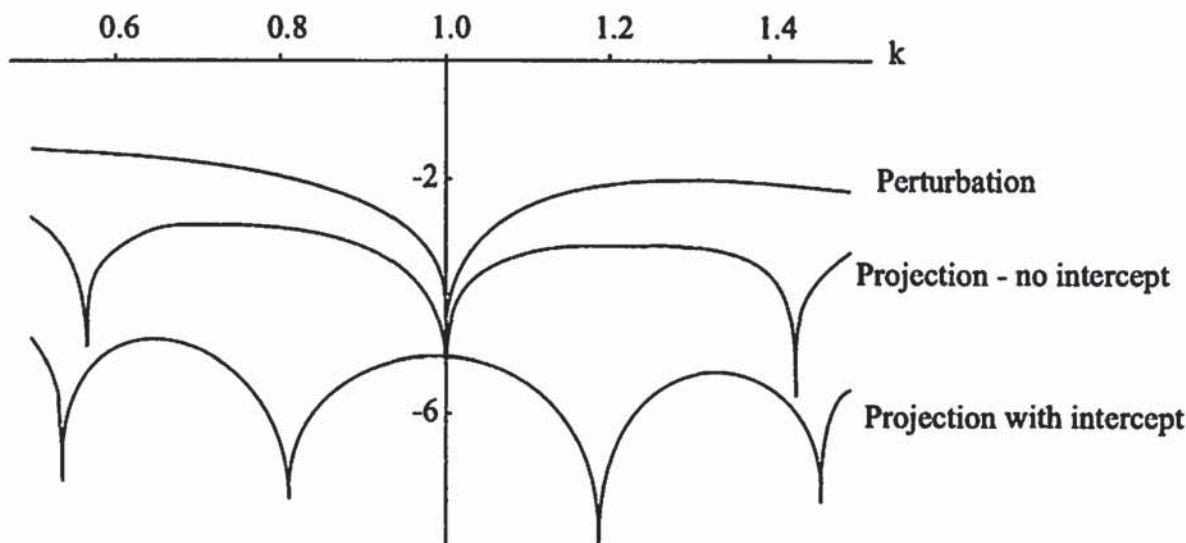


Figure 15.2
Euler equation errors for hybrid methods

basis retains the information from the perturbation process but will help to avoid ill-conditioning problems.

The hybrid method is potentially very useful in problems with several dimensions where general bases, even complete polynomial bases, are large. All that is necessary is a single parameter that can smoothly move the problem from a case with a known solution to the case of interest. The number of perturbation functions needed for the projection step depends on how much the problem changes as the parameter changes, not on the number of dimensions of the problem. Better bases may be possible by using more than one perturbation parameter and perturbing around more than one degenerate case. The key point is that the size of the resulting basis is under the control of the analyst, not dictated by the dimensionality of the problem.

15.7 Further Reading and Summary

The asymptotic methods discussed in this chapter have not been used much in economics. Judd and Guu (1996) follow up on Samuelson (1970) and compute asymptotic expansions for equilibrium asset prices. Dixit (1991) derives asymptotic rules for stochastic control problems with adjustment costs. Laplace's method is used in the statistical literature; for example, see Tierney and Kadane (1986) and Tierney et al. (1986). Similar asymptotic methods were used in Holly and Phillips (1979) and Ghysels and Lieberman (1983).

The asymptotic methods presented in this chapter go beyond simple applications of the Taylor series and implicit function theorems. The basic bifurcation ideas presented here are based on the Lyapounov-Schmidt procedure; see Zeidler (1986) and Aubin and Ekeland (1984) for extended presentations. The unknown gauge procedures are based on the Newton polygon method; see Zeidler (1986) for a discussion. Bleistein and Handelsman (1976) and Estrada and Kanwal (1994) present the foundations behind Laplace's method. Economists are somewhat more familiar with the Hopf bifurcation and other methods from dynamical systems theory; see Chow and Hale (1982), Benhabib and Nishimura (1979), and Zhang (1988). The bifurcation methods from dynamical theory are related to the bifurcation methods we presented and can also be used for computational purposes. Bender and Orszag (1978) give a good general introduction to many applied mathematics techniques.

Symbolic software, such as Macsyma, Maple, and Mathematica, is useful in deriving all the critical formulas in this chapter and possible extensions. For example, it would be very difficult to derive high-order terms in a multidimensional application of Laplace's method, but symbolic software following Laplace's approach could sort out the critical elements.

This and the previous chapters have just touched on asymptotic methods. Bensoussan (1986) presents a much greater variety of procedures, including much more difficult singular perturbation problems. Budd et al. (1993) applies singular perturbation methods to a dynamic game. Judd (1996) reviews perturbation methods.

Exercises

- Suppose that there are three risk types in the RSW model, an agent has an endowment of w_1 or w_2 with $w_1 > w_2$, and with the probability of a type i agent having the higher-value endowment being p_i , $i = 1, 2, 3$, $p_1 \geq p_2 \geq p_3$. Express the Rothschild-Stiglitz Nash equilibrium allocations. At $p_1 = p_2 = p_3 = p$ the equilibrium degenerates to perfect insurance. Consider the perturbation $p_1 = p$, $p_2 = p - \alpha\epsilon$, $p_3 = p - \epsilon$, where $0 \leq \alpha \leq 1$. Compute first- and second-order asymptotic expansions for equilibrium allocations in terms of ϵ . If $\alpha = 0$ or $\alpha = 1$, this problem reduces to a two-type model. For small ϵ , what is the worst α from the type 1 agents' perspective? Compare the answers from perturbation methods to those obtained using nonlinear equation methods.
- Assume that there are two investors of different tastes over final wealth and different initial wealth investing in two risky assets. Compute the equilibrium asset prices for risky assets using theorem 15.1.2, assuming exponential utility and log normal returns. Compare results to nonlinear equation methods of solution.
- Assume that $u(c) = c^{1+\gamma}/(1+\gamma)$. Write programs to compute

$$\frac{1}{\sqrt{2\pi}\sigma} \int_{-\infty}^{\infty} u(R + e^{z+\mu}) e^{-z^2/2\sigma^2} dz$$

in two ways. First, use Gauss-Hermite quadrature rules of varying precision. Second, replace $u(c)$ with an n th-order Taylor expansion based at $c = c_0$. Compute and compare the two ways for $R = 1.01$, $\mu = 0.06, 0.12$, $\sigma = 0.2, 0.4, 1.0$, $c_0 = R + e^\mu, R + e^{\mu-\sigma}, R + e^{\mu+\sigma}$, and $n = 1, 2, 3, 4, 5$. Report on the time versus accuracy trade-offs across the methods. Which are efficient?

4. Repeat exercise 3 for the multidimensional case. That is, use the two methods to compute approximations to

$$\int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} u(c) e^{-z_1^2/2\sigma_1^2} \cdots e^{-z_n^2/2\sigma_n^2} dz_1 \cdots dz_n,$$

where

$$c = R + e^{z_1+\mu_1} + \cdots + e^{z_n+\mu_n}.$$

Use the range of μ 's and σ 's listed in exercise 3. How do the methods compare as n goes from 1 up to 6?

5. Consider the first-order condition for portfolio allocation

$$0 = \int_{-\infty}^{\infty} u'(\omega R + (1 - \omega)e^{z+\mu})(R - e^{z+\mu})e^{-z^2/2\sigma^2} dz$$

for various CRRA functions u . The optimal ω can be computed by solving this first-order condition. How are the solutions for ω affected by the choice of technique used to compute the integral? (Use the techniques described in exercise 3 and the parameter values specified there.)

6. Solve (15.6.2) using the hybrid perturbation–projection method for various CRRA utility functions and various CES production functions. Compare to other alternative bases such as bases in $\log k$.

V APPLICATIONS TO DYNAMIC EQUILIBRIUM ANALYSIS

16 Solution Methods for Perfect Foresight Models

Dynamic models of equilibrium are increasingly used in economic analysis, but they present computational challenges. Sometimes dynamic models have equilibria that are solutions to some social planner's problem. In those cases we can apply the numerical dynamic programming and optimal control methods presented in chapters 10–12. However, many interesting problems have neither a closed-form solution nor any optimality characterization; this is particularly the case for problems in public finance, macroeconomics, and industrial organization where distortions of some sort preclude a social planner characterization. In those cases we must develop more general methods.

In this and the next chapter, we focus on methods for solving dynamic equilibrium models. In this chapter we discuss methods for solving nonlinear perfect foresight competitive models. There are two approaches to computing equilibrium in such models. First, we can focus on computing the equilibrium path of endogenous variables, such as prices and quantities. This is essentially the same approach as that of computable general equilibrium: Treat goods consumed at different times as different commodities, and compute the price of each time-differentiated good; we call this the *time domain* approach. Time domain methods produce large systems of nonlinear equations. Some methods exploit the dynamic structure to reduce equilibrium to the solution of a system of difference or differential equations. In general, the challenge is to find some structure that replaces large systems of nonlinear equations with more manageable problems.

The second approach is a recursive one, similar to dynamic programming methods of chapter 12. This approach first identifies state variables and expresses the equilibrium decision and pricing feedback rules as functions of the state variables. Recursive approaches are particularly useful in autonomous problems. Even though the recursive approach is not necessary for most of the deterministic models of this chapter, we introduce recursive equilibrium methods here because it is easy to highlight their key features. In the next chapter we take up stochastic rational expectations models where the recursive approach is the only practical approach. In this chapter we also apply the recursive approach to a simple time consistency problem, indicating the usefulness of the recursive approach in dynamic strategic contexts.

In all cases methods for solving perfect foresight models combine many tools of numerical analysis. Computing demand and supply in dynamic models require us to combine optimization and optimal control methods. Computing equilibrium requires us to apply nonlinear equation methods. We will often use approximation and projection methods to construct approximate solutions. We can create efficient algorithms by appropriately combining these various techniques.

16.1 A Simple Autonomous Overlapping Generations Model

We first examine computational methods for a simple, real, two-period overlapping generations (OLG) model. Assume that each individual lives for two periods, working in the first, and consuming in both the first and second periods. We assume that the common concave utility function is $u(c^y) - v(l) + \beta u(c^o)$, where c^y (c^o) is consumption in youth (old age) and l is labor supply in youth. We assume that a single good is used for both consumption and investment, and produced according to a concave gross production function $F(k, l)$, where k is the beginning-of-period capital stock; hence $k_{t+1} = F(k_t, l_t) - c_t$. Capital is owned solely by the old.

Let c_t^a be consumption in period t by age group $a \in \{y, o\}$ agents. The wage in period t is $w_t = F_l(k_t, l_t)$, and the gross return on capital is $R_t \equiv F_k(k_t, l_t)$. The old consume their wealth implying $c_t^o = R_t k_t$. The first-order condition for the young leisure-consumption decision is $w_t u'(c_t^y) = v'(l_t)$, and the intertemporal savings-consumption choice implies that $u'(c_t^y) = R_{t+1} u'(c_{t+1}^o)$. The budget constraint for the young born in period t is $(w_t l_t - c_t^y) R_{t+1} - c_{t+1}^o = 0$. The capital stock obeys the law of motion $k_{t+1} = w_t l_t - c_t^y$.

Dynamic Representation

In some cases we may be able to form a simple dynamic representation of equilibrium; this example is one such case. We attempt to represent equilibrium in the form

$$x_{t+1} = G(x_t) \quad (16.1.1)$$

for some list of variables x . In forming this system, it generally takes some introspection to choose x . In this model the beginning-of-period capital stock, k_t , is a predetermined state variable. We could include many other variables (and their lags) in the predetermined state vector, but we prefer to use a state of minimal size. We need to include endogenous variables in the state variable, enough so as to fix all other current endogenous variables. In our problem the current labor supply, l_t , is such a variable. The predetermined plus endogenous variables should be chosen so that they fix all future variables. We choose $x_t \equiv (k_t, l_t)^\top$.

Second, we express G efficiently. In this case, given $x_t = (k_t, l_t)^\top$, we compute

$$\begin{pmatrix} k_{t+1} \\ l_{t+1} \end{pmatrix} = x_{t+1} = G(x_t) \equiv \begin{pmatrix} G^1(k_t, l_t) \\ G^2(k_t, l_t) \end{pmatrix} \quad (16.1.2)$$

as follows:

1. Compute $R_t = F_k(k_t, l_t)$, $w_t = F_l(k_t, l_t)$, and $c_t^o = R_t k_t$.
2. Solve for c_t^y in $w_t u'(c_t^y) = v'(l_t)$.

3. Compute $k_{t+1} = w_t l_t - c_t^y$.
4. Solve for R_{t+1} in $u'(c_t^y) = R_{t+1} u'(R_{t+1} k_{t+1})$.
5. Solve for l_{t+1} in $R_{t+1} = F_k(k_{t+1}, l_{t+1})$.

There may be multiple solutions in some steps. In particular, step 4 may have multiple solutions if $u(c) = \log c$. In our examples we presume step 4 has a unique solution.

More generally, $x_t \in R^n$ is a list of variables, the equilibrium path is expressed implicitly as $H(x_t, x_{t+1}) = 0$ for some system of equations $H: R^{2n} \rightarrow R^n$, and the corresponding dynamic evolution law $x_{t+1} = G(x_t)$ is implicitly defined by $H(x, G(x)) = 0$. In such cases one must use nonlinear equation methods to compute x_{t+1} from x_t . Now that we have formulated our model as a dynamic system, we can begin computing the solution.

Steady States of Overlapping Generations Models

The next step is computing the steady-state. A steady state is characterized in our example by the equations

$$\begin{aligned} wu'(c^y) &= v'(l), \quad u'(c^y) = Ru'(c^o), \quad R = F_k(k, l), \\ c^o &= (wl - c^y)R, \quad k = wl - c^y, \quad w = F_l(k, l). \end{aligned} \tag{16.1.3}$$

Let R^* , w^* , k^* , and l^* denote the steady state values for R , w , k , and l .

Local Analysis: Determinacy of the Steady State and Perturbations

Once we have computed a steady state, we must check that it is locally determinate. That is, we must show that for capital stocks near the steady state capital stock, k^* , there is a unique path consistent with the dynamical system $x_{t+1} = G(x_t)$ which converges to (k^*, l^*) . If the system is not locally determinate, then either there is no such path or there is a continuum of such equilibrium paths. It is reasonable to assume that there is a steady state, but there is no presumption that it is locally determinate.

We check determinacy by applying the linearization methods discussed in chapters 13 and 14 to the dynamical system (16.1.2). In this case we need to compute the matrix $\begin{pmatrix} G_k^1 & G_l^1 \\ G_k^2 & G_l^2 \end{pmatrix}$ at the steady state values for k and l , and to compute its eigenvalues. Since we have one predetermined variable and one endogenous variable, determinacy requires one eigenvalue with modulus less than one and one eigenvalue with modulus greater than one. If these eigenvalue conditions do not hold, then the

system is not locally determinate, and we must stop. We proceed under the assumption that the system is locally determinate.

Global Analysis of Autonomous Systems

Once we have computed a steady state and checked that it is locally determinate, we can compute a convergent path. We first compute convergent paths for k_0 close to k^* . Let $L(k)$ be the unique choice of l for each k that guarantees convergence. We want to know its slope near the steady state. The identity $l_{t+1} = L(k_{t+1})$, implies the functional relation

$$L(G^1(k, L(k))) = G^2(k, L(k)) \quad (16.1.4)$$

for all k . Differentiation of (16.1.4) at $k = k^*$ implies that

$$L'(k^*)(G_k^1(k^*, l^*) + G_l^1(k^*, l^*)L'(k^*)) = G_k^2(k^*, l^*) + G_l^2(k^*, l^*)L'(k^*),$$

which is a quadratic equation in $L'(k^*)$. We pick that solution that implies convergence.

Local determinacy implies that we can use the shooting methods of chapter 10 to solve for convergent paths. Forward shooting is probably a poor choice, so we discuss reverse shooting for our example. Once we have the steady state values k^* and l^* , we pick an $\varepsilon > 0$, define $l^- \equiv l^* - \varepsilon L'(k^*)$, $k^- \equiv k^* - \varepsilon$, and execute the implicitly defined iteration

$$(k_s, l_s) = G(k_{s+1}, l_{s+1}) \quad (16.1.5)$$

with the initial condition $(k_0, l_0) = (k^-, l^-)$. Here s is the number of periods until (k, l) hits (k^-, l^-) , a point on the stable manifold near the steady state if we start at (k_s, l_s) . Symmetrically we also execute (16.1.5) with $(k_0, l_0) = (k^+, l^+)$ where $l^+ \equiv l^* + \varepsilon L'(k^*)$, $k^+ \equiv k^* + \varepsilon$. The two reverse shootings trace out two (k, l) paths, whose union approximates the stable manifold in (k, l) space.

Both of these methods can be applied to compute convergent paths for general OLG models. The key fact is that autonomous OLG models are dynamic systems, and their dynamics can be computed using perturbation and shooting methods if the equilibrium is locally determinate.

16.2 Equilibrium in OLG Models: Time Domain Methods

We next discuss methods to compute equilibria for a wider variety of OLG models, including ones with long finite lives and temporary time-varying factors. The logic

here differs from the previous section in that it relies on simple supply and demand aspects of the problem, and less on the formal dynamics. In this section we maintain the assumptions of a single good used for investment and consumption, and produced according to a CRS production function $F(k, l)$. We assume each agent lives for D periods and has a general utility function $U(c_1, l_1, \dots, c_D, l_D)$.

Computing Dynamic Supply and Demand

The first step in solving general equilibrium model in chapter 5 was to compute the demand of each agent. We can similarly compute supply and demand for each agent in OLG models in terms of the dynamic sequence of prices. The key organizing idea in solving the general OLG model is that each agent in an OLG model cares about only a few prices, and equilibrium is a path of prices along which supply and demand are in balance. We first choose some combination of variables whose paths characterize equilibrium. This could be a sequence of factor supply variables, a sequence of price variables, or some combination. We analyze the simple example of section 16.1 and focus on the gross interest rate, R_t , and wage, w_t , paths. The idea is to search for interest rate and wage paths that imply that supply equals demand in the factor markets at every time t . If we had several commodities or several factors, we would have to compute a price path for each such good; otherwise, all the details are unchanged.

The factor supply implications of a (R_t, w_t) sequence in the two-period OLG model of the previous section can also be easily computed. The (k_{t+1}^s, l_t^s) factor supply path is fixed by the life-cycle first-order conditions and budget constraint

$$\begin{aligned} u'(c_t^y) &= R_{t+1}\beta u'(R_{t+1}k_{t+1}^s), \\ w_t u'(c_t^y) &= v'(l_t^s), \\ k_{t+1}^s &= w_t l_t^s - c_t^y. \end{aligned} \tag{16.2.1}$$

Equation (16.2.1) has a unique solution for (k_{t+1}^s, l_t^s) and c_t^y , since utility is concave.

Since $F(k, l)$ is CRS, for each (R_t, w_t) pair there is a unique capital stock demand k_t^d determined by the marginal productivity condition $R_t = F_k(k_t^d, l_t^d)$ and labor demand l_t^d determined by $w_t = F_l(k_t^s, l_t^d)$. This (k_t^d, l_t^d) path describes the demand implications of the (R_t, w_t) path.

In more general OLG models, the factor supplies and demands of each generation depend on several realizations of R and w , and factor supply in each period depends on the decisions of agents from several cohorts. Suppose that agents live for D periods and that each agent in each cohort has the same tastes and endowment. The

factor supply problem for each agent is the solution to a life-cycle problem; let $K_t^{s,a}((R_t, w_t)_{t=1}^{\infty})$ be the supply of capital at time t by agents born in period $t - a + 1$. Then

$$K_t^s((R_t, w_t)_{t=1}^{\infty}) = \sum_{a=1}^D K_t^{s,a}((R_t, w_t)_{t=1}^{\infty})$$

expresses the total supply of capital at time t as a function of the factor price sequence $(R_t, w_t)_{t=1}^{\infty}$; similarly $L_t^s((R_t, w_t)_{t=1}^{\infty})$, $K_t^d((R_t, w_t)_{t=1}^{\infty})$, and $L_t^d((R_t, w_t)_{t=1}^{\infty})$ represent labor supply, and capital and labor demand at time t . Equilibrium is then expressed as a sequence $(R_t, w_t)_{t=1}^{\infty}$ such that

$$\begin{aligned} K_t^s((R_t, w_t)_{t=1}^{\infty}) &= K_t^d((R_t, w_t)_{t=0}^{\infty}), \quad t = 1, 2, \dots, \\ L_t^s((R_t, w_t)_{t=1}^{\infty}) &= L_t^d((R_t, w_t)_{t=1}^{\infty}), \quad t = 1, 2, \dots. \end{aligned} \tag{16.2.2}$$

Initially capital is owned by the agents alive at time $t = 1$, and it is an endowment that enters the budget constraint of the agents alive at $t = 1$. The aggregate capital stock specifies the capital supply at $t = 1$.

The simple OLG model we described above is just one such example of (16.1.2). All intertemporal perfect foresight models boil down to systems such as (16.2.2) for appropriate definitions of k and l . We are not assuming that the model is autonomous; policy variables can change with time, and we could have time-varying technology and tastes. Below we present methods for solving general perfect foresight models expressed in the form (16.2.2).

A Tatonnement Method

The system (16.2.2) states the problem in terms of basic supply and demand concepts, but since it is a problem with an infinite number of unknowns, it is not possible to solve numerically. We need to reduce this problem to a finite-dimensional problem, but we also want to exploit special features of OLG models that can help.

The first step we take is to assume that equilibrium converges to the steady state, that $(R_t, w_t) = (R^*, w^*)$, their steady state values, for $t > T$ for some large T , and find the subsequence $(R_t, w_t)_{t=1}^T$ at which factor markets clear in periods 1 through T . This truncates the system and generates little error if T is large enough. The desire to make T large means that we still have a large number of unknowns.

There are two ways commonly used to solve the resulting finite-dimensional system. The first approach uses a price-adjustment method motivated by the supply-demand intuition. Given a $(R_t, w_t)_{t=1}^T$ sequence, one calculates the resulting excess

demand sequence and adjusts the factor price sequence guess so as to reduce (hopefully) the excess demand. A Gauss-Jacobi tatonnement procedure is outlined in algorithm 16.1.

Algorithm 16.1 OLG Tatonnement Algorithm

Objective: Solve (16.2.2) given the initial capital stock k_0^s .

Initialization. Choose target horizon T , adjustment factor $\lambda > 0$, and convergence criterion $\varepsilon > 0$. Choose an initial guess $(R_t^0, w_t^0)_{t=0}^T$ for the initial factor price path; set R_t and w_t equal to their steady state values for $t > T$.

Step 1. Given the guess $(R_t^i, w_t^i)_{t=0}^T$, compute the demand and supply paths, $(k_t^d, l_t^d)_{t=1}^T$ and $(k_t^s, l_t^s)_{t=0}^T$, assuming that $(R_t, w_t) = (R^*, w^*)$ for $t > T$.

Step 2. If each component of $(k_t^d - k_t^s, l_t^d - l_t^s)_{t=0}^T$ has magnitude less than ε , STOP; else, go to step 3.

Step 3. Compute a new guess for $(R_t, w_t)_{t=0}^T$ from the excess demand vector. That is, for each $t = 0, 1, \dots, T$,

$$R_t^{i+1} = R_t^i + \lambda(k_t^d - k_t^s), \quad w_t^{i+1} = w_t^i + \lambda(l_t^d - l_t^s);$$

then go to step 1.

This problem uses basic supply-demand intuition, raising (lowering) the factor price in periods when the factor is in excess demand (supply). At best this method converges linearly, but it often has problems converging. It is a Gauss-Jacobi approach. A Gauss-Seidel approach would be impractical because it is too costly to recompute the excess demands $k_t^d - k_t^s$ and $l_t^d - l_t^s$ whenever some R_t or w_t is changed. A block Gauss-Seidel adaptation may be sensible if large blocks are used.

Algorithm 16.1 computes equilibrium for periods 1 through T . We would also like the factor supplies at $t > T$ to equal their steady state values. We cannot also impose that condition. Instead, we check the excess demand for labor and capital at some times $t > T$ implied by our computed prices for $t \leq T$ and steady state prices for $t > T$. If they are all “small,” we then accept the solution. If the excess demands are unacceptably large, we then try again using a larger T . Even if a choice of T is deemed to small, its “solution” could be used in constructing the initial guess for a larger T .

This procedure *assumes* convergence to a steady state. It also works best if the convergence is monotonic. If convergence involves cycles, a condition which is likely if there are any complex eigenvalues in the linearized system, then the results can be very sensitive to the choice of T .

It is natural to truncate the horizon when solving an infinite-horizon problem, and this is inherent in any time domain procedure where we compute the sequence of prices. However, these methods must be used carefully. In general, the solution of the time T truncated problem should be accepted only if the candidate solution is not sensitive to T and only if the deviations from equilibrium at times $t > T$ are not economically significant. These comments apply to any method that truncates the horizon, including the ones we explore next.

A Fixed-Point Iteration Method

A second procedure is a fixed-point iteration approach. Here we take a factor price sequence, use supply conditions to compute factor supplies, and then apply the factor demand equations to that factor supply sequence to arrive at the factor demand prices. This continues until the factor price sequence nearly repeats itself. Algorithm 16.2 makes this explicit.

Algorithm 16.2 OLG Fixed-Point Iteration Algorithm

Objective: Solve (16.2.2) given the initial capital stock k_0^s .

Initialization. Choose target horizon T , adjustment factor $\lambda > 0$, and convergence criterion $\varepsilon > 0$. Choose initial guess $(R_t^0, w_t^0)_{t=0}^T$ for the initial factor price path; set R_t and w_t equal to their steady state values for $t > T$.

Step 1. Given the guess $(R_t^i, w_t^i)_{t=0}^T$, compute the factor supply paths, $(k_t^i, l_t^i)_{t=0}^T$, assuming that $(R_t, w_t) = (R^*, w^*)$ for $t > T$.

Step 2. Given factor supply sequence $(k_t^i, l_t^i)_{t=0}^T$, compute the factor return sequence $(R_t^+ - R_t^i, w_t^+ - w_t^i)_{t=0}^T$ implied by the marginal product sequence $(F_k(k_t^i, l_t^i), F_l(k_t^i, l_t^i))_{t=0}^T$.

Step 3. If each component of $(R_t^+ - R_t^i, w_t^+ - w_t^i)_{t=0}^T$ has magnitude less than ε , STOP. Else, go to step 4.

Step 4. Compute a new guess $(R_t^{i+1}, w_t^{i+1})_{t=0}^T$: For each $t = 0, 1, \dots, T$,

$$R_t^{i+1} = R_t^i + \lambda(R_t^+ - R_t^i), \quad w_t^{i+1} = w_t^i + \lambda(w_t^+ - w_t^i),$$

and go to step 1.

This is a conventional fixed-point iteration, “hog-cycle” approach with the extrapolation parameter λ . If $\lambda = 1$ leads to unstable iterations, one should try $\lambda < 1$ to stabilize the sequence. If the iterates appear to converge but slowly, then one should try values for λ that exceed unity to accelerate convergence. As with the tatonnement algorithm, we check if T is large enough by evaluating excess demands at some times $t > T + 1$.

There is no reason to be confident that either the tatonnement or fixed-point iteration methods will work. If the system of equations were diagonally dominant, we would expect convergence. However, in many models the supply of capital at time t depends significantly on the several interest rates, not just on R_t , and diagonal dominance is unlikely to hold. Generally, we cannot expect good performance from either of these methods.

Newton's Method and Nonlinear Equation Methods

A third approach is to use Newton's method or any of the other standard nonlinear equation solution methods without taking any advantage of the dynamic structure of the problem. A potentially important advantage of Newton's method is its local quadratic convergence. The strategy is fairly simple. We use step 1 in the tatonnement algorithm to compute the excess demand vector,

$$E \equiv (k_t^d - k_t^s, l_t^d - l_t^s)_{t=1}^T$$

as a function of

$$P \equiv (R_1, w_1, R_2, w_2, \dots, R_T, w_T, R^*, w^*, \dots).$$

The variable P has $2T$ unknowns, and the E vector has $2T$ components. Once we have written a routine that computes E as a function of P , we then use a Newton routine to compute a zero of $E(P)$.

Parametric Path Method

We saw in section 11.3 how to solve dynamic control problems by parameterizing the solution and choosing coefficients for the parametrization which result in the best solution. We can again use that idea to solve discrete-time perfect foresight models. In the previous section we proposed solving the system

$$E(P) = 0,$$

where $E(P)$ is the excess demand vector for a price vector P . There we allowed P to vary freely for periods 1 to T but imposed steady state values for $t > T$.

Let $\Psi(t; a)$ be some parameterization of functions of time. The *parametric path* method parameterizes the R_t sequence with $\Psi(t; a^R)$, parameterizes w_t with $\Psi(t; a^w)$ where $a^R, a^w \in \mathbb{R}^m$, $m < T$, where hopefully m is much less than T . Let $P(a^R, a^w)$ represent the price vector P , which is implied by the parameter choices a^R and a^w . The task is to find coefficients a^R and a^w such that $E(P(a^R, a^w)) = 0$. This is generally impossible, since a^R and a^w have dimension less than T . Instead, we use some

projection method to find a^R and a^w coefficients such that $E(P(a^R, a^w))$ is close to being the zero vector. The least squares approach solves

$$\min_{a^R, a^w} \sum_{i=1}^{2T} E_i(P(a^R, a^w))^2,$$

and alternative methods would fix some system of projection functions $\Pi: R^{2T} \rightarrow R^{2m}$ and solve the system

$$\Pi(E(P(a^R, a^w))) = 0.$$

Price path parameterizations should utilize basic features of factor price paths. The basic assumption in our methods is that factor prices converge to a steady state after some time T . If this is true, then it is also likely that such convergence takes place smoothly. One parameterization that models such paths is $a + (\sum_i b_i t^i) e^{-\lambda t}$ where a, b , and $\lambda > 0$ are free parameters. The polynomial piece allows for substantial flexibility at early times t , but this parameterization forces the path to converge to a through the dampening powers of the negative exponential.

In many applications the solution will not be smooth in the first few periods. This can arise, for example, if one is modeling anticipated changes in taxation or monetary policy. The parametric path can still be used in many cases. As long as these discrete changes are confined to early times $t < T_1$, then the solution for $t > T_1$ will likely be smooth. One could compute each factor price for each $t < T_1$ but approximate prices for $t > T_1$ parametrically.

An advantage of the parametric path method is that one can choose T to be large without having to solve a system of $2T$ equations in $2T$ unknowns. In any case the parametric path method can exploit smoothness properties of the price path and substantially reduce the dimensionality of the problem.

Application: Evaluation of Empirical Methods

Solutions to dynamic models can be used to address many questions. One important application is using them to evaluate empirical analyses. In this section we discuss one such exercise.

One of the great debates in empirical public finance is the effect of Social Security on savings. Feldstein (1974) argued that Social Security has reduced national saving, whereas Barro (1974) argued that Social Security has had no effect on savings. The empirical approaches to this question typically regress consumption on a variety of variables, but focus on a Social Security wealth (SSW) variable, that is, some vari-

able that measures the privately perceived value of Social Security promises. Under Feldstein's arguments, the coefficient on SSW should be positive due to a wealth effect, but Barro invokes Ricardian neutrality to argue for a zero coefficient.

Since these empirical studies were not based on a structural model, the implications of the regression coefficients are unclear. Auerbach and Kotlikoff (1983) uses an OLG model to address the reliability of the methods used in this literature. We know the true answer in the Auerbach-Kotlikoff model: People live finite lives and regard SSW as wealth because it will be financed by taxes on people not yet born. Using their model, Auerbach and Kotlikoff (1983) simulate the response to the introduction of Social Security, assuming that the economy is initially in the no-Social Security steady state. This generates consumption, output, wage, interest rate, and SSW data. They then use this data in regression methods to "estimate" the response of consumption to SSW. They find that the regression results have little informative value in predicting the actual long-run response of consumption and savings to Social Security in their model. This is not surprising, for the regressions are not structural in any way nor based on even a reduced form for the critical behavioral functions.

While the Auerbach-Kotlikoff (1983) exercise is limited to a particular issue, the strategy is clear. The idea is to numerically solve a structural model, use it to generate data, and then apply an empirical method to the data. Since we know the true structure of the model generating the data, we can determine whether the empirical procedure can be used to infer properties of the structural model. Full structural empirical analysis is preferable but often infeasible. This approach allows one to use structural ideas to evaluate reduced-form empirical methods.

16.3 Fair-Taylor Method

The methods discussed in the previous section focused on supply-demand ideas, often ignoring dynamic elements. The Fair-Taylor method uses dynamic structure to solve perfect foresight models that have a relatively simple dynamic structure. We illustrate it here by applying it to the linear perfect foresight problem

$$y_t = \alpha y_{t+1} + x_t \quad (16.3.1)$$

with the side condition

$$-\infty < \lim_{t \rightarrow \infty} y_t < \infty. \quad (16.3.2)$$

In (16.3.1), y is the endogenous variable and x is a bounded exogenous variable. We focus on the problem of finding y_0 such that (16.3.1) and (16.3.2) hold. The true

solution is

$$y_0 = \sum_{t=0}^{\infty} \alpha^t x_t.$$

The simple dynamic structure of (16.3.1) allows us to exploit dynamic ideas. The Fair-Taylor method first fixes a horizon T . Given a value for y_{T+1} (for the sake of simplicity we set $y_{T+1} = 0$), the collection

$$y_t = \alpha y_{t+1} + x_t, \quad t = 0, 1, \dots, T, \quad (16.3.3)$$

is a system of $T + 1$ equations in the $T + 1$ unknowns y_t . To solve (16.3.3), Fair-Taylor first makes a guess $Y_{T,t}^0$ for each y_t , $t = 1, \dots, T$; again, assume that each initial guess is 0. Next the Fair-Taylor method creates a sequence of approximations. Let $Y_{T,t}^j$ denote the approximation for y_t in iteration j when we fix the horizon at T . Given the guess $Y_{T,t}^j$, the next iterate $Y_{T,t}^{j+1}$ is defined by the type I iteration (over t)

$$Y_{T,t}^{j+1} = \alpha Y_{T,t+1}^j + x_t, \quad t = 0, 1, \dots, T. \quad (16.3.4)$$

The iteration in (16.3.4) is called a *Type I iteration*. In (16.3.4) the new guess for y_t , $Y_{T,t}^{j+1}$, is computed by applying the RHS of (16.3.1) to the old guess of y_{t+1} , $Y_{T,t+1}^j$. For each j , $j = 1, \dots, T$, we repeat (16.3.4); iterations over j are called *Type II iterations*. The combination of Type I and II iterations constitute a Gauss-Jacobi approach to solving (16.3.3), since the $j + 1$ guesses $Y_{T,t}^{j+1}$ are based only on the $Y_{T,t}^j$. $Y_{T,t}^{j+1}$ does not change if we set $j > T$, so we stop at $j = T$. More generally, the Fair-Taylor method stops at some earlier j if, for all t , $|Y_{T,t}^{j+1} - Y_{T,t}^j|$ is less than some convergence criterion.

Type I and II iterations solve the problem for a given terminal time T and terminal guess y_{T+1} . We next adjust the horizon. To do so, we now take the $Y_{T,t}^T$ solution and use it as the initial guess to compute the next problem where we let $T + 1$ be the terminal period; that is

$$Y_{T+1,t}^0 = Y_{T,t}^T, \quad t = 0, 1, \dots, T.$$

We now need a value for $Y_{T+1,T+1}^0$; again we choose zero. We continue to increase the terminal time until the successive paths change little; the iterations over successively larger T are called *Type III iterations*. As long as $\alpha < 1$, $Y_{T,0}^T$ converges to the correct value of y_0 as $T \rightarrow \infty$. In this example we set $Y_{T,T+1}^j = 0$ for each Type II iteration j and terminal date T ; this choice should not affect the solution and has no impact on the final solution for this problem since $\alpha < 1$.

More generally, we try to find some bounded sequence y_t which solves the equation

$$g(y_t, y_{t+1}, x_t) = 0. \quad (16.3.5)$$

The Type I and II iterations in Fair-Taylor algorithm are implicitly defined by

$$g(Y_{T,t}^{j+1}, Y_{T,t+1}^j, x_t) = 0, \quad t = 0, \dots, T, j = 0, 1, \dots \quad (16.3.6)$$

In (16.3.6) each $Y_{T,t}^{j+1}$ is solved by a nonlinear equation solver if a direct solution is not available. Type III iterations then increase T until such changes produce little effect.

We next compare the Fair-Taylor method to reverse shooting. The reverse shooting method also fixes a horizon T , makes a guess for y_{T+1} , and solves (16.3.1). However, it implements the Gauss-Seidel iteration

$$Y_{T,t} = \alpha Y_{T,t+1} + x_t, \quad t = T, T-1, \dots, 0, \quad (16.3.7)$$

in one pass and with operation cost proportional to T . By computing the $Y_{T,t}$ sequence beginning with $t = T$ instead of $t = 0$, we bring the information at T immediately down to all earlier values of $Y_{T,t}$; in this example reverse shooting exploits the same idea as that used in the upwind Gauss-Seidel method of dynamic programming. The reverse shooting idea here is also equivalent to back-substitution in linear systems, since the linear system in (16.3.3) is triangular. Since reverse shooting solves the problem in one pass for a given T , we might as well take a large T and solve (16.3.7) once as opposed to trying out several different smaller T . In the more general problem (16.3.5), reverse shooting is the iteration

$$g(Y_{T,t}, Y_{T,t+1}, x_t) = 0, \quad t = T, T-1, \dots, 0.$$

Here we eliminate the need for Type I and II iterations.

This discussion has examined only problems with purely expectational variables since none of the components of y were predetermined. The Fair-Taylor method can be adapted to include predetermined variables; see Fair and Taylor (1983) for details.

16.4 Recursive Models and Dynamic Iteration Methods

The previous sections examined dynamic models and computed the time path of equilibrium prices. When a model depends on calendar time, those methods are the most natural to use.

For many perfect foresight models, equilibrium prices depend on some time-independent “state” of the economy at the time, where by state we mean a vector of variables that describes the economic condition of the economy. Such state variables may include capital stock, consumption habit, lagged prices, stock of government bonds, and so forth. In the equilibria of these deterministic models, the equilibrium at time t is determined solely by the state at time t , and it is independent of calendar time conditional on knowing the state. The transition from time t to time $t+1$ is determined by the state at time t . In time-autonomous models we may compute the time-autonomous transition rules instead of paths.

To illustrate recursive methods, we return to the simple deterministic, representative agent growth problem

$$\begin{aligned} \max_{c_t} & \sum_{t=0}^{\infty} \beta^t u(c_t) \\ \text{s.t. } & k_{t+1} = F(k_t) - c_t. \end{aligned} \tag{16.4.1}$$

We focus not on the equilibrium time path for c_t but rather on the rule, $c_t = C(k_t)$, relating the current capital stock to the equilibrium consumption choice. The equilibrium rule contains all the information we need to compute the equilibrium price path for any initial condition; therefore, by solving for $C(k)$, we are solving *all* perfect foresight problems of this model. Also, since we are looking for a function that is used at all time t , we have no need to truncate the horizon; therefore the problems that arise from truncation in time domain approaches, such as the sensitivity to the choice of the truncation time T , will not arise here.

In this section we will focus on the abstract iteration procedures in function spaces as well as the ways in which we implement the ideal iterations on real computers. While there will be some differences between the ideal and numerical processes, we will prescribe methods that minimize the differences.

Operator Representation of Equilibrium

The equilibrium consumption rule $C(k)$ satisfies

$$u'(C(k)) = \beta u'(C(F(k) - C(k)))F'(F(k) - C(k)) \tag{16.4.2}$$

This Euler equation can be represented as a zero of an operator as in

$$\begin{aligned} 0 &= u'(C(k)) - \beta u'(C(F(k) - C(k)))F'(F(k) - C(k)) \\ &\equiv (\mathcal{N}(C))(k). \end{aligned} \tag{16.4.3}$$

\mathcal{N} is an operator from continuous functions to continuous functions. From this perspective we are looking for a function C that \mathcal{N} maps to the zero function.

Even though the Euler equation defines (16.4.3) in terms of one function $C(k)$, it is useful to think about the Euler equation more abstractly in ways not related to the economics. Note that the unknown function, C , occurs in four spots in (16.4.2). Let us consider the four occurrences of C independently and define the operator \mathcal{F} :

$$\begin{aligned} 0 &= u'(C_1) - \beta u'(C_2(F - C_3))F'(F - C_4) \\ &\equiv \mathcal{F}(C_1, C_2, C_3, C_4). \end{aligned} \quad (16.4.4)$$

When we rewrite the Euler equation in terms of the operator \mathcal{F} , we find that we are looking for a function C that solves the equation

$$0 = \mathcal{F}(C, C, C, C) \equiv \mathcal{N}(C). \quad (16.4.5)$$

The more general representation of (16.4.4) suggests a wide variety of iteration schemes. Before plunging ahead with a problem, it is best to examine various formulations, since different formulations will suggest different numerical approaches.

Sufficiency Problems and Boundary Conditions

Euler equations are necessary conditions for equilibrium and are the focus of our analysis. However, they are not sufficient conditions, for there often are solutions to the Euler conditions corresponding to excessive savings paths that violate the transversality condition at infinity. In this particular problem, all of those bad paths have the property that the capital stock does not converge to the steady state; therefore, if we find a solution with this property, we can reject it. In this model Amir et al. (1991) have shown that a sufficient condition for a policy function to be a local interior solution to the dynamic optimization problem is that $(C(x) - C(y))/(x - y) < 1$ for all x, y in the interval of permissible states. Therefore, if we have a solution C to the Euler equation, we can check this condition. In many optimal growth problems, we know that there is a bounded capture region (see Brock and Mirman 1972) under the optimal policy and, hence, in equilibrium; this is another condition that can be checked. The approach we use is to first find a $C(k)$ that solves (16.4.5) and check that it satisfies sufficient conditions.

Most solution methods for recursive models focus only on the Euler equations and ignore these global considerations. However, we must check that any candidate solution satisfies these critical global conditions. If a candidate does not, we then try again to find an acceptable solution. From a mathematical perspective this is bad

practice. It would be far better to incorporate these global considerations as boundary conditions imposed on (16.4.5) that eliminate the extraneous solutions. Fortunately the models that economists have studied so far have enough inherent stability that the Euler equation methods are generally successful, but possibly only because of luck. This issue is not unique to this particular model, but generic in rational expectations modeling. We will not deal formally with this problem, since *ex post* checking and good initial guesses appear to suffice for the problems we discuss in this book; also such a discussion would take us far too deeply into functional analysis and beyond the scope of this text. However, this does not diminish the fact that analysts must be aware of the dangers involved when one does not incorporate stability conditions from dynamic theory into their numerical algorithms.

Since all of the methods to compute policy functions are examples of the projection approach, we will use the general projection method outline of chapter 11 as an outline of our discussion of them. We will first review various ways in which the problem may be formulated, and then discuss ways to find the solution.

Policy Function Representation

The discussion after (16.4.1) focused on the mathematically pure versions of the problem. The function $C(k)$ which solves (16.4.2) is not likely to be a simple function. All methods that do not discretize the state space parameterize the policy function $C(k)$ in some way. That is, they choose some functional form $\hat{C}(k; \alpha)$ where each vector of coefficients α represents an approximation of $C(k)$ and then find α such that

$$0 \doteq \mathcal{N}(\hat{C}(\cdot; \alpha)). \quad (16.4.6)$$

This representation limits our search for an approximate solution to the collection of functions spanned by various choices of α .

The literature has used various approximation schemes discussed in chapter 5 as well as some others. Obvious choices are piecewise linear functions, ordinary polynomials, splines, and Chebyshev polynomials. As long as we expect $C(k)$ to be smooth and lacking regions of high curvature, polynomial forms for $\hat{C}(k; \alpha)$ are likely to work well. We know $C(k)$ is increasing; this indicates that we may want to use a shape-preserving method.

We also know that $C(k)$ is always positive; we can guarantee this by using a shape-preserving spline, or approximating $\log C(k)$. Positivity of $C(k)$ can be important since many common specifications of tastes imply that $u'(c)$ is not defined if $c < 0$. Approximation schemes that keep $\hat{C}(k; \alpha)$ positive will guarantee that $u'(\hat{C}(k; \alpha))$ is defined at all k for all choices of the α parameters.

Recursive methods do not truncate the horizon as did time domain methods. Since any numerical method must discretize the problem somewhere, we see here that we are replacing time truncation with discretization in the space of permissible policy functions, a discretization in the spectral domain. Which is better will depend on the problem. Time domain problems produce prices that satisfy equilibrium conditions for some initial period but may do poorly at later times. Recursive methods aim to produce approximations that have small errors at all times.

Time Iteration

We have seen that there are various ways to parameterize the policy function. The common factor is that they all reduce the problem to finding a finite number of parameters, a . We next discuss three iterative methods for computing the solution. This leads to choose the kind of iteration scheme used to determine a , the second key difference among methods. We first examine one problem-specific, iteration scheme that has been used in several economic contexts.

Time iteration is a natural iterative scheme for solving Euler equations. Time iteration implements the iterative scheme

$$0 = u'(C_{i+1}) - \beta u'(C_i(F - C_{i+1}))F'(F - C_{i+1}), \quad (16.4.7)$$

where i indexes the successive iterates. In terms of the notation defined in (16.4.4), time iteration implements the nonlinear functional iteration implicitly defined by

$$0 = \mathcal{F}(C_{i+1}, C_i, C_{i+1}, C_{i+1}). \quad (16.4.8)$$

The unique feature of time iteration is that it has some economic intuition. The basic idea is that if $C_i(k)$ is tomorrow's consumption policy function, then today's policy, denoted by $C_{i+1}(k)$, must satisfy

$$u'(C_{i+1}(k)) = \beta u'(C_i(F(k) - C_{i+1}(k)))F'(F(k) - C_{i+1}(k)). \quad (16.4.9)$$

As we let i increase, $C_i(k)$ hopefully converges to the infinite horizon solution. The critical fact is the monotonicity property of (16.4.9); that is, if $C'_i(k) > 0$ and $C_i(k) < C_{i-1}(k)$, then $C_{i+1}(k) < C_i(k)$ and C_{i+1} is an increasing function. Because of the monotonicity properties of the Euler equation, time iteration is reliable and convergent even though it uses only the Euler equation. Since we are solving the problem backward from a finite initial condition, time iteration will not converge to paths that violate global conditions such as the TVC_∞ . This is one case where we are imposing a boundary condition.

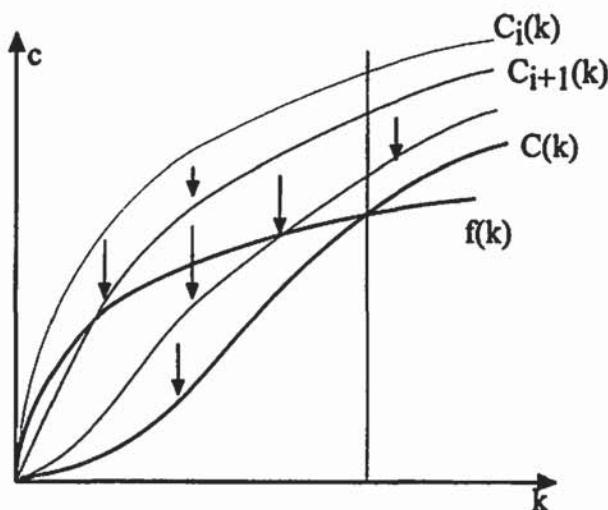


Figure 16.1
Time iteration method

The convergence result for the theoretical time iteration¹ scheme suggests that we use it for computational purposes. Algorithm 16.3 presents a computational implementation of time iteration.

Algorithm 16.3 Time Iteration for (16.4.2)

Initialization. Pick a set of points k_j , $j = 1, \dots, n$. Choose an approximation scheme $\hat{C}(k; a)$, $a \in R^m$, $m \leq n$, and choose a^0 so that $\hat{C}(k; a^0) \doteq F(k)$. Choose a stopping criterion $\varepsilon > 0$.

- Step 1.* Given $\hat{C}(\cdot; a^i)$, compute $C_{i+1}(k_j)$ from (16.4.9) for k_j , $j = 1, \dots, n$.
- Step 2.* Find an $a^{i+1} \in R^n$ so that the function $\hat{C}(\cdot; a^{i+1})$ approximates the data $(k_j, C_{i+1}(k_j))$.
- Step 3.* If $\|\hat{C}(\cdot; a^{i+1}) - \hat{C}(\cdot; a^i)\| < \varepsilon$, STOP; else go to step 1.

Figure 16.1 shows successive iterates of $C_i(k)$ where the initial guess has consumption exceeding output at all k . The iterates converge monotonically downward to the solution. We define $f(k) = F(k) - k$.

Unfortunately, time iteration is slow. Various acceleration schemes can speed up convergence, but it will remain slow. The reasons for its slow speed is that (16.4.9) involves a separate nonlinear equation for each k_j . Furthermore convergence for time iteration, as with value function iteration, is linear at best. Therefore we need many

1. Since $C(k)$ is related one to one to the marginal value of k , it is clear that this scheme for computing $C(k)$ is equivalent to using value iteration to compute the value function but is not the same as applying policy iteration to the dynamic programming version of this problem.

iterations to find a good approximation to the infinite-horizon problem, and each iteration is slow.

Unfortunately, it is the pure mathematical process described in (16.4.9) that is reliable and convergent, not necessarily the numerical procedure. In algorithm 16.3 we need to specify a particular approximation scheme $\hat{C}(k; a)$, and each iteration computes the solution to the Euler equation at only a finite number of k values. Convergence of the pure mathematical scheme in (16.4.8) does not imply convergence of algorithm 16.3 in general.

It is nontrivial to find a numerical process that inherits the reliability properties of the pure mathematical process, (16.4.8). One must choose the points and approximation scheme carefully. One reliable procedure is to choose a shape-preserving approximation scheme. It is obvious that the $\hat{C}(k; a)$ iterates in algorithm 16.3 will be monotone in k if the approximation procedure in step 2 is shape-preserving, but not otherwise in general. Many applications choose a piecewise linear approximation scheme, which is shape-preserving, when choosing the functional form for $\hat{C}(k; a)$. However, piecewise linear schemes will create kinks in the nonlinear equation problems in step 1 of algorithm 16.3. Better would be smoother shape-preserving schemes, such as the Schumaker shape-preserving quadratic spline scheme. Shape nonpreservation can generate convergence problems here as discussed in chapter 12. Other approximation schemes, such as polynomials and splines, may work, but they may also produce convergence problems.

Fixed-Point Iteration

We next develop a fixed-point iteration scheme for solving (16.4.3). Fixed-point iteration schemes were discussed in both chapters 3 and 5 to solve equations in R^n . The time iteration scheme discussed in the preceding section is an important special case of an iterative process since monotonicity properties of the iteration procedure lead to global convergence. In general, iteration schemes do not possess such properties. We now turn to a simple fixed-point iteration method.

Fixed-point iteration applied to (16.4.3) implements the iterative scheme defined implicitly by

$$0 = \mathcal{F}(C_{i+1}, C_i, C_i, C_i). \quad (16.4.10)$$

This has an advantage in that C_{i+1} is easy to compute since at any k ,

$$\begin{aligned} C_{i+1}(k) &= (u')^{-1}(\beta u'(C_i(F(k) - C_i(k))) F'(F(k) - C_i(k))) \\ &\equiv (T_{fp}(C_i))(k). \end{aligned} \quad (16.4.11)$$

Algorithm 16.4 implements fixed-point iteration.

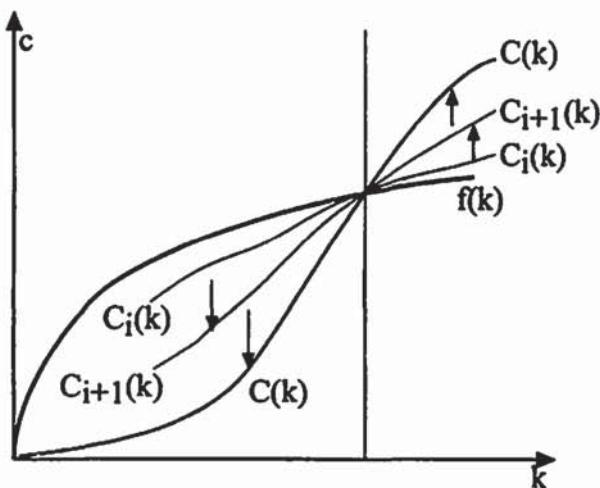


Figure 16.2
Fixed-point iteration method

Algorithm 16.4 Fixed-Point Iteration for (16.4.2)

Initialization. Pick a set of points $k_j, j = 1, \dots, n$. Choose an approximation scheme $\hat{C}(k; a)$, $a \in R^m$, $m \leq n$, and choose a^0 so that $\hat{C}(k; a^0) \doteq F(k) - k$. Choose a stopping criterion $\varepsilon > 0$.

Step 1. Compute $C_{i+1}(k_j)$ using (16.4.11) for $k_j, j = 1, \dots, n$.

Step 2. Find an $a^{i+1} \in R^n$ so that the function $\hat{C}(\cdot; a^{i+1})$ approximates the data $(k_j, C_{i+1}(k_j))$.

Step 3. If $\|\hat{C}(\cdot; a^{i+1}) - \hat{C}(\cdot; a^i)\| < \varepsilon$, STOP; else go to step 1.

Algorithm 16.4 is the same as algorithm 16.3 except that we replace a nonlinear equation problem in step 1 of algorithm 16.3 with simply computing $C_{i+1}(k_j)$ by just plugging k_j into the expression for $(T_{fp}(C_i))(k)$ in (16.4.11).

Figure 16.2 shows successive iterates of $C_i(k)$ where the initial guess has consumption equal to output. We define $f(k) = F(k) - k$. Since the initial guess goes through the steady state, the iterates rotate counterclockwise to the solution.

We have seen that both fixed-point iteration and time iteration are cases of different indexing schemes in (16.4.8) and (16.4.10). We could define many other schemes by using different combinations of i and $i + 1$ indexing. We could even define schemes which include $i - 1$ indexes. Once we have expressed the problem in the form (16.4.4) and (16.4.5), many schemes suggest themselves. We have just presented two common schemes. Since they both arise from economic intuition, it is not clear that they are the best from a numerical approach.

Fixed-Point Iteration and Stability

Unlike time iteration, fixed-point iteration may be unstable. It is not surprising that stability is a problem for fixed-point iteration, since it is essentially executing a hog-cycle iteration. We now consider the stability of fixed-point iteration. This exercise will introduce us to a way that will help us choose among iteration schemes and convert unstable ones into stable ones. Fixed-point iteration is defined by $C_{i+1} = \Psi$, where Ψ is implicitly defined by

$$u'(\Psi(k)) = \beta u'(C(F(k) - C(k)))F'(F(k) - C(k)). \quad (16.4.12)$$

Let k^* denote the steady state capital stock. Suppose that $C(k, \varepsilon) = \bar{C}(k) + \varepsilon\xi(k)$, for some fixed consumption function $\bar{C}(k)$ and some deviation function $\xi(k)$. We want to study how Ψ changes as C changes in the direction of $\xi(k)$. As we change ε away from $\varepsilon = 0$, Ψ will change, with the (Gateaux) derivative in the ξ direction at $\varepsilon = 0$ being defined by

$$\begin{aligned} & u''(\Psi(k))\Psi_\varepsilon(k) \\ &= \beta u''(\bar{C}(F(k) - \bar{C}(k)))\xi(F(k) - \bar{C}(k))F'(F(k) - \bar{C}(k)) \\ &\quad + \beta u''(\bar{C}(F(k) - \bar{C}(k)))\bar{C}'(F(k) - \bar{C}(k))[-\xi(k)]F'(F(k) - \bar{C}(k)) \\ &\quad + \beta u'(\bar{C}(F(k) - \bar{C}(k)))F''(F(k) - \bar{C}(k))[-\xi(k)]. \end{aligned} \quad (16.4.13)$$

At the fixed point C , $C = \Psi$, and at the steady state k^* , $\beta F'(k^*) = 1$. Therefore at k^* we conclude that

$$u''(C(k^*))\Psi_\varepsilon(k^*) = \xi(k^*)[u''(C(k^*))[1 - C'(k^*)] - \beta u'(C(k^*))F''(k^*)]. \quad (16.4.14)$$

If fixed-point iteration is to be stable, we must have $|\Psi_\varepsilon(k^*)| < |\xi(k^*)|$, which, since $C' > 0$ and $u'F''/u'' > 0$, is equivalent to

$$1 - \bar{C}'(k^*) - \beta \frac{u'}{u''} F''(k^*) > -1, \quad (16.4.15)$$

where the value of $C'(k^*)$ was computed in (13.7.17).

The LHS of (16.4.15) is always less than one, implying that any failure of convergence is due to exploding oscillations. Since C' is greater as u is more linear, the LHS will exceed 2 if absolute risk aversion, u'/u'' , is sufficiently small. Therefore instability is a possibility. In particular, it is straightforward to show that if $f(k) = 0.2k^{0.25}$, $\beta = 0.9$, and $u = c^{0.01}$, then (16.4.15) is not satisfied and fixed-point

iteration is unstable. For this example, fixed-point iteration is stable for empirically relevant choices of tastes and technology.

This stability analysis focused on the pure mathematical problem (16.4.12), not on any particular computational implementation. However, convergence problems in (16.4.12) will probably cause convergence problems in algorithm 16.4. A natural way to address convergence problems of the kind in (16.4.12) is to dampen the iteration, as in the scheme

$$\begin{aligned} 0 &= \mathcal{F}(C_{i+1/2}, C_i, C_i, C_i), \\ C_{i+1} &= (1 - \lambda)C_{i+1/2} + \lambda C_i, \end{aligned} \tag{16.4.16}$$

with $\lambda < 1$.

Initial Guesses and Boundary Conditions

We have focused on the iterative steps of time iteration and fixed-point iteration. We need to emphasize two details that may greatly affect an algorithm's efficiency. First, the initial guess in any iterative scheme is important. Second, many functional equations are not properly specified until one imposes some side-conditions, called boundary conditions. In perfect foresight models we have good answers to these problems.

First, knowledge of the steady state gives us a boundary condition. If the steady state capital stock is k^* and the steady state consumption is c^* , then we know $C(k^*) = c^*$. We can impose this on each iterate of time iteration or fixed-point iteration, or parameterize our solution family so as to make this true for each element.

Second, linearizing the problem around the steady state often provides us with a good initial guess. Here the tangent hyperplane of the stable manifold will be useful, since it is the linear approximation of the equilibrium decision rule in the neighborhood of the steady state.

16.5 Recursive Models with Nonlinear Equation Methods

The previous sections used solution methods that are closely related to the theoretical treatment of the pure mathematical problem, (16.4.5). However, our parametric approximation schemes reduce the problem to a finite-dimensional nonlinear equation in the unknown coefficients. In this section we basically ignore the dynamic nature of the true problem and treat the approximate Euler equation (16.4.6) as a nonlinear equation in the unknown parameters and apply standard methods for nonlinear equations to solve it.

This has the advantage that it actually formulates the problem in the most flexible fashion; that is, after making the approximation decisions implicit in (16.4.6), the problem is just a nonlinear equation in R^n , nothing more and nothing less. For example, we could use Newton's method; if Newton's method works, it has quadratic convergence asymptotically. Other methods, such as a homotopy method, could also be used. The choice will depend on the size of the problem and its curvature properties. It may be that some special structure of a problem will make time iteration or fixed-point iteration useful. However, by focusing on the nonlinear equation nature of (16.4.6), we now begin to consider a variety of methods that do not immediately come to mind when we view the functional problem (16.4.6) from an economic perspective. While economics ideas may be useful in devising numerical solutions, it is often useful to stop thinking economics and start thinking mathematics once one has a characterization of equilibrium.

We will now describe more fully the details of a Chebyshev-Newton projection method approach to the problem. First of all, we approximate C with the linear representation

$$\hat{C}(k; a) = \sum_{i=1}^n a_i \psi_i(k), \quad (16.5.1)$$

where $\psi_i(k) \equiv T_{i-1}(2(k - k_m)/(k_M - k_m) - 1)$ and n is the number of terms used. The domain D of our approximation will be some interval $[k_m, k_M]$. Since the special properties of Chebyshev polynomials apply to their restriction to $[-1, 1]$, we need to apply the linear transformation $2(k - k_m)/(k_M - k_m) - 1$ to k to permit us to form Chebyshev polynomials on D . k_m and k_M are chosen so that the solution will have k confined to $[k_m, k_M]$. In particular, $[k_m, k_M]$ must contain the steady state, a point that we can determine before calculations begin. The Euler equation implies the residual function

$$R(k; a) = u'(\hat{C}(k; a)) - \beta u'(\hat{C}(F(k) - \hat{C}(k; a); a))F'(F(k) - \hat{C}(k; a)). \quad (16.5.2)$$

We have several ways to compute a . First, we consider collocation. We choose n values of k , denoted by k_j , $j = 1, \dots, n$. Orthogonal collocation chooses the k_j to be the n zeros of ψ_{n+1} , which are themselves linear transforms of the Chebyshev zeros. We then form the collection of equations

$$R(k_j; a) = 0, \quad j = 1, \dots, n. \quad (16.5.3)$$

This is now a nonlinear system of equations in the unknown $a \in R^n$. To find a , we can use any nonlinear equation method. We choose to use Newton's method.

Table 16.1
Chebyshev coefficients for consumption function

i	$n = 2$	$n = 5$	$n = 9$	$n = 15$
1	0.0589755899	0.0600095844	0.0600137797	0.0600137922
2	0.0281934398	0.0284278730	0.0284329464	0.0284329804
3		-0.0114191783	-0.0113529374	-0.0113529464
4		0.0007725731	0.0006990930	0.0006988353
5		-0.0001616767	-0.0001633928	-0.0001634209
6			0.0000427201	0.0000430853
7			-0.0000123570	-0.0000122160
8			0.0000042498	0.0000036367
9			-0.0000011464	-0.0000011212
10				0.0000003557
11				-0.0000001147
12				0.0000000370
13				-0.0000000129
14				0.0000000052
15				-0.0000000015

Note: Each entry is the coefficient of the i th Chebyshev polynomial (over the interval $[0.333, 1.667]$) in the n -term approximation of the consumption policy function in (16.5.3) for the case discussed in section 16.5.

Table 16.1 displays an instance of this procedure where capital share is 0.25, $\gamma = -0.9$, and $[k_m, k_M] = [0.333, 0.667]$. The different columns display the solutions for a for various choices of n . The key fact to note is that the coefficients are stable as n increases. This is due to the orthogonality properties of the Chebyshev basis. This property is useful because one can first solve the $n = 2$ case, an easy problem solved rapidly, and then use that solution as an initial guess for higher choices of n . Note also how the coefficients drop rapidly in the $n = 5, 9$, and 15 cases. In light of theorem 6.4.2, this indicates that we are approximating a smooth function. Since the high-order Chebyshev coefficients are so small and stable as we increase n , we feel comfortable in accepting the solution to (16.5.3) with $n = 15$ as our solution to (16.4.2).

The Galerkin method is another alternative. We need to define a norm. Since we use Chebyshev polynomials as a basis, we will use the inner product

$$\langle f(k), g(k) \rangle \equiv \int_{k_m}^{k_M} f(k)g(k)w(k)dk,$$

where

$$w(k) \equiv \left(1 - \left(2 \frac{k - k_m}{k_M - k_m} - 1 \right)^2 \right)^{-1/2}.$$

With this choice of inner product, the basis is orthogonal. The Galerkin method computes the n projections

$$P_i(a) \equiv \int_{k_m}^{k_M} R(k; a) \psi_i(k) w(k) dk, \quad i = 1, \dots, n, \quad (16.5.4)$$

and chooses a so that $P(a) = 0$. Here the difficulty is that each $P_i(a)$ needs to be computed numerically. We use Gauss-Chebyshev quadrature because the basis elements are Chebyshev polynomials. That is, we approximate $P_i(a)$ with

$$\hat{P}_i(a) \equiv \sum_{j=1}^m R(k_j; a) \psi_i(k_j) \quad (16.5.5)$$

for some $m > n$, with the k_j being the m zeros of ψ_{m+1} . We then solve the nonlinear system

$$\hat{P}_i(a) = 0, \quad i = 0, \dots, n. \quad (16.5.6)$$

The system (16.5.6) can also be solved using any nonlinear equation solving algorithm. The differences between collocation and Galerkin turn out to be trivial for the problem in table 16.1, a fact that indicates the power of collocation even for nonlinear problems.

Alternative Bases

We approximated $C(k)$ with polynomials in k in our example, but this is only one possibility. We next consider the alternative of approximating the log of consumption with polynomials in $\log k$, that is,

$$\hat{C}(k; a) = e^{\sum_{i=0}^n a_i (\log k)^i}. \quad (16.5.7)$$

This functional form spans a different space of functions compared to the Chebyshev polynomial approximation scheme. By the Stone-Weierstrass theorem, both approximation schemes will do well as n goes to infinity, but the practical objective is to find an approximation to C involving a small number of free parameters.

After some adjustments we can combine the approximation scheme (16.5.7) with projection methods. Consider, for example, the collocation method. We first choose the collocation points. The functional form is no longer a polynomial in k ; instead, it

is a polynomial in $\log k$. This suggests using points x_i such that $\log x_i$ are zeros of Chebyshev polynomials in the interval $[\log k_m, \log k_M]$. With this set of collocation points we then solve the collocation equations just as we (16.5.3). The only difference is the functional form used for \hat{C} and the collocation points.

There is no general rule for which basis is better. For the example presented in table 16.1, the log basis does better. There are cases where polynomials in k will do better. For example, suppose that one wants to solve a linear-quadratic problem using projection methods. We know that the solution is linear in k , implying that a basis in k would do better than a basis on $\log k$. These observations indicate that it is valuable to be flexible in the choice of bases. One way to improve an approximation is to add terms from an existing basis, but another way is to try an alternative basis.

16.6 Accuracy Measures

An important question is how good is the resulting approximation and how many polynomial terms do we need to get a good approximation. There are two ways to do this. First, find another, more “reliable” way to solve the problem and compare the answer. Second, compute a measure of inaccuracy.

For this deterministic problem, we can do both. First, we compute a conservative, reliable benchmark using discretization procedures described in chapter 10. We get some idea of the accuracy of these projection method techniques by comparing it with the presumably very accurate solution from the discretized problem. The results are shown in table 16.2. We assumed a CRRA utility function, $u(c) = c^{\gamma+1}/(\gamma + 1)$ and a Cobb-Douglas production function $f(k) = Ak^\alpha$, $\alpha = 0.25$, where A is chosen so

Table 16.2
Policy function errors

k	y	c	$n = 20$	$n = 10$	$n = 7$	$n = 4$	$n = 2$
0.5	0.1253211	0.1010611	1(-7)	5(-7)	5(-7)	2(-7)	5(-5)
0.6	0.1331736	0.1132936	2(-6)	1(-7)	1(-7)	2(-6)	8(-5)
0.7	0.1401954	0.1250054	2(-6)	3(-7)	3(-7)	1(-6)	2(-4)
0.8	0.1465765	0.1362965	1(-6)	4(-7)	4(-7)	4(-6)	2(-4)
0.9	0.1524457	0.1472357	1(-6)	3(-7)	3(-7)	5(-6)	2(-4)
1.0	0.1578947	0.1578947	4(-6)	0(-7)	1(-7)	2(-6)	1(-4)
1.1	0.1629916	0.1683016	4(-6)	2(-7)	2(-7)	1(-6)	9(-5)
1.2	0.1677882	0.1784982	3(-6)	2(-7)	2(-7)	4(-6)	7(-6)
1.3	0.1723252	0.1884952	7(-7)	4(-7)	4(-7)	3(-6)	9(-5)

that the deterministic steady state is $k = 1$. We choose $\gamma = -0.9$. The third column gives the consumption choice as a function of capital indicated by the discrete state space method where we used 800,001 uniformly distributed capital stock levels in the interval $[0.5, 1.3]$. The fourth through eighth columns give the difference between the discrete solution and the projection method (collocation at Chebyshev zeros) solution for $n = 20, 10, 7, 4$, and 2, where n is the maximum degree Chebyshev polynomial included in the approximation. Note that the approximation of the aggregate consumption function for $n = 10$ disagrees with the discrete state space result by no more than one part in 10,000, an acceptable error for most purposes. Note that the $n = 10$ approximation appears to be better than $n = 20$. It is unclear which is better, since there is error in the discrete state space algorithm and the differences between the $n = 10$ and $n = 20$ solutions are within the error bounds for the discrete state space algorithm. This numerical experiment shows that the projection method works well for the discrete-time optimal growth model, demonstrating its usefulness even for the nonstandard functional equations that arise often in discrete-time economic models.

16.7 Tax and Monetary Policy in Dynamic Economies

We have primarily examined equilibria that are solutions to optimization problems. However, these methods can also be used to analyze models with distortions. We demonstrate this by applying them to simple dynamic models of equilibrium with money and taxes. Considerations of space limit our discussions below; the reader should turn to the cited papers for the details.

Taxation

Brock and Turnovsky (1980) show that equilibrium with taxation in the simple growth model described in section 11.5 solves the system

$$\begin{aligned}\dot{c} &= \gamma c(\rho - f'(k)(1 - \tau)), \\ \dot{k} &= f(k) - c - g,\end{aligned}\tag{16.7.1}$$

where $\tau(t)$ is the tax on capital income at time t , and $g(t)$ is government expenditure at t . The boundary conditions are

$$\begin{aligned}k(0) &= k_0, \\ 0 < \lim_{t \rightarrow \infty} |c(t)| &< \infty.\end{aligned}$$

The fact that equilibrium reduces to the differential equation (16.7.1) implies that we can depend on any of the methods we have used to solve differential equations. Following Judd (1985), we can proceed by a perturbation method analysis of (16.7.1) to examine various policy “shocks” using linearization and Laplace transform methods. The conceptual experiment is as follows: Let us assume that the “old” tax policy was constant, $\tau(t) = \bar{\tau}$, and that it has been in place so long that, at $t = 0$, the economy is at the steady state corresponding to $\bar{\tau}$. Note that this also assumes that for $t < 0$, agents expect that $\tau(t) = \bar{\tau}$ for all t , even $t > 0$. Hence, at $t = 0$, $k(0) = k^{ss}(\bar{\tau})$.

However, at $t = 0$, agents are told that future tax policy will be different. Say that they find out that $\tau(t) = \bar{\tau} + h(t)$, $t \geq 0$; that is, $h(t)$ will be the change in the tax rate at time t . Similarly spending is $\bar{g} + g(t)$. The new system is

$$\begin{aligned}\dot{c} &= \gamma c(\rho - f'(k)(1 - \bar{\tau} - h)), \\ \dot{k} &= f(k) - c - \bar{g} - g,\end{aligned}\tag{16.7.2}$$

together with the boundary conditions of (16.7.1). We could just solve the new nonlinear BVP. Instead we will use perturbation methods to approximate the effects.

We need to parameterize the new policy so that it fits the setup in section 13.4 for perturbing boundary value problems. We do this by defining

$$\tau(t, \varepsilon) = \bar{\tau} + \varepsilon h(t), \quad g(t, \varepsilon) = \bar{g} + \varepsilon g(t)\tag{16.7.3}$$

and the corresponding continuum of BVPs

$$\begin{aligned}c_t(t, \varepsilon) &= \gamma c(t, \varepsilon)(\rho - f'(k(t, \varepsilon))(1 - \tau(t, \varepsilon))), \\ k_t(t, \varepsilon) &= f(k(t, \varepsilon)) - c(t, \varepsilon) - g(t, \varepsilon),\end{aligned}\tag{16.7.4}$$

where we explicitly indicate the dependence of c and k on both ε and t .

We differentiate (16.7.4) with respect to ε and solve for $c_\varepsilon(t, 0)$ and $k_\varepsilon(t, 0)$ after imposing the asymptotic boundedness, all a direct application of (13.4.9). In particular, Judd (1985) shows that if $I = \dot{k}$ is net investment, the impact of ε on I is

$$I_\varepsilon(0) = \frac{\gamma c \rho}{1 - \bar{\tau}} H(\mu) + \mu G(\mu) - g(0),\tag{16.7.5}$$

where

$$G(s) = \int_0^\infty g(t)e^{-st} dt, \quad H(s) = \int_0^\infty h(t)e^{-st} dt,$$

$$\mu = \frac{\rho}{2(1-\bar{\tau})} \left(1 + \sqrt{1 - \frac{4\gamma(1-\bar{\tau})\theta_L\theta_c}{\sigma\theta_K}} \right),$$

$$\theta_K = \frac{k^*f'(k^*)}{f(k^*)}, \quad \theta_L = 1 - \theta_K, \quad \theta_c = \frac{f(k^*) - \bar{g}}{f(k^*)},$$

and σ is the elasticity of substitution between k and l in production. In this problem, μ is the positive eigenvalue of the linearized system, and θ_K , θ_L , and θ_c are the steady state shares of capital income, labor income, and consumption. $G(s)$ and $H(s)$ are the Laplace transforms of the policy innovation functions $g(t)$ and $h(t)$; note that (16.7.5) depends on sums of $g(t)$ and $h(t)$ discounted at the rate μ . This implies that if the tax policy change is h , the $\varepsilon = 1$ case in (16.7.4), then the first-order approximation to the new level of investment at $t = 0$ is $I_\varepsilon(0)$.

The formula (16.7.5) is an example of the kind of qualitative information revealed by perturbation exercises. First, future tax increases reduce investment. Second, government spending has an ambiguous impact on investment, with current spending crowding out investment but future spending “crowding in” investment. Third, since investment and output are related, we know that any initial increase in investment will immediately lead to an increase in output.

Local Analysis of Monetary Models

Monetary economic problems differ from other problems because of the special properties of money. One of the simplest models of money is the Sidrauski model. It assumes that each infinitely lived representative agent maximizes $\int_0^\infty e^{-\rho t} u(c, m) dt$ where c is consumption and m is real balances, subject to his budget constraint. Fischer (1979) shows that if θ is the constant rate of money creation, then equilibrium paths for c , m , the inflation rate π , and the capital stock k solve the system

$$0 = -[\rho + \pi]u_c(c, m) + u_m(c, m) + u_{cc}(c, m)\dot{c} + u_{cm}(c, m)\dot{m},$$

$$0 = [f'(k) + \pi]u_c(c, m) - u_m(c, m), \tag{16.7.6}$$

$$0 = \dot{k} - [f(k) - c],$$

where $\pi \equiv \theta - (\dot{m}/m)$.

Fischer looked for a Tobin effect in this model; that is, he examined the effect of inflation on capital accumulation. To do this, he computed the steady state, which is

characterized by $f'(k^*) = \rho$, $\pi = \theta$, and $c = f(k^*)$. The real side of the steady state is unaffected by inflation. He then asked how investment is affected by inflation when the capital stock is close to but not equal to the steady state. To do this, he linearized the system (16.7.6) around its steady state. The linear approximation implies a net investment process equal to $\dot{k} = -\lambda(k - k^*)$, where λ is the stable eigenvalue of the linearized system. Assuming reasonable utility functions, Fischer showed that if $k < k^*$, savings is higher in economies with higher rates of money creation, θ .

The two previous examples show how the perturbation methods we discussed in chapters 14 and 15 can be applied to local analysis of perfect foresight models. Local analysis of such models is a powerful technique that can be applied to both continuous-time models and discrete-time models.² Since we only need to compute the Jacobian of a dynamic system and its eigenvector-eigenvalue decomposition, we can apply these noniterative methods to many large systems.

Solving the Grossman-Weiss Model of Money

A recent paper by Li (1998) is a good example of how to rigorously approach the numerical analysis of a dynamic economic problem. She takes the Grossman-Weiss (1983) model of money demand. If θ is the constant growth rate of money supply and p_t is the price level, then define $q_1 \equiv (p_1 + \theta)/(1 + \theta)$ and $q_t \equiv p_t/(1 + \theta)$ for $t \geq 2$. Equilibrium in the model is a solution to the nonlinear second-order difference system

$$q_t + \phi\left(\frac{q_t}{q_{t+1}}\right)q_{t-1} = 1, \quad (16.7.7)$$

where ϕ is a known function depending on tastes and technology; in Li's example,

$$\phi(x) = \frac{\beta^{-1/\gamma}}{\beta^{-1/\gamma} + x^{(1+\gamma)/\gamma}},$$

where $u(c) = c^{1+\gamma}/(1 + \gamma)$ and β is the discount factor. The initial value q_1 is fixed by the stability condition $\lim_{t \rightarrow \infty} q_t < \infty$. It can be shown that the equilibrium satisfies $q_{t+1} = g(q_t)$ for some function g which we will compute. Applying (16.7.7) implies that g satisfies the fixed-point expression

2. Cooley and Hansen (1989) advocate an iterative approach similar to time iteration to solve discrete-time versions of these problems but do not present any evidence that their iterative approach has any advantages over the direct noniterative approach presented here and in previous papers.

$$g(x) = 1 - x\phi\left(\frac{g(x)}{g(g(x))}\right) \equiv \mathcal{F}(g). \quad (16.7.8)$$

The steady state price is $p^* \equiv (1 + \theta)/(1 + \phi(1))$.

Li first shows that if $3\phi'(1) \leq 1 - \phi(1)$, then the operator \mathcal{F} is a contraction mapping on a problem-specific space³ of functions X . Therefore the iteration

$$g^{j+1} = \mathcal{F}(g^j) \quad (16.7.9)$$

converges to the unique fixed point.

So far this is just standard existence theory. We want to develop a numerical approach and analyze its properties. It is important to realize that the properties of the abstract operator \mathcal{F} do not necessarily carry over to its computable approximations. Li recognizes this and proceeds to analyze a particular numerical approximation of the iteration in (16.7.9). Let X^h be the set of piecewise linear splines in X with nodes in $\{0, h, 2h, \dots, 1\}$, and let $\mathcal{F}^h(g)$ be the spline approximation of $\mathcal{F}(g)$. Since $\mathcal{F}^h(g)$ is defined by (16.7.8), it can be directly computed. Li shows that $\mathcal{F}^h(g)$ is a contraction on X^h and that the iteration $g^{j+1} = \mathcal{F}^h(g^j)$ converges linearly to a unique spline function. She then implements the fixed-point iteration for $h = 1/29$ and finds a spline which satisfies (16.7.7) with error less than 10^{-10} .

Li's careful and complete analysis is common in the mathematical literature but is unusual in the economics literature. Her approach is promising for many other fixed-point problems: Define the relevant space of computable functions, define a numerical operator on that space, demonstrate a contraction property for the *computable* approximation of the original operator, compute a fixed point of the computable operator, and show it to be close to the fixed point of the original operator. We should also note that her analysis of (16.7.7) applies to any problem of the form (16.7.8) not just the initial motivating example from monetary economics.

16.8 Recursive Solution of an OLG Model

The methods we used to produce recursive solutions of infinite-horizon models can also be used to solve the OLG model we first examined in section 16.1. More generally, these recursive methods can be used to compute equilibria of OLG models.⁴

3. We refer the reader to Li (1998) for the specific details.

4. Given the large literature on indeterminacy in OLG models, we must emphasize that any recursive equilibrium that we compute may represent just one out of many equilibria. Also there are sunspot equilibria that are being ignored by these methods. The goal here is to find one equilibrium, not all.

Recall the model from section 16.1. The natural state variable is the beginning-of-period capital stock k which is owned by the old. Let $S(k)$ be the savings of the young, $R(k)$ the marginal product of capital, $W(k)$ the marginal product of labor, and $L(k)$ the labor supply if k is the capital stock. These functions must satisfy the Euler equations:

$$\begin{aligned} 0 &= u'(L(k)W(k) - S(k)) - \beta u'(S(k)R(S(k)))R(S(k)), \\ 0 &= W(k)u'(L(k)W(k) - S(k)) - v'(L(k)), \\ R(k) &\equiv F_k(k, L(k)), \\ W(k) &\equiv F_l(k, L(k)). \end{aligned} \tag{16.8.1}$$

In (16.8.1), we view $S(k)$ and $L(k)$ as the unknown functions, substituting the definitions for $R(k)$ and $W(k)$ into the first two equations. We then parameterize S and L and solve for their unknown coefficients using any of the methods discussed above. The easiest to implement would be Newton's method, since that could be applied directly to (16.8.1). Both time and fixed-point iteration could also be implemented once we properly index the occurrences of $S(k)$ and $L(k)$. In all these methods, if we knew that the steady state was locally determinate and stable, it would be natural to impose steady state conditions requiring that S and L equal their steady state values when k is the steady state value of capital. This approach can be easily extended to the case of OLG models where agents live three, four, or more periods. As always, the key step is characterizing equilibrium in terms of some functional equation.

16.9 “Consistent” Capital Income Taxation

Optimal taxation of factor income is a typical example of dynamic inconsistency of optimal plans. Formulation and computation of “consistent” taxation policies is a difficult problem since closed-form solutions are generally unavailable (see Turnovsky and Brock 1980). In this section we will characterize dynamically consistent taxation of capital income to finance a public good, and we will use a projection technique to solve the resulting nonlinear equations.

We assume the continuous-time, inelastic labor supply, single-good continuous-time growth model with the addition of a public good. We assume that the inter-temporal utility function is

$$\max \int_0^\infty e^{-\rho t} (u(c + g) + v(g)) dt, \tag{16.9.1}$$

where g is a public good. We assume capital evolves according to

$$\dot{k} = f(k) - c - g. \quad (16.9.2)$$

This may appear to be a strange utility function, since it says that all consumption should be through the public good; we choose it because it is simple. If taxes were lump-sum, then all income would be taxed away to finance g ; hence lower tax rates indicate the degree to which the distortionary costs of taxation are recognized. We assume that public good is financed by a tax on capital income, $kf'(k)$, at a rate of τ . We will allow no bonds; hence $g = \tau kf'(k)$.

We focus on consistent equilibria in which all decisions depend only on the real state variables, which in this model is only aggregate capital. Therefore we are looking for an aggregate consumption function, $C(k)$, and the equilibrium tax function, $T(k)$. We will also compute the planner's value function, $V(k)$, which is defined to be the present value of social utility given that the current capital stock is k and that future governments will follow the consistent equilibrium in making tax policy. We will let $\Psi(k)$ denote total consumption, $C(k) + T(k)$.

Dynamic programming ideas impose several conditions for these functions. The dynamic programming problem of the representative agent shows that the aggregate consumption function will obey

$$0 = \Psi'(k)(f(k) - \Psi(k)) - \frac{u'(\Psi(k))}{u''(\Psi(k))} (\rho - f'(k)(1 - T(k))). \quad (16.9.3)$$

The policymaker wants to maximize the present value of utility, including the utility of public good consumption. Let $V(k)$ be the equilibrium value function for a policymaker when the current capital stock is k . Since the policymaker can determine the tax rate for only a moment, he will have no influence on contemporaneous consumption. Therefore, at the margin, any decision to increase taxation and public good consumption crowds out investment one for one. Given a private consumption function, $C(k)$, the policymaker solves the problem

$$\begin{aligned} \rho V(k) = \max_{\tau} & [u(C(k) + \tau kf'(k)) + v(\tau kf'(k))] \\ & + V'(k)(f(k) - C(k) - \tau kf'(k)). \end{aligned} \quad (16.9.4)$$

The first-order condition for the government is

$$u'(\Psi(k)) + v'(T(k)kf'(k)) = V'(k). \quad (16.9.5)$$

The three equations (16.9.3) and (16.9.4), and (16.9.5) characterize the three functions $V(k)$, $C(k)$, and $T(k)$, which describe the consistent equilibrium. Each unknown

Table 16.3
Dynamically consistent tax policy

$\lambda_g = 10^{-4}$					$\lambda_g = 10^{-3}$				
k	c	g	τ	s	k	c	g	τ	s
0.5	0.1360	0.0058	0.14	0.0322	0.5	0.1557	0.0125	0.30	0.0125
0.6	0.1547	0.0064	0.15	0.0213	0.6	0.1764	0.0142	0.32	-0.0004
0.7	0.1727	0.0071	0.15	0.0102	0.7	0.1983	0.0161	0.35	-0.0153
0.8	0.1903	0.0077	0.16	-0.0011	0.8	0.2192	0.0175	0.37	-0.0300
0.9	0.2073	0.0083	0.17	-0.0125	0.9	0.2382	0.0187	0.38	-0.0434
1.0	0.2241	0.0089	0.18	-0.0241	1.0	0.2563	0.0200	0.40	-0.0563
1.1	0.2406	0.0096	0.19	-0.0358	1.1	0.2744	0.0215	0.42	-0.0696
1.2	0.2568	0.0102	0.20	-0.0475	1.2	0.2925	0.0230	0.44	-0.0832
1.3	0.2729	0.0109	0.20	-0.0593	1.3	0.3104	0.0245	0.46	-0.0968
1.4	0.2887	0.0116	0.21	-0.0712	1.4	0.3281	0.0261	0.48	-0.1106

function will be approximated by a high-order polynomial. For the case of a valueless public good, the solution is known—set $T(k) = 0$, and $C(k)$ and $V(k)$ equal to their first-best values. This provides us with a test case.

Table 16.3 gives some results for two particular cases. It assumes that $u(c) = (c + g)^{\gamma_c + 1}/(\gamma_c + 1)$ and $v(g) = \lambda_g g^{\gamma_g + 1}/(\gamma_g + 1)$, with $\gamma_c = \gamma_g = -2$. We assume that $\rho = 0.05$ and $f(k) = 0.2k^{0.25}$. The entries give consumption (c), savings (s), government expenditure (g), and the tax rate (τ) at various values of the capital stock (k). We use Chebyshev polynomials and Chebyshev collocation. We first solved the model for $\lambda_g = 0$ and found that the result was accurate to several digits, giving us some confidence in the approach. The two panels in table 16.3 correspond to two positive values for λ_g . The results are interesting. As the value of the public good increases, the consumption function rises at all capital stocks, since the higher value of g causes higher taxation of investment income which in turn discourages savings. Also note that the tax rate rises as the economy grows through capital accumulation. This reflects the fact that the efficient ratio between public and private consumption should be roughly constant, and as the economy grows, the share of output that goes into consumption rises.

The other interesting feature is how the steady state of the economy is affected by the fiscal policy. When $\lambda_g = 0$, taxes are always zero and the capital stock converges to 1. However, when $\lambda_g = 10^{-4}$, the steady state tax rate is 0.16 and capital converges to 0.8, which is 20 percent below the no-tax case. As λ_g increases, the steady state tax rate increases and the steady state capital stock falls. Note, however, that government expenditures are always a small fraction of output and that the tax rates are moderate,

despite the fact that first-best for this problem would have all consumption be in the form of g . Therefore even the time-consistent solution involves a substantial amount of foresight and intertemporal forebearance on the part of successive governments.

This was obviously a stripped down analysis of an important problem. However, it shows that basic numerical methods can be used to model strategic decision-making by a foresighted policymaker with limited commitment powers.

16.10 Further Reading and Summary

This chapter has shown how to solve some simple perfect foresight dynamic economic models. While we have focused on competitive equilibrium models, that was not necessary. To solve for a time path of prices and allocations, we compute the dynamic demand and supply paths implied by a truncated path of prices and apply some nonlinear equation procedure to find the truncated price sequence that implies zero excess demand. This approach combines two kinds of problems we studied before. First, computing the dynamic supply and demand often involves solving optimal control problems at the level of the consumers and firms. Second, computing equilibrium prices is a problem in nonlinear equations.

To solve for the laws of motion and price laws in a recursive model, we construct an operator which characterizes the equilibrium laws and use projection methods to approximate them. These projection methods use approximation methods, integration techniques, plus nonlinear equations. In either case, solving perfect foresight models is made feasible by appropriately combining basic numerical methods.

Newton's method is increasingly used in this literature; for example, see Burdick (1994) and Gilli and Pauletto (1998). Even if the horizon is too long for Newton's method, we could still feed $E(P)$ to conventional equation solvers. Hughes Hallett and Piscitelli (1998) advocates reordering the equations and then using Gaussian substitution schemes. Juillard et al. (1998) compares Newton's method with Gaussian substitution methods. The tatonnement and fixed-point methods are just two possible approaches but, as with standard Gauss-Jacobi and fixed-point methods, are unlikely to be the best choices.

Exercises

- Assume an OLG model where agents live for four periods with utility function over consumption c_a at age a and labor supply l_a at age a equal to $\sum_{a=1}^4 c_a^{1+\gamma}/(1+\gamma) - \sum_{a=1}^3 l_a^\nu$. Also assume that the production function is $f(k, l) = k^\alpha l^{1-\alpha}$. Assume that $\alpha \in [0.2, 0.5]$, $\nu \in [1.1, 5]$, and $\gamma \in [-0.5, -5]$. Write a routine to compute steady states and their local determinacy. For the determinate cases, use

and compare the tatonnement, fixed-point iteration, Newton's method, Fair-Taylor, and the parametric path methods to compute the time path of interest rates and wages starting at a point where each cohort has half the steady state wealth for that cohort.

2. Reconsider exercise 1. For the determinate cases, compute the linear equilibrium decision rules in the neighborhood of the steady state.
3. Consider (16.4.2) for $u(c) = c^{1+\gamma}/(1 + \gamma)$ and $F(k) = k + Ak^\alpha$. Let $\gamma \in [-0.5, -5]$, $\alpha \in [0.2, 0.5]$, and $\beta \in [0.6, 0.99]$; choose A so that the steady state capital stock is 1. Use and compare the time iteration, fixed-point iteration, and Newton iteration methods with various bases. Which method with which bases does best in general? Which projection methods—collocation, Galerkin, least squares, or subdomain—do best?
4. Compute and compare solutions to the model in Auerbach, Kotlikoff, and Skinner (1983) using their Gauss-Jacobi scheme, Newton's method, and parameterized paths.

17

Solving Rational Expectations Models

The previous chapter presented solution methods for perfect foresight models of dynamic economic equilibrium. We next extend these models to include stochastic elements. As in the deterministic case, there are a few special models with explicit solutions. While these special cases are sufficient in some interesting contexts, they cannot be applied to most economic problems.¹

In this chapter we apply present basic methods to solve rational expectations models of stochastic dynamic competitive equilibrium. We begin with solving three models with exogenous state variables. The Lucas (1978) asset pricing model is particularly easy to solve numerically, since it reduces to a linear Fredholm integral equation. We will discuss both simulation and integral equation methods to solve the Lucas asset pricing model. Second, we examine monetary models; they lead to non-linear Fredholm integral equations which we solve by adapting the linear Fredholm equation methods presented in chapter 10. Third, we solve for a rational expectations equilibrium of an asset market with asymmetric information. This requires the full use of the projection method technology, since discretization methods cannot succeed.

We then examine models with endogenous state variables. We begin with the commodity storage problem examined by Gustafson (1958), Wright and Williams (1982, 1984), and Miranda and Helmburger (1988). We use the commodity model to illustrate the time iteration and fixed-point iteration solution methods in a stochastic context. An important aspect of the commodity market problem is that commodity stocks cannot be negative, a fact that causes equilibrium price and storage functions to have a kink. The kinks make simple polynomial approximations of the price function inappropriate, forcing us to use less efficient approximation methods. Wright and Williams instead solve the commodity model by parameterizing a critical conditional expectation that is a smooth function of the conditioning information and can be well-approximated by a low-order polynomial. Wright and Williams's idea to parameterize a smooth conditional expectations function results in a substantially better method.

We next discuss the solution of a simple stochastic growth model with both an endogenous and an exogenous state variable. We use this model to illustrate three kinds of procedures. We first present a standard projection method combining a Chebyshev polynomial approximation of the unknown function, a Gaussian quadrature method to compute the conditional expectation, and the Powell hybrid

1. See Hansen and Sargent (1990) for an excellent and wide-ranging discussion of linear-quadratic rational expectations models and their computational methods.

method to fix the unknown coefficients. Second, we develop fixed-point iteration methods, one using deterministic quadrature ideas and the other a simulation method used in den Haan-Marcet (1990). Third, we present a time iteration procedure.

The problems we tackle in this chapter show how to combine standard numerical techniques for approximation, integration, and nonlinear equations to solve rational expectations models. We finish the chapter by indicating the robustness of these methods. Section 17.9 indicates how they could be applied to more general models, with several states, agents, and goods, and with distortions. Also the matching of computational method with economic examples in this chapter (and in the literature) has little substantive content, since each method discussed below could be applied to each of the problems. In summary, this chapter presents several methods for solving rational expectations models.

17.1 Lucas Asset-Pricing Model

One of the simplest rational expectations problems is asset pricing with homogeneous investors. In this section we present the Lucas (1978) asset pricing model and use various procedures to compute the equilibrium asset pricing rule.

The Model

Assume that there is a single asset that pays dividends, y_{t+1} , according to the AR (1) stochastic process,

$$y_{t+1} = \rho y_t + \varepsilon_{t+1}, \quad (17.1.1)$$

where ε_t is an i.i.d. innovation process. Suppose that there is no other source of income in this economy and that all investors have a common concave von Neumann-Morgenstern utility function over the consumption process c_t equal to $E\{\sum_{t=0}^{\infty} \beta^t u(c_t)\}$. If we let p_t denote the asset price at time t , then the equilibrium price process must satisfy

$$u'(y_t)p_t = \beta E\{u'(y_{t+1})(y_{t+1} + p_{t+1})|y_t\}. \quad (17.1.2)$$

Equation (17.1.2) is an expression for the time series p_t and as such something that is difficult to compute directly. If instead we define $p(y)$ to be the ex-dividend price of a share in any period when the dividend is y , then Lucas (1978) shows that $p(y)$ solves

$$u'(y_t)p(y_t) = \beta E\{u'(y_{t+1})(y_{t+1} + p(y_{t+1}))|y_t\}. \quad (17.1.3)$$

We can rewrite (17.1.3) as

$$p(y_t) = \beta E \left\{ \frac{u'(y_{t+1})}{u'(y_t)} y_{t+1} | y_t \right\} + \beta E \left\{ \frac{u'(y_{t+1})}{u'(y_t)} p(y_{t+1}) | y_t \right\}. \quad (17.1.4)$$

Equation (17.1.4) has the form of the linear Fredholm integral equation

$$p(y) = g(y) + \beta \int K(y, z) p(z) dz, \quad (17.1.5)$$

where

$$g(y) = \beta \int \frac{u'(z)}{u'(y)} z q(z|y) dz,$$

$$K(y, z) = \frac{u'(z)}{u'(y)} q(z|y),$$

and $q(z|y)$ is the density of the time $t + 1$ dividend conditional on the time t dividend.

Solution by Simulation

We first use a simple simulation strategy to solve (17.1.3). Simple recursion arguments show that $p(y)$ can be expressed as

$$p(y_0) = (u'(y_0))^{-1} E \left\{ \sum_{t=1}^{\infty} \beta^t u'(y_t) | y_0 \right\}. \quad (17.1.6)$$

Therefore we need only to approximate the infinite sum in (17.1.6). More precisely, we first draw a sequence of T i.i.d. innovations, ε_t , and form the dividend sequence $y_t, t = 1, \dots, T$, implied by the process (17.1.1) with y_0 given. Since the sum in (17.1.6) is infinite, T needs to be large enough so that the truncation error of using (17.1.6) is acceptably small. We next define the sum

$$P(y_0; \varepsilon) = (u'(y_0))^{-1} \sum_{t=1}^T \beta^t u'(y_t). \quad (17.1.7)$$

$P(y_0; \varepsilon)$ is the price of the asset if the time zero dividend is y_0 , the ε sequence is perfectly foreseen, and the world ends at time T . However, the ε_t shocks are not known at $t = 0$. We then repeat this for several draws of $\varepsilon \in R^T$; denote the j 'th sequence as $\varepsilon^j, j = 1, \dots, m$. Each ε^j sequence implies a different dividend sequence and a different value for $P(y_0; \varepsilon^j)$. Our estimate of $p(y_0)$, denoted $\hat{p}(y_0)$, is the average of the $P(y_0; \varepsilon)$:

$$\hat{p}(y_0) = \frac{1}{m} \sum_{j=1}^m P(y_0; \varepsilon^j). \quad (17.1.8)$$

If we want to approximate the price function $p(y)$, we need to repeat this for a variety of y_0 values. We implement (17.1.8) for N such initial values, denoted y_0^i , $i = 1, \dots, N$, to generate the $\hat{p}(y_0^i)$ estimate of $p(y_0^i)$. We then use these data to construct an approximation for the pricing function. Since the procedure behind (17.1.7) is a Monte Carlo method, the errors in the $\hat{p}(y_0^i)$ estimates will be nontrivial. Therefore, to compute an approximation to the $p(y)$ function, it is best to use several y_0^i values, use different ε draws for each y_0^i , and then approximate $p(y)$ by regressing the $\hat{p}(y_0^i)$ values on the y_0^i values.

Simulation methods are most appropriate when the desired accuracy is moderate and the dividend process is complicated. For example, if dividends follow

$$y_t = a_0 + a_1 y_{t-1} + a_2 y_{t-2} + \dots + a_l y_{t-l} + \varepsilon_t,$$

then the l lags create an l -dimensional structure which would challenge other methods. However, simulation methods would be applied just as above.

Discrete-State Approximation

Tauchen (1991) recognized that (17.1.3) is a linear Fredholm integral equation of the second kind, and applied the procedures similar to those described in section 10.8. The key step replaces the continuous-space Markov process over y with a finite-state Markov process. We apply (10.8.7) directly to (17.1.5), using the Gauss-Hermite quadrature nodes y_i and weights ω_i . This results in the linear system of equations for the unknown values $p(y_i)$,

$$p(y_i) = g(y_i) + \beta \sum_{j=1}^n p(y_j) \frac{u'(y_j)}{u'(y_i)} \omega_j \frac{q(y_j|y_i)}{w(y_j)}. \quad (17.1.9)$$

We solve (17.1.9) to compute approximations to $p(y)$ for $y_i \in Y$; call the solutions \hat{p}_i . The procedure so far just approximates $p(y)$ on Y . To approximate p elsewhere, we use the Nystrom extension as defined in equation (10.8.10). The final approximation is

$$\hat{p}(y) = g(y) + \beta \sum_{j=1}^n \hat{p}_j \frac{u'(y_j)}{u'(y)} \omega_j \frac{q(y_j|y)}{w(y_j)}, \quad (17.1.10)$$

which is defined for all y .

17.2 Monetary Equilibrium

We next develop methods to compute equilibrium in a stochastic monetary model. More precisely, assume an infinite-life representative agent model with utility function $u(c, m)$ over consumption c and real balances m , and discount factor β . We assume that the (real) endowment process, w_t , is stochastic with w_t known before consumption is chosen in period t . Let p_t be the price level. The consumption choice and level of real balances are functions of the current wage. However, in equilibrium, consumption must equal the endowment. Assume that θ_t is the ratio of money stock at time t to that at time $t - 1$; hence $\theta - 1$ is the money growth rate in period t . We assume lump-sum subsidies (or lump-sum taxes) are used to finance the monetary policy. The equilibrium price and real balances path then satisfy

$$u_c(w_t, m_t)m_t = \beta E \left\{ (u_c(w_{t+1}, m_{t+1}) + u_m(w_{t+1}, m_{t+1})) \frac{m_{t+1}}{\theta_{t+1}} | w_t, \theta_t \right\}. \quad (17.2.1)$$

We will first assume that money growth is constant, equal to θ , and that w_t are i.i.d. random variables with density $f(w)$. In this case the current (exogenous) wage is a sufficient statistic for the system, and real balances are a function of the current wage. The equilibrium function for real balances can be written $m(w)$ and must satisfy

$$u_c(w_t, m(w_t))m(w_t) = \beta E \left\{ (u_c(w_{t+1}, m(w_{t+1})) + u_m(w_{t+1}, m(w_{t+1}))) \frac{m(w_{t+1})}{\theta} | w_t \right\}. \quad (17.2.2)$$

We immediately recognize that (17.2.2) is a nonlinear Fredholm integral equation. We solve it by generalizing the methods we used for linear Fredholm equations. We first choose some integration formula with n nodes $\{w_1, \dots, w_n\}$ and n weights $\{\omega_1, \dots, \omega_n\}$ for the integral in (17.2.2), arriving at the finite set of equations

$$u_c(w, m(w_i))m(w_i) = \beta \frac{1}{n} \sum_{j=1}^n (u_c(w_j, m(w_j)) + u_m(w_j, m(w_j))) \omega_j m(w_j) \theta^{-1} f(w_j), \quad (17.2.3)$$

where $i = 1, \dots, n$. We fix the n values of $m(w_i)$ by solving the n nonlinear algebraic equations in (17.2.3).

We can also take a projection approach. We begin by approximating m with a polynomial, \hat{m} ,

$$\hat{m}(w; a) = \sum_{j=1}^n a_j w^{j-1},$$

and then rewrite the equilibrium condition (17.2.2) as an integral equation for the approximation m :

$$\begin{aligned} u_c(w, \hat{m}(w; a)) \hat{m}(w; a) &= \beta \int (u_c(z, \hat{m}(z; a)) \\ &\quad + u_m(z, \hat{m}(z; a))) \hat{m}(z; a) \theta^{-1} f(z) dz. \end{aligned} \quad (17.2.4)$$

Our task is to find a set of coefficients, a , which approximately solves (17.2.4). To do this, we construct the finite set of equations

$$\begin{aligned} u_c(w, \hat{m}(w_i; a)) \hat{m}(w_i; a) &= \frac{1}{n} \beta \sum_{j=1}^n (u_c(w_j, \hat{m}(w_j; a)) \\ &\quad + u_m(w_j, \hat{m}(w_j; a))) \hat{m}(w_j; a) \theta^{-1} f(w_j), \end{aligned} \quad (17.2.5)$$

where $i = 1, \dots, n$. We fix the n undetermined coefficients in a by solving the n nonlinear algebraic equations in (17.2.5). The discretization approach in (17.2.5) is equivalent to a collocation strategy. One could also use least squares, Galerkin, or subdomain conditions.

This is a simple model of monetary equilibrium, but we must make some cautionary remarks. We have ignored important features of monetary theory, such as indeterminacy and sunspot equilibria. Our recursive formulation implicitly assumes uniqueness and excludes dependence on sunspotlike variables.

17.3 Information and Asset Markets

We next develop numerical methods for a model of asymmetric information in asset markets. This is an area of economic research that has relied almost exclusively on models with very special functional forms—exponential utility and Gaussian returns. They often critically rely on the presence of “noise traders” whose behavior is insensitive to price and information. We see that projection methods can handle these problems with ease and apparent accuracy, and do so without making any ad hoc deviations from agent rationality.

We assume three groups of identical investors, each individual having W dollars per person to invest in two assets. The safe asset pays out R dollars in the second period per dollar invested, and the risky asset pays out Z dollars per share. Therefore, if an investor buys ω shares of the risky asset at price p dollars per share, his second

period consumption will be $c = (W - \omega p)R + \omega Z$, and the first-order condition for ω will be

$$0 = E\{u'(c)(Z - pR)|I\}, \quad (17.3.1)$$

where I is the investor's information set. We will assume that there is only one share of the risky asset available; hence, the price p must be set so that the total demand for the risky asset is one.

We examine a generalization of Grossman's (1976) model. We assume that three types of investors have different information. As in Grossman, Z is normally distributed, $Z \sim N(\mu, \sigma^2)$, and an agent of type i receives a signal $y_i = Z + \varepsilon_i$, where $\varepsilon_i \sim N(\mu, \sigma_\varepsilon^2)$ and are independently distributed. We will assume that each agent initially has W dollars in cash.

We deviate from Grossman by permitting general utility functions. The equilibrium price of the risky asset is a function of the distribution information $p = p(y)$, since each agent can condition only on the price and his own information, which implies that his equilibrium trading strategy can be expressed as $\omega_i = \omega_i(p, y_i)$. Equilibrium is then any price function $p(y)$ and trading function triple $(\omega_1(p, y_1), \omega_2(p, y_2), \omega_3(p, y_3))$ that satisfies two sets of conditions. First, each agent must be on his demand curve, implying that

$$E_{y, Z} \{u'_i(C_i(y, Z))(Z - p(y)R)|p, y_i\} = 0, \quad i = 1, 2, 3, \quad (17.3.2)$$

where $C_i(y, Z) \equiv (W - \omega_i(p(y), y_i)p(y))R + \omega_i(p(y), y_i)Z$. Second, the market clears for each possible information distribution; that is, for all ω , total demand equals one, the supply, implying that

$$\sum_1^3 \omega_i(p(y), y_i) - 1 = 0. \quad (17.3.3)$$

We have four unknown functions: $p(y)$, $\omega_1(p, y_1)$, $\omega_2(p, y_2)$, and $\omega_3(p, y_3)$. They are hopefully identified by the four conditions in (17.3.2) and (17.3.3). Do there exist such functions? That is an existence question that we do not address. In fact we will proceed without any assurance that there does exist any equilibrium. This may make some uncomfortable. We proceed anyway for three reasons. First, if we did have an existence proof, or if we used the approximate equilibrium notions presented in Allen (1985a, b, 1992) and Anderson and Sonnenschein (1982) we would proceed as below. Second, if equilibrium does not exist, then it would be surprising if the method below would work. Third, any candidate solution produced by the method below that passes our diagnostics is an ε -equilibria for small ε ; if there are no pure equilibria,

then they provide natural alternative solutions wherein the agents make small optimization errors. None of this is to say that existence theory is unimportant, just that it is not an absolute prerequisite to doing computational work.

We return to computing an equilibrium. It is impossible to solve for (17.3.2) and (17.3.3) exactly; we attempt to find good polynomial approximations to the critical functions. We approximate the ω_i , $i = 1, 2, 3$, with the finite-order polynomials

$$\hat{\omega}_i(p, y_i) = \sum_{j,k=0}^n a_{jk}^i p^j y_i^k, \quad i = 1, 2, 3, \quad (17.3.4)$$

and we approximate $p(y)$ with a finite-order polynomial

$$\hat{p}(y) = \sum_{j,k,l=0}^n a_{jkl} y_1^j y_2^k y_3^l. \quad (17.3.5)$$

We use polynomials² in (17.3.4)–(17.3.5) to approximate the price and trading functions; this choice is motivated by the belief (or, more precisely, the guess) that the true equilibrium price and trading functions are smooth in y .

To identify the unknown coefficients in p and ω_i , we replace the conditional expectations conditions in (17.3.2) with a finite number of projections of its unconditional expectation, that is, we form the system of equations

$$0 = E_{y,Z} \{ u'_i(\hat{C}_i(y, Z))(Z - \hat{p}R) p^l y_i^j \}, \quad j, l = 0, 1, \dots, n, i = 1, 2, 3, \quad (17.3.6)$$

where $\hat{C}_i(y, Z) \equiv (W - \omega_i(\hat{p}(y), y_i)\hat{p}(y)) R + \hat{\omega}_i(\hat{p}(y), y_i)Z$. We also impose a finite number of projection conditions on the market-clearing condition, (17.3.3):

$$0 = E_{y,Z} \left\{ \left(\sum_1^3 \hat{\omega}_i(p, y_i) - 1 \right) y_1^j y_2^k y_3^l \right\}, \quad j, k, l = 0, 1, \dots, n. \quad (17.3.7)$$

We have reduced an infinite-dimensional problem, (17.3.2)–(17.3.3), to (17.3.6)–(17.3.7) which has a finite number of unknowns. We next need to replace the integrals in (17.3.6)–(17.3.7) with numerical quadrature formulas; Bernardo and Judd (1994) use product Gaussian quadrature. This step transforms (17.3.6)–(17.3.7) into a computable system of equations. At this point we can now solve the nonlinear system for the unknown coefficients; Bernardo and Judd use the Powell hybrid method because they were able to find satisfactory initial guesses. Otherwise, one could use a homotopy approach to guarantee convergence to a solution.

2. In practice, one should use orthogonal polynomial representations; we use polynomials here to reduce the notational burden.

Note that we don't know if there is a solution to the finite-dimensional system (17.3.6)–(17.3.7), with or without quadrature approximations of the integrals. Even if there is an equilibrium to the underlying infinite-dimensional model in (17.3.2)–(17.3.3), there may not be a solution to the finite-dimensional approximate system (17.3.6)–(17.3.7). Bernardo and Judd find solutions that satisfy the nonlinear equation solver's stopping rule, but that only indicates the existence of approximate solutions. Even if we have a solution of (17.3.6)–(17.3.7), we need to ask if it is satisfactory. To do so, Bernardo and Judd evaluate versions of (17.3.6) with some $j, l > n$ to see if there were economically significant errors in the first-order conditions. If significant errors are detected, then one should begin again with higher-order polynomials. The market-clearing condition (17.3.7) is only a weighted average condition; to see if it is an acceptable approximation, Bernardo and Judd compute the excess demand at values of y not used in the procedure; if the excess demand is excessive at too many values of y , we should also begin again. Bernardo and Judd find examples where solutions using cubic polynomials do very well when tested by these diagnostics. That may not always be the case, which is why these diagnostics must be evaluated.

This asset market example is a particularly interesting application of the projection method. Many dynamic economic models, such as the Lucas asset pricing model, can be solved by using a finite-state approximation, and the collocation method often implicitly reduces to such an approach. Theory tells us that this problem cannot be solved in an accurate and robust fashion with a finite-state approximation. If there were only a finite number of information states, then equilibrium will generically be fully revealing; see Radner (1979). Not only are discrete-state approximations probably inefficient for asymmetric information models but would inappropriately pre-judge the nature of the final outcome. Therefore, if we are interested in a robust analysis of partially revealing rational expectations equilibria, we cannot use discrete approximations.

17.4. Commodity Storage Models

Euler equation methods for rational expectation models were first developed in studying commodity markets, beginning with Gustafson (1958) and continued with papers by Wright and Williams (1982a, 1984), and Miranda and Helmburger (1988). Numerical methods are almost the only way to analyze this model.³ We discussed the

3. Aiyagari et al. (1989) examined a special case with a closed-form solution.

dynamic programming solution to this problem in chapter 12. We study this problem because it is an important class of models and clearly highlights many of the problems involved in solving rational expectations models generally.

Let p_t be the price in period t , x_t the cumulative postharvest stock in period t , I_t the inventory on hand at the beginning of period t , and r the constant interest rate. We assume that $D(p)$ is the demand function and that the current price equals $P(c)$ if current consumption is c ; hence $c = D(P(c))$. Let θ_t be the period- t harvest; hence the period- t postharvest stock is $x_t = I_t + \theta_t$, and inventories follow the law $I_{t+1} = x_t - D(p_t)$. Let $g(I)$ be the marginal convenience value of inventories, which is also the rental return to holding inventories.

Prices are assumed set by a competitive market with risk-neutral speculators. If stocks are low, then the current price may be so high relative to the expected future price that the speculators hold nothing; otherwise, speculators will increase their holdings until the current price equals the discounted expected future price plus convenience value. Therefore competitive equilibrium is characterized by the conditions:

$$\begin{aligned} I_{t+1} &= x_t - D(p_t) \geq 0, \\ p_t &\geq g(0) + \beta E\{p_{t+1}|p_t\}, \\ 0 &= (p_t - g(I_{t+1}) - \beta E\{p_{t+1}|p_t\})I_{t+1}. \end{aligned} \tag{17.4.1}$$

These conditions constitute a simultaneous and dynamic set of relations among price, inventories, and demand.

Equilibrium without Stockouts

If we assume that $g(0) = \infty$, the nonnegativity condition $I_t \geq 0$ will never bind, and we can ignore it. In this “no-stockout” case, the equilibrium condition reduces to the simple equations

$$x_{t+1} = x_t - D(p_t) + \theta_{t+1}, \tag{17.4.2}$$

$$p_t = g(x_t - D(p_t)) + \beta E_t\{p_{t+1}|p_t\}. \tag{17.4.3}$$

While g has the economic interpretation of convenience return, it can also be viewed as a barrier function approach to solving the nonnegativity constraints in (17.4.1), since it makes it infinitely costly to hit the nonnegativity constraint. Equation (17.4.3) is based only on the first-order condition, and it is simpler than the complementary slackness expressions of (17.4.1). In many cases one can introduce a barrier term

like g , possibly giving it economic interpretation, and proceed using only first-order conditions.

A natural state variable for this problem is x_t . In a stationary equilibrium, all other equilibrium variables, in particular, the price and carryover, will depend solely on x_t . In our first solution method, we characterize the equilibrium in terms of the carryover rule. Let $I(x)$ be the carryover rule, equal to $x - D(p)$ if the current postharvest stock is x . Then consumption equals $x - I(x)$, and price equals $P(x - I(x))$. If $I(x)$ is carried over today, the next period's total postharvest stock equals $I(x) + \theta$. The Euler equation (17.4.2) and equation (17.4.3) can be combined to imply that

$$P(x - I(x)) = g(I(x)) + \beta E\{P(I(x) + \theta - I(I(x) + \theta))\}. \quad (17.4.4)$$

Equation (17.4.4) characterizes $I(x)$. We now turn to methods for solving (17.4.4).

We first approach (17.4.4) in the same way we approached dynamic models in chapter 16. If we define an operator $\mathcal{F}: (C^0)^5 \rightarrow C^0$,

$$\begin{aligned} \mathcal{F}(I_1, I_2, I_3, I_4, I_5)(x) &= P(x - I_1(x)) - g(I_2(x)) \\ &\quad - \beta E\{P(I_3(x) + \theta - I_4(I_5(x) + \theta))\}, \end{aligned} \quad (17.4.5)$$

then finding a solution to (17.4.4) is equivalent to finding a function $I(x)$ such that $0 = \mathcal{F}(I, I, I, I, I) \equiv \mathcal{N}(I)$. We have the same range of methods available to use here as in chapter 16, since the only added wrinkle here is the presence of an expectation. We will examine both the time iteration and fixed-point iteration methods presented in chapter 16.

Time Iteration

Time iteration is implicitly defined in

$$P(x - I^{k+1}(x)) - g(I^{k+1}(x)) = \beta E\{P(I^{k+1}(x) + \theta - I^k(I^{k+1}(x) + \theta))\}. \quad (17.4.6)$$

Specifically, given the function I^k and some x , we can solve for the value of $I^{k+1}(x)$ that solves (17.4.6); since this can be done for each x , we have defined the function I^{k+1} . The advantage of this procedure is that convergence is likely (often easily proved), for it follows a simple dynamic economic logic. However, convergence will be slow.

The iteration in (17.4.6) is an abstract, ideal mathematical process that is useful for economic theory but cannot be exactly represented on the computer. There are two basic numerical issues to implementing (17.4.6). First, we need to choose how to approximate the $I^k(x)$ functions. Let $\hat{I}(x; a)$ denote the functional form we use where a is the vector of parameters in the approximation.

Second, we need to approximate the expectation in (17.4.6); we use the integration formula

$$E\{P(\hat{I}(x; a) + \theta - \hat{I}(\hat{I}(x; a) + \theta; a))\} \doteq \sum_{i=1}^m \omega_i P(\hat{I}(x; a) + \theta_i - \hat{I}(\hat{I}(x; a) + \theta_i; a)) \quad (17.4.7)$$

for some set of θ_i nodes and ω_i weights. The choice of quadrature formulas in (17.4.7) depends on the nature of the problem. If $P(c)$ is smooth, we expect $I(x)$ to also be smooth and smooth \hat{I} approximation schemes are appropriate. If $P(c)$ and the $\hat{I}(x; a)$ are both smooth, then Gaussian quadrature rules are likely to be efficient if θ has a smooth density. Note how the form of \hat{I} is important. If $\hat{I}(x; a)$ is not smooth in x , such as would be the case if \hat{I} were piecewise linear interpolation, then it may not be wise to use a quadrature formula designed for smooth integrands; it is the form of, \hat{I} , not $I(x)$, which is critical since we are integrating \hat{I} , not I , in (17.4.6). Algorithm 17.1 implements time iteration to determine the coefficients of \hat{I} .

Algorithm 17.1 Time Iteration in (17.4.6)

Initialization. Choose a finite set of points $x_i \in X$, and an approximation scheme $\hat{I}(x; a)$. Choose stopping criterion $\varepsilon > 0$.

Step 1. For each $x_i \in X$, solve for I_i in

$$P(x_i - I_i) = g(I_i) + \beta \left(\sum_{j=1}^m \omega_j P(I_i + \theta_j - \hat{I}(I_i + \theta_j; a^k)) \right).$$

Step 2. Determine a^{k+1} such that $\hat{I}(x; a^{k+1})$ approximates the (I_i, x_i) data generated in step 1.

Step 3. STOP if $\|a^k - a^{k+1}\| < \varepsilon$; otherwise go to step 1.

We should note a significant but subtle difference between what we do here and what was done in (17.1.8) in the asset pricing model. One interpretation of the (17.1.8) approach is that the continuous-state dividend process is approximated by a finite grid consisting of the x_i points and that (17.1.8) solves a finite-state problem. In this problem, solving the Euler equation for only the finite collection of x_i is not equivalent to assuming that the commodity inventory process is confined to a discrete grid. This is clear because $x_i + \theta_j$, a value for the future stock level that appears in the algorithm, may not be a member of X . If we forced the state x to lie on the grid in each period then the Euler equation would generally not have a solution. Euler equation methods are basically inconsistent with discrete endogenous state problems.

These points also make clear the difference between Euler equation methods and Bellman equation methods from dynamic programming. One can use finite-state approximations for dynamic programming problems, since a finite-state dynamic programming problem does have a Bellman equation, whereas there is no Euler equation for a problem with a discrete choice set.

Fixed-Point Iteration

A simple fixed-point iteration scheme rewrites (17.4.6) as

$$\begin{aligned} I^{k+1}(x) &= x - P^{-1}(\beta E\{P(I^k(x) + \theta - I^k(I^k(x) + \theta))\} + g(I^k(x))) \\ &\equiv \mathcal{G}(I)(x). \end{aligned} \quad (17.4.8)$$

We also choose some integration formula for the integrals in (17.4.8) to arrive at an approximation of \mathcal{G} :

$$\hat{\mathcal{G}}(I)(x) = x - P^{-1}\left(\beta \sum_{i=1}^m \omega_i P(I(x) + \theta_i - I(I(x) + \theta_i)) + g(I(x))\right), \quad (17.4.9)$$

where ω_i and θ_i are the integration weights and nodes. A computer implementation of fixed-point iteration combines the \hat{I} approximation scheme and $\hat{\mathcal{G}}$ to arrive at algorithm 17.2.

Algorithm 17.2 Fixed-Point Iteration in (17.4.8)

Initialization. Choose a finite set of points $x_i \in X$, and an approximation method $\hat{I}(x; a)$.

Step 1. Evaluate $I_i = \hat{\mathcal{G}}(I)(x_i)$ for each $x_i \in X$.

Step 2. Determine a^{k+1} such that $\hat{I}(x; a^{k+1})$ approximates the (I_i, x_i) data generated in step 1.

Step 3. STOP if $\|a^k - a^{k+1}\|$ is small; otherwise, go to step 1.

Equilibrium with Stockouts

In some cases we will not want to include convenience value in our analysis. Then $g \equiv 0$, and it becomes possible that the nonnegativity constraint will bind in equilibrium with nontrivial probability. In this case the equilibrium conditions (17.4.1) imply the complementary slackness conditions on $I(x)$:

$$P(x - I(x)) = \begin{cases} \beta E\{P(I(x) + \theta - I(I(x) + \theta))\}, & \text{if } I(x) > 0, \\ P(x), & \text{if } I(x) = 0. \end{cases}$$

In this case we will focus on the equilibrium prices instead of the equilibrium inventory holdings.⁴ Either all current supply is consumed, in which case the price is the demand price for that quantity, or some is stored, in which case the current price must equal the expected future value. In equilibrium the current price is the greater of those two prices, implying the dynamic relation

$$p_t = \max[\beta E_t\{p_{t+1}\}, P(x_t)].$$

We solve for the equilibrium price function, denoted $h(x)$. Since $h(x)$ is the current price in a stationary equilibrium, (17.4.1) implies the functional equation

$$h(x) = \max[\beta E\{h(\theta + (x - D(h(x))))\}, P(x)]. \quad (17.4.10)$$

Theory (see Gustafson 1958) tells us that there does exist a solution to this equation, that it is nonnegative, continuous, nonincreasing, and that there is a p^* such that $h(x) = P(x)$ whenever $P(x) \geq p^*$ and $h(x) > P(x)$ whenever $P(x) < p^*$. Equation (17.4.10) can be solved in two ways. Gustafson parameterized $h(x)$ in a piecewise-linear fashion and computed equilibrium using the time iteration scheme

$$h_{j+1}(x) = \max[\beta E\{h_j(\theta + (x - P^{-1}(h_{j+1}(x))))\}, P(x)].$$

Deaton (1992) also parameterized h in a piecewise-linear fashion (with 100 nodes) and executed the iteration

$$h_{j+1}(x) = \max[\beta E\{h_j(\theta + (x - P^{-1}(h_j(x))))\}, P(x)],$$

which is an application of the fixed-point iteration scheme.

This shows that we can use the same methods for the constrained case as for the unconstrained case, but we must be careful due to the kink. Piecewise-linear functions can handle the kink but at some cost, since the presence of kinks makes it more difficult to solve the nonlinear equations implicit in time iteration. This fact makes fixed-point iteration attractive. However, one needs a large set of nodes in either case to get a good approximation. While these problems are not important for this small problem, they will be far more complex problems. This leads us to ask if there is a better way to handle problems with constraints.

Wright-Williams Smoothing

The procedures above focused on finding an approximation to either the inventory holding rule, $I(x)$, or the equilibrium price rule, $h(x)$. These functions will have kinks

4. We could have taken this approach to the no-stockout case also. We are switching approaches to demonstrate the variety of possible methods.

if a nonnegativity constraint binds, a feature that complicates the approximation problem. In particular, we cannot take advantage of the strong approximation methods available for smooth functions.

Wright and Williams (1984) proposed that one instead focus on approximating the conditional expectation $E\{p_{t+1}|I_{t+1}\}$. This conditional expectation is also a sufficient description of equilibrium, since I_{t+1} completely describes the situation at the end of period t . The advantage is that $E\{p_{t+1}|I_{t+1}\}$ is likely to be smooth in I_{t+1} , making it easier to approximate and permitting the use of more powerful approximation methods. This insight has turned out to be of substantial value in solving rational expectations problems.

We first define $\psi(I_{t+1}) = E\{p_{t+1}|I_{t+1}\}$ to be today's expectation of tomorrow's price conditional on the end-of-period- t grain inventory being I_{t+1} . With this function we can compute the current price and inventory decisions. If the postharvest stock is x , the current price and end-of-period inventory must satisfy the pair of equations

$$\begin{aligned} p &= \max[P(x), \beta\psi(I)], \\ I &= x - D(p). \end{aligned} \tag{17.4.11}$$

The system (17.4.11) has a unique solution for any x as long as ψ is nondecreasing function. Let $h(x; \psi)$ and $I(x; \psi)$ denote the solutions for price and carryover given x . We are not parameterizing these functions; h and I here are just used to express the current equilibrium relation between x and the resulting price and carryover if ψ is the conditional expectation function; note that the function ψ is the argument, not $\psi(x)$ or any other instance of ψ . Whenever we need to know $h(x; \psi)$ or $I(x; \psi)$ at some particular x , we solve the system (17.4.11). The presence of a maximum does not make (17.4.11) difficult to solve. The trick is to first check if $I = 0$ and $p = P(x)$ is an equilibrium; if not, the problem has an interior solution that can be computed quickly if ψ and D are smooth.

Using these facts, we arrive at the equation

$$\psi(y) = \beta E\{h(y + \theta; \psi)\}. \tag{17.4.12}$$

Equation (17.4.12) expresses the stationary equilibrium condition for the conditional expectation function ψ . Time iteration is the sequence defined by

$$\psi^{j+1}(y) = E\{h(y + \theta; \psi^j)\}. \tag{17.4.13}$$

The Wright-Williams procedure is now clear. One first picks a collection of values for I and θ , a polynomial parameterization scheme for ψ , and an initial (increasing)

guess ψ^0 . Suppose that we have the j guess, ψ^j . For each (I, θ) pair, we compute $h(I + \theta; \psi^j)$. This generates data that are used in a fitting step to compute the new approximation $\psi^{j+1}(I)$, which is the expectation of $h(I + \theta; \psi^j)$ given I . This continues until the difference between the ψ^j and ψ^{j+1} functions is small.

We mentioned that ψ should be a nondecreasing function. One could ensure this by using shape-preserving approximation schemes for ψ . More commonly, ψ is parameterized using a low-order polynomial, and the number of θ values used exceeds the polynomial degree so that the fitting step is a regression. This will often avoid shape problems.

17.5 A Simple Stochastic Dynamic Growth Model

The commodity storage model has a single, endogenous state variable. We next examine a simple model with one exogenous and one endogenous state variable. We use a simple stochastic growth model to present several computational methods for solving these more general rational expectations problems. This stochastic growth model is also one that has been studied in the computational literature.

We consider the stochastic optimization problem

$$\max_{c_t} E \left\{ \sum_{t=0}^{\infty} \beta^t u(c_t) \right\} \quad (17.5.1)$$

$$\text{s.t. } k_{t+1} = F(k_t, \theta_t) - c_t,$$

$$\ln \theta_{t+1} = \rho \ln \theta_t + \varepsilon_{t+1},$$

where θ_t is a productivity parameter and $\varepsilon_t \sim N(0, \sigma^2)$ is a series of i.i.d. innovations. The solution is characterized by

$$u'(c_t) = \beta E\{u'(c_{t+1})F_k(k_{t+1}, \theta_{t+1})|k_t, \theta_t\}. \quad (17.5.2)$$

In this problem both the beginning-of-period capital stock and the current value of θ are needed for a sufficient description of the state. Hence $c_t = C(k_t, \theta_t)$ for some policy function C , and the Euler equation implies that

$$0 = u'(C(k, \theta)) - \beta E\{u'(C(F(k, \theta) - C(k, \theta), \theta^+)) \\ \times F_k(F(k, \theta) - C(k, \theta), \theta^+)|\theta\}, \quad (17.5.3)$$

where θ is the current productivity level and θ^+ is the random variable denoting the next period's productivity level. Even though we begin with an optimization problem,

the solution is also the unique competitive equilibrium where (17.5.3) characterizes the equilibrium decision rules. Our presentation is of general interest because many equilibrium analyses, with and without distortions, often reduce to equations similar to (17.5.3).

There are a variety of ways to express equilibrium. One way is to focus on the time series (c_t, k_t) of consumption and capital. This approach attempts to find a time series that satisfies

$$u'(c_t) = \beta E\{u'(c_{t+1})F_k(k_{t+1}, \theta_{t+1})\},$$

$$k_{t+1} = F(k_t, \theta_t) - c_t.$$

This is the stochastic version of the Arrow-Debreu approach, wherein we compute a sequence of state-contingent factor prices and capital stock levels that satisfies the equilibrium conditions. While this approach can be used it quickly becomes intractable as we add both time and uncertainty, since the number of state-contingent commodities grows rapidly.

We instead must take a recursive approach focusing on (17.5.3). The functional equation (17.5.3) can be written in several ways. Note that the consumption policy function C appears in four places. If we define

$$\begin{aligned} \mathcal{F}(C_1, C_2, C_3, C_4) &= u'(C_1(k, \theta)) - \beta E\{u'(C_2(F(k, \theta) - C_3(k, \theta)), \theta^+) \\ &\quad \times F_k(F(k, \theta) - C_4(k, \theta), \theta^+)|\theta\}, \end{aligned}$$

then equilibrium is defined by the functional equation $0 = \mathcal{F}(C, C, C, C) \equiv \mathcal{N}(C)$. In this way, we can deal with several methods in a unified fashion.

17.6 Projection Methods with Newton Iteration

In this section we review the details of applying alternative versions of the projection method to (17.5.3); this is largely taken from Judd (1992). In this section we will use the Powell hybrid method to find the unknown coefficients. This approach to finding the unknown coefficients is totally lacking in economic motivation but takes advantage of the quadratic (or, at least, superlinear) local convergence properties of Newton-type methods. We examine a variety of choices for the other pieces of the projection method.

We could use (17.5.3) as our residual function. However, it could be too nonlinear, particular for highly concave utility functions. Our algorithm might do better if we make it more like a linear problem. To that end we rewrite (17.5.3) it as

$$\begin{aligned} 0 &= C(k, \theta) - (u')^{-1}(\beta E \{u'(C(k^+, \theta^+)) F_k(k^+, \theta^+) | \theta\}) \\ &\equiv \mathcal{N}(C), \end{aligned} \tag{17.6.1}$$

where

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

$$\ln \theta^+ \sim N(\rho \ln \theta, \sigma^2).$$

Note that the RHS of (17.6.1) has two terms, one linear in $C(k, \theta)$ and the other is similar to a CRTS function of next period's potential consumption values. Note also that the $(u')^{-1}(\cdot)$ operation in (17.6.1) will unwrap some of the nonlinearity arising from the u' operation inside the expectation, hopefully leaving us with a more linear problem.

Galerkin and Orthogonal Collocation Methods

The procedure is similar to the deterministic case, but there are some extra twists due to the stochastic shocks. First of all our approximation of the policy function is now given by the double sum

$$\hat{C}(k, \theta; a) = \sum_{i=1}^{n_k} \sum_{j=1}^{n_\theta} a_{ij} \psi_{ij}(k, \theta), \tag{17.6.2}$$

where $\psi_{ij}(k, \theta) \equiv T_{i-1}(2(k - k_m)/(k_M - k_m) - 1) T_{j-1}(2(\theta - \theta_m)/(\theta_M - \theta_m) - 1)$. The form in (17.6.2) is a tensor-product approach to two-dimensional approximation. It is reasonable here because the dimension is low, and it is a straightforward extension of the one-dimensional techniques to higher dimensions.

Projection methods generally assume that the problem lives on a compact domain. The stochastic growth problem we specified in (17.5.1) often has an unbounded ergodic set. In practice, we specify a compact set of (k, θ) pairs over which we try to solve (17.6.1). Our choice for this compact set is represented by four parameters in (17.6.2): k_M , k_m , θ_M , and θ_m . The capital stock bounds k_m and k_M are chosen so that k is usually confined to $[k_m, k_M]$, and similarly for θ_M and θ_m . Since θ is exogenous, this can be easily accomplished for θ . If we truncate the innovation ε so that $\varepsilon \in [\varepsilon_m, \varepsilon_M]$, then θ is confined to $[\theta_m, \theta_M]$ where $\ln \theta_m = \varepsilon_m(1 - \rho)^{-1}$ and $\ln \theta_M = \varepsilon_M(1 - \rho)^{-1}$. Appropriate choices for k_m and k_M are more problematic and generally cannot be made until after some experimentation finds a capture region. In this problem we do know the steady state for the deterministic case of $\theta = 1$ and $\varepsilon \equiv 0$; one suspects that this deterministic steady state is roughly in the middle of the range

of frequently visited capital stocks, implying that k_m should be less than the deterministic steady state and k_M greater. We initially aim to find a $\hat{C}(k, \theta; a)$ which nearly satisfies (17.6.2) on the rectangle $[k_m, k_M] \times [\theta_m, \theta_M]$.

Since tomorrow's log productivity level, $\ln \theta^+$, conditional on today's log productivity level, $\ln \theta$, is distributed as $\rho \ln \theta + \sigma Z$ for $Z \sim N(0, 1)$, combining (17.6.1) and (17.6.2) implies that

$$\begin{aligned} 0 &= \hat{C}(k, \theta; a) - (u')^{-1} \left(\beta \int_{-\infty}^{\infty} I(k, \theta, z; a) \frac{e^{-z^2/2}}{\sqrt{2}} dz \right) \\ &= \mathcal{N}(\hat{C}(\cdot; a))(k, \theta), \end{aligned} \quad (17.6.3)$$

where

$$\begin{aligned} I(k, \theta, z; a) &\equiv u'(\hat{C}(F(k, \theta) - \hat{C}(k, \theta; a), e^{\sigma z} \theta^\rho; a)) \\ &\quad \times F_k(F(k, \theta) - \hat{C}(k, \theta; a), e^{\sigma z} \theta^\rho) \pi^{-1/2}. \end{aligned}$$

In (17.6.3), \mathcal{N} involves an integral that cannot generally be evaluated explicitly. In forming the residual function, we need to use an approximation $\hat{\mathcal{N}}$ of \mathcal{N} , formed here by replacing the integral in (17.6.3), with a finite sum

$$\int_{-\infty}^{\infty} I(k, \theta, z; a) \frac{e^{-z^2/2}}{\sqrt{2}} dz \doteq \sum_{j=1}^{m_z} I(k, \theta, \sqrt{2}z_j; a) \omega_j,$$

where ω_j , z_j are Gauss-Hermite quadrature weights and nodes.

We can now define the residual function

$$\begin{aligned} R(k, \theta; a) &= \hat{C}(k, \theta; a) - (u')^{-1} \left(\beta \sum_{j=1}^{m_z} I(k, \theta, \sqrt{2}z_j; a) w_j \right) \\ &\equiv \hat{\mathcal{N}}(\hat{C})(k, \theta). \end{aligned} \quad (17.6.4)$$

With this residual function, we can proceed with standard projection methods. A collocation method starts by choosing m_k capital stocks, $\{k_i\}_{i=1}^{m_k}$, and m_θ productivity levels, $\{\theta_i\}_{j=1}^{m_\theta}$, and then finds some a so that $R(k_i, \theta_j; a) = 0$ for all $i = 1, \dots, m_k$ and $j = 1, \dots, m_\theta$.

A Galerkin approach first forms the $n_k n_\theta$ projections

$$P_{ij}(a) \equiv \int_{k_m}^{k_M} \int_{\theta_m}^{\theta_M} R(k, \theta; a) \psi_{ij}(k, \theta) W(k, \theta) d\theta dk, \quad i = 1, \dots, n_k, j = 1, \dots, n_\theta, \quad (17.6.5)$$

where $W(k, \theta)$ is the product Chebyshev weight adapted for the rectangle $[k_m, k_M] \times [\theta_m, \theta_M]$, and then chooses a so that $P_{ij}(a) = 0$ for all i and j . Note that at this point each $P_{ij}(a)$ is a function only of a , since (17.6.5) integrates the integrands over the k and θ dimensions. Again $P_{ij}(a)$ needs to be computed numerically. Since the ψ_{ij} are Chebyshev polynomials, we use Gauss-Chebyshev quadrature points. Specifically, if we use m_k values of k and m_θ values of θ to compute the projections in (17.6.5), we solve the system

$$\hat{P}_{ij}(a) \equiv \sum_{l_k=1}^{m_k} \sum_{l_\theta=1}^{m_\theta} R(k_{l_k}, \theta_{l_\theta}; a) \psi_{ij}(k_{l_k}, \theta_{l_\theta}) = 0, \quad i = 1, \dots, n_k, j = 1, \dots, n_\theta, \quad (17.6.6)$$

where

$$\begin{aligned} k_{l_k} &= k_m + \frac{1}{2}(k_M - k_m)(z_{l_k}^{m_k} + 1), & l_k &= 1, \dots, m_k, \\ \theta_{l_\theta} &= \theta_m + \frac{1}{2}(\theta_M - \theta_m)(z_{l_\theta}^{m_\theta} + 1), & l_\theta &= 1, \dots, m_\theta, \\ z_l^n &\equiv \cos\left(\frac{(2l-1)\pi}{2n}\right), & l &= 1, \dots, n. \end{aligned}$$

The nonlinear system of equations (17.6.6) imposes $n_k n_\theta$ conditions that hopefully determine the $n_k n_\theta$ unknown coefficients in a , which then fixes our approximation $\hat{C}(k, \theta)$.

Accuracy Checks—A Bounded Rationality Measure

Once we have a candidate solution, we want to check its quality. A direct procedure is to check how much, if at all, $\hat{\mathcal{N}}(\hat{C})$ differs from the zero function. First we should understand what a deviation from zero means in economic terms. Consider (17.6.4). It is the difference between consumption at a capital stock k and productivity level θ , and what that consumption would be if an optimizing agent knew that tomorrow he will use the consumption rule \hat{C} , and that personal and aggregate wealth will both be $F(k, \theta) - C(k, \theta)$. Therefore our residual function (17.6.4) applied to the approximate solution is the one-period optimization error in consumption terms. The function

$$E(k, \theta; a) \equiv \frac{R(k, \theta; a)}{\hat{C}(k, \theta; a)} \quad (17.6.7)$$

yields a dimension-free quantity expressing that optimization error as a fraction of current consumption. We use $E(k, \theta; a)$ for our accuracy index.

This approach to accuracy checking expresses the resulting errors in economic terms, essentially in terms of how irrational agents would be in using the approximate rule. If this relative optimization error were found to be about 0.1, then we would know that the approximation implies that agents make 10 percent errors in their period-to-period consumption decisions, a magnitude that few economists would find acceptable. However, if this index were found to be 10^{-6} , then the approximation implies that agents made only a \$1.00 mistake for every \$1,000,000 they spent. While such an approximation, \hat{C} , may not be the mathematically exact equilibrium decision rule, it is hard to argue that real world agents would actually do better. Therefore \hat{C} is as plausible a description of human behavior as the mathematically exact zero of the operator \mathcal{N} .

The philosophy behind this accuracy check is that we should find an ε such that our approximation is an ε -equilibrium. The advantage of this approach is that our attempt to approximate an exact equilibrium becomes reinterpreted as a search for an approximate equilibrium. The disadvantage of focusing on ε -equilibrium is the likely existence of an open set of such equilibria. However, as long as the problem is well-conditioned, something that can be numerically checked, that set is likely to be small, and even negligible for many purposes.

Initial Guess

The initial guess is always important in any nonlinear equation procedure, particularly for methods that may not converge. Our initial guess is the value for a such that $\hat{C}(k, \theta; a) = k(F(1, 1) - 1)$. This initial guess is the linear consumption function going between the origin and the deterministic steady state and does not depend on θ . Since \hat{C} is a polynomial, there is such an a , but the exact value of a will depend on k_m and k_M . Its advantage is that it is a simple one; we do not do any problem-specific manipulations to construct a good initial guess. One important property is that it is a rule that is consistent with global stability and the transversality condition at infinity for the deterministic problem.

Finding an initial guess is often the most difficult part of solving nonlinear systems when one is not using a homotopy method. It is good to have a variety of strategies for generating initial guesses. In solving (17.6.1), an alternative procedure would be to use the perturbation methods discussed in chapter 14 to construct the locally valid linear approximation to $C(k, \theta)$ near the deterministic steady state. Another possibility is to first use the least squares approach to find a low-order solution and use it as the initial guess for the projection system (17.6.6). The global convergence properties of least squares algorithms would keep this step from diverging, and the result would hopefully be a good initial guess for (17.6.6). Some generate an initial guess by

using the simple continuation method of section 5.8 beginning with a version of (17.6.1) with a known closed-form solution. This will probably work, but it will be slow, and at that point one might as well use a full homotopy approach. Fortunately the simple linear guess worked fine for solving (17.6.6), and the Powell hybrid solver always converged for the examples below.

Results for Galerkin and Orthogonal Collocation Methods

With our approximation method and accuracy checks determined, we can now see how good our approximate solutions are and how fast we can compute them. To do so, we make taste and technology specifications that bracket a wide range of empirically plausible values. Table 17.1 summarizes typical cases. We again have assumed that $u(c) = c^{\gamma+1}/(\gamma + 1)$, and $F(k) = k + Ak^\alpha$ where A is chosen so that the deter-

Table 17.1
Log₁₀ Euler equation errors

γ	ρ	σ	$\ E\ _\infty$	$\ E\ _1$	$\ E_I\ _\infty$	$\ E\ _\infty$	$\ E\ _1$	$\ E_I\ _\infty$
(2, 2, 2, 2) ^a							(4, 3, 4, 3) ^a	
-15.00	0.80	0.01	-2.13	-2.80	-2.58	-3.00	-3.83	-3.70
-15.00	0.80	0.04	-1.89	-2.54	-2.28	-2.44	-2.87	-2.59
-15.00	0.30	0.04	-2.13	-2.80	-2.58	-2.97	-3.83	-3.70
-0.10	0.80	0.01	-0.01	-1.22	-1.34	-1.68	-2.65	-2.70
-0.10	0.80	0.04	-0.01	-1.19	-1.20	-1.48	-2.22	-1.89
-0.10	0.30	0.04	0.18	-1.22	-1.35	-1.63	-2.65	-2.74
(7, 5, 7, 5) ^a							(7, 5, 20, 12) ^a	
-15.00	0.80	0.01	-4.28	-5.19	-5.00	-4.43	-5.18	-4.91
-15.00	0.80	0.04	-3.36	-4.00	-3.70	-3.30	-3.95	-3.67
-15.00	0.30	0.04	-4.24	-5.19	-4.96	-4.38	-5.18	-4.87
-0.10	0.80	0.01	-3.40	-4.37	-4.35	-3.47	-4.39	-4.32
-0.10	0.80	0.04	-2.50	-3.22	-2.93	-2.60	-3.17	-2.91
-0.10	0.30	0.04	-3.43	-4.37	-4.36	-3.49	-4.39	-4.33
(10, 6, 10, 6) ^a							(10, 6, 25, 15) ^a	
-15.00	0.80	0.01	-5.48	-6.43	-6.19	-5.61	-6.42	-6.11
-15.00	0.80	0.04	-3.81	-4.38	-4.11	-3.88	-4.37	-4.11
-15.00	0.30	0.04	-5.45	-6.43	-6.15	-5.57	-6.42	-6.08
-0.10	0.80	0.01	-5.09	-6.12	-5.94	-5.17	-6.15	-5.94
-0.10	0.80	0.04	-2.99	-3.68	-3.37	-3.09	-3.64	-3.38
-0.10	0.30	0.04	-5.17	-6.12	-6.01	-5.23	-6.14	-5.99

a. This four-tuple denotes the values of $(n_k, n_\theta, m_k, m_\theta)$ for the block below.

ministic steady state is $k = 1$. Throughout table 17.1, $\alpha = 0.25$, $k_m = 0.3$, and $k_M = 2.0$. We also chose $m_z = 8$, the eight-point, fifteenth-order accurate Gauss-Hermite quadrature rule to compute the conditional expectation. Table 17.1 lets σ and ρ vary; for each choice, θ_M is set equal to the long-run value of θ which would occur if $\varepsilon_t = 3\sigma$ for all t (in the notation above, this implies that $\varepsilon_m = 3\sigma$), and $\theta_m = 1/\theta_M$. It is extremely unlikely for θ to spend much if any time outside of $[\theta_m, \theta_M]$.

Table 17.1 is composed of six blocks of error entries, each block corresponding to a particular choice of the 4-tuple $(n_k, n_\theta, m_k, m_\theta)$. For example, the block headed by the 4-tuple $(2, 2, 2, 2)$ is the case where the $\hat{C}(k, \theta) = a_1 + a_2 k + a_3 \theta + a_4 k\theta$. The label $(2, 2, 2, 2)$ also indicates that we choose an orthogonal collocation procedure which fixes a so that the Euler equation fits exactly at the four zeros of $\psi_{3,3}$. The 4-tuple $(10, 6, m_k, m_\theta)$ corresponds to allowing k terms up to k^9 , θ terms up to θ^5 , and all possible pairwise products of those k and θ terms.

To test for the quality of the candidate solution, we evaluate $E(k, \theta; a)$ defined in (17.6.7) at a large number of (k, θ) combinations that themselves were not used in finding a solution. We define the relative error at (k, θ) to be $E(k, \theta; a)$. The entries are the base 10 logarithm of various norms of $E(\cdot)$. Columns 4, 5, 7, and 8 report $\log_{10}\|E\|_\infty$ and $\log_{10}\|E\|_1$. All of these norms were calculated by using 8000-grid points in $[k_m, k_M] \times [\theta_m, \theta_M]$, 100 in $[k_m, k_M]$ and 80 in $[\theta_m, \theta_M]$. Base 10 logs of these norms are natural measures for our exercise. $\log_{10}\|E\|_\infty$ is the maximum error we found, and $\log_{10}\|E\|_1$ represents the average error. For example, an entry of -3 under the $\log_{10}\|E\|_\infty$ column says that a person with \$1,000 of consumption makes at *most* a one dollar error in current consumption in each period relative to the next period's consumption. Since solution paths concentrate near the center of the state space, we are particularly concerned about accuracy there. We define E_I to be E restricted to the inner rectangle $[k'_m, k'_M] \times [\theta'_m, \theta'_M]$ where $[k'_m, k'_M] \subset [k_m, k_M]$ and $[\theta'_m, \theta'_M] \subset [\theta_m, \theta_M]$; we take the smaller k (θ) interval to be the middle half of the larger interval. Columns 6 and 9 reports $\log_{10}\|E_I\|_\infty$.

There are several points to note. First, note that the errors are rather small. Even for the $(2, 2, 2, 2)$ case, the errors are roughly one dollar per hundred, as long as the utility function is as concave as $\log c$. Second, as we allow the approximation to use more terms the errors fall until in the $(10, 6, 10, 6)$ case, we often find optimization errors of less than one dollar per million. Third, the various norms of the residual function have very similar values, indicating that the errors are uniformly small. In particular, the similarity in values for the norms of E and E_I indicates that the solution is almost as good at the edges of the state space as in the middle.

Fourth, these methods are fast. We report times on a Compaq 386/20, a dinosaur by today's standards; the content of the times we report lies in the relative speeds.

The solutions in the $(2, 2, 2, 2)$ case were solved in 0.2 to 0.4 seconds and in the $(4, 3, 4, 3)$ case in 1.1 to 2 seconds. The slow parameterization throughout table 17.1 was $\gamma = -15.0$, $\rho = 0.8$, and $\sigma = 0.04$, which took 3 seconds for the $(4, 3, 4, 3)$ case. The speed advantage of orthogonal collocation is demonstrated by the fact that the $(7, 5, 7, 5)$ cases generally took 8 to 18 seconds, whereas the $(7, 5, 20, 12)$ Galerkin cases took three times as long, which is expected since the projections were integrals using 240 points instead of 35. An intriguing exception was the slow parameterization which took nearly two minutes for collocation but only a minute and a half for Galerkin. Apparently the extra information used by the Galerkin procedure helped the nonlinear equation solver avoid bad directions. The $(10, 6, 10, 6)$ cases generally took 27 to 72 seconds with the bad parameterization taking 100 seconds. The corresponding $(10, 6, 20, 15)$ cases took roughly four times as long.

Fifth, note that the orthogonal collocation method does remarkably well given the small amount of computation. This is indicated by the small optimization errors and that the Galerkin procedures which use many more points achieved only slightly greater accuracy. In general, the collocation schemes yielded the most accuracy per unit of time.

Another way to check for accuracy is to see how the computed solution changes when we use higher-order and different quadrature schemes. Again we find trivial sensitivity to these changes. For example, using $m_z = 4$ instead of 8 resulted in very few differences (table 17.1 was unchanged) and cut the running time by almost half.

As with the deterministic case we have ignored transversality considerations in our solution method. Again we should check that the solutions are stable, as they always were. In the stochastic case we have no clear alternative that will ensure convergence to a stable solution, since we do not know a priori any point on the policy function. While solving operator equations without imposing boundary conditions is not proper procedure, it appears to be possible for these problems.

Alternative Bases and Projections

Tables 17.2 and 17.3 discuss the results when we attempt alternative implementations of the projection ideas. Each choice was motivated by some optimality or conditioning consideration. We will now see how important they were. Table 17.2 re-examines some of the cases in table 17.1 using theoretically inferior methods.

The pair of columns under G gives the $\log_{10} \|E\|_\infty$ error measure and running times when we used the projection procedures with Chebyshev polynomials and Chebyshev zeros. The pair of columns under P refers to $\log_{10} \|E\|_\infty$ and running times when we used ordinary polynomials instead of Chebyshev polynomials, but still fit the residual conditions at the Chebyshev zeros. The results under G and P should be identical

Table 17.2
Alternative implementations

γ	ρ	σ	G ^a		P ^b		U ^c		UP ^d	
$n_k = 7 \quad n_\theta = 5 \quad m_k = 7 \quad m_\theta = 5$										
-15.0	0.8	0.04	-3.18	1:15	-2.13	:40	-3.06	1:05	-2.19	:44
	0.3	0.01	-4.35	:11	-4.35	:52	-4.07	:08	-4.07	1:47
$n_k = 10 \quad n_\theta = 6 \quad m_k = 25 \quad m_\theta = 15$										
-15.0	0.8	0.04	-3.87	4:20	-3.90	24:44	-3.90	3:41	-3.36	42:15
	0.3	0.01	-5.68	2:19	-5.14	11:31	-5.49	2:14	-5.30	8:06
$n_k = 10 \quad n_\theta = 6 \quad m_k = 25 \quad m_\theta = 15$										
-0.9	0.8	0.04	-4.00	1:31	-4.00	5:17	-4.01	1:31	-4.01	5:02
	0.3	0.01	-5.40	1:23	-4.63	7:13	-5.25	1:20	-5.13	6:01
-0.1	0.8	0.04	-3.09	1:31	-3.09	9:16	-3.10	1:32	-3.07	12:01
	0.3	0.01	-5.27	1:32	-4.02	7:25	-5.09	1:27	-3.27	8:32

a. Chebyshev polynomial basis, Chebyshev zeros used in evaluating fit.

b. Ordinary polynomial basis, Chebyshev zeros used in evaluating fit.

c. Chebyshev polynomial basis, uniform grid points.

d. Ordinary polynomial basis, uniform grid points.

Table 17.3
Tensor product versus complete polynomials

γ	ρ	σ	Tensor product			Complete polynomials		
			$n = 3$	$n = 6$	$n = 10$	$n = 3$	$n = 6$	$n = 10$
-15.0	0.8	0.04	-2.34 ^a :01 ^b	-3.26 :13	-3.48 14:21	-1.89 :03	-3.10 :07	-4.06 1:09
-0.9	0.3	0.10	-2.19 :01	-3.60 :08	-5.27 1:21	-2.14 :01	-3.55 :05	-5.22 :32
-0.1	0.3	0.01	-1.00 :01	-2.84 :08	-5.21 1:24	-0.99 :01	-2.83 :05	-5.17 :35

Note: The tensor product cases in this table used orthogonal collocation with $n_k = n_\theta = m_k = m_\theta = n$ to identify the n^2 free parameters. The complete polynomial cases used Galerkin projections to identify the $1 + n + n(n + 1)/2$ free parameters.

a. $\log_{10}\|E\|_\infty$.

b. Computation time expressed in minutes:seconds.

if we had infinite precision arithmetic and Newton's method always converged. In several cases the results were the same, but the P times were far slower. P was faster in one case but yielded an approximation with substantially larger error: The solver could not make good progress so it stopped early. The slower time and premature stopping are both reflections of the conditioning problems associated with ordinary polynomials.

The columns under U give the accuracy and running time when we used Chebyshev polynomials but used a uniform grid to compute the projections. Here G should do better, since interpolation at a uniform grid is generally inferior to interpolation at Chebyshev zeros. Running times should not be affected much, since the Chebyshev approximation approach helps keep the nonlinear system well-conditioned. We see that accuracy is generally the same or worse, and running times are the same or slower.

The final column, labeled UP , refers to the straightforward approach of using ordinary polynomials and uniform grid points. Here we have a substantial loss of speed to attain the same or lower-quality approximations. The loss of speed by a factor of two to ten demonstrates clearly the value of orthogonal polynomial approximations and Gaussian quadrature. Since this is a rather simple problem, one suspects that the difference will be far greater in more complex problems.

As predicted by theory, the condition numbers of the Jacobian were strongly related to these performance indices. The cases in table 17.1 always had Jacobians with condition numbers under 10^3 and usually of the order 10. The P cases in table 17.2 had condition numbers several orders of magnitude greater, sometimes as great as 10^7 . The U cases had condition numbers between 10^2 and 10^4 , and the UP cases had condition numbers as large as those in the P cases. Our experiments also indicated that the condition number of the jacobian at our initial guess was of the same order of magnitude as the condition number at the solution. This suggests a useful procedure. If the Jacobian's condition number is large at the initial guess, then one should change the basis, fitting conditions, or something to reduce the initial conditioning of the problem, whereas a low initial condition number is evidence that the problem will be well-behaved.

Table 17.3 demonstrates the advantages of a complete polynomial basis. We report $\log_{10}\|E\|_\infty$ and, below it, the running time for a few cases. The parameter n is one more than the maximum exponent. For example, the $n = 3$ case under "tensor product" refers to using the tensor product of quadratic polynomials in k and θ , where under "complete polynomials" it refers to using the basis $\{1, k, \theta, k^2, k\theta, \theta^2\}$. The complete polynomial basis generally yields a lower-quality fit; the exception

occurs because the Newton solver had difficulty converging for the tensor product basis but not for the complete polynomial basis. However, the slightly lower accuracy for fixed n was achieved in much less time. Since the real objective is to find a method that achieves maximal accuracy for fixed time, table 17.3 shows that the complete polynomial basis generally gives most accuracy per unit time. Since this is clear for a two-dimensional problem, we can expect that there will be even larger gains from using complete bases in higher dimensions.

Value of Nonlinear Transformations

All of these examples of solutions to (17.5.3) used (17.6.1) instead of Euler equation (17.5.3). When (17.5.3) is used instead, the results are substantially inferior in terms of speed and accuracy, particularly when γ is large in magnitude. This is not surprising since we used a Newton-style method for solving (17.6.2), and the Jacobian of (17.6.5) was better conditioned when R is taken from (17.6.1) than when it is taken from (17.5.3). While this may appear lucky, the key fact is that (17.6.1) has two pieces, one linear in the unknown Chebyshev coefficients and the other “more linear” than (17.5.3). Again the initial condition number was very good at predicting performance.

17.7 Fixed-Point Iteration

We next examine a fixed-point iteration procedure similar to the fixed-point iteration examined in chapter 16. We first discuss a direct implementation of the fixed-point iteration idea and then present a simulation alternative.

Gaussian Quadrature Implementation

In fixed-point iteration we execute the iteration

$$\begin{aligned} C_{i+1}(k, \theta) &= (u')^{-1}(\beta E\{u'(C_i(k^+, \theta^+)) F_k(k^+, \theta^+) | \theta\}), \\ k^+ &\equiv F(k, \theta) - C_i(k, \theta), \\ \theta^+ &\sim N(\rho \ln \theta, \sigma^2). \end{aligned} \tag{17.7.1}$$

Note that this is a simple rewriting of (17.6.1). The key computational task is to compute the right hand side for several (k, θ) choices and interpolate to get successive iterates of C . Essentially the RHS of (17.7.1) is tomorrow’s return on saving one more dollar today conditional on today’s (k, θ) and assuming that one follows the rule $c = C(k, \theta)$ at all times.

In this model the RHS of (17.7.1) is just a simple integral over θ^+ , tomorrow's productivity level conditional on the current k and θ . Since all the elements are smooth and the disturbances Gaussian, the four- or five-point Gauss-Hermite rule is adequate. Therefore, to compute the, say, quadratic, solution to this problem, one need only examine a handful of k and θ values. This results in a very rapid way to compute an iterate. This method may converge slowly if at all, just as was the case with fixed-point iteration in the perfect foresight case in chapter 16. However, convergence problems may be solved by the extrapolation or dampening adjustments.

Simulation Implementation

A natural approach to solving (17.7.1) is to use numerical quadrature methods to evaluate the conditional expectation on the RHS of (17.7.1); that is, for each θ , use quadrature methods to compute the integral which corresponds to the conditional expectation. In this model it is easy to compute the conditional expectation, but this direct approach may not be available in general.

den Haan and Marcet (1990) use the fixed-point iteration method in (17.7.1), but they use simulation and regression methods instead of deterministic integration methods to approximate the key conditional expectation in (17.7.1). They first define the conditional expectation function

$$\psi(k_t, \theta_t) = E\{u'(c_{t+1})F_k(k_{t+1}, \theta_{t+1})|\theta_t, k_t\}$$

and parameterize ψ with a functional form $\hat{\psi}(k, \theta; a)$.⁵ They next simulate a θ_t path to use in each subsequent iteration.

Their iteration procedure begins with a guess for the a coefficients in $\hat{\psi}(k, \theta; a)$. From this guess, they compute the implied consumption rule $c = \hat{C}(k, \theta; a) = (u')^{-1}(\beta\hat{\psi}(k, \theta; a))$. They then simulate the stochastic system

$$k_{t+1} = F(k_t, \theta_t) - \hat{C}(k_t, \theta_t; a)$$

for T periods to generate a time series $(c_t, k_t)_{t=1}^T$. They then choose the next value of a to solve

$$\min_a \sum_{t=1}^T (u'(c_{t+1})F_k(k_{t+1}, \theta_{t+1}) - \hat{\psi}(k_t, \theta_t; a))^2,$$

5. den Haan and Marcet express log consumption as a polynomial in $\log k$ and $\log \theta$ to keep consumption positive, but that is not critical for the basic idea.

a nonlinear regression of $u'(c_{t+1})F_k(k_{t+1}, \theta_{t+1})$ on the $(\theta_t, k_t)_{t=1}^T$ data. The den Haan-Marcet (dHM) procedure continues until the α iterates converge.⁶

The dHM procedure is a fixed-point iteration using simulation and regression to compute the critical conditional expectation. Since it is a fixed-point iteration, extrapolation and dampening can be used to address convergence problems. The use of simulation is intuitively natural and related to rational expectations learning ideas. However, the dHM method suffers from sensitivity to random simulation error. Since the critical integrals in this problem are of low-dimension, it is doubtful that there is any advantage to using simulation methods over deterministic quadrature methods. Comparisons of the running times reported in den Haan and Marcet with the times reported in tables 17.2 and 17.3 show that conventional projection methods are far faster than the dHM procedure for the stochastic growth problem 17.5.1.

17.8 Time Iteration

The time iteration method revolves around a dynamic interpretation of the Euler equation (17.5.2). Suppose that $C_T(k, \theta)$ is the time T policy function. To compute $C_{T-1}(k, \theta)$, we solve (17.5.1) for each (k, θ) pair. More precisely, for each (k, θ) we find the c that solves

$$0 = u'(c) - \beta E\{u'(C_T(F(k, \theta) - c, \theta^+))F_k(F(k, \theta) - c, \theta^+) | \theta\}. \quad (17.8.1)$$

The solution to (17.8.1) is the consumption that equates current marginal utility of consumption with the expected marginal utility of consumption in the next period where the consumption rule is C_T . With C_{T-1} so determined at each (k, θ) , we then iterate and compute C_{T-2}, C_{T-3} , and so on. When the iterates are close, we quit.

Pure time iteration is not possible for a computer since we cannot solve (17.8.1) at each (k, θ) . We must therefore use some approximation scheme for C . Coleman (1990) used bilinear finite elements to approximate C . Specifically, he divided $(\log k, \log \theta)$ space into equi-sized rectangles and on each rectangle approximated the policy

6. Den Haan and Marcet refer to their method as “parameterized expectations.” This is a misleadingly broad term, since parameterizing expectations goes back to parametric methods in dynamic programming (Bellman 1962 and Daniel 1978), and the use of polynomial parameterizations of conditional expectations functions in Euler equations was introduced previously by Wright and Williams (1984). Also the parameterization aspects of the dHM algorithm are independent of the simulation aspects. Therefore the term “parameterized expectations” does not focus on the key novel and distinguishing feature of the method in den Haan and Marcet (1990), that being the use of a simulated time series to compute the key integrals instead of the conventional quadrature methods used by earlier writers. To avoid confusion, we refer to their method as the dHM method.

function with a linear combination of 1, $\log k$, $\log \theta$, and $(\log k)(\log \theta)$. The basis is the tensor product of tent functions in $(\log k, \log \theta)$ space.

Suppose that we have an approximation $\hat{C}_T(k, \theta)$. To compute $\hat{C}_{T-1}(k, \theta)$, we first solve (17.5.1) for each $(\log k, \log \theta)$ pair in the grid. That is, for each $(\log k, \log \theta)$ in the grid, we find the c which solves (17.8.1) and set $\hat{C}_{T-1}(k, \theta) = c$. With \hat{C}_{T-1} so determined at each $(\log k, \log \theta)$ on the grid, Coleman defines \hat{C}_{T-1} to be the bilinear interpolant of the data on the grid.

A drawback of the Coleman procedure is the slow convergence of the integration method. Coleman used Gauss-Hermite quadrature that is appropriate given the log normal density for θ . However, because \hat{C}_T is only C^0 , the rate of convergence is not greater than a Newton-Cotes rule. Therefore Coleman used a large number of points in the quadrature rule.

Such low-order approximations necessitate many grid points; Coleman used a grid of 50 capital stocks and 20 productivity levels for a total of 1000 free parameters. Such a large number of free parameters results in a time-consuming computation, much slower than the projection procedures using orthogonal polynomial approximations. One can use smooth approximation schemes for \hat{C} instead of the piecewise linear log procedure used by Coleman. However, Coleman's finite-element procedure is less likely to have shape problems than more aggressive schemes such as collocation with orthogonal polynomials.

17.9 Generalizations

Our examples have been relatively simple, involving only single goods and single agents, and were cases where equilibrium was equivalent to a known social planning problem. The techniques presented above are not limited to such simple models. In this section we present some examples that indicate how to apply these methods to more substantive problems.

Growth with Tax Distortions

We first show how to add a tax distortion to the simple stochastic growth model presented in section 17.5. First, since the rate of return on capital net of depreciation is $f_k \equiv F_k - 1$, if it is taxed at a state-contingent rate $\tau(k, \theta)$, then equilibrium is the solution to the Euler equation

$$\begin{aligned} u'(C(k, \theta)) &\doteq \beta E\{u'(C(k^+, \theta^+))(1 + (1 - \tau(k^+, \theta^+))f_k(k^+, \theta^+))|\theta\}, \\ k^+ &\equiv F(k, \theta) - C(k, \theta). \end{aligned} \tag{17.9.1}$$

Note that (17.9.1) is a simple modification of (17.5.3). This is not the solution to any social planner's problem, but the key mathematical structure of this problem is similar to the first-order conditions in (17.5.3).

In tax problems we often need to compute a revenue function. Let $R(k, \theta)$ be the discounted value of current and future government tax revenue if the current state is (k, θ) and all revenues are lump-sum rebated. $R(k, \theta)$ is the solution to the integral equation

$$u'(C(k, \theta))(R(k, \theta) - \tau(k, \theta)kf_k(k, \theta)) = \beta E \{ u'(C(k^+, \theta^+)) R(k^+, \theta^+) | \theta \}. \quad (17.9.2)$$

Note that (17.9.2) is linear in $R(k, \theta)$ once we have the solution to $C(k, \theta)$. Therefore we can apply linear integral methods to solve (17.9.2) for $R(k, \theta)$.

We also will want to compute the equilibrium value function. Let $W(k, \theta)$ be the discounted value of current and future utility if the current state is (k, θ) . $W(k, \theta)$ is the solution to the integral equation

$$W(k, \theta) = u(C(k, \theta)) + \beta E \{ W(k^+, \theta^+) | \theta \}. \quad (17.9.3)$$

As with (17.9.2), (17.9.3) is linear in $W(k, \theta)$ once we have the solution to $C(k, \theta)$, and we can apply linear integral equation methods to solve (17.9.3) for $W(k, \theta)$.

Heterogeneous Agent Models

We next present an example with heterogeneous agents taken from Gaspar and Judd (1997). Suppose we take the model of section 17.5 but assume two infinitely lived types of agents with different utility functions; let $u_i(c_i)$ be the utility function of type i agents. Suppose that the only available asset is the risky capital stock. Then the state variable is $(k, \theta) \equiv (k_1, k_2, \theta)$ where k_i is the capital stock held by type i agents, $i = 1, 2$. The equilibrium consumption rule for type i agents is $C_i(k, \theta)$ and the consumption policy functions are characterized by the pair of Euler equations

$$\begin{aligned} 0 &= u'_i(C_i(k, \theta)) - \beta E \{ u'_i(C_i(k^+, \theta^+)) F_1(k_1^+ + k_2^+, \theta^+) | \theta \}, \quad i = 1, 2, \\ k_i^+ &\equiv k_i F_1(k_1 + k_2, \theta) + \frac{1}{2}(F(k_1 + k_2, \theta) - (k_1 + k_2)F_1(k_1 + k_2, \theta)) - C_i(k, \theta), \end{aligned} \quad (17.9.4)$$

where $k_i F_1$ is type i 's asset income and $\frac{1}{2}(F - (k_1 + k_2)F_1)$ is a type i 's share of labor income.

The system (17.9.4) consists of two Euler equations with two unknown two-dimensional functions. We can use the methods used to solve (17.5.3). In (17.9.4) each C_i appears on the LHS alone, just as in (17.5.3); therefore we can use time iteration or function iteration just as easily here as before. For example, function

iteration would implement the iteration

$$C_i^{j+1}(k, \theta) = (u'_i)^{-1}(\beta E\{u'_i(C_i^j(k^+, \theta^+)F_k(k_1^+ + k_2^+, \theta^+)|\theta)\}), \quad i = 1, 2. \quad (17.9.5)$$

One could also construct a system of projection conditions equal in number to the number of unknown coefficients and use Newton's method to solve for the unknown coefficients. Judd, Kubler, and Schmedders (1997a, b) examine some two-agent models. Gaspar and Judd (1997) evaluates alternative methods for two-, three-, four-, and five-agent problems. Furthermore Gaspar and Judd (1997) present a model where different agents have stochastic lifetimes and where different agents face different tax rates on asset income. The addition of multiple agents to the analysis creates larger systems but do not present fundamentally different challenges. However, as the number of unknown functions increases, the relative rankings of various methods change; Gaspar and Judd provide some examples of this.

Elastic Labor Supply

Many models have multiple choice variables, leading to multiple Euler equations. We next illustrate how to use our Euler equation methods to where we have both consumption and labor supply decisions. The representative agent model with elastic labor supply solves the problem

$$\begin{aligned} & \max_{c_t, l_t} E \left\{ \sum \beta^t u(c_t, l_t) \right\} \\ \text{s.t. } & k_{t+1} = k_t + \theta_t f(k_t, l_t) - c_t, \\ & \ln \theta_{t+1} = \rho \ln \theta_t + \varepsilon_{t+1}. \end{aligned} \quad (17.9.6)$$

In this model we assume that θ_t is known when c_t and l_t are chosen. This implies that a stationary solution for c and l choices will depend on k and θ , and they can be denoted $c = C(k, \theta)$ and $l = L(k, \theta)$. There are two Euler equations, one for each choice variable; they are

$$\begin{aligned} u_c(c, l) &= \beta E\{u_c(c^+, l^+)(1 + \theta^+ f_k(k^+, l^+))\}, \\ u_c(c, l) \theta f_l(k, l) &= -u_l(c, l), \end{aligned} \quad (17.9.7)$$

where

$$c \equiv C(k, \theta), \quad l \equiv L(k, \theta),$$

$$k^+ \equiv k + \theta f(k, l) - c, \quad c^+ \equiv C(k^+, \theta^+), \quad l^+ \equiv L(k^+, \theta^+).$$

To solve (17.9.6), we just apply our methods to the Euler equation pair in (17.9.7) to compute the policy functions $C(k, \theta)$ and $L(k, \theta)$. In general, problems with several choice variables are solved by specifying policy functions and a system of Euler equations for these variables, parameterizing the policy functions, and applying some iterative scheme to construct a sequence of policy function approximations that appear to converge to policy functions. After convergence, one should examine the apparent solutions fit with the Euler equation system at states not used in the computation.

17.10 Further Reading and Summary

This chapter has presented an overview of the problems in solving rational expectations models of various types. The examples presented illustrate how one can combine basic numerical analysis tools from approximation theory, nonlinear equations, and numerical quadrature to solve rational expectations problems. The Lucas asset pricing model reduces to a linear Fredholm equation and simple monetary models reduce to nonlinear Fredholm equations. The commodity market model is a simple example of rational expectations modeling with an endogenous state, as is the stochastic growth model. We again see the use of various iteration schemes, time iteration and other fixed-point iteration approximation schemes being the most common but with a Newton approach often dominating. Better examples of the value of these methods is the analysis in Wright and Williams (1984) which adds endogenous supply considerations to the model described in (17.4.1), and the several applications in Williams and Wright (1991). The stochastic growth model has been used to evaluate various methods; Danthine et al. (1989) and the Taylor and Uhlig (1990) symposium reviews the issues in rational expectations computations.

Bernardo and Judd (1994, 1997a, b) contain several applications of the approach to asset market equilibrium presented in section 17.3. Ausubel (1990a, b) presents a numerical analysis of markets with asymmetric information. His approach is somewhat more problem specific but reduces to a system of differential equations, a more conventional numerical problem. Corb used projection methods to solve a nonlinear multidimensional extension of the Kyle model. Ashcroft (1995) uses special bases and projection methods to solve option pricing problems.

We gave in this chapter only the barest introduction to heterogeneous agent analysis. Gaspar and Judd (1997) present some simple examples. den Haan (1995) analyzes models with a nontrivial income distribution. We have ignored dynamic games; Judd (1990) and Rui and Miranda (1996) discuss projection solution methods for solving dynamic games.

Table 17.4
Projection method menu

Approximation	Integration	Projections	Equation solver
Piecewise linear	Newton-Cotes	Galerkin	Newton
Polynomials	Gaussian rules	Collocation	Powell
Splines	Monte Carlo	Method of moments	Fixed-point iteration
Neural networks	Quasi-Monte Carlo	Subdomain	Time iteration
Rational functions	Monomial rules	Least squares	Homotopy
Hybrid constructions	Interpolatory rules Asymptotics		

This chapter has also presented just a small sampling of rational expectations models we can solve. The key idea is that there are four basic decisions to make: how to parameterize the unknown functions, how to compute integrals, what projection criterion to use in fixing the unknown parameters, and how to solve the projection conditions. Table 17.4 lists several possibilities for each choice. Since these decisions are mostly independent, it is clear that there are many possible combinations. While some combinations are unlikely to be useful, many combinations are competitive with the best depending on the problem. Only a few combinations have been explored in the literature. The important lesson is that an analyst should be flexible, and be able to try several alternatives.

Exercises

1. Use both the simulation method and the linear Fredholm integral equation methods to solve the Lucas asset pricing model (17.1.1)–(17.1.2). Specifically, assume $u(c) = c^{1+\gamma}/(1+\gamma)$ and $\varepsilon_t \sim N(0, \sigma^2)$ with $\gamma \in [-0.5, -5]$ and $\rho \in [0.01, 0.99]$.
2. Solve (17.2.1) with $u(c, m) = -c^{-2} + m - m^2/2$, $\theta_t = 1.0$, $\beta = 0.95$, and w_t i.i.d. distributed $U[1, 2]$. Recompute for the cases of $\theta = 1.1$ and $\theta = 0.9$. Which value of θ produced the greatest expected utility at the mean wage level?
3. Solve (17.4.1) with $\theta \sim U[0, 3]$, $P(q) = 5 - q$, and $g(I) = \sqrt{I}$ using fixed-point iteration and time iteration.
4. Resolve (17.4.1) with $\log \theta_{t+1} = \rho \log \theta_t + \varepsilon_t$, $\varepsilon_t \sim N[0, 1]$, and $\rho = 0.5$.
5. Solve (17.9.6) with $u(c, l) = c^{1+\gamma}/(1+\gamma) - l^\nu$, $\gamma \in \{-.5, -1.1, -3\}$, $\nu \in \{1.1, 2, 5\}$ and Cobb-Douglas $f(k, l)$ with capital share equal to 0.33.
6. Redo exercise 5 except add a labor income tax rate τ_L with lump-sum rebate of revenues. Compute the revenue function and the function relating expected utility to the state of the economy.

7. Redo exercise 6 with an autoregressive labor income tax rate.
8. Solve (17.5.3) using time iteration, fixed-point iteration with quadrature, the dHM procedure, and Newton iteration methods. Use ordinary and Chebyshev polynomials and in both levels and logs. Use the initial guess $C(k, \theta; a) = k(F(1, 1) - 1)$. Use the specifications for utility, production, and β used in table 17.1. Which bases produce the approximations with the smallest Euler equation errors? Which coefficient solution method does best per unit time?
9. Repeat exercise 8 for a variety of cases with attention to initial guesses. First take your solution from exercise 8, randomly perturb each coefficient by 1 percent, and use that for the initial guess for each method. Which methods do better for this initial condition? Repeat this several times. Do this again with random 5 percent and 10 percent errors.
10. Repeat exercise 8 but now first use the Newton iteration method with collocation, a Chebyshev polynomial basis in logs, and initial guess $C(k, \theta; a) = k(F(1, 1) - 1)$. Then take that answer and use it as the initial guess for the fixed-point iteration methods, both the quadrature method and the dHM method. Compare the running time of the Newton method with initial guess $C(k, \theta; a) = k(F(1, 1) - 1)$ and the running times of the fixed-point iteration methods using the Newton solution as the initial guess. In the case of the dHM method, repeat several times using different simulated innovation samples, and compute the variance of the predicted consumption at $k = 0.8$ and $\theta = 1$.

References

- Aarts, E., and J. Korst. 1990. *Simulated Annealing and Boltzmann Machines*. New York: Wiley.
- Acton, F. 1996. *Real Computing Made Real: Preventing Errors in Scientific and Engineering Calculations*. Princeton: Princeton University Press.
- Adomian, G. 1986. *Nonlinear Stochastic Operator Equations*. Orlando: Academic Press.
- Aiyagari, S. R., Z. Eckstein, and M. Eichenbaum. 1989. Inventories and price fluctuations under perfect competition and monopoly. In T. Kollintzas, ed., *The Rational Expectations Equilibrium Inventory Model: Theory and Applications*. Lecture Notes in Economics and Mathematical Systems 322. New York: Springer, pp. 34–68.
- Akin, J. E. 1982. *Application and Implementation of Finite Element Methods*. Orlando: Academic Press.
- Albrecht, J. W., B. Holmlund, and H. Lang. 1991. Comparative statics in dynamic programming models with an application to job search. *Journal of Economic Dynamics and Control* 15: 755–69.
- Allen, B. 1992. Approximate equilibria in microeconomic rational expectations models. *Journal of Economic Theory* 26: 244–60.
- Allen, B. 1985. The existence of fully rational expectations approximate equilibria with noisy price observations. *Journal of Economic Theory* 37: 213–53.
- Allen, B. 1985b. The existence of rational expectations equilibria in a large economy with noisy price observations. *Journal of Mathematical Economics* 14: 67–103.
- Allgower, E. L., and K. Georg. 1990. *Numerical Continuation Methods: An Introduction*. New York: Springer.
- Aluffi-Pentini, F., V. Parisi, and F. Zirilli. 1984. A differential-equations algorithm for nonlinear equations. *ACM Transactions on Mathematical Software* 10: 299–316.
- Aluffi-Pentini, F., V. Parisi, and F. Zirilli. 1984. Algorithm 617. DAFNE—A differential equations algorithm for nonlinear equations. *ACM Transactions on Mathematical Software* 10: 317–24.
- Ames, W. F. 1977. *Numerical Methods for Partial Differential Equations*. Orlando: Academic Press.
- Amir, R., L. J. Mirman, and W. R. Perkins. 1991. One-sector nonclassical optimal growth: Optimality conditions and comparative dynamics. *International Economic Review* 32: 625–44.
- Amman, H., D. Kendrick, and J. Rust, eds. 1996. *Handbook of Computational Economics*, vol. 1. Amsterdam: Elsevier.
- Amman, H. 1996. Numerical methods for linear-quadratic models. In H. Amman et al., eds., *Handbook of Computational Economics*, vol. 1. Elsevier: Amsterdam.
- Anderson, E., et al. 1992. *LAPACK User's Guide*. Philadelphia: SIAM, 1992.
- Anderson, E. W., L. P. Hansen, E. R. McGrattan, and T. J. Sargent. 1996. Mechanics of forming and estimating dynamic linear economies. In H. Amman et al., eds., *Handbook of Computational Economics*, vol. 1, Amsterdam: Elsevier.
- Anderson, R. M., and H. Sonnenschein. 1982. On the existence of rational expectations equilibrium. *Journal of Economic Theory* 26: 261–78.
- Anderson, G. S. 1993. Symbolic algebra programming for analyzing the long-run dynamics of economic models. In H. Varian, ed., *Economic and Financial Modeling with Mathematica*. New York: Springer.
- Araujo, A., and J. A. Scheinkman. 1977. Smoothness, comparative dynamics, and the turnpike property. *Econometrica* 45: 601–20.
- Arifovic, J. 1994. Genetic algorithm learning and the cobweb model. *Journal of Economic Dynamics and Control* 18: 3–28.
- Arrow, K. J., and F. H. Hahn. 1971. *General Competitive Analysis*. San Francisco: Holden-Day.
- Ascher, U. M., R. M. M. Mattheij, and R. D. Russell. 1988. *Numerical Solution of Boundary Value Problems for Ordinary Differential Equations*. Englewood Cliffs, NJ: Prentice-Hall.
- Ashcroft, R. N. 1995. Asset pricing with spectral methods. Ph. D. dissertation. Stanford University.

- Atkinson, K. 1989. *An Introduction to Numerical Analysis*. New York: Wiley.
- Aubin, J.-P., and I. Ekeland. 1984. *Applied Nonlinear Analysis*. New York: Wiley.
- Auerbach, A. J., and Laurence J. Kotlikoff. 1987. *Dynamic Fiscal Policy*. Cambridge: Cambridge University Press.
- Auerbach, A. J., and Laurence J. Kotlikoff. 1983. An examination of empirical tests of social security and savings. In E. Helpman, A. Razin, and E. Sadka, eds., *Social Policy Evaluation: An Economic Perspective*. New York: Academic Press.
- Auerbach, A., L. Kotlikoff, and J. Skinner. 1983. The Efficiency Gains from Dynamic Tax Reform. *International Economic Review* 24: 81–100.
- Ausubel, L. M. 1990a. Partially-revealing rational expectations equilibrium in a competitive economy. *Journal of Economic Theory* 50: 93–126.
- Ausubel, L. M. 1990b. Insider trading in a rational expectations economy. *American Economic Review* 80: 1022–41.
- Balcer, Y., and K. L. Judd. 1985. Dynamic effects of tax policy. Mimeo. Northwestern University.
- Barro, R. 1974. Are government bonds net wealth? *Journal of Political Economy* 82: 1095–1117.
- Barron, A. R. 1993. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory* 39: 930–45.
- Bazaraa, M., and C. M. Shetty. 1979. *Nonlinear Programming: Theory and Algorithms*. New York: Wiley.
- Bellman, R. E. 1957. *Dynamic Programming*. Princeton: Princeton University Press.
- Bellman, R., R. Kalaba, and B. Kotkin. 1963. Polynomial approximation—A new computational technique in dynamic programming: Allocation processes. *Mathematics of Computation* 17: 155–61.
- Bender, C. M., and S. A. Orszag. 1978. *Advanced Mathematical Methods for Scientists and Engineers*. New York: McGraw-Hill.
- Benhabib, J., and K. Nishimura. 1979. The Hopf bifurcation and the existence and stability of closed orbits in multisector models of optimal economic growth. *Journal of Economic Theory* 21: 421–44.
- Bennett, C. H., and R. Landauer. 1985. The fundamental physical limits of computation. *Scientific American* 253-1: 48–71.
- Bensoussan, A. 1988. *Perturbation Methods in Optimal Control*. New York: Wiley, 1988.
- Beneveniste, A., M. Métivier, and P. Priouret. 1990. *Adaptive Algorithms and Stochastic Approximations*. Berlin: Springer.
- Bernardo, A., and K. L. Judd. 1994. Asset market equilibrium with general securities, tastes, returns, and information asymmetries. Mimeo. Hoover Institution. 1994.
- Bernardo, A. 1996. The choice between regulatory and contractual restrictions on insider trading: A welfare analysis. Mimeo. University of California-Los Angeles.
- Bernardo, A., and K. L. Judd. 1997a. Efficiency of asset markets with asymmetric information. Mimeo. Hoover Institution.
- Bernardo, A., and K. L. Judd. 1997b. Volume and price formation in an asset trading model with asymmetric information. UCLA Working Paper.
- Bertsekas, D. 1982. *Constrained Optimization and LaGrange Multiplier Methods*. New York: Academic Press.
- Bertsekas, D. 1976. *Dynamic Programming and Stochastic Control*. New York: Academic Press.
- Bertsekas, D. P. 1995. *Dynamic Programming and Optimal Control*, vol. 1. Belmont, MA: Athena Scientific.
- Bertsekas, D. P. 1995. *Dynamic Programming and Optimal Control*, vol. 2. Belmont, MA: Athena Scientific.
- Bertsekas, D. P., and J. N. Tsitsiklis. 1996. *Neuro-dynamic Programming*. Belmont, MA: Athena Scientific.
- Bertsekas, D. P., and S. E. Shreve. 1978. *Stochastic Control: The Discrete Time Case*. New York: Academic Press.

- Bertsekas, D. P., and J. N. Tsitsiklis. 1989. *Parallel and Distributed Computation: Numerical Methods*. Englewood Cliffs, NJ: Prentice Hall.
- Berz, M., C. Bischof, G. Corliss, and A. Griewank. 1996. *Computational Differentiation: Techniques, Applications, and Tools*. Philadelphia: SIAM.
- Bizer, D., and K. L. Judd. 1989. Taxation and uncertainty. *American Economic Review* 79: 331–36.
- Blackwell, D. 1965. Discounted dynamic programming. *Annals of Mathematical Statistics* 36: 226–35.
- Bleistein, N., and R. A. Handelsman. 1976. *Asymptotic Expansions of Integrals*. New York: Holt, Rinehart and Winston.
- de Boor, C. 1978. *A Practical Guide to Splines*. New York: Springer.
- de Boor, C., and B. Swartz. 1977. Piecewise monotone interpolation. *Journal of Approximation Theory* 21: 411–16.
- Border, K. C. 1985. *Fixed Point Theorems with Applications to Economics and Game Theory*. Cambridge: Cambridge University Press.
- Botha, J. F., and G. F. Pinder. 1983. *Fundamental Concepts in the Numerical Solution of Differential Equations*. New York: Wiley.
- Boucekkine, R. 1995. An alternative methodology for solving nonlinear forward-looking models. *Journal of Economic Dynamics and Control* 19: 711–34.
- Bovenberg, A. L., and L. H. Goulder. 1991. Introducing intertemporal and open economy features in applied general equilibrium models. In H. Don, T. van de Klundert, and J. van Sinderen, eds., *Applied General Equilibrium Modelling*. Dordrecht: Kluwer Academic, pp. 47–64.
- Boyd, J. P. 1989. *Chebyshev and Fourier Spectral Methods*. Berlin: Springer.
- Braess, D. 1986. *Nonlinear Approximation Theory*. Berlin: Springer.
- Bratley, P., and B. L. Fox. 1988. Algorithm 659 implementing Sobol's quasirandom sequence generator. *ACM Transactions on Mathematical Software* 14: 88–100.
- Bratley, P., B. L. Fox, and L. E. Schrage. 1987. *A Guide to Simulation*. 2d ed. New York: Springer.
- Brent, R. P. 1973. Some efficient algorithms for solving systems of nonlinear equations. *SIAM Journal on Numerical Analysis* 10: 327–44.
- Brock, W. A. 1975. A simple perfect foresight monetary model. *Journal of Monetary Economics* 1: 133–50.
- Brock, W. A., and W. D. Dechert. 1988. Theorems on distinguishing deterministic from random systems. In W. A. Barnett, E. R. Berndt, and H. White, eds., *Dynamic Econometric Modeling: Proceedings of the Third International Symposium in Economic Theory and Econometrics*. International Symposia in Economic theory and Econometrics series. New York: Cambridge University Press, pp. 247–65.
- Brock, W. A., D. Hsieh, and B. LeBaron. 1991. *Nonlinear Dynamics, Chaos, and Instability*. Cambridge: MIT Press.
- Brock, W. A., and L. J. Mirman. 1972. Optimal economic growth and uncertainty: The discounted case. *Journal of Economic Theory* 4: 479–513.
- Brock, W. A. 1986. Distinguishing random and deterministic systems: Abridged version. *Journal of Economic Theory* 40: 168–95.
- Brock, W. A., and S. J. Turnovsky. 1981. The analysis of macroeconomic policies in perfect foresight equilibrium. *International Economic Review* 22: 179–209.
- Brown, D. J., P. M. DeMarzo, and B. C. Eaves. 1996. Computing equilibria when asset markets are incomplete. *Econometrica* 64: 1–27.
- Brock, W. A., and A. G. Malliaris. 1989. *Differential Equations, Stability and Chaos in Dynamic Economies*. Amsterdam: North Holland.
- Budd, C., C. Harris, and J. Vickers. 1993. A model of the evolution of duopoly: Does the asymmetry between firms tend to increase or decrease? *Review of Economic Studies* 60: 543–73.
- Burnett, D. S. 1987. *Finite Element Analysis*. Reading, MA: Addison-Wesley.

- Canzoneri, M. B., and D. W. Henderson. 1991. *Monetary Policy in Interdependent Economies: A Game-Theoretic Approach*. Cambridge: MIT Press.
- Caputo, M. R. 1990. How to do comparative dynamics on the back of an envelope in optimal control theory. *Journal of Economic Dynamics and Control* 14: 655–83.
- Caputo, M. R. 1990. Comparative Dynamics via envelope methods in variational calculus. *Review of Economic Studies* 57: 689–97.
- Burdick, C. A. 1994. Transitional dynamics in a monetary economy. Ph.D. dissertation. Stanford University.
- Burnett, D. S. 1987. *Finite Element Analysis*. Reading, MA: Addison-Wesley.
- Canuto, C., M. W. Hussaini, A. Quarteroni, and T. A. Zang. 1988. *Spectral Methods in Fluid Dynamics*. New York: Springer.
- Carter, R. G. 1993. Numerical experience with a class of algorithms for nonlinear optimization using inexact function and gradient information. *SIAM Journal of Scientific Computing* 14: 368–88.
- Chaitin-Chatelin, F., and V. Frayssé. 1996. *Lectures on Finite Precision Computations*. Philadelphia: SIAM.
- Chang, F.-W. 1988. The inverse optimal problem: A dynamic programming approach. *Econometrica* 56: 147–72.
- Cheney, E. W. 1966. *Introduction to Approximation Theory*. New York: McGraw-Hill.
- Cho, I.-K., and T. J. Sargent. 1996. Neural networks for encoding and adapting in dynamic economics. In Amman et al., eds., *Handbook of Computational Economics*, vol. 1. Amsterdam: Elsevier.
- Chow, S.-N., J. Mallet-Paret, and J. A. Yorke. 1978. Finding zeroes of maps: Homotopy methods that are constructive with probability one. *Mathematics of Computation* 32: 887–99.
- Chow, S.-N., and J. K. Hale. 1982. *Methods of Bifurcation Theory*. New York: Springer.
- Christiano, L. J. 1990. Solving the stochastic growth model by linear-quadratic approximation and by value-function iteration. *Journal of Business and Economic Statistics* 8: 23–26.
- Christopeit, N. 1983. Discrete approximation of continuous time stochastic control systems. *SIAM Journal on Control and Optimization* 21: 17–40.
- Chung, K. L. 1974. *A Course in Probability Theory*, 2d ed. New York: Academic Press.
- Chung, K. L. 1949. An estimate concerning the Kolmogoroff limit distribution. *Transactions of the American Mathematical Society* 67: 36–50.
- Cohen, A. M., and D. A. Gismalla. 1985. The construction of quadrature rules by parameter optimization. *International Journal of Computational Mathematics* 17: 203–14.
- Cohen, A. M., and D. A. Gismalla. 1986. Some integration for symmetric functions of two variables. *International Journal of Computational Mathematics* 19: 57–68.
- Coleman, W. J., II. 1990. Solving the stochastic growth model by policy function iteration. *Journal of Business and Economic Statistics* 8: 27–29.
- Cooley, T., and G. Hansen. 1989. The inflation tax in a real business cycle model. *American Economic Review* 79: 733–48.
- Costantini, P., and F. Fontanella. 1990. Shape-preserving bivariate interpolation. *SIAM Journal of Numerical Analysis* 27: 488–506.
- Cranley, R., and T. N. L. Patterson. 1976. Randomization of number theoretic methods for multiple integration. *SIAM Journal of Numerical Analysis* 13: 904–14.
- Cronshaw, M. B., and D. G. Luenberger. 1994. Strongly symmetric subgame perfect equilibria in infinitely repeated games with perfect monitoring and discounting. *Games and Economic Behavior* 6: 220–37.
- Cuyt, A., and L. Wuytack. 1986. *Nonlinear Numerical Methods: Theory and Practice*. Amsterdam: North-Holland.

- Daniel, J. W. 1976. Splines and efficiency in dynamic programming. *Journal of Mathematical Analysis and Applications* 54: 402–407.
- Danthine, J.-P., J. B. Donaldson, and R. Mehra. 1989. On some computational aspects of equilibrium business cycle theory. *Journal of Economic Dynamics and Control* 13: 449–70.
- Dantzig, G. B., R. P. Harvey, Z. F. Lansdowne, and R. D. McKnight. 1974. DYGAM—A computer system for the solutions of dynamic programs. Control Analysis Corporation, Palo Alto, CA, August.
- Davis, P. J., and P. Rabinowitz. 1984. *Methods of Numerical Integration*, 2d ed. New York: Academic Press.
- Deaton, A., and G. Laroque. 1992. On the behavior of commodity prices. *Review of Economic Studies* 59: 1–23.
- den Haan, W. 1995. Aggregate shocks and cross-sectional dispersion. Discussion Paper. Department of Economics, UCSD.
- Dixit, A. 1991. Analytical approximations in models of hysteresis. *Review of Economic Studies* 58: 141–51.
- Dotsey, M., and C. S. Mao. 1992. How well do linear approximation methods work? *Journal of Monetary Economics* 29: 25–58.
- Denardo, E. V. 1967. Contraction mappings in the theory underlying dynamic programming. *SIAM Review* 9: 165–77.
- Dennis, J. E., Jr., and R. B. Schnabel. 1983. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice Hall Series in Computational Mathematics. Englewood Cliffs, NJ: Prentice Hall.
- Dennis, J. E., Jr., and R. B. Schnabel. 1989. A view of unconstrained optimization. In G. L. Nemhauser, A. H. G. Rinnooy Kan, and M. J. Todd, eds., *Optimization*. Amsterdam: North-Holland.
- Devroye, L. 1986. *Non-uniform Random Variate Generation*. New York: Springer.
- Dixon, P., and B. Parmenter. 1996. Computable general equilibrium modelling for policy analysis and forecasting. In H. Amman et al., eds., *Handbook of Computational Economics*, vol. 1. Amsterdam: Elsevier.
- Donovan, G. C., A. R. Miller, and T. J. Moreland. 1993. Pathological functions for Newton's method. *American Mathematical Monthly*, 100: 53–58.
- Doren, C. C. Y. 1986. Limiting distribution for random optimization methods. *SIAM Journal on Control and Optimization* 24: 76–82.
- Dorsey, R. E., and W. J. Mayer. 1992. Genetic algorithms for estimation problems with multiple optima, non-differentiability and other irregular features. Mimeo. University of Mississippi.
- Dwyer, G. P., Jr. 1997. Random number generators. Mimeo.
- Eaves, B. C. 1971. Computing kakutani fixed points. *SIAM Journal of Applied Mathematics* 21: 236–44.
- Eaves, B. C. 1972. Homotopies for computation of fixed points. *Mathematical Programming* 3: 1–22.
- Elhay, S., and J. Kautsky. 1987. Algorithm 655 IQPACK: FORTRAN subroutines for the weights of interpolatory quadratures. *ACM Transactions on Mathematical Software* 13: 399–416.
- Ermoliev, Y., and R. J.-B. Wets, eds. 1988. *Numerical Techniques for Stochastic Optimization*. Berlin: Springer.
- Estrada, R., and R. P. Kanwal. 1994. *Asymptotic Analysis: A Distributional Approach*. Boston: Birkhauser Press.
- Evans, G. 1993. *Practical Numerical Integration*. New York: Wiley.
- Fair, R., and J. Taylor. 1983. Solution and maximum likelihood estimation of dynamic nonlinear rational expectation models. *Econometrica* 51: 1169–85.
- Feldstein, M. 1974. Social security, induced retirement, and aggregate capital accumulation. *Journal of Political Economy* 82: 905–26.

- Fershtman, C., and M. Kamien. Dynamic duopolistic competition with sticky prices. *Econometrica* 55: 1151–64.
- Fischer, S. 1979. Capital accumulation on the transition path in a monetary optimizing model. *Econometrica* 47: 1433–39.
- Fleming, W. H. 1971. Stochastic control for small noise intensities. *SIAM Journal of Control* 9: 473–517.
- Fleming, W., and P. E. Souganides. 1986. Asymptotic series and the method of vanishing viscosity. *Indiana University Mathematics Journal* 35: 425–47.
- Fletcher, C. A. J. 1984. *Computational Galerkin Techniques*, New York: Springer.
- Fletcher, C. A. J. 1988. *Computational Techniques for Fluid Dynamics, 2 vols.* Berlin: Springer.
- Fox, B. L. 1986. Algorithm 647: Implementation and relative efficiency of quasirandom sequence generators. *ACM Transactions on Mathematical Software* 12: 362–76.
- Fritsch, F. N., and R. E. Carlson. 1980. Monotone piecewise cubic interpolation. *SIAM*, 17: 238–46.
- Fudenberg, D., and J. Tirole. 1991. *Game Theory*. Cambridge: MIT Press.
- Garcia, C. B., and W. I. Zangwill. 1981. *Pathways to Solutions, Fixed Points, and Equilibria*. Englewood Cliffs, NJ: Prentice Hall.
- Gaspar, J., and K. L. Judd. 1997. Solving large-scale rational expectations models. *Macroeconomic Dynamics* 1: 45–75.
- Geer, J. F., and C. M. Andersen. 1990. A hybrid perturbation-Galerkin technique that combines multiple expansions. *SIAM Journal of Applied Mathematics* 50: 1474–95.
- Geweke, J. 1996. Monte Carlo simulation and numerical integration. In H. Amman et al., eds., *Handbook of Computational Economics*, vol. 1. Amsterdam: Elsevier.
- Ghysels, E., and O. Lieberman. 1993. Dynamic regression and filtered data series: A Laplace approximation to the effects of filtering in small samples. Mimeo. University of Montreal.
- Gill, P. E., W. Murray, and M. H. Wright. 1981. *Practical Optimization*. London: Academic Press, 1981.
- Gill, P. E., W. Murray, M. A. Saunders, and M. White. 1989. Constrained nonlinear programming. In G. L. Nemhauser, A. H. G. Rinnooy Kan, and M. J. Todd eds., *Optimization*. Amsterdam: North-Holland.
- Gill, P., W. Murray, M. A. Saunders, J. Tomlin, and M. H. Wright. 1982. A note on interior point methods for linear programming. *COAL Newsletter* 13: 13–19.
- Gilli, M., and G. Pauletto. 1998. Nonstationary iterative methods for solving models with forward looking variables. *Journal of Economic Dynamics and Control* (forthcoming).
- Goffe, W. L., G. C. Ferrier, and J. Rogers. 1992. Simulated annealing: An initial application in econometrics. *Computational Economics* 5: 133–46.
- Goffe, W. L., G. C. Ferrier, and J. Rogers. 1994. Global optimization of statistical functions with simulated annealing. *Journal of Econometrics*, 60: 65–99.
- Goldberg, D. E. 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison-Wesley.
- Goldfarb, D., and M. J. Todd. 1989. Linear Programming. In G. L. Nemhauser, A. H. G. Rinnooy Kan, and M. J. Todd, eds., *Optimization*. Amsterdam: North-Holland.
- Golomb, M. 1959. Approximation by Functions of Fewer Variables. In R. E. Langer, *On Numerical Approximation*. Madison: University of Wisconsin Press, pp. 275–327.
- Golub, G. H., and C. F. van Loan. 1983. *Matrix Calculations*. Baltimore: Johns Hopkins University Press.
- Gottlieb, D., and S. A. Orszag. 1977. *Numerical Analysis of Spectral Methods: Theory and Applications*, Philadelphia: SIAM-CBMS.
- Greenblatt, S. A. 1994. Tensor methods for full-information maximum likelihood estimation: Unconstrained estimation. *Journal of Computational Economics* 7: 89–108.

- Griewank, A., and G. F. Corliss, ed. 1991. *Automatic Differentiation of Algorithms: Theory, Implementation, and Application*. Philadelphia: SIAM.
- Grossman, S. J., and O. D. Hart. 1983. An analysis of the principal–agent problem. *Econometrica* 51: 7–45.
- Grossman, S. 1976. On the efficiency of competitive stock markets where agents have diverse information. *Journal of Finance* 18: 81–101.
- Grossman, S. J., and J. E. Stiglitz. 1980. On the impossibility of informationally efficient markets. *American Economic Review* 70: 393–408.
- Gustafson, R. L. 1958. Carryover levels for grains. Technical Bulletin No. 1178. U.S. Department of Agriculture.
- den Haan, W., and A. Marcet. 1990. Solving the stochastic growth model by parameterizing expectations. *Journal of Business and Economic Statistics* 8: 31–34.
- Haas, A. 1987. The multiple prime random number generator. *ACM Transactions on Mathematical Software* 13: 368–81.
- Hageman, L. A., and D. M. Young. 1981. *Applied Iterative Methods*. Orlando: Academic Press.
- Halton, J. M. On the efficiency of evaluating certain quasi-random sequences of points in evaluating multi-dimensional integrals. *Numerische Mathematik* 3: 84–90.
- Halton, J. H., and D. C. Handscomb. 1957. A method for increasing the efficiency of Monte Carlo integration. *Journal of the ACM* 4: 329–40.
- Hammersley, J. M., and D. C. Handscomb. 1964. *Monte Carlo Methods*. London: Chapman and Hall.
- Hansen, L. P., and T. J. Sargent. 1996. Recursive linear models of dynamic economies. Unpublished manuscript.
- Hart, O. D. 1975. On the optimality of equilibrium when the market structure is incomplete. *Journal of Economic Theory* 11: 418–43.
- Haselgrove, C. B. 1961. A method for numerical integration. *Mathematical Computation* 15: 323–37.
- Hartman, P. 1964. *Ordinary Differential Equations*. New York: Wiley.
- Haubrich, J. G. 1994. Risk aversion, performance pay, and the principal-agent problem. *Journal of Political Economy* 102: 258–76.
- Hlawka, E. 1961. Funktionen von beschränkter Variation in der Theorie der Gleichverteilung. *Annali di Matematica Pura ed Applicata* (Bologna) 54: 325–33.
- Holland, J. H. 1975. *Adaptation in Natural and Artificial Systems*. Ann Arbor: University of Michigan Press.
- Holly, A., and P. C. B. Phillips. 1979. A saddlepoint approximation to the distribution of the k -class estimator in a coefficient in a simultaneous system. *Econometrica* 47: 1527–48.
- Hornik, K., M. Stinchcombe, and H. White. 1989. Multi-layer feedforward networks are universal approximators. *Neural Networks* 2: 359–66.
- Hornik, K., M. Stinchcombe, and H. White. 1990. Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks. *Neural Networks* 3: 551–60.
- Horst, R., and P. M. Pardalos, eds. 1995. *Handbook of Global Optimization*. Dordrecht: Kluwer.
- Hua, L. K., and Y. Wang. 1981. *Applications of Number Theory to Numerical Analysis*. Berlin: Springer.
- Hughes Hallett, A. J., and L. Piscitelli. 1998. Simple reordering techniques for expanding the convergence radius of first-order iterative techniques. *Journal of Economic Dynamics and Control* (forthcoming).
- Hull, J. 1989. *Options, Futures, and Other Derivative Securities*. Englewood Cliffs, NJ: Prentice Hall.
- Jensen, M. C., and K. J. Murphy. 1990. Performance pay and top-management incentives. *Journal of Political Economy* 98: 225–64.

- Johnson, S., J. R. Stedinger, C. A. Shoemaker, Y. Li, and J. A. Tehada-Guibert. 1993. Numerical solution of continuous-state dynamic programs using linear and spline interpolation. *Operations Research* 41: 484–500.
- Juillard, M., D. Laxton, P. McAdam, and H. Pioro. 1998. An algorithm competition: First-order iterations versus Newton-based techniques. *Journal of Economic Dynamics and Control* (forthcoming).
- Judd, K. L. 1982. An alternative to steady-state comparisons in perfect foresight models. *Economics Letters* 10: 55–59.
- Judd, K. L. 1985. Credible Spatial Preemption, *Rand Journal of Economics*, 16: 153–66.
- Judd, K. L. 1985. Short-run analysis of fiscal policy in a simple perfect foresight model. *Journal of Political Economy* 93: 298–319.
- Judd, K. L. 1985. Closed-loop equilibrium in a multistage innovation race. Unpublished manuscript.
- Judd, K. L. 1987. Debt and distortionary taxation in a simple perfect foresight model. *Journal of Monetary Economics* 20: 51–72.
- Judd, K. L. 1987. Welfare cost of factor taxation in a perfect foresight model. *Journal of Political Economy* 95: 675–709.
- Judd, K. L. 1990. Asymptotic methods in dynamic economic models. Mimeo. Hoover Institution, Stanford University.
- Judd, K. L. 1992. Projection methods for solving aggregate growth models. *Journal of Economic Theory* 58: 410–52.
- Judd, K. L. 1997. Computational economics and economic theory: Substitutes or complements? *Journal of Economic Dynamics and Control* 21: 907–42.
- Judd, K. L., and S.-M. Guu. 1993. Perturbation solution methods for economic growth models. In H. Varian, eds., *Economic and Financial Modeling with Mathematica*. New York: Springer.
- Judd, K. L. 1996. Approximation, perturbation, and projection methods in economic analysis. In H. Amman, D. Kendrick, and J. Rust, eds., *Handbook of Computational Economics*. Amsterdam: Elsevier.
- Judd, K. L., and J. Conklin. 1995. Computing supergame equilibrium. Mimeo.
- Judd, K. L., and S.-M. Guu. 1996. Bifurcation approximation methods applied to asset market equilibrium. Mimeo. Hoover Institution.
- Judd, K. L., and S.-M. Guu. 1997. Asymptotic methods for aggregate growth models. *Journal of Economic Dynamics and Control* 21: 1025–42.
- Judd, K. L., F. Kubler, and K. Schmedders. 1997. Computing equilibria in infinite horizon finance economies: The case of one asset. Mimeo. Hoover Institution.
- Judd, K. L., F. Kubler, and K. Schmedders. 1997. Incomplete asset markets with heterogeneous tastes and idiosyncratic income. Mimeo.
- Judd, K. L., and A. Solnick. 1994. Numerical dynamic programming with shape-preserving splines. Mimeo.
- Kalaba, R., L. Tesfatsion, and J.-L. Wang. 1983. A finite algorithm for the exact evaluation of higher order partial derivatives of functions of many variables. *Journal of Mathematical Analysis and Applications* 92: 552–63.
- Kalaba, R., and L. Tesfatsion. 1991. Solving nonlinear equations by adaptive homotopy continuation. *Applied Mathematics and Computation* 41: 99–115.
- Kamien, M. I., and N. L. Schwartz. 1981. *Dynamic Optimization: The Calculus of Variations and Optimal Control in Economics and Management*. New York: North Holland.
- Kan, A., H. G. Rinnooy, and G. T. Timmer. 1989. Global optimization. In G. L. Nemhauser, A. H. G. Rinnooy Kan, and M. J. Todd, eds., *Optimization*. Amsterdam: North-Holland.
- Karmarkar, N. 1984. A new polynomial-time algorithm for linear programming. In *Proceedings of the 16th Annual ACM Symposium on the Theory of Computing*, pp. 302–11.

- Kaufmann, W. J., III, and L. L. Smarr. 1993. *Supercomputing and the Transformation of Science*. New York: Scientific American Library.
- Kehoe, T. J. 1991. Computation and multiplicity of equilibria. In W. Hildenbrand and H. Sonnenschein, eds., *Handbook of Mathematical Economics*, vol. 4. Amsterdam: North-Holland.
- Kendrick, D. Research opportunities in computational economics. *Computational Economics* 6: 257–314.
- Kendrick, D. 1995. Ten Wishes. *Computational Economics* 8: 67–80.
- Kendrick, D. A. 1996. Sectoral economics. In H. Amman et al., eds., *Handbook of Computational Economics*, vol. 1. Amsterdam: Elsevier.
- Keane, M., and K. Wolpin. 1994. The solution and estimation of discrete choice dynamic programming models by simulation: Monte Carlo evidences. *Review of Economics and Statistics* 76: 648–72.
- Kollerstrom, N. 1992. Thomas Simpson and “Newton’s method of approximation”: An enduring myth. *British Journal of History of Science* 25: 347–54.
- Lemarechal, C. Nondifferentiable Optimization. 1989. In G. L. Nemhauser, A. H. G. Rinnooy Kan, and M. J. Todd, eds., *Optimization*. Amsterdam: North-Holland.
- Kautsky, J., and S. Elhay. 1982. Calculation of the weights of interpolatory quadratures. *Numerical Mathematics* 40: 407–22.
- Keast, P. 1973. Optimal parameters for multidimensional integration. *SIAM Journal of Numerical Analysis* 10: 831–38.
- Kiefer, J. 1962. On large deviations of the empiric d.f. of vector chance variables and a law of the iterated logarithm. *Pacific Journal of Mathematics* 11: 649–60.
- Kirkpatrick, S., C. D. Gelatt, Jr., and M. P. Vecchi. 1983. Optimization by simulated annealing. *Science* 220: 671–75.
- Kloek, T., and H. K. van Dijk. 1978. Bayesian estimates of equation system parameters: An application of integration by Monte Carlo. *Econometrica* 46: 1–20.
- Koksma, J. F. 1942. Een algemeene stelling uit de theorie der gelijkmatige verdeeling modulo 1. *Mathematica B (Zutphen)* 7–11.
- Kolmogorov, A. N. 1957. On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition. *Doklady Akademii Nauk SSSR* 114: 953–56.
- Korobov, N. M. 1959. On approximate calculations of multiple integrals. *Doklady Akademii Nauk SSSR* 124: 1207–10 (in Russian).
- Korobov, N. M. 1960. Properties and calculation of optimal coefficients. *Soviet Mathematics, Doklady* 1: 696–700.
- Krasnosel'skii, M. A., and P. Zabreiko. 1984. *Geometrical Methods of Nonlinear Analysis*. Berlin: Springer.
- Kubicek, M., and V. Hlavacek. 1983. *Numerical Solution of Nonlinear Boundary Value Problems with Applications*. Englewood Cliff, NJ: Prentice Hall.
- Kuipers, L., and H. Niederreiter. 1974. *Uniform Distribution of Sequences*. New York: John Wiley.
- Kulisch, U. W., and W. L. Miranker. 1986. The arithmetic of the digital computer: A new approach. *SIAM Review* 28: 1–40.
- Kushner, H. J., and D. S. Clark. 1978. *Stochastic Approximation Methods for Constrained and Unconstrained Systems*. New York: Springer.
- Kushner, H. J., and P. G. Dupuis. 1992. *Numerical Methods for Stochastic Control Problems in Continuous Time*. New York: Springer.
- Kushner, H. J., and H. Huang. 1979. Rates of Convergence for Stochastic Approximation Type Algorithms. *SIAM Journal of Control and Optimization* 17: 607–17.
- Kushner, H. J., and H. Huang. 1981. Asymptotic Properties on Stochastic Approximations with Constant Coefficients. *SIAM Journal of Control and Optimization* 19: 87–105.

- Kydland, F. E., and E. C. Prescott. 1982. Time to build and aggregate fluctuations. *Econometrica* 50: 1345–70.
- Kyle, A. S. 1985. Continuous auctions and insider trading. *Econometrica* 53: 1315–35.
- Laitner, J. 1990. Tax changes and phase diagrams for an overlapping-generations model. *Journal of Political Economy* 98: 193–220.
- Laitner, J. 1987. The dynamic analysis of continuous-time life cycle saving growth models. *Journal of Economic Dynamics and Control* 11: 331–57.
- Laitner, J. 1989. Transition time paths for overlapping-generations models. *Journal of Economic Dynamics and Control* 7: 111–29.
- Lapidus, L., and G. F. Pinder. 1982. *Numerical Solution of Partial Differential Equations in Science and Engineering*. New York: Wiley.
- Lemke, C. E., and J. T. Howson. 1964. Equilibrium points of bimatrix games. *SIAM Journal of Applied Mathematics* 12: 413–23.
- Li, J. 1998. Numerical analysis of a nonlinear operator equation arising from a monetary model. *Journal of Economic Dynamics and Control* (forthcoming).
- Lick, W. J. 1989. Difference equations from differential equations. Lecture Notes in Engineering 41. Berlin: Springer.
- Lipton, D., J. Poterba, J. Sachs, and L. Summers. 1982. Multiple shooting in rational expectations models. *Econometrica* 50: 1329–34.
- Ljung, L., and T. Soderstrom. 1983. *Theory and Practice of Recursive Identification*. Cambridge: MIT Press.
- Lucas, R. E., Jr. 1978. Asset prices in an exchange economy. *Econometrica* 46: 1429–45.
- Luenberger, D. 1984. *Linear and Nonlinear Programming*. Reading, MA: Addison-Wesley.
- Magill, M. 1977. A Local Analysis of N-Sector Capital Accumulation under Uncertainty. *Journal of Economic Theory* 15: 211–18.
- Malliaris, A. G., and W. A. Brock. 1982. *Stochastic Methods in Economics and Finance*. Amsterdam: North-Holland.
- Marimon, Ramon, E. McGrattan, and T. J. Sargent. 1990. Money as a medium of exchange in an economy with artificially intelligent agents. *Journal of Economic Dynamics and Control* 14: 329–73.
- Marks, R. E. 1992. Breeding hybrid strategies: Optimal behaviour for oligopolists. *Journal of Evolutionary Economics* 2: 17–38.
- Marsaglia, G. 1968. Random numbers fall mainly in the planes. *Proceedings of the National Academy of Sciences* 60: 25–28.
- Mas-Colell, A. 1974. A note on a theorem of F. Browder. *Mathematical Programming* 6: 229–33.
- McGrattan, E. R. 1990. Solving the stochastic growth model by linear-quadratic approximation. *Journal of Business and Economic Statistics* 8: 41–43.
- McKelvey, R. D. 1996. A Lyapunov function for Nash equilibria. Social Science Working Paper 953. California Institute of Technology.
- McKelvey, R. D. 1998. Computation of equilibria in finite games. *Handbook of Computational Economics*, forthcoming.
- Miranda, M. J., and P. G. Helmburger. 1988. The effects of commodity price stabilization programs. *American Economic Review* (March): 46–58.
- Miranda, M. J., and J. W. Glauber. 1993. Estimation of dynamic nonlinear rational expectations models of primary commodity markets with private and government stockholding. *Review of Economics and Statistics* 75: 463–70.
- Miranda, M., and X. Rui. 1997. Maximum likelihood estimation of the nonlinear rational expectations asset pricing model. *Journal of Economic Dynamics and Control* 21: 1493–510.

- More, J. J., and S. J. Wright. 1993. *Optimization Software Guide*. Philadelphia: SIAM.
- Murdock, J. A. 1991. *Perturbations: Theory and Methods*, New York: Wiley-Interscience.
- Mysovskikh, I. P. 1975. On Chakalov's theorem. *USSR Comput. Math. Math. Phys.* 15: 221–27.
- Nagurney, A. 1993. *Network Economics: A Variational Inequality Approach*. Dordrecht: Kluwer.
- Nagurney, A. 1996. Parallel computation. In H. Amman et al., eds., *Handbook of Computational Economics*. Amsterdam: Elsevier.
- Niederreiter, H. 1972. On a number-theoretical integration method. *Aequationes Mathematicae* 8: 304–311.
- Niederreiter, H. 1978. Quasi-Monte Carlo methods and pseudo-random numbers. *Bulletin of the American Mathematical Society* 84: 957–1041.
- Niederreiter, H. 1992. *Random Number Generation and Quasi-Monte Carlo Methods*. Philadelphia: SIAM.
- Niederreiter, H., and K. McCurley. 1979. Optimization of functions by quasi-random search methods. *Computing* 22: 119–23.
- Nurnberger, G. 1989. *Approximation by Spline Functions*. Berlin: Springer.
- Ortega, J. M., and W. C. Rheinboldt. 1970. *Iterative Solution of Nonlinear Equations in Several Variables*. New York: Academic Press.
- Paskov, S. H. 1993. Average case complexity of multivariate integration for smooth functions. *Journal of Complexity* 9: 291–312.
- Paskov, S. H., and J. F. Traub. 1995. Faster evaluation of financial derivatives. *Journal of Portfolio Management* 22: 113–20.
- Paskov, S. H. 1994. New methodologies for valuing derivatives. Technical Report. Computer Sciences Department, Columbia University.
- Paskov, S. H. 1995. Termination criteria for linear problems. *Journal of Complexity* 11: 105–37.
- Papageorgiou, A., and J. F. Traub. 1997. Faster evaluation of multidimensional integrals. Mimeo. Columbia University.
- Papageorgiou, A., and J. F. Traub. 1996. Beating Monte Carlo. *RISK* 9: 63–65.
- Phelan, C. J., and R. M. Townsend. 1991. Formulating and computing solutions to the infinite period principal-agent problem. *Review of Economic Studies* 58: 853–81.
- Piessens, R., E. de Doncker-Kapenga, C. W. Überhuber, and D. K. Kahaner. 1983. *QUADPACK: A Subroutine Package for Automatic Integration*. Berlin: Springer.
- Pissanetzky, S. 1984. *Sparse Matrix Technology*. London: Academic Press.
- Powell, M. J. D. 1981. *Approximation Theory and Methods*. Cambridge: Cambridge University Press.
- Powell, M. J. D. 1970. A hybrid method for nonlinear equations. In P. Rabinowitz, ed., *Numerical Methods for Nonlinear Algebraic Equations*. London: Gordon and Breach.
- Prenter, P. M. 1989. *Splines and Variational Methods*. New York: Wiley.
- Press, W. H., B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. 1986. *Numerical Recipes: The Art of Scientific Computing*. New York: Cambridge University Press.
- Puterman, M. L. 1994. *Markov Decision Processes*. New York: Wiley.
- Puterman, M. L., and S. L. Brumelle. 1979. On the convergence of policy iteration in stationary dynamic programming. *Mathematics of Operations Research* 4: 60–69.
- Puterman, M. L., and M. C. Shin. 1978. Modified policy iteration algorithms for discounted Markov decision problems. *Management Science* 24: 1127–37.
- Quirmbach, H. 1993. R&D: Competition, risk, and performance. *Rand Journal of Economics* 24: 157–97.
- Radner, R. 1979. Rational expectations equilibrium: Generic existence and the information revealed by prices. *Econometrica* 47: 655–78.

- Rall, L. B. 1981. *Automatic Differentiation: Techniques and Applications*. Lecture Notes in Computer Science 120. Berlin: Springer.
- Rasch, and Williamson. 1990. On shape-preserving interpolation and semi-Lagrangian transport. *SIAM Journal of Scientific Statistical Computation* 11: 656–87.
- Rheinboldt, W. C. 1986. *Numerical Analysis of Parameterized Nonlinear Equations*. New York: Wiley.
- Rice, J. R. 1983. *Numerical Methods, Software, and Analysis*. New York: McGraw-Hill.
- Richtmeyer, R. D. 1952. On the evaluation of definite integrals and a quasi-Monte Carlo method based on properties of algebraic numbers. Report LA-1342. Los Alamos: Los Alamos Scientific Laboratories.
- Ripley, B. D. 1987. *Stochastic Simulation*. New York: Wiley.
- Rivlin, T. J. 1969. *An Introduction to the Approximation of Functions*. Waltham, MA: Blaisdell.
- Rivlin, T. J. 1990. *Chebyshev Polynomials: From Approximation Theory to Algebra and Number Theory*. New York: Wiley-Interscience.
- Robbins, H., and S. Monro. 1951. A stochastic approximation method. *Annals of Mathematical Statistics* 22: 400–407.
- Roberts, S. M., and J. S. Shipman. 1971. *Two Point Boundary Value Problems: Shooting Methods*. New York: Elsevier.
- Ross, S. M. 1990. *A Course in Simulation*. New York: Macmillan.
- Ross, S. M. 1983. *Introduction to Stochastic Dynamic Programming*. New York: Academic Press.
- Roth, K. F. 1954. On the irregularities of distribution. *Mathematika* 1: 73–79.
- Rothschild, M., and J. Stiglitz. 1976. Equilibrium in competitive insurance markets: An essay on the economics of imperfect information. *Quarterly Journal of Economics* 90: 629–49.
- Rubinstein, R. V. 1981. *Simulation and the Monte Carlo Method*. New York: Wiley.
- Rui, X., and M. Miranda. 1996. Solving nonlinear dynamic games via orthogonal collocation: An application to international commodity markets. *Annals of Operations Research* 68: 89–108.
- Rust, J. 1996. Numerical Dynamic Programming in Economics. In H. Amman et al., eds., *Handbook of Computational Economics*. Amsterdam: Elsevier.
- Saad, Y. 1984. Practical use of some Krylov subspace methods for solving indefinite and nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing* 51: 203–28.
- Saad, Y., and M. H. Schulz. 1986. GMRES: A generalized minimum residual algorithm for solving non-symmetric linear systems. *SIAM Journal on Scientific and Statistical Computing* 7: 856–69.
- Samuelson, P. A. 1970. The fundamental approximation theorem of portfolio analysis in terms of means, variances and higher moments. *Review of Economic Studies* 37: 537–42.
- Santos, M. S. 1994. Smooth dynamics and Computation in models of economic growth. *Journal of Economic Dynamics and Control* 18: 879–95.
- Santos, M. S., and J.-L. Vila. 1991. Smoothness of the policy function in continuous times economic models: The one-dimensional case. *Journal of Economic Dynamics and Control* 15: 741–53.
- Sargent, T. J. 1987. *Macroeconomic Theory*. Orlando: Academic Press.
- Sargent, T. J. 1987. *Dynamic Macroeconomic Theory*. Cambridge: Harvard University Press.
- Scarf, H. E. 1967. The approximation of fixed points of a continuous mapping. *SIAM Journal of Applied Mathematics* 15: 328–43.
- Scarf, H. 1982. The computation of equilibrium prices: An exposition. In K. Arrow and M. Intriligator, eds., *Handbook of Mathematical Economics*. Amsterdam: North-Holland.
- Scarf, H., with T. Hansen. 1973. *Computation of Economic Equilibria*. New Haven: Yale University Press.
- Schmedders, K. 1996. Damped asset trading in the general equilibrium model with incomplete asset markets. Technical Report 96-11. Department of Operations Research, Stanford University.

- Schmedders, K. 1998. Computing equilibria in the general equilibrium model with incomplete asset markets. *Journal of Economic Dynamics and Control* (forthcoming).
- Schnabel, R. B., and T.-T. Chow. 1985. Tensor methods for unconstrained optimization using second derivatives. *SIAM Journal on Optimization* 1: 293–315.
- Schumaker, L. L. 1983. On shape-preserving quadratic spline interpolation. *SIAM Journal of Numerical Analysis* 20: 854–64.
- Shoven, J. B., and J. Whalley. 1992. *Applying General Equilibrium*. New York: Cambridge University Press.
- Smith, A., Jr. 1990. *Three Essays on the Solution and Estimation of Dynamic Macroeconomic Models*. Ph.D. dissertation. Duke University.
- Sobol, I. M. 1979. On the systematic search in a hypercube. *SIAM Journal of Numerical Analysis*, 16: 790–93.
- Solis, F. J., and R. J. B. Wets. 1981. Minimization by random search techniques. *Mathematics of Operations Research* 1: 19–30.
- Spear, S., and S. Srivastava. On repeated moral hazard with discounting. *Review of Economic Studies* 54: 599–617.
- Spence, A. M. 1974. Competitive and optimal responses to signals: An analysis of efficiency and distribution. *Journal of Economic Theory* 7: 296–332.
- Stewart, G. W. 1984. *Introduction to Matrix Computations*. New York: Academic Press.
- Stoer, J., and R. Burlisch. 1980. *Introduction to Numerical Analysis*. New York: Springer.
- Stokey, N., and R. Lucas. 1989. *Recursive Methods in Economic Dynamics*, Cambridge: Harvard University Press.
- Stroud, A. H. 1971. *Approximate Calculation of Multiple Integrals*. Englewood Cliffs, NJ: Prentice Hall.
- Stroud, A., and D. Secrest. 1966. *Gaussian Quadrature Formulas*. Englewood Cliffs, NJ: Prentice Hall.
- Sugihara, M., and Murota, K. 1982. A note on Haselgrave's method for numerical integration. *Mathematics of Computation* 39: 549–54.
- Tauchen, G., and R. Hussey. 1991. Quadrature-based methods for obtaining approximate solutions to the integral equations of nonlinear rational expectations models. *Econometrica* 59: 371–96.
- Tauchen, G. 1986. Statistical properties of generalized method-of-moments estimates of structural parameters obtained from financial market data. *Journal of Business and Economic Statistics* 4: 397–416.
- Taylor, J. B., and H. Uhlig. 1990. Solving nonlinear stochastic growth models: A comparison of alternative solution methods. *Journal of Business and Economic Statistics* 8: 1–18.
- Tesfatsion, L. 1992. Nonlocal automated comparative static analysis. *Computer Science in Economics and Management* 5: 313–31.
- Tierney, L., and Kadane, J. B. 1986. Accurate approximations of posterior moments and marginal densities. *Journal of the American Statistical Association* 81: 82–86.
- Tierney, L., R. E. Kass, and J. B. Kadane. 1986. Fully exponential Laplace approximations to expectations and variances of nonpositive functions. *Journal of the American Statistical Association* 81: 82–86.
- Traub, J. F., and H. Wozniakowski. 1980. *A General Theory of Optimal Algorithms*. New York: Academic Press.
- Traub, J. F., and H. Wozniakowski. 1980. Convergence and complexity of interpolatory-Newton iteration in a Banach space. *Computational Mathematics with Applications* 6: 385–400.
- Traub, J. F., and H. Wozniakowski. 1992. The Monte Carlo algorithm with a pseudorandom generator. *Mathematics of Computation* 58: 323–39.
- Traub, J. F., G. W. Wasilkowski, and H. Wozniakowski. 1988. *Information-based complexity*. New York: Academic Press.

- Trick M., and S. Zin. 1997. Spline approximations to value functions: A linear programming approach. *Macroeconomic Dynamics* 1: 255–77.
- Turnovsky, S. J., and W. A. Brock. 1980. Time consistency and optimal government policies in perfect foresight equilibrium. *Journal of Public Economics* 13: 183–212.
- Varian, H. 1978. *Microeconomic Theory*. New York: Norton, 1978.
- Vose, M. D. 1991. Generalizing the notion of schema in genetic algorithms. *Artificial Intelligence* 50: 385–96.
- Weiser, A., and S. Zarantonello. 1988. A note on piecewise linear and multilinear table interpolation in many dimensions. *Mathematics of Computation* 50: 189–96.
- White, H. 1992. *Artificial Neural Networks: Approximation and Learning Theory*. Oxford: Blackwell.
- White, R. E. 1985. *An Introduction to the Finite Element Method with Applications to Nonlinear Problems*. New York: Wiley.
- Wright, B., and J. Williams. 1984. The welfare effects of the introduction of storage. *Quarterly Journal of Economics* 99: 169–92.
- Wright, B., and J. Williams. 1982. The economic role of commodity storage. *Economic Journal* 92: 596–614.
- Wright, B., and J. Williams. 1982. The roles of public and private storage in managing oil import disruptions. *Bell Journal of Economics* 13: 341–53.
- Williams, J., and B. Wright. 1991. *Storage and Commodity Markets*. Cambridge: Cambridge University Press.
- Wilson, C. 1977. A model of insurance markets with incomplete information. *Journal of Economic Theory* 16: 167–207.
- Wilson, R. 1971. Computing equilibria of N -person games. *SIAM Journal of Applied Mathematics* 21: 80–87.
- Wilson, R. 1996. Nonlinear pricing and mechanism design. In H. Amman et al., eds., *Handbook of Computational Economics*, vol. 1. Amsterdam: Elsevier.
- Wilson, R. 1992. Computing simply stable equilibria. *Econometrica* 60: 1039–70.
- Wozniakowski, H. 1991. Average case complexity of multivariate integration. *Bulletin of the American Mathematical Society* 24: 185–94.
- Young, D. M. 1971. *Iterative Solution of Large Linear Systems*. New York: Academic Press.
- Young, D. M., and R. T. Gregory. 1988. *A Survey of Numerical Mathematics*, 2 vols. New York: Dover.
- Young, S. W. 1967. Piecewise monotone polynomial interpolation. *Bulletin of the American Mathematical Society* 73: 642–43.
- Ypma, T. J. 1995. Historical development of the Newton-Raphson method. *SIAM Review* 37: 531–51.
- Zeidler, E. 1986. *Nonlinear Functional Analysis and Its Applications*, vol 1. New York: Springer.
- Zhang, W.-B. 1988. Hopf bifurcations in multisector models of optimal economic growth. *Economic Letter* 26: 329–34.
- Zwillinger, D. 1992. *Handbook of Integration*. Boston: Jones and Bartlett.

Index

- A^{-1} , calculating of, 84–85
Accuracy checks, 592–93
Accuracy measures, 562–63
Active set methods, 127
Adaptive integration rules, 269
Adaptive SOR methods, 81
Adverse selection
 RSW model of, 130–33
 and undetermined gauges, 519–22
Agent models, heterogeneous, 603–604
Algebra, linear, 22–23, 88, 89. *See also* Linear equations
Algorithms, 22
 asynchronous, 32
 development of, 5, 16–17
 and direct vs. iterative methods, 39
 synchronous, 32
Alternating sweep method, 421
Analysis, quantitative vs. qualitative, 6–8
Analytic derivatives, 37–39
Annealing, simulated, 285, 299–301. *See also* Monte Carlo methods
Antithetic variates, 293
Approximation, 44–45, 50, 195, 247–48, 448–49.
 See also Advanced asymptotic methods;
 Regular perturbation methods; Regular perturbation methods in multidimensional systems
examples of, 228–31
finite element, 240–44, 381
interpolation, 195, 216–20 (*see also* Interpolation)
local, 195–96
and Markov chains, 364–65
multidimensional, 235–40
neural networks for, 244–47
orthogonal polynomials, 203–207
 least squares, 207–11, 222
parametric, 436–37
piecewise polynomial interpolation, 224
and projection approach, 378–81
regression as, 202–203, 222–23
shape-preserving, 231–35
software for, 247
splines, 225–28
uniform, 211–12
Arithmetic, computer, 29–31
Arrow-Debreu approach, 147, 190
 stochastic version of, 589
Asset markets, and information, 578–81
Asset-pricing model, Lucas, 573, 574–76, 581, 584
Asymptotic expansion, 517
 of integrals, 522–28
Asymptotic methods, 511, 532–33. *See also* Regular perturbation methods; Regular perturbation methods in multidimensional systems
approximation methods, 45, 447
bifurcation methods, 511–13
expansion, 517
 of integrals, 522–28
gauge functions, 516–17
and portfolio choices for small risks, 513–16
undetermined gauges method, 517–22
Asynchronous algorithms, 32
Auerbach-Kotlikoff model, 547
Augmented Lagrangian methods, 126
Automatic differentiation, 37–38, 50
Autonomous systems, global analysis of, 540
Back-propagation, 303
Back-substitution, 56
Backward error analysis, 48, 50
Banded sparse matrix methods, 61–62
Basis, choice of, 380–82, 386
Bellman equation(s), 402, 411, 412, 413, 424, 431, 433–34, 440–41, 472, 474, 498, 502, 504, 585
Bernardo, A., 580–81
Bernstein polynomials, 212
BFGS (Broyden-Fletcher-Goldfarb-Shanno) method, 114–15
Bifurcation methods of approximation, 45, 511–13
 and portfolio choices for small risks, 513–16
Bilinear interpolation, 240–42
Bisection, 147–48
 convergence for, 149–50
 stopping rules for, 149
BLAS (basic linear algebra subroutines), 88
Block coordinate direction method, 110–11
Block Gaussian algorithms, 72–73
Block iteration, 72
Boundary conditions, 469, 551–52, 558
Boundary value problem (BVP)
 for ODEs, 350–51
 perturbing of, 456–62
 and projection methods, 388–92
 two-point (2PBVP), 336
Bounded rationality, and accuracy checks, 592–93
Box-Muller method, 290
Bracketing algorithm, 94–96
Brownian motion, 407
Broyden-Fletcher-Goldfarb-Shanno (BFGS) method, 114–15
Broyden's method, 168–70
 B -splines, 227–28
Bubnov-Galerkin method, 383
Budget example, optimization methods for, 98

- Cardinal function interpolation, 218–19
 Central limit theorem, 309, 309n.2
 CGE (computational general equilibrium), 3–4, 17, 19, 187–91
 Change of variables, 255–56, 268–69
 and importance sampling, 294–96
 multivariate, 276–77
 Chebyshev approximation algorithm, 238
 Chebyshev collocation, 374, 375, 391
 Chebyshev economization, 215–16
 Chebyshev evaluation algorithm, 206–207
 Chebyshev interpolation, 221–22, 228, 230, 384
 Chebyshev least squares approximation, 209–10, 213–15
 Chebyshev minimax property, 221
 Chebyshev-Newton projection method, 559–60
 Chebyshev polynomial, 204, 381
 Chebyshev regression algorithm, 223
 Cholesky factorization (decomposition), 59–60
 Coleman, J., 601–602
 Collocation methods, 373, 384, 385
 Chebyshev collocation, 374, 391
 orthogonal collocation, 384, 590–92, 594–96
 spline collocation, 384
 Commodity storage problem or model, 401, 428–30, 581–88
 Comparative perfect foresight dynamics, 456–62
 Comparative statics, 451–53
 multidimensional, 487–88
 Comparison methods, for optimization, 93
 Competitive equilibrium, computation of, 187–91
 Complete polynomials, 239–40
 Complexity, computational, 48–49
 Composite rule, 252–53
 Computational analyses of theory, vs. deductive analyses, 8–11
 Computational economic theory, economic theory of, 12–13
 Computational general equilibrium (CGE), 3–4, 17, 19, 187–91
 Computational technology, 3
 future of, 15–17
 Computer arithmetic, 29–31
 Compute and verify procedure, 46
 Concave stopping rule, 414
 Concavity, and value iteration, 414–15
 Conditional expectation, computing of, 393–95
 Conditioning, 55
 Condition number, 67–70
 Conjugate direction methods, 116–17
 Consistency, 388
 “Consistent” capital income taxation, 568–71
 Constrained optimization, 131–32
 nonlinear, 121–27
 Consumer budget example, optimization methods for, 98
 Consumption models
 continuous-time life-cycle, 389–92
 life-cycle, 353–54
 Consumption policy function, 589, 590
 Continuation methods
 homotopy, 179–87
 simple, 176–79
 Continuous Markov chains, discrete approximations of, 87–88
 Continuous-time deterministic control, 462–71
 Continuous-time dynamic programming problems, 406–409, 440–42
 Continuous-time growth model, 355–57, 392–93, 460–61
 single-sector, 466–68
 Continuous-time life-cycle consumption models, 389–92
 Contracting, dynamic principal-agent, 10–11
 Control variates, 293
 Convergence
 for bisection, 149–50
 global, 171–74, 176
 of iterative schemes, 77–78
 radius of, 196
 rates of, 42
 Convergence tests, for Newton’s method, 151–52
 Coordinate direction method, 110–11
 Cournot duopoly game, 163
 Cramer’s rule, 60
 Cubature, 251. *See also Quadrature; Numerical integration*
 Cubic splines, 225–27
 Curse of dimensionality, 270, 430
 Dampening, 79
 Decomposition methods, 61, 70
 Cholesky decomposition, 59–60
 LU decomposition, 56–57, 69
 Deductive analyses of theory, vs. computational analyses, 8–11
 den Haan-Marcel (dHM) procedure, 600–601, 601n.6
 Dense matrices, 55
 Derivatives
 analytic, 37–39
 efficient computation of, 35–39
 Deterministic control
 continuous-time, 462–71
 discrete-time, 474–76
 Deterministic growth model, 403, 424–28, 476–77
 Deterministic-vs.-random distinction, 310
 Diagonal dominance, 77

- Differential equations
ordinary (ODE), 335–36, 350–51, 369–75, 376
partial, 375–77
and perturbation, 471
system of, 335, 345–46
- Differentiation
automatic, 37–38, 50
numerical, 279–82
- Direction set methods, 109–17
- Direct methods, 39, 70
- Discrepancy, 314–16
- Discrete-time BVPs, perturbing of, 461–62
- Discrete-time dynamic programming problems, 399–406
- Discrete-time problems, perturbations of, 474–80, 496, 502–504
- Discretization methods, for continuous-state problems, 424–30
- Distributed computing, 32
- Dorsey, R., 297, 299
- Double precision, 30
- Duopoly example, 162–65
- Dynamic contracts, 10–12
- Dynamic economic models
linearization procedures for, 507
tax and monetary policy in, 563–67
- Dynamic iteration methods, 553–58
- Dynamic models
of equilibrium, 537
in evaluation of empirical analyses, 546–47
growth (stochastic), 588–89
- Dynamic optimization problem, 140–42
- Dynamic programming, 4, 399, 442–43
acceleration methods for infinite-horizon problems, 415–24
and computational strategies, 430
continuous methods for continuous-state problems, 433–36
continuous-time problems in, 406–409, 440–42
discrete-time problems in, 399–406
discretization methods for continuous-state problems, 424–30
finite-state methods, 399, 409–15
and linear-quadratic problems, 431–33
parametric approximations, 436–37
shape-preserving methods, 437–40
simulation methods, 437
special structures for, 415
- Dynamic recursive models, 447
- Dynamic systems, multidimensional, 490–96
- Economic analysis
roles of computation in, 6–13
and uncertainty, 511
- Economic equilibrium, computation of (literature), 193
- Economic models, computation of, 3
- Economic theory
of computational economic theory, 12–13
and mathematics, 8
- Efficient polynomial evaluation, 34–35
- Einstein summation notation, 488
- Envelope theorem, 453
- Equidistributed sequences, 311–13
- Equilibrium. *See also* General equilibrium
competitive, 187–91
dynamic models of, 537
economic (computation), 193
monetary, 577–78
in OLG models, 540–47
operator representation of, 550–51
signaling, 347–49
- Equioscillation property, 212
- Ergodic distributions, 85–88
and Markov chains, 364–65
- Error analysis, 66–70
- Error bounds, 45–46, 67–70, 413–14
- Errors, 39–40
in final result, 45–48
propagation of, 41, 50
- Euler equations, 476, 551–52, 553, 584–85, 594, 595
with uncertainty, 478–79
- Euler homotopy method, generically convergent, 185
- Euler's method, 341–43
implicit, 343–44, 345–46, 349
- Executive compensation, 7, 130
- Expanded Chebyshev array, 222
- Extrapolation, 78–80, 166
- Fair-Taylor method, 547–49
- Fibonacci generator, 289
- Finite difference methods, 36, 335, 365–66
and boundary value problems for ODEs, 350–51
and economic examples of IVPs, 346–49
and finite-horizon optimal control problems, 351–54
infinite-horizon optimal control and reverse shooting, 355–61
for initial value problems, 340–46
integral equations, 362–65
and linear dynamic systems, 337–40
and ordinary differential equations, 335–36
and projection methods, 369, 370
- Finite element approximation, 240–44, 381
- Finite-horizon optimal control problems, 351–54

- Finite-horizon problems in dynamic programming, 399–401, 409–11
 Finite Markov chains, 85–87
 First-order ordinary differential equation (ODE), 335
 Fixed-point iteration, 71, 159, 165–67, 188, 544–45, 555–56, 585, 599–601
 and operator splitting approach, 75, 76
 and stability, 557–58
 Forward error analysis, 45–46, 50
 Fourier analytic methods, 321–25
 Fourier approximation, 208–209
 Fredholm equations, linear, 362, 363–64, 575
 Function iteration, 165
 Functions and functional operators, 24
 Galerkin method, 373, 560–61, 590–92, 594–96
 Galerkin-Petrov method, 383
 Gaspar, J., 604
 Gauge functions, 516–17
 Gauges, undetermined, 517–22
 Gaussian approach, and numerical problem, 435
 Gaussian elimination, 55–58, 69
 Gaussian formulas, 257–59
 Gauss-Chebyshev quadrature, 259–60
 Gauss-Hermite quadrature, 261–63, 600
 Gauss-Laguerre quadrature, 263–64
 Gauss-Legendre quadrature, 260–61
 interpolatory quadrature rules, 265–67
 Gaussian iteration methods, 417–18
 Gauss-Jacobi algorithm, 71, 73–75, 78, 160–61, 164, 418
 linearized, 161, 165
 Gauss-Newton approach, 119, 139–40
 Gauss-Seidel algorithm, 72, 73–75, 78, 161–62, 164–65
 linearized, 161, 165
 Gauss-Seidel method, 418, 436
 and reverse shooting, 549
 upwind (UGS), 418–23, 427–28, 436
 General equilibrium. *See also* Equilibrium
 Arrow-Debreu model of, 147, 190
 computational (CGE), 3–4, 17, 19, 187–91
 General equilibrium example, with limited domain problem, 153–58
 General equilibrium theory, qualitative results in, 6
 Generically convergent Euler homotopy method, 185
 Generic direction method, 109
 Genetic algorithms, 285, 297–99. *See also* Monte Carlo methods
 Dorsey-Mayer implementation, 299
 Gibbs sampling, 290–91
 Global convergence
 continuation method for, 176
 enhancing of, 171–74
 Global optimization, 99
 Global quality test, for asymptotic approximations, 482–84
 Good lattice points, 325–28
 Gradient methods
 for optimization, 93–94
 reduced, 126–27
 Gradients, computing of, 108
 Grid search, 100
 Gross substitutability, 78
 Growth model
 deterministic, 403, 424–28, 476–77
 optimal, 476–77
 continuous-time, 355–57, 392–93, 460–61, 466–68
 single-state, 424–28
 stochastic, 477–80, 504–508
 dynamic, 588–89
 with tax distortions, 602–603
 Guesses, initial, 385–86, 558, 593–94
 Gustafson, R., 429, 586
 Halton sequence, 316
 Hamilton-Jacobi-Bellman (HJB) equation, 408
 Hammersley set, 316
 Haubrich, J. G., 130
 Hermite integrals, 523–24
 Hermite interpolation, 217–18, 220
 shape-preserving, 440
 Hermite interpolation polynomials, 224
 Hermite polynomials, 204, 211
 Hermite spline, 226
 Hessian, 21
 computation of, 107–108
 Hessian update algorithms, 115
 Heterogeneous agent models, 603–604
 Hidden-layer activation function, 244
 Hierarchical numerical methods, 189
 Hilbert matrices, 69
 Hlawka theorem, 318–19, 324
 Homotopy continuation methods, 179–87
 Homotopy functions, 179
 Homotopy solution methods, 349
 Horner's method, 34–35
 Hybrid genetic method, 299
 Hybrid perturbation-projection methods, 528–32
 Ill-conditioned problems, 70
 Implicit Euler method, 343–44, 345–46, 349
 Implicit function theorem (IFT), 447, 448, 487, 508, 509, 511

- Importance sampling, 293–96
Impulse response function, 493
Incentive problems, 7, 128–33
Infinite-horizon optimal control, 355–61
Infinite sequences, heuristics for, 41–44
Information, and asset markets, 578–81
Initial guesses, 385–86, 558, 593–94
Initial value problem (IVP), 335–36
 economic examples of, 346–49
 finite-difference methods for, 340–46
 perturbing of, 453–56
Integral equations, 362–65
Integrals, asymptotic expansions of, 522–28
Integration, numerical. *See* Numerical integration
Integration, and variation, 316–19
Integration methods, 331
Interpolation, 195, 216–20
 bilinear, 240–42
 Chebyshev, 221–22, 228, 230, 384
 Lagrange, 216–17, 236–37
 linear, 231, 242–43, 439
 multidimensional
 linear, 242–43
 simplicial, 243–44
 multilinear, 439–40
 piecewise-linear, 231
 piecewise polynomial, 224
 shape-preserving quadratic spline, 231–33
 simplicial 2-D linear, 242
Interpolation error, 220–21
Interpolatory quadrature rules, 265–67
Inverse demand equation, 73–75
Iterative methods (schemes), 39, 70–75
 block, 72
 convergence of, 77–78
 fixed-point, 71, 75, 76, 159, 165–67, 188, 544–45,
 555–58, 585, 599–601
 general, 77
 Newton, 589–99
 and operator splitting, 76
 time, 553–55, 583–85, 601–602
- Jackson's theorem, 213, 214
Jordan canonical form, 337–38
Jordan decomposition, 23
Jump process, 407
Jump process control problems, perturbing of,
 480–82
Jupiter's Red Spot, 13–14
- Keast good lattice point, 328
Koksma's theorem, 317–18
Kolmogorov's theorem, 236, 246
Korobov good lattice point, 327–28
Korobov's theorem, 323
- Kubler, F., 604
Kuhn-Tucker approach, 122–23, 127
Kydland, F., 506–508
- Lagrange data, 216
Lagrange interpolation, 216–17, 236–37
Lagrange interpolation polynomial, 220
Lagrangian, augmented, 126
Laguerre polynomials, 204, 211
LAPACK, 88
Laplace's approximation, 524–27
Lattice points
 good, 325–28
 uniformly distributed, 315
Law of large numbers (LLN), 291–92, 309, 309n.2
Least squares approximation, Chebyshev, 213–15
Least squares method or problem, 371, 385
 nonlinear, 117–19
Least squares orthogonal polynomial
 approximation, 207–11, 222
Least squares projection method, 382
Least squares regression, 202
Legendre approximation, 214
Legendre polynomials, 211, 377
Levenberg-Marquardt algorithm, 119
l'Hospital's rule, 358, 511–12
Li, J., 566
Life-cycle model
 of consumption and labor supply, 353–54
 continuous-time consumption, 389–92
Limited domain problem, general equilibrium
 example with, 153–58
Linear algebra, 22–23. *See also* Linear equations
Linear approximation, 449
 multivariate, 487–88
Linear congruential method (LCM), 287–88
Linear dynamic systems, solution of, 337–40
Linear equations, 55, 89
 acceleration and stabilization methods for, 78–
 84
 and calculation of A^{-1} , 84–85
 Cholesky factorization for, 59–60
 and computing of ergodic distributions, 85–88
 Cramer's rule for, 60
 and error analysis, 66–70
 Gaussian elimination and LU decomposition for,
 55–58
 iterative methods for, 70–75
 convergence of, 77–78
 matrix methods for
 banded sparse, 61–62
 general sparse, 62–66
 operator splitting approach for, 75–77
 QR factorization for, 58–59
 software for, 88

- Linear Fredholm equations, 363–64
 of the first kind, 362
 of the second kind, 362, 363
- Linear Fredholm integral equation, 575
- Linear(ized) Gauss-Jacobi method, 161, 165
- Linear(ized) Gauss-Seidel method, 161, 165
- Linear interpolation, 242–43, 439
 piecewise, 231
- Linearization methods, 539
- Linearization of multidimensional dynamic systems, 490–96
- Linear programming, 120–21, 143, 423–24
- Linear-quadratic problems, 431–33, 506–507
- Linear Volterra equations of first and second kind, 362
- Line search, Newton's method with, 112–13
- Local approximation methods, 195–202
 rational, 197–200
 Taylor series, 196–97, 505
- Log-linearization, 200–201
- Log-quadraticization, 201
- Low-discrepancy methods, 313–21
- L^p approximation, 195
- Lucas asset-pricing model, 573, 574–76, 581, 584
- LU decomposition, 56–57, 69
- Lyapunov approach, 134
- Machine epsilon, 30
- Machine infinity, 30
- Machine zero, 30
- Magill, M., 506–508
- Managerial incentives, 7, 130
- Markov chains, 364–65
 finite, 85–87
- Markov process, continuous-space and finite-state, 576
- Mathematical truncation, 40
- Matrices
 inverse of, 84
 notation for, 20–21
- Matrix analysis, 66
- Matrix methods
 banded sparse, 61–62
 general sparse, 62–66
- Mayer, W., 297, 299
- McGrattan, E., 507–508
- McKelvey, R., 133
- Method of moments, 372, 384
- Midpoint rule, 252–53
- Minimax approximation, 212–13
- Minimax property, Chebyshev, 221
- Minimization, 93n.1
 one-dimensional, 94–99
- Modified policy iteration with k steps, 416–17
- Moments, method of, 372, 384
- Monetary equilibrium, 577–78
- Monetary models, local analysis of, 565–66
- Monomial formulas, 271–76
- Monomial rules, 331
- Monopoly example, and optimization, 105–108
- Monotonicity, and value iteration, 414–15
- Monte Carlo integration, 269–70
- Monte Carlo methods, 285
 crude, 292
 low-discrepancy methods, 313–21
- Monte Carlo integration, 291–96
 and numerical problem, 435
 and one-dimensional integrals of smooth functions, 318
 optimization by stochastic search, 296–301
 and projection methods, 394, 395
 pseudorandom number generation, 285–91
 and quasi-Monte Carlo methods, 309, 319–21
(see also Quasi-Monte Carlo methods)
- standard optimization methods with simulated objectives, 303–305
 stochastic approximation, 301–303
- Moore's law, 15
- Multidimensional approximation, 235–40
- Multidimensional comparative statics, 487–88
- Multidimensional control, locally asymptotically stable, 496–501
- Multidimensional interpolation
 linear, 242–43
 simplicial, 243–44
- Multidimensional optimization, 99–102
- Multidimensional quadrature, 269–77
- Multidimensional systems, regular perturbations in. *See* Regular perturbations in multidimensional systems
- Multilinear interpolation methods, 439–40
- Multiple prime random number generator (MPRNG), 288–89
- Multisector stochastic growth, 504–508
- Multivariate calculus, tensor-style, 490
- Multivariate changes of variables, 276–77
- Multivariate problems, Newton's method for, 103–108, 167–71
- Mysovskikh theorem, 273–74
- Nash equilibrium(a), 5
 computing of, 133–135, 143
 as nonlinear equation problem, 147
- Natural spline, 225–26
- Near-diagonal methods, 61–62
- Near-minimax approximation, 213–15
- Negishi method, 190–91
- Neuman's lemma, 84–85

- Neural network learning, 303
Neural networks, 244–47
Newton-Cotes formulas, 251–57
Newton iteration, projection methods with, 589–99
Newton's method, 96–97, 98, 143
 in econometric example, 138–39
 homotopy methods combined with, 187
 with line search, 112–13
 modified, 107
 multivariate, 103–108, 167–71
 and numerical problem, 435
 for one-dimensional problems, 150–58
 and perfect foresight models, 545, 559, 571
 and quasi-Newton methods, 113–14
 and transformations, 175–76
Nonautonomous perturbations of perfect foresight model, 457–60
Nonlinear equations, 147, 192–93
 and advantageous transformations, 174–76
 and global convergence, 171–74
 and homotopy continuation methods, 179–82
 multivariate
 elementary methods for, 159–67
 Newton's method for, 167–71
one-dimensional problems
 bisection, 147–50
 Newton's method, 150–58
 special methods for, 158–59
 and perfect foresight models, 545
 recursive models with, 558–62
 and simple CGE problem, 187–91
 and simple continuation method, 176–79
 software for, 191–92
Nonlinear least squares, 117–19
Nonlinear optimization, constrained, 121–27
Nonlinear perfect foresight competitive models.
 See Perfect foresight models
Nonlinear stochastic rational expectations models, 4–5
Nonuniform random number generation, 289–90
Nuclear weaponry, development of, 14–15
Number theoretic methods, 310
Numerical differentiation, 279–82
Numerical dynamic programming. *See* Dynamic programming
Numerical integration, 251, 282
 adaptive quadrature, 269
 Gaussian formulas, 257–67
 multidimensional quadrature, 269–77
 Newton-Cotes formulas, 251–57
 and portfolio problems, 277–79
 singular integrals, 267–69
 software for, 282
Numerical linear algebra, monographs on, 89
Nyström extension, 364
One-dimensional minimization, 94–99
One-dimensional problems, 150–58, 337
Operator representation of equilibrium, 550–51
Operator splitting approach, 75–77
Optimal control problems
 finite-horizon, 351–54
 infinite-horizon, 355–61
Optimal growth models, 476–77
 continuous-time, 355–57, 392–93, 460–61, 466–68
 single-state, 424–28
Optimal policy function, 400
Optimization, 93–94, 142–43
 computing Nash equilibria, 133–35, 143
 constrained, 131–32
 constrained nonlinear, 121–27
 direction set methods, 109–17
 dynamic optimization problem, 140–42
 global, 99
 in incentive problems, 128–33
 linear programming, 120–21, 143
 multidimensional, 99–102
 Newton's method for multivariate problems, 103–108, 143
 and nonlinear equations, 171–73
 nonlinear least squares, 117–19
 one-dimensional minimization, 94–99
 in portfolio problem, 135–37
 in simple econometric example, 137–40
 software for, 142
 standard optimization methods with simulated objectives, 303–305
 by stochastic search, 296–301
Optimization rules, 415
Order of convergence notation, 23
Ordinary differential equations (ODE)
 boundary value problems for (shooting), 350–51
 classification of, 335–36
 initial conditions for, 376
 and projection method, 369–75
Ordinary least squares regression, 202
Orthogonal collocation method, 384, 590–92, 594–96
Orthogonal polynomials, 203–207
 least squares, 207–11, 222
Overflow, 30
Overidentification approach, 387
Overidentified systems, 88, 174
Overlapping generations (OLG) model
 equilibrium in (time domain methods), 540–47
 recursive solution of, 567–68
 simple autonomous, 538–40
Overrelaxation, successive, 80–81

- Padé approximation, 197–200, 235
 Padé expansions, 202, 484
 Parallel processing, 32
 Parametric approximations, 436–37
 Parametric dynamic programming algorithm, 434–36
 Parametric path following, 182–83
 Parametric path method, 545–46
 Partial differential equation, 375–77
 Path following, parametric, 182–83
 Pathological examples, for Newton's method, 153
 Penalty function method, 123–25
 Perfect foresight dynamics, comparative, 456–62
 Perfect foresight models, 4, 537, 571
 and accuracy measures, 562–63
 “consistent” capital income taxation, 568–71
 dynamic iteration methods for, 553–58
 and equilibrium in OLG models (time domain method), 540–47
 Fair-Taylor method, 547–49
 nonautonomous perturbations of, 457–60
 recursive, 549–53
 with nonlinear equation methods, 558–62
 and recursive solution of OLG model, 567–68
 simple autonomous overlapping (OLG) model, 538–40
 and tax or monetary policy in dynamic economics, 563–67
 Periodic function, 209
 Periodization, 324–25
 Perturbation methods of approximation, 29, 44–45, 447. *See also* Asymptotic methods; Regular perturbation methods; Regular perturbation methods in multidimensional systems
 Perturbation-projection methods, hybrid, 528–32
 Piecewise-linear interpolation, 231
 Piecewise polynomial interpolation, 224
 Pivoting, 58
 Policy function
 consumption, 589, 590
 optimal, 400
 Policy function iteration method, 416–17
 Policy function representation, 552–53
 Policy iteration, 426–27
 Polynomial evaluation, efficient, 34–35
 Polynomial interpolation, piecewise, 224
 Polynomial methods, in dynamic programming, 439
 Polynomials
 complete, 239–40
 orthogonal, 203–207
 least squares, 207–11, 222
 Polytope methods, 100–102, 296
 Portfolio applications, of Laplacian approximation, 524–27
 Portfolio choices, for small risks, 513–16
 Portfolio problems or theory, 135–37, 277–79
 alternative gauges in, 518–19
 Gauss-Hermite quadrature in, 262–63
 and Monte Carlo integration, 304–305
 Powell's hybrid method, 173–74
 Power series method, 374
 Preconditioning, 70
 Predetermined variables, 338
 Pre-Gauss-Jacobi iteration, 418
 Pre-Gauss-Seidel method, 418
 Prescott, E., 506–508
 Principal-agent problems, 128–30
 Probabilistic arguments, 309
 Processing mode, serial, 31
 Production possibility set, 33
 Product rules, 270
 Projection methods, 369, 395–96
 and boundary value problems, 388–92
 Chebyshev-Newton, 559–60
 computing of conditional expectations, 393–95
 continuous-time growth model, 392–93
 for continuous-time problems, 441
 existence problems, 387–88
 general, 377–88
 with Newton iteration, 589–99
 and ordinary differential equation, 369–75
 and partial differential equation, 375–77
 in perturbation-projection hybrid, 528–32
 and policy functions, 552
 Pseudo-Monte Carlo (pMC) methods, 309
 Pseudorandom number generation, 285–91
 Pseudorandom sequences, 309–10
 and equidistributed sequences, 313

 QR factorization, 58–59
 Quadrature, 251. *See also* Numerical integration
 adaptive, 269
 Gaussian, 257–58, 265
 Gauss-Chebyshev, 259–60
 Gauss-Hermite, 261–63, 600
 Gauss-Laguerre, 263–64
 Gauss-Legendre, 260–61
 interpolatory rules of, 265–67
 multidimensional, 269–77
 Newton-Cotes, 251–57
 Qualitative analysis, vs. quantitative analysis, 6–8
 Quasi-Monte Carlo (qMC) methods, 309–11,
 330–31
 and acceleration methods, 330
 equidistributed sequences, 311–13
 estimating errors in, 329–30
 Fourier analytic methods, 321–25
 low-discrepancy methods, 313–21
 method of good lattice points, 325–28

- and Monte Carlo methods, 309, 319–21 (*see also* Monte Carlo methods)
and numerical problem, 435
and one-dimensional integrals of smooth functions, 318
and projection methods, 395
Quasi-Newton methods, 113–14
Quasi-random search, 321
Quirmbach, H., 11
- Radical-inverse function, 315–16
Radius of convergence, 196
Radon seven-point formula, 272–73
Random-vs.-deterministic distinction, 310
Randomization of quasi-Monte Carlo rule, 329
Random number generation
nonuniform, 289–90
uniform, 287–89
Random search, optimization by, 296
Random variables, 285–86
Rational approximation, 197–200
Rational expectations models, 573, 605–606
and accuracy checks, 592–93
commodity storage models, 581–88
and fixed-point iteration, 599–601
generalizations of, 602–605
and information in asset markets, 578–81
Lucas asset-pricing model, 573, 574–76, 581, 584
and monetary equilibrium, 577–78
nonlinear stochastic, 4–5
projection methods with Newton iteration, 589–99
simple stochastic dynamic growth model, 588–89
and time iteration, 601–602
Rational function, 197
Rayleigh-Ritz method, 383
Recursion formulas, 206
Recursive approaches and models, 537, 549–53
with nonlinear equation methods, 558–62
Recursive solution, of OLG model, 567–68
Reduced gradient methods, 126–27
Regression, 195
and approximation, 202–203, 222–23
and projection approach, 369
Regular perturbation methods, 447–51, 511
comparative statics, 451–53
multidimensional, 487–88
continuous-time deterministic control, 462–71
and discrete-time systems, 474–80
global quality test for asymptotic approximations, 482–84
and implicit function theorem, 447, 448, 508, 509, 511
and perturbation-projection hybrid, 528–32
- perturbing of BVP, 456–62
perturbing of IVP, 453–56
perturbing of jump process control problems, 480–82
stochastic control, 471–74
and uncertainty, 508
Regular perturbations in multidimensional systems, 487
of discrete-time problems, 502–504
linearization of multidimensional dynamic systems, 490–96
locally asymptotically stable multidimensional control, 496–501
multidimensional comparative statics, 487–88
and multisector stochastic growth, 504–508
and tensor notation, 488–90, 497–98
Residual function, 371
Reverse shooting, 335, 355–61
and Fair-Taylor method, 549
multidimensional, 360–61
Riccati system or equations, 4, 431–32 503
Richtmeyer's theorem, 323–24
Risks, and portfolio choices, 513–16
Rounding, 40
Runge-Kutta method, 344–45, 346
fourth-order, 345
- Sampling
importance, 293–96
stratified, 292–93
Sampling method, 310
Samuelson, P., 515
Scaling, 99
Schema theorem, 298
Schmedders, K., 604
Schumaker quadratic shape-preserving spline procedure, 440
Schumaker shape-preserving interpolation, 233–35
Scientific notation, 20
Secant Hermite spline, 226
Secant method, 158–59
Secant method (Broyden), 168–70
Sequential quadratic method, 125–26
Serial processing mode, 31
Shape-preserving approximation, 231–35
Shape-preserving dynamic programming methods, 437–40
Shocks
equilibrium response to, 457
perturbation method analysis of, 564
Shooting methods, 335, 350–51
forward, 357–60
life-cycle, 354
reverse, 335, 355–61, 549

- Sigmoid function, 246
 Signaling equilibrium, 347–49
 Simplex method, 120–21
 Simplicial homotopy methods, 185–87
 Simplicial interpolation, multidimensional, 243–44
 Simplicial 2-D linear interpolation, 242
 Simpson’s rule, 253–55
 Simulated annealing, 285, 299–301. *See also*
 Monte Carlo methods
 Simulated objectives, standard optimization
 methods with, 303–305
 Simulated upwind Gauss-Seidel algorithm (SUGS),
 422–23
 Simulation methods, 437
 in den Haan-Marcret (dHM) procedure, 600–601
 and Monte Carlo or quasi-Monte Carlo methods,
 20
 Single precision, 30
 Single-state optimal growth, 424–28
 Singular integrals, 267–69
 Sobol sequence, 331
 Social Security, and savings, 546–47
 Sparse Markov transition matrix, 64–65
 Sparse matrices, 55
 Sparse matrix methods
 banded, 61–62
 general, 62–66
 Spectral condition number, 68
 Spectral methods, 381
 Spline collocation methods, 384
 Spline interpolation, shape-preserving, 231–33
 Splines, 225–28
 multidimensional, 237
 Schumaker shape-preserving, 440
 Squashing function, 245–46
 Stability, and fixed-point iteration, 557–58
 Stabilization methods, 78–84
 State variable, 401
 Statics, comparative, 451–53, 487–88
 Steady states, of overlapping generations models,
 539
 Steepest descent method, 111–12
 Stigler, George, 14n.3
 Stirling’s formula, 527–28
 Stochastic accumulation problems, 405–406
 Stochastic approximation, 285, 301–303. *See also*
 Monte Carlo methods
 Stochastic control, 471–74
 Stochastic dynamic programming, 408–409
 Stochastic growth model, 477–80, 588–89
 multisector, 504–508
 with tax distortions, 602–603
 Stochastic rational expectations models, nonlinear,
 4–5
 Stochastic search, optimization by, 296–301
 Stochastic transition problems, 406
 Stockouts
 equilibrium with, 585–86
 equilibrium without, 582–83
 Stone’s theorem, 235
 Stone-Weierstrass theorem, 381
 Stopping rules, 42–44
 for bisection, 149
 concave, 414
 for multivariate systems, 162
 and optimization, 97, 104–105
 Storage scheme for general sparse matrices, 62–64
 Stratified sampling, 292–93
 Subdomain method, 372, 384
 Successive approximation, 165
 Successive overrelaxation, 80–81
 Successive substitution, 165
 Sufficiency problems, 551–52
 Symbolic methods, of approximation, 44
 Symbolic software, 38, 533
 Synchronous algorithms, 32
 Tatonnement process, 73–75, 81, 542–44
 Taxation, 563–65
 capital income, 568–71
 Tax distortions, growth with, 602–603
 Taylor expansions, 511, 522
 for general optimal control problems, 498–501
 and global quality test, 484
 and perturbation, 475
 Taylor series approximation, 196–97, 505
 Taylor series method, 375
 Taylor’s theorem, 23–24, 239, 447
 Tchakaloff theorem, 273–74
 Tensor notation, 488–90, 497–98
 Tensor product bases, 237–38, 239
 Tensors, 487
 Tensor-style multivariate calculus, 490
 Theory, deductive vs. computational analyses of,
 8–11
 Theory without theorems, 11–12
 Thin tails problem, 294
 Three-point formulas, general, 281–82
 Time domain methods, 537
 and equilibrium in OLG models, 540–47
 Time iteration, 553–55, 583–85, 601–602
 Transcritical bifurcation, 513
 Transformations, advantageous, 174–76
 Transition matrix, sparse Markov, 64–65
 Trapezoid rule, 253, 344
 Triangular matrices, 55–57
 Truncation, mathematical, 40
 Two-sided differences, 281–82

Underflow, 30
Undetermined gauges, method of, 517–22
Uniform approximation, 211–12
Uniform random number generation, 287–89
Units, importance of, 152–53
Upwind Gauss-Seidel method (UGS), 418–22,
 427–28, 436
 simulated (SUGS), 422–23

Value function, 400–401
Value function iteration, 412–13
 for infinite-horizon problems, 415–16
 maximization step in
 and adaptive control choices or optimization
 rules, 415
 and concavity or monotonicity, 414–15
Value iteration, 425–26
van der Corput sequence, 316
Vandermonde matrix, 217
Variables
 change of, 255–56, 268–69
 and importance sampling, 294–96
 multivariate, 276–77

Variates
 antithetic, 293
 control, 293

Variation, and integration, 316–19

Vector processors, 31

Vectors, notation for, 20–21

Volterra equations, 362

Volterra integral equation, 362

Wealth accumulation examples, 403–405, 407–408

Weierstrass theorem, 211–12, 235, 370

Weighted residual methods, 379–80. *See also*
 Projection methods

Welfare economics, first theorem of, 190

Williams, J., 586–88

Wright, B., 586–88

Wright-Williams smoothing, 586–88