ELSEVIER Journal of Economic Dynamics & Control 26 (2002) 1557–1583

JOURNAL OF Economic Dynamics & Control

www.elsevier.com/locate/econbase

The parametric path method: an alternative to Fair–Taylor and L–B–J for solving perfect foresight models

Kenneth L. Judd*

Hoover Institution and NBER, Stanford University, Stanford, CA 94305, USA

Abstract

The parametric path method applies projection methods to compute the equilibrium time path of economic variables in infinite-horizon dynamic models. We exploit the special structure of equilibrium paths common in such models to construct a low-dimensional set of candidate solutions, and then use efficient integration and equation solution methods to find an approximate solution. Simple illustrative examples show that the parametric path method can find excellent approximations with little computational cost. © 2002 Elsevier Science B.V. All rights reserved.

JEL classification: C63; C68

Keywords: Perfect foresight models; Numerical methods; Projection methods

0. Introduction

Large-scale dynamic general equilibrium models are increasingly used in analyses of economic problems. However, their use is limited by the numerical difficulty of solving such models. The perfect foresight aspect of dynamic general equilibrium analysis creates links between current and future economic variables. This simultaneity generates an infinite system of nonlinear equations. Typically, we solve these infinite-dimensional systems by truncating them to form a finite but large system of nonlinear equations. If x_t is a sequence of endogenous economic variables to be computed, conventional methods try to solve for each x_t up to some large T. Even these truncated systems of equations are too large to be solved by conventional computational general equilibrium procedures like Scarf's algorithm or homotopy procedures. Economists then resort to general methods for large systems of equations, but they make limited use of the

* Tel.: +1-650-723-5866; fax: +1-650-723-1687.
E-mail address: judd@hoover.stanford.edu (K.L. Judd).

special structure of dynamic models. This paper introduces the method of parametric paths for solving perfect foresight models. The parametric path method chooses a functional form $x_t = \Phi(t;a)$ and then finds some value for the coefficient vector a which produces an acceptable approximation for the x_t path. We use a priori known properties of the equilibrium path x_t to choose a functional form $\Phi(t;a)$ which can approximate the true path with a low-dimensional vector of parameters a. This reduces an infinite-dimensional problem to one of small dimension which can then be solved by a wide variety of methods.

Previous methods break into three groups. First, Gauss–Jacobi schemes break down the system of equations into smaller blocks and then solves each of them in sequence. Fair and Taylor (1983), Fisher et al. (1986) and Hughes Hallett and Piscitelli (1998) fall into this class. A second approach is contained in Auerbach et al. (1987). They formulate equilibrium as a fixed point $p = D^{-1}(S(p))$ where $S(p)$ is the sequence of factor supplies if p is the sequence of factor prices, and $D^{-1}(p)$ is the inverse demand function. They execute the iteration $p^{k+1} = D^{-1}(S(p^k))$, which is essentially a "hog-cycle" process: make a guess for prices, compute the supply of goods at those prices, and then compute the inverse demand function at that supply. Both the Gauss–Seidel and the hog-cycle approach are intuitive methods but are slow and may not converge. These methods are examples of first-order methods since they use only the values of $f(x)$ in solving $f(x) = 0$ and, at best, converge linearly. As is normal with first-order methods, convergence problems often force users to rely on ad hoc dampening parameters.

More recently, Laffargue (1990), Boucekkine (1995), Juillard (1996) and Juillard et al. (1998) have developed a Newton-style method (called the L–B–J method) for solving perfect foresight problems. Newton methods use the Jacobian of $f : \mathbb{R}^n \to \mathbb{R}^n$ to solve $f(x) = 0$ and are second-order methods since they converge quadratically near the solution. Newton methods are generally impractical since computing the Jacobian of f requires n^2 derivatives, an impossible amount of work if n is large. However, the Jacobian for many perfect foresight models is sparse, allowing for efficient computation of the Jacobian. The L–B–J method exploits this sparseness and is substantially faster and more reliable than first-order methods. Another recent contribution by Mercenier and Michel (1994a) uses a time aggregation scheme which chooses a nonuniform set of time intervals and a finite-horizon approximation so that the terminal behavior of the finite-horizon model matches the steady-state behavior of the original infinite-horizon model.

We propose an algorithm which uses standard methods from numerical functional analysis and exploits the special structure of many dynamic general equilibrium models. While there are an infinite number of unknowns in an infinite-horizon general equilibrium model, the dynamic path is often relatively well-behaved. Specifically, dynamic general equilibrium analyses generally assume convergence to a steady state, or, more generally, convergence to some known (or easily computed) dynamic path. During some initial phase, that convergence need not be well-behaved but asymptotically the convergence is governed by linear approximations about the asymptotic path. The idea of the parametric path method is to express the time path of economic variables as some function of time where the number of free parameters in the parameterization is far smaller than the number of unknown prices and quantities in the infinite-horizon economic model. It is common in numerical rational expectations models to parameterize the critical unknown functions; see Gustafson (1958), Wright and Williams (1982), Wright and Williams (1984), and Miranda and Helmberger (1988) for the seminal contributions to this literature, and Judd (1992) for generalizations of those ideas that incorporate efficient methods from numerical analysis. As long as the asymptotic behavior of equilibrium is well-behaved, it is possible to construct flexible and parsimonious parameterizations which can accurately approximate equilibrium. This reduction in the number of unknowns substantially reduces the complexity of the numerical problem since the cost of solving methods for nonlinear equations is quadratic (at best) in the number of unknowns.

We first describe the general perfect foresight model and traditional solution methods. We then use the projection method (see Judd, 1992) to develop the parametric path method for solving perfect foresight models. We finish with a detailed application to a familiar perfect foresight model. Even though the basic idea is simple, there are only a few ways to implement it successfully. We use a simple application to highlight the critical problems and indicate their solutions. Even though the example is simple, it illustrates important problems which will come up in any application. Implementing the parametric path method for large systems will probably present more problems, but space limitations require us to leave that for further study.

1. Traditional perfect foresight model solution methods

Let $x_t \in \mathbb{R}^n$ be a list of the time t values for economic variables such as consumption, labor supply, capital stock, output, prices, interest rates, wages, etc., and $z_t \in \mathbb{R}$ a list of exogenous variables, such as productivity levels, tax rates, monetary growth rates, etc., at time t. Perfect foresight models[1] have the form

$$g(t,x,z) = 0, \quad t = 0,1,2,\ldots, \tag{1}$$

$$x_{i,0} = \bar{x}_{i,0}, \quad i = 1,2,\ldots,n_I, \tag{2}$$

$$\sup_t \|x_t\| < \infty, \tag{3}$$

where $x \equiv (x_0,x_1,x_2,\ldots,x_s,\ldots)$, $z \equiv (z_0,z_1,z_2,\ldots,z_s,\ldots)$, and $g(t,x,z):\mathbb{R} \times \mathbb{R}^{n\times\infty} \times \mathbb{R}^{\infty} \to \mathbb{R}^n$ is a collection of n functions concerning supply, demand, expectations, or other equilibrium relations among the economic variables. The equations in (1) represent Euler equations, market clearing conditions, and any other equations in the definition of equilibrium. Some of the economic variables may have fixed predetermined values at $t=0$. These initial conditions are represented by the $n_I < n$ conditions in (2). We also want only solutions which are bounded, the requirement in (3). The objective

[1] We examine only discrete-time models since that is the focus of most of the literature. The application of the parametric path method to continuous-time models is a straightforward adaptation of our discrete-time presentation.

is to find a sequence of values for x_t which satisfy the equation in (1), the initial conditions in (2), and the boundedness conditions in (3).

We use a simple example to illustrate our analysis. Consider the optimal growth problem

$$\begin{aligned} \max_{c_t} \quad & \sum_{t=0}^{\infty} \beta^t u(c_t), \\ s.t. \quad & k_{t+1} = F(k_t) - c_t, \\ & k_0 = \bar{k}_0. \end{aligned} \tag{4}$$

The solution to this problem is the solution to the Euler equations

$$u'(c_t) = \beta u'(c_{t+1}) F'(k_{t+1}), \quad t = 0, 1, \ldots$$

together with the boundedness condition $\lim_{t\to\infty} |k_t| < \infty$. In the notation of (1,2), the growth problem can be expressed as

$$\begin{aligned} & g_1(t,c,k) \equiv u'(c_t) - \beta u'(c_{t+1}) F'(k_{t+1}) = 0, \quad t = 0, 1, 2, \ldots, \\ & g_2(t,c,k) \equiv k_{t+1} - F(k_t) + c_t = 0, \quad t = 0, 1, 2, \ldots, \\ & k_0 = \bar{k}_0. \end{aligned} \tag{5}$$

In this problem, $x_t \equiv (c_t, k_t)$ and there are no exogenous variables. The functions in $g(t,c,k)$ are the time t Euler equation and time t gross investment equation. The capital stock has a predetermined value at $t = 0$ but consumption is free at all times. The equations in (5) form a first-order nonlinear system in two variables at each t. We can eliminate c_t and formulate the solution in terms of k_t sequence in the second-order nonlinear system

$$u'(F(k_t) - k_{t+1}) - \beta u'(F(k_{t+1}) - k_{t+2}) F'(k_{t+1}) = 0, \quad t = 0, 1, \ldots . \tag{6}$$

We shall use (6) below as an example since the notation is a bit simpler than (5). In fact, no matter which method is used, we should use identities like $c_t = F(k_t) - k_{t+1}$ to reduce the number of unknowns.

The system (1)–(3) is an infinite set of equations with an infinite number of unknowns. Under some conditions, there will be a locally unique solution; we will make that assumption as is implicitly done by all other methods. For example, standard theory tells us that the problem in (4) has a unique solution for any given initial capital stock k_0.

Any solution method must reduce the system (1)–(3) in some way. Most methods use *domain truncation* to reduce the problem to a finite-horizon problem. That is, they solve the truncated problem

$$g(t, x_0, x_1, \ldots, x_{T-1}, x^{ss}, x^{ss}, \ldots, z) = 0, \quad t = 0, 1, 2, \ldots, T-1, \tag{7}$$

$$x_{i,0} = \bar{x}_{i,0}, \quad i = 1, 2, \ldots, n_I, \tag{8}$$

$$x_{i,T-1} = x_i^{ss}, \quad i = n_I + 1, 2, \ldots, n, \tag{9}$$

where x^{ss} is the steady-state value of x, or some proxy for the long run. $n - n_I$ components of x_{T-1} are also fixed at their long-run values to make the number of unknowns equal to the number of equations. Domain truncation reduces (1) to (7), a system of nT nonlinear equations in nT unknowns, and reduces (3) to (9).

There are two key decisions in forming (7) and (3). First, the long-run proxy x^{ss} need not be the steady state. The objective is that the choice of x^{ss} in (7) and (3) should not significantly affect the solution since it fixes the solution only in the distant future at $t = T$. Second, we must choose T. All traditional methods try alternative values for T and accept a solution only when the choice of T does not substantially affect the solution.

Since T is typically large, we need to develop special methods to handle the large system of equations. Fortunately, we can apply methods from the literature on solving large systems (see, e.g., Kelley, 1995; Saad, 1996; Young, 1971). Some algorithms break the problem into smaller systems which are linked and then solve the individual systems iteratively until the full system converges. For example, Fair and Taylor (1983) use a block Gauss–Jacobi procedure for a specific choice of T and then tests the sensitivity of the solution to T. Other examples of this approach are Hall (1985), Fisher et al. (1986), Fisher (1992), and Hughes Hallett and Piscitelli (1998). Convergence of such methods depends on the order of the equations and is linear at best. The advantages are their simplicity and small memory requirements. However, they may not converge even after using various strategies including reordering of equations and dampening factors.

These methods often have an economic motivation. Given expectations, one computes the best actions, which in turn generates expectations. While this approach to solving dynamic economic models has some intuitive appeal, it produces algorithms which converge linearly if at all. There are alternative methods which have no economic intuition but are suggested by numerical analysis. One such alternative is Newton's method. Newton's method has no obvious economic "story", but it offers the possibility of rapid convergence. Using Newton's method appears difficult because the Jacobian is large, but the Jacobian for perfect foresight problems is often sparse. Juillard et al. (1998) pursued a Newton strategy exploiting sparseness. Gilli and Pauletto (1998) economize on this by using a Newton-style method together with a Krylov method to compute approximate Newton steps instead of exactly solving for the Newton step. The Newton approach has been generally useful in solving dynamic problems with a sparse structure, but not all dynamic models are sparse. For example, the overlapping generations model used in Auerbach et al. (1987) is not sparse.

This paper continues this approach of incorporating methods from numerical analysis, and refines it. We show that we can achieve even greater efficiency by exploiting properties of solutions to perfect foresight models and applying ideas from orthogonal polynomial theory and numerical integration along with Newton's method.

2. Parametric path method for perfect foresight models

The parametric path approach employs a strategy substantially different from traditional methods. Instead of treating each value of x_t as an independent variable it applies some a priori knowledge about how x_t evolves over time. For example, $1,2,1,2,\ldots$ is not a plausible candidate for the quarterly series for capital stock since it is physically impossible for the capital stock to change drastically in short periods of time. Similarly, consumers want to smooth their consumption and consumption paths will often be relatively smooth. This feature of the solution is not exploited by standard methods since they allow x_t and x_{t+1} be completely independent. Instead, our intuition says that x_t should be a smooth function of time t. This insight allows us to dramatically reduce (1)–(3) to a much smaller system. This smaller system can be solved using a much wider variety of methods than can be applied to (1)–(3). This section outlines the basic ideas, and the next section gives more precise description of the method in the context of the optimal growth example.

2.1. A simple parameterization

The parametric path method is an application of projection methods (see Judd, 1992, 1998) for solving functional equations. The key idea behind the parametric path method is to replace the sequence x with a parameterization of components of x in some way representing our beliefs that x evolves smoothly. First, consider the functional form

$$x_{i,t} = \Phi(t; a^i, \lambda) \equiv \left(\sum_{j=0}^{m} a_j^i t^j\right) \mathrm{e}^{-\lambda t} + x_i^{\mathrm{ss}}(1 - \mathrm{e}^{-\lambda t}). \tag{10}$$

In (10) the matrix of coefficients

$$a = (a_j^i)_{j=0,\ldots,m}^{i=1,\ldots,n} \in \mathbb{R}^{(m+1)\times n}$$

parameterize several polynomials in t. The initial value of the path $x_i(t; a, \lambda)$ is $\Phi(0; a^i, \lambda) = a_0^i$. For any coefficient matrix a, the path in (10) is a convex combination of the initial value a_0 and the long-run value x^{ss} where the time-dependent weights are $\mathrm{e}^{-\lambda t}$ and $1 - \mathrm{e}^{-\lambda t}$. The exponential term $\mathrm{e}^{-\lambda t}$ dominates any polynomial term in (10); therefore, $\Phi(\infty; a^i, \lambda) = x_i^{\mathrm{ss}}$ for any choice of a_j, $j = 1,2,\ldots$. Moreover, for any choice of a the path in (10) asymptotically satisfies the linear adjustment process

$$\frac{\mathrm{d}}{\mathrm{d}t}(x - x^{\mathrm{ss}}) = -\lambda(x - x^{\mathrm{ss}}). \tag{11}$$

Therefore, the x_t path in (10) begins at a_0 and converges linearly to the steady-state x^{ss}. The initial conditions (2) fix some components of a_0 since they imply

$$a_0^i = \bar{x}_{0,i}, \quad i = 1,2,\ldots,n_I. \tag{12}$$

Other than the conditions in (12), we are free to choose the components of a so that the sequence in (10) approximately solves (1).

Many features of (10) match our a priori knowledge about (1). The form in (10) imposes convergence to x^{ss} in the long run. Since we often know the steady-state x^{ss}, we can use this knowledge in (10). Traditional methods use knowledge of x^{ss} by specifying that it is the terminal state at some time T but make no restrictions about

x_t at $t < T$. Eq. (10) imposes the natural condition that x_t approach the steady state smoothly.

The choice of λ is also one which can use a priori information. Given the asymptotic behavior in (11), we want λ to be the rate of convergence associated with the dominant stable eigenvalue of the linearization of (1) around x^{ss}. Sometimes we can compute this. In fact, no matter what method one uses, the entire spectrum of the linearized system should be computed to ascertain local stability and uniqueness properties. Therefore, using the asymptotic rate of convergence for (1) to choose λ imposes no extra cost. Even if we do not know the spectrum, we often have good guesses about the asymptotic rate of convergence. We will see that some good guesses are satisfactory. Traditional methods make no use of the spectrum of the linearized system.

For many problems, the truth is that $x_t - x_{t+1}$ is small, x^{ss} is achieved only in the limit, and x converges smoothly at a known rate to x^{ss}, properties which hold for all examples of (10). Otherwise, (10) is quite flexible. When t is small, the $\mathrm{e}^{-\lambda t}$ terms have little importance, the polynomial terms dominate, and the free coefficients allow $\Phi(t; a, \lambda)$ to be flexible. This parallels our relative ignorance about x_t for small t. Therefore, (10) is a form which incorporates all of our a priori knowledge, short- and long-run, about equilibrium but is flexible enough to approximate equilibrium well.

The formula in (10) treats t as a continuous variable. This may initially seem odd in a discrete-time framework. However, there is nothing in (1) which requires t to be an integer. For example, the key expression for the time t in Eq. (6) is

$$u'(F(k_t) - k_{t+1}) - \beta u'(F(k_{t+1}) - k_{t+2})F'(k_{t+1}) \tag{13}$$

and is well-defined for noninteger t if we define $k_t = \Phi(t; a, \lambda)$. We shall proceed as if t were continuous, making it an integer only when desired. This is important since the whole idea of the parametric path method is to apply approximation methods for functions of a continuous variable to functions on the integers.

2.2. General parameterizations

The general idea of a projection method is to find some functional form $\Phi(t; a)$ such that the sequence

$$x_{t,i} = \Phi(t; a^i), \quad a^i \in \mathbb{R}^m, \ i = 1, \ldots, n \tag{14}$$

nearly solves (1)–(3). The simple parameterization in (10) is natural, but suffers from some defects. Judd (1992) emphasizes that it is generally better to use orthogonal polynomials.[2] To this end, we replace (10) with

$$\Phi(t; a) = \left(\sum_{j=0}^{m} a_j \phi_j(t) \right) \mathrm{e}^{-\lambda t} + x^{ss}(1 - \mathrm{e}^{-\lambda t}), \tag{15}$$

[2] See Judd (1992,1998) for discussions of orthogonal polynomials and their use in developing projection method algorithms for dynamic general equilibrium methods.

where the family $\{\phi_j(t)\}_{j=0}^{\infty}$ is orthogonal.[3] Since we want to allow paths similar to those in (10), a natural choice is

$$\phi_j(t) = L_j(2\lambda t)\mathrm{e}^{-\lambda t}, \tag{16}$$

where $L_j(x)$ is the degree j Laguerre polynomial. Laguerre polynomials are defined by the recursive formulas

$$L_0(x) = 1, \qquad L_1(x) = 1 - x$$
$$L_{j+1}(x) = \left(\frac{2j+1-x}{j+1}\right) L_j(x) - \frac{j}{j+1} L_{j-1}(x).$$

The key property of the ϕ_i is that they are mutually orthogonal; that is,

$$\int_0^{\infty} \phi_j(t)\phi_i(t)\,\mathrm{d}t = \int_0^{\infty} (L_j(2\lambda t)\mathrm{e}^{-\lambda t})(L_i(2\lambda t)\mathrm{e}^{-\lambda t})\,\mathrm{d}t = 0, \quad i \neq j$$

because the Laguerre polynomials $L_i(2\lambda t)$ are orthogonal with respect to the weight $\mathrm{e}^{-2\lambda t}$. The space of functions spanned by the form (15) is the same as that spanned by the form (10), and (15) is as flexible as (10) and imposes the same initial and asymptotic conditions as (10). We prefer the orthogonal representation in (15) since, as we will see, it has some numerical advantages.

Of course, there are many possible forms for the functional form $\Phi(t;a)$. Φ should be flexible enough to parsimoniously approximate any likely solution. The best choice of m in (15) cannot be determined a priori. Generally, the only correct choice is $m=\infty$. If the choice of the functional form is good then large m will yield very good approximations. We are most interested, however, in the smallest m that yields an acceptable approximation. We initially begin with small m and increase m until a backward error diagnostic, such as the Euler equation error, indicates little is gained by increasing m. Computational considerations also play a role in choosing a functional form. For example, Φ should be simple to compute.

Our task is somewhat simplified since each component of Φ is a one-dimensional function of t. This fact implies that we can use a wide variety of possible functional forms such as splines and rational polynomials. We will stay with polynomial systems for the purposes of this study.

2.3. Projection conditions

Once we choose a parameterization $\Phi(t;a)$, we need some way to compute the coefficients a in our approximation so that $\Phi(t;a)$ approximately solves (1) and (2). We will follow the procedure for projection methods laid out in Judd (1992). Define

[3] Here we use the notation $\Phi(t;a)$ and do not include λ explicitly. If there were a parameter in (14) like the λ parameter in (10), we fold it into the parameter list a. For the rest of the paper we view λ as a parameter which is fixed first and then vary the coefficients a to find a solution.

the residual function

$$R_i(t;a) = g_i(t, x(0;a), x(1;a), \ldots, x(s;a), \ldots), \quad i = 1, \ldots, n,$$

where again we treat t as a continuous variable. We want to find some a such that $\|R(t;a)\|$ is "practically" zero for all t, implying that Eqs. (1) and (2) nearly hold. For such an a, the time path $x(t;a)$ will be an approximate solution to the perfect foresight model (1) and (2).

To proceed, we need to specify our notion of $\|R(t;a)\|$ being small. There are several ways to do this. The first direct way is to define the L^2 norm (which equals the sum of squared residuals)

$$SSR(a) = \int_0^\infty \|R(t;a)\|^2 w(t)\,\mathrm{d}t, \tag{17}$$

where $w(t) > 0$ is some weighting function. The *least squares projection method* defines $SSR(a)$ for some w and chooses a to solve

$$\min_a SSR(a). \tag{18}$$

We cannot choose an arbitrary $w(t)$ since the domain of integration is infinite. However, the functional form (10) converges to the true steady state at rate λ, which implies that $R(t;a)$ also goes to zero at rate λ for any choice of a. Therefore, (17) is well-defined even if $w(t) = 1$.

The least squares method reduces the problem of solving an infinite number of equations to solving a nonlinear minimization problem in $\mathbb{R}^{mn}$, a more tractable problem. Of course, the standard difficulties will arise. For example, there may be local minima which are not global minima; since we want a good solution to (18), we would have to use a global optimization method.

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 solution in a less direct but more effective fashion. The key concept is that of a projection. Specifically, we choose a weight function $w(t)$, a set of test functions, $p_i(t)$, and form projections of the form

$$\langle R_i(t;a), p_j(t)\rangle \equiv \int_0^\infty R_i(t;a) p_j(t) w(t)\,\mathrm{d}t. \tag{19}$$

For these techniques the basic idea is that the true solution would produce a zero residual error function; in particular, the residual function would have a zero projection in all directions. Therefore one way to find the nm components of a is to fix m projections of the n residual functions, and choose a so that the projection of the resulting residual function in each of those m directions is zero. That is, we want to find a such that

$$\langle R_i(t;a), p_j(t)\rangle = 0, \quad i = 1, \ldots, n, \; j = 1, \ldots, m \tag{20}$$

for all residual functions $R_i(t;a)$ and m test functions $p_j(t)$. Eqs. (20) are not sufficient since the initial conditions must also be included in the analysis. The initial conditions

(2) imply

$$x_{0,i}(0) = \Phi_i(t; a^i) = \bar{x}_{1,i}, \quad i = 1, 2, \ldots, n_I. \tag{21}$$

Therefore, we aim to find some $a \in \mathbb{R}^{mn}$ which satisfies (21) as well as $nm - n_I$ conditions of the form (20).

We have presented the basic conceptual idea. First, parameterize the x_t path with some parsimonious functional form which can approximate well the equilibrium path. Second, fix the coefficients by directly imposing the initial and asymptotic conditions on the functional form, and by imposing enough projections from (20). While the concept is clear, there are many important details in the execution. The important detail for the parametric path method is the numerical approximation of the integrals in the projections. We next discuss this step in detail.

2.4. Integral approximations of projection conditions

The integrals in (19) need to be computed approximately. We continue with the fiction of t being continuous. In fact, the fiction of a continuous t is essential for the next step since quadrature methods apply, strictly speaking, only to functions on $\mathbb{R}$. We will use standard integration methods to approximate the projections in (19). Numerical integration methods generally take the form

$$\int_0^\infty h(t) w(t)\, \mathrm{d}t \doteq \sum_{\ell=1}^{N} \omega_\ell h(t_\ell) \tag{22}$$

for quadrature weights ω_ℓ and quadrature nodes t_ℓ. The integration formula in (22) tells us which t's to use. The maximum t_ℓ tells us how much into the future we need to look in order to approximate the x_t path. Fair–Taylor, L–B–J, and other traditional methods make an ad hoc determination of the horizon. Also, the t_ℓ's used depend on the particular integration formula used in (22). In fact, the t_ℓ's we use are often optimal given the integration formula used.

In the end, the parametric path method uses some integral approximation (22) and reduces (1) and (2) to the system of nonlinear equations

$$P_{ij}(a) \equiv \sum_{\ell=1}^{N} \omega_\ell R_i(t_\ell; a) p_j(t_\ell) = 0, \quad i = 1, \ldots, n, \; j = 0, \ldots, m, \tag{23}$$

$$\Phi(0; a) = \bar{x}_{1,i}, \quad i = 1, 2, \ldots, n_I. \tag{24}$$

The system (23) and (24) may be overidentified. We want to impose (24), so we drop n_I equations from (23); it is normally preferable to drop projections with some of the higher-order polynomial test functions. When we refer to the system (23) and (24) we will be referring to an exactly identified subsystem.

Here we see a critical feature of the parametric path method. In the end, we evaluate the equilibrium equations at only a small number of times t. Traditional methods choose some T and compute (1) for all $t < T$ for some large T but ignore $t > T$ except to assume that $x_t = x^{\mathrm{ss}}$. In the parametric path method, we do not have to hunt for a good

T. Instead, the integration formulas we use produce an efficient collection of t's to use. We may increase N in order to increase accuracy and reliability of the method, a strategy similar to choosing a larger T. In our examples, we will discuss three possible integration strategies and show that the parametric path method differs significantly from traditional methods.

The quadrature nodes t_ℓ will typically not be integers. If there is a reason to doubt the legitimacy of treating t as a continuous variable, one could use the approximation

$$\int_0^\infty h(t)w(t)\,\mathrm{d}t \doteq \sum_{\ell=1}^{N} \omega_\ell h(t_\ell^I), \tag{25}$$

where t_i^I is the nearest integer to t_i, and solve the projection system

$$P_{ij}(a) \equiv \sum_{\ell=1}^{m_q} \omega_\ell R_i(t_\ell^I;a)p_j(t_\ell^I) = 0, \quad i=1,\ldots,n, \ j=0,\ldots,m, \tag{26}$$

$$\Phi(0;a) = \bar{x}_{1,i}, \quad i=1,2,\ldots,n_I. \tag{27}$$

It is only at this point, the final detail in the algorithm, that we even need to consider restricting t to be an integer. Even in this case, the integration formulas tell us what choices to make for t.

Different choices of the test functions p_j define different implementations of the projection method and require different integration formulas. We will use the *Galerkin* method. In the Galerkin method, the approximation is a linear combination of basis functions, the test functions are the basis functions, and the weighting function is chosen so that the basis functions are mutually orthogonal. This produces a projections of the form

$$P_{ij}(a) \equiv \langle R_i(x;a), \varphi_j(x)\rangle = 0, \quad i=1,\ldots,n.$$

The Galerkin method is just one possible alternative, but one with a good track record. Many of the others described in Judd (1992) could also be used.

2.4.1. Gauss–Laguerre quadrature

The first natural choice is Gauss–Laguerre quadrature. Since $R(t;a)$ is bounded and each $\phi_i(t)$ contains an $\mathrm{e}^{-\lambda t}$ factor, it is natural to apply Gauss–Laguerre integration formula to (34). This approach is expressed in

$$\begin{aligned} P_j(a) = \int_0^\infty R(t;a)\phi_j(t)\,\mathrm{d}t &= \int_0^\infty R(t;a)L_j(2\lambda t)\mathrm{e}^{-\lambda t}\,\mathrm{d}t \\ &= \lambda^{-1}\int_0^\infty R(s/\lambda;a)L_j(2s)\mathrm{e}^{-s}\,\mathrm{d}s \\ &\doteq \lambda^{-1}\sum_{i=1}^{N} \omega_i R(s_i/\lambda;a)L_j(2s_i), \end{aligned}$$

where s_i is the ith node in the N-point Gauss–Laguerre quadrature formula and ω_i is the corresponding weight. Therefore, the integration formula tells us to examine the residual function at times s_i/λ, and give $R(s_i/\lambda; a)L_j(2s_i)$, the integrand at $t = s_i/\lambda$, weight ω_i. If we insisted on using integers, we would use the approximation

$$P_j(a) \doteq \sum \omega_i R(t_i^I; a) L_j(2\lambda t_i^I).$$

The mathematics literature, however, does not recommend Gauss–Laguerre quadrature. Gauss–Laguerre quadrature works best when the integrand is a polynomial multiplied by $e^{-\lambda t}$, but $R(t; a)L_j(2\lambda t)$ is not a polynomial since it is asymptotically zero. Our example below will display some of the weaknesses of Gauss–Laguerre in this context.

2.4.2. Logistic change of variable

We next examine two change of variable (COV) methods. First, consider the logistic map

$$t(x) = -\frac{1}{\lambda}\log\left(\frac{1-x}{2}\right) \tag{28}$$

which maps $x \in [-1, 1]$ to $t \in [0, \infty)$. Its inverse is $x = 1 - 2e^{-\lambda t}$. The projection equations' integrals over $[0, \infty)$ are transformed into integrals over $[-1, 1]$, according to

$$P_j(a) = \int_0^\infty R(t; a)\phi_j(t)\,dt = \int_{-1}^{1} R(t(x); a)\phi_j(t(x))t'(x)\,dx. \tag{29}$$

The integral in (29) is approximated using some integration formula for functions over $[-1, 1]$. Applying Gauss–Chebyshev integration produces the approximation[4]

$$P_j(a) \doteq \sum_{\ell=1}^{N} R(t(x_\ell); a)\phi_j(t(x_\ell))(1 - x_\ell^2)^{1/2} t'(x_\ell),$$

where the Chebyshev integration nodes are

$$x_\ell = \cos\left(\frac{2\ell + 1}{2N}\pi\right).$$

In this case, the weight on $R(t(x_\ell); a)\phi_j(t(x_\ell))$ is $(1 - x_\ell^2)^{1/2} t'(x_\ell)$.

The logistic transformation is somewhat natural since, by construction, $R(t; a)$ is asymptotically proportional to $e^{-\lambda t}$. However, it is not as stable as we desire since the integrand $R(t(x); a)\phi_j(t(x))t'(x)$ is not bounded on $[-1, 1]$. This unboundedness reduces the effectiveness of any integration formula we may apply to (29).

2.4.3. Algebraic change of variable

We next examine a second COV transformation. Consider the algebraic map

$$t(x) = L\frac{1 + x}{1 - x}, \tag{30}$$

where L is a free parameter. It's inverse is $x = (t - L)/(t + L)$. The projection equations again take the form of (29) but now with the algebraic COV defining $t(x)$. We choose L

[4] We have dropped a π/N factor since we want to solve $P = 0$.

to help the integration in (34). The integral in (29) has range $x \in [-1, 1]$. Since $R(t; a)$ is asymptotically proportional to $e^{-\lambda t}$ we choose L so that the product $e^{-\lambda t(x)} t'(x)$ has a small derivative with respect to x; this makes the integral in (29) easier to approximate. The value of L which minimizes the maximum slope of $e^{-\lambda t(x)} t'(x)$ over $x \in [-1, 1]$ is $L = 1/\lambda$. Therefore, we choose $L = 1/\lambda$. Otherwise, we proceed as we did with the logistic COV. The mathematics literature finds the algebraic COV to be more stable asymptotically. In fact, it corresponds to approximating the solution with a rational Chebyshev polynomial, a generalization of Chebyshev polynomials to the half-infinite interval. See Boyd (1989) and the literature cited there for a discussion of these details.

Our example below will present all three methods. The example will demonstrate some of the weaknesses and strengths of the alternative integration methods. It is useful to know all three methods since one may work when the others do not.

2.5. *Solving the projection conditions*

To identify the coefficients a we either use a minimization algorithm to solve (18) or a nonlinear algebraic equation solver to solve (23) and (24). The nonlinear equations associated with Galerkin and other inner product methods can be solved by the variety of nonlinear equation methods. A key advantage of the parametric path method is that it reduces the problem to a small dense nonlinear equation system to which several nonlinear equation solution methods can be effectively applied. Newton's method can converge rapidly. Furthermore, if Newton and Gaussian methods do not work, then one can use globally convergent homotopy methods since they do not require good initial guesses. We will use Newton's method in our example, but the user should keep in mind the full range of methods made available by the reduction in dimensionality.

2.6. *Initial guesses*

Good initial guesses are important since projection methods involve either a system of nonlinear equations or optimizing a nonlinear objective. One advantage of the parametric path method is that there is a natural initial guess which may be quite close to the solution. We know the steady-state values for all variables, and we often have a good guess for the asymptotic rate of convergence to the steady state. A natural initial guess is the path which smoothly moves from the initial state x_0 to the steady-state x^{ss} at the asymptotic rate of convergence λ

$$x^{\text{init}}(t) = x_0 e^{-\lambda t} + x^{ss}(1 - e^{-\lambda t}). \tag{31}$$

If λ is the asymptotic rate of convergence of the solution to (1), then (31) is probably a good initial guess. For example, if x is the single state variable in a problem, $x^{\text{init}}(t)$ is the global extension of the linear approximation based at the steady-state x^{ss}.

The guess in (31) is a very simple one. We may be able to improve on it by using the least squares method. The least squares approach will generally not produce 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 for a poor initial guess. These facts motivate a two-stage approach to finding an initial guess. First, begin with (31). Second, use a least squares approach with a loose convergence criterion to quickly compute a low-quality approximation better than (31). We did not need to do this in our examples, but we offer it here as a general guide to finding a satisfactory initial guess.

2.7. *Checking the solution—error analysis*

The system (23) and (24) computes (1) only at a small number, often a very small number, of t's. Therefore, a solution to (23) and (24) is perhaps a solution to (1), the problem of interest, at only those t's, and not a solution at other times. Before we accept any solution to (23) and (24) as an approximate solution to (1), we need to check it at some of the t's we did not use in (23) and (24). This is a critical step in any projection method which uses information at only a few points to approximate a general solution. We will use the approach to backward error analysis used in Judd (1992,1998).

Suppose that the solution to (23) and (24) implies the approximation $x_t = \Phi(t;a^*)$. To make the parametric path method comparable to traditional approaches, we evaluate

$$E = \max_{t=0,1,\dots,T} \|g(t,x,z)\|,$$

where $x_t = \Phi(t;a^*)$ and T is some large time. The index E is the maximum error in Eqs. (1) over a long range of time. Most traditional methods for solving perfect foresight models continue until E is small since testing E is part of any conventional stopping criterion for solving nonlinear systems of equations. This check allows us to use the same stopping rule as traditional methods. In the parametric path method, if a solution to (23) does not imply a sufficiently small value for E we can begin again with a more flexible parameterization. In the case of an orthogonal polynomial approximation this means using higher-order polynomial terms. In traditional methods, a high value of E implies that one must increase T. Therefore, the use of E to test a solution is a common feature of most algorithms. Table 1 summarizes the parametric path method.

Table 1
Summary of parametric path method

Step 1:	Choose parameterization $x = \Phi(t;a)$
Step 2:	Form residual functions $R_i(t;a) = g_i(t,\Phi(t;a),z)$
Step 3:	Select test functions $p_j(t)$
Step 4:	Form projections $P_{ij}(a) \doteq \langle R_i(t;a), p_j(t)\rangle$, using integration formulas where necessary
Step 5:	Solve a system of equations consisting of initial conditions plus projection equations $P_{ij}(a) = 0$
Step 6:	Compute $E = \max_{t=0,1,\dots,T} \|g(t,x,z)\|$; accept a if E is sufficiently small; otherwise begin again at Step 1 with a more flexible approximation

2.8. Comparisons with alternative methods and extensions

The parametric path method initially appears to be different from standard methods, but a general perspective shows that they have a common structure. We next take a general perspective to show the common elements and the differences. This comparison will also help us develop hybrid methods combining the strengths of different approaches.

Problem (1) is an infinite system of equations in an infinite number of unknowns with a solution in $\mathbb{R}^\infty$. Any solution method replaces $\mathbb{R}^\infty$ with a finite-dimensional approximation. Solution methods differ primarily in the finite-dimensional approximations they make. Methods using domain truncation find an approximate solution to (1) in the finite-dimensional subspace

$$X_T^{\mathrm{DT}} = \{x \in \mathbb{R}^\infty | x_t = x^{\mathrm{ss}},\ t > T\}.$$

X_T^{DT} is the space of sequences which equal x^{ss} for $t > T$. Domain truncation methods find some element of X_T^{DT} which satisfies some initial set of equations in (1) and neglects equations at later t. Successively better approximations are produced by increasing the terminal date T.

The parametric path methods also finds an approximate solution to (1) in a finite-dimensional space of $\mathbb{R}^\infty$. The space used implicitly by the parameterization (10) is

$$X_{m,\lambda}^{\mathrm{PP}} = \left\{ x \in \mathbb{R}^\infty | x_t - x^{\mathrm{ss}} = \mathrm{e}^{-\lambda t} \sum_{i=0}^{m} a_i t^i \right\}.$$

As we take m to infinity, $X_{m,\lambda}^{\mathrm{PP}}$ spans elements of $\mathbb{R}^\infty$ which converge to x^{ss} at rate λ. As we increase T, X_T^{DT} also spans the relevant portion of $\mathbb{R}^\infty$ in some sense, but less efficiently. The key question is which approach uses the smallest finite-dimensional space containing good approximations to the true solution of (1). For problems with smooth solutions, the space $X_{m,\lambda}^{\mathrm{PP}}$ has many advantages over X_T^{DT} and they are exploited in the parametric path method.

Mercenier and Michel (1994a) present a time aggregation which is similar to the parametric path in that they also produce a nonuniform sample of times, t_i, to produce a model with substantially fewer unknowns. However, they do not use an explicit interpolation scheme across those points (their graphs appear to use piecewise linear interpolation) and they impose an approximate Euler equation between times t_i and t_{i+1}, whereas we impose the true Euler equation between t_i and $t_i + 1$. The parametric path method imposes the asymptotic conditions directly through the functional form whereas Mercenier and Michel (1994a) use a terminal valuation function in a finite-horizon approximation. Also, their preferred method of choosing the t_i (see Mercenier and Michel, 1994b) aims at improving asymptotic performance whereas the parameter λ in our parametric form handles the asymptotics and our choices of t_i aim at improving initial and intermediate performance.

Some perfect foresight problems do not produce sparse systems of equations. In particular, the overlapping generations model used by Auerbach et al. (1987) produces systems of equations which are too dense for sparse methods. The parametric path method does not rely on sparseness and could be used to solve overlapping generations models; we must leave examination of that application to future work. The key assumption is that the endogenous economic variables behave in a relatively smooth fashion and that a low-dimensional approximation is good once one focuses the search on a suitable finite-dimensional space of functions.

The parametric path method can be useful even in cases where the equilibrium paths of economic variables are not smooth. The projection aspect of the parametric path method implies that the parametric path method will likely produce a smoothed approximation of the true path. This can then be used as an initial guess for a more refined method such as Fair–Taylor or any of the other methods.

A hybrid approach is also obvious for some problems. Suppose that large monetary and tax policy changes are expected at $t = 5, 10$, and 15, but that after $t = 15$ the economy experiences no more shocks. The equilibrium paths for consumption, investment, and prices are likely to be volatile and not smooth for $t < 15$. However, after $t = 15$ the equilibrium paths will likely be smooth. If the solution is smooth after $t = 15$ then one could use a parameterization similar to (15) for $t > 15$ but allow equilibrium at $t < 15$ to be represented by a flexible sequence, as in domain truncation methods. The collection of unknowns would be the x_t for $t < 15$, plus the coefficients of a parameterization for x_t at $t > 15$. The equations would be the full set of equations from (1) for $t < 15$ plus a representative sampling of equations from (1) at suitable t's after $t = 15$. In most problems studied in economics, the economy eventually settles into a smoothly convergent path. While some initial component may need to be approximated as in the truncated domain method, the smooth terminal path is best approximated using the ideas from the parametric path method.

3. Implementations of the growth example application

We now apply the parametric path method to the optimal growth problem displayed in (4). We solve the system (6) and compare the parametric path solution with another solution known to be of high quality. While this example is a very simple one, it does share the essential features present in many other perfect foresight models: the system is sparse and nearly diagonal, and equilibrium converges linearly to the steady state. Also, this simple example allows us to discuss many of the critical details which go into an effective implementation of the parametric path method.

Our central example assumes $u(c) = c^{1-\gamma}/(1-\gamma)$ with $\gamma = 1.1$. We also examined the cases $\gamma = 0.5, 5.0$ and found no significant differences in algorithm performance. We choose $\beta = 0.99$, implying that utility is discounted by 1% during one period of time. If we assume that the annual discount rate is 4%, a common assumption, then our unit of time is a quarter. All examples assume $F(k) = k + Ak^{\alpha}$ where $\alpha = 0.25$ and A is chosen so that the steady-state capital stock is $k^{\mathrm{ss}} = 1$. Other sensible values of α produced the same results. We choose the initial capital stock so as to present a challenge to the parametric path method. If the initial capital stock k_0 is close to the steady state then the first-order approximation expressed in (31) is a very good approximation without any help from polynomial terms. Therefore, we choose $k_0 = 0.5$, implying that the initial capital stock is half of the steady-state capital stock.

3.1. Parameterization

We use the parameterization

$$k(t) = \sum_{j=0}^{m} a_j \phi_j(t) + k^{\mathrm{ss}}(1 - \mathrm{e}^{-\lambda t}), \tag{32}$$

where $\phi_j(t) = L_j(2\lambda t)\mathrm{e}^{-\lambda t}$. The initial condition is $k(0) = k_0 = \sum_{j=0}^{m} a_j \phi_j(0)$ and implies

$$a_0 = \phi_0(0)^{-1} \left(k_0 - \sum_{j=1}^{m} a_j \phi_j(0) \right).$$

Therefore, the full parameterization is

$$k(t) = \sum_{j=1}^{m} a_j \phi_j(t) + \phi_0(0)^{-1} \left(k_0 - \sum_{j=1}^{m} a_j \phi_j(0) \right) + k^{\mathrm{ss}}(1 - \mathrm{e}^{-\lambda t}) \tag{33}$$

and the unknowns are the coefficients a_i, $i = 1, 2, \ldots, m$.

There are two key features of (33). First, the exponential decay terms $\mathrm{e}^{-\lambda t}$, some of which are in the ϕ terms and one explicitly in (33), impose the boundedness conditions. Second, the initial conditions are also satisfied for any choice of $a \in \mathbb{R}^m$. These facts allow us to focus on finding an a which produces a good solution to (6).

The first key step in applying the parametric path method is choosing λ in (33). We know from theory that the path of capital solving (6) converges asymptotically to the steady state at a linear rate equal to μ, the stable eigenvalue of the linearized system around the steady-state k^{ss}. This suggests that we use $\lambda = \mu$ in the parameterization (33) but we will investigate a range of values for λ. It is easy to compute μ for each parametric case of (4) we examine; in particular, when $\gamma = 1.1$ the stable eigenvalue is $\mu = 0.0122$. The second key step is deciding the number of polynomial terms, m, we include in (33); we will consider several choices of m.

3.2. Projection conditions and integration formulas

Once we have chosen a particular form for (33), we next specify the conditions which will fix $a \in \mathbb{R}^m$. The residual function for (6) is

$$\begin{aligned} R(t,a) = {} & u'(F(k(t)) - k(t+1)) \\ & - \beta u'(F(k(t+1)) - k(t+2))F'(k(t+1)), \end{aligned}$$

$R(t;a)$ is well-defined for any real value of t, not just integer values of t, since $k(t)$ is defined for all t in (33). This observation makes us more comfortable with using continuous variable methods. We want to choose a so that $R(t;a)$ is nearly zero for

Table 2
Gauss–Laguerre weights and times

10-point formula	
Times	22, 115, 285, 536, 875, 1312, 1866, 2564
Weights	0.31, 0.40, 0.22, 0.062, 0.0095, 7.5(−2), 2.8(−3), 4(−5), 2(−7), 1(−11)
20-point formula	
Times	5.8, 30, 75, 140, 225, 332, 460, 611, 786, 987, 1214, 1471, 1760, 2086, 2453, 2870, 3347, 3903, 4574, 5452
Weights	0.17, 0.29, 0.27, 0.17, 0.075, 0.025, 0.0062, 0.0011, 1.6(−4), 1.5(−5) 1(−6), 5(−8), 2(−9), 4(−11), 5(−13), 3(−15), 1(−17), 2(−20), 5(−24), 2(−28)

Note: $x(-m)$ means $x \times 10^{-m}$.

all t. By construction, $R(t;a) \to 0$ and $t \to \infty$. The coefficients a adjust $k(t)$ so that $R(t;a)$ is small for finite t. We will use the Galerkin method where the test functions are the basis functions $\phi_j = L_j(2\lambda t)\mathrm{e}^{-\lambda t}$. To this end, we define the set of projection formulas

$$P_j(a) = \int_0^\infty R(t;a)L_j(2\lambda t)\mathrm{e}^{-\lambda t}\,\mathrm{d}t, \quad i = 0,1,\ldots \tag{34}$$

and approximate the $P_j(a)$ with numerical quadrature formulas of the form

$$P_{ij}(a) \equiv \sum_{\ell=1}^{N} \omega_\ell R_i(t_\ell;a)L_j(2\lambda t_\ell) = 0, \quad i = 1,\ldots,n, \;\; j = 0,\ldots,m. \tag{35}$$

We apply the three basic integration formulas discussed above to illustrate the differences.

First, we used Gauss–Laguerre quadrature. Table 2 displays the times,[5] t_ℓ, used for the 10- and 20-point quadrature formulas in the case $\gamma = 1.1$ and $\lambda = \mu = 0.0122$. The most striking feature is the dispersion of times used by the formulas. The first time used in the 10-point formula is $t = 22$. Therefore, if we use Gauss–Laguerre integration our algorithm ignores the Euler equation errors in the first 21 periods. This appears odd since it is the early portion of the equilibrium path which is unknown to us. The last time used is the distant $t = 2564$. The 20-point formula uses more "early" times—$t = 5.8$ and 30 are included—but it is still remarkable how few early times are used. If we chose λ different from μ, the impact on the $t_i = s_i/\lambda$ is proportional. For example, if $\lambda = 2\mu$ then each time t is replaced by $t/2$. A larger magnitude for λ implies that the approximate solution converges to the steady state faster and that earlier times are used in the projection equations.

The second important feature of Gauss–Laguerre integration is the pattern of weights. The weights for both the 10- and 20-point formulas are heavily concentrated on the early times. The first projection is $\phi_1(t) = \mathrm{e}^{-\lambda t}$ and only the early times matter for

[5] To economize on space, we only show the first few significant digits of the t_ℓ values. No time t_ℓ in Table 2, or any other similar table, is an integer.

Table 3
Logistic transformation—weights and times

10-point formula	
Times	$1.01, 9.2, 26, 52, 90, 142, 213, 315, 477, 834$
Weights	$0.049, 0.13, 0.19, 0.2, 0.18, 0.13, 0.08, 0.03, 8(-3), 3(-4)$
20-point formula	
Times	$0.25, 2.3, 6.4, 13, 21, 32, 45, 61, 79, 101,$ $127, 157, 193, 235, 285, 348, 427, 536, 702, 1061$
Weights	$0.012, 0.036, 0.058, 0.076, 0.09, 0.1, 0.1, 0.1, 0.09, 0.08,$ $0.07, 0.06, 0.05, 0.03, 0.02, 0.01, 6(-3), 2(-3), 5(-4), 2(-5)$

Note: $x(-m)$ means $x \times 10^{-m}$.

$P_1(a)$. The formula puts very little weight on the most distant times, but even $t = 536$ contributes makes a 6% contribution to the final approximation of the 10-point formula. For projections $P_j(a)$ with large j, the small weights for large t_i values is partially balanced by the fact that the $L_j(2\lambda t_i)$ values are quite large for large t_i, implying that most $\omega_\ell R_i(t_\ell; a) L_j(2\lambda t_\ell)$ terms in (35) are nontrivial. Therefore, the large t_i values are important. However, if one were to use more than 20 points, many of the new points' weights would come close to machine zero. This would make their contribution a possible source of error. Therefore, there is little value in using many points. While the Gauss–Laguerre quadrature method may be useful in small problems like ours, it is not likely to be reliable in general.

We next used the logistic COV methods. Table 3 displays the times used in (35) and their weights, again for the case $\lambda = 0.0122$. In this case, we see that early times are better represented than in the Gauss–Laguerre case, and that the weights are also less concentrated. If we chose λ different from μ, the impact on the times is transparent in (28). For example, if $\lambda = 2\mu$ then each time t is replaced by $t/2$. Also, note that the weights for the large t values are much larger than the weights for the large t's in the Gauss–Laguerre formulas; therefore, the terms in (35) make more uniform contributions to the integral approximation.

The final case used the algebraic COV. Table 4 contains the times and weights used in the 10- and 20-point formulas when $\gamma = 1.1$, $\lambda = \mu = 0.0122$. We see that the t's used by the algebraic transformation are somewhat more concentrated at small values than in Table 3. As in Table 3, the weights are relatively uniform except for large t's. Since $R(t(x); a)t'(x)$ has bounded derivatives on $[-1, 1]$, we expect that integral formulas are more reliable in general. As before, if we choose λ different from μ, the impact on the times is transparent from (30) and our choice $L = 1/\lambda$. For example, if $\lambda = 2\mu$ then each time t is replaced by $t/2$.

Tables 2–4 display the key differences among the integration methods. In particular, the COV formulas use more uniform weights than the Gauss–Laguerre formula and samples more heavily from the early times. This is a natural feature of using Gauss–Chebyshev quadrature which also oversamples points at the ends of the $[-1, 1]$ interval. The Gauss–Chebyshev oversampling near $x = -1$ corresponds to using early t's.

Table 4
Algebraic transformation—weights and times

10-point formula	
Times	0.51, 4.7, 14, 31, 60, 112, 218, 478, 1422, 13232
Weights	0.025, 0.075, 0.13, 0.18, 0.22, 0.22, 0.13, 0.015, 7.0(−7), 5.0(−68)
20-point formula	
Times	0.13, 1.1, 3.2, 6.5, 11, 17, 26, 37, 51, 70 96, 132, 184, 261, 386, 602, 1030, 2071, 5851, 53091
Weights	0.0062, 0.019, 0.031, 0.044, 0.057, 0.071, 0.084, 0.097, 0.11, 0.11 0.11, 0.10, 8(−2), 5(−2), 2(−2), 2(−3), 3(−5), 2.2(−10), 9.6(−30), 1.2(−278)

Note: $x(-m)$ means $x \times 10^{-m}$.

3.3. Numerical results

The power and precision, as well as the sensitivity to implementation details, of the parametric path method are illustrated by the numerical results for the example in (4) with $u(c) = c^{1-\gamma}/(1-\gamma)$ and $F(k) = k + Ak^{\alpha}$. The tables below assume $\gamma = 1.1$ and $\beta = 0.99$. The problem (4) is a trivial one in many ways but we can be confident about the parametric path method only if it performs well for simple problems like (4). Therefore, it is appropriate to focus on this application before moving on to larger problems.

We used Newton's method to solve the various implementations of (35), with the proviso that we allow only 10 iterates and take that iterate for which the norm of (35) is smallest. The initial guess is always (31). These choices handicap us somewhat since there are better nonlinear equation solution methods, such as the Powell hybrid method. We do this so that we can focus on the basic ideas. We also imposed a very tight convergence criterion on our Newton algorithm, stopping only when the norm of (35) was below 10^{-10}. If the parametric path method works well with these handicaps, then it is likely to work even better generally.

We evaluate the solutions in three ways. First, we compute the maximum Euler equation error over the first 2500 periods. This is defined in

$$E_{\mathrm{E}} = \max_{t=1,\ldots,2500} \frac{|g(t, \hat{k}_t, \hat{k}_{t+1})|}{u'(\hat{c}_t)\hat{c}_t}, \tag{36}$$

where $\hat{k}_t$ and $\hat{c}_t$ are the capital and consumption paths implied by the solution to (35). This index can be used for any problem, and should be used to evaluate the accuracy of any numerical solution. In general we do not know how large m should be to get a good solution. Therefore, the parametric path method starts with a small m and increases it until we arrive at a solution where E_{E} is below some target level.

Second, since problem (4) is simple, we compute the optimal consumption policy function $C(k)$ for (4) using the method outlined in Judd (1992). The normalized Euler equation errors, as defined in (36), from that method were 10^{-10} or less, so we took those solutions to be the truth. This means that if a social planner used $C(k)$ at every

Table 5
Maximum Euler equation error

m	$\lambda = 0.1\mu$	$\lambda = 0.5\mu$	$\lambda = \mu$	$\lambda = 2\mu$
0	6.4(−3)	4.2(−3)	2.7(−4)	1.8(−2)
1	4.6(−3)	1.4(−3)	7.8(−5)	1.3(−2)
2	3.4(−3)	3.4(−4)	5.2(−6)	1.6(−2)
3	2.5(−3)	2.7(−4)	3.1(−5)	1.8(−2)
4	1.7(−3)	5.9(−5)	8.6(−6)	2.0(−2)
5	1.2(−3)	7.6(−5)	1.2(−5)	2.0(−2)
6	8.4(−4)	7.5(−6)	3.7(−6)	1.9(−2)

point in time then at every point in time his choice for consumption is less than one dollar per 10^{10} dollars, a very small error. We used $C(k)$ to compute the true path for k and c up to $t = 2500$ given the initial value $k = 0.5$. These paths were then used to assess the accuracy of the parametric path method. Specifically, we take the true path k_t and the path $\hat{k}_t$ produced by the parametric path method, and report the maximum error relative error as defined in

$$E_k = \max_{t=1,\ldots,2500} \frac{|k_t - \hat{k}_t|}{k_t}.$$

Similarly, the maximum relative error for consumption is defined by

$$E_c = \max_{t=1,\ldots,2500} \frac{|c_t - \hat{c}_t|}{c_t}$$

The first step in the parametric path method is choosing λ in (33). The asymptotic rate of convergence to the steady state of (4) is a natural choice; let μ be that value. For $\gamma = 1.1$, $\mu = 0.0122$. It is easy to compute μ for each case of (4) we examine. Table 5 examines the four choices $\lambda = 0.1\mu, 0.5\mu, \mu, 2\mu$. Table 5 also examines several choices of m, the degree of the polynomial pieces of our approximation (16).

Tables 5–7 present results when we use 20-point Gauss–Chebyshev integration with the algebraic transformation in (30). The tables examine $\lambda \in \{0.1\mu, 0.5\mu, \mu, 2\mu\}$ and $m \in \{0, 1, 2, 3, 4, 5, 6\}$. The case $m = 0$ takes the initial guess as the final solution. The differences between the $m = 0$ and $m > 0$ cases tell us how much we gain from adding the polynomial terms to the approximations in (10) and (15).

Table 5 presents the maximum Euler equation error. As expected, the choice of $\lambda = \mu$ does best, presumably since the asymptotic convergence in the approximated path, λ, equals the known true rate μ. In fact, the solution with $m = 1$ does almost as well as $m = 6$. Going from $m = 0$ to 6 reduced the Euler equation errors by a factor of 100, showing that the polynomial terms reduce the error substantially. When we choose $\lambda = 0.5\mu$, the procedure does almost as well. The approximation with $\lambda = 0.5\mu$ and $m = 1$ is 100 times worse than that for $\lambda = \mu$ and $m = 1$, but the $\lambda = 0.5\mu$ solutions improve substantially as we increase m, and essentially catches up to the $\lambda = \mu$ case when $m = 6$. The $\lambda = 0.1\mu$ choice also improves steadily as we increase m, but much more slowly. These cases show that if we use a λ in (16) which is less than the eigenvalue μ, the true asymptotic rate of convergence, then the polynomial terms can compensate.

Table 6
Maximum error in k_t

m	$\lambda = 0.1\mu$	$\lambda = 0.5\mu$	$\lambda = \mu$	$\lambda = 2\mu$
0	3.5(−1)	1.2(−1)	9.8(−4)	1.3(−1)
1	1.3(−1)	2.7(−2)	5.7(−4)	1.0(−1)
2	1.0(−1)	5.6(−3)	1.6(−4)	6.3(−2)
3	7.0(−2)	2.2(−3)	9.6(−5)	3.7(−2)
4	4.7(−2)	7.1(−4)	3.7(−5)	2.1(−2)
5	3.1(−2)	3.5(−4)	2.2(−5)	1.4(−2)
6	2.0(−2)	1.2(−4)	8.7(−6)	9.4(−3)

Table 7
Maximum error in c_t

m	$\lambda = 0.1\mu$	$\lambda = 0.5\mu$	$\lambda = \mu$	$\lambda = 2\mu$
0	1.3(−1)	7.3(−2)	2.0(−3)	1.5(−1)
1	7.7(−2)	1.0(−2)	2.2(−4)	4.9(−2)
2	4.9(−2)	2.3(−3)	7.3(−5)	3.8(−2)
3	3.0(−2)	1.2(−3)	8.0(−5)	3.3(−2)
4	1.9(−2)	4.0(−4)	3.1(−5)	2.8(−2)
5	1.1(−2)	2.6(−4)	2.5(−5)	2.5(−2)
6	7.0(−3)	7.6(−5)	9.4(−6)	2.0(−2)

The last column in Table 5 shows that choosing $\lambda > \mu$ is much more dangerous. In fact, when $\lambda = 2\mu$, the approximations do not improve as we increase m from 1 to 6. Therefore, it is much easier for the polynomial terms in (16) to compensate for a too small choice of λ than for an excessively large choice of λ. This is intuitive since large values of λ crush polynomial terms quickly, making it difficult for the polynomial terms to compensate for a bad choice of λ. In contrast, if the λ choice is too small, the polynomial terms can help the path get to the steady state more quickly since they control the path for small t. For most problems, one has to rely on the E_{E} errors computed in Table 5 when evaluating a numerical solution. The typical algorithm will continue until E_{E} is below some target. Table 5 shows that the parametric path method can achieve small values for E_{E}.

Since our problem is simple, we can also use projection methods from Judd (1992) to compute the consumption policy function $C(k)$. This will not be possible for large problems where Fair–Taylor and L–B–J are used, but the ability to use projection methods to compute $C(k)$ for this problem gives us a chance to directly measure the error of the parametric path solution (or any other method applied to the example problem). Tables 6 and 7 report the maximal capital path errors, E_k, and the maximal consumption path errors, E_c. These errors follow the same patterns as the E_{E} errors. The errors decline as m increases and increase as λ deviates from μ. However, the relative errors in k and c are often larger than the normalized Euler equation errors in Table 5. In some cases, the relative errors in k and c are an order of magnitude or more greater than the relativized Euler equation errors. When we solve systems like

Table 8
Performance indices

Number of iterations	3–5, except when $\lambda = 2\mu$
Jacobian condition number	0–5, except when m is close to N

(1) we often rely on the size of the errors in (1) to indicate when we can stop and accept a solution. These results tell us that we need to put stringent stopping rules on (1) if we are going to have accurate solutions for the economic variables of interest.

We present only the results for the algebraic COV approach since the results for the other methods were almost identical. There were a few differences. The Gauss–Laguerre method did worse when $\lambda=2\mu$ than the other two methods, probably reflecting the facts that it is difficult for the method to work when $\lambda = 2\mu$ and that the Gauss–Laguerre weights are nearly machine zero for some points. Also, most methods had difficulty when $m = 6$ and $N = 10$. The method where we set $m = N$, commonly called collocation, never worked. This arises because, as Tables 3 and 4 indicate, there are always some points in a quadrature formula which have essentially zero weight. Therefore, an N-point integration rule is really an $N - 2$ or $N - 3$ point formula or worse. Therefore, in order to avoid ill-conditioning, we need to choose $N \gg m$.

The details of these procedures were important. We also tested some Newton–Cotes rules combined with the algebraic transformation approach and found that they did much worse. When we used the algebraic transformation (30), we had to choose the free parameter L, and we argued that $L = 1/\lambda$ was a good value. We tried some alternative values for L and found that the rule $L = 1/\lambda$ was a good rule, performing better than alternatives where L was substantially different. The results concerning the choice of λ, the integration rules, and the choice of L in (30), show that the performance of the parametric path method is sensitive to the implementation. We cannot use arbitrary functions of the form (10). However, we did find good implementations of the parametric path method once we applied basic ideas from approximation and quadrature theory.

Some other indices indicate the efficiency of the parametric path algorithm. Table 8 reports some important statistics. Only a few iterations of Newton's method were needed in all cases. Recall that we used a very tight convergence criterion of 10^{-10}; a more standard criterion of 10^{-6} would have reduced the number of iterations by one always and sometimes two.

The condition number of a nonlinear system of equations is important for the performance of Newton's method, or any method using the Jacobian. The condition numbers of the Jacobians were always low, below 6, except for the nearly underidentified problems. The low condition number of the Jacobian is part of the reason why so few iterations of Newton's method were needed. We got low condition numbers because we used an orthogonal basis for our approximation. When we used (10) the condition numbers were much higher and convergence much slower. In fact, Newton's method often failed when we used (10). Furthermore, the Jacobians were nearly triangular with the diagonal elements and elements above the diagonal dominating the elements below the diagonal.

We maintained the fiction that t was continuous throughout the computations, only restricting ourselves to integer t when we evaluated the errors E_{E}, E_k, and E_c. We could have rounded all the noninteger values of t used in the projection equations, but when we did so the approximations were substantially worse, often increasing our error indices by a factor of 10. There was no problem arising from treating t as a continuous variable.

Our example is a particularly simple one whereas we hope that the parametric path method is useful for more complex problems. Space limitations prevent us from pursuing more interesting examples, but some multidimensional models can be considered using our results. Consider the problem

$$\begin{aligned}
\max_{c_t} \quad & \sum_{t=0}^{\infty} \beta^t (u_1(c_{1,t}) + u_2(c_{2,t})) \\
s.t. \quad & k_{i,t+1} = F(k_{i,t}) - c_{i,t}, \quad i = 1,2 \\
& k_{i,0} = 0.5.
\end{aligned} \tag{37}$$

This is a problem with two capital stocks and two consumption goods, but utility is additively separable across time and goods, only good i is used to produce capital stock i, and only capital stock i is used to produce good i. We are also implicitly assuming that labor supply to each sector is fixed. The separability implies that the good 1 and capital stock 1 can be treated independently from good 2 and capital stock 2. Therefore, we have already solved the two-good model in (37). Our procedure would first find the dominant eigenvalue, which would be the eigenvalue for the single good economy for good 1 if u_1 has the smaller intertemporal elasticity of substitution. It would then use that eigenvalue to approximate both the k_1 and k_2 paths. Our results show that the parametric path method would do well even for the faster converging k_2 since we would be using the dominant (slowly convergent) eigenvalue to approximate more rapidly convergent k_2 path. Tables 5–7 showed that the parametric path algorithm worked well even when $\lambda \ll \mu$. This simple example indicates that the parametric path method would work well even in more complex multidimensional contexts with a variety of eigenvalues as long as we correctly identify the dominant eigenvalue.

3.4. Comparisons with alternative methods

Our example is far simpler than any interesting general equilibrium model. Therefore, any comparisons with other algorithms must be considered preliminary, and any firm conclusions must wait for comparisons based on much larger models. However, our simple example does highlight trade-offs which will be present in any application. The parametric path method performed far faster in our examples than either Fair–Taylor or L–B–J possibly could. If one truncated the problem at $T = 100$, then L–B–J must invert a sparse near-diagonal matrix of dimension 100. This is not difficult, but it at least requires evaluating the Euler equation for $t = 1$ to $t = 100$ each time the Jacobian is needed. The parametric path method evaluated the Euler equation only at 20 points in our examples, a factor of five times fewer points. Furthermore, the 20 points chosen were optimal according to some integration scheme, implying a more accurate sample.

Since 100 quarters, or 25 years, is a rather short horizon in these models and since the parametric path method converged in just a few Newton steps, it is clear that the parametric path method dominates for our examples. Also, the advantages of the parametric path method will likely increase when we move to more complex models since the Jacobians in the L–B–J method grow larger and more complex faster than do the Jacobians in the parametric path method.

By transitivity, the parametric path method also outperforms Fair–Taylor by orders of magnitude. Again, the key factor is that the Fair–Taylor evaluates the Euler equations at each $0 < t < T$. Furthermore, the slow convergence of such Gauss–Jacobi schemes implies that this needs to be done many times.

The main idea behind the parametric path method is similar in spirit to Krylov methods (and their early forms, such as conjugate gradient methods). The inner loop of a Krylov method reduces a large problem to a smaller one which generates an approximation using ideas similar to our projection method. The outer loop examines a succession of finite-dimensional approximations, where the new directions are chosen to keep the smaller finite-dimensional problems well-behaved. The parametric path method also continues by examining successively larger approximation spaces until the apparent error is small. In our version of the parametric path method, the sequence of spaces used is exogenously specified, but a more refined version could endogenize the sequence of approximations used.

A closer competitor is the Mercenier and Michel (1994a) approach. The total computational effort is similar, but direct comparisons (compare their Fig. 1 with our Tables 5–7) show that the parametric path method produces substantially more accurate solutions, particularly at early t, than the Mercenier–Michel approach. However, the examples used in Mercenier and Michel (1994a) and here are too simple to draw any firm conclusions. In particular, combining the time aggregation approach in Mercenier and Michel (1994a) with the L–B–J Newton-style algorithm would likely produce a good method. Comparisons of these methods using large models will be needed before we can make any judgment.

The projection equations system (23) could also be a large system. The system (23) is nearly triangular in many of our examples. This property is expected given the orthogonality of the projections and indicates that Gauss–Seidel methods, such as those used in Hughes Hallet and Piscitelli could be applied to (23) if (23) was too large for Newton's method. If we used a finite-element approach, (23) would be sparse and we could use the Newton-style methods used in Juillard et al. (1998) and Gilli and Pauletto (1998). The main accomplishment of the parametric path method is the more efficient reduction in dimensionality. The reduced system can still take advantage of many other techniques for solving large systems.

4. Conclusion

The parametric path method offers a new approach to solving perfect-foresight models. It parsimoniously parameterizes the time path of the unknown economic variables and then solves for the unknown coefficients. The parametric path method combines methods from approximation theory, numerical integration, and nonlinear equations to quickly find a parsimonious approximation to the dynamic equilibrium. Some simple examples show that these parsimonious approximations can also be excellent approximations. The algorithm is more flexible than most alternatives. In particular, it does not just apply to economic problems which reduce to sparse systems of nonlinear equations. The ultimate value of the parametric path method can only be determined by its applications to nontrivial problems, but the flexibility of the method indicates substantial promise.

Acknowledgements

The author wants to thank Michel Juillard, an anonymous referee, and participants at the 2001 meeting of the Society for Computational Economics for valuable comments, and the National Science Foundation (grant SBR-9708991) for financial support.

References

Auerbach, A.J., Kotlikoff, L.J., Skinner, J., 1987. The efficiency gains from dynamic tax reform. International Economic Review 24, 81–100.

Boucekkine, R., 1995. An alternative methodology for solving nonlinear forward-looking models. Journal of Economic Dynamics & Control 19, 711–734.

Boyd, J.P., 1989. Chebyshev and Fourier Spectral Methods. Springer, Berlin.

Fair, R.C., Taylor, J.B., 1983. Solution and maximum likelihood estimation of dynamic nonlinear rational expectations models. Econometrica 51, 1169–1185.

Fisher, P.G., 1992. Rational Expectations in Macroeconomic Models. Kluwer Academic Publishers, Dordrecht.

Fisher, P.G., Holly, S., Hughes Hallett, A.J., 1986. Efficient solution techniques for dynamic non-linear rational expectations models. Journal of Economic Dynamics & Control 10, 139–145.

Gilli, M., Pauletto, G., 1998. Nonstationary iterative methods for solving models with forward looking variables. Journal of Economic Dynamics & Control 22, 1275–1289.

Gustafson, R.L., 1958. Carryover levels for grains: a method for determining amounts that are optimal under specified conditions. USDA Technical Bulletin 1178.

Hall, S.G., 1985. On the solution of large economic models with consistent expectations. Bulletin of Economic Research 37, 157–161.

Hughes Hallett, A.J., Piscitelli, L., 1998. Simple reordering techniques for expanding the convergence radius of first order iterative techniques. Journal of Economic Dynamics & Control 22, 1319–1333.

Judd, K.L., 1992. Projection methods for solving aggregate growth models. Journal of Economic Theory 58, 410–452.

Judd, K.L., 1998. Numerical Methods in Economics. MIT Press, Cambridge, MA.

Juillard, M., 1996. DYNARE: a program for the resolution and simulation of dynamic models with forward variables through the use of a relaxation algorithm, CEPREMAP Working Paper, No. 9602, Paris, France.

Juillard, M., Laxton, D., McAdam, P., Pioro, H., 1998. An algorithm competition: first-order iterations versus Newton-based techniques. Journal of Economic Dynamics & Control 22, 1291–1318.

Kelley, C.T, 1995. Iterative Methods for Linear and Nonlinear Equations. SIAM, Philadelphia, PA.

Laffargue, J.P., 1990. Resolution d'un mode'le macroeconomique avec anticipations rationnelles. Annales d'Economie et Statistique 17, 97–119.

Mercenier, J., Michel, P., 1994a. Discrete-time finite horizon approximation of infinite horizon optimization problems with steady-state invariance. Econometrica 62, 635–656.

Mercenier, J., Michel, P., 1994b. A criterion for time aggregation in intertemporal dynamic models. Mathematical Programming 64, 179–197.

Miranda, M.J., Helmberger, P.G., 1988. The effects of commodity price stabilization programs. American Economic Review 78, 46–58.

Saad, Y., 1996. Iterative Methods for Sparse Linear Systems. PWS Publishing Company, Boston, MA.

Wright, B.D., Williams, J.C., 1984. The welfare effects of the introduction of storage. Quarterly Journal of Economics 99, 169–182.

Young, D.M., 1971. Iterative Solution of Large Linear Systems. Academic Press, New York.