Finite-Difference Methods for Continuous-Time Dynamic Programming

Graham V. Candler
Aerospace Engineering and Mechanics
University of Minnesota
Minneapolis, MN 55455

Introduction

In this paper, we discuss the use of finite-difference methods for the simulation of dynamic economies. We consider a standard stochastic dynamic programming model of a macroeconomy and apply the finite-difference methods to this problem. This model results in a second-order, non-linear partial differential equation that has some features in common with the governing equations of fluid dynamics. We will introduce the idea of "upwind" or type-dependent differencing methods and discuss the stability of the methods through the analysis of model problems. Then, an implicit solution to the problem is discussed with the motivation of reducing the computer time required to solve the problem. Finally, we consider a two-state dynamic programming problem.

Stochastic Dynamic Programming

Consider the following dynamic programming problem for an infinitely-lived representative agent who maximizes expected utility

$$\begin{aligned} &\max_c E \int_0^\infty e^{-\rho t}\, u(c)\, dt \\ &\text{s.t.} \quad dk = \big(f(k) - c\big)\, dt + \sigma(k)\, dz, \quad k(0) \text{ given} \end{aligned} \tag{1}$$

where c is consumption, k is the capital stock, and z is Brownian motion. The general dynamic programming equation for the value function, $V(t,k)$, is

$$-V_t + \rho V = \max_c \Big\{ u(c) + V_k\big(f(k) - c\big) + \tfrac{1}{2}\sigma^2 V_{kk} \Big\}, \tag{2}$$

where V_t and V_k are first derivatives of V with respect to time and capital, and V_{kk} is the second derivative of V with respect to capital. Let the utility function have the form $u(c) = c^\omega/\omega$, assume a production function $f(k) = \lambda k^\theta - \delta k$, and choose $\sigma(k) = \sigma k$. Then, using the first-order condition, $u'(c) = V_k = c^{\omega-1}$, we obtain the second-order, time-dependent partial differential equation

$$V_t + \rho V = \frac{1-\omega}{\omega} V_k^{\omega/\omega-1} + V_k\big(\lambda k^\theta - \delta k\big) + \tfrac{1}{2}\sigma^2 k^2 V_{kk}. \tag{3}$$

In going from (2) to (3), we have changed the sign on V_t for the following reason. In time-dependent problems, a terminal condition at some final date is given and the solution of $V(t,k)$ is found by solving the problem backwards in time. To avoid cumbersome notation, we simply change the sign on V_t and integrate (3) with a positive time step; this is equivalent to integrating the original equation backwards in time. The derivation and solution of this type of continuous-time dynamic programming problem is discussed in Kamien and Schwartz (1981).

It is useful to consider exact solutions to this problem. With no stochastic term ($\sigma = 0$) in the steady-state, $V_t = 0$ and $k = k_s$, the steady-state value of the capital stock. We can solve the first-order conditions of original problem

$$\begin{aligned} f'(k_s) &= \rho, \\ \theta\lambda k_s^{\theta-1} - \delta &= \rho, \end{aligned} \tag{4}$$

to obtain

$$\begin{aligned} k_s &= \left(\frac{\rho+\delta}{\lambda\theta}\right)^{1/\theta-1}, \\ c_s &= \lambda k_s^{\theta} - \delta k_s. \end{aligned} \tag{5}$$

In the special case case where $\theta = \omega = \frac{1}{2}$, we have

$$\begin{aligned} V &= \frac{\phi^{\omega-1}}{\omega}k^{\omega} + \frac{\lambda}{\rho}\phi^{\omega-1} \\ c &= V_k^{1/\omega-1} = \phi k \\ \phi &= \frac{\rho+\delta\omega}{1-\omega} \end{aligned} \tag{6}$$

This solution will be useful for evaluating the numerical solutions obtained with the finite-difference methods.

Finite-Difference Solution of $V(t,k)$

We would like to find the function $V(k)$ that satisfies (3) when the time derivative, V_t, is zero. We can find this time independent solution in two ways. The most obvious thing to do is to set $V_t = 0$, and solve the functional equation

$$\rho V = \frac{1-\omega}{\omega}V_k^{\omega/\omega-1} + V_k\big(\lambda k^{\theta} - \delta k\big) + \tfrac{1}{2}\sigma^2 k^2 V_{kk} \tag{7}$$

subject to boundary conditions at $k = 0$ and $k = k_{max}$ (a boundary value problem).

Alternatively, we can solve the time-dependent partial differential for the time independent solution, with an initial condition and subject to the boundary conditions (an initial boundary value problem). In this case, we start with an arbitrary initial condition and integrate in time until the solution is no longer a function of the initial condition.

In computational fluid dynamics (CFD), the time dependent solution is preferred. By solving the time-dependent equations starting with a reasonable initial condition, we solve along a physically-consistent trajectory in space-time (or in this case, capital stock-time) to the final solution. This is generally more efficient and less prone to numerical problems than trying to find the roots of the non-linear functional equation given by (7).

Finite-Difference Methods

There are many different approaches for the numerical solution of partial differential equations. In this paper, we focus on finite-difference methods; this approach is the most straight-forward to develop and implement, and the other methods have many things in common with this method.

The finite-difference method is based on the use of Taylor series expansions to represent the partial derivatives that appear in the dynamic programming problem. Let us approximate the function $V(t,k)$ on an equispaced time and capital-stock grid

$$V(t,k) = V(t_n, k_i) = V(n\,\Delta t, i\,\Delta k) = V_i^n, \tag{8}$$

where $n = 0, 1, 2, \ldots$ and $i = 0, 1, 2, \ldots, i_{max}$. Using the definition of the partial derivative, we can represent the derivative of the value function with respect to the capital stock as

$$V_k(t,k) = \frac{\partial V}{\partial k} = \lim_{\Delta k \to 0} \frac{V(t, k+\Delta k) - V(t,k)}{\Delta k}. \tag{9}$$

A Taylor series approximation for $V(t, k+\Delta k)$ is given by

$$V(t, k+\Delta k) = V(t,k) + \Delta k\, V_k(t,k) + \tfrac{1}{2}\Delta k^2\, V_{kk}(t,k) + \ldots. \tag{10}$$

And therefore, we can show that the partial derivative at $t = n\,\Delta t$ and $k = i\,\Delta k$ is

$$\begin{aligned} V_k(t,k) = (V_k)_i^n = & \frac{V(t,k+\Delta k) - V(t,k)}{\Delta k} - \tfrac{1}{2}\Delta k\, V_{kk}(t,k) + \ldots \\ = & \frac{V_{i+1}^n - V_i^n}{\Delta k} + O(\Delta k). \end{aligned} \tag{11}$$

The first non-zero term in the approximation, $-\frac{1}{2}\Delta k\, V_{kk}$, is the scheme's truncation error, and the variation of the error term with the grid spacing gives the order of accuracy. Thus, this is a first-order accurate forward-difference approximation to V_k.

Similarly, we can derive the first-order accurate backward difference and second-order accurate central difference approximations

$$
\begin{aligned}
(V_k)_i^n &= \frac{V_i^n - V_{i-1}^n}{\Delta k} + O(\Delta k), \\
(V_k)_i^n &= \frac{V_{i+1}^n - V_{i-1}^n}{2\Delta k} + O(\Delta k^2),
\end{aligned} \tag{12}
$$

respectively. We can derive approximations of arbitrary order of accuracy by taking linear combinations of Taylor series expansions of $V_{i\pm m}$. Generally, the more grid points used in the difference operator, the higher the accuracy of the approximation. Also, central-differences are more accurate than biased approximations. We can also derive higher-order derivatives using the same approach. For example, a second-order accurate central difference approximation to the second derivative is

$$
(V_{kk})_i^n = \frac{V_{i+1}^n - 2V_i^n + V_{i-1}^n}{\Delta k^2} + O(\Delta k^2). \tag{13}
$$

Figure 1 gives a graphical representation of the three first-derivative approximations discussed above. For this smoothly varying function, we can see that the central difference method gives a more accurate approximation than the first-order, one-sided differences. Thus in principle, the higher the order of accuracy, the fewer grid points are required to obtain a solution of a given accuracy. Or, for a fixed grid the solution will be more accurate. However, we will see that this is an overly simplified view of the quality of the solution.

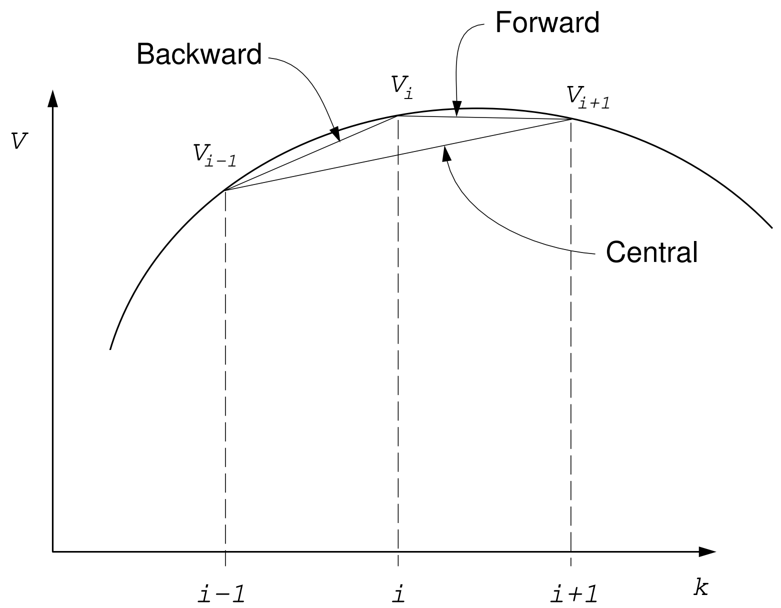

FIGURE 1. Finite-difference approximations to the first derivative of a smoothly-varying function.

Finite-Difference Approximation of $V(t,k)$

We next consider the derivation of a finite-difference approximation to the time-dependent differential equation derived above. For ease of solution, let us use a backward-difference approximation for V_t. This is an explicit time integration scheme, because we can solve explicitly for the future solution (at $n+1$), given the present solution (at n). We will discuss other implicit time integration approaches later. Thus, we have

$$(V_t)_i^n = \frac{V_i^{n+1} - V_i^n}{\Delta t} + O(\Delta t). \tag{14}$$

From the preceding analysis, we know that the central-difference approximations to the derivatives with respect to the capital stock are more accurate than the one-sided difference approximations given the same grid spacing. Therefore, let us use the central-difference approximations for V_k and V_{kk}. We then obtain the difference equation

$$\begin{aligned} \frac{V_i^{n+1} - V_i^n}{\Delta t} + \rho V_i^n = \frac{1-\omega}{\omega}\left(\frac{V_{i+1}^n - V_{i-1}^n}{2\Delta k}\right)^{\omega/\omega-1} + \left(\frac{V_{i+1}^n - V_{i-1}^n}{2\Delta k}\right)\left(\lambda k_i^\theta - \delta k_i\right) \\ + \tfrac{1}{2}\sigma^2 k_i^2 \left(\frac{V_{i+1}^n - 2V_i^n + V_{i-1}^n}{\Delta k^2}\right) + O(\Delta t) + O(\Delta k^2). \end{aligned} \tag{15}$$

Note that we can solve explicitly for $V_i^{n+1}, \quad i = 0, 1, 2, \ldots, i_{max}$, given V_i^n. We also need to specify some boundary conditions at the locations $i = 0$ and $i = i_{max}$, corresponding to $k = k_{min}$ and $k = k_{max}$, respectively. This will be discussed later.

Therefore, the solution of the differential equation is now straight-forward. We define a grid over the relevant region of the capital stock ($k_{min} \le k \le k_{max}$) by choosing a maximum number of grid points, i_{max}. For the case considered here, let $k_{min} = 0$, $k_{max} = 2k_s$. Then $\Delta k = \frac{k_{max}}{i_{max}}$, $k_i = i\,\Delta k$. Next, we specify and initial condition, V_i^0; for this example, we know that the value function is concave, and therefore we choose $V_i^0 = \sqrt{k_i}$.

Then, the solution of the difference equation proceeds with the following steps:

- Compute the time step:

$$\Delta t = \mathrm{CFL}\frac{\Delta k}{a}$$

  (CFL is a constant and a is a characteristic rate of change of the solution; to be discussed later)
- Use (15) to compute the change in the solution: $\Delta V_i^n = V_i^{n+1} - V_i^n, \quad$ for all i
- Compute the residual (L_2 error norm): $\varepsilon^n = \frac{1}{i_{max}\,\Delta t}\sqrt{\sum_i \left(\Delta V_i^n\right)^2}$
- Update the solution: $V_i^{n+1} = V_i^n + \Delta V_i^n$
- Continue until ε^n is small (the solution is no longer changing)

Results Using Central Difference Method

Consider the case where we use (15) to solve the dynamic programming problem with parameters chosen such that we know the exact solution (given by (6)). We choose $\omega = \theta = 0.5$, $\rho = 0.05$, $\lambda = 1.0$, $\delta = 0.025$, $\sigma = 0.0$, and use the exact solution as the initial condition. In this case, the finite-difference approximation should preserve the exact solution within the differencing error, $O(\Delta x^2)$.

Figure 2 plots the results of this calculation after 40 time steps. We see that the numerical approximation is completely corrupted, with huge oscillations in the solution near $k = 0$. Clearly, this is not a good method.

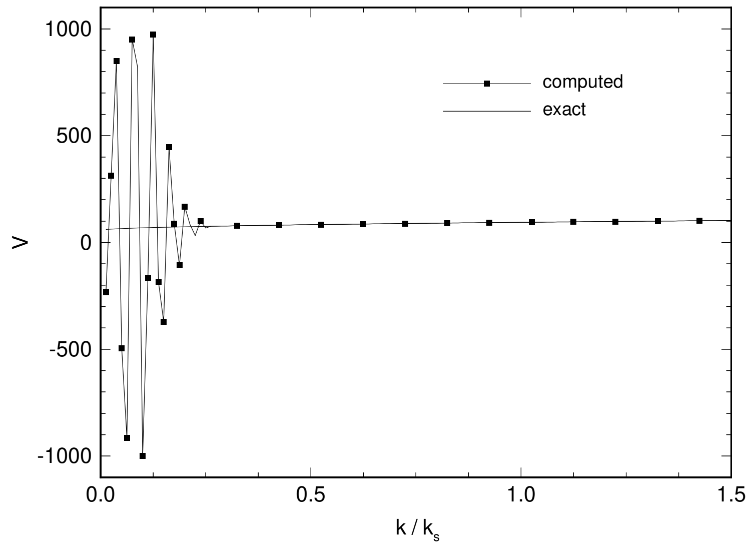

FIGURE 2. Numerical solution of the dynamic programming problem using a central-difference method after 40 time steps, starting from the exact solution.

Model Equation

Let us consider a model equation that has the essential characteristics of the differential equation (15), without the non-linearity. Note that with $\sigma = 0$, there are only first derivatives in the expression for V_t. Therefore, it is useful to consider the wave equation given by

$$V_t + aV_k = 0, \tag{16}$$

with a constant. This equation is linear and has an exact solution. For the case where $a > 0$, the initial data are simply propagated in the direction of increasing k, as illustrated in Fig. 3. That is, the solution is $V(t, k) = V(0, k - at)$ with $V(0, k)$ given.

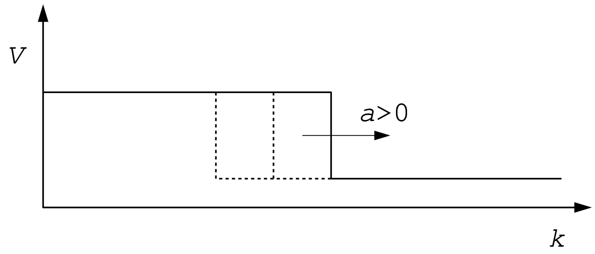

FIGURE 3. Solution of the wave equation (16) for $a > 0$.

Let us try solving this problem with the central, backward, and forward difference approximations. For example, the central difference approximation is

$$\frac{V_i^{n+1} - V_i^n}{\Delta t} = -a\frac{V_{i+1}^n - V_{i-1}^n}{2\Delta k}. \tag{17}$$

If we start with a step function ($V = 1$ for $k < 0$, and $V = 0$ for $k > 0$), and use a time step given by $\Delta t = \frac{\Delta k}{a}$, we obtain the results plotted in Fig. 4 after 5 time steps. In this case, the backward difference gives the exact solution and the central- and forward-difference methods produce spurious oscillations similar to those obtained in Fig. 3. These oscillations continue to grow with more time steps.

If a is negative, the solution propagates in the direction of negative capital stock; in this case, the forward-difference method gives the exact solution and the other two methods produce spurious oscillations. Clearly, the trajectory of the solution in capital stock-time determines which differencing scheme should be used. We will see that in general, the differencing stencil should use data that reflects the direction of information propagation. That is, if the solution is moving to the right in capital stock-time, the differencing stencil should be biased to the left, and *vice versa.* In aerodynamics terminology, this biasing is known as “upwind” differencing. In the next section, we show that there is a mathematical basis for this approach.

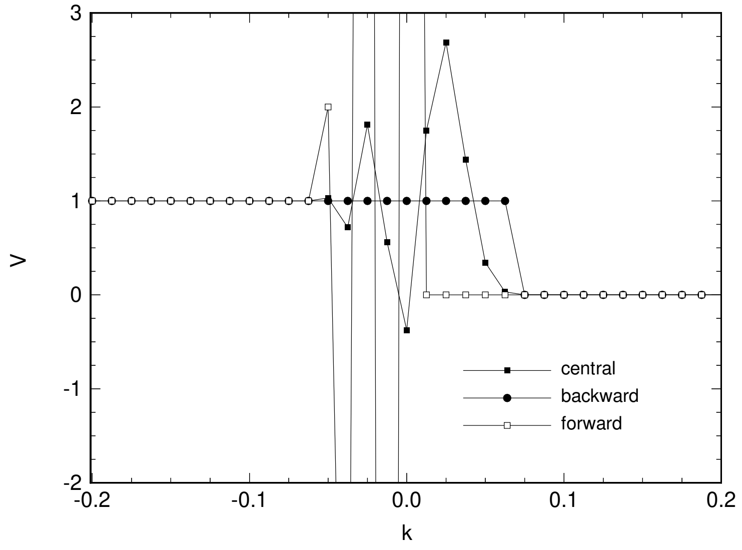

FIGURE 4. Numerical solution of the wave equation (16) for $a > 0$ for three different finite-difference approximations after 5 time steps.

Von Neumann Stability Analysis

In the previous examples, we saw that depending on the differencing scheme, the error may be amplified. It is possible to formalize the numerical results discussed above by computing the amplification factor of a finite-difference method. To do so, we decompose the solution, V_i^n, into the exact solution, $\bar{V}_i^n$, and the error, ϵ_i^n

$$V_i^n = \bar{V}_i^n + \epsilon_i^n. \tag{18}$$

Then, we substitute this expression into the differencing method and solve for the evolution of the error.

Let us consider the central difference method applied to the wave equation; the finite-difference approximation is given by (17). Since $\bar{V}_i^n$ satisfies the equation exactly, we obtain the equation for the errors

$$\frac{\epsilon_i^{n+1} - \epsilon_i^n}{\Delta t} = -\frac{a}{2\Delta k}\left(\epsilon_{i+1}^n - \epsilon_{i-1}^n\right). \tag{19}$$

Because the equation is linear, we can decompose the solution into its Fourier components. If all possible Fourier components of the error are not amplified by the differencing method, then the error as a whole is not amplified. Thus, represent the error as a Fourier series

$$\epsilon_i^n = \sum_{j=-N}^{N} E_j^n e^{\imath\, i\phi(j)},$$

where $\imath = \sqrt{-1}$ and E_j^n is the amplitude of the j^{th} harmonic. $\phi = j\pi/N$ is the phase angle, where $-\pi \le \phi \le \pi$. If we substitute this expression for the error into the difference equation, we can determine how each harmonic is amplified by the scheme. For the jth harmonic, we have

$$\begin{aligned} \frac{E^{n+1} - E^n}{\Delta t} e^{\imath\, i\phi} &= -\frac{a}{2\Delta k}\left(E^n e^{\imath\,(i+1)\phi} - E^n e^{\imath\,(i-1)\phi}\right), \\ E^{n+1} - E^n &= -\frac{a\Delta t}{2\Delta k} E^n \left(e^{\imath\,\phi} - e^{-\imath\,\phi}\right). \end{aligned} \tag{20}$$

If we define the amplification factor, G, as

$$G = \frac{E^{n+1}}{E_n}, \tag{21}$$

the error will not grow over time if

$$|G| = \left|\frac{E^{n+1}}{E_n}\right| \le 1, \qquad \text{for all } \phi. \tag{22}$$

For the case of the central-difference method, we have

$$\begin{aligned} G &= 1 - \imath\,\frac{a\Delta t}{\Delta k}\sin\phi, \\ |G|^2 &= 1 + \left(\frac{a\Delta t}{\Delta k}\right)^2 \sin^2\phi. \end{aligned} \tag{23}$$

Therefore, $|G| > 1$ for non-zero time steps, and the error is always amplified. Thus, the solution is always unstable, and the scheme is *unconditionally unstable.*

For the backward difference scheme applied to the wave equation with a positive a, the amplification factor is

$$G = 1 - 2\frac{a\Delta t}{\Delta k}\sin^2\frac{\phi}{2} - \imath\,\frac{a\Delta t}{\Delta k}\sin\phi, \tag{24}$$

and the scheme is stable if

$$0 < \frac{a\Delta t}{\Delta k} \le 1. \tag{25}$$

This result is commonly known as the Courant-Friedrichs-Lewy or CFL condition. The quantity, $\frac{a\Delta t}{\Delta k}$, is known as the CFL number or the Courant number, and represents a non-dimensional time step. The backward difference method is said to be *conditionally stable.*

The decomposition of the error into its Fourier components is the basis of von Neumann stability analysis. It should be noted that it relies on the linearity of the governing equations; for non-linear problems, it is not possible to isolate the individual Fourier components. Thus, von Neumann stability analysis only serves as a guide to the solution of non-linear equations. However, one can be assured that if the differencing scheme is not suitable for a linear problem, it will not be suitable for an analogous (non-constant coefficient) non-linear problem. In practice, non-linear problems usually have a more restrictive time step limit than that implied by the CFL condition.

Note that for the conditionally stable schemes, the maximum stable time step is proportional to the grid spacing, Δk. Thus, as the grid is refined to reduce the error in the solution, the maximum stable time step is also reduced. This typically results in more time steps to produce a converged steady-state solution. To counter this, we can derive an implicit time integration method that, at least for the linear wave equation, does not have this stringent time step limit. Consider the following differencing scheme

$$\frac{V_i^{n+1} - V_i^n}{\Delta t} = -\frac{a}{2\Delta k}\left(V_{i+1}^{n+1} - V_{i-1}^{n+1}\right) + O(\Delta x^2) + O(\Delta t). \tag{26}$$

This method uses a central-difference approximation for the k derivative, but the derivative is evaluated at the future time level, $n+1$. This is an implicit scheme because the solution at the new time level cannot be obtained explicitly from the solution at the present time level. Rather, it must be computed implicitly; in this case, we must solve a tridiagonal system of equations.

The amplification factor of this implicit method is given by

$$G = \frac{1}{1 + \imath\,\frac{a\Delta t}{\Delta k}\sin\phi}, \tag{27}$$

which always satisfies the stability condition. Therefore, the scheme is *unconditionally stable* for the linear wave equation. In principle, we can take infinite time steps and obtain a stable solution. This solution will not be time accurate because the differencing scheme is first-order in time. However, this is not a concern because we are only interested in the steady-state solution which is not a function of time. Therefore, any stable time step may be used to obtain the converged solution. For non-linear problems, the method will be stable up to a limiting time step, but this time step cannot be predicted from von Neumann stability analysis.

Upwind Differencing Method for $V(t,k)$

The foregoing analysis can be used to develop methods that sense the appropriate direction of differencing, and locally adjust the differencing scheme. For the linear wave equation, we saw that when $a > 0$ we use a backward difference, and when $a < 0$ we use a forward difference. We can develop a scheme that automatically switches the differencing method depending on the local coefficient on V_k. For example if $a = a(k)$

$$\frac{V_i^{n+1} - V_i^n}{\Delta t} = -\frac{a_i^+}{\Delta k}\big(V_i^n - V_{i-1}^n\big) - \frac{a_i^-}{\Delta k}\big(V_{i+1}^n - V_i^n\big) + O(\Delta x) + O(\Delta t), \tag{28}$$

where a_i^+ and a_i^- are determined by sign of a

$$\begin{aligned} a_i^+ &= \tfrac{1}{2}\big(a_i + |a_i|\big) \\ a_i^- &= \tfrac{1}{2}\big(a_i - |a_i|\big) \end{aligned} \tag{29}$$

Thus, $a^+ = a$ and $a^- = 0$ when $a > 0$, and *vice versa*, and we obtain the correct biasing of the difference approximation for all values of a. This is an "upwind" method because the differencing scheme uses data from the upwind direction (in the opposite direction of the solution's trajectory in capital stock-time).

We can now return to our dynamic programming problem, and derive an upwind finite-difference approximation to the differential equation. Let us group terms in (3) to get a single coefficient on V_k

$$V_t + \rho V = \left(\frac{1-\omega}{\omega}V_k^{1/\omega - 1} + \lambda k^\theta - \delta k\right)V_k + \tfrac{1}{2}\sigma^2 k^2 V_{kk}. \tag{30}$$

Then, the first-order upwind difference method is

$$\begin{aligned} \frac{V_i^{n+1} - V_i^n}{\Delta t} + \rho V_i^n = &-\frac{a_i^+}{\Delta k}\big(V_i^n - V_{i-1}^n\big) - \frac{a_i^-}{\Delta k}\big(V_{i+1}^n - V_i^n\big) \\ &+ \tfrac{1}{2}\sigma^2 k_i^2\left(\frac{V_{i+1}^n - 2V_i^n + V_{i-1}^n}{\Delta k^2}\right) + O(\Delta t) + O(\Delta k) \end{aligned} \tag{31}$$

where

$$a_i = -\frac{1-\omega}{\omega}(V_k)_i^{1/\omega - 1} - \lambda k_i^\theta + \delta k_i \tag{32}$$

This is an extension of the upwind approach; in this case, the coefficient depends on k and V_k. Of course, since this is a non-linear equation, there is no theory that dictates that this is a stable method. It is simply a matter of seeing if the method works in practice. However, we have a much better chance of getting the method to work because it is based on a method that is (conditionally) stable for a linear model equation. It is clear that the previous central-difference method, (15), did not have a chance of working.

The second derivative term in (3) has been represented with a central-difference approximation. This was chosen because a von Neumann stability analysis for the model equation

$$V_t - \alpha V_{kk} = 0 \tag{33}$$

shows that a central difference approximation is conditionally stable. The time step is governed by the Fourier number, $\mathrm{Fo} = \frac{\alpha \Delta t}{\Delta k^2}$. The scheme is stable for

$$0 \leq \frac{\alpha \Delta t}{\Delta k^2} \leq \frac{1}{2}, \tag{34}$$

which implies that $\alpha \geq 0$, and $\mathrm{Fo} \leq \frac{1}{2}$. Generally, the second derivative terms are less difficult to approximate because they are diffusive in nature, and do not have a preferred direction.

Numerical Results

Figure 5 plots the results of a calculation using the upwind method started from an initial guess for V_i^0, and run until the residual is small. The parameters used are: $\omega = \theta = 0.5$, $\rho = 0.05$, $\lambda = 1.0$, $\delta = 0.025$, $\sigma = 0.0$. We see that the solution is very close to the exact solution. Figure 6 plots the consumption function, $V_k^{1/\omega - 1}$, obtained from the numerical solution. Again, the numerical solution predicts the exact solution very well.

These calculations show that the finite-difference methods can indeed be used to solve the partial differential equations that govern the dynamic economies. However, the differencing methods must be chosen carefully, and the most obvious method is not always the best method.

At this point, it is useful to discuss the accuracy of the finite-difference solution of the dynamic programming problem. We know the exact value of the steady-state value of the capital stock, k_s, from (5). Thus, we can compare the computations to this result for various values of ω and different numbers of grid points. This is done in Fig. 7, and we see that the first-order accurate finite difference approximation gives an accurate prediction of k_s for large values of ω. However, at $\omega = 0.1$, a large number of grid points has to be used to get a small error in k_s. This is because the solution, $V(k)$, increases in non-linearity with decreasing ω. This is an illustration that this first-order accurate method has a truncation error that is proportional to $\frac{1}{2} \Delta k V_{kk}$. Note also from Fig. 7 that for a given value of ω, the error is reduced by a factor of two by doubling the grid size. This motivates the development of higher-order accurate methods that have a more rapid error reduction with grid spacing.

Figure 8 presents the results of the finite-difference calculation for the case where $\sigma \neq 0$ with $\omega = 0.5$, and all of the other parameters unchanged. We see that the consumption function decreases at large values of capital stock as expected.

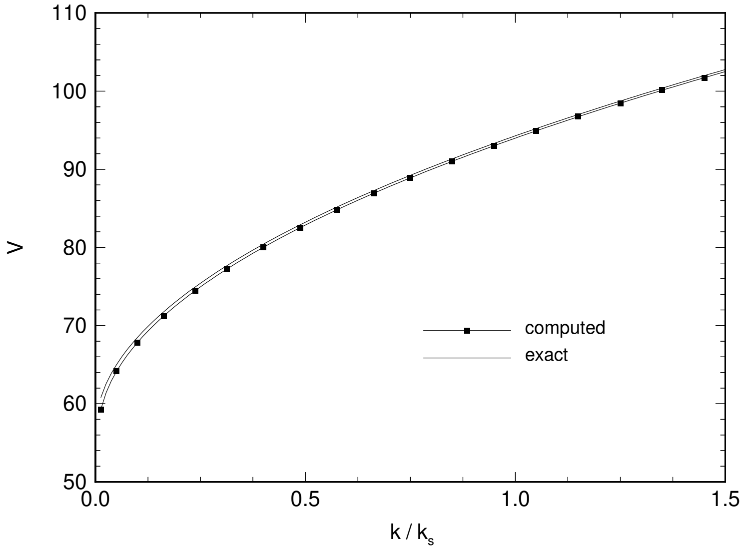

FIGURE 5. Numerical solution of the dynamic programming problem (3) using an upwind differencing method. $\omega = \theta = 0.5$, $\rho = 0.05$, $\lambda = 1.0$, $\delta = 0.025$, $\sigma = 0.0$.

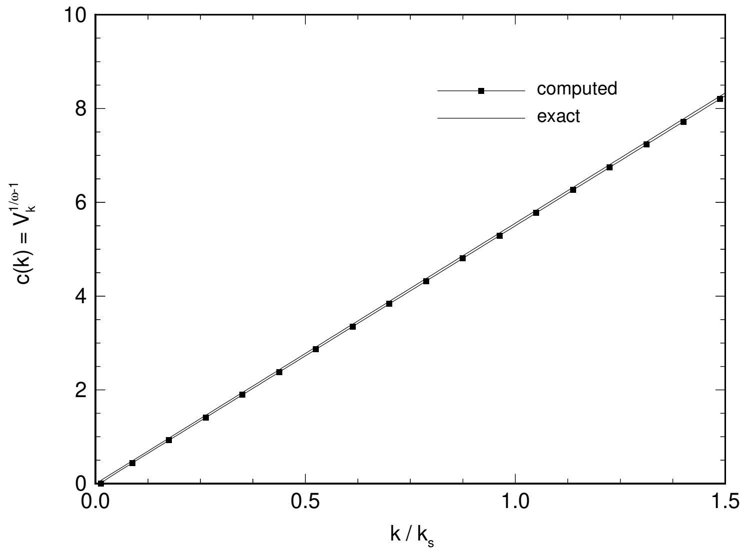

FIGURE 6. Consumption function, $V_k^{1/\omega-1}$, computed from the numerical solution of the dynamic programming problem.

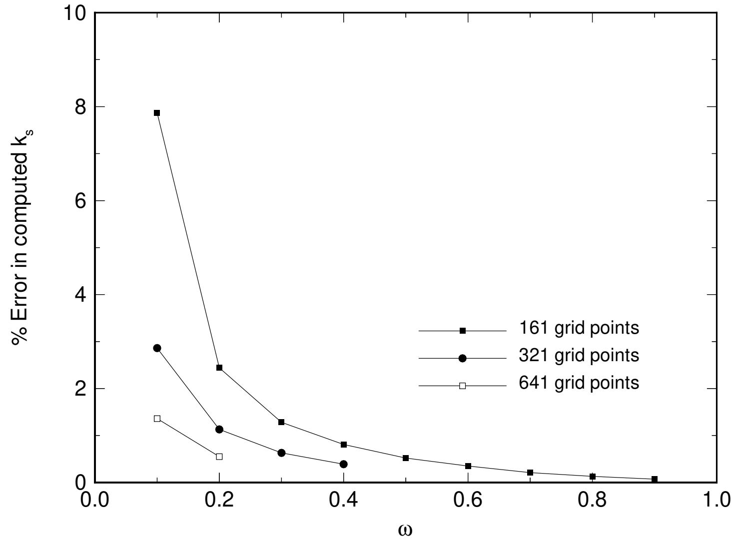

FIGURE 7. Variation of error in the computed steady-state capital stock, k_s, as a function of ω. $\theta = 0.4$, $\rho = 0.05$, $\lambda = 1.0$, $\delta = 0.025$, $\sigma = 0.0$.

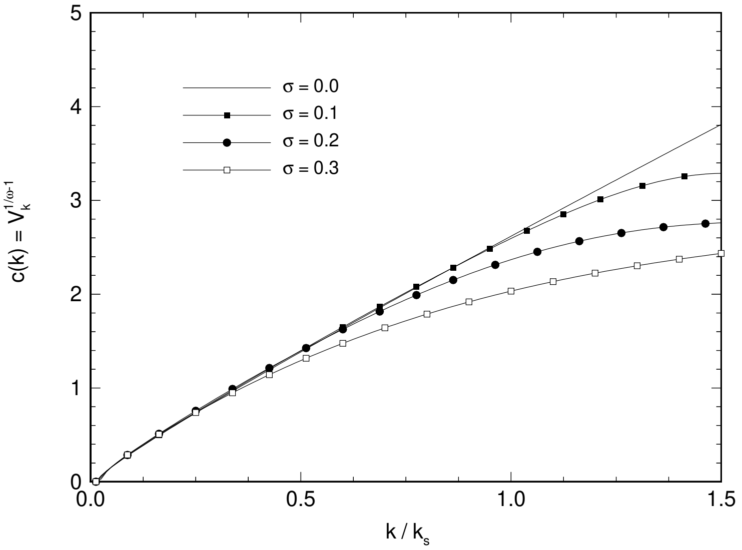

FIGURE 8. Dependence of computed consumption function on σ, with $\omega = 0.5$, $\theta = 0.4$, $\rho = 0.05$, $\lambda = 1.0$, $\delta = 0.025$.

Implicit Method for $V(t,k)$

The previous results show that the upwind-differencing method can be used to solve the dynamic programming problem. However, these calculations can be costly if the grid is highly refined or the problem is extended to multiple dimensions. The computational cost is a result of starting with an initial condition, and integrating the differential equation in time until the solution is time independent. With an explicit method, the time step of this integration is limited because of the stability constraints. Thus, if we can develop a method that has a less stringent time step limitation, we should be able to converge to a steady state in fewer time steps and with less computer time.

We discussed an implicit method above and saw that for the linear wave equation it is unconditionally stable. In this section, we develop a similar implicit method for the non-linear dynamic programming problem and test its performance. We use the same upwind differencing method for derivatives with respect to the capital stock and evaluate those derivatives at the future $(n+1)$ time level. This results in the difference equation

$$\begin{aligned}\frac{V_i^{n+1}-V_i^n}{\Delta t}+\rho V_i^{n+1} &= -\frac{a_i^+}{\Delta k}\big(V_i^{n+1}-V_{i-1}^{n+1}\big)-\frac{a_i^-}{\Delta k}\big(V_{i+1}^{n+1}-V_i^{n+1}\big)\\ &\quad+\tfrac{1}{2}\sigma^2k_i^2\left(\frac{V_{i+1}^{n+1}-2V_i^{n+1}+V_{i-1}^{n+1}}{\Delta k^2}\right)+O(\Delta t)+O(\Delta k).\end{aligned} \tag{35}$$

Note that the coefficients on the first-derivative approximations, $a_i^{\pm}$, are functions of V also. To be completely consistent, we should evaluate those terms at the future time level as well. However, this is not required to obtain the correct answer, because at the time-independent solution $V_i^{n+1}=V_i^n$. Therefore, to simplify the derivation of the implicit method, let us evaluate these quantities at the present (n) time level.

If we let $\Delta V_i^n = V_i^{n+1}-V_i^n$, we can write the implicit problem as:

$$\begin{aligned}&\big(\alpha_i^- -\beta_i\big)\Delta V_{i+1}^n+\big(1+\alpha_i^+-\alpha_i^-+2\beta_i+\Delta t\rho\big)\Delta V_i^n+\big(-\alpha_i^+-\beta_i\big)\Delta V_{i-1}^n=\\ &\qquad -\alpha_i^+\big(V_i^n-V_{i-1}^n\big)-\alpha_i^-\big(V_{i+1}^n-V_i^n\big)+\beta_i\big(V_{i+1}^n-2V_i^n+V_{i-1}\big)-\Delta t\rho V_i^n,\end{aligned} \tag{36}$$

where $\alpha^{\pm}=\frac{\Delta t\, a^{\pm}}{\Delta k}$ and $\beta=\frac{\Delta t\,\sigma^2k^2}{2\Delta k^2}$. We can see that this is a tridiagonal system of equations for ΔV_i^n. Symbolically, we have

$$\begin{pmatrix} B_0 & A_0 & C_0 & & & & \\ & \ddots & \ddots & \ddots & & & \\ & & B_i & A_i & C_i & & \\ & & & \ddots & \ddots & \ddots & \\ & & & & B_{imax} & A_{imax} & C_{imax}\end{pmatrix}^n \begin{pmatrix}\vdots\\ \Delta V_{i-1}\\ \Delta V_i\\ \Delta V_{i+1}\\ \vdots\end{pmatrix}^n = \begin{pmatrix}\vdots\\ RHS_i\\ \vdots\end{pmatrix}^n \tag{37}$$

Again, the solution cannot be computed explicitly from the present solution, however the tridiagonal system is easy and efficient to solve.

Note that because of the upwind-differencing approach for the k derivatives, the system is diagonally dominant for any Δt. Thus, there are many approximate solution methods for the system of equations. For example, we could move the off-diagonal terms to the right-hand side, and iteratively include them in the solution using a series of relaxation steps. Because of the diagonal dominance of the system, this approach will converge. Also, we could completely neglect the off-diagonal terms, and again this method will converge. In any case, the time-dependent answer is independent of the implicit operator used.

The simplest approximate method is a "diagonal" method that may be derived by ignoring the off-diagonal terms

$$\begin{aligned}&\big(1+\alpha_i^+-\alpha_i^-+2\beta_i+\Delta t\rho\big)\Delta V_i^n = \\ &\quad -\alpha_i^+\big(V_i^n-V_{i-1}^n\big)-\alpha_i^-\big(V_{i+1}^n-V_i^n\big)+\beta_i\big(V_{i+1}^n-2V_i^n+V_{i-1}\big)-\Delta t\rho V_i^n.\end{aligned} \tag{38}$$

This set of equations is no more expensive to solve than the explicit method, and in some cases is considerably more effective. It also generalizes easily to multiple dimensions, unlike the full implicit method given by (36).

Comparison of Convergence Rates

Figure 9 compares the convergence rates of the explicit, implicit, and diagonal methods for the baseline parameters. We see that the implicit method converges to the final solution in the smallest number of iterations. Note that there is an optimal time step for the implicit method – in this case CFL = 10 gives the best convergence rate. At higher time steps, the method does not converge to machine zero. The diagonal method is not as effective as the more exact implicit method, but it is still at least twice as fast as the explicit method.

Even for the implicit method, about 2000 iterations are required to obtain a steady-state solution. It should be noted that a very poor initial guess for V was used and that this differential equation has a different form than those commonly solved in physics and engineering. Thus, it is very likely that with more work the convergence properties could be improved significantly. For example, the coefficient on V_k could be linearized and included in the implicit solution. Also, to reduce the number of grid points, a non-uniform grid spacing could be used to cluster grid points near regions of strong non-linearity.

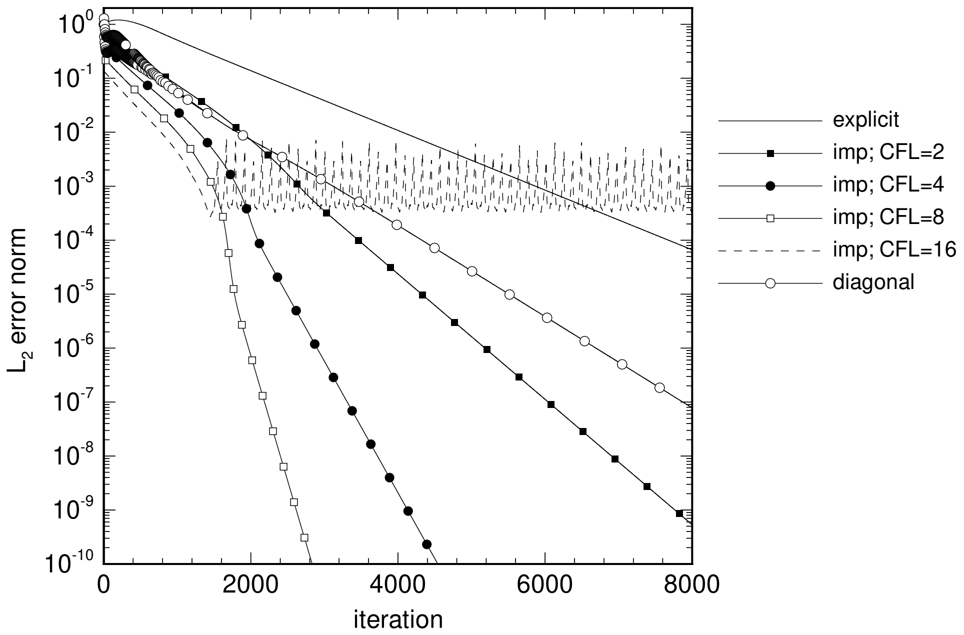

FIGURE 9. Convergence history for different time integration methods.

Upwind Method for $c(t, k)$

In this section, we discuss the solution of the differential equation for the consumption function, rather than the value function. To derive this equation, we consider the case where the stochastic term is zero, and obtain from (1) the time-dependent differential equation

$$c_t = \big(f(k) - c(k)\big)c_k + \frac{u'(c)}{u''(c)}\big(f'(k) - \rho\big). \tag{39}$$

With $u(c) = c^\omega/\omega$ and $f(k) = \lambda k^\theta - \delta k$, we obtain

$$c_t = \big(\lambda k^\theta - \delta k - c\big)c_k + \frac{1}{\omega - 1}\big(\lambda\theta k^{\theta-1} - \delta - \rho\big)c. \tag{40}$$

Using the same approach as above, an upwind finite-difference approximation to this equation is

$$\frac{c_i^{n+1} - c_i^n}{\Delta t} = -\frac{a_i^+}{\Delta k}\big(c_i^n - c_{i-1}^n\big) - \frac{a_i^-}{\Delta k}\big(c_{i+1}^n - c_i^n\big) + S_i^n, \tag{41}$$

where

$$\begin{aligned} a_i &= -\lambda k_i^\theta + \delta k_i + c_i, \\ S_i &= \frac{1}{\omega - 1}\big(\lambda\theta k_i^{\theta-1} - \delta - \rho\big)c_i. \end{aligned} \tag{42}$$

Figure 10 plots the computed and exact consumption functions. In this case, the numerical method gives the exact result for all of values of k. This is not surprising because the solution is linear in k, and therefore the first-order method has a zero truncation error.

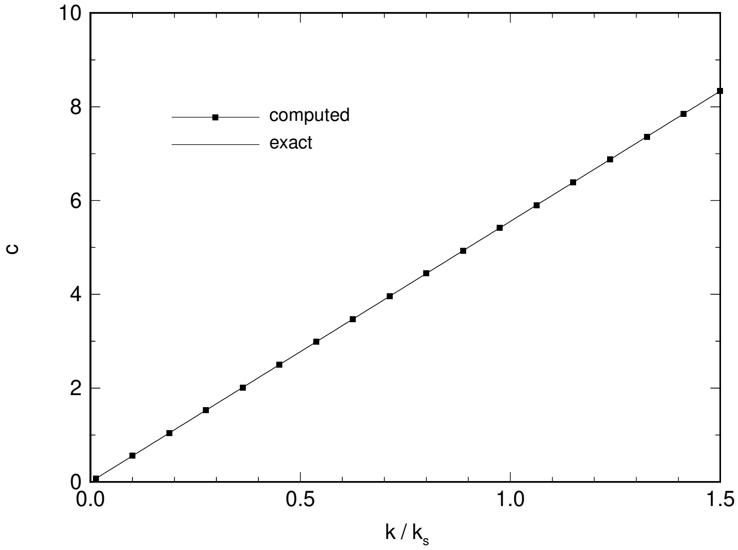

FIGURE 10. Comparison of computed consumption function with exact solution. $\omega = \theta = 0.5$, $\rho = 0.05$, $\lambda = 1.0$, $\delta = 0.025$, $\sigma = 0$.

Implicit Method for $c(t, k)$

Now let us consider an implicit time integration method for the consumption function variation. Again, evaluate all of the quantities at the future time level

$$\frac{c_i^{n+1} - c_i^n}{\Delta t} = -\frac{a_i^+}{\Delta k}\left(c_i^{n+1} - c_{i-1}^{n+1}\right) - \frac{a_i^-}{\Delta k}\left(c_{i+1}^{n+1} - c_i^{n+1}\right) + S_i^{n+1}. \tag{43}$$

To estimate the source term at the future time level, linearize using

$$S_i^{n+1} = S_i^n + \left.\frac{\partial S}{\partial c}\right|_i^n \left(c_i^{n+1} - c_i^n\right) + O(\Delta t). \tag{44}$$

Thus, the implicit approximation is

$$\begin{aligned} \alpha_i^- \Delta c_{i+1}^n + \left(1 + \alpha_i^+ - \alpha_i^- - \Delta t \frac{\partial S}{\partial c}\right)\Delta c_i^n - \alpha_i^+ \Delta c_{i-1}^n = \\ - \alpha_i^+\left(c_i^n - c_{i-1}^n\right) - \alpha_i^-\left(c_{i+1}^n - c_i^n\right) + \Delta t S_i^n. \end{aligned} \tag{45}$$

This method converges to the answer shown in Fig. 10, but the convergence rate is poor, and the maximum CFL number that can be used is about 2.

Let us consider a model equation for this problem to see if it is possible to determine the reason for this poor convergence and stringent time step limitation. A useful model equation results from the situation where c_k is small. Then

$$c_t = \frac{1}{\omega - 1}\big(\lambda\theta k^{\theta-1} - \delta - \rho\big)c = S \tag{46}$$

An implicit representation of the form used in (45) is

$$\begin{aligned} \frac{c_i^{n+1} - c_i^n}{\Delta t} &= S^{n+1} \simeq S^n + \frac{\partial S}{\partial c}\big(c^{n+1} - c^n\big), \\ c^{n+1} &= c^n + \frac{\Delta t\, S^n}{1 - s\,\Delta t}, \end{aligned} \tag{47}$$

where $s = \frac{\partial S}{\partial c}$. We can see that if $s > 0$ and Δt is large, the method will produce erroneous results. In fact, when $s\,\Delta t = 1$ the change in the solution is singular. For larger values of Δt, the solution changes in the *wrong* direction.

Because $\omega < 1$, $s = \frac{1}{\omega-1}\big(\lambda\theta k^{\theta-1} - \delta - \rho\big)$ will be positive for small values of k. Therefore, there is an inherent time step limitation given by $\Delta t < \frac{1-\omega}{\delta+\rho}$. Note that this limitation is a result of the implicit time integration method. We can remove this limitation by modifying the linearization of S so that $\frac{\partial S}{\partial c}$ is always negative. For example, we can use

$$\frac{\partial S}{\partial c} \simeq \frac{1}{\omega - 1}\lambda\theta k^{\theta-1} < 0 \tag{48}$$

It seems somewhat arbitrary to have removed these terms from the implicit operator. (Remember that by modifying the operator, we do not change the converged solution, we simply change the evolution toward the converged solution.) However, this change is motivated by the analysis of the model equation, but there is no guarantee that the method will actually work better.

In practice, this change in the method makes a huge improvement in the convergence of the method, as can be seen in Fig. 11. Now there is no time step limitation, and the implicit method converges in about 10 iterations with CFL $= 10^4$. With this modification, the diagonal method also works well, converging in about 130 iterations. Thus, the use of the model equation was instructive in showing how to improve the method.

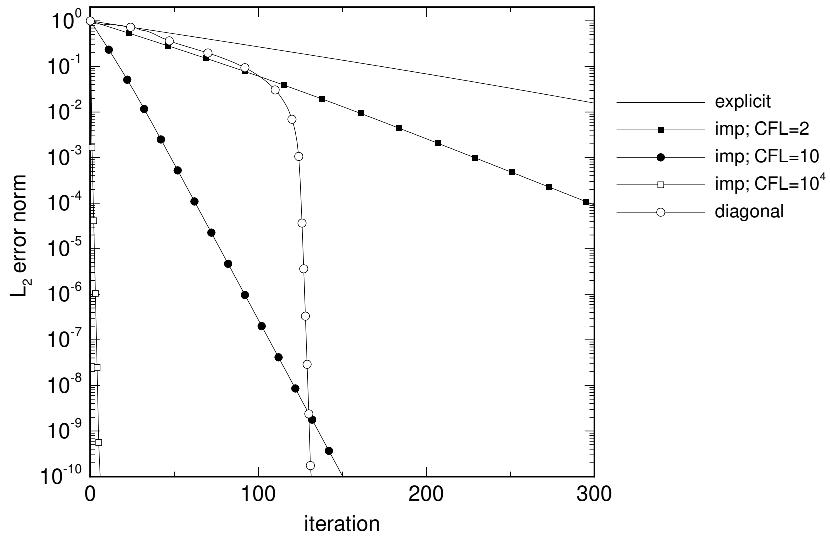

FIGURE 11. Convergence history of modified implicit method for the consumption function.

Upwind Method for a Two-State Problem

Now let us extend the finite-difference method to a problem with two states. Consider the agent's problem given by:

$$\begin{aligned} &\max_c E \int_0^\infty e^{-\rho t}\, u(c)\, dt \\ \text{s.t.}\quad dk &= \big((1-\tau)r(K)k + w(K) - c + T(K)\big)dt + \sigma(k,K)dz \\ dK &= \big(F(K) - C(K) - G(K)\big)dt + \sigma(K,K)dz \end{aligned} \tag{49}$$

where c is the individual consumption level; k is individual capital stock; $C(K)$ is aggregate consumption; K is aggregate capital stock; $F(K)$ is aggregate output; $G(K)$ is government spending; and $T(K)$ is transfers. Let:

$$u(c) = c^\omega/\omega, \quad F(k) = \lambda k^\theta - \delta k, \quad r = F_K, \quad w = F - F_K K, \quad G = \tau F_K K$$

The dynamic programming equation for the value function, $V(k,K)$, is

$$\begin{aligned} V_t + \rho V = \max_c \Big\{ u(c) &+ V_k\big((1-\tau)(\theta\lambda K^{\theta-1} - \delta)k + \lambda K^\theta(1-\theta) - c\big) \\ &+ V_K\big(\lambda K^\theta - C(K) - \tau(\theta\lambda K^\theta - \delta K)\big) \\ &+ \tfrac{1}{2}\big(\sigma(k,K)^2 V_{kk} + 2\sigma(k,K)\sigma(K,K)V_{kK} + \sigma(K,K)^2 V_{KK}\big)\Big\}. \end{aligned} \tag{50}$$

The first-order condition, $u'(c) = V_k(k, K)$, gives the differential equation

$$
\begin{aligned}
V_t + \rho V = \frac{1-\omega}{\omega} V_k^{\omega/\omega-1} &+ V_k\big((1-\tau)(\theta\lambda K^{\theta-1} - \delta)k + \lambda K^{\theta}(1-\theta)\big) \\
&+ V_K\big(\lambda K^{\theta} - \delta K - V_k(K,K)^{1/\omega-1} - \tau(\theta\lambda K^{\theta} - \delta K)\big) \\
&+ \tfrac{1}{2}\big(\sigma(k,K)^2 V_{kk} + 2\sigma(k,K)\sigma(K,K)V_{kK} + \sigma(K,K)^2 V_{KK}\big).
\end{aligned}
\tag{51}
$$

We define the value function on a two-dimensional, equispaced grid using

$$
V(t,k,K) = V(t_n, k_i, K_j) = V(n\,\Delta t, i\,\Delta k, j\,\Delta K) = V_{i,j}^n. \tag{52}
$$

Now we can derive the upwind finite-difference approximation to (51)

$$
\begin{aligned}
\frac{V_{i,j}^{n+1} - V_{i,j}^n}{\Delta t} + \rho V_{i,j}^n = &- \frac{a_{i,j}^+}{\Delta k}\big(V_{i,j}^n - V_{i-1,j}^n\big) - \frac{a_{i,j}^-}{\Delta k}\big(V_{i+1,j}^n - V_{i,j}^n\big) \\
&- \frac{b_{i,j}^+}{\Delta K}\big(V_{i,j}^n - V_{i,j-1}^n\big) - \frac{b_{i,j}^-}{\Delta K}\big(V_{i,j+1}^n - V_{i,j}^n\big),
\end{aligned}
\tag{52}
$$

where

$$
\begin{aligned}
a_{i,j} &= -\frac{1-\omega}{\omega}(V_k)_{i,j}^{1/\omega-1} - (1-\tau)(\theta\lambda K_{i,j}^{\theta-1} - \delta)k_{i,j} - \lambda K_{i,j}^{\theta}(1-\theta) \\
b_{i,j} &= -\lambda K^{\theta} + \delta K + V_k(K,K)^{1/\omega-1} + \tau(\theta\lambda K^{\theta} - \delta K).
\end{aligned}
$$

Again, we choose $\omega = \theta = 0.5$, $\rho = 0.05$, $\lambda = 1.0$, $\delta = 0.025$. For $\sigma = \tau = 0$, we obtain the exact solution given by (6) along the line $k = K$; the numerical solution agrees with this result.

The extension of the method to implicit time integration is not completely straightforward because the resulting system of equations is expensive to solve. Instead, because the system of equations is diagonally dominant, it is customary to move the off-diagonal terms to the right-hand side and solve the system iteratively. One effective approach is to perform several relaxation steps where the off-diagonal terms are evaluated using the data from the previous relaxation step. For the first relaxation step, the off-diagonal terms are ignored. Usually about 4 of these relatively inexpensive relaxation steps is sufficient to solve the system of equations. This is far more efficient that directly inverting the system.

Boundary Conditions

In the preceding discussion, we did not discuss the treatment of the boundary conditions at the edges of the computational domain. However, this is an important concern. In this section, we outline some of the approaches for determining the boundary conditions.

In the calculation of the single-state value function discussed above, boundary conditions had to be applied at $k = 0$ and $k = 2k_s$ for the grid chosen. We know that that

$c(t, 0) = 0$, and thus for the solution of the consumption function the first of these boundary conditions may be set easily. For the value function, $V_k(t, 0) \to \infty$; this is a very difficult boundary condition to set numerically. However, in this problem the boundary condition takes care of itself because for k near zero, $a^+ = 0$ in (31). Then, the differencing method uses data biased to the right of the origin, and the value of the solution at $k = 0$ is never used. Thus, in practice the boundary condition at $k = 0$ is not needed.

At the $k = k_{max}$ boundary we know that $V_k(t, k \to \infty) = 0$, and thus it makes sense to set $V_k = 0$ at the edge of the domain. However, at $k = 2k_s$, the slope of the function is not close to zero and the boundary condition is not appropriate. In this case, we can do one of the following things:

- Use the boundary condition $V_k(t, k_{max}) = 0$ and extend k_{max} far enough away from the region of interest that errors at the boundary are not important. This can be done efficiently by increasing the grid spacing for large values of k. (The differencing methods have to be modified slightly for the non-equispaced grid.)
- Assume a value for V_k at the boundary. This must be obtained from some other information.
- Extrapolate the value of $V(t, k_{max})$ from the interior by assuming that the slope of the solution at the boundary is given by the slope at the last point in the interior. This approach may seem sensible, but it is prone to errors because extrapolations that do not contain any information about the solution are likely to cause large errors at the boundary.
- In some cases, $a^- = 0$ at k_{max} and the solution does not depend on the boundary condition. In this situation, any reasonable boundary condition (zeroth-order extrapolation, for example) will work effectively.

The first of these approaches is generally preferred (unless the latter condition holds). In the calculations presented above, this method is used and the solution for $k > 1.5k_s$ is affected by the boundary condition. The validity of this approach was tested by extending the domain and checking that the solution does not change.

Summary

Finite-difference methods have been used to solve the partial differential equations that result from a continuous time representation of a dynamic economy. We have shown that with the appropriate choice of differencing methods we can obtain accurate solutions of these differential equations. The choice of the differencing method was motivated by the analysis of a simple linear model equation. This showed that the differencing method must be chosen depending on the trajectory of the solution. This idea was used to develop upwind or type-dependent differencing methods that sense the appropriate direction of differencing.

Several implicit time integration methods were discussed. In principle, these methods are stable for any time step, and therefore it should be possible to reduce the cost of the calculation. We found that for the value function formulation we were able to increase the time step to about 10 times that of an explicit method; it is likely that this could be improved with some work. In the case of the consumption function formulation, we found that by analyzing a model problem we were able to stabilize the time integration for any time step, resulting in converged solutions in 10 time steps.

There are many books that may be consulted for more details concerning the development of these methods. Classical references that are useful concerning the analysis of model equations are those of Courant and Hilbert (1962) and Richtmyer and Morton (1967). A more recent reference that discusses the types of methods described in this paper is by Yee (1989); a less detailed, but adequate description is also given in Hirsch (1988). These references and several other useful books are listed below.

References

The following book discusses the derivation and analysis of continuous-time dynamic programming problems.

Kamien, M.I. and Schwartz, N.L. (1981) *Dynamic Optimization: The Calculus of Variations and Optimal Control in Economics and Management.* New York, North Holland.

The following are general-purpose books on finite-difference methods for partial differential equations.

Courant, R., and Hilbert, D. (1962). *Methods of Mathematical Physics,* Vols. I and II, New York, Interscience.

Richtmyer, R.D., and Morton, K. W. (1967). *Difference Methods for Initial Value Problems,* 2nd Ed., Chichester, John Wiley/Interscience.

Gear, G.W. (1971). *Numerical Initial Value Problems in Ordinary Differential Equations,* Englewood Cliffs, NJ, Prentice-Hall.

Dahlquist, G., and Bjork, A. (1974). *Numerical Methods,* Englewood Cliffs, NJ, Prentice-Hall.

Ames, W.F. (1977). *Numerical Methods for Partial Differential Equations,* 2nd Ed., New York, Academic Press.

Mitchell, A.R., and Griffiths, D.F. (1980). *The Finite Difference Method in Partial Differential Equations,* New York, Wiley.

LeVeque, R.L., (1992) *Numerical Methods for Conservation Laws,* 2nd Ed., Basel, Birkhauser.

These books are oriented towards computational fluid dynamics, but may be useful.

Peyret, R., and Taylor, T.D. (1983). *Computational Methods for Fluid Flow,* New York, Springer Verlag.

Anderson, D.A., Tannehill, J.C., and Pletcher, R.H., (1984). *Computational Fluid Dynamics and Heat Transfer,* New York, Hemisphere.

Hirsch, C. (1988). *Numerical Computation of Internal and External Flows,* Vols. I and II, New York, Wiley.

Yee, H.C., (1989). "A Class of High-Resolution Explicit and Implicit Shock-Capturing Methods," *NASA TM-101088.*