

# A Collocation Approach to Solving Riccati Equations Arising in Finance

Paul L. Fackler\*

first draft: December 1, 1999

current revision: June 19, 2000

## Abstract

Most existing multivariate models in finance are based on affine diffusion models. These typically lead to the need to solve a system of Riccati differential equations. A method for solving these equations by collocation using Newton's method is described and compared to the more commonly used Runge-Kutta approach.

**Keywords:** Asset Pricing, Differential Equations

---

\*The author is an Associate Professor at North Carolina State University.

Mail: Department of Agricultural and Resource Economics

NCSU, Box 8109

Raleigh NC, 27695, USA

e-mail: [paul\\_fackler@ncsu.edu](mailto:paul_fackler@ncsu.edu)

Web-site: <http://www4.ncsu.edu/~pfackler/>

© 2000, Paul L. Fackler

# 1 Introduction

Many asset pricing models are based on affine diffusions, including some stochastic volatility models and most multifactor models for pricing bond and futures, as well as options written on bonds and futures. The advantage of the affine class stems from the fact that asset values can be expressed in terms of an affine function of underlying state variables. The time-varying coefficients of the affine function satisfy a system of Riccati differential equations subject to specified initial conditions. The main computational challenge in pricing such assets is therefore the numerical solution of these equations.

A widely used solution approach uses Runge-Kutta methods, for which standard off-the-shelf software is readily available. Runge-Kutta methods are evolutionary, solving the differential equation by taking time steps away from an initial time at which the solution is known. This paper presents an alternative that can be more efficient in some situations, especially when the Riccati equations must be solved repeatedly. The method utilizes collocation, which approximates the solution to the differential equation simultaneously at a prespecified set of time values, rather than in an evolutionary fashion.

The remainder of the paper is organized in the following fashion. In section 2 the affine asset pricing model is described, along with the specific form of the Riccati equations that arise from this model. Section 3 describes the collocation solution along

with an explicit formula that implements Newton's method to solve the collocation equations. Section 4 discusses how derivatives with respect to model parameters can be computed at little additional cost. The paper ends with some concluding comments, including an assessment of the merits of the proposed approach relative to Runge-Kutta methods. An appendix describes some details required for extending the method to complex valued Riccati equations that arise in computing option prices. Computer code (in MATLAB) implementing the methods discussed in this paper are available from the author's web site.

## 2 The Affine Asset Pricing Model

Affine diffusions are characterized by instantaneous means and covariances that are affine in the process variables and can generically be expressed in the form (Duffie and Kan)

$$dx = [a(t) + A(t)x]dt + C(t)\text{diag}\left(\sqrt{b(t) + B(t)x}\right) dW, \quad (1)$$

where  $W$  is a  $n$ -vector of independent standard Weiner processes.<sup>1</sup>

---

<sup>1</sup>The following notational conventions are used: capital letters denote matrices, lower case letters denote vectors and the subscript 0 is used on a scalar coefficient in association with a coefficient vector, e.g., in an expression such as  $g_0 + gx$ . In general coefficients may be functions of time or, where appropriate, time-to-maturity, although this dependence will often be suppressed for clarity of exposition.

Affine asset pricing models are based on a underlying (often unobservable) factor model that can be represented, under the risk-neutral (equivalent martingale) measure, as an affine diffusion. Furthermore, the instantaneous risk free interest rate, any proportional dividends generated by the asset and the log of the terminal value of the asset are affine functions of the factors. These functions will be denoted as  $r_0(t) + r(t)x(t)$ ,  $w_0(t) + w(t)x(t)$ , and  $h_0(T) + h(T)x(T)$ , respectively, where  $T$  is the asset's terminal or maturity date ( $r$ ,  $w$  and  $h$  are all  $1 \times n$  vector-valued functions).

Using standard no-arbitrage arguments, the value of an asset,  $V(x, t; T)$ , satisfies

$$0 = V_t + V_x(a + Ax) + \frac{1}{2} \text{trace} \left( C \text{diag}(b + Bx) C^\top V_{xx} \right) - (g_0 + gx)V, \quad (2)$$

where  $g_0(t) = r_0(t) - w_0(t)$  and  $g(t) = r(t) - w(t)$ , subject to the boundary condition

$$V(x, T; T) = \exp \left( h_0(T) + h(T)x(T) \right).$$

The solution to this problem is readily verified to be of the exponential affine form

$$V(x, t; T) = \exp \left( \beta_0(T - t) + \beta(T - t)x(t) \right)$$

where<sup>2</sup>

$$\beta'(\tau) = \beta(\tau)A + \frac{1}{2} \beta(\tau)C \text{diag} \left( \beta(\tau)C \right) B - g, \quad (3)$$

and

$$\beta_0'(\tau) = \beta(\tau)a + \frac{1}{2} \beta(\tau)C \text{diag} \left( \beta(\tau)C \right) b - g_0. \quad (4)$$

---

<sup>2</sup>The demonstration uses the fact that  $\text{diag}(x)y = \text{diag}(y)x$ , when  $x$  and  $y$  are vectors of equal length.

These Riccati differential equations are solved subject to the initial conditions  $\beta_0(0) = h_0(T)$  and  $\beta(0) = h(T)$ . Note that  $\beta_0$  and  $\beta$  are defined in terms of time to maturity,  $\tau = T - t$ , whereas the parameters  $a, A, b, B, C, g_0, g$  are evaluated at  $t = T - \tau$ .

Specific uses of this framework include the valuation of discount bonds, for which  $g_0(t) = r_0, g(t) = r, h_0(0) = 0$  and  $h(t) = 0$ . The affine model includes the well known one-factor models of Vasicek and of Cox et al., the two factor model of Longstaff and Schwartz, and the three factor models of Chen and Scott and Dai and Singleton. Futures prices can also be expressed in this framework by setting  $g_0(t) = 0$  and  $g(t) = 0$  and defining the spot price of the underlying asset by  $S(t) = \exp(h_0(t) + h(t)x(t))$ . The commodity futures models of Schwartz are in this class. In addition, European option prices can be evaluated using this framework by utilizing the Fourier inversion techniques first discussed in this context by Heston.

### 3 Computational Considerations

The main computational problem with affine asset pricing models is solving the system of Riccati equations given in (3) and (4). This is an initial value problem that can be easily solved using such standard methods as Runge-Kutta (Press et al., chap.13). Runge-Kutta are evolutionary methods that compute solutions recursively, starting from an initial point where the solution is known. In their simplest form they use a fixed time step; more sophisticated versions adaptively choose the size of the time

step to guarantee a prescribed level of accuracy.

Runge-Kutta methods can be viewed as using iteratively computed local low order polynomial approximations. An alternative is to use a global approximation that is evaluated for the entire solution function simultaneously. The solution is approximated using  $\beta(\tau) \approx \phi(\tau)\Psi$ , where  $\phi : [0, \bar{\tau}] \rightarrow \mathbf{R}^N$  is a set of  $N$  basis functions and  $\Psi$  is an  $N \times n$ -matrix of coefficients to be determined. The basis functions  $\phi(\tau)$  are chosen to provide a good approximation; common choices are polynomials and splines (piecewise polynomials with continuity restrictions).

The  $n$ -dimensional system of differential equations (3) for  $\beta$  can then be approximated by choosing  $\Psi$  so the residual function,

$$r(\Psi) = \phi(\tau)\Psi A + \frac{1}{2}\phi(\tau)\Psi C \text{diag}(\phi(\tau)\Psi C)B - g - \phi'(\tau)\Psi, \quad (5)$$

is small and the boundary condition  $\phi(0)\Psi = h(T)$  is satisfied. The coefficient matrix  $\Psi$  can be determined using collocation (Judd, chap. 11) by satisfying (5) with equality at a set of  $N - 1$  specified time values. Together with the initial condition, this defines a set of  $nN$  equations to be solved for the  $nN$  elements in  $\Psi$ .<sup>3</sup>

Once  $\Psi$  is found,  $\beta_0$  can be approximated by  $\phi(\tau)\psi$  using the residual function

---

<sup>3</sup>Collocation is only one of a number of so-called weighted residual methods, which also include minimizing the integral of squared residual function and the Galerkin method, which solves for the coefficients that make the residual function orthogonal to the basis functions (see Judd for discussion).

implied by (4)

$$r_0(\psi) = \phi(\tau)\Psi a + \frac{1}{2}\phi(\tau)\Psi C \text{diag}(\phi(\tau)\Psi C)b - g_0 - \phi'(\tau)\psi. \quad (6)$$

The collocation solution satisfies (6) at  $N - 1$  values of  $\tau$ , together with the initial condition that  $\phi(\tau)\psi = h_0$ .

Several comparisons of the collocation approach with Runge-Kutta methods are in order. First, the number of time values at which the differential equation must be evaluated can be far less with collocation when the basis functions are well chosen for the specified problem. For example, for fairly smooth functions, low order polynomials can often approximate the solution with a high degree of accuracy. Even difficult problems, such as ones with boundary layers, can be well approximated with spline functions if the breakpoints were placed more densely around the boundary layer. In the current context, boundary layers will occur when speeds of mean reversion are fast (large negative eigenvalues of  $A$ ).

Second, evolutionary methods like Runge-Kutta must solve each problem from scratch. Collocation methods, on the other hand, improve in efficiency with good starting values. This is an important advantage in situations such as estimation and numerical comparative static exercises, in which the underlying parameter values are repeatedly altered and the Riccati equations re-solved.

With Runge-Kutta methods, the accuracy of the solution depends on the size of the time steps relative to the curvature of the solution function. With collocation, ac-

curacy depends on the number and type of approximating functions used. Although, it is possible that the family of basis functions used is not capable of representing the solution to an acceptable degree of accuracy, the quality of the solution is easily checked by examining the residual functions (5) and (6) at non-collocation points. Unlike adaptive Runge-Kutta methods, which continuously monitor accuracy, accuracy checks need only be done infrequently. For example, in calibrating a model to market asset prices, there is no need for great accuracy in the early stages of fitting. Instead, one can fit parameter values using a fairly crude approximation and then refine the approximation as the parameter values achieve a closer fit to the data.

To elaborate on the particulars of the collocation solution, it is useful to vectorize  $\Psi$ ; define  $\nu = \text{vec}(\Psi)$  and note that  $\text{vec}(XYZ) = (Z^\top \otimes X)\text{vec}(Y)$ . (5) can then be written as

$$r(\nu) = \left[ A^\top \otimes \phi(\tau) \right] \nu + \frac{1}{2} \left[ B^\top \text{diag} \left( [C^\top \otimes \phi(\tau)] \nu \right) C^\top \otimes \phi(\tau) \right] \nu - g - \left[ I_n \otimes \phi'(\tau) \right] \nu$$

and the boundary condition as

$$\left[ I_n \otimes \phi(0) \right] \nu = h(T).$$

Define a set of  $q = N - 1$  values of  $\tau$  and let  $\phi_i = \phi(\tau_i)$ ,  $A_i = A(T - \tau_i)$ , etc. The

collocation problem can be written as the system of  $nN$  equations in  $nN$  unknowns:

$$\begin{bmatrix} g_1 \\ \dots \\ g_q \\ h \end{bmatrix} = \begin{bmatrix} [A_1^\top \otimes \phi_1] - [I_n \otimes \phi_1'] \\ \dots \\ [A_q^\top \otimes \phi_q] - [I_n \otimes \phi_q'] \\ I_n \otimes \phi(0) \end{bmatrix} + \frac{1}{2} \begin{bmatrix} B_1^\top \text{diag}([C_1^\top \otimes \phi_1]\nu) C_1^\top \otimes \phi_1 \\ \dots \\ B_q^\top \text{diag}([C_q^\top \otimes \phi_q]\nu) C_q^\top \otimes \phi_q \\ 0 \end{bmatrix} \nu.$$

This expression has the form

$$f = [W_0 + \frac{1}{2}W(\nu)]\nu \tag{7}$$

and hence defines a fixed point iteration

$$\nu \leftarrow [W_0 + \frac{1}{2}W(\nu)]^{-1} f.$$

Notice that if  $B = 0$  (the Gaussian case), then  $W(\nu) = 0$  and  $\nu$  can be computed with a single linear solve. When  $B \neq 0$ , iterations can be initialized by treating the  $W(\nu)$  as zero if no starting values are supplied.

An alternative uses Newton's method to solve the collocation problem. Noting

that<sup>4</sup>

$$\frac{dW(\nu)\nu}{d\nu} = 2W(\nu),$$

Newton's method uses the iteration

$$\begin{aligned} \nu &\leftarrow \nu - \left[W_0 + W(\nu)\right]^{-1} \left(\left[W_0 + \frac{1}{2}W(\nu)\right]\nu - f\right) \\ &= \left[W_0 + W(\nu)\right]^{-1} \left(f + \frac{1}{2}W(\nu)\nu\right). \end{aligned} \tag{8}$$

The fixed point iteration and Newton's method both require a linear solve of  $nN$  equations in each iteration. Experience with the two algorithms suggests that Newton's method typically converges in fewer iterations and justifies the slight increase in time needed for each iteration. Neither method is guaranteed to converge for arbitrary parameter values but no difficulties have been encountered for parameter values encountered for published financial models. Newton's method typically take 4-5 iterations to achieve convergence on the order of  $10^{-12}$ .

---

<sup>4</sup>This result can be demonstrated by examining the element of  $W(\nu)\nu$  corresponding to  $\tau_i$  and  $\beta_j$ :

$$\phi_i \Psi Q_{ij} \Psi^\top \phi_i^\top = [\phi_i \Psi Q_{ij} \otimes \phi_i] \nu,$$

where  $Q_{ij} = C_i \text{diag}([B_i]_{\cdot j}) C_i^\top$ . Differentiating with respect to  $\nu$  yields

$$2\phi_i \Psi Q_{ij} \otimes \phi_i$$

(see Dhrymes, chap. 4, for a discussion of matrix calculus).

Once the value of  $\Psi$  is determined, the solution to (4) can be approximated using

$\beta_0(\tau) \approx \phi(\tau)\psi$  by solving

$$\begin{bmatrix} \phi'_1 \\ \dots \\ \phi'_q \\ \phi(0) \end{bmatrix} \psi = \begin{bmatrix} \phi_1 \Psi \left( a_1 + \frac{1}{2} C_1 \text{diag}(\phi_1 \Psi C_1) b_1 \right) \\ \dots \\ \phi_q \Psi \left( a_q + \frac{1}{2} C_q \text{diag}(\phi_q \Psi C_q) b_q \right) \\ h_0 \end{bmatrix} - \begin{bmatrix} g_0(\tau_1) \\ \dots \\ g_0(\tau_q) \\ 0 \end{bmatrix}, \quad (9)$$

a linear equation with  $N$  unknowns. The computed values of  $\Psi$  and  $\psi$  can then be used to calculate  $\beta$  and  $\beta_0$  for any time-to-maturity  $\tau$  by evaluating  $\phi(\tau)\Psi$  and  $\phi(\tau)\psi$ .

## 4 Computing Derivatives

In many applications it is useful to be able to compute the derivatives of the solution function with respect to the model parameters ( $a, A, b, B, C, g, g_0, h$  and  $h_0$ ). This section discusses how derivatives can be expressed as linear first order differential equations in  $\tau$  with non-constant coefficients and solved via collocation. The computations involve only two linear solves with the same matrices used in computing the solution functions themselves, so the computation of derivatives requires very little additional time.

The differential equation for  $\beta$  has the form

$$\frac{\partial \beta(\tau; \theta)}{\partial \tau} = f(\beta(\tau; \theta); \theta),$$

where  $\theta$  an  $m$ -vector of model parameters:

$$\theta = \begin{bmatrix} a \\ \text{vec}(A) \\ b \\ \text{vec}(B) \\ \text{vec}(C) \\ g^\top \\ g_0 \\ h^\top \\ h_0 \end{bmatrix}$$

with  $m = 3n^2 + 4n + 2$ . The derivative  $\partial\beta(\tau; \theta)/\partial\theta$  satisfies the differential equation

$$\frac{\partial \frac{\partial\beta}{\partial\tau}}{\partial\theta} = \frac{\partial \frac{\partial\beta}{\partial\theta}}{\partial\tau} = \frac{\partial f(\tau, \beta; \theta)}{\partial\theta} + \frac{\partial f(\tau, \beta; \theta)}{\partial\beta} \frac{\partial\beta}{\partial\theta}, \quad (10)$$

a first order linear differential equation (with non-constant coefficients) in  $\tau$ .

Defining  $\beta_\theta(\tau; \theta) = \partial\beta(\tau; \theta)/\partial\theta$ , the differential equation is

$$\beta'_\theta = \frac{\partial \left( \beta A + \frac{1}{2} \beta C \text{diag}(\beta C) B - g \right)}{\partial\theta} + \left[ A^\top + B^\top \text{diag}(\beta C) C^\top \right] \beta_\theta. \quad (11)$$

Explicit expressions for the affine term are given in Table 1. The initial condition for this differential equation is 0 except for the elements of  $\theta$  associated with  $h$ , in which case  $\beta_h(0; \theta) = I_n$ .

Using  $\phi(\tau)$  again as a basis for a family of approximating functions,

$$\beta_\theta(\tau; \theta) \approx [\mathbf{I}_n \otimes \phi(\tau)] \Psi_\theta,$$

where  $\Psi_\theta$  is  $nN \times m$ . From (11) the residual function for the derivative is

$$r_\theta(\Psi_\theta) = \frac{\partial \left( \phi(\tau) \Psi A + \frac{1}{2} \phi(\tau) \Psi C \text{diag}(\phi(\tau) \Psi C) B - g \right)}{\partial \theta} + \left[ A^\top + B^\top \text{diag}(\beta C) C^\top \otimes \phi(\tau) - \mathbf{I}_n \otimes \phi'(\tau) \right] \Psi_\theta$$

(here  $\beta(\tau)$  is replaced by its approximate  $\psi(\tau) \Psi$ ). To obtain the collocation approximation, form the stack of the terms in [ ] at the  $q = N - 1$  values of  $\tau_i$  and append to that  $n$  equations representing the initial conditions:

$$\begin{bmatrix} A_1^\top + B_1^\top \text{diag}(\phi(\tau_1) \Psi C_1) C_1^\top \otimes \phi(\tau_1) - \mathbf{I}_n \otimes \phi'(\tau_1) \\ \dots \\ A_q^\top + B_q^\top \text{diag}(\phi(\tau_q) \Psi C_q) C_q^\top \otimes \phi(\tau_q) - \mathbf{I}_n \otimes \phi'(\tau_q) \\ \mathbf{I}_n \otimes \phi(0) \end{bmatrix} \Psi_\theta = \begin{bmatrix} \frac{\partial \left( \phi(\tau_1) \Psi (A_1 + \frac{1}{2} C_1 \text{diag}(\phi(\tau_1) \Psi C_1) B_1 - g_1) \right)}{\partial \theta} \\ \dots \\ \frac{\partial \left( \phi(\tau_q) \Psi (A_q + \frac{1}{2} C_q \text{diag}(\phi(\tau_q) \Psi C_q) B_q - g_q) \right)}{\partial \theta} \\ \frac{\partial \beta(0; \theta)}{\partial \theta} \end{bmatrix} \quad (12)$$

The coefficients of the approximate to the derivative are obtained by solving this system of equations. Furthermore, the matrix on the left hand side is  $W_0 + W(\nu)$ , the matrix used in solving the original differential equation for  $\Psi$  by Newton's method (eq. 8). Thus there is little additional work required to compute  $\Psi_\theta$ .

A similar approach can be used to compute the derivatives of  $\beta_0(\tau; \theta)$ . Let  $\beta_{0\theta}(\tau; \theta) = \partial\beta_0(\tau; \theta)/\partial\theta$ ;  $\beta_{0\theta}$  satisfies

$$\beta'_{0\theta} = \frac{\partial\left(\beta a + \frac{1}{2}\beta C \text{diag}(\beta C)b - g_0\right)}{\partial\theta} + \left[a^\top + b^\top \text{diag}(\beta C)C^\top\right]\beta_\theta. \quad (13)$$

The specific values of the first term on the right hand side are given in Table 1. The initial conditions are all 0 except for the derivative with respect to  $h_0$ :  $\partial\beta_{0\theta}/\partial h_0 = 1$ .

Using the approximation

$$\beta_{0\theta}(\tau; \theta) \approx \phi(\tau)\psi_\theta,$$

values of  $\psi_\theta$  can be found using the collocation equations

$$\begin{bmatrix} \phi'(\tau_1) \\ \dots \\ \phi'(\tau_q) \\ \phi(0) \end{bmatrix} \psi_\theta = \Upsilon + \begin{bmatrix} a^\top + b^\top \text{diag}(\phi(\tau_1)\Psi C)C^\top \otimes \phi(\tau_1) \\ \dots \\ a^\top + b^\top \text{diag}(\phi(\tau_q)\Psi C)C^\top \otimes \phi(\tau_q) \\ 0_{1 \times n} \end{bmatrix} \Psi_\theta, \quad (14)$$

where

$$\Upsilon = \begin{bmatrix} \frac{\partial \left( \phi(\tau_1)\Psi a_1 + \frac{1}{2} \phi(\tau_1)\Psi C_1 \text{diag}(\phi(\tau_1)\Psi C_1) b_1 - g_{01} \right)}{\partial \theta} \\ \dots \\ \frac{\partial \left( \phi(\tau_q)\Psi a_q + \frac{1}{2} \phi(\tau_q)\Psi C_q \text{diag}(\phi(\tau_q)\Psi C_q) b_q - g_{0q} \right)}{\partial \theta} \\ \frac{\partial \beta_0(0; \theta)}{\partial \theta} \end{bmatrix}.$$

The matrix on the left hand side of (14) is the same matrix used to compute  $\psi$  in (9).

It is worthwhile to compare this approach to the alternatives. First, one could use generic numerical derivative software that re-solves the problem using slightly perturbed values of the parameters; this requires resolving  $m$  additional times (for one-sided approximations) or  $2m$  times (for two-sided approximations). The derivatives computed using this method are considerably less accurate than the original function and the method is relatively slow due to the number of function evaluations required. A second alternative is to use Runge-Kutta methods to solve for  $\beta$ ,  $\beta_0$ ,  $\beta_\theta$ , and  $\beta_{0\theta}$  simultaneously by appending (11) and (13) to (3) and (4). Although straightforward, the additional work is great relative to solving for  $\beta$  and  $\beta_0$  alone and can slow computation down considerably, making collocation relatively more attractive for the computation of derivatives.

**Table 1: Affine Terms in Equations (11) and (13)**

Parameter	$\frac{\partial(\beta A + \frac{1}{2}\beta C \text{diag}(\beta C)B - g)}{\partial\theta}$	$\frac{\partial(\beta a + \frac{1}{2}\beta C \text{diag}(\beta C)b - g_0)}{\partial\theta}$
$a$	$0_{n \times n}$	$\phi(\tau)\Psi$
$A$	$I_n \otimes \phi(\tau)\Psi$	$0_{1 \times n^2}$
$b$	$0_{n \times n}$	$\frac{1}{2}\phi(\tau)\Psi C \text{diag}(\phi(\tau)\Psi C)$
$B$	$I_n \otimes \frac{1}{2}\phi(\tau)\Psi C \text{diag}(\phi(\tau)\Psi C)$	$0_{1 \times n^2}$
$C$	$B^\top \text{diag}(\phi(\tau)\Psi C) \otimes \phi(\tau)\Psi$	$b^\top \text{diag}(\phi(\tau)\Psi C) \otimes \phi(\tau)\Psi$
$g$	$-I_n$	$0_{1 \times n}$
$g_0$	$0_{n \times 1}$	$-1$
$h$	$0_{n \times n}$	$0_{1 \times n}$
$h_0$	$0_{n \times 1}$	$0$

## 5 Concluding Comments

This note describes an approach to computing the solution to a set of Riccati differential equations used in modeling financial prices based on affine diffusions. The affine asset pricing model includes many commonly used models of bond, futures and options prices, especially those based on multiple factors.

Although these equations are easily solved using standard Runge-Kutta methods, there are a number of potential advantages to using the collocation approach. In general, which method is preferred depends on a number of factors, including how the solution is to be used, how many times it must be evaluated (at alternative parameter values) and what computing environment is used.

Collocation approximation provide a complete representation of the solution rather than solution values at discrete points. Thus the solution function can be evaluated at any time value without prespecifying the time values. This adds to the flexibility with which the solution can be used. The relative advantages of collocation increase when the Riccati equations must be solved many times for alternative parameter values. For example, pricing options using the affine model requires that the solution be computed with multiple starting values. In such settings, the overhead in setting up the basis functions becomes relatively insignificant. Furthermore, using previous solutions as starting values can reduce the number of iterations needed by Newton's method to determine coefficient values. The relative advantages of collocation

tion increase still more if derivatives of the solution function with respect to model parameters are required, as they are in applications that fit parameters to market data.

Moreover, in some computing environments generic ODE solvers are not particularly efficient. This is especially true in interpreted computing environments such as MATLAB, Gauss and Mathematica. Such computational platforms are increasingly popular for financial computation because they typically reduce development time and facilitate interactive model exploration. Solving differential equations in such an environment, however, requires callbacks by a solver routine to user defined functions that compute model parameters and evaluate time derivatives. Such callbacks can cause dramatic speed reductions. By contrast, the collocation algorithm can be designed to make a single call to a user specified function, thereby eliminating this source of inefficiency. The author has developed a suite of MATLAB functions that can speed computations by an order of magnitude relative to Runge-Kutta methods (for a given level of accuracy).

It is, thus, difficult to draw general conclusions about the relative merits of collocation versus Runge-Kutta because the efficiency of the methods depends critically on details of implementation and the uses of the solution. Clearly the linear solves required of the collocation method will make the method unattractive for high-dimensional problems ( $n$  large) and/or ones that require a high dimensional ap-

proximate ( $N$  large) to achieve desired accuracy. For more moderately sized problems ( $n \leq 3$  is typical in many financial applications), with smooth solution paths (allowing small  $N$ ), the linear solve operations may not be very burdensome. The collocation approach thus represents an alternative to standard evolutionary algorithms that can provide useful performance improvements.

## Appendix: Solving Complex Valued Problems

In some applications, notably in option pricing (Heston, Duffie et al.) and GMM estimation using the empirical characteristic function (Singleton), it is useful to be able to solve Riccati equations with complex valued initial conditions. No changes are needed if the computer language used supports complex matrix arithmetic. For other computing environments this appendix provides additional implementation details.

Denoting the real and imaginary parts of matrices with superscripts  $r$  and  $i$ , respectively, (7) can be expressed as

$$\begin{bmatrix} f^r \\ f^i \end{bmatrix} = \begin{bmatrix} W_0 + \frac{1}{2}W^r & -\frac{1}{2}W^i \\ 0 & W_0 + W^r \end{bmatrix} \begin{bmatrix} \nu^r \\ \nu^i \end{bmatrix} \quad (15)$$

where  $W^r = W(\nu^r)$  and  $W^i = W(\nu^i)$  and utilizing the fact that  $W^r \nu^i = W^i \nu^r$ . A simple fixed point approach iterates

$$\nu^i \leftarrow [W_0 + W^r]^{-1} f^i$$

and

$$\nu^r \leftarrow [2W_0 + W^r]^{-1} [f^r + W^i [W_0 + W^r]^{-1} f^i],$$

until convergence.

Newton's method uses the iteration

$$\begin{bmatrix} \nu^r \\ \nu^i \end{bmatrix} \leftarrow \begin{bmatrix} \nu^r \\ \nu^i \end{bmatrix} - \begin{bmatrix} W_0 + W^r & -W^i \\ W^i & W_0 + W^r \end{bmatrix}^{-1} \begin{bmatrix} e^r \\ e^i \end{bmatrix}$$

where the collocation residuals are

$$\begin{bmatrix} e^r \\ e^i \end{bmatrix} = \begin{bmatrix} W_0 + \frac{1}{2}W^r & -\frac{1}{2}W^i \\ 0 & W_0 + W^r \end{bmatrix} \begin{bmatrix} \nu^r \\ \nu^i \end{bmatrix} - \begin{bmatrix} f^r \\ f^i \end{bmatrix}$$

or, equivalently,

$$\begin{bmatrix} \nu^r \\ \nu^i \end{bmatrix} \leftarrow \begin{bmatrix} W_0 + W^r & -W^i \\ W^i & W_0 + W^r \end{bmatrix}^{-1} \begin{bmatrix} f^r + \frac{1}{2}(W^r \nu^r - W^i \nu^i) \\ f^i + W^i \nu^r \end{bmatrix}.$$

The partitioned matrix inverse in this expression is given by

$$\begin{bmatrix} (\tilde{W} + W^i \tilde{W}^{-1} W^i)^{-1} & (\tilde{W} + W^i \tilde{W}^{-1} W^i)^{-1} W^i \tilde{W}^{-1} \\ -(\tilde{W} + W^i \tilde{W}^{-1} W^i)^{-1} W^i \tilde{W}^{-1} & (\tilde{W} + W^i \tilde{W}^{-1} W^i)^{-1} \end{bmatrix},$$

where  $\tilde{W} = W_0 + W^r$ . Both algorithms must solve two  $nN$  linear equations on each iteration. As in the real case, Newton's method typically converges in fewer iterations and justifies the increase in time needed for each iteration.

Once the value of  $\Psi$  is determined,  $\beta_0(\tau)$  can be approximated using  $\phi(\tau)\psi$ , by

solving

$$\begin{bmatrix} \phi'_1 \\ \dots \\ \phi'_q \\ \phi(0) \end{bmatrix} \psi^r = \begin{bmatrix} \phi_1 \Psi^r \left( a_1 + \frac{1}{2} C_1 \text{diag}(\phi_1 \Psi^r C_1) b_1 \right) \\ \dots \\ \phi_q \Psi^r \left( a_q + \frac{1}{2} C_q \text{diag}(\phi_q \Psi^r C_q) b_q \right) \\ h_0^r \end{bmatrix}$$

$$- \frac{1}{2} \begin{bmatrix} \phi_1 \Psi^i C_1 \text{diag}(\phi_1 \Psi^i C_1) b_1 \\ \dots \\ \phi_q \Psi^i C_q \text{diag}(\phi_q \Psi^i C_q) b_q \\ 0 \end{bmatrix} - \begin{bmatrix} g_0(\tau_1) \\ \dots \\ g_0(\tau_q) \\ 0 \end{bmatrix},$$

and

$$\begin{bmatrix} \phi'_1 \\ \dots \\ \phi'_q \\ \phi(0) \end{bmatrix} \psi^i = \begin{bmatrix} \phi_1 \Psi^i \left( a_1 + C_1 \text{diag}(\phi_1 \Psi^r C_1) b_1 \right) \\ \dots \\ \phi_q \Psi^i \left( a_q + C_q \text{diag}(\phi_q \Psi^r C_q) b_q \right) \\ h_0^i \end{bmatrix},$$

two linear equations with  $N$  unknowns each.

## References

- Chen, Ren-RAW and Louis Scott. “Interest Rate Options in Multi-Factor Cox-Ross-Rubinstein Models of the Term Structure.” *Journal of Derivatives* 3(1995):53–72.
- Cox, John C., Jonathan E. Ingersoll, and Stephen A. Ross. “A Theory of the Term Structure of Interest Rates.” *Econometrica* 53(1985):385–407.
- Dai, Qiang and Kenneth J. Singleton. “Specification Analysis for Affine Term Structure Models.” *Journal of Finance* 55(2000):forthcoming.
- Dhrymes, Phoebus J. *Mathematics for Econometrics*. New York: Springer-Verlag, 1978.
- Duffie, Darrell and Rui Kan. “A Yield-Factor Model of Interest Rates.” *Mathematical Finance* 6(1996):379–406.
- Duffie, Darrell, Jun Pan, and Kenneth Singleton. “Transform Analysis and Asset Pricing for Affine Jump-Diffusions.” Graduate School of Business, Stanford University, 1999.
- Heston, Steven L. “A Closed-Form Solution for Options with Stochastic Volatility with Applications to Bond and Currency Options.” *Review of Financial Studies* 6(1993):327–343.
- Judd, Kenneth L. *Numerical Methods in Economics*. Cambridge, MA: MIT Press, 1998.
- Longstaff, Francis A. and Eduardo S. Schwartz. “Interest Rate Volatility and the Term Structure: A Two Factor General Equilibrium Model.” *Journal of Finance* 47(1992):1259–1282.
- Press, William H., Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes*, 2nd ed. Cambridge: Cambridge University Press, 1992.
- Schwartz, Eduardo S. “The Stochastic Behavior of Commodity Prices: Implications for Valuation and Hedging.” *Journal of Finance* 52(1997):923–973.
- Singleton, Kenneth J. “Estimation of Affine Asset Pricing Models Using the Empirical Characteristic Function.” Stanford University and NBER, April, 1999.
- Vasicek, Oldrich. “An Equilibrium Characterization of the Term Structure.” *Journal of Financial Economics* 5(1977):177–188.