On the Solution of Complementarity Problems Arising in American Options Pricing

Liming Feng* Vadim Linetsky† José Luis Morales‡ Jorge Nocedal§

August 3, 2010

Abstract

In the Black-Scholes-Merton model, as well as in more general stochastic models in finance, the price of an American option solves a parabolic variational inequality. When the variational inequality is discretized, one obtains a linear complementarity problem that must be solved at each time step. This paper presents an algorithm for the solution of these types of linear complementarity problems that is significantly faster than the methods currently used in practice. The new algorithm is a two-phase method that combines the active-set identification properties of the projected SOR iteration with the second-order acceleration of a (recursive) reduced-space phase. We show how to design the algorithm so that it exploits the structure of the linear complementarity problems arising in these financial applications and present numerical results that show the effectiveness of our approach.

Key Words: American options pricing, linear complementarity, projected SOR method

*Department of Industrial and Enterprise Systems Engineering, University of Illinois at Urbana-Champain. This author was supported by National Science Foundation grant CMMI-0927367.

†Department of Industrial Engineering and Management Sciences, Northwestern University. This author was supported by National Science Foundation grant DMS-0802720.

‡Departamento de Matemáticas, Instituto Tecnológico Autónomo de México. This author was supported by Asociación Mexicana de Cultura AC and CONACyT-NSF grant J110.388/2006.

§Department of Electrical Engineering and Computer Science, Northwestern University. This author was supported by National Science Foundation grant DMS-0810213 and Department of Energy grant DE-FG02-87ER25047-A004.

1 Introduction

This paper concerns the numerical solution of American options pricing problems. Most options traded on option exchanges world-wide and a large fraction of options traded over-the-counter are of the American-style. These include options on stocks of individual companies, stock indexes, foreign currencies, interest rates, commodities, and energy. Options books of a large financial institution may contain options on thousands of different underlying assets, and perhaps several dozen different contracts (with expiration dates ranging from days to years, and different strike prices). As the underlying asset prices change throughout the trading day, the options prices change as well. Re-pricing a large options book in real time may thus require re-computing thousands of option prices quickly. For such large scale applications, fast numerical algorithms are essential.

When the prices of underlying assets are assumed to follow a diffusion process, such as in the classical Black-Scholes-Merton model based on the geometric Brownian motion process, or in extensions such as Heston's stochastic volatility model, the pricing function of an American-style option solves a parabolic variational inequality. After this system is discretized in space and time, it yields a linear complementarity problem, which must be solved at each time step. Thus, the fast solution of linear complementarity problems (LCPs) is of great practical importance in computational finance. The most popular LCP method in practice is the projected SOR iteration, or the closely related variant, the projected Gauss-Seidel iteration [3]. The standard treatment of LCPs for American options pricing can be found, for example, in [14] for the simple case of the Black-Scholes-Merton model and in [8] for several more complicated settings.

Several new active-set methods [1, 13] have recently been proposed for solving these LCPs more efficiently (interestingly, interior point methods are not well suited for this application). Borici and Luethi [1] report promising results using a variant of the simplex-like method for LCPs with Z-matrices [3]. Ikonen and Toivanen [9] develop and test four other methods: a projected multigrid method, an operator splitting method, a penalty method, and a component-wise splitting method. In their experiments, the best results are obtained with the component-wise splitting method.

In this paper, we study a different algorithm that combines iterations of the projected SOR method with reduced-space steps. This two-phase approach exploits the fact that the projected SOR iteration often makes a quick estimation of the optimal active set, while the reduced-space iteration can dramatically improve upon this estimate and yield a fast rate of convergence. We illustrate the performance of this algorithm on both the Black-Scholes-Merton model (using various values of volatility and maturity) and the Heston model [7] with stochastic volatility. The algorithm studied in this paper is an adaptation of the method recently developed by Morales et al. [11] for rigid body simulations. By tailoring this approach to the structure of the linear complementarity problems studied in this paper, the algorithm achieves speedups up to thirty one times on our tests in the Black-Scholes-Merton model, and up to six times on our tests in the Heston model, compared to the projected SOR method. The savings are particularly significant in the Black-Scholes-Merton model with long maturity or high volatility.

2 Pricing American Options in the Black-Scholes-Merton model

Consider an American put option with strike price $K > 0$ and maturity time $T > 0$. If the option is exercised when the underlying asset price is S, the option holder receives the payoff $\Psi(S) = (K - S)^+ = \max(K - S, 0)$. Similarly, the payoff function for an American call option is $\Psi(S) = (S - K)^+$. Let $V(t, S)$ be the option value at time $t \in [0, T]$ when the asset price is S. We assume that V solves the following parabolic variational inequality (see [10], [16]):

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + (r - q)S\frac{\partial V}{\partial S} - rV \leq 0, \quad t \in [0, T), S \in (0, \infty),$$
$$V \geq \Psi, \quad t \in [0, T), S \in (0, \infty),$$
$$(\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + (r - q)S\frac{\partial V}{\partial S} - rV) \cdot (V - \Psi) = 0, \quad t \in [0, T), S \in (0, \infty),$$

subject to the terminal condition (payoff at maturity):

$$V(T, S) = \Psi(S), \quad S \in (0, \infty),$$

where σ is the volatility of the underlying asset, r is the risk free interest rate, and q is the dividend yield paid by the underlying asset.

For the convenience of numerical implementation, we let $\psi(x) = \Psi(Ke^x)$ and $\bar{u}(t, x) = V(T - t, Ke^x)$ (i.e., we make a state variable change $x = \ln(S/K)$, and transform the terminal value problem into an initial value problem). Then $\bar{u}(t, x)$ solves

$$\frac{\partial \bar{u}}{\partial t} - \mathcal{A}\bar{u} \geq 0, \quad t \in (0, T],\ x \in \Omega, \tag{1a}$$

$$\bar{u} \geq \psi, \quad t \in (0, T],\ x \in \Omega, \tag{1b}$$

$$(\frac{\partial \bar{u}}{\partial t} - \mathcal{A}\bar{u}) \cdot (\bar{u} - \psi) = 0, \quad t \in (0, T],\ x \in \Omega, \tag{1c}$$

with the initial condition

$$\bar{u}(0, x) = \psi(x), \quad x \in \Omega, \tag{1d}$$

where $\Omega = \mathbb{R}$ and the operator $\mathcal{A}$ is given by

$$\mathcal{A}f = \frac{1}{2}\sigma^2 \frac{\partial^2 f}{\partial x^2} + \mu\frac{\partial f}{\partial x} - rf, \quad \mu = r - q - \frac{1}{2}\sigma^2.$$

To numerically solve (1a-1d), we localize the problem to a bounded computational domain $\Omega = [\underline{x}, \bar{x}]$ and impose the following boundary condition on $\partial\Omega = \{\underline{x}, \bar{x}\}$:

$$\bar{u}(t, x) = \psi(x), \quad t \in (0, T],\ x \in \partial\Omega. \tag{1e}$$

We follow [10] and [16] and consider the variational formulation of (1). Let $\bar{\psi}$ be a C^2 function such that $\bar{\psi} = \psi$ on an open neighborhood of $\partial\Omega$, and define $u = \bar{u} - \bar{\psi}$. Then from equation (10) of [16], u solves the following variational inequality:

$$u(0, x) = \psi(x) - \bar{\psi}(x), \quad x \in \Omega, \tag{2a}$$

$$u(t,x) = 0, \quad t \in (0,T],\ x \in \partial\Omega, \tag{2b}$$

$$u(t,x) \geq \psi(x) - \bar{\psi}(x), \quad t \in (0,T],\ x \in \Omega, \tag{2c}$$

and for any test function $w \geq \psi - \bar{\psi}$,

$$(\frac{\partial u}{\partial t}, w-u) + a(u, w-u) - (\mathcal{A}\bar{\psi}, w-u) \geq 0, \tag{2d}$$

where $(\cdot,\cdot)$ is the inner product in $L^2(\Omega)$ and the bilinear form $a(\cdot,\cdot)$ is given by

$$a(u,w) = \frac{1}{2}\sigma^2 \int_{\underline{x}}^{\bar{x}} \frac{\partial u}{\partial x}\frac{\partial w}{\partial x} dx - \mu \int_{\underline{x}}^{\bar{x}} \frac{\partial u}{\partial x} w dx + r \int_{\underline{x}}^{\bar{x}} uw\, dx.$$

We apply the linear finite element method to solve the variational inequality (2). Divide $[\underline{x}, \bar{x}]$ into $m+1$ subintervals, each having length $h = (\bar{x} - \underline{x})/(m+1)$. Let $x_i = x + ih$, $0 \leq i \leq m+1$, be the nodes, and let $\phi(x) = (x+1)\mathbf{1}_{\{-1 \leq x \leq 0\}} + (1-x)\mathbf{1}_{\{0 < x \leq 1\}}$. Define the following piecewise linear finite element basis functions: $\phi_{h,i}(x) = \phi((x - x_i)/h)$. The function $\phi_{h,i}(x)$ takes value 1 at node x_i and zero at all other nodes. We seek a finite element approximation u_h to the solution of (2) in the space $\mathcal{V}_h$ spanned by the basis functions $\{\phi_{h,1}, \cdots, \phi_{h,m}\}$:

$$u_h(t,x) = \sum_{i=1}^{m} u_i(t)\phi_{h,i}(x), \quad t \in [0,T].$$

We consider test functions that are also in the space $\mathcal{V}_h$. It is easy to verify using integration by parts that $(\mathcal{A}\bar{\psi}, w-u) = -a(\bar{\psi}, w-u)$. For simplicity, we let $\bar{\psi} = \psi$. Although such chosen $\bar{\psi}$ is only piecewise differentiable for the American options we consider, numerical results show that this choice is sufficient. Equation (2c) then requires that the coefficient $u_i(t)$ of u_h be non-negative:

$$u_i(t) \geq 0, \quad 1 \leq i \leq m. \tag{3a}$$

Note that by construction, the vanishing boundary condition (2b) is automatically satisfied. The vanishing initial condition (2a) requires that

$$u_i(0) = 0, \quad 1 \leq i \leq m. \tag{3b}$$

Denote the coefficient vector of u_h by $\mathbf{u}(t) = (u_1(t), \cdots, u_m(t))^\top$. Consider an arbitrary test function $w \geq 0$ in the space $\mathcal{V}_h$ with coefficient vector $\mathbf{w} = (w_1, \cdots, w_m)^\top$. Then from (2d) we obtain

$$(\mathbf{w} - \mathbf{u}(t))^\top \cdot [\mathbb{M} \cdot \dot{\mathbf{u}}(t) + \mathbb{A} \cdot \mathbf{u}(t) + \mathbf{F}] \geq 0, \quad \forall \mathbf{w} \geq 0, \tag{3c}$$

where $\dot{\mathbf{u}}(t) = \left(\frac{du_1}{dt}, \cdots, \frac{du_m}{dt}\right)^\top$; $\mathbb{M} = (\mathbb{M}_{ij})$ with $\mathbb{M}_{ij} = (\phi_{h,j}, \phi_{h,i})$ is the mass matrix; $\mathbb{A} = (\mathbb{A}_{ij})$ with $\mathbb{A}_{ij} = a(\phi_{h,j}, \phi_{h,i})$ is the stiffness matrix; and $\mathbf{F} = (F_1, \cdots, F_m)^\top$ with

$F_i = a(\psi, \phi_{h,i})$ is the load vector. For the Black-Scholes-Merton model, the matrices $\mathbb{M}$ and $\mathbb{A}$ can be computed analytically:

$$\mathbb{A} = \begin{pmatrix} a_0 & a_1 & & \\ a_{-1} & a_0 & \ddots & \\ & \ddots & \ddots & a_1 \\ & & a_{-1} & a_0 \end{pmatrix}, \quad \mathbb{M} = \frac{h}{6} \begin{pmatrix} 4 & 1 & & \\ 1 & 4 & \ddots & \\ & \ddots & \ddots & 1 \\ & & 1 & 4 \end{pmatrix},$$

where

$$a_0 = a(\phi_{h,i}, \phi_{h,i}) = \frac{2}{3} rh + \frac{1}{h}\sigma^2,$$

$$a_{\pm 1} = a(\phi_{h,i}, \phi_{h,i\mp 1}) = \mp\frac{1}{2}\mu + \frac{1}{6} rh - \frac{1}{2h}\sigma^2, \quad \mu = r - q - \frac{1}{2}\sigma^2.$$

The matrix $\mathbb{A}$ is tri-diagonal but slightly non-symmetric. The load vector $\mathbf{F}$ can be easily approximated by replacing ψ by its linear finite element interpolant.

For the temporal discretization of the ordinary differential system (3), we divide $[0, T]$ into N equal subintervals, each with length $k = T/N$. Let $t_j = jk$, $0 \le j \le N$, be the temporal nodes. Denote $\mathbf{u}(t_j)$ by $\mathbf{u}^j$. When a standard $\theta-$scheme is used, we obtain the following:

$$(\mathbf{w} - \mathbf{u}^j)^\top \cdot [(\mathbb{M} + k\theta\mathbb{A})\mathbf{u}^j - (\mathbb{M} - k(1-\theta)\mathbb{A})\mathbf{u}^{j-1} + k\mathbf{F}] \ge 0, \quad \forall \mathbf{w} \ge 0,$$

$$\mathbf{u}^0 = 0, \quad \mathbf{u}^j \ge 0, \ 1 \le j \le N.$$

This is equivalent to the following linear complementarity problem (LCP) in the unknown vector $\mathbf{u}^j$:

$$(\mathbf{u}^j)^\top \cdot [(\mathbb{M} + k\theta\mathbb{A})\mathbf{u}^j - (\mathbb{M} - k(1-\theta)\mathbb{A})\mathbf{u}^{j-1} + k\mathbf{F}] = 0, \tag{4a}$$

$$(\mathbb{M} + k\theta\mathbb{A})\mathbf{u}^j - (\mathbb{M} - k(1-\theta)\mathbb{A})\mathbf{u}^{j-1} + k\mathbf{F} \ge 0, \tag{4b}$$

$$\mathbf{u}^0 = 0, \quad \mathbf{u}^j \ge 0, \ 1 \le j \le N. \tag{4c}$$

The commonly used implicit Euler scheme and Crank-Nicolson scheme correspond to $\theta = 1$ and $\theta = 1/2$, respectively. It is shown in [6] that an appropriate combination of the Crank-Nicolson scheme with the implicit Euler scheme greatly improves the performance of the numerical solution. This is known as the Rannacher scheme. More specifically, instead of using the Crank-Nicolson scheme directly, we replace the first step by four quarter implicit Euler steps (corresponding to $\theta = 1$ and time interval $0.25k$). We then proceed with the Crank-Nicolson scheme in the remaining time steps (corresponding to $\theta = 1/2$ and time interval k). Our numerical results show that the Rannacher scheme performs better than the standard Crank-Nicolson scheme.

Thus, to price American options in the Black-Scholes-Merton model, we need to solve a sequence of LCPs (4).

3 Pricing American Options in Heston's Model

In Heston's stochastic volatility model [7], the asset price process S_t and the variance process $v_t := \sigma_t^2$ solve the following two-dimensional stochastic differential equation:

$$\begin{aligned} dS_t &= (r-q)S_t dt + \sqrt{v_t} S_t dW_1(t), \\ dv_t &= \kappa(\eta - v_t)dt + \xi\sqrt{v_t} dW_2(t). \end{aligned}$$

That is, the volatility σ that was assumed to be constant in the Black-Scholes-Merton model is now stochastic and its square is assumed to follow the square-root diffusion process with a mean-reverting drift. The two Brownian motions W_1, W_2 (Wiener processes) driving the asset price process and the variance process are correlated, with correlation coefficient $\rho \in [-1, 1]$. Here $\xi > 0$ is the volatility parameter of the variance process, $r \geq 0$ is the risk-free interest rate, $q \geq 0$ is the dividend yield, $\kappa > 0$ is the rate of mean reversion, and $\eta > 0$ is the long run variance level (η is often denoted as θ in the literature). The infinitesimal generator of the two-dimensional Markov process (S_t, v_t) solving this stochastic differential equation is given by:

$$\mathcal{G}f = \frac{1}{2}vS^2\frac{\partial^2 f}{\partial S^2} + \rho\xi v S\frac{\partial^2 f}{\partial v \partial S} + \frac{1}{2}\xi^2 v\frac{\partial^2 f}{\partial v^2} + (r-q)S\frac{\partial f}{\partial S} + \kappa(\eta - v)\frac{\partial f}{\partial v}.$$

The formulation of the variational inequality and its discretization proceeds along the same lines as in the Black-Scholes-Merton model. The option price $V = V(t, S, v)$ is now a function of two state variables, the asset price S and its variance v, as well as time t. Doing the same change of variables as in the Black-Scholes-Merton model, localizing the problem to a bounded computational domain $\Omega = [\underline{x}, \overline{x}] \times [\underline{v}, \overline{v}]$, and imposing the following boundary condition

$$\bar{u}(t, x, v) = \psi(x), \quad t \in (0, T],\ (x, v) \in \partial\Omega,$$

we arrive at the variational formulation (2) with the following two-dimensional differential operator

$$\mathcal{A}f = \frac{1}{2}v\frac{\partial^2 f}{\partial x^2} + \rho\xi v\frac{\partial^2 f}{\partial v \partial x} + \frac{1}{2}\xi^2 v\frac{\partial^2 f}{\partial v^2} + (r - q - \frac{1}{2}v)\frac{\partial f}{\partial x} + \kappa(\eta - v)\frac{\partial f}{\partial v} - rf$$

and bilinear form

$$\begin{aligned} a(u, w) = \int_{\underline{x}}^{\overline{x}}\int_{\underline{v}}^{\overline{v}} \Bigg(& \frac{1}{2}v\frac{\partial u}{\partial x}\frac{\partial w}{\partial x} + \rho\xi v\frac{\partial u}{\partial v}\frac{\partial w}{\partial x} + \frac{1}{2}\xi^2 v\frac{\partial u}{\partial v}\frac{\partial w}{\partial v} \\ & -(r - q - \frac{1}{2}v)\frac{\partial u}{\partial x}w - (\kappa\eta - \kappa v - \frac{1}{2}\xi^2)\frac{\partial u}{\partial v}w + ruw\Bigg)\, dv dx. \end{aligned} \tag{5}$$

We discretize spatially using two-dimensional rectangular finite elements. We divide $[\underline{x}, \overline{x}]$ into $m+1$ equal intervals of length $h_x = (\overline{x} - \underline{x})/(m+1)$ and $[\underline{v}, \overline{v}]$ into $n+1$ equal intervals of length $h_v = (\overline{v} - \underline{v})/(n+1)$. The nodes are $(x_i, v_j) = (\underline{x} + ih_x, \underline{v} + jh_v)$, $i = 0, 1, \cdots, m+1, j = 0, 1, \cdots, n+1$. The rectangular two-dimensional finite element

basis functions are defined for any $i = 1, \cdots, m$ and $j = 1, \cdots, n$ as the product of the one-dimensional basis functions:

$$\phi_{ij}(x, v) = \phi_{h_x,i}(x)\phi_{h_v,j}(v) = \phi((x - x_i)/h_x)\phi((v - v_j)/h_v),$$

where $\phi_{h,i}(\cdot)$ and $\phi(\cdot)$ are defined as previously. The basis function ϕ_{ij} is equal to one at the node (x_i, v_j) and zero at all other nodes. There are $m \times n$ nodes in $[\underline{x}, \overline{x}] \times [\underline{v}, \overline{v}]$. We arrange the nodes as follows: (x_1, v_1), (x_1, v_2), $\cdots$, (x_1, v_n), (x_2, v_1), (x_2, v_2), $\cdots$, (x_m, v_n). Let $\mathcal{V}_h$ be the space spanned by the basis functions $\{\phi_{ij,1\le i\le m,1\le j\le n}\}$. We seek a finite element approximation u_h in $\mathcal{V}_h$ with non-negative time dependent coefficients:

$$u_h(t, x, v) = \sum_{i=1}^{m}\sum_{j=1}^{n} u_{ij}(t)\phi_{ij}(x, v), \quad u_{ij}(t) \ge 0, t \in (0, T].$$

We discretize temporally by the Rannacher scheme. Denote $\mathbf{u}(t) = (u_{11}(t), \cdots, u_{1n}(t), \cdots, u_{m1}(t), \cdots, u_{mn}(t))^\top$, and $\mathbf{u}^j = \mathbf{u}(t_j)$. The resulting discrete linear complementarity problem has the form (4), as before. The mass matrix is block-tridiagonal and is given by

$$\mathbb{M} = \begin{pmatrix} \mathbb{M}_{11} & \mathbb{M}_{12} & & 0 \\ \mathbb{M}_{21} & \mathbb{M}_{11} & \ddots & \\ & \ddots & \ddots & \mathbb{M}_{12} \\ 0 & & \mathbb{M}_{21} & \mathbb{M}_{11} \end{pmatrix}, \ \mathbb{M}_{11} = \frac{h_x h_v}{9}\begin{pmatrix} 4 & 1 & & 0 \\ 1 & 4 & \ddots & \\ & \ddots & \ddots & 1 \\ 0 & & 1 & 4 \end{pmatrix}, \ \mathbb{M}_{12} = \mathbb{M}_{21} = \frac{1}{4}\mathbb{M}_{11};$$

thus, there is a total of nine non-zero diagonals.

To compute the elements of the stiffness matrix $\mathbb{A}$, we need to compute $a(\phi_{kl}, \phi_{ij})$ for $1 \le i, k \le m$ and $1 \le j, l \le n$, with the bilinear form defined in (5). The integrands in $a(\phi_{kl}, \phi_{ij})$ are polynomials with the highest order terms x^2v^2 and xv^3. For such integrands, the 2×2 Gaussian quadrature rule (tensor product of two-point Gaussian quadrature rules for each coordinate) is exact and is used in our implementation. For fixed j and l, $a(\phi_{kl}, \phi_{ij})$ depends only on the difference $i - k$. Moreover, $a(\phi_{kl}, \phi_{ij}) = 0$ for $|i - k| > 1$ or $|j - l| > 1$. So $\mathbb{A}$ is also a block tri-diagonal matrix with tri-diagonal blocks:

$$\mathbb{A} = \begin{pmatrix} \mathbb{A}_{11} & \mathbb{A}_{12} & & 0 \\ \mathbb{A}_{21} & \mathbb{A}_{11} & \ddots & \\ & \ddots & \ddots & \mathbb{A}_{12} \\ 0 & & \mathbb{A}_{21} & \mathbb{A}_{11} \end{pmatrix}$$

with a total of nine non-zero diagonals. It suffices to compute the blocks $\mathbb{A}_{11}$, $\mathbb{A}_{12}$, $\mathbb{A}_{21}$ with a total of $3(3n - 2)$ non-zero values.

4 Description of the Algorithm

The linear complementarity problem (4) has the general form

$$z^T(Bz+b) = 0, \tag{6a}$$

$$Bz+b \geq 0, \tag{6b}$$

$$z \geq 0, \tag{6c}$$

where the $m \times m$ matrix B and the $m-$vector b are constant, and $z \in \mathbb{R}^m$ is the vector of unknowns. A variety of algorithms have been proposed for solving problems of this form, including matrix-splitting methods such as the projected SOR method, pivoting methods, and interior point methods; see, e.g. [3, 8, 15]. The most popular method in the context of American options pricing is the projected SOR method [14], which is given as follows.

Projected SOR Method
Initialize $z^0 \geq 0$; set $k \leftarrow 0$.
repeat until a stop test is satisfied:
  for $i = 1, \ldots, m$
    $\Delta z_i = \frac{1}{B_{ii}} \left(b_i + \sum_{j<i} B_{ij} z_j^{k+1} + \sum_{j \geq i} B_{ij} z_j^k \right)$;
    $\gamma_i = (1-\omega) z_i^k + \omega (z_i^k - \Delta z_i)$;
    $z_i^{k+1} = \max\{0, \gamma_i\}$;
  end
  $k \leftarrow k+1$
end repeat

This method is simple to implement and has a small computational cost per iteration on the problems considered in this paper, but may converge slowly and is difficult to parallelize due to its sequential nature.

Convergence can be accelerated by employing a two phase method that first applies the projected SOR iteration to obtain a guess of the active set, and then performs a *reduced-space phase* in which the components of z corresponding to the active set are kept at zero and the other components are chosen so as to satisfy (6a)-(6b). The cycle of projected SOR and subspace minimization iterations is repeated until an acceptable solution of the linear complementarity problem (6) is found.

Let us describe this approach in more detail. Suppose that after performing a few iterations of the projected SOR method, ℓ components of z are zero. (We assume without loss of generality that these are the first ℓ components of z.) We then improve this estimate by fixing the first ℓ components of z at zero and computing the remaining components so that (6a)-(6b) are satisfied. Given this zero structure of z and the fact that $z \geq 0$, conditions (6a)-(6b) imply that the last $m-\ell$ components of the vector $Bz+b$ must be zero, i.e.,

$$P(Bz+b) = 0 \quad \text{with} \quad P := \begin{bmatrix} 0 & I_{m-\ell} \end{bmatrix}. \tag{7}$$

The zero structure of z also implies that $z = P^T P z$, and thus (7) can be expressed as

$$\hat{B}\hat{z} + \hat{b} = 0, \tag{8}$$

where

$$\hat{B} = PBP^T, \quad \hat{b} = Pb, \qquad \hat{z} = Pz.$$

Next, we solve the square system (8) to obtain a vector $\hat{z}_+$. Since some of the components of $\hat{z}_+$ could be negative, which would conflict with (6c), we project $\hat{z}_+$ onto the nonnegative orthant by setting

$$\hat{z}_+ \leftarrow \max(0, \hat{z}_+). \tag{9}$$

This projection can cause some elements of the new vector $\hat{z}_+$ to become zero. If so, we apply the reduced-space phase again to a problem of the form (8), but of smaller dimension. The reduced-space phase is repeated in this manner until the solution of (8) contains only a few negative components (say, at most 20), meaning that the active-set prediction changes little. We denote by $z^0 \in \mathbb{R}^m$ the iterate computed at the end of the cycle of reduced-space iterations. (The nonzero components of z^0 are given by the final value of $\hat{z}_+$.) The proposed algorithm is summarized as follows.

Algorithm I: Projected SOR with Reduced-Space Phase

Choose an initial point z^0, a parameter $\Delta_{ac} > 0$, and set $z \leftarrow z^0 \geq 0$.
repeat
  Perform k_{sor} iterations of the projected SOR Method, starting from z^0 to
    obtain an iterate z;
  repeat (Reduced-Space Phase)
    Define $\hat{z}$ to be the subvector of z whose components are positive;
    Let $\hat{m}$ denote the dimension of the vector $\hat{z}$;
    Form and solve the $\hat{m} \times \hat{m}$ reduced system (8) to obtain $\hat{z}_+$;
    Set $\hat{z}_+ \leftarrow \max(0, \hat{z}_+)$;
    Set $nz \leftarrow$ number of zero components in $\hat{z}_+$;
    if $nz \geq \Delta_{ac}$
      Set $z \leftarrow \hat{z}_+$;
    else
      Define the new iterate $z^0 \in \mathbb{R}^m$ by placing $\hat{z}_+$ in appropriate positions
        and setting all other elements to zero; **break**;
    end if
  end repeat
end repeat

In our experiments, we set $k_{sor} = 3$ and $\Delta_{ac} = 20$. We terminate Algorithm I when two consecutive SOR iterates differ by less than a prescribed constant; see Section 5.

The success of the method depends crucially on the repeated application of the reduced-space phase. It greatly accelerates the estimation of the optimal active set and endows the method with a fast rate of convergence. The technique used in the solution of the linear system (8) has an important effect on the overall computing time. For the Black-Scholes-Merton model, it is appropriate to apply a direct factorization technique, since the coefficient

matrix is tridiagonal, whereas for the Heston model it is more effective to employ iterative linear algebra techniques, as discussed in the next section.

5 Numerical Experiments

In this section we report the results of numerical experiments comparing the projected SOR (P-SOR) method and the method proposed in this paper (Algorithm I) on the Black-Scholes-Merton and Heston models. All computations reported in this paper were performed on a 32-bit quad-core Intel 2.66GHz system with 4GB of RAM, running RHEL 4, using Fortran 77.

Tests with the Black-Scholes-Merton Model.

We consider pricing American put options in the Black-Scholes-Merton model with the following parameters:

$$r = 5\%, q = 0\%, K = 100.$$

We make the computational domain large enough and compare the two linear complementarity solvers. The P-SOR method and Algorithm I require an initial guess for the solution of a linear complementarity problem. For the first time step we set the initial guess to zero. For the jth time step in (4a-4c), we use the solution $\mathbf{u}^{j-1}$ obtained in the $(j-1)$th time step as the initial guess for $\mathbf{u}^j$. The P-SOR method and the new algorithm were terminated when two consecutive iterates in the projected SOR iteration satisfy $\|z^{k+1} - z^k\|_\infty \leq \epsilon$ for a sufficiently small error tolerance level ϵ. The tridiagonal linear systems (8) are solved by means of an LU factorization of matrix B. More specifically, we exploit the fact that B is a Toeplitz matrix, and that all matrices $\hat{B}$ in (8) inherit that property, and have smaller dimension. Therefore, the LU factorization is computed just once, at the beginning of Algorithm I. The relaxation parameter for the P-SOR method was chosen as in [9]:

$$\omega = \frac{2}{1 + \sqrt{1 - \rho^2}}, \quad \text{with} \quad \rho = \max_i \frac{1}{B_{ii}} \sum_{j \neq i} |B_{i,j}|.$$

Tables 1-4 present the results for the two methods applied to the Black-Scholes-Merton model with high volatility ($\sigma = 0.4$) or low volatility ($\sigma = 0.2$), and with long maturity ($T = 5$) or short maturity ($T = 0.5$). The first two columns give the number of spatial steps $m + 1$ and the number of time steps N. The third column gives the relaxation parameter determined as above for the Crank-Nicolson steps (the relaxation parameter for the first four Euler steps is determined similarly and is not reported). The fourth column reports the pricing error in the maximum norm evaluated at 41 points: $S = Ke^x$, $x = -0.22, -0.21, \cdots, 0.18$, which correspond to 41 asset prices in an approximation domain of interest $[80, 120]$. Benchmark prices are computed using the Hilbert transform method of [5] with an accuracy of about 10^{-6}. For each method, we report the total computing time in seconds (CPU) and the number of iterations of the projected SOR method (Iter). For the new algorithm we also report the number of reduced-space iterations (Red iter).

				Projected SOR		Algorithm I		
$m+1$	N	ω	Error	Iter	CPU(s)	CPU(s)	Iter	Red iter
90	5	1.53	2.1E-02	32	**0.001**	**0.000**	6	1
90	10	1.41	9.4E-03	27	**0.000**	**0.001**	4	1
90	20	1.28	4.0E-03	22	**0.001**	**0.000**	4	1
90	40	1.16	3.2E-03	16	**0.002**	**0.001**	4	1
180	5	1.73	2.0E-02	60	**0.002**	**0.000**	8	2
180	20	1.53	3.1E-03	42	**0.005**	**0.001**	5	1
180	80	1.28	6.2E-04	23	**0.009**	**0.002**	4	1
180	320	1.06	3.2E-04	11	**0.017**	**0.009**	4	1
360	10	1.80	6.7E-03	99	**0.011**	**0.001**	8	2
360	40	1.64	1.1E-03	60	**0.024**	**0.002**	4	1
360	160	1.41	2.4E-04	31	**0.047**	**0.009**	4	1
360	640	1.16	9.0E-05	15	**0.092**	**0.033**	4	1

Table 1: Black-Scholes-Merton: $\sigma = 0.2$, $T = 0.5$, $\underline{x} = -0.3$, $\bar{x} = 0.6$, $\epsilon = 10^{-8}$. At-the-money put price 4.655684.

				Projected SOR		Algorithm I		
$m+1$	N	ω	Error	Iter	CPU(s)	CPU(s)	Iter	Red iter
150	5	1.73	3.0E-02	78	**0.002**	**0.000**	9	2
150	20	1.53	5.0E-03	54	**0.005**	**0.001**	6	1
150	80	1.28	1.2E-03	31	**0.010**	**0.002**	4	1
150	320	1.06	5.4E-04	15	**0.020**	**0.008**	4	1
300	10	1.80	1.2E-02	129	**0.013**	**0.002**	10	3
300	40	1.64	2.1E-03	81	**0.028**	**0.003**	6	1
300	160	1.41	4.4E-04	42	**0.055**	**0.008**	4	1
300	640	1.16	1.6E-04	21	**0.108**	**0.029**	4	1
600	20	1.85	4.7E-03	204	**0.074**	**0.005**	10	3
600	80	1.73	8.8E-04	114	**0.150**	**0.011**	5	1
600	320	1.53	1.8E-04	57	**0.291**	**0.030**	4	1
600	1280	1.28	4.9E-05	28	**0.568**	**0.113**	4	1

Table 2: Black-Scholes-Merton: $\sigma = 0.4$, $T = 0.5$, $\underline{x} = -0.5$, $\bar{x} = 1.0$, $\epsilon = 10^{-10}$. At-the-money put price 10.141399.

				Projected SOR		Algorithm I		
$m+1$	N	ω	Error	Iter	CPU(s)	CPU(s)	Iter	Red iter
190	5	1.82	2.6E-02	111	**0.005**	**0.000**	7	2
190	10	1.75	2.0E-02	96	**0.006**	**0.000**	6	1
190	20	1.67	1.1E-02	79	**0.010**	**0.001**	5	1
190	40	1.57	3.6E-03	62	**0.014**	**0.002**	4	1
380	5	1.90	2.6E-02	215	**0.018**	**0.001**	11	3
380	20	1.82	8.6E-03	154	**0.037**	**0.002**	7	2
380	80	1.67	1.3E-03	90	**0.078**	**0.006**	4	1
380	320	1.45	3.5E-04	45	**0.155**	**0.020**	4	1
760	10	1.93	1.8E-02	360	**0.096**	**0.005**	13	4
760	40	1.87	2.0E-03	233	**0.205**	**0.009**	7	2
760	160	1.75	3.5E-04	124	**0.417**	**0.022**	4	1
760	640	1.57	1.1E-04	61	**0.811**	**0.077**	4	1

Table 3: Black-Scholes-Merton: $\sigma = 0.2$, $T = 5$, $\underline{x} = -0.3$, $\bar{x} = 1.6$, $\epsilon = 10^{-10}$. At-the-money put price 9.897570.

				Projected SOR		Algorithm I		
$m+1$	N	ω	Error	Iter	CPU(s)	CPU(s)	Iter	Red iter
400	5	1.90	4.4E-02	227	**0.020**	**0.001**	13	4
400	20	1.82	9.3E-03	161	**0.040**	**0.003**	7	2
400	80	1.67	2.1E-03	94	**0.085**	**0.007**	5	1
400	320	1.45	5.3E-04	47	**0.169**	**0.021**	4	1
800	10	1.93	2.0E-02	380	**0.105**	**0.006**	16	5
800	40	1.87	4.3E-03	244	**0.225**	**0.010**	7	2
800	160	1.75	9.4E-04	130	**0.457**	**0.025**	4	1
800	640	1.57	2.1E-04	64	**0.890**	**0.081**	4	1
1600	5	1.98	4.4E-02	858	**0.292**	**0.020**	43	14
1600	40	1.93	4.3E-03	471	**0.868**	**0.028**	11	3
1600	320	1.82	4.1E-04	176	**2.474**	**0.095**	4	1
1600	2560	1.57	4.3E-05	60	**6.644**	**0.622**	4	1

Table 4: Black-Scholes-Merton: $\sigma = 0.4$, $T = 5$, $\underline{x} = -0.8$, $\bar{x} = 3.2$, $\epsilon = 10^{-10}$. At-the-money put price 24.462532.

Since an LCP is solved at every time step, Tables 1-4 report the average performance over all LCPs solved.

We note that the new algorithm is significantly faster than the projected SOR method. Although the computing time of the new algorithm increases with the number of time steps N, the savings in CPU time are quite substantial in all cases. Borici and Luethi [1] report that their simplex method is between 2 and 9 times faster than the projected SOR method in their tests, whereas Algorithm I is up to 31 times faster in our tests. In particular, greater improvement over the projected SOR is observed when the option time to maturity or the volatility is large.

The stop test $\|z^{k+1} - z^k\|_\infty \leq \epsilon$ with ϵ ranging from 10^{-8} to 10^{-10} may seem overly stringent. However, it was necessary to reduce the stop tolerance to these levels so that the P-SOR method was able to achieve the reported accuracy.

A well known method for pricing American options in the Black-Scholes-Merton model is the Brennan-Schwartz algorithm [2]. This method solves the LCPs using the LU decomposition with projection. In the Black-Scholes-Merton model, the coefficient matrix of the LCPs is tridiagonal. This makes the LU decomposition efficient in this case. Unfortunately, in more general option pricing models such as the Heston's model, the coefficient matrix of the LCPs is not tridiagonal any longer and the LU decomposition is not applicable. This limits the applicability of the Brennan-Schwartz algorithm and requires the development of more general algorithms, such as the one proposed in this paper.

Tests with the Heston Model.

We now compare the performance of the two methods on the Heston model described in Section 3. The parameters for the model are as follows:

$$r = 5\%,\ q = 0\%,\ K = 100,\ T = 1,\ \rho = -0.5,\ \xi = 0.1,\ \kappa = 4,\ v_0 = \eta = 0.06.$$

Here v_0 is the initial variance level. Since the coefficient matrix in the Heston model is banded, its factorization gives rise to significant fill-in. It is therefore attractive to use an iterative method in the reduced-space phase, and we choose the generalized minimum residual method (GMRES) preconditioned by an incomplete LU factorization; see e.g. [12]. Specifically, we employed GMRES with a restart parameter of 5, and the modified LU decomposition (MILU) using no fill-in, i.e., MILU(0). The P-SOR method and the new algorithm were terminated when two consecutive projected SOR iterates satisfy $\|z^{k+1} - z^k\|_\infty \leq \epsilon = 10^{-6}$ (this stop tolerance was sufficient in these experiments to achieve the reported accuracy). GMRES stops when $||r_j||/||r_0|| \leq \epsilon/10$, where $||\cdot||$ is the Euclidean norm, r_j is the current residual, and r_0 is the initial residual. It is not simple to estimate the optimal value of the projected SOR relaxation parameter in this case. In Table 5 we display the results for $\omega = 1$. The pricing error is again in the maximum norm evaluated at the same 41 points as in the Black-Scholes-Merton model. Benchmark prices are obtained with an accuracy of around 3×10^{-6} by solving the system using large enough numbers of steps in the space and in time. The results show that our method has greatly reduced the number of iterations and achieves a speedup of up to six times in the above tested cases.

				Projected SOR		Algorithm I		
$m+1$	$n+1$	N	Max Error	Iter	CPU(s)	CPU(s)	Iter	Red iter
140	28	5	3.8E-02	200	**0.16**	**0.09**	11	3
140	28	10	1.5E-02	136	**0.18**	**0.11**	9	2
140	28	20	6.0E-03	89	**0.21**	**0.15**	8	2
140	28	40	2.6E-03	60	**0.26**	**0.21**	7	2
140	28	80	1.4E-03	40	**0.35**	**0.34**	6	1
280	56	10	1.2E-02	501	**2.63**	**0.84**	13	4
280	56	20	4.1E-03	311	**2.89**	**1.02**	11	3
280	56	40	2.0E-03	181	**3.16**	**1.27**	8	2
280	56	80	8.0E-04	101	**3.44**	**1.97**	7	2
280	56	160	4.0E-04	55	**3.72**	**3.47**	7	2
560	112	20	3.9E-03	1093	**50.97**	**8.62**	14	4
560	112	40	1.6E-03	619	**54.14**	**10.58**	11	3
560	112	80	7.4E-04	334	**56.62**	**13.16**	9	2
560	112	160	3.4E-04	176	**58.96**	**20.79**	7	2
560	112	320	1.6E-04	92	**61.62**	**34.57**	7	2

Table 5: Heston: $\underline{x} = -0.4$, $\bar{x} = 1.0$, $\underline{v} = 0.01$, $\bar{v} = 0.15$, $\epsilon = 10^{-6}$. At-the-money put price 7.798628 ($v_0 = 0.06$).

On the other hand, the speedup is not as significant as in the Black-Scholes-Merton model where a much faster direct solver is available for solving (8).

We should note that the benefits of the MILU(0) preconditioner are quite substantial. For example, for the case where $m+1 = 560$, $n+1 = 112$, $N = 320$, the (average) time required by the unpreconditioned GMRES method to solve the linear systems (8) was 259 seconds, compared to 28 seconds for the preconditioned GMRES method. The average number of GMRES iterations decreased from 120 to 5 by employing preconditioning. We experimented with various levels of fill-in and drop tolerances for the incomplete MILU factorization and observed that the choice MILU(0) (no fill-in) was the most efficient overall.

Finally, it is practically very relevant to compute the early exercise boundary of an American option. For example, in Heston's model, this corresponds to a surface, where for any given time to maturity t and variance level v (or volatility $\sqrt{v}$), there is an asset price level $S^*(t, v)$ so that the American put option should be exercised only if the underlying asset price is less than or equal to $S^*(t, v)$. Figure 1 shows the early exercise boundary for the American put option we consider. When the time to maturity t is zero, $S^*(0, v) = K$ for any variance level v. For $t > 0$, $S^*(t, v)$ is smaller for larger v, as expected. Note that we have plotted the early exercise boundary in terms of the time to maturity and volatility.

6 Final Remarks

We presented an algorithm for solving linear complementarity problems arising in American options pricing, and have demonstrated that it is highly efficient in practice. The crucial component in the new algorithm is a (recursive) subspace minimization phase that greatly accelerates the active-set prediction made by a projected SOR iteration. The subspace phase can be tailored to the structure of the linear complementarity problem; we have

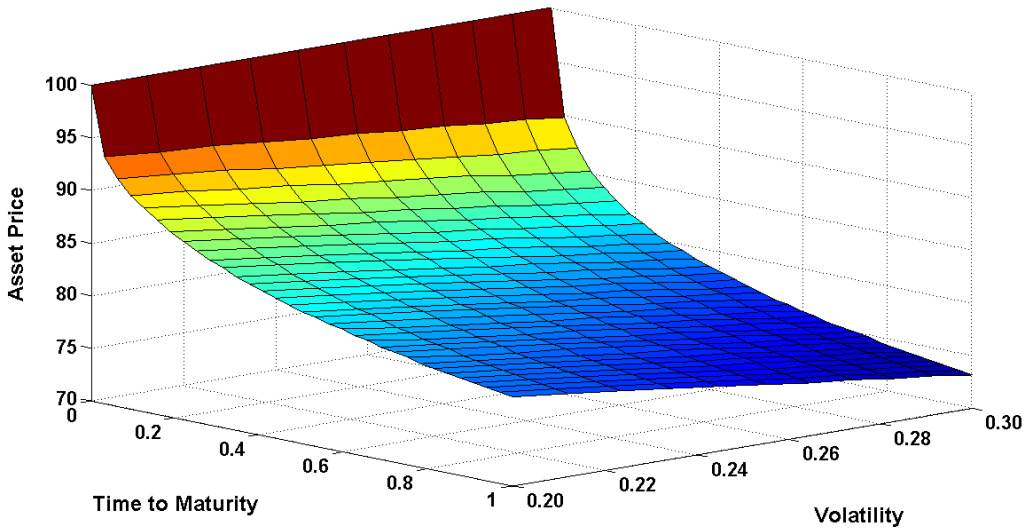

Figure 1: Early exercise boundary for the American put option in Heston's model.

shown how to do so for the classical Black-Scholes-Merton model as well as for Heston's stochastic volatility model.

Acknowledgment. The authors are grateful to Jong-Shi Pang for his comments and advice during the preparation of this manuscript.

References

[1] A. Borici and H-J. Luethi. Fast solutions of complemenatirty formulations in American put pricing. *Journal of Computational Finance*, 9(1), 2005.

[2] M. J. Brennan and E. S. Schwartz. The valuation of American put options. *Journal of Finance*, 32(2):449–462, 1977.

[3] R. W. Cottle, J.-S. Pang, and R. E. Stone. *The Linear Complementarity Problem.* Academic Press, London, 1992.

[4] L. Feng, P. Kovalov, V. Linetsky, and M. Marcozzi. Variational methods in derivatives pricing. In J.R. Birge and V. Linetsky, editor, *Handbooks in Operations Research and Management Science*, volume 15, pages 301–342, Amsterdam, 2008. Elsevier.

[5] L. Feng and X. Lin. Pricing Bermudan options in Lévy process models. Working paper University of Illinois at Urbana-Champaign, 2009.

[6] M. Giles and R. Carter. Convergence analysis of Crank-Nicolson and Rannacher time-marching. *Journal of Computational Finance*, 9(4):89–112, 2006.

[7] S. Heston. A closed-form solution for options with stochastic volatility with applications to bond and currency options. *Review of Financial Studies*, 6(2):327–343, 1993.

[8] J. Huang and J-S. Pang. Option pricing and linear complementarity. *Journal of Computational Finance*, 2:31–60, 1998.

[9] S. Ikonen and J. Toivanen. Efficient numerical methods for pricing American options under stochastic volatility. *Numerical Methods for Partial Differential Equations*, 24(1):104–126, 2007.

[10] P. Jaillet, D. Lamberton, and B. Lapeyre. Variational inequalities and the pricing of american options. *Acta Applicandae Mathematicae*, pages 263–289, 1990.

[11] J.L. Morales, J. Nocedal, and M. Smelyanskiy. An algorithm for the fast solution of symmetric linear complementarity problems. *Numerische Mathematik*, 111:251–266, 2008.

[12] Y. Saad. *Iterative Methods for Sparse Linear Systems.* PWS Publishing Company, Boston, USA, 1996.

[13] A.H. Siddiqi, P. Manchanda, and M. Kocvara. An iterative two-step algorithm for American option pricing. *IMA Journal on Managment Mathematics*, 11:71–84, 2000.

[14] P. Wilmott, S. Howison, and J. Dewynne. *The Mathematics of Financial Derivatives.* Cambridge University, Press, Cambridge, UK, 1995.

[15] S. J. Wright. *Primal-Dual Interior-Point Methods.* SIAM, Philadelphia, USA, 1997.

[16] X. L. Zhang. *Numerical analysis of American option pricing in a jump-diffusion model.* Mathematics of Operations Research, 22(3):668–690, 1997.