Iterative Methods for the Solution of a Singular Control Formulation of a GMWB Pricing Problem *

Y. Huang † P.A. Forsyth ‡ G. Labahn §

February 28, 2012

Abstract

Discretized singular control problems in finance result in highly nonlinear algebraic equations which must be solved at each timestep. We consider a singular stochastic control problem arising in pricing a Guaranteed Minimum Withdrawal Benefit (GMWB), where the underlying asset is assumed to follow a jump diffusion process. We use a scaled direct control formulation of the singular control problem and examine the conditions required to ensure that a fast fixed point policy iteration scheme converges. Our methods take advantage of the special structure of the GMWB problem in order to obtain a rapidly convergent iteration. The direct control method has a scaling parameter which must be set by the user. We give estimates for bounds on the scaling parameter so that convergence can be expected in the presence of round-off error. Example computations verify that these estimates are of the correct order. Finally, we compare the scaled direct control formulation to a formulation based on a block version of the penalty method [15]. We show that the scaled direct control method has some advantages over the penalty method.

Keywords: HJB equation, singular control, scaled direct control, iterative methods, jump diffusion

AMS Classification 65N06, 93C20

1 Introduction

Stochastic control problems arise in many financial applications, often in one of two forms: impulse control [7] and singular control [26, 22, 10]. The main computational cost of solution of an impulse control problem is the search for a global minimum along trajectories emanating from each node [7]. In contrast, discretization of singular control formulations requires the solution of a system of nonlinear algebraic equations at each timestep, which is a significant computational hurdle. In this

*This work was supported by Credit Suisse, New York and the Natural Sciences and Engineering Research Council of Canada

†Department of Electrical and Computer Engineering, University of Waterloo, Waterloo ON, Canada N2L 3G1, yqhuang@ecemail.uwaterloo.ca

‡Cheriton School of Computer Science, University of Waterloo, Waterloo ON, Canada N2L 3G1 paforsyt@uwaterloo.ca

§Cheriton School of Computer Science, University of Waterloo, Waterloo ON, Canada N2L 3G1 glabahn@uwaterloo.ca

article, we consider the singular control formulation of a particular financial contingent claim, the Guaranteed Minimum Withdrawal Benefit (GMWB) contract [21, 10, 15] and provide an efficient method for numerically pricing such a contract. Although we focus specifically on GMWB problems here, our results and methods of analysis are applicable to many other singular control problems in finance.

It is worthwhile to note that there are several approaches for solution of singular stochastic control problems. Markov chain methods [19, 13, 6] are essentially explicit finite difference schemes, and hence suffer from the usual timestep limitations due to stability considerations. Another approach is based on front tracking, whereby the boundaries between different regions (in our case the finite and infinite withdrawal domains) are tracked, usually by enforcing some some type of smooth pasting condition [18]. This method becomes complex if the different regions become multiply connected. We restrict attention in the following to methods which are unconditionally stable, and do not require explicit tracking of internal boundaries.

The GMWB pricing problem requires the solution of a Hamilton Jacobi Bellman (HJB) Variational Inequality (VI). It is necessary to discretize this HJB VI so that convergence to the viscosity solution is ensured [3, 2]. Generally, the viscosity solution of the HJB VI can be shown to be the solution of the contract valuation problem posed as a dynamic program, which is the financially relevant solution. In [10, 15], a penalty method is used to discretize the HJB VI arising from a GMWB contract, assuming Geometric Brownian Motion. The main focus of paper [15] is to show that the discretization scheme is monotone, consistent and l_∞ stable, and hence converges to the viscosity solution as the mesh parameter becomes small. The authors in [15] use a fully implicit timestepping method which guarantees an unconditionally monotone discretization. The method gives rise to a nonlinear system of algebraic equations at each timestep.

In this paper we use a scaled direct control method for solving the singular control formulation of the GMWB problem. The direct control technique was previously suggested for solving American option type problems [16, 5]. In addition, we consider the case where the underlying risky asset follows a jump diffusion process [9]. The method makes use of a fully implicit discretization which is shown to be monotone, consistent and l_∞ stable. We introduce a scaling factor in our direct control formulation, which allows us to solve the associated nonlinear algebraic equations using the fixed-point policy iteration method of [16]. We make use of some of the general results derived in [16]. However, verification of the properties required in the Theorems in [16] for the direct control formulation of the GMWB problem under jump diffusion requires a different approach compared to the examples in [16].

Our discretization of the singular control GMWB problem has a special structure that also allows for rapid solution of the associated nonlinear algebraic equations. We present a block matrix fixed-point policy iteration scheme for solving the nonlinear algebraic equations and show that it has the same convergence properties as fixed-point policy iteration. Numerical experiments imply that the number of iterations required for convergence of the block method is an order of magnitude less than required for the full matrix version. The block matrix fixed-point policy iteration scheme can also be applied to the penalty formulation [15].

The direct control technique (and also the penalty method) require the specification of a parameter which may affect solution accuracy and convergence of the iteration. We carry out an analysis of this parameter. We estimate bounds for the scaling factor (direct control) and the penalty factor (penalty method) so that convergence can be expected in the presence of roundoff error. Numerical experiments verify that these bounds are of the correct order of magnitude. Both the analysis and

the experimental results indicate that the useful numerical range of the scaling parameter (direct control) is much larger than for the penalty factor (penalty method).

The block matrix fixed-point policy iteration for the GMWB problem also has the advantage of being relatively simple to implement. Indeed the primary tools needed are a tridiagonal matrix solver and a procedure for finding maxima of finite sets of values.

The remainder of the paper is organized as follows. The following section gives the details of a GMWB contract and its formulation as a singular stochastic control problem. Section 3 gives the discretization of the scaled direct control form of the HJB VI and describes the nonlinear algebraic equations that need to be solved at every timestep. In Section 4 we show that the fixed-point policy iteration algorithm of [16] can be used to solve the nonlinear equations in full matrix form. Section 5 describes a block matrix fixed-point policy iteration method and gives the details for verifying the necessary convergence properties. Section 6 considers the case where floating point arithmetic impacts the mathematical results. Numerical tests are presented in the following section. The paper ends with a conclusion and an Appendix, which gives the the discretization details, some floating point error analysis and some additional numerical experiments.

2 Singular Control Formulation of the GMWB Problem

2.1 Motivation

Many holders of a defined contribution pension plan are responsible for investing a portfolio of assets which generate cash flows during their retirement. A GMWB contract consists of an initial lump sum payment to an insurance company which is then invested in risky assets. The holder is allowed to withdraw a specified amount each year of the contract, regardless of the performance of the risky asset, with withdrawals above the contract rate being subject to a penalty. At the end of the contract, the holder receives any investment amount remaining. Thus the holder of such a contract receives a minimum guaranteed yearly income, but can also participate in market gains. In return for providing this guarantee, the insurance company charges a proportional fee, which is extracted from the risky investment portfolio. The pricing problem is to determine the proportional fee that needs to be charged.

2.2 Formulation

Dai et al [10] formulated the GMWB pricing problem using a singular stochastic control formulation under the assumption that the underlying risky asset follows a constant volatility geometric Brownian motion process. In this section we review an extension to the Dai formulation given in Huang et al [16] where the risky asset is assumed to follow a jump diffusion process. The dynamics of the underlying risky asset W is given by the following stochastic differential equation

$$dW = (r - \eta - \lambda\rho)W dt + \sigma W dZ + (\xi - 1)W dq + dA, \quad \text{if } W > 0 \tag{2.1}$$

$$dW = 0, \quad \text{if } W = 0, \tag{2.2}$$

where dZ is the increment of a Weiner process and A is the investor's virtual guarantee withdrawal account. In the above, r is the risk free rate, σ is the volatility and η the fee charged for the guarantee. The variable λ is the jump intensity representing the mean arrival rate of the Poisson

process:

$$dq = \begin{cases} 0 & \text{with probability } 1 - \lambda dt \\ 1 & \text{with probability } \lambda dt \end{cases} , \tag{2.3}$$

with ξ a random variable representing the jump size of W. A typical modelling choice for the probability density of the jump size $p(\xi)$ is to assume a log-normal distribution [20, 1, 17],

$$p(\xi) = \frac{1}{\sqrt{2\pi}\zeta\xi} \exp\Big(-\frac{(\log(\xi) - \nu)^2}{2\zeta^2}\Big) , \tag{2.4}$$

with parameters ζ and ν, $\rho = E[\xi - 1]$, where $E[\cdot]$ is the expectation, and $E[\xi] = \exp(\nu + \zeta^2/2)$ given the distribution function $p(\xi)$ in (2.4).

The withdrawal feature is modeled with parameters G, the contractual withdrawal rate and $\kappa < 1$, the proportional penalty charge. Define $\tau = T - t$ where t is the forward time, and T is the expiry time of the contract and set $V = V(W, A, \tau)$ to be the no arbitrage value of the guarantee. Generalizing the formulation in [21, 10, 15] to the case with a stochastic jump process (2.1), the value of the guarantee is given from the solution to the following singular control problem

$$\min\Big[\; V_\tau - \mathcal{L}V - \lambda \mathcal{J}V - G\max(\mathcal{F}V, 0) \; , \quad \kappa - \mathcal{F}V \; \Big] = 0 \; . \tag{2.5}$$

Here the operators $\mathcal{L}, \mathcal{F}, \mathcal{J}$ are defined as

$$\begin{aligned} \mathcal{L}V &= \frac{\sigma^2}{2} W^2 V_{WW} + (r - \eta - \lambda\rho) W V_W - (r + \lambda) V \\ &= \frac{\sigma^2}{2} W^2 \mathcal{D}_{WW} V + (r - \eta - \lambda\rho) W \mathcal{D}_W V - (r + \lambda) V \\ \mathcal{F}V &= 1 - V_W - V_A = 1 - \mathcal{D}_W V - \mathcal{D}_A V \\ \mathcal{J}V &= \int_0^\infty V(\xi W, A, \tau) p(\xi) \; d\xi \; , \end{aligned} \tag{2.6}$$

while $\mathcal{D}_A$, $\mathcal{D}_W$, and $\mathcal{D}_{WW}$ denote the usual partial derivative operators. For details concerning the derivation of equation (2.5), we refer readers to [21, 7, 8, 10, 15].

2.3 The Scaled Direct Control form of the Pricing Problem

The control form of equation (2.5) is given by

$$\min_{\substack{\varphi\in\{0,1\},\psi\in\{0,1\} \\ \varphi\psi = 0}} \Big[\; \psi(\kappa - \mathcal{F}V) + (1 - \psi)(V_\tau - \mathcal{L}V - \lambda\mathcal{J}V - \varphi G \mathcal{F}V) \Big] = 0 \; . \tag{2.7}$$

Observe that that the term $\kappa - \mathcal{F}V$ is dimensionless whereas $(V_\tau - \mathcal{L}V - \lambda\mathcal{J}V - G\max(\mathcal{F}V, 0))$ has dimensions of *currency/time*. Hence equation (2.7) compares quantities having different units. Of course, in exact arithmetic, this is not an issue of importance. However, an iterative procedure for solution of the discretized equations will involve a test comparing two (in general) non-zero quantities. Hence scaling becomes important. Consequently, we introduce a scaling factor $\Pi > 0$ into equation (2.7)

$$\min_{\substack{\varphi\in\{0,1\},\psi\in\{0,1\} \\ \varphi\psi = 0}} \Big[\; \Pi\psi(\kappa - \mathcal{F}V) + (1 - \psi)(V_\tau - \mathcal{L}V - \lambda\mathcal{J}V - \varphi G \mathcal{F}V) \Big] = 0 \; . \tag{2.8}$$

Remark 2.1 (Scaling Factor)**.** *By introducing a scaling factor with dimension of* currency/time*, we ensure the comparison in equation (2.8) is conducted on two items with the same units. Of course, this still leaves the size of the scaling factor as arbitrary. We will exploit this fact to ensure that convergence of our iterative method can be guaranteed with a suitable choice for* Π.

Problem (2.5), or equivalently, (2.8) is solved on the computational domain

$$(W, A, \tau) \in [0, W_{\max}) \times [0, \omega_0] \times [0, T] \ , \tag{2.9}$$

where ω_0 denotes the initial premium. At expiry time $\tau = 0$, the value of the contract is

$$V(W, A, \tau = 0) \quad = \quad \max\Big[W, (1-\kappa)A\Big]. \tag{2.10}$$

Other boundary conditions are

$$\begin{aligned} \min\Big[V_\tau - rV - G\max(1 - V_A, 0), \kappa - (1 - V_A)\Big] = 0 \quad &; \quad W = 0 \ , \\ V(W_{\max}, A, \tau) = e^{-\eta\tau} W_{\max} \quad &; \quad W = W_{\max} \ , \\ V_{WW} \to 0 \quad &; \quad W \to W_{\max} \ , \\ V_\tau = \mathcal{L}V - \lambda \mathcal{J} V \quad &; \quad A = 0 \ . \end{aligned} \tag{2.11}$$

The boundary condition at $W = 0$ is a variational inequality which normally can also be solved as a control equation. No boundary condition is required at $A = \omega_0$. The condition $V_{WW} \to 0$, $W \to W_{\max}$ assumes linear behaviour of the solution for $W \to W_{\max}$. This condition was used in [11, 28] so that the combined jump terms vanish, which eliminates the necessity of estimating values outside the computational domain. Imposing a boundary condition at $W = W_{\max}$ introduces a localization error. We will verify that choosing $W_{\max}$ sufficiently large results in negligible error at values of (W, A) of interest.

2.4 No-arbitrage Fee

Since no fee is paid up-front, the insurance company needs to charge a proportional fee η (see equation (2.1)), such that the value of the contract is equal to the initial premium. If ω_0 denotes this initial premium and $V(\eta; W, A, \tau)$ is the value of the contract as a function of η, then the no-arbitrage fee is the solution to the equation

$$V(\eta; \omega_0, \omega_0, T) \quad = \quad \omega_0 \ . \tag{2.12}$$

3 GMWB: Discretization

In order to solve equation (2.8), we discretize the equation over a finite grid in the $W \times A$ plane. Define a set of nodes in the W direction $\{W_1, W_2, ..., W_{i_{\max}}\}$ and in the A direction $\{A_1, A_2, ..., A_{j_{\max}}\}$. Denote the n^{th} timestep by $\tau^n = n\Delta\tau$ and let $V_{i,j}^n$ be the approximate solution of equation (2.8) at (W_i, A_j, τ^n). We discretize equation (2.8) using fully implicit timestepping and central, forward and backward differencing so that the positive coefficient condition is satisfied [12, 15, 29]. For efficiency, central differencing is used as much as possible [29].

Let $\mathcal{L}^h, \mathcal{F}^h, \mathcal{D}_W^h, \mathcal{D}_A^h$ (defined in Appendix A.1 and A.2) be the discrete forms of the operators $\mathcal{L}, \mathcal{F}, \mathcal{D}_W, \mathcal{D}_A$, respectively and let $\mathcal{J}^h$ be the discretized $\mathcal{J}$ operator. The integral term $\mathcal{J}V$ is discretized via transformation into a correlation integral combined with a use of the midpoint rule as described in detail in [11, 28].

Equation (2.8) is then written in the discrete form

$$\begin{aligned}(1-\psi_{i,j})\left(\frac{1}{\Delta\tau}V_{i,j}^{n+1}-\mathcal{L}^hV_{i,j}^{n+1}+\varphi_{i,j}G(\mathcal{D}_W^hV_{i,j}^{n+1}+\mathcal{D}_A^hV_{i,j}^{n+1})\right)+\Pi\ \psi_{i,j}(\mathcal{D}_W^hV_{i,j}^{n+1}+\mathcal{D}_A^hV_{i,j}^{n+1})\\ =(1-\psi_{i,j})\frac{1}{\Delta\tau}V_{i,j}^{n}+\Pi\ \psi_{i,j}(1-\kappa)+(1-\psi_{i,j})\left(\lambda[\mathcal{J}^hV^{n+1}]_{i,j}+\varphi_{i,j}G\right)\end{aligned} \tag{3.1}$$

where

$$\begin{aligned}\{\varphi_{i,j},\psi_{i,j}\}\ \in\ \underset{\substack{\varphi\in\{0,1\},\psi\in\{0,1\}\\ \varphi\psi=0}}{\arg\max}\Bigg\{-\Pi\ \psi(\kappa-\mathcal{F}^hV_{i,j}^{n+1})-(1-\psi)\Bigg(\frac{V_{i,j}^{n+1}-V_{i,j}^{n}}{\Delta\tau}\\ -\left(\mathcal{L}^hV_{i,j}^{n+1}+\lambda[\mathcal{J}^hV^{n+1}]_{i,j}+\varphi G\mathcal{F}^hV_{i,j}^{n+1}\right)\Bigg)\Bigg\}\ .\end{aligned} \tag{3.2}$$

We can also rewrite equation (3.1) in an equivalent form (replacing $\mathcal{D}_A^h$ by a backward difference)

$$\begin{aligned}(1-\psi_{i,j})\left(\frac{1}{\Delta\tau}V_{i,j}^{n+1}\right. &-\left.\mathcal{L}^hV_{i,j}^{n+1}+\varphi_{i,j}G(\frac{V_{i,j}^{n+1}}{\Delta A_j^-}+\mathcal{D}_W^hV_{i,j}^{n+1})\right)+\Pi\ \psi_{i,j}(\frac{V_{i,j}^{n+1}}{\Delta A_j^-}+\mathcal{D}_W^hV_{i,j}^{n+1})\\ &=(1-\psi_{i,j})\frac{1}{\Delta\tau}V_{i,j}^{n}+\Pi\ \psi_{i,j}(1-\kappa+\frac{V_{i,j-1}^{n+1}}{\Delta A_j^-})\\ &\quad+(1-\psi_{i,j})\left(\lambda[\mathcal{J}^hV^{n+1}]_{i,j}+\varphi_{i,j}G(\frac{V_{i,j-1}^{n+1}}{\Delta A_j^-}+1)\right)\end{aligned} \tag{3.3}$$

where $\Delta A_j^- = A_j - A_{j-1}$.

The boundary conditions at $W = W_{\max}$ translate into the discrete equations

$$V_{i_{\max},j} = e^{-\eta\tau^n}W_{\max}, \tag{3.4}$$

which we approximate by

$$(\frac{1}{\Delta\tau}+\eta)V_{i_{\max},j}^{n+1}\ =\ \frac{1}{\Delta\tau}V_{i_{\max},j}^{n}\ ;\ V_{i_{\max},j}^{0}=W_{\max}\ . \tag{3.5}$$

At $W = 0$, equation (3.1) holds with $\mathcal{D}_W^h \equiv 0$. At $A = 0$, equation (3.1) holds with $\varphi_{i,j} = 0$, $\psi_{i,j} = 0$.

Remark 3.1. *Using the methods as in [15] it is straightforward to show that scheme (3.1) is monotone, consistent and l_∞ stable.*

3.1 Matrix Form of the Discretized Equations

Equations (3.1-3.3) are discretized using fully implicit method timestepping. The resulting set of discrete algebraic equations then needs to be solved. We do this via a modified version of policy iteration.

Let $N = i_{\max} \times j_{\max}$ be the size of the $W \times A$ plane grid and set

$$v_{*,j} \;=\; (v_{1,j}, v_{2,j}, \ldots, v_{i_{\max},j})' \quad \text{and } v = ((v_{*,1})', (v_{*,2})', \ldots, (v_{*,j_{\max}})')' \tag{3.6}$$

where $v_{*,j}$ is of length $i_{\max}$ and v is of length N. Similarly, we can write the controls as vectors

$$q_{*,j} \;=\; (q_{1,j}, q_{2,j}, \ldots, q_{i_{\max},j})' \quad \text{and } q = ((q_{*,1})', (q_{*,2})', \ldots, (q_{*,j_{\max}})')' \tag{3.7}$$

where $q_{i,j} \in Q = \big\{(\varphi, \psi) \;\mid\; \varphi \in \{0,1\}, \psi \in \{0,1\}, \varphi\psi = 0\big\}$. Let

$$\ell = i + (j-1) i_{\max} \quad \text{with} \quad 1 \le i \le i_{\max} \quad \text{and} \quad 1 \le j \le j_{\max}, \tag{3.8}$$

so that

$$\begin{aligned} [v_{*,j}]_i \;&=\; v_{i,j} \\ &=\; v_\ell \quad \text{with} \quad \ell = i + (j-1) i_{\max} \; . \end{aligned} \tag{3.9}$$

As a result, we will sometimes refer to the same entry in the N-length vector v as v_ℓ or $v_{i,j}$, with the reference being clear from the context. It is also convenient to represent the algebraic equations by matrix notation. In this paper we use boldface capital letter $\mathbf{T}$ to represent an $N \times N$ matrix with entries $[\mathbf{T}]_{\ell,m} = T_{\ell,m}$. We will also refer to the j^{th} $i_{\max} \times i_{\max}$ subblock of $\mathbf{T}$ using the notation $\mathbf{T_j}$. These subblocks will be defined in later sections.

At each timestep, we must solve for the unknowns $V_{i,j}^{n+1}$. We can write these equations in nonlinear matrix form as follows. Let $[v_{*,j}]_i = V_{i,j}^{n+1}$. Then, as generalized in [16], the algebraic equations at each timestep can be written as:

$$\sup_{q \in Q} \Big\{ -\mathbf{T}(q) v + c(q) \Big\} = 0 \; , \tag{3.10}$$

where Q is a set of controls and for $y = (y_1, y_2, \ldots, y_N)'$ we have

$$\begin{aligned} [\mathbf{T}(q^k) y]_\ell \;&=\; [\mathbf{T}^k y]_\ell = (1 - \psi_\ell^k) \bigg(\frac{1}{\Delta\tau} y_\ell - \mathcal{L}^h y_\ell + \varphi_\ell^k G(\mathcal{D}_W^h y_\ell + \mathcal{D}_A^h y_\ell) \bigg) \\ &\qquad + \Pi \; \psi_\ell^k (\mathcal{D}_W^h y_\ell + \mathcal{D}_A^h y_\ell) - (1 - \psi_\ell^k) \lambda [\mathcal{J}^h y]_\ell \\ \Big[c(q^k, V^n) \Big]_\ell \;&=\; (1 - \psi_\ell^k) \frac{1}{\Delta\tau} V_\ell^n + \Pi \; \psi_\ell^k (1 - \kappa) \; . \end{aligned} \tag{3.11}$$

Due to the nature of the jump term $\mathcal{J}$ [11] the matrix $\mathbf{T}$ is dense, which makes the standard policy iteration algorithm inefficient. To avoid this problem, we split $\mathbf{T}$ into two components $\mathbf{A} - \mathbf{B}$ such that $\mathbf{A}$ is sparse, and $\mathbf{B}$ is dense. Here

$$\begin{aligned} [\mathbf{A}(q^k) y]_\ell \;&=\; [\mathbf{A}^k y]_\ell = (1 - \psi_\ell^k) \bigg(\frac{1}{\Delta\tau} y_\ell - \mathcal{L}^h y_\ell + \varphi_\ell^k G(\mathcal{D}_W^h y_\ell + \mathcal{D}_A^h y_\ell) \bigg) \\ &\qquad + \Pi \; \psi_\ell^k (\mathcal{D}_W^h y_\ell + \mathcal{D}_A^h y_\ell) \\ \Big[\mathbf{B}(q^k) y \Big]_\ell \;&=\; \Big[\mathbf{B}^k y \Big]_\ell = (1 - \psi_\ell^k) \lambda [\mathcal{J}^h y]_\ell \; . \end{aligned} \tag{3.12}$$

Remark 3.2. *(Structure of $\mathbf{T}(q)$, $c(q)$) It is important to observe that $[\mathbf{T}(q)]_{\ell,m}$, $[\mathbf{A}(q)]_{\ell,m}$, $[\mathbf{B}(q)]_{\ell,m}$, $[c(q)]_\ell$ depend only on q_ℓ. This is typical of discretized HJB equations.*

Assuming that Q is a finite set, or that Q is compact and $\mathbf{T}(q)$ is an upper semi-continuous function of q, then equation (3.10) is interpreted as

$$\mathbf{T}(q^*)v \;=\; \big[\mathbf{A}(q^*) - \mathbf{B}(q^*)\big]v \;=\; c(q^*)$$

$$\text{with} \quad q^*_\ell = \arg\max_{q_\ell \in Q} \Big\{ -\mathbf{T}(q)v + c(q) \Big\}_\ell \;=\; \arg\max_{q_\ell \in Q} \Big\{ -\big[\mathbf{A}(q) - \mathbf{B}(q)\big]v + c(q) \Big\}_\ell . \quad (3.13)$$

Thus the problem has a potentially different control for each row of the linear system. Note that equation (3.10) is highly nonlinear.

Remark 3.3 (Properties of $\mathbf{T}(q)$)**.** *It is commonplace in practice for the set of controls Q to be finite or compact. In some circumstances, $\mathbf{T}(q)$ can be a discontinuous function of the control q [29]. In this case, we can carry out the analysis by replacing $\mathbf{T}(q^*)$ by its upper semi-continuous envelope, as discussed in [16]. However, for ease of exposition, and to avoid notational clutter, we make the assumption that $\mathbf{T}(q)$ is upper semi-continuous.*

Splitting the sparse and dense components implies that, rather than using the traditional policy iteration to solve the nonlinear system in (3.10), a fixed point policy iteration will be used. This has been shown to be more efficient [16] than the standard policy iteration. In Algorithm 1, each iteration requires a sparse solve, and a dense matrix vector multiply, $\mathbf{B}v^k$. The latter can be carried out efficiently using an FFT as shown in [11].

Algorithm 1 Fixed Point-Policy Iteration

1: $v^k = (v)^k$ with $v^0 =$ Initial solution vector of size N
2: **for** $k = 0, 1, 2, \ldots$ until converge **do**
3: $\quad q^k_\ell = \arg\max_{q_\ell \in Q} \Big\{ -\big[\mathbf{A}(q) - \mathbf{B}(q)\big]v^k + c(q) \Big\}_\ell$
4: $\quad$ Solve $\mathbf{A}(q^k)v^{k+1} = \mathbf{B}(q^k)v^k + c(q^k)$
5: $\quad$ **if** $k > 0$ and $\max_\ell \dfrac{|v^{k+1}_\ell - v^k_\ell|}{\max\left[scale, |v^{k+1}_\ell|\right]} < tolerance$ **then**
6: $\quad\quad$ break from the iteration
7: $\quad$ **end if**
8: **end for**

The factor *scale* in Algorithm 1 ensures that unrealistic levels of accuracy are not required for small values of the solution. For example, if the option is priced in dollars, a typical value would be $scale = 1$.

4 Full Matrix Form

While splitting our full matrix into components as in (3.12) does improve the linear solution step 4 in Algorithm 1, we need to ensure that the fixed point policy iteration will converge. Sufficient conditions for convergence of the fixed point policy iteration were determined in [16]. These requirements are summarized in Condition 4.1.

Condition 4.1. *The matrices $\mathbf{A}(q), \mathbf{B}(q)$ and vector $c(q)$ satisfy:*

(i) The matrices $\mathbf{A}(q)$ and $\mathbf{A}(q) - \mathbf{B}(q)$ are M matrices.

(ii) The vector $c(q)$ and matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{A}^{-1}(q)$ are bounded, independent of q.

(iii) There is a constant $C_1 < 1$ such that

$$\|\mathbf{A}^{-1}(q^k)\mathbf{B}(q^{k-1})\|_\infty \leq C_1 \text{ and } \|\mathbf{A}^{-1}(q^k)\mathbf{B}(q^k)\|_\infty \leq C_1. \tag{4.1}$$

Remark 4.1. *We remind the reader that a matrix $\mathbf{A}$ is an M matrix if $\mathbf{A}$ is nonsingular, $\mathbf{A}$ has non-positive off-diagonals, and $\mathbf{A}^{-1} \geq 0$ [27].*

Theorem 4.1 (Convergence of Scheme)**.** *If the matrices $\mathbf{A}(q), \mathbf{B}(q)$ and vector $c(q)$ satisfy Condition 4.1, then the scheme in Algorithm 1 converges to the unique solution of equation (3.13), for any initial iterate v^k.*

Proof. We give a brief sketch of this proof here, and refer the reader to [16] for details. The basic idea of the proof of Theorem 4.1 is to show the algorithm generates a convergent sequence by re-arranging Algorithm 1 as:

$$\begin{aligned}\mathbf{A}(q^k)(v^{k+1} - v^k) &= -\mathbf{A}(q^k)v^k + \mathbf{B}(q^k)v^k + c(q^k) &(4.2)\\ &= \mathbf{B}(q^{k-1})(v^k - v^{k-1}) + \left[-\mathbf{A}(q^k)v^k + \mathbf{B}(q^k)v^k + c(q^k)\right] \\ &\quad - \left[-\mathbf{A}(q^{k-1})v^k + \mathbf{B}(q^{k-1})v^k + c(q^{k-1})\right]. &(4.3)\end{aligned}$$

The proof proceeds by noting that q^k maximizes $\left[-\mathbf{A}(q^k)v^k + \mathbf{B}(q^k)v^k + c(q^k)\right]$ from line 3 in Algorithm 1. Consequently

$$\left[-\mathbf{A}(q^k)v^k + \mathbf{B}(q^k)v^k + c(q^k)\right] - \left[-\mathbf{A}(q^{k-1})v^k + \mathbf{B}(q^{k-1})v^k + c(q^{k-1})\right] \geq 0 \tag{4.4}$$

and thus (since $\mathbf{A}^{-1} \geq 0$)

$$(v^{k+1} - v^k) \geq \quad \mathbf{A}^{-1}(q^k)\mathbf{B}(q^{k-1})(v^k - v^{k-1}). \tag{4.5}$$

Since $\|\mathbf{A}^{-1}(q^k)\mathbf{B}(q^{k-1})\|_\infty \leq C_1 < 1$, the sequence v^k can be shown to be convergent [16]. If $\lim_{k\to\infty} v^k = v^*$, then it is easily seen from equation (4.2) that v^* is a solution of equation (3.10). Uniqueness follows from the M matrix property of $(\mathbf{A}(q) - \mathbf{B}(q))$ [23, 5, 16]. □

Remark 4.2. *The proof of Theorem 4.1 gives some idea of what might go wrong in floating point environments. Namely, in exact arithmetic, equation (4.4) is always true. However, in floating point arithmetic, equation (4.4) is not always true and indeed convergence does not always hold in such environments. We discuss this further in a later section.*

4.1 Verification of Conditions 4.1

In this subsection we verify that Conditions 4.1 are satisfied for our Direct Control discretization, at least when the scaling parameter is large enough. Unfortunately, the technique used in [16] to verify these conditions does not work in this case as there are zero row sums in the matrix $\mathbf{A}$.

It is useful to separate the diagonal blocks from the lower triangular part of $\mathbf{A}$ by

$$\begin{aligned}
[\mathbf{A}_D^k y]_\ell &= (1-\psi_\ell^k)\bigg(\frac{1}{\Delta\tau}y_\ell - \mathcal{L}^h y_\ell + \varphi_\ell^k G\bigg(\mathcal{D}_W^h y_\ell + \frac{y_\ell}{\Delta A_j^-}\bigg)\bigg) \\
&\quad + \Pi\ \psi_\ell^k\bigg(\mathcal{D}_W^h y_\ell + \frac{y_\ell}{\Delta A_j^-}\bigg) \\
[\mathbf{A}_L^k y]_\ell &= -(1-\psi_\ell^k)\varphi_\ell^k G\bigg(\frac{y_{\ell-i_{\max}}}{\Delta A_j^-}\bigg) - \Pi\psi_\ell^k\bigg(\frac{y_{\ell-i_{\max}}}{\Delta A_j^-}\bigg)\ .
\end{aligned} \tag{4.6}$$

Then $\mathbf{A}^k = \mathbf{A}_D^k + \mathbf{A}_L^k$ for each k.

Proposition 4.1. *Suppose a positive coefficient discretization [12] is used and the jump operator $\mathcal{J}^h$ is discretized using the method in [11]. Furthermore assume there is linear behavior of the solution for $i \ge \hat{i}$ [11, 28]. Then discretization (3.11) satisfies*

(a) $\mathbf{B}(q) \ge 0$,

(b) The ℓ^{th} row sum for $\mathbf{B}(q^k)$ is

$$\textit{Row\_Sum}_{\ \ell}\ (\ \mathbf{B}(q)\ )\ \ \le\ \ \begin{cases} (1-\psi_\ell^k)\lambda & 1 < i < \hat{i}; \\ 0 & i = 1,\ i \ge \hat{i}\ . \end{cases} \tag{4.7}$$

(c) The ℓ^{th} row sum for $\mathbf{A}_D(q^k)$ is

$$\textit{Row\_Sum}_{\ \ell}\ (\ \mathbf{A}_D(q^k))\ \ =\ \ \begin{cases} (1-\psi_\ell^k)\big(\frac{1}{\Delta\tau} + (r+\lambda) + \varphi_\ell^k G\frac{1}{\Delta A_j^-}\big) + \psi_\ell^k\Pi\frac{1}{\Delta A_j^-} & 1 < i < \hat{i}; \\ (1-\psi_\ell^k)\big(\frac{1}{\Delta\tau} + r + \varphi_\ell^k G\frac{1}{\Delta A_j^-}\big) + \psi_\ell^k\Pi\frac{1}{\Delta A_j^-} & i = 1, \\ & \hat{i} \le i < i_{\max}; \\ \frac{1}{\Delta\tau} + \eta & i = i_{\max}\ . \end{cases} \tag{4.8}$$

(d) The matrices $\mathbf{A}(q) - \mathbf{B}(q)$ and $\mathbf{A}(q)$ in equation (3.11) are M matrices.

Proof. The construction of $\mathbf{B}(q)$ using the discretization of $\mathcal{J}V$ as detailed in [11] implies that

$$\sum_\ell [\mathcal{J}^h]_{\mu,\ell} \le 1 \text{ and } [\mathcal{J}^h]_{\mu,\ell} \ge 0\ . \tag{4.9}$$

This holds since $p(\xi)$ in (2.6) is a probability density function. When the grid node (i,j) satisfies $i > \hat{i}$ then the ℓ^{th} row of $\mathbf{B}(q)$ is identically zero [11, 28]. This gives (a) and (b).

In order to prove (c), we first observe that

$$\mathcal{D}^h_{WW}1 = 0 \ ; \ \mathcal{D}^h_W 1 = 0 \ ; \ \ ; \ \mathcal{D}^h_A 1 = 0$$

$$\text{and} \qquad \mathcal{L}^h 1 = \begin{cases} -(r+\lambda) & 1 < i < \hat{i} \\ -r & i = 1; \ \hat{i} \le i < i_{\max} \end{cases} \tag{4.10}$$

The row sum of $\mathbf{A}_D$ is $[\mathbf{A}_D(q^k)e]_i$ with $e = (1, ..., 1)'$, and consequently (c) follows using results (4.10), for $1 < i < i_{\max}$. When $i = i_{\max}$ then from the boundary assignment of equation (3.5), the row sum is just $(1/\Delta\tau + \eta)$.

To prove (d), note that $\mathbf{A}(q) = \mathbf{A}_D(q) + \mathbf{A}_L(q)$ with $\mathbf{A}_D(q)$ block diagonal and $\mathbf{A}_L(q)$ lower triangular. From (c), the row sums of $\mathbf{A}_D(q)$ are strictly positive, and off-diagonals are non-positive since a positive coefficient discretization is used. Hence $\mathbf{A}_D(q)$ consists of diagonal blocks, each of which is a strictly diagonally dominant M matrix. Since $\mathbf{A}_L(q)$ is non-positive, a straightforward computation shows that $\mathbf{A}(q)$ is non-singular and that $\mathbf{A}^{-1}(q) \ge 0$. Writing $(\mathbf{A} - \mathbf{B})$ as $(\mathbf{A}[I - (\mathbf{A})^{-1}\mathbf{B}])$, then it is straightforward to show that $\mathbf{A}(q) - \mathbf{B}(q)$ is also an M matrix.

□

The following Lemma will be useful later on.

Lemma 4.1. *Suppose* $\mathbf{A}$ *is a strictly diagonally dominant* M *matrix and* $\mathbf{B} \ge 0$. *Then*

$$\|\mathbf{A}^{-1}\mathbf{B}\|_\infty \ \ \le \ \ \max_i \frac{\textit{Row\_Sum}_{\ i}\ (\mathbf{B})}{\textit{Row\_Sum}_{\ i}\ (\mathbf{A})}. \tag{4.11}$$

Proof. See [16]. □

Lemma 4.2. *Suppose the discretization for the GMWB direct control method satisfies the conditions required for Proposition 4.1, and in addition*

$$\Pi > A_{\max}\lambda \frac{1 + (r+\lambda)\Delta\tau}{1 + r\Delta\tau} \ . \tag{4.12}$$

Then the matrices $\mathbf{A}, \mathbf{B}$ *satisfy Condition 4.1, and hence from Theorem 4.1, Algorithm 1 converges.*

Proof. We need to show that there is a constant C_1 such that

$$\|\mathbf{A}^{-1}(q^k)\mathbf{B}(q^p)\|_\infty \le C_1 \ , \tag{4.13}$$

where $p = k, k-1$. Consider an arbitrary vector z, and a vector y such that

$$\mathbf{A}(q^k)y \ \ = \ \ \mathbf{B}(q^p)z \ . \tag{4.14}$$

Then condition (4.13) is equivalent to requiring that

$$\max_{\|z\|_\infty \ne 0} \left[\frac{\|y\|_\infty}{\|z\|_\infty} \right] \le C_1 < 1 \tag{4.15}$$

From equation (3.11), we can see that $[\mathbf{A}e]_\ell = 0; \ i < \hat{i}, \ \psi^k_\ell = 1$, and hence we are obliged to use a different method from that in [16] to prove this result.

First, note that

$$\Big[\mathbf{A}(q^k)y\Big]_\ell \;=\; [\mathbf{B}(q^p)z\,]_l \;;\; \ell = i + (j-1) * i_{\max} \;;\; 1 < i < \hat{i} \tag{4.16}$$

$$\Big[\mathbf{A}(q^k)y\Big]_\ell \;=\; 0 \qquad\quad ;\; \ell = i + (j-1) * i_{\max} \;;\; i = 1;\; i \geq \hat{i} \tag{4.17}$$

due to the linear behaviour assumed for $i \geq \hat{i}$ [11]. Define a bounding grid function $\hat{y}$

$$[\hat{y}]_\ell \;=\; \frac{\|z\|_\infty \lambda \Delta\tau}{1 + (r+\lambda)\Delta\tau} + A_j \frac{\|z\|_\infty \lambda}{\Pi} \;\;;\;\; \ell = i + (j-1) * i_{\max}\;. \tag{4.18}$$

Noting properties (4.10), and

$$\mathcal{D}^h_W A_j = 0 \quad ; \quad \mathcal{D}^h_A A_j = 1\;, \tag{4.19}$$

and substituting equation (4.18) into equation (3.11) gives

$$\Big[\mathbf{A}(q^k)\hat{y}\Big]_\ell \;\geq\; \|z\|_\infty \lambda \;;\; 1 < i < \hat{i} \tag{4.20}$$

$$\Big[\mathbf{A}(q^k)\hat{y}\Big]_\ell \;\geq\; 0 \qquad ;\; i = 1;\; i \geq \hat{i}. \tag{4.21}$$

Subtracting equation (4.16) from equation (4.20) yields (noting properties (a) and (b) of $\mathbf{B}$ in Proposition 4.1)

$$\begin{aligned}\Big[\mathbf{A}(q^k)\,(\hat{y} - y)\Big]_\ell \;&\geq\; \|z\|_\infty \lambda - [\mathbf{B}(q^p)z]_\ell \\ &\geq\; \|z\|_\infty \lambda - \|z\|_\infty \lambda \\ &=\; 0 \qquad\qquad\qquad ;\;\; 1 < i < \hat{i}\;\;.\end{aligned} \tag{4.22}$$

Similarly, subtracting equation (4.17) from equation (4.21) gives

$$\Big[\mathbf{A}(q^k)\,(\hat{y} - y)\Big]_\ell \geq 0 \;\;;\;\; i = 1;\; i \geq \hat{i}\;. \tag{4.23}$$

Thus in all cases

$$\mathbf{A}(q^k)\,(\hat{y} - y) \geq 0 \;\;. \tag{4.24}$$

Since $\mathbf{A}(q^k)$ is an M matrix, we have that $y \leq \hat{y}$. A similar argument gives $y \geq -\hat{y}$. Hence

$$\|y\|_\infty \leq \frac{\|z\|_\infty \lambda \Delta\tau}{1 + (r+\lambda)\Delta\tau} + A_{\max} \frac{\|z\|_\infty \lambda}{\Pi}\;. \tag{4.25}$$

As we require that $\|y\|_\infty < \|z\|_\infty$, the condition on Π then becomes

$$\Pi > A_{\max} \lambda \frac{1 + (r+\lambda)\Delta\tau}{1 + r\Delta\tau}\;. \tag{4.26}$$

□

5 Block Matrix Form

Let $v_{*,j} = V_{*,j}^{n+1}$, and let $(v_{*,j})^k$ be the k^{th} iterate for $v_{*,j}$. From the boundary condition (2.11), we can observe that $v_{*,1}$ can be computed without any knowledge of interior nodes in the computational domain. To ensure a positive coefficient discretization, the $\mathcal{D}_A^h$ term is always backward differenced, hence $v_{*,j}$ depends only on $v_{*,j-1}$ for $j > 1$. This special structure of the system makes the iteration more efficient when we solve $v_{*,j}$ before proceeding to solve $v_{*,j+1}$. We can write the full matrix system in (3.11) in block form as

$$\begin{pmatrix} \mathbf{A_1} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{A_2}(q_{*,2}) & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{A_{j_{\max}}}(q_{*,j_{\max}}) \end{pmatrix} \begin{pmatrix} v_{*,1} \\ v_{*,2} \\ \vdots \\ v_{*,j_{\max}} \end{pmatrix}$$
$$= \begin{pmatrix} \mathbf{B_1} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{B_2}(q_{*,2}) & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{B_{j_{\max}}}(q_{*,j_{\max}}) \end{pmatrix} \begin{pmatrix} v_{*,1} \\ v_{*,2} \\ \vdots \\ v_{*,j_{\max}} \end{pmatrix} + \begin{pmatrix} c_{*,1} \\ c_{*,2}(q_{*,2}, v_{*,1})) \\ \vdots \\ c_{*,j_{\max}}(q_{*,j_{\max}}, v_{*,j_{\max}-1})) \end{pmatrix}, \tag{5.1}$$

with

$$q_{i,j} = \underset{q_{i,j} \in Q}{\arg\max} \Big\{ -\mathbf{A_j}(q_{*,j}) v_{*,j} + \mathbf{B_j}(q_{*,j}) v_{*,j} + c_{*,j}(q_{*,j}, v_{*,j-1}) \Big\}_i . \tag{5.2}$$

Note that $\mathbf{A_1}, \mathbf{B_1}, c_{*,1}$ are independent of q. Each smaller block matrix system $\mathbf{A_j} v_{*,j} = \mathbf{B_j} v_{*,j} + c_j$ is then resolved by using a fixed point policy iteration as in Algorithm 1 with the previous computed $v_{*,j-1}^{n+1}$ appearing only in $c_{*,j}$. The detailed procedure is given in Algorithm 2.

Algorithm 2 Block Matrix Fixed Point Policy Iteration

1: Solve $v_{*,1}$ from $\mathbf{A_1} v_{*,1} = \mathbf{B_1} v_{*,1} + c_1$
2: **for** $j = 2, 3, \ldots, j_{\max}$ **do**
3: With initial solution $(v_{*,j})^0 = V_{*,j}^n$, use Algorithm 1 to solve $v_{*,j}$ from
$\sup_{q_{*,j}} \Big\{ -\mathbf{A_j}(q_{*,j}) v_{*,j} + \mathbf{B_j}(q_{*,j}) v_{*,j} + c_{*,j}(q_{*,j}, v_{*,j-1}) \Big\} = 0$
4: **end for**

We represent the discretization (3.3) in terms of matrices $\mathbf{A_j}, \mathbf{B_j}$ and vector c_j, given by

$$\begin{aligned} [\mathbf{A_j}(\varphi_{*,j}^k, \psi_{*,j}^k) u_{*,j}]_i = [\mathbf{A_j}^k u_{*,j}]_i &= (1 - \psi_{i,j}^k) \Big(\frac{1}{\Delta\tau} u_{i,j} - \mathcal{L}^h u_{i,j} + \varphi_{i,j}^k G(\frac{u_{i,j}}{\Delta A_j^-} + \mathcal{D}_W^h u_{i,j}) \Big) \\ &\quad + \psi_{i,j}^k \Pi\, G(\frac{u_{i,j}}{\Delta A_j^-} + \mathcal{D}_W^h u_{i,j}) \\ \Big[\mathbf{B_j}(\varphi_{*,j}^k, \psi_{*,j}^k) u_{*,j} \Big]_i = [\mathbf{B_j}^k u_{*,j}]_i &= (1 - \psi_{i,j}^k) \lambda [\mathcal{J}_j^h u_{*,j}]_i \\ \Big[c_j(\varphi_{*,j}^k, \psi_{*,j}^k, u_{*,j-1}) \Big]_i = [c_{*,j}]_i^k &= (1 - \psi_{i,j}^{n+1}) \frac{1}{\Delta\tau} V_{i,j}^n + \Pi\ \psi_{i,j}^{n+1} (1 - \kappa + \frac{u_{i,j-1}^{n+1}}{\Delta A_j^-}) \end{aligned} \tag{5.3}$$

with controls

$$q^k_{i,j} \;=\; \Big(\varphi^k_{i,j},\psi^k_{i,j}\Big) \in \underset{\substack{\varphi\in\{0,1\},\psi\in\{0,1\}\\ \varphi\psi=0}}{\arg\max} \Big[-\mathbf{A_j}(\varphi_{*,j},\psi_{*,j})u_{*,j} + \mathbf{B_j}(\varphi_{*,j},\psi_{*,j})u_{*,j} + c_{*,j}(\varphi_{*,j},\psi_{*,j},u_{*,j-1})\Big]_i \,. \tag{5.4}$$

The discretized equations (3.3) then result in the set of equations

$$\begin{aligned} -\mathbf{A_j}u_{*,j} + \mathbf{B_j}u_{*,j} + c_{*,j} = 0\;, &\qquad j=1\\ \sup_{q_{i,j}\in Q}\Big[-\mathbf{A_j}(q_{*,j})u_{*,j} + \mathbf{B_j}(q_{*,j})u_{*,j} + c_{*,j}(q_{*,j},u_{*,j-1})\Big]_i = 0\;, &\qquad j=2,3,\ldots,j_{\max} \end{aligned} \tag{5.5}$$

Note that our discretization using central, forward and backward differencing implies that for each i we have

$$[\mathbf{A_j}{}^k u_{*,j}]_i \;=\; -\alpha_i u_{i-1,j} + \gamma_i u_{i,j} - \beta_i u_{i+1,j} \tag{5.6}$$

and where $\alpha_i,\beta_i,\gamma_i$ are all nonnegative. Thus the block diagonals are in tridiagonal form

$$\mathbf{A_j} = \begin{bmatrix} \gamma_i & -\beta_i & & & & \\ -\alpha_i & \gamma_i & -\beta_i & & & \\ & \ddots & \ddots & \ddots & & \\ & & & -\alpha_i & \gamma_i & -\beta_i \\ & & & & -\alpha_i & \gamma_i \end{bmatrix}. \tag{5.7}$$

5.1 Convergence Conditions for Block Form

Proposition 5.1. *Suppose a positive coefficient discretization [12] is used and the jump operator $\mathcal{J}^h$ is discretized using the method in [11] with linear behaviour assumed for $i \geq \hat{i}$ [11, 28]. Then*

(a) $\mathbf{B_j}(q^k_{*,j}) \geq 0$,

(b) The i^{th} row sums for $\mathbf{A_j}(q^k_{*,j})$ and $\mathbf{B_j}(q^k_{*,j})$ are

$$\begin{aligned} \textit{Row\_Sum}_{\,i}\ (\ \mathbf{A_j}(q^k_{*,j})\ ) \;&=\; \begin{cases} (1-\psi^k_{i,j})\big(\frac{1}{\Delta\tau} + (r+\lambda) + \varphi^k_{i,j}G\frac{1}{\Delta A^-_j}\big) + \psi^k_{i,j}\Pi\frac{1}{\Delta A^-_j} & 1<i<\hat{i};\\ (1-\psi^k_{i,j})\big(\frac{1}{\Delta\tau} + r + \varphi^k_{i,j}G\frac{1}{\Delta A^-_j}\big) + \psi^k_{i,j}\Pi\frac{1}{\Delta A^-_j} & i=1,\ \hat{i}\leq i<i_{\max};\\ \frac{1}{\Delta\tau}+\eta & i=i_{\max}\,. \end{cases}\\ \textit{Row\_Sum}_{\,i}\ (\ \mathbf{B_j}(q^k_{*,j})\ ) \;&\leq\; \begin{cases} (1-\psi^k_{i,j})\lambda & 1<i<\hat{i};\\ 0 & i=1,\ i\geq\hat{i}\,. \end{cases} \end{aligned} \tag{5.8}$$

(c) The matrices $\mathbf{A_j}(q_{*,j})-\mathbf{B_j}(q_{*,j})$ and $\mathbf{A_j}(q_{*,j})$ in equation (5.5) are strictly diagonally dominant M matrices.

Proof. The proof follows using similar arguments as in the proof of Proposition 4.1. □

Define $\Delta A_{\max} = \max_j [A_j - A_{j-1}]$. The following Lemma gives the conditions under which Algorithm 2 converges.

Lemma 5.1. *If the discretization for the GMWB direct method satisfies the conditions required for Proposition 5.1 and* $\Pi > \lambda \Delta A_{\max}$ *, then the matrices* $\mathbf{A_j}, \mathbf{B_j}$ *defined in equation* (5.5) *satisfy Condition 4.1, hence Algorithm 2 converges from Theorem 4.1.*

Proof. Suppose that

$$\max_i \frac{\text{Row\_Sum }_i \ \mathbf{B_j}(q^k_{*,j})}{\text{Row\_Sum }_i \ \mathbf{A_j}(q^k_{*,j})} = \frac{\text{Row\_Sum }_p \ \mathbf{B_j}(q^k_{*,j})}{\text{Row\_Sum }_p \ \mathbf{A_j}(q^k_{*,j})}. \tag{5.9}$$

If $1 < p < \hat{i}$ and $\psi^k_{p,j} = 0$, then Lemma 4.1 and Proposition 5.1 implies

$$\|\mathbf{A_j}^{-1}(q^k_{*,j})\mathbf{B_j}(q^k_{*,j})\|_\infty \leq \frac{\lambda}{\frac{1}{\Delta\tau} + (r+\lambda) + \frac{\varphi^k_{p,j} G}{\Delta A^-_j}} < 1 \tag{5.10}$$

When $p = 1, p \geq \hat{i}$ or $\psi^k_{p,j} = 1$, Row_Sum $_p$ $(\mathbf{B_j}(q^k_{*,j})) = 0$. In either case bound (5.10) holds.

If $1 < p < \hat{i}$, $\psi^{k-1}_{p,j} = 0$ and $\psi^k_{p,j} = 1$

$$\|\mathbf{A_j}^{-1}(q^k_{*,j})\mathbf{B_j}(q^{k-1}_{*,j})\|_\infty \leq \frac{\lambda}{\Pi \frac{1}{\Delta A^-_j}} < \frac{\Delta A^-_j}{\Delta A_{\max}} \leq 1 \ . \tag{5.11}$$

In all other cases, $\|\mathbf{A_j}^{-1}(q^k_{*,j})\mathbf{B_j}(q^{k-1}_{*,j})\|_\infty \leq C_1 < 1$ unconditionally. Repeating the above argument setting $\mathbf{B_j}(q_{*,j})$ to the identity shows that $\|\mathbf{A_j}^{-1}(q_{*,j})\|_\infty$ is bounded independent of q. □

Remark 5.1. *Choosing a scaling factor which satisfies condition* (iii) *in Condition 4.1 means that this same scaling factor must be used in the optimization step in line* 3 *of Algorithm 1. Consequently, choosing different scaling factors will result, in general, in different choices for the control at each iteration.*

Remark 5.2. *The fact that each* $\mathbf{A_j}$ *is a tridiagonal matrix simplifies the linear solution step in each fixed-point policy iteration step. In particular, implementation of the block method is quite straightforward.*

6 Floating Point Considerations

6.1 Floating Point Considerations: General Results

During the course of our numerical experiments, we observed that, even if the conditions (4.1) were satisfied, Algorithm 1 sometimes failed to converge for certain values of the direct control scaling factor. This non-convergence was a result of the oscillatory behavior of the iterates. These oscillations were above the level of the convergence tolerance, hence the scheme did not terminate.

Testing Algorithm 1 with $\mathbf{B} = 0$ was revealing. In this case, we can see that in exact arithmetic, equations (4.4-4.5) show that the iterates are monotone non-decreasing, that is, oscillations cannot

occur. However, in floating point arithmetic, equation (4.4) is not always true. In [15] (no jump case), this problem was ameliorated by forcing the right hand side of equation (4.2) to always be non-negative. However, we cannot use this approach here, when $\mathbf{B} \neq 0$.

Let $fl(x)$ be the floating point representation of a real number x. Define the error vector Δe_δ^k generated by the unit roundoff δ.

$$\Delta e_\delta^k \;=\; fl\left(-\mathbf{A}(q^k)v^k + \mathbf{B}(q^k)v^k + c(q^k)\right) - \left[-\mathbf{A}(q^k)v^k + \mathbf{B}(q^k)v^k + c(q^k)\right] \tag{6.1}$$

The floating point error in the fixed point policy iteration is dominated by the computation in equation (6.1), since the computation of these terms involves computing numerical derivatives of v^k. Numerical experiments showed that this source of error far outweighed any other source of floating point error (for example, the linear equation solve).

Note that in [4, 24], the effect of propagation of errors in policy iteration is discussed. However, in those works the error bound depended on the *effective discount rate*. In our context, the effective discount rate tends to unity as the mesh is refined. Hence the upper bound for the accumulated error in [4, 24] becomes infinite in this limit.

Consequently, we will adopt a somewhat informal, but instructive, approach in analyzing these errors in the following. Suppose that in exact arithmetic Algorithm 1 would terminate at step $k+1$. Let v^k, v^{k+1} be the iterates computed in exact arithmetic, and let Δv_δ^k be the floating point error in v^{k+1} generated by Δe_δ^k at step k. Then, from equations (4.2), (6.1),

$$\mathbf{A}(q^k)\left[(v^{k+1} - v^k) + \Delta v_\delta^k\right] = \left[-\mathbf{A}(q^k)v^k + \mathbf{B}(q^k)v^k + c(q^k)\right] + \Delta e_\delta^k \;, \tag{6.2}$$

which gives $\Delta v_\delta^k = \mathbf{A}^{-1}(q^k)\Delta e_\delta^k$. Clearly, problems will arise if

$$\frac{|(\Delta v_\delta^k)_\ell|}{\max(scale, |v_\ell^{k+1}|)} > tolerance \tag{6.3}$$

since, even if $|\left[(v^{k+1} - v^k)\right]_\ell|$ is small, the iteration will not converge according to the criteria in Algorithm 1.

Consequently, we can estimate bounds for parameters that will minimize the effect of floating point errors by requiring that

$$\max_\ell \left[\frac{|(\Delta v_\delta^k)_\ell|}{\max(|v_\ell^{k+1}|, scale)}\right] = \max_\ell \left[\frac{[\mathbf{A}^{-1}(q^k)\Delta e_\delta^k]_\ell}{\max(|v_\ell^{k+1}|, scale)}\right] < tolerance \;. \tag{6.4}$$

A rigorous bound for condition (6.4) is too pessimistic to be useful. We make the following approximation

$$\max_\ell \left[\frac{[\mathbf{A}^{-1}(q^k)\Delta e_\delta^k]_\ell}{\max(|v_\ell^{k+1}|, scale)}\right] \;\simeq\; \max_\ell \left[\frac{\|\mathbf{A}^{-1}(q^k)\|_\infty |\Delta e_\delta^k|_\ell}{\max(|v_\ell^k|, scale)}\right] , \tag{6.5}$$

so that requirement (6.4) is estimated as

$$\max_\ell \left[\frac{\|\mathbf{A}^{-1}(q^k)\|_\infty |\Delta e_\delta^k|_\ell}{\max(|v_\ell^k|, scale)}\right] < tolerance \;. \tag{6.6}$$

6.2 Floating Point Considerations: Scaled Direct Control

For the Scaled Direct Control approach, the worst case floating point error in equation (6.1) (for Π large) will be generated by the term

$$\Pi\left(1-\kappa-(\mathcal{D}_W^h v_{i,j}^k+\mathcal{D}_W^h v_{i,j}^k)\right) \ . \tag{6.7}$$

The worst roundoff error in this term occurs in the area where the grid is fine, where we subtract two nearly equal numbers. This error is then magnified by dividing by the grid spacing and multiplying by Π. In Appendix B.3, we obtain the following result (equation (B.20)),

$$\left|\left[\Delta e_\delta^k\right]_{i,j}\right| \leq 4\delta\Pi(\frac{1}{\Delta W_{\min}}+\frac{1}{\Delta A_{\min}})\max(|v_{i,j}^k|, scale) \ , \tag{6.8}$$

where $\Delta W_{\min}=\min_i(W_i-W_{i-1})$ and $\Delta A_{\min}=\min_j(A_j-A_{j-1})$. From Lemma 4.1, and Proposition 5.1 we obtain

$$\|\mathbf{A_j}^{-1}(q_{*,j}^k)\|_\infty \leq \max_p \frac{1}{\frac{1}{\Delta\tau}+(r+\lambda)+\frac{\varphi_{p,j}^k G}{\Delta A_j^-}} < \Delta\tau. \tag{6.9}$$

Consequently, from equations (6.12) and (6.9) we obtain

$$\|\mathbf{A_j}^{-1}(q_{*,j}^k)\|_\infty[\Delta e_\delta^k]_{i,j} \ \leq \ 4\Pi\delta\Delta\tau(\frac{1}{\Delta W_{\min}}+\frac{1}{\Delta A_{\min}})\max(|v_{i,j}^k|, scale) \ . \tag{6.10}$$

Now substitute equations (6.10) into equation (6.6) to obtain (where we consider $\Pi\to\infty$ when estimating $\|\mathbf{A_j}^{-1}\|_\infty$)

$$\Pi \ < \ (\frac{\text{tolerance}}{4\delta})(\frac{\Delta W_{\min}}{\Delta\tau})(\frac{1}{1+\frac{\Delta A_{\min}}{\Delta W_{\min}}}) \ . \tag{6.11}$$

Conversely, if Π is small, then the worst case floating point error will be generated by the term $\frac{1}{2}\sigma^2 W_i^2\mathcal{D}_{WW}^h V_{i,j}^{n+1}$ in equation (3.1), since this term involves a numerical second derivative. (See the definition of $\mathcal{L}$ in equation (2.6)). From the result in Appendix B, equation (B.21), we have

$$|(\Delta e_\delta^k)_{i,j}| \leq 4\delta\frac{\sigma^2 W_i^2}{(\Delta W_{\min})_i^2}\max(scale, |v_{i,j}|) \ , \tag{6.12}$$

where $(\Delta W_{\min})_i = \min(W_{i+1}-W_i, W_i-W_{i-1})$. From Lemma 4.1, using Proposition 5.1, and considering the case where $\Pi\to 0$, we obtain

$$\|\mathbf{A_j}^{-1}(q_{*,j}^k)\|_\infty \ \leq \ \frac{\Delta A_{\max}}{\Pi} \tag{6.13}$$

where $\Delta A_{\max}=\max_j(A_j-A_{j-1})$. Substituting (6.12) and (6.13) into (6.6) then gives

$$\Pi \ > \ 4\sigma^2\Delta A_{\max}\left(\frac{W_{\underline{i}}^2}{(\Delta W_{\min})_{\underline{i}}^2}\right)\left(\frac{\delta}{\text{tolerance}}\right) \ , \tag{6.14}$$

where

$$\max_i \left(\frac{W_i^2}{(\Delta W_{\min})_i^2} \right) = \frac{W_{\underline{i}}^2}{(\Delta W_{\min})_{\underline{i}}^2} \ . \tag{6.15}$$

Combining equation Lemma 5.1 and (6.14) we obtain

$$\Pi \ > \ \max\left[\lambda \Delta A_{\max},\ 4\sigma^2 \Delta A_{\max} \left(\frac{W_{\underline{i}}^2}{(\Delta W_{\min})_{\underline{i}}^2} \right) \left(\frac{\delta}{\text{tolerance}} \right) \right] \ . \tag{6.16}$$

In order to later compare the scaled direct control method with the penalty method [10, 15], let C^* be the dimensionless constant

$$\Pi = C^* \frac{\omega_0}{\Delta \tau} \ . \tag{6.17}$$

This also gives us a scaling factor having units of *currency/time* (see Remark 2.1). The upper and lower bounds of C^* for the scaled direct control method are then (from equations (6.11), (6.16), (6.17)),

$$\begin{aligned} C^* \ &> \ \max\left[\frac{\lambda \Delta A_{\max} \Delta \tau}{\omega_0},\ 4 \left(\frac{\sigma^2 \Delta A_{\max} \Delta \tau}{\omega_0} \right) \left(\frac{W_{\underline{i}}^2}{(\Delta W_{\min})_{\underline{i}}^2} \right) \left(\frac{\delta}{\text{tolerance}} \right) \right] \ , \\ C^* \ &< \ \frac{1}{4} \left(\frac{\text{tolerance}}{\delta} \right) \left(\frac{\Delta W_{\min}}{\Delta \omega_0} \right) \left(\frac{1}{1 + \frac{\Delta A_{\min}}{\Delta W_{\min}}} \right) \ . \end{aligned} \tag{6.18}$$

7 Numerical Results

In this section, we present the results for pricing a sample GMWB contract using our block formulation of the scaled direct control method. At the same time we compare these results to a block version of the penalty method [16]. A brief overview of the penalty method is given in Appendix C. The analysis of convergence of the fixed point policy iteration (block version) for the penalty method follows the same steps as used in Section 5, and will not be repeated here. As well, the methods in Section 6.1 can be used to analyse the effect of floating point errors for the penalty formulation. For the convenience of the reader, the details of this analysis are given in [14].

The contract parameters are presented in Table 7.1, with the jump diffusion parameters given in Table 7.2 and where Table 7.3 gives the mesh size and timestep parameters. In the localized computational domain $\Omega = [0, W_{\max}] \times [0, \omega_0] \times [0, T]$, we set $W_{\max} = 10^3 \omega_0$. Experiments with larger values of $W_{\max}$ produced no change in the solution to eight digits.

7.1 Convergence Results

Table 7.4 presents the convergence results for the GMWB for the scaled direct control and penalty methods (see [16] and Appendix C). Both the full matrix fixed point policy iteration scheme as described in Algorithm 1 and block matrix fixed point policy iteration scheme as described in Algorithm 2 are used. The no arbitrage fee used in the example was determined by using Newton iteration [15, 16] to solve equation (2.12). The scaling factor parameter is set to $\Pi = 10^3$ in Algorithm 1, and so satisfies Condition (4.26) and $\Pi = 1$ in Algorithm 2, which satisfies Condition (6.18). The penalty parameter [15, 14] was set to $\varepsilon = 10^4 \omega_0 / \Delta \tau$. We use an outer Newton iteration

Example GMWB Contract	
Parameter	Value
Expiry time T	10.0 years
Interest rate r	0.05
Maximum no penalty withdrawal rate G	10 per year
Withdrawal penalty κ	0.10
Initial lump-sum premium ω_0	100
Initial guarantee account balance $A(0)$	100
Initial personal annuity account balance $W(0)$	100

TABLE 7.1: *A sample GMWB contract parameters used in the numerical experiments*

Parameter	Value
ζ	.45
ν	-.9
λ	.1

TABLE 7.2: *Jump diffusion parameters.*

Level	W Nodes	A Nodes	Time steps	$\Delta W_{\min}$	$\Delta A_{\min}$	$\Delta A_{\max}$	$\Delta\tau$
0	63	56	60	1	1	2	0.16667
1	125	111	120	0.5	0.5	1	0.08333
2	249	221	240	0.25	0.25	0.5	0.04167
3	497	441	480	0.125	0.125	0.25	0.02083
4	993	881	960	0.0625	0.0625	0.125	0.01042
5	1985	1761	1920	0.03125	0.03125	0.0625	0.00521

TABLE 7.3: *Grid and timestep data for convergence experiments*

Refinement Level	Penalty Method			Scaled Direct Control Method		
	Value	Itns/step	Ratio	Value	Itns/step	Ratio
Algorithm 1 Full Matrix Iteration						
0	100.19905	83.90	N/A	101.19906	57.63	N/A
1	100.33789	189.72	N/A	100.33789	100.83	N/A
Algorithm 2 Block Matrix Iteration						
0	101.19905	4.28	N/A	101.19906	4.13	N/A
1	100.33789	4.16	N/A	100.33789	4.09	N/A
2	100.08441	4.03	3.40	100.08441	3.98	3.40
3	100.02144	3.89	4.03	100.02145	3.93	4.03
4	100.00498	3.89	3.82	100.00498	3.89	3.82
5	100.00003	3.88	3.33	100.00003	3.87	3.33

TABLE 7.4: *Convergence experiments for the GMWB guarantee value at* $t = 0$ *and* $W = A = \omega_0 = 100$ *using the penalty method, see [14],* ($\varepsilon = 10^4\omega_0/\Delta\tau$) *and scaled direct control method. Contract parameters are given in Table 7.1. Volatility* $\sigma = 0.3$ *and fair insurance fee* $\eta = 0.045452043$ *are imposed. Itns/step refers to the average number of iterations per timestep for the lines* $2-4$ *in Algorithm 2 and lines* $2-9$ *in Algorithm 1 respectively. Ratio is the ratio of successive changes in the solution as the mesh/timesteps are refined. Since the no-arbitrage fee is imposed, the numerical solution should converge to* $Value = \omega_0 = 100$.

to solve equation (2.12) for the no-arbitrage fee (relative convergence tolerance of 10^{-8}). The fixed point policy iteration convergence *tolerance* is set to 10^{-6}.

The results show that both the scaled direct control and penalty methods converge with the computed results from both methods agreeing to seven digits. The number of iterations for the full matrix fixed point policy iteration scheme is an order magnitude larger than the number of iterations for the block matrix fixed point policy iteration scheme for both methods. This is because the full matrix iteration does not take advantage of the special structure of the matrix by computing $v^{n+1}_{*,j}$ before computing $v^{n+1}_{*,j-1}$. The ratio of computational cost for these two methods is approximately equal to the ratio of iterations per timestep.

In the following, because of efficiency considerations, we will consider only the block matrix scheme.

7.2 Effect of Scaling Factor Π and Penalty Parameter ε [15]

In Section 6.2, we expressed the scaling factor Π in terms of a dimensionless parameter C^* as in (6.17). Note that the scaled direct control method does not require the existence of a constant C such that $1/\Pi = C\Delta\tau$ (see [15] for details). Writing $\Pi = C^*\omega_0/\Delta\tau$ is only for the purpose of comparing the scaled direct control method with the penalty method.

We refer to the bound on C^* imposed by effect of floating point arithmetic as a *Type I* bound. The bound on C^* imposed by requiring that $\|\mathbf{A}^{-1}(q^k)\mathbf{B}(q^{k-1})\|_\infty < 1$, will be referred to as a *Type II* bound.

Table 7.5 compares the GMWB value priced by both scaled direct control and penalty methods when $C^* \in [10^{-9}, 10^7]$. The left two columns show the estimated bounds of C^* from equation

	$tolerance = 10^{-6}$		Scaled Direct Control		Penalty Method	
Type	Bound	Π or $1/\varepsilon$	Value	Itns/step	Value	Itns/step
I	$0.67 \times 10^{-9}\omega_0/\Delta\tau$	$10^{-9}\omega_0/\Delta\tau$	N/A	N/A		
		$10^{-8}\omega_0/\Delta\tau$	99.999992*	4.03		
		$10^{-7}\omega_0/\Delta\tau$	99.999992*	3.97		
II	$0.33 \times 10^{-6}\omega_0/\Delta\tau$	$10^{-6}\omega_0/\Delta\tau$	100.00003	3.94		
		...	...	...	...	
		$10^{-1}\omega_0/\Delta\tau$	100.00003	3.82	99.969172	3.83
		$10^{0}\omega_0/\Delta\tau$	100.00003	3.83	99.996899	3.85
		$10^{1}\omega_0/\Delta\tau$	100.00003	3.87	99.999715	3.87
		$10^{2}\omega_0/\Delta\tau$	100.00003	3.88	99.999998	3.88
		$10^{3}\omega_0/\Delta\tau$	100.00003	3.88	100.00002	3.88
		$10^{4}\omega_0/\Delta\tau$	100.00003	3.88	100.00003	3.88
		$10^{5}\omega_0/\Delta\tau$	100.00003	3.88	100.00003	3.88
I	$0.35 \times 10^{6}\omega_0/\Delta\tau$	$10^{6}\omega_0/\Delta\tau$	100.00003	3.88	100.00002	3.88
		$10^{7}\omega_0/\Delta\tau$	N/A	N/A	N/A	N/A

TABLE 7.5: *The effect of the scaling factor* Π *and penalty parameter [15, 14]* $1/\varepsilon$ *in terms of* C^* *on pricing the GMWB guarantee at refinement level 5.* $\sigma = 0.3, W = A = 100$ *and* $t = 0$. *Fair insurance fee (i.e.* $\eta = 0.045452043$*) is imposed. Contract parameters are given in Table 7.1. Jump diffusion parameters are given in Table 7.2. Itns/step refers to the average number of iterations per timestep for the lines* $2-4$ *in Algorithm 2. Type I bounds refer to bounds based on floating point considerations. Type II bounds refer to sufficient conditions for convergence in exact arithmetic, from Condition 4.1.*

(6.18). (These same bounds are also valid for the penalty method, see [14] for details). Recall the definition of $\underline{i}$ from equation (6.15). The finest grids are around node $(W = 100, A = 100)$, so we set $W_{\underline{i}} = 100$, in the estimate of the floating point errors in equation (6.18). We take double precision machine epsilon to be $\delta = 1.11 \times 10^{-16}$. The "N/A" entries in the table indicate that the iterative scheme did not satisfy the convergence criteria in Algorithm 1 after 6000 iterations.

For the entries where the computed values have asterisks, although the convergence criterion in line 3 of Algorithm 1 was satisfied, we view these results as unreliable. Note that the convergence criterion in Algorithm 1 is not able to clearly distinguish between very slowly diverging sequences and truly converging sequences. We remind the reader that the Type II bound from Lemma 5.1 is a sufficient condition for convergence in exact arithmetic, from Condition 4.1. However, choosing a C^* smaller than the estimated lower bound of C^* from bound (6.18) produces questionable results. It is obvious the values with asterisks deviate somewhat from the other values.

In the previous numerical examples, the lower bound for the scaling factor Π is dominated by a *Type II* bound. To see the effect of *Type I* lower bound in isolation, we remove the jump diffusion from the underlying asset model (e.g. $\lambda = 0$). Consequently, $\|\mathbf{A}^{-1}(q^k)\mathbf{B}(q^{k-1})\|_\infty < 1$ always holds since $\mathbf{B}(q^k) = 0$ for all k and hence the *Type II* bound disappears.

Table 7.6 shows the GMWB values priced at refinement level 5 ($\lambda = 0$). Without the presence of *Type II* bound, we can further decrease the scaling factor by three orders of magnitude. The estimated *Type I* lower bound for C^* is remarkably close to the experimental result.

Remark 7.1 (Range of Values)**.** *These examples clearly show that the range of useful values of the scaling parameter for the scaled direct control method is much larger than the range of useful values*

	$tolerance = 10^{-6}$		Scaled Direct Control		Penalty Method	
Type	Bound	Π or $1/\varepsilon$	Value	Itns/step	Value	Itns/step
		$10^{-11}\omega_0/\Delta\tau$	N/A	N/A		
		$10^{-10}\omega_0/\Delta\tau$	115.88596	2.69		
I	$0.67 \times 10^{-9}\omega_0/\Delta\tau$	$10^{-9}\omega_0/\Delta\tau$	115.88596	2.69		
		$10^{-8}\omega_0/\Delta\tau$	115.88596	2.69		
		$10^{-7}\omega_0/\Delta\tau$	115.88596	2.69		
II	$0.33 \times 10^{-6}\omega_0/\Delta\tau$	$10^{-6}\omega_0/\Delta\tau$	115.88596	2.69		
		...	...	...	...	
		$10^{-1}\omega_0/\Delta\tau$	115.88596	2.84	115.85508	2.85
		$10^{0}\omega_0/\Delta\tau$	115.88596	2.85	115.88281	2.84
		$10^{1}\omega_0/\Delta\tau$	115.88596	2.85	115.88565	2.85
		$10^{2}\omega_0/\Delta\tau$	115.88596	2.85	115.88593	2.85
		$10^{3}\omega_0/\Delta\tau$	115.88596	2.85	115.88596	2.85
		$10^{4}\omega_0/\Delta\tau$	115.88596	2.85	115.88596	2.85
		$10^{5}\omega_0/\Delta\tau$	115.88596	2.85	115.88596	2.85
I	$0.35 \times 10^{6}\omega_0/\Delta\tau$	$10^{6}\omega_0/\Delta\tau$	115.88596	2.87	115.88596	2.86
		$10^{7}\omega_0/\Delta\tau$	N/A	N/A	N/A	N/A

TABLE 7.6: *The effect of Type I upper and lower bounds on the scale factor* Π *and penalty parameter* $1/\varepsilon$ *on pricing the GMWB guarantee at refinement level 5. No jump diffusion presented.* $\sigma = 0.3, W = A = 100$ *and* $t = 0$. *No insurance fee (i.e.* $\eta = 0$*) is imposed. Fully implicit method is used. Contract parameters are given in Table 7.1. Itns/step refers to the average number of iterations per timestep for the lines* $2 - 4$ *in Algorithm 2. Type I bounds refer to bounds based on floating point considerations. Type II bounds refer to sufficient conditions for convergence in exact arithmetic. In this case the Type II bound is not required since the jump term is absent.*

for the penalty parameter in the penalty method.

In Appendix D, we report some further numerical experiments which confirm that our estimates for the upper and lower bounds for the scaling factor are of the correct order.

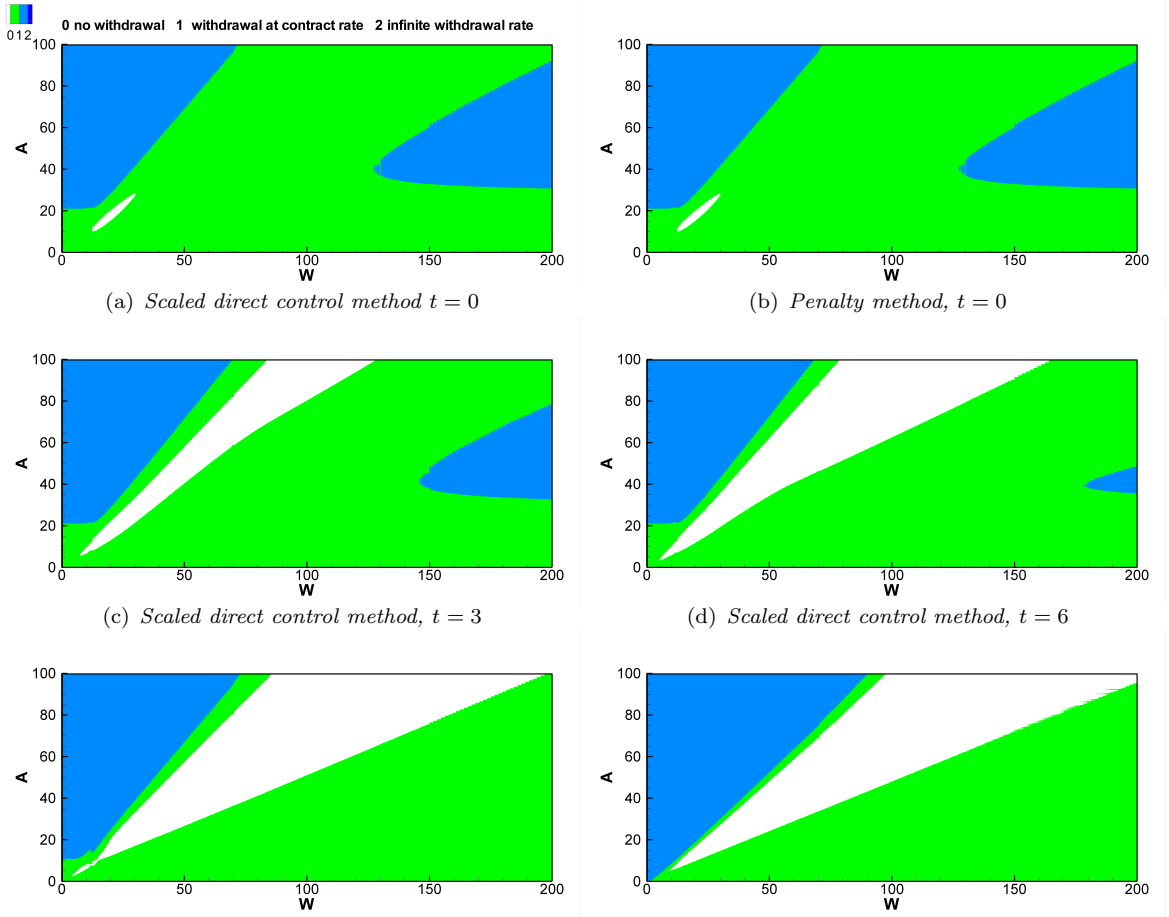
A GMWB contract holder is perhaps more interested in the optimal withdrawal strategy. Figure 7.1 shows contour plots of the optimal withdrawal strategy at various times. The top two plots in Figure 7.1 are generated by both the scaled direct control and penalty methods. It can be observed that these contour plots are very similar.

The other plots in Figure 7.1 are generated by the scaled direct control method. It is interesting to observe that the top left corner infinite withdrawal region is almost time-invariant, except when the contract is close to expiration. The no withdrawal region widens as time moves forward. These results are consistent with the discrete withdrawal computations in [8].

8 Conclusions

In this paper, we have carried out a systematic study of a scaled direct control iterative method for solution of the GMWB pricing problem formulated as a singular control problem. The underlying asset is assumed to follow a jump diffusion process.

We represent the discretized equations for the scaled direct control technique using the general form of controlled HJB equations as in [16]. Sufficient conditions for convergence for a fixed point



(a) *Scaled direct control method* $t = 0$

(b) *Penalty method,* $t = 0$

(c) *Scaled direct control method,* $t = 3$

(d) *Scaled direct control method,* $t = 6$

(e) *Scaled direct control method,* $t = 9$

(f) *Scaled direct control method,* $t = 10 - \Delta\tau$

FIGURE 7.1: *Contour plot of the optimal withdrawal strategy for the GMWB guarantee at different times in the* (W, A) *-plane.* $\sigma = 0.2$. *A fair insurance fee of* $\eta = 0.032296686$ *is imposed. Contract parameters are given in Table 7.1 and jump diffusion parameters are given in Table 7.2. The penalty parameter [15, 14] is set to* $1/\varepsilon = 10^4\omega_0/\Delta\tau$ *and the scaling factor is set to* $\Pi = 1$. *The iteration convergence tolerance is set to* 10^{-6}.

policy iteration are determined. Convergence can always be guaranteed with a suitable choice for the scaling parameter.

The equations resulting from the singular control formation have a special structure that can significantly improve the efficiency of solving the resulting nonlinear system. A block matrix fixed point policy iteration scheme is given and the conditions required for convergence are determined. We compare the block version of the scaled direct control method with a block version of the penalty method [16, 15, 14]. Numerical experiments were carried out to compare the direct control and penalty methods (both in block and non-block forms). The experiments show that the block methods are an order of magnitude better in terms of number of iterations, compared to a full matrix formulation, in both cases.

Both the direct control and the penalty method require specification of a parameter. This parameter affects both convergence and accuracy. We have estimated bounds for these parameters for both methods, so that convergence in floating point arithmetic can be expected. To the best of our knowledge, such analysis has not been carried out previously. Numerical experiments indicate that these estimates are reasonably accurate.

From a practical perspective, it is safe to choose the dimensionless penalty parameter, and the dimensionless direct control scaling factor, two orders of magnitude below the estimated upper bounds.

It would appear that the order of magnitude useful range of the scaling parameter for the direct control method is much larger than the useful range for the penalty parameter in the penalty method. The accuracy and convergence rate for both methods is similar for parameters within the useful range. Consequently, it would appear that the scaled direct control method is superior to the penalty method in this regard.

Appendix

A Finite Difference Approximation

A.1 First and second derivatives approximation

In this Appendix, we use a standard finite difference method to approximate the first and second partial derivatives in the PDE. The discretized differential operators $\mathcal{D}_A^h$, $\mathcal{D}_W^h$ and $\mathcal{D}_{WW}^h$ are given

by

$$\mathcal{D}_A^h V_{i,j}^n = \frac{V_{i,j}^n - V_{i,j-1}^n}{\Delta A_j^-}, \quad \text{backward differencing,}$$

$$\mathcal{D}_W^h V_{i,j}^n = \begin{cases} \frac{V_{i,j}^n - V_{i-1,j}^n}{\Delta W_i^-} & \text{backward differencing,} \\ \frac{V_{i+1,j}^n - V_{i,j}}{\Delta W_i^+} & \text{forward differencing,} \\ \frac{V_{i+1,j}^n - V_{i-1,j}^n}{\Delta W_i^\pm} & \text{central differencing,} \end{cases}$$

$$\begin{aligned} \mathcal{D}_{WW}^h V_{i,j}^n &= \frac{\frac{V_{i-1,j}^n - V_{i,j}^n}{\Delta W_i^-} + \frac{V_{i+1,j}^n - V_{i,j}^n}{\Delta W_i^+}}{\frac{\Delta W_i^\pm}{2}} \\ &= \frac{\mathcal{D}_W^h V_{i+1,j}^n - \mathcal{D}_W^h V_{i,j}^n}{\frac{\Delta W_i^\pm}{2}} \quad (\mathcal{D}_W^h \text{ is backward differenced}). \end{aligned} \tag{A.1}$$

where $\Delta A_j^- = A_j - A_{j-1}$, $\Delta W_i^- = W_i - W_{i-1}$, $\Delta W_i^+ = W_{i+1} - W_i$, and $\Delta W_i^\pm = W_{i+1} - W_{i-1}$.

A.2 The discretized $\mathcal{L}^h$ and $\mathcal{F}^h$ operators

Define the following sets of point s $(W, A, \tau) \in \Omega$

$$\begin{aligned} \Omega_{\tau^0} &= [0, W_{\max}] \times [0, \omega_0] \times \{0\}\ , \\ \Omega_{W_0} &= \{0\} \times (0, \omega_0] \times (0, T] \\ \Omega_{W_{\max}} &= \{W_{\max}\} \times (0, \omega_0] \times (0, T] \\ \Omega_{A_0} &= [0, W_{\max}] \times \{0\} \times (0, T] \\ \Omega_{in} &= \Omega \backslash \Omega_{\tau^0} \backslash \Omega_{W_0} \backslash \Omega_{W_{\max}} \backslash \Omega_{A_0}\ . \end{aligned} \tag{A.2}$$

Together with the boundary condition (2.11), the discretized $\mathcal{L}^h$ and $\mathcal{F}^h$ [15] operators are

$$\mathcal{L}^h V_{i,j}^n = \begin{cases} \frac{\sigma^2}{2} W_i^2 \mathcal{D}_{WW}^h V_{i,j}^n + (r - \eta) W_i \mathcal{D}_W^h V_{i,j}^n - r V_{i,j}^n, & (W_i, A_j, \tau^n) \in \Omega_{in} \cup \Omega_{A_0} \\ -r V_{i,j}^n, & (W_i, A_j, \tau^n) \in \Omega_{W_0} \end{cases}, \tag{A.3}$$

$$\mathcal{F}^h V_{i,j}^n = \begin{cases} 1 - \mathcal{D}_W^h V_{i,j}^n - \mathcal{D}_A^h V_{i,j}^n, & (W_i, A_j, \tau^n) \in \Omega_{in} \\ 1 - \mathcal{D}_A^h V_{i,j}^n, & (W_i, A_j, \tau^n) \in \Omega_{W_0} \\ 0, & (W_i, A_j, \tau^n) \in \Omega_{A_0} \end{cases}. \tag{A.4}$$

B Floating Point Arithmetic Error Analysis

B.1 Roundoff Error Propagation

Let $x = (x_1, x_2, \ldots, x_n)'$ to compute a function $y = \phi(x)$ by using floating point arithmetic, an error Δy of $y\delta_0$ has to be expected, where $|\delta_0| < \delta$, the machine epsilon [25]. Further more, there exists an input error $\Delta x = (\Delta x_1, \Delta x_2, \ldots, \Delta x_n)'$ due to the floating point representation of real numbers or previous calculation of x (we do not consider the measurement input error because it is beyond the control of numerical computation method). The two sources of error are unavoidable

no matter how we arrange the floating point operations. The third source of error comes from the intermediate roundoff errors and it depends how we arrange the floating point operations. Based on differential error analysis, the total floating point arithmetic error of computing y denoted by Δy, to the first order approximation, is given by

$$\Delta y = \mathcal{D}\phi(x)\Delta x + y\delta_0 \ + \sum_{i=1}^{r} \Delta^{(i)} y \tag{B.1}$$

with $\mathcal{D}\phi(x)$ being the Jacobian matrix of $\phi(x)$ and $\Delta^{(i)}y$ being the intermediate roundoff error generated at step i. We assume there are r intermediate steps and each step performs elementary operations such as $+, -, \times, \div$ and $\sqrt{}$ [25].

B.2 Derivative Roundoff Error by Finite Difference

Using the standard finite difference method to compute the first derivative involves floating point arithmetic of computing the function with form

$$y = \phi_1(x) = \frac{x_1 - x_2}{x_3} \ . \tag{B.2}$$

Let the input relative error be denoted by $\delta_x = (\delta_{x_1}, \delta_{x_2}, \delta_{x_3})' = (\Delta x_1/x_1, \Delta x_2/x_2, \Delta x_3/x_3)'$, If we compute $y_1 = x_1 - x_2$ first, then proceed to divide the intermediate result y_1 by x_3, from equation (B.1), we have

$$\Delta y \ = \ \delta_{x_1}\frac{x_1}{x_3} - \delta_{x_2}\frac{x_2}{x_3} - \delta_{x_3} y + \delta_0 y + \delta_1 y$$

where $|\delta_i| < \delta (i = 1, 2)$ and $\delta_1 y$ is the intermediate roundoff error. Further assuming $|\delta_{x_3}| \leq \delta$ and $|x_3| \leq \Delta h_{\min}$, we obtain the bound of Δy as follows

$$|\Delta y| \ \leq \ |\delta_{x_1}|\frac{|x_1|}{\Delta h_{\min}} + |\delta_{x_2}|\frac{|x_2|}{\Delta h_{\min}} + 3\delta|y| \tag{B.3}$$

Let

$$x_2 = (1 + a_1)x_1, \ x_1 = (1 + a_2)x_2, \qquad |x_3| \leq \Delta h_{\min} \ . \tag{B.4}$$

The bound of y is given by

$$|y| \leq |a_i|\frac{|x_i|}{\Delta h_{\min}} \ , \qquad i = 1, 2 \ . \tag{B.5}$$

Suppose input error Δx is due to representing the real number in floating point system or previous calculation whose error is within machine epsilon δ, so we have $\|\delta_x\|_\infty \leq \delta$ and consequently

$$|\Delta y| \leq \delta(2 + 4|a_i|)\frac{|x_i|}{\Delta h_{\min}} \qquad i = 1, 2 \ . \tag{B.6}$$

Apply the result to discretized $\mathcal{D}^h_W V^{n+1}_{i,j}$ and $\mathcal{D}^h_A V^{n+1}_{i,j}$ in equation (A.1), we obtain the absolute roundoff error of computing first derivatives by using backward difference as follows

$$\begin{aligned}
|\Delta\mathcal{D}^h_A V^{n+1}_{i,j}| \leq &\quad \delta(2+4|a_3|)\frac{|V^{n+1}_{i,j}|}{\Delta A^-_j} \ , \quad V^{n+1}_{i,j-1} = (1+a_3)V^{n+1}_{i,j} \\
|\Delta\mathcal{D}^h_W V^{n+1}_{i,j}| \leq &\quad \delta(2+4|a_4|)\frac{|V^{n+1}_{i,j}|}{\Delta W^-_i} \ , \quad V^{n+1}_{i-1,j} = (1+a_4)V^{n+1}_{i,j} \\
|\Delta\mathcal{D}^h_W V^{n+1}_{i+1,j}| \leq &\quad \delta(2+4|a_5|)\frac{|V^{n+1}_{i,j}|}{\Delta W^-_i} \ , \quad V^{n+1}_{i+1,j} = (1+a_5)V^{n+1}_{i,j}
\end{aligned} \tag{B.7}$$

Apply the result in equation (B.5) to $\mathcal{D}^h_A V_{i,j}$, $\mathcal{D}^h_W V_{i,j}$ and $\mathcal{D}^h_W V^{n+1}_{i+1,j}$, we have

$$|\mathcal{D}^h_A V^{n+1}_{i,j}| \leq |a_3|\frac{|V^{n+1}_{i,j}|}{\Delta A^-_j} \ , \quad |\mathcal{D}^h_W V^{n+1}_{i,j}| \leq |a_4|\frac{|V^{n+1}_{i,j}|}{(\Delta W_{\min})_i} \ , \qquad |\mathcal{D}^h_W V^{n+1}_{i+1,j}| \leq |a_5|\frac{|V^{n+1}_{i,j}|}{(\Delta W_{min})_i} \ , \tag{B.8}$$

where $(\Delta W_{\min})_i = \min(W_{i+1} - W_i, W_i - W_{i-1})$. Together with the standard 3 point finite difference method to compute the second derivative as in equation (A.1) , we obtain

$$\begin{aligned}
|\mathcal{D}^h_{WW} V^{n+1}_{i,j}| &\leq (|\mathcal{D}^h_W V^{n+1}_{i+1,j}| + |\mathcal{D}^h_W V^{n+1}_{i,j}|)\frac{1}{(\Delta W_{\min})_i} \\
&\leq (|a_4| + |a_5|)\frac{|V^{n+1}_{i,j}|}{(\Delta W_{\min})^2_i} \ .
\end{aligned} \tag{B.9}$$

To bound the roundoff error of $\mathcal{D}^h_{WW} V^{n+1}_{i,j}$, set

$$x = (\mathcal{D}^h_W V^{n+1}_{i+1,j}, \mathcal{D}^h_W V^{n+1}_{i,j}, \frac{\Delta W^{\pm}_i}{2})' \ , \qquad \delta_x = (\frac{\Delta\mathcal{D}^h_W V^{n+1}_{i+1,j}}{\mathcal{D}^h_W V^{n+1}_{i+1,j}}, \frac{\Delta\mathcal{D}^h_W V^{n+1}_{i,j}}{\mathcal{D}^h_W V^{n+1}_{i,j}}, \delta_{x_3})' \ . \tag{B.10}$$

Assuming $|\delta_{x_3}| < \delta$, by equations (A.1), (B.3), (B.8) and (B.9) and the fact that $\Delta W^{\pm}_i/2 \geq \Delta(W_{\min})_i$, we obtain the following bound

$$\begin{aligned}
|\Delta\mathcal{D}^h_{WW} V^{n+1}_{i,j}| &\leq \frac{|\Delta\mathcal{D}^h_W V^{n+1}_{i,j}| + |\Delta\mathcal{D}^h_W V^{n+1}_{i+1,j}|}{(\Delta W_{\min})_i} + 3|\mathcal{D}^h_{WW} V^{n+1}_{i,j}| \\
&\leq \delta(4 + 5|a_4| + 5|a_5|)\frac{|V^{n+1}_{i,j}|}{(\Delta W_{\min})^2_i}
\end{aligned} \tag{B.11}$$

B.3 Roundoff Error Estimation of Local Optimization Problem

During each iteration, we solve local optimization problem and the objective function involves calculating the following two terms

$$f_1(W_i, A_j, \tau^{n+1}) = \kappa - 1 + \mathcal{D}^h_W V^{n+1}_{i,j} + \mathcal{D}^h_A V^{n+1}_{i,j} \tag{B.12}$$

$$f_2(W_i, A_j, \tau^{n+1}) = \frac{\sigma^2 W^2_i}{2}\mathcal{D}^h_{WW} V^{n+1}_{i,j} + O(W_i) \ . \tag{B.13}$$

Computing f_1 involves calculating a function of the form $g(x) = (x_1 + x_2) + (x_3 + x_4)$. From equation (B.1), we obtain

$$\begin{aligned} |\Delta g| &\leq \sum_{i=1}^{4} |\Delta x_i| + \delta|g| + \delta|x_1 + x_2| + \delta|x_3 + x_4| \quad (B.14) \\ &\leq \sum_{i=1}^{4} |\Delta x_i| + 2\delta|x_1 + x_2| + 2\delta(|x_3| + |x_4|) \ . \quad (B.15) \end{aligned}$$

Set $x_1 = \kappa, x_2 = -1, x_3 = \mathcal{D}_W^h V_{i,j}^{n+1}, x_4 = \mathcal{D}_A^h V_{i,j}^{n+1}$ and apply equations (B.7) and (B.8) with the fact that $0 < \kappa < 1$, we obtain the bound of absolute roundoff error of f_1 as follows

$$\begin{aligned} |\Delta f_1| &\leq \delta(2 + 6|a_3|)\frac{|V_{i,j}^{n+1}|}{\Delta A_j^-} + \delta(2 + 6|a_4|)\frac{|V_{i,j}^{n+1}|}{(\Delta W_{\min})_i} + 3(1-\kappa)\delta \\ &\lessapprox 2\delta(\frac{1 + 3|a_3|}{\Delta A_{\min}} + \frac{1 + 3|a_4|}{\Delta W_{\min}})|V_{i,j}^{n+1}| \ , \quad (B.16) \end{aligned}$$

where we discard the smaller error term of $3\delta(1-\kappa)$, and $\Delta A_{\min} = \min_j(A_j - A_{j-1}$, $\Delta W_{\min} = \min_i(W_i - W_{i-1})$.

To analyze the roundoff error of f_2, we notice that only multiplication and division operations are involved given $\mathcal{D}_{WW}^h V_{i,j}^{n+1}$ as one of the operands. From equation (B.1), it can be easily seen that the roundoff error of

$$g_2(x) = x_1 \times x_2 \qquad |\Delta g_2| \leq (|\delta_{x_1}| + |\delta_{x_2}| + |\delta_0|)|g_2| \quad (B.17)$$

$$g_3(x) = x_1 \div x_2 \qquad |\Delta g_2| \leq (|\delta_{x_1}| + |\delta_{x_2}| + |\delta_0|)|g_3| \quad (B.18)$$

So computing of $c_i = \sigma^2 W_i^2/2$ will accumulate $9\delta|c_i|$ roundoff errors assuming the input error of σ and W_i is smaller than δ, the machine epsilon. The final roundoff error of $f_2 = c_i \mathcal{D}_{WW}^h V_{i,j}^{n+1}$ is then given by

$$\begin{aligned} |\Delta f_2| &\leq (10|\delta| + \frac{|\Delta \mathcal{D}_{WW}^h V_{i,j}^{n+1}|}{|\mathcal{D}_{WW}^h V_{i,j}^{n+1}|})|c_i||\mathcal{D}_{WW}^h V_{i,j}^{n+1}| \\ &\leq \delta(4 + 15|a_4| + 15|a_5|)\frac{\sigma^2 W_i^2}{2(\Delta W_{\min})_i^2}|V_{i,j}^{n+1}| \quad (B.19) \end{aligned}$$

In the area where the grids are fine, we have $V_{i,j} \approx V_{i\pm1,j} \approx V_{i,j-1}$. So normally $|a_i| \ll 1$ for $i = 3, 4, 5$. It may be safe to estimate that $|a_i| \leq 0.1, i = 3, 4, 5$. Finally the following estimation of roundoff errors of computing f_1 and f_2 are obtained.

$$\begin{aligned} |\Delta(\kappa - 1 - \mathcal{D}_W^h V_{i,j}^{n+1} + \mathcal{D}_A^h V_{i,j}^{n+1})| &\leq 4\delta(\frac{1}{\Delta A_{\min}} + \frac{1}{\Delta W_{\min}})|V_{i,j}^{n+1}| \\ &\leq 4\delta(\frac{1}{\Delta A_{\min}} + \frac{1}{\Delta W_{\min}})\max(|V_{i,j}^{n+1}|, scale) \ , \quad (B.20) \\ |\Delta(\frac{\sigma^2 W_i^2}{2}\mathcal{D}_{WW}^h V_{i,j}^{n+1})| &\leq 4\delta\frac{\sigma^2 W_i^2}{(\Delta W_{\min})_i^2}|V_{i,j}^{n+1}| \\ &\leq 4\delta\frac{\sigma^2 W_i^2}{(\Delta W_{\min})_i^2}\max(|V_{i,j}^{n+1}|, scale) \ . \quad (B.21) \end{aligned}$$

C The Penalty Method for solving the GMWB Problem

A penalty method for solving the singular GMWB problem under a jump diffusion process has been given in [16]. The penalized form of equation (2.5) is

$$V_\tau^\varepsilon = \mathcal{L}V^\varepsilon + \lambda \mathcal{J}V^\varepsilon + \max_{\substack{\varphi\in\{0,1\},\psi\in\{0,1\}\\ \varphi\psi=0}} \left[\varphi G \mathcal{F} V^\varepsilon + \psi \left(\frac{(\mathcal{F}V^\varepsilon - \kappa)}{\varepsilon} + \kappa G \right) \right]. \tag{C.1}$$

or equivalently

$$V_\tau^\varepsilon - \lambda \mathcal{J}V^\varepsilon + \min_{\substack{\varphi\in\{0,1\},\psi\in\{0,1\}\\ \varphi\psi=0}} \left[-\mathcal{L}V^\varepsilon - \varphi G \mathcal{F} V^\varepsilon - \psi \left(\frac{(\mathcal{F}V^\varepsilon - \kappa)}{\varepsilon} + \kappa G \right) \right] = 0. \tag{C.2}$$

The basic idea of the penalty method is to discretize equation (C.1), and let the penalty parameter $\varepsilon \to 0$ as the mesh and timesteps tend to zero. In [16] the penalty method was discretized with a fully implicit method and a fixed point policy iteration method was given to solve the resulting nonlinear algebraic equations.

D Confirmation of the Upper and Lower Bound Estimates for the Scaling Factor

In this appendix, we report some further experiments which confirm that our estimates for $[C^*_{\min}, C^*_{\max}]$ are of the correct order. We determined the order of magnitude of $[C^*_{\min}, C^*_{\max}]$, as a function of convergence tolerance, such that $\forall C^* \in [C^*_{\min}, C^*_{\max}]$ the computed GMWB values agree to $(n+1)^{th}$ digit, where $tolerance = 10^{-n}$. Table D.1 compares the computed order of magnitude of $C^*_{\max}$ and $C^*_{\min}$ with the estimated C^* upper and lower bounds from equation (6.18).

A similar experiment was also conducted to seek the order of magnitude of the range $[\Pi_{\min}, \Pi_{\max}]$, as a function of iteration tolerance, such that $\forall\, \Pi \in [\Pi_{\min}, \Pi_{\max}]$ the computed GMWB values agree to $(n+1)^{th}$ digit, where $tolerance = 10^{-n}$. Table D.2 compares the computed order of magnitude of $\Pi_{\max}$ and $\Pi_{\min}$ with the estimated Π upper and lower bounds from equations (6.11) and (6.16).

References

[1] L. Andersen and J. Andreasen. Jump-diffusion processes: Volatility smile fitting and numerical methods for option pricing. *Review of Derivatives Research*, 4:231–262, 2000.

[2] G. Barles. Convergence of numerical schemes for degenerate parabolic equations arising in finance. In L. C. G. Rogers and D. Talay, editors, *Numerical Methods in Finance*, pages 1–21. Cambridge University Press, Cambridge, 1997.

[3] G. Barles and P.E. Souganidis. Convergence of approximation schemes for fully nonlinear equations. *Asymptotic Analysis*, 4:271–283, 1991.

[4] D.P. Bertsekas and J. Tsitsiklis. *Neuro-dynamic Programming.* Athena, 1996.

	Tolerance					
	10^{-6}		10^{-8}		10^{-10}	
Level	C^* *upper bound*	$C^*_{\max}$	C^* *upper bound*	$C^*_{\max}$	C^* *upper bound*	$C^*_{\max}$
0	0.11×10^8	10^8	0.11×10^6	10^7	0.11×10^4	10^6
1	0.56×10^7	10^7	0.56×10^5	10^6	0.56×10^3	10^5
2	0.28×10^7	10^7	0.28×10^5	10^5	0.28×10^3	10^5
3	0.14×10^7	10^7	0.14×10^5	10^5	0.14×10^3	10^4
4	0.70×10^6	10^6	0.70×10^4	10^4	0.70×10^2	10^3
5	0.35×10^6	10^6	0.35×10^4	10^4	0.35×10^2	10^2
Level	C^* *lower bound*	$C^*_{\min}$	C^* *lower bound*	$C^*_{\min}$	C^* *lower bound*	$C^*_{\min}$
0	0.33×10^{-3}	10^{-4}	0.33×10^{-3}	10^{-4}	0.33×10^{-3}	10^{-4}
1	0.83×10^{-4}	10^{-4}	0.83×10^{-4}	10^{-5}	0.83×10^{-4}	10^{-5}
2	0.21×10^{-4}	10^{-5}	0.21×10^{-4}	10^{-6}	0.21×10^{-4}	10^{-5}
3	0.52×10^{-5}	10^{-5}	0.52×10^{-5}	10^{-7}	0.67×10^{-5}	10^{-5}
4	0.13×10^{-5}	10^{-5}	0.13×10^{-5}	10^{-6}	0.67×10^{-5}	10^{-5}
5	0.33×10^{-6}	10^{-6}	0.33×10^{-6}	10^{-7}	0.67×10^{-5}	10^{-5}

TABLE D.1: *Experimental C^* upper ($C^*_{\max}$)and lower ($C^*_{\min}$) bounds as a function of iteration convergence tolerance. The theoretical bounds C^* upper bound and C^* lower bound are also shown. Both penalty and scaled direct control method with block matrix implementation as in Algorithm 2 and produce the same results of $C^*_{\max}$. Scaled direct control method is used for computing $C^*_{\min}$. Contract parameters are in Table 7.1 and Jump diffusion parameters are in Table 7.2. $\sigma = 0.3, \eta = 0.045452043$. Finest grids are around node $(W, A) = (100, 100)$, which are used to compute the bounds.*

	Tolerance					
	10^{-6}		10^{-8}		10^{-10}	
Level	Π *upper bound*	$\Pi_{\max}$	Π *upper bound*	$\Pi_{\max}$	Π *upper bound*	$\Pi_{\max}$
0	0.68×10^{10}	10^{11}	0.68×10^{8}	10^{9}	0.68×10^{6}	10^{9}
1	0.68×10^{10}	10^{11}	0.68×10^{8}	10^{9}	0.68×10^{6}	10^{8}
2	0.68×10^{10}	10^{10}	0.68×10^{8}	10^{9}	0.68×10^{6}	10^{8}
3	0.68×10^{10}	10^{10}	0.68×10^{8}	10^{9}	0.68×10^{6}	10^{7}
4	0.68×10^{10}	10^{10}	0.68×10^{8}	10^{8}	0.68×10^{6}	10^{7}
5	0.68×10^{10}	10^{10}	0.68×10^{8}	10^{8}	0.68×10^{6}	10^{6}
Level	Π *lower bound*	$\Pi_{\min}$	Π *lower bound*	$\Pi_{\min}$	Π *lower bound*	$\Pi_{\min}$
0	0.20×10^{0}	10^{-2}	0.20×10^{0}	10^{-2}	0.20×10^{0}	10^{-2}
1	0.10×10^{0}	10^{-1}	0.10×10^{0}	10^{-2}	0.10×10^{0}	10^{-2}
2	0.50×10^{-1}	10^{-1}	0.50×10^{-1}	10^{-2}	0.50×10^{-1}	10^{-1}
3	0.25×10^{-1}	10^{-2}	0.25×10^{-1}	10^{-3}	0.32×10^{-1}	10^{-1}
4	0.13×10^{-1}	10^{-1}	0.13×10^{-1}	10^{-2}	0.64×10^{-1}	10^{-1}
5	0.63×10^{-2}	10^{-2}	0.63×10^{-2}	10^{-2}	0.13×10^{0}	10^{0}

TABLE D.2: *Experimental upper bounds ($\Pi_{\max}$) for Π and $1/\varepsilon$ and lower bound for Π ($\Pi_{\min}$), as a function of iteration convergence tolerance. The theoretical bounds Π upper bound and Π lower bound also shown. Both penalty and scaled direct control method with block matrix implementation as in Algorithm 2 are used in upper bound experiment and produce the same results for $\Pi_{\max}$. Scaled direct control method is used to compute $\Pi_{\min}$. Contract parameters are in Table 7.1 and Jump diffusion parameters are in Table 7.2. $\sigma = 0.3, \eta = 0.045452043$. Finest grids are around node $(W, A) = (100, 100)$. These nodes are used to compute the theoretical bounds.*

[5] O. Bokanowski, S. Maroso, and H. Zidani. Some convergence results for Howard's algorithm. *SIAM Journal on Numerical Analysis*, 47:3001–3026, 2009.

[6] A. Budhriaja and K. Ross. Convergent numerical scheme for singular stochastic control with state constraints in a portfolio selection problem. *SIAM Journal on Control and Optimization*, 45:2169–2206, 2007.

[7] Z. Chen and P. A. Forsyth. A numerical scheme for the impulse control formulation for pricing variable annuities with a guaranteed minimum withdrawal benefit (GMWB). *Numerische Mathematik*, 109:535–569, 2008.

[8] Z. Chen, K. Vetzal, and P.A. Forsyth. The effect of modelling parameters on the value of GMWB guarantees. *Insurance: Mathematics and Economics*, 43:165–173, 2008.

[9] R. Cont and P. Tankov. *Financial Modelling with Jump Processes*. Chapman and Hall, 2004.

[10] M. Dai, Y. K. Kwok, and J. Zong. Guaranteed minimum withdrawal benefit in variable annuities. *Mathematical Finance*, 18:595–611, 2008.

[11] Y. d'Halluin, P.A. Forsyth, and K.R. Vetzal. Robust numerical methods for contingent claims under jump diffusion processes. *IMA Journal of Numerical Analysis*, 25:87–112, 2005.

[12] P. A. Forsyth and G. Labahn. Numerical methods for controlled Hamilton-Jacobi-Bellman PDEs in finance. *Journal of Compuational Finance*, 11 (Winter):1–44, 2008.

[13] A. Hindy, C. Huang, and S. Zhu. Numerical analysis of a free boundary singular control problem in financial economics. *Journal of Economic Dynamics and Control*, 21:297–327, 1997.

[14] Y. Huang. *Numerical Methods for Pricing a Guaranteed Minimum Withdrawal Benefit (GMWB) as a Singular Control Problem*. PhD thesis, School of Computer Science, University of Waterloo, 2011.

[15] Y. Huang and P.A. Forsyth. Analysis of a penalty method for pricing a guaranteed minimum withdrawal benefit (GMWB). Working paper, University of Waterloo, to appear in the IMA Journal of Numerical Analysis, 2010.

[16] Y. Huang, P.A. Forsyth, and G. Labahn. Combined fixed point policy iteration for HJB equations in finance. Working paper, University of Waterloo, submitted to SIAM journal on Numerical Analysis, 2010.

[17] J.S. Kennedy, P.A. Forsyth, and K.R. Vetzal. Dynamic hedging under jump diffusion with transaction costs. *Operations Research*, 57:541–559, 2009.

[18] S. Kumar and K. Mithiraman. A numerical method for solving singular stochastic control problems. *Operations Research*, 52:563–582, 2004.

[19] H.J. Kushner and P.G Dupuis. *Numerical Methods for Stochastic Control Problems in Continupus Time*. Springer-Verlag, New York, 1991.

[20] R.C. Merton. Option pricing when underlying stock returns are discontinuous. *Journal of Financial Economics*, 3:125–144, 1976.

[21] M. A. Milevsky and T. S. Salisbury. Financial valuation of guaranteed minimum withdrawal benefits. *Insurance: Mathematics and Economics*, 38:21–38, 2006.

[22] H. Pham. On some recent aspects of stochastic control and their applications. *Probability Surveys*, 2:506–549, 2005.

[23] D.M. Pooley, P.A. Forsyth, and K.R. Vetzal. Numerical convergence properties of option pricing PDEs with uncertain volatility. *IMA Journal of Numerical Analysis*, 23:241–267, 2003.

[24] Munos R. Error bounds for approximate policy iteration. pages 560–567, 2003. Proceedings of the 20th International Congress on Machine Learning, Washington.

[25] J. Stoer and R. Bulirsch. *Introduction to Numerical Analysis*. Springer-Verlag, second edition, 1993.

[26] A. Tourin and T. Zariphopoulou. Viscosity solutions and numerical schemes for investment/consumption models with transaction costs. in *Numerical Methods in Finance*, editors: L.C.G. Rogers and D. Talay, Cambridge University Press, 1997.

[27] R. Varga. *Matrix Iterative Analysis*. Prentice Hall, 1961.

[28] I.R. Wang, J.W.I. Wan, and P.A. Forsyth. Robust numerical valuation of European and American options under the CGMY proces. *Journal of Computational Finance*, 10:4(Summer):86–115, 2007.

[29] J. Wang and P.A. Forsyth. Maximal use of central differencing for Hamilton-Jacobi-Bellman PDEs in finance. *SIAM Journal on Numerical Analysis*, 46:1580–1601, 2008.