ERASMUS UNIVERSITY ROTTERDAM

Erasmus School of Economics

Master Thesis Quantitative Finance

Poisson Optional Stopping Times as a Robust Numerical Method for Valuing Path-Dependant American Options with a Jump-Diffusion Process

Author
Jean-Claude Hessing
413975

Supervisor
dr. Rutger-Jan Lange
Second assessor
prof. dr. Michel van der wel

February 24, 2021

Abstract

We use the Poisson Optional Stopping Times (POST) algorithm to value American options and compute early exercise boundaries. We examine extensions to the Black-Scholes model such as Kou's jump-diffusion model and the Heston stochastic volatility model and analyse how different jump parameters influence the exercise boundaries. We find that POST accurately values American options following a jump-diffusion process and that average jump size and skewness are the most important factors influencing the exercise boundary. Extending POST to stochastic volatility with jumps and American-type Asian options also proved possible, although less accurate due to the added dimension.

Acknowledgements

This paper would not have been possible without the cheerfull guidance of my supervisor, Rutger-Jan Lange. I would like to thank him for the opportunity to work on his algorithm and the many hours of interesting discussion and feedback.

Contents

1 Introduction

In their seminal paper, Black and Scholes, 1973 laid down the fundamentals of option pricing. Their contingent claims analysis of a vanilla (put/call) option on an asset following a geometric brownian motion (GBM) led to a partial differential equation (PDE) with an elegant analytical solution. Since then, a growing body of literature is devoted to extending their techniques to price more complicated options following more complicated stochastic processes. Minor changes in the option pay-off (such as the possibility of early exercise) or the stochastic process of the asset price (such as time-varying volatility) quickly diminish the availability of analytical solutions. Numerical methods face a trade-off between versatility and speed. The universally applicable Monte-Carlo Simulation comes with a huge computational burden, especially for path-dependant options and faster specialized tools for specific option types require arduous derivations. A new numerical method developed in Lange et al., 2020 presents a middle ground: Poisson Optional Stopping Times (POST). Originally proposed to solve real option valuation problems, the method combines traditional finite difference discretization with a stochastic solution to the free-boundary problem that arises when pricing options with early exercise conditions. They solve the problem by allowing the option to only be exercised at random, Poisson distributed, stopping times. In financial terms this can be seen as imposing a liquidity constraint. This provides a computationally simple and graspable numerical solution to the difficult differential equations arising from expanded Black-Scholes models. This paper demonstrates that POST can be used to compute exercise boundaries for a variety of option types with minimal technical derivations, specifically options with characteristics that make them difficult to value. The applications increase in complexity, starting with American options following a jump-diffusion (JD) process, after which American options following a Stochastic Volatility (SV) process and American-type Asian options following a JD process are examined. We demonstrate how the algorithm can be used to quickly compare jump distributions and model specifications, with few technical derivations.

Extensions to the Black-Scholes model are necessary because the empirical applications of the Black-Scholes model show two major inconsistencies with the its theoretical assumptions. The first is that the Black-Scholes model assumes constant variance, yet the volatility implied from market prices of derivatives often shows a slight curve (the volatility 'smile' or 'smirk') over time to maturity. The second inconsistency is that the assymetric leptokurtic features of asset returns are inconsistent with the assumption that asset prices follow a geometric Brownian motion. To explain the assymetric leptokurtic features a number of changes to the BS model have been proposed. Kou, 2002 provides an extensive list, which is summarized as fol-

lows: The first type of models are fractional Brownian motion (Rogers, 1997), Chaos theory (Mandelbrot, 1963) and stable processes (Samorodnitsky and Taqqu, 1994). The second type concern generalized hyperbolic models (Barndorff-Nielsen and Shephard, 2001, Blattberg and Gonedes, 1974). The third type are time-changing Brownian motions (Clark, 1973, Madan and Seneta, 1990, Heyde, 1999). To explain the the volatility smile five different types of models have been proposed: 1) Stochastic volatility and ARCH models (Hull and White, 1987, Engle, 1995 and Fouque et al., 2000), 2) constant elasticity of variance models (Cox and Ross, 1976, Davydov and Linetsky, 2001), 3) the original normal jump model by Merton, 1976, 4) affine stochastic volatility and affine jump-diffusion models (Heston, 1993, Duffie et al., 1998) and 5) Lévy processes (Geman et al., 2001). The majority of the presented models provide analytic solutions for European options, whereas the numerical methods for path-dependent options are often much slower, for a survey see Boyle et al., 1997.

Evidence that asset prices in stock markets and foreign exchange markets contain jumps is presented in Jarrow and Rosenfeld, 1984, Ball and Torous, 1985, Jorion, 1988 and Bates, 1996. While this provides a valid argument for the attention jump-diffusion models have received over the past years, a comprehensive model proposed by Kou, 2002 sparked popularity in the field. Merton, 1976 and Cox and Ross, 1976 started the field by deriving closed form solutions for jump-diffusion and pure jump processes respectively. Modern literature on jump-diffusion process mainly builds upon the double exponential jump-diffusion model proposed by Kou, 2002, which assumes the price process is a Brownian motion with a double exponential jump component. An advantage of this model is that it produces ready to use expressions for path-dependent options such as lookback, barrier, Asian and occupation-time-related options (see Kou and Wang, 2004 and Cai et al., 2010). As these expression contain some approximations and assumptions, they are called semi-analytic. A host of similar models have also been proposed, which include the phase-type diffusion model (PHM) and the hyperexponential jump-diffusion model (HEM), see Cai and Kou, 2011 for an overview.

The remainder of this article is structured as follows: Section 2 gives a brief overview of Kou's Jump-Diffusion model, section 3 provides a brief summary of the POST algorithm, section 4 discusses how POST can be used to compute American put prices for Kou's model, gives a numerical validation and examines the effect of jumps on the exercise boundary, section 5 extends POST for American options following stochastic volatility, section 6 extends POST for American-type Asian options and section 7 summarises our findings.

2 Kou's Jump-Diffusion Model for Option Pricing

The asset price dynamic introduced in Kou, 2002 consists of a Brownian motion and a jump component. Under the physical measure $\mathbb{P}$ this is defined as

$$\frac{\mathrm{d}S\left(t\right)}{S(t-)} = \mu \mathrm{d}t + \sigma \mathrm{d}W\left(t\right) + \mathrm{d}\left[\sum_{i=1}^{N(t)} (V_i - 1)\right], \tag{1}$$

where $\mu \mathrm{d}t + \sigma \mathrm{d}W(t)$ is the increment of a Brownian motion with drift μ, volatility σ and $W(t)$ is a standard Wiener process. The jump component $\sum_{i=1}^{N(t)} (V_i - 1)$ can be interpreted as a sequence of independent identically distributed (i.i.d.) jumps. The number of jumps is determined by Poisson process $N(t)$ with rate λ_J, where J denotes it relates to the jump rate and not to the POST rate defined later, and the magnitude of a jump is determined by i.i.d. random variables V_i. Specifically in Kou, 2002 V_i is chosen such that $Y_i = \log(V_i)$ has an asymmetric double exponential distribution with the density

$$\begin{gathered} f_Y\left(y\right) = p\,\eta_1\, e^{-\eta_1 y} 1_{\{y \geq 0\}} + q\,\eta_2\, e^{\eta_2 y} 1_{\{y<0\}}, \\ \eta_1 > 1, \quad \eta_2 > 0, \end{gathered} \tag{2}$$

where $p, q \geq 0$ represent the probabilities of upward and downward jumps, respectively and $p + q = 1$. Treating upwards and downwards jumps as separate cases provides an intuitive way to define Y as

$$\log(V) = Y \coloneqq \begin{cases} \xi_1 & \text{with probability } p, \\ -\xi_2 & \text{with probability } q, \end{cases} \tag{3}$$

where ξ_k is an exponential random variable with mean $1/\eta_k$. The stochastic elements of the asset price dynamic $(W(t), N(t), Y)$ are all assumed to be independent. In order to get analytical results, Kou, 2002 assumes drift and volatility to be constant. Solving the stochastic differential equation yields the asset price:

$$S\left(t\right) = S\left(0\right) \exp\left\{\left(\mu - \frac{1}{2}\sigma^2\right) t + \sigma W\left(t\right)\right\} \prod_{i=1}^{N(t)} V_i, \tag{4}$$

which displays the stock price as a product of the gains from the Brownian motion and the cumulative gains from all jumps. The expected value of jump size V explains the restrictions on η_1 and η_2: $\mathbb{E}(V) = \mathbb{E}(e^Y) = \frac{p\eta_1}{\eta_1 - 1} + \frac{q\eta_2}{\eta_2 + 1}$, which means $\eta_1 > 1, \eta_2 > 0$ are necessary to ensure $\mathbb{E}(V) < \infty$ and subsequently $\mathbb{E}(S) < \infty$. In practice this means the average jump cannot exceed 100%, which is a reasonable assumption to make.

The jumps in the asset price process have implications for the deriving the risk-neutral probability measure. Due to the jumps it is impossible to use the risk-free hedging argument,

which means the risk-neutral measure is not unique. Kou, 2002 shows that by assuming rational expectations and HARA type utility it is possible to derive a probability measure $\mathbb{P}$ for which the equilibrium price of an option is equal to the expectation of the discounted option pay-off under said probability measure. The asset price process retains its double exponential jump-diffusion form under this measure. The asterisks denote the value of the parameters under the risk-neutral measure (see Kou, 2002 for the exact derivation of these parameters), in which case equation 1 becomes:

$$\frac{\mathrm{d}S\left(t\right)}{S(t-)} = (r - \lambda^*\zeta^*)\mathrm{d}t + \sigma \mathrm{d}W^*\left(t\right) + \mathrm{d}\left[\sum_{i=1}^{N^*(t)} Y_i\right]. \tag{5}$$

The return process $X(t) = \log(S(t)/S(0))$ under $\mathbb{P}$ has the following form:

$$X(t) = (r - \frac{1}{2}\sigma^2 - \lambda^*\zeta^*)t + \sigma W^*(t) + \sum_{i=1}^{N^*(t)} Y_i^*, \quad X(0) = 0, \tag{6}$$

where $W^*(t)$ is a standard Brownian motion under $\mathbb{P}$, $N^*(t)$ is a Poisson process with intensity λ_J^* and log jump sizes Y^* are i.i.d. random variables with a double exponential distribution. The density for the double exponential distribution under $\mathbb{P}$ is $f_{Y^*}(y) \sim p^*\eta_1^* e^{-\eta_1^* y}\mathbf{1}_{y\geq 0} + q^*\eta_2^* e^{\eta_2^* y}\mathbf{1}_{y<0}$. The term $\lambda_J^*\zeta^*$ corrects the drift for the extra variance introduced by the jump process, to ensure the process is risk-neutral under $\mathbb{P}$. It can be interpreted as the expected change over time due to the jumps, as λ_J^* determines the jump intensity and ζ^* is defined as the expected value of the change in $S(t)$ due to a jump:

$$\zeta^* \coloneqq \mathbb{E}^*[V^*] - 1 = \frac{p^*\eta_1^*}{\eta_1^* - 1} + \frac{q^*\eta_2^*}{\eta_2^* + 1} - 1. \tag{7}$$

Another important characteristic of the process which is necessary for option valuation is the infinitesimal generator of $X(t)$:

$$(\mathcal{L}V)(x) \coloneqq \frac{1}{2}\sigma^2 V''(x) + (r - \frac{1}{2}\sigma^2 - \lambda^*\zeta^*)V'(x) + \lambda^* \int_{-\infty}^{\infty} [V(x+y) - V(x)] f_Y(y)\mathrm{d}y. \tag{8}$$

Due to the jump term this takes the form of a partial integro-differential equation (PIDE). From now on we will drop the asterisks in the notations, all parameters will refer to the risk-neutral value.

3 Poisson Optional Stopping Times

POST was originally proposed in Lange et al., 2020 to solve real option problems. Such options usually encompass a problem in which two or more stochastic processes (e.g. costs and revenues) determine the future cash flows of a project. The owner of the project has the option to stop

receiving the cash flow, receiving a one-time stopping gain in exchange. Take for example owning a company; future cash flow $f(X, Y)$ is a simple function of costs X and revenues Y. The stopping gain $g(X, Y)$ can be seen as selling the company for a lump sum, which depends on the value of X and Y at the time of sale. This presents obvious parallels with option pricing. Instead of a sale there is a pay-off at maturity, which depends on the stochastic asset price.

Assume the option can not be exercised continuously, only at certain moments generated by an exogenous Poisson process with rate $\lambda < \infty$ (not to be confused with λ_J from the jump process). Then denote the intrinsic value of the project $V_\lambda(X, Y)$. This value should reflect the expected future cash flows as well as the option to receive the stopping gain. Receiving this stopping gain is only possible at a generated stopping time. The intensity of the arrivals thus affects the intrinsic value, which is reflected in subscript λ. We are interested in finding the optimal exercise policy of the real option. To this end the state space is divided in two regions. In the continuation region it is optimal to continue receiving the cash flow. The intrinsic value of holding on to the option should exceed the stopping gain in this situation, $V_\lambda(X, Y) > g(X, Y)$ for each (X, Y) in the continuation region. In the stopping region the stopping gains should exceed the value of holding on to the option, $V_\lambda(X, Y) \leq g(X, Y)$. The equation that should hold for each possible value of $V(X, Y)$ is derived from contingent-claims analysis:

$$rV_\lambda(X, Y) = \mathrm{L}V_\lambda(X, Y) + f(X, Y) + \lambda[g(X, Y) - V_\lambda(X, Y)]^+, \tag{9}$$

for each $(X, Y) \in \mathbb{R}$. Here $r > 0$ is the risk-free rate, L is the infinitesimal generator of the process (X_t, Y_t) regarding small changes in t, hence the subscript, and $[\,\cdot\,]^+ = \max(0,\ \cdot\ )$. The equation can intuitively be understood as follows: the discounted return $rV_\lambda(X, Y)$ is equal to expected change in $V_\lambda(X, Y)$ attributable to a change in (X, Y), as measured by $\mathrm{L}V_\lambda(X, Y)$, plus the cash flow $f(X, Y)$, plus the additional gain from exercising the option, which is active when the project is in the stopping region. POST arrival rate λ reflects that opportunities to exercise arrive at Poisson rate λ. A detailed explanation of this equation can be found in Lange et al., 2020, in Appendix A, who explain the connection of the term $\lambda[g(X, Y) - V_\lambda(X, Y)]^+$ with penalty methods. For our application it is only necessary to know that it solves the free-boundary problem encountered in liquidity-constrained stopping problems.

For standard diffusion processes, L takes the form of a partial differential equation (PDE). If we take (X_t, Y_t) to be a 2-dimensional Brownian motion L is given by:

$$\mathrm{L} = \mu_x \frac{\mathrm{d}}{\mathrm{d}X} + \mu_y \frac{\mathrm{d}}{\mathrm{d}Y} + \frac{\sigma_x^2}{2}\frac{\mathrm{d}^2}{\mathrm{d}X^2} + \frac{\sigma_y^2}{2}\frac{\mathrm{d}^2}{\mathrm{d}Y^2} + \rho\sigma_x\sigma_y\frac{\mathrm{d}^2}{\mathrm{d}X\mathrm{d}Y}, \tag{10}$$

where μ_x and μ_y are the risk-neutral drifts, σ_x^2 and σ_y^2 are the variance coefficients and ρ measures the correlation between X_t and Y_t. Solving equation 9 then amounts to solving a

PDE.

L measures the instantaneous change of $V_\lambda(X,Y)$ due to a change in (X,Y). It does so indefinitely, which means we assume an infinite horizon for this project. To tackle projects with a finite horizon we can define one state variable to represent time and truncate the state space to the interval we are interested in. Our approach remains virtually the same, as we simply set the drift to 1 and variance to 0, to obtain a deterministic process. When we set $X_t = t$ and Y_t a Brownian motion with drift, L becomes

$$\mathrm{L} = \frac{\mathrm{d}}{\mathrm{d}X} + \mu_y \frac{\mathrm{d}}{\mathrm{d}Y} + \frac{\sigma_Y^2}{2}\frac{\mathrm{d}^2}{\mathrm{d}Y^2}. \tag{11}$$

Here X_t is a deterministic process with drift 1. To find the optimal value $V_\lambda(X,Y)$ rewrite equation 9 to a form that allows us to introduce the algorithm:

$$(r - \lambda - \mathrm{L})V_\lambda(X,Y) = f(X,Y) + \lambda \max\{g(X,Y), V_\lambda(X,Y)\}. \tag{12}$$

The derivation of this equation can be found in Lange et al., 2020. It introduces $V_\lambda(X,Y)$ on the left-hand side as a fixed point that can be found by the following sequence:

$$(r + \lambda - \mathrm{L})V_\lambda^{(1)}(X,Y) = f(X,Y) + \lambda g(X,Y) \tag{13}$$

$$(r + \lambda - \mathrm{L})V_\lambda^{(j)}(X,Y) = f(X,Y) + \lambda \max\{g(X,Y), V_\lambda^{(j-1)}(X,Y)\},\ j = 1,2,...,J \tag{14}$$

where J denotes the total number of iterations. In each iteration $(r + \lambda - \mathrm{L})$, $f(X,Y)$, $g(X,Y)$ and $V_\lambda^{(j-1)}(X,Y)$ are known. We obtain the new value by solving for $V_\lambda^{(j)}(X,Y)$. The random arrival of stopping times provides an intuitive explanation for the algorithm. For the first iteration, the owner of the option is forced to stop at the arrival of the first stopping time. Every iteration after that adds a Poisson generated opportunity, whereby sub-optimal value $V_\lambda^{(j-1)}(X,Y)$ takes into account that owner of the project has already had $j-1$ opportunities to exercise the option. If it is not optimal to stop at one of the $j-1$ previous opportunities, adding a j^{th} stopping time increases $V_\lambda(X,Y)$. If it is optimal to stop before stopping time j, an additional opportunity will neither increase nor decrease $V_\lambda(X,Y)$. Following this logic it should hold that $V_\lambda^{(j)}(X,Y)$ is weakly monotonically increasing. Convergence of the algorithm is then guaranteed when $V_\lambda^{(j)}(X,Y)$ is bounded from above. Lange et al., 2020 prove this indeed holds and that $V_\lambda^{(j)}(X,Y)$ converges to $V_\lambda(X,Y)$ at geometric rate $\lambda/(r+\lambda)$. The POST arrival rate λ affects the algorithm in two ways. On the one hand a larger λ brings us closer to real world applications with continuous exercise opportunities, while on the other hand a higher λ means slower convergence.

The algorithm operates in function space. To actually use it we need to discretize (X,Y) to a finite, bounded subspace of $\mathbb{R}^2$. We choose this subspace to contain M grid point. To

illustrate this: When we choose a square grid with N_y different values of Y and N_x the number of values for X, we get $M = N_x \cdot N_y$ grid points. For computational purposes we then put the different combinations of X and Y for each grid point in two $M \times 1$ vectors $\mathbf{M}_x$ and $\mathbf{M}_y$, together containing the different combinations of X and Y for all grid points. We then use these vectors to compute the values of $g(X, Y)$ and $f(X, Y)$ for each grid point, which we call $\mathbf{f}$ and $\mathbf{g}$ respectively. The discretized version of $\mathbf{L}$ becomes an $M \times M$ matrix. See appendix B for details on constructing the grid and $\mathbf{L}$ matrix. The values for $V_\lambda^{(j)}(X, Y)$ will be stored in the $M \times 1$ vector $\mathbf{V}_\lambda^{(j)}$. The discretized version of the algorithm then becomes:

$$[(r + \lambda)\mathbf{I} - \mathbf{L}]\mathbf{V}_\lambda^{(1)} = \mathbf{f} + \lambda\mathbf{g} \tag{15}$$

$$[(r + \lambda)\mathbf{I} - \mathbf{L}]\mathbf{V}_\lambda^{(j)} = \mathbf{f} + \lambda \max\{\mathbf{g}, \mathbf{V}_\lambda^{(j-1)}\}, \quad j = 2, 3, ..., J. \tag{16}$$

Here, $\mathbf{I}$ is an $M \times M$ identity matrix and $\max\{\cdot, \cdot\}$ is applied element-wise. Executing an iteration of the algorithm comes down to solving a system of M linear equations. When $\mathbf{L}$ is sparse, standard linear solvers suffice for fast computations. Convergence of the algorithm is guaranteed when $\mathbf{L}$ is weakly diagonally dominant, and also monotone if L further contains nonpositive diagonal entries and nonnegative off-diagonal elements.

4 Applying POST to American jump-diffusion options

In this section we explain how to value an American put option when the underlying asset follows a double exponential jump-diffusion process. We will explain in detail what steps are necessary to derive the algorithm for Kou's (2002) model.

An American put with strike price K and maturity T has a pay-off at time t and stock price S_t of $[K - S_t]^+$, which can be exercised at any given time up until T. Other than the constant K, the variables that determine the pay-off of an American put are the asset price and time to maturity. For our state variables we choose $X_t = t$ to represent time and $Y_t = \log(S_t/S_0)$ to represent the log asset return. This transformation is equivalent to working with the asset price but allows us to separate the jump component in the infinitesimal generator from the diffusion as in log space these are additive instead of multiplicative. The pay-off (or one-time stopping gain) function for this state space becomes $g(t, Y_t) = [K - S_0 e^{Y_t}]^+$ for $t \leq T$. We truncate the state space at $t = T$ and impose a terminal boundary condition $g(T, Y_T) = [K - S_0 e^{Y_T}]^+$ to replicate that the option expires after T (see appendix D). As there is no cash-flow coming from an American option, all values of $f(t, Y_t)$ are set to 0.

4.1 Deriving L

The asset price follows Kou's double exponential jump-diffusion process. Recall equation 8 that specifies the infinitesimal generator for a jump-diffusion process:

$$(\mathcal{L}V)(x) \coloneqq \frac{1}{2}\sigma^2 V''(x) + (r - \frac{1}{2}\sigma^2 - \lambda_J\zeta)V'(x) + \lambda_J \int_{-\infty}^{\infty} [V(x+y) - V(x)] f_Y(y)\mathrm{d}y.$$

This infinitesimal generator can intuitively be understood as a second order Taylor expansion that considers changes in $V(x)$ due to the expected change in process X_t over some small $\mathrm{d}t$. The three terms have clear interpretations: The term containing $V''(x)$ captures changes due to the variance of X_t, the term with $V'(x)$ captures the drift and the integral term captures the expected change in $V(x)$ due to the occurrence of jumps. Discretizing the jump term will create a dense matrix, which severely impacts computation times. To alleviate this computational burden we employ an operator-splitting method, (Feng and Linetsky, 2008). This divides L in an implicit ($\mathrm{L_{imp}}$) and explicit part ($\mathrm{L_{ex}}$), where the discretized jump term is handled explicitly and the diffusion terms are handled implicitly.This changes equation 14 into (taking into account the change from (X, Y) to (t, Y)

$$(r + \lambda - \mathrm{L_{imp}})V_\lambda^{(j)}(t, Y) = f(t, Y) + \mathrm{L_{ex}}V^{(j-1)} + \lambda \max\{g(t, Y), V_\lambda^{(j-1)}(t, Y)\}. \tag{17}$$

Discretizing the diffusion term using a central difference scheme and the drift term $V'(x)$ yields the full discretization of $\mathcal{L}$

$$\begin{aligned}(\mathcal{L}V)(t, y) \approx &\left(-\frac{1}{\mathrm{d}y^2}\sigma^2 - \frac{1}{\mathrm{d}y}(r - \frac{1}{2}\sigma^2 - \lambda_J\zeta) - \frac{1}{\mathrm{d}t}\right) V(t, y) \\ + &\left(\frac{1}{2\mathrm{d}y^2}\sigma^2 + \frac{1}{\mathrm{d}y}(r - \frac{1}{2}\sigma^2 - \lambda_J\zeta)\right) V(t, y + \mathrm{d}y) + \frac{1}{2\mathrm{d}y^2}\sigma^2 V(t, y - \mathrm{d}y) + \frac{1}{\mathrm{d}t}V(t + \mathrm{d}t, y). \end{aligned} \tag{18}$$

Here $\mathrm{d}t$ and $\mathrm{d}y$ are assumed constant and, in slight abuse of notation denote our grid spacing. A non-constant grid can easily be implemented, see appendix F.

The discretization finds it way into the algorithm through $\mathbf{L}_{\mathrm{imp}}\mathbf{V}_\lambda^{(j)}$. To capture above equations in this format we construct $\mathbf{L}_{\mathrm{imp}}$ such that every row contains the right constants corresponding to their place in $\mathbf{V}_\lambda^{(j)}$. This is visualised in the stencil below, where the central cell shows what value the element of $\mathbf{L}_{\mathrm{imp}}$ corresponding to $V(t, y)$ should be. The cells above and under show the values corresponding to $V(t, y + \mathrm{d}y)$ and $V(t, y - \mathrm{d}y)$, with left and right corresponding to $V(t - \mathrm{d}t, y)$ and $V(t + \mathrm{d}t, y)$.

Recall the stability condition on $\mathbf{L}$ requiring nonpositive elements on the diagonal. In equation the stencil above it is guaranteed when the drift $(r - \frac{1}{2}\sigma^2 - \lambda_J\zeta)$ is positive. If the drift happens to be negative, backward difference can be used instead. An all-encompassing method to code this into a stencil uses the maximum and absolute value operators, the notation for

	$\frac{1}{2\mathrm{d}y^2}\sigma^2 + \frac{1}{\mathrm{d}y}(r - \frac{1}{2}\sigma^2 - \lambda_J\zeta)$	
	$-\frac{1}{\mathrm{d}y^2}\sigma^2 - \frac{1}{\mathrm{d}y}(r - \frac{1}{2}\sigma^2 - \lambda_J\zeta) - \frac{1}{\mathrm{d}t}$	$\frac{1}{\mathrm{d}t}$
	$\frac{1}{2\mathrm{d}y^2}\sigma^2$	

this can be found in appendix C. Also note that above stencil assumes no correlation between processes, this can be added in the corner elements of the stencil if necessary, which apply to $V(t + \mathrm{d}t, y + \mathrm{d}y)$ and the other combinations.

The explicit part of L contains the term $\lambda_J \int_{\mathbb{R}}[V(y+z) - V(y)]f_z(z)dz$, which is discretized using a standard Riemann sum. Observe that jumps happen instantly and over Y_t, such that notation containing t can be dropped. As we have the same set of Y's for every t, computing the discretization once suffices. Rewriting the equation slightly yields the following discretization, which we computes the value of the integral at y_i as a sum of functions of all grid points y_j (see appendix G for derivation):

$$\int_{\mathbb{R}}[V(y_i + z) - V(y_i)]f_z(z)\mathrm{d}z \approx \sum_{j=1}^{N_y}[F_z(y_{j+1} - y_i) - F_z(y_j - y_i)]V(y_j) - V(y_i). \tag{19}$$

Here $F_z(z)$ is the cumulative density function (CDF) of the double exponential distribution on $\mathbb{R}$. The integral by definition measures the expected change in $V(t, y)$ when a jump occurs at $y = y_i$. This is reflected in the discretization, where the sum on the right hand side can be seen as a discrete expected value: the probability that a jump from y_i to somewhere between y_j and y_{j-1} times the value of the option at y_j. $\mathbf{L}_{\text{ex}}$ can then be constructed in a similar manner to the implicit discretization. We fill $\mathbf{L}_{\text{ex}}$ such that each row enforces equation 19 for its corresponding y_i.

The jump distribution spans $\mathbb{R}$, but our finite grid does not. This becomes especially problematic for values of y at the edge of the grid, where a large part of the integral falls outside of the grid. In this case we enforce that jumps can only happen to existing grid points and any jump outside the grid will end up at the edge of the grid instead. The boundary conditions we enforce for the implicit part of the algorithm depend on whether put or call options are valued. For call options we enforce Dirichlet boundary conditions (assume value is 0 outside grid) at $y_{\min}$ and extrapolate the value at $y_{\max}$. For put options these are reversed, see also appendix E. We have now derived all the necessary steps to apply the algorithm to an American put following a double exponential jump-diffusion process.

4.2 Numerical validation

As a base scenario we value an American put option with strike $K = 100$, maturity $T = 0.25$ and asset price at $t = 0$, $S_0 = 100$. The parameters for the base scenario are $r = 0.05$, $\sigma = 0.2$, $p = 0.6$, $\eta_1 = 25$ and $\eta_2 = 25$. The grid consists of $N_x = 250$ equidistant gridpoints and $N_y = 1100$ gridpoints centered in density around the strike. We vary the parameters one at a time and compare the resulting valuation and exercise boundary at $t = 0, S_0 = 100$ with Kou's approximation (details can be found in appendix J). The exercise boundary for Kou's approximation is part of the output and the exercise boundary can be found with POST by checking for which S_0 the exercise value becomes larger than the intrinsic value. In mathematical terms finding the largest y_i for which $[K - S_0]^+ > V_\lambda^{(J)}(0, S_0)$ holds.

Table 1: American Put value for Jump-Diffusion model with strike $K = 100$

Parameters					Value at $S_0 = 100$				Boundary at $S_0 = 100$			
σ	λ	P	η_1	η_2	POST	Kou	Δ	Δ%	POST	Kou	Δ	Δ%
0.01	3	0.6	25	25	0.906	0.949	-0.043	-4.48	97.43	97.19	0.25	0.25
0.2	3	0.6	25	25	3.878	3.871	0.006	0.17	85.30	85.87	-0.57	-0.66
0.5	3	0.6	25	25	9.567	9.545	0.021	0.22	62.50	65.06	-2.56	-3.93
0.7	3	0.6	25	25	13.439	13.410	0.028	0.21	50.16	53.11	-2.95	-5.55
0.2	3	0.6	25	25	3.878	3.871	0.006	0.17	85.30	85.87	-0.57	-0.66
0.2	7	0.6	25	25	4.386	4.368	0.018	0.41	82.94	83.77	-0.83	-0.99
0.2	3	0.1	25	25	3.870	3.884	-0.014	-0.37	85.98	86.30	-0.32	-0.37
0.2	3	0.3	25	25	3.862	3.878	-0.016	-0.40	85.81	86.13	-0.32	-0.37
0.2	3	0.5	25	25	3.882	3.873	0.009	0.23	85.38	85.96	-0.57	-0.67
0.2	3	0.6	25	25	3.878	3.871	0.006	0.17	85.30	85.87	-0.57	-0.66
0.2	3	0.9	25	25	3.874	3.868	0.006	0.15	84.96	85.59	-0.63	-0.74
0.2	3	0.6	5	25	7.627	7.612	0.016	0.21	54.88	57.15	-2.27	-3.97
0.2	3	0.6	15	25	4.206	4.209	-0.003	-0.07	83.03	83.60	-0.57	-0.68
0.2	3	0.6	25	25	3.878	3.871	0.006	0.17	85.30	85.87	-0.57	-0.66
0.2	3	0.6	50	25	3.720	3.705	0.015	0.39	86.16	86.78	-0.63	-0.72

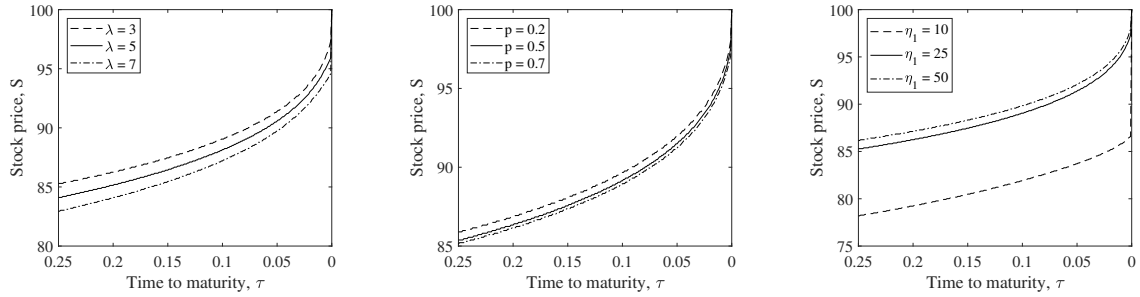
POST closely approximates the value of the option in most scenarios. The difference in value estimations is under 0.5% in most cases, the only exceptions being extreme case $\sigma = 0.01$. The exercise boundary stays within a 1% margin for most scenarios. In situations with a low exercise boundary (< 65), the difference becomes larger. This is mainly due to a coarse grid at these points. Overall the value and exercise boundary computations are satisfactory. Computation times are much longer for the algorithm as opposed to Kou's approximation, but this was to be expected given the difference in methods.

4.3 Examining the exercise boundary

One execution of the algorithm does give us the option price over the entire grid, which is not the case for Kou's method, which only provides the value and boundary at S_0. This enables us to immediately draw the exercise boundary with POST output for the different scenarios. In figure 3 the exercise boundaries for different parameter values are pictured.

Jump intensity λ_J influences the occurrence of jumps, p the direction and η_1 the size of

Figure 1: Exercise boundaries for varying λ_J, p and η_1



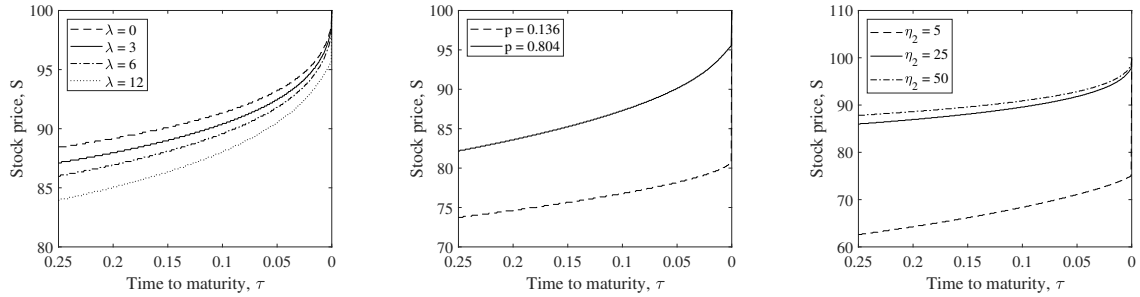
upwards jumps. Note that an upward jump decreases the value of a put option and vice versa. This is reflected in the exercise boundaries. In the base scenario the expectation of upwards and downwards jumps is equal and relatively small ($\eta_1 = \eta_2 = 25$). As such changing p has little influence on the exercise boundary. Increasing the arrival rate of jumps has a more noticeable effect on the exercise boundary, the difference between $\lambda_J = 3$ and $\lambda_J = 7$ being 2.5. We start seeing larger differences when we vary the expected value of jumps. When upward jumps are large ($\eta_1 = 10$) the exercise boundary shifts downward, suggesting that is better to hold onto an option longer if the upward jumps become larger. The difference between the exercise boundaries for $\eta_1 = 10$ and $\eta_1 = 25$ is much larger than the difference between boundaries for $\eta_1 = 25$ and $\eta_1 = 50$. This is likely because at $\eta_1 = 25$ the expected value of jumps is small enough not to have a significant effect on the exercise boundary and at $\eta_1 = 50$ this expected value is even smaller. The pay-off of a put option becomes lower when an upward jump occurs, making it counter intuitive to hold on to an option if the possibility of such a jump exists. In search of a possible explanation we examine the two ways in which the jump process influences the option price. The first is through $\mathrm{L_{ex}}$, which translates the direct effect of the jump process to the asset price. The second is through a correction in the risk-neutral drift of the diffusion process:$(r - \lambda_J\zeta)$. To isolate the effect of the jump itself without changing the diffusion process we keep $\theta = \lambda_J\zeta$ constant, while varying other parameters of the double exponential distribution and examine the effect on the exercise boundary. We test the effect of three characteristics of the jump: the intensity, the skewness and the variance. We can change these characteristics and keep $\theta = \lambda_J(p\eta_1/(\eta_1 - 1) + q\eta_2/(\eta_2 + 1) - 1)$ constant by varying the parameters that affect the characteristic and change the other parameters accordingly. Varying the jump intensity means varying λ_J. Varying the skewness is less straightforward. The constraint on θ means we compare two situations: one with small downward jumps with a high probability (high p and high η_2) and large upwards jumps with a low probability (low q and low η_1), and one with the opposite jump sizes and probabilities. Adjusting the variance comes down to varying η_1 and η_2 simultaneously. Table 4.4 shows the combinations of parameters used. The resulting exercise

boundaries are shown in figure 4.3.

Table 2: Parameter combinations for computing exercise boundaries

Jump intensity				Skewness				Variance			
λ_J	P	η_1	η_2	λ_J	P	η_1	η_2	λ_J	P	η_1	η_2
0	0.5	25	25	6	0.136	5	25	6	0.5	6.94	5
3	0.5	25	25	6	0.804	25	5	6	0.5	26	25
6	0.5	26	25					6	0.5	48.15	50
12	0.5	26.47	25								

Figure 2: Exercise boundaries for varying jump intensity, skewness and variance



The exercise boundaries are indeed affected by the different parameter specifications. An increasing jump intensity leads to a lower exercise boundary. We learn from this that the increasing odds of a downward jump (resulting in a higher option exercise value) carry more weight than the simultaneously increasing odds of an upward jump, making it optimal to hold on to the option longer. By varying the skewness we look at it from a different angle. When there are many small upward jumps and few large downward jumps we exercise early. Intuitively this means that as we expect the asset price to jump up it is not worth waiting for a less likely large downward jump. Changing the variance has a similar effect to changing the intensity. Larger jumps make it optimal to hold on the option longer. From these insights we conclude that it is optimal to hold on to the option when downward jumps occur more frequently and when downward jumps become larger, but when there is a trade-off between frequently occurring downward jumps and large downward jumps it is optimal to hold on when the jumps occur frequently.

4.4 Changing the jump distribution

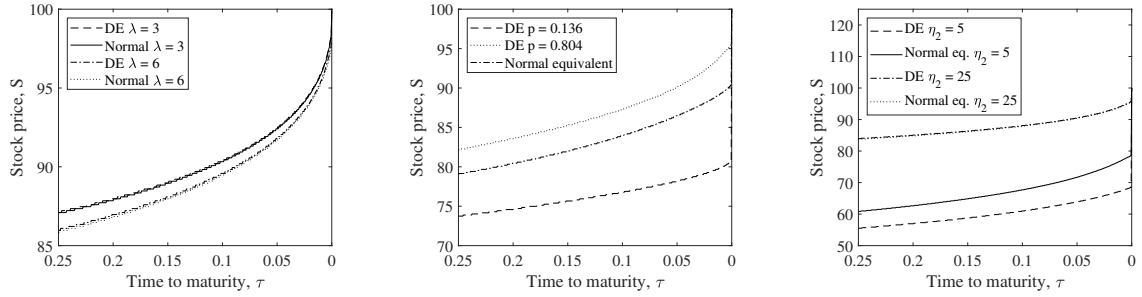
We have seen what happens when we change the parameters of the double exponential distribution, but we have yet to see what happens when we choose a different distribution altogether. POST allows us to do so with minimal effort. Recall the condition for a risk-neutral jump-diffusion process in equation 6, which holds equally for any distribution as long as $\mathbb{E}[V]$ exists, which is necessary to compute $\zeta := \mathbb{E}[V] - 1$. For the normal distribution with parameters

(μ_n, σ_n) this quantity is $\exp(\mu_n + 0.5\sigma_n)$. L_{ex} is constructed with the CDF of the jump distribution. The normal CDF can be plugged into equation 19 without any further adjustments. To compare the exercise boundary of a normal jump-diffusion with a double exponential jump-diffusion, some way to match the jump distributions is needed. We match the parametrisation of the distributions based on $\mathbb{E}[e^z]$ and $\mathbb{E}[z]$, which is sufficient to identify μ_n and σ_n when the parameters of the double exponential distribution are known. We examine similar scenarios as before and adjust jump intensity, skewness and variance while keeping θ constant. The resulting parameters and exercise boundaries are shown below.

Table 3: Parameter combinations for computing exercise boundaries

	Jump intensity				Skewness				Variance			
Double Exponential	λ_J	P	η_1	η_2	λ_J	P	η_1	η_2	λ_J	P	η_1	η_2
	0	0.5	25	25	6	0.136	5	25	6	0.5	6.94	5
	3	0.5	25	25	6	0.804	25	5	6	0.5	26	25
Normal	λ_J		μ	σ	λ_J		μ	σ	λ_J		μ	σ
	3		0	0.057	6		-0.007	0.117	6		-0.028	0.245
	6		-0.001	0.056	6		-0.007	0.135	6		-0.001	0.056

Figure 3: Exercise boundaries comparing normal jumps with double exponential jumps



The exercise boundaries for normal jumps closely follow the DE boundaries when only the jump intensity is changed. In these cases the normal and DE distribution do not differ too much and it does not matter what distribution is chosen when it comes to exercise boundaries. The results do differ when looking at the skewed DE distributions. As expected the non skewed normal exercise boundary lies in between the two DE exercise boundaries. The takeaway is that it is very important to take into account the skew of jumps when deciding what distribution is appropriate to model exercise boundaries with a jump-diffusion process.

5 American options with Stochastic Volatility

Seeing that the algorithm works in the base-case jump-diffusion model, we test the algorithm on a more complex stochastic process. Letting go of jumps for now, we focus on relaxing the

constant volatility requirement. Following Chockalingam and Muthuraman, 2011 we focus on pricing American put options with the Heston model. Here stock price S follows a familiar GBM, but the volatility σ_t is no longer constant over time and follows a Brownian motion with mean-reverting drift. The stochastic process for σ_t is defined as

$$\mathrm{d}Y_t = \kappa(m - Y_t)\mathrm{d}t + \nu\sqrt{Y_t}\mathrm{d}W_t^y, \tag{20}$$

with $\sigma_t = \sqrt{Y_t}$ the volatility of the asset price process, $\kappa > 0$ the mean reversion coefficient, $m > 0$ the long term average volatility, $\nu > 0$ the diffusion of the asset price volatility and W_t^y a standard Brownian Motion that can be correlated with the BM driving the asset price, with correlation coefficient ρ. The American option price now follows a three-dimensional PDE. Extending the POST algorithm to three dimensions is straightforward and does not change any convergence requirements. Instead of a two dimensional equation we now solve its three dimensional equivalent:

$$(r - \lambda - \mathrm{L})V_\lambda(X, Y, Z) = f(X, Y, Z) + \lambda \max\{g(X, Y, Z), V_\lambda(X, Y, Z)\}. \tag{21}$$

We easily see this does not change the functional form of the algorithm. Discretizing $f(X, Y, Z)$ and $g(X, Y, Z)$ is straightforward. We obtain infinitesimal generator L for the Heston model from Chockalingam and Muthuraman, 2011. We take X as the return process, Y as the volatility process and Z as time t:

$$\mathrm{L} = (r - \frac{1}{2}\sigma_t^2)\frac{\mathrm{d}}{\mathrm{d}X} + \frac{1}{2}\sigma_t\frac{\mathrm{d}^2}{\mathrm{d}X^2} + (\kappa(m - Y) - \nu\sigma_t\Lambda)\frac{\mathrm{d}}{\mathrm{d}Y} + \frac{1}{2}\nu^2\sigma_t^2\frac{\mathrm{d}^2}{\mathrm{d}Y^2} + \rho\nu\sigma_t^2\frac{\mathrm{d}^2}{\mathrm{d}X\mathrm{d}Y} + \frac{\mathrm{d}}{\mathrm{d}t}. \tag{22}$$

Discretizing the new Y dimension follows the usual logic, applying forward difference for the drift term and central difference for the diffusion term. The covariance term can be discretized with a simple middle difference scheme as well, resulting in the following stencil (in X and Y dimensions, as t is trivial). Note that the drift is once again assumed to be positive, see appendix C for handling a negative drift.

$-\rho\nu\sigma_t^2\frac{1}{4\mathrm{d}x\mathrm{d}y}$	$(r - \frac{1}{2}\sigma^2)\frac{1}{\mathrm{d}x} + \sigma^2\frac{1}{2\mathrm{d}x^2}$	$\rho\nu\sigma_t^2\frac{1}{4\mathrm{d}x\mathrm{d}y}$
$\frac{1}{2}\nu^2\sigma_t^2\frac{1}{2\mathrm{d}y^2}$	$-(r - \frac{1}{2}\sigma^2)\frac{1}{\mathrm{d}x} - \sigma^2\frac{1}{2\mathrm{d}x^2} - (\kappa(m - y) - \nu\sigma_t\Lambda)\frac{1}{\mathrm{d}y} - \frac{1}{2}\nu^2\sigma_t^2\frac{1}{\mathrm{d}^2\mathrm{y}^2} - \frac{1}{\mathrm{d}t}$	$(\kappa(m - y) - \nu\sigma_t\Lambda)\frac{1}{\mathrm{d}y} + \frac{1}{2}\nu^2\sigma_t^2\frac{1}{\mathrm{d}2y^2}$
$\rho\nu\sigma_t^2\frac{1}{4\mathrm{d}x\mathrm{d}y}$	$\sigma^2\frac{1}{2\mathrm{d}x^2}$	$-\rho\nu\sigma_t^2\frac{1}{4\mathrm{d}x\mathrm{d}y}$

As $V(X, Y, t)$ is not very sensitive to Y, the solution is not very sensitive to boundary conditions on Y. For $Y = 0$, we impose Neumann boundary conditions and for $Y = y_{\max}$ we extrapolate the option value. For a computational example we take $K = 10$, $T = 0.25$,

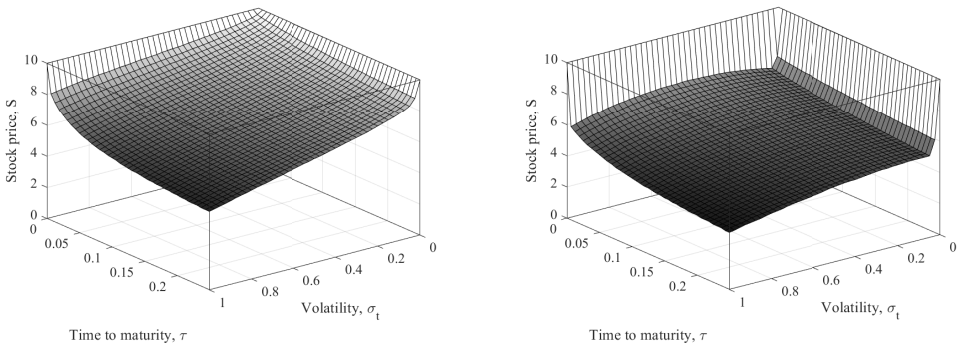
$r = 0.1$, $\kappa = 5$, $m = 0.16$, $\nu = 0.9$ and $\rho = 0.1$. We compute the asset value and the exercise boundary across X and τ and compare them with the true values given in Chockalingam and Muthuraman, 2011. The grid consists of $N_x = 30$, $N_y = 300$, and $N_t = 50$ grid points.

Table 4: American put value for Heston model with strike $K = 10$

	$y = 0.0625$			$y = 0.25$		
X_0	POST	Chock.	Δ	POST	Chock.	Δ
8	2.007	2.000	0.006	2.039	2.078	-0.039
9	1.067	1.108	-0.040	1.260	1.334	-0.073
10	0.483	0.520	-0.038	0.732	0.796	-0.064
11	0.189	0.214	-0.025	0.395	0.449	-0.053
12	0.073	0.082	-0.009	0.211	0.242	-0.031

Overall the approximation is fairly accurate. Due to the increased computational burden of three dimensions the accuracy obtainable within a reasonable time frame is less than what can be achieved with σ constant. The results are accurate enough to be used to compute an exercise boundary, which is shown in figure 4. The exercise boundary acquired from POST closely resembles the boundary presented in Chockalingam and Muthuraman, 2011. As somewhat of a computational exercise we also check what happens to the exercise boundary if we add jumps to the stock price. Without any technical derivation, we compute $\mathbf{L}_{\text{ex}}$ exactly the same as with the constant volatility case, as the jumps do not depend on the volatility of the diffusion process. In $\mathbf{L}_{\text{imp}}$ we only change the drift in the stencil, which then becomes the familiar $r - 0.5\sigma^2 - \lambda_J\zeta$. For the most part this should be technically correct, but we skip out on proving that there is no change necessary in the stencil for possible covariance between jumps and the volatility process. The parameters for the jump process are $p = 0.6$, $\eta_1 = \eta_2 = 5$ and $\lambda_J = 6$.

Figure 4: Exercise boundary for American Put option following the Heston model without and with jumps



Whereas this little exercise proved it is simple to add complex features to our model easily, the shape of the exercise boundary with jumps hints that skipping out on deriving the proper infinitesimal generator did not yield proper results. As expected the exercise boundary lowered significantly, but the somewhat parabolic shape and the sharp angle when volatility nears zero

have no logical explanation.

6 American-type Asian options with jump-diffusion

To test whether POST can handle path-dependent options we apply it to Asian options. The pay-off of an Asian option depends on the arithmetic average of the underlying asset price. We focus on continuously monitored Asian options, for which the average is defined as $A_t = \frac{1}{t}\int_0^t S_\tau \mathrm{d}\tau$. The pay-off then becomes $[A_t - K]^+$ for call options and $[K - A_t]^+$ for put options. From Cai and Kou, 2012 we obtain the infinitesimal generator for Asian options following a jump-diffusion process.

$$\mathrm{L} = \frac{\mathrm{d}}{\mathrm{d}t} + \frac{1}{2}\sigma^2\frac{\mathrm{d}^2}{\mathrm{d}x^2} + (r - \frac{1}{2}\sigma^2 - \lambda_J\zeta)\frac{\mathrm{d}}{\mathrm{d}x} + \frac{1}{t}(S_t - A_t)\frac{\mathrm{d}}{\mathrm{d}A} + \lambda_J\int_{-\infty}^{\infty}[V(x+y) - V(x)]f_y(y)\mathrm{d}y. \quad (23)$$

It has a familiar form. The added term $\frac{1}{t}(A_t - S_t)\frac{\mathrm{d}}{\mathrm{d}A}$ represents the derivative in A direction. Finite difference methods tend to struggle with this expression due to the lack of a diffusion term. When the convergence requirements on the discretized L hold, this should pose no problem. Discretizing the $\mathrm{d}A_t$ term with forward (or backward, depending on the sign of $S_t - A_t$) difference gives us the following stencil. Again the forward difference for the t dimension is excluded as it is trivial.

	$\frac{1}{2dx^2}\sigma^2 + \frac{1}{\mathrm{d}x}(r - \frac{1}{2}\sigma^2 - \lambda\zeta)$	
	$-\frac{1}{\mathrm{d}x^2}\sigma^2 - \frac{1}{\mathrm{d}x}(r - \frac{1}{2}\sigma^2 - \lambda\zeta) - \frac{1}{t}(S_t - A_t)\frac{1}{\mathrm{d}A} - \frac{1}{\mathrm{d}t}$	$\frac{1}{t}(S_t - A_t)\frac{1}{\mathrm{d}A}$
	$\frac{1}{2\mathrm{d}x^2}\sigma^2$	

Adding in the jumps in at a later stage works the same as in the two-dimensional case. A jump in S does not cause an immediate jump in A_t, rather it influences A_t through a jump in $\mathrm{d}A_t$, which is already accounted for through $\mathrm{L}_{\mathrm{imp}}$. Therefore our operator splitting method works in the same way and we can apply our formulation of L_{ex} in the same manner as the two-dimensional case. To calibrate the grid and test convergence we replicate the European-type Asian call prices without a jump component from 7 different scenario's in Cai and Kou, 2012. The POST algorithm needs a small adjustment to do this, as it assumes the option can be exercised at any point in the grid when an opportunity to do so arrives. To replicate a European-style option we set the pay-off to zero ($g(t, x, A_t) = 0$) for every point in time before maturity: $t \neq T$, which is equivalent to not being able to exercise the option until maturity. Note that the POST algorithm then becomes a rather cumbersome finite difference method and other methods handle these simpler problems more efficiently. These computations

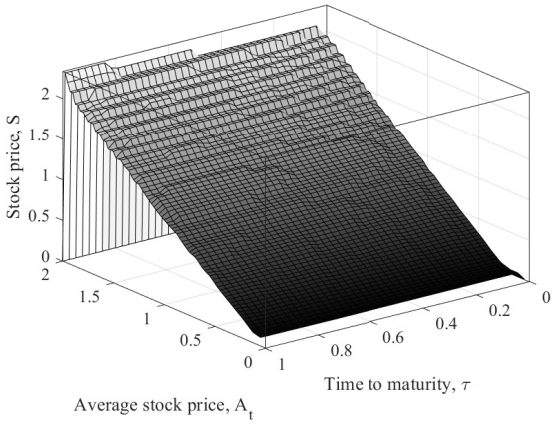
only serve to examine the accuracy of our grid before we compute exercise boundaries for the more complicated American-type Asian options with a jump component. The grid consists of $N_t = 200$, $N_x = 132$ and $N_A = 299$ points.

Table 5: European-style Asian call following a jump-diffusion process

Parameters					Value at $S_0 = 100$			
S_0	K	r	σ	t	POST	Kou	Δ	$\Delta\%$
2.0	2	0.02	0.10	1	0.0571	0.0560	0.0011	1.9
2.0	2	0.18	0.30	1	0.2252	0.2184	0.0068	3.2
2.0	2	0.0125	0.25	2	0.1747	0.1723	0.0024	1.4
1.9	2	0.05	0.50	1	0.1996	0.1932	0.0064	3.3
2.0	2	0.05	0.50	1	0.2501	0.2464	0.0037	1.5
2.1	2	0.05	0.50	1	0.3103	0.3062	0.0041	1.3
2.0	2	0.05	0.50	2	0.3557	0.3501	0.0056	1.6

The grid gives reasonable accuracy, with fast computation times for the European-style options. The accuracy seems not to depend on parameter values, as both situations with less accuracy ($> 3\%$ deviation) have no mutual parameter deviation. The results for European-style options provide enough confidence that we can use the method to compute an early exercise boundary for Asian options with an American-style early exercise feature. To attain the same level of accuracy the grid is kept the same. The only change necessary is to revert $g(t, x, A_t)$ to its original form, such that for every value of t it reflects the payoff for exercising: $g(t, x, A_t) = [A_t - K]^+$.

Figure 5: Exercise boundary for American-type Asian put option with jump-diffusion process



The exercise boundary implies that it is always optimal to exercise if $S_t > A_t$ and $A_t < K$, regardless of τ. If the stock price is higher than the average it is certain that the average will rise and the exercise value will fall. It is therefore always optimal to exercise in that situation.

7 Conclusion

Finding methods or models for option valuation requires facing an inherent dilemma. If the usefulness of a model is measured as its ability to represent reality and the success of a method to solve a model is measured in speed and accuracy, it is often necessary to choose between those successes. For fast and accurate computations an oversimplified model is needed, whereas a lifelike model usually requires intricate and hard to solve additions, hindering the applicability. POST provides a middle ground between between those extremes. By solving a basic option valuation problem we demonstrated that it provides a foundation for analysis, in our case through computing exercise boundaries, and by extending to complicated models we showed its versatility. It can be applied to any underlying asset pricing process, as long as an infinitesimal generator is available or can be deduced. Additions such as a jump process, using whatever jump distribution the researcher sees fit, are also straightforward to implement. We used this to demonstrate the effects of double exponential and normal jump distributions on the exercise boundaries of a number of options. POST handles path-dependent options relatively well compared to equally versatile methods such as simulation. Herein lies the strength of the method. The researcher does not need to adjust his models to accommodate a certain methodology, but can analyse what model comes closest to reality. The flip side of this versatility is that it does not beat methods specifically designed to solve specific models in speed or accuracy. For applications that require very high accuracy the first-order convergence of POST might need too much computation time for satisfactory results. Another noteworthy challenge in the implementation is that it depends on two separate quantities to increase accuracy. For a given grid the accuracy can be increased by increasing λ and the number of POST iterations N_P, after which the overall accuracy can be increased by refining the grid through increasing N_X, N_Y and N_t. A large part of this complexity can be alleviated by using iterative methods to increase both quantities in a systematic way.

References

Ball, C., & Torous, W. (1985). On jumps in common stock prices and their impact on call option pricing. *Journal of Finance*, *40*, 155–173.

Barndorff-Nielsen, O. E., & Shephard, N. (2001). Non-Gaussian Ornstein–Uhlenbeck-based models and some of their uses in financial economics. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *63*(2), 167–241.

Bates, D. (1996). Jumps and stochastic volatility: Exchange rate processes implicit in Deutsche mark options. *The Review of Financial Studies*, *9*, 69–107.

Black, F., & Scholes, M. (1973). The pricing of options and corporate liabilities. *Journal of Political Economy*, *81*, 637–659.

Blattberg, R. C., & Gonedes, N. J. (1974). A comparison of the stable and student distributions as statistical models for stock prices. *The Journal of Business*, *47*(2), 244–280.

Boyle, P., Broadie, M., & Glasserman, P. (1997). Simulation methods for security pricing. *Journal of Economic Dynamics and Control*, *21*(8), 1267–1321.

Cai, N., Chen, N., & Wan, X. (2010). Occupation times of jump-diffusion processes with double exponential jumps and the pricing of options. *Mathematics of Operations Research*, *35*(2), 412–437.

Cai, N., & Kou, S. (2011). Option pricing under a mixed-exponential jump diffusion model. *Management Science*, *57*(11), 2067–2081.

Cai, N., & Kou, S. (2012). Pricing asian options under a hyper-exponential jump diffusion model. *Operations Research*, *60*(1), 64–77.

Chockalingam, A., & Muthuraman, K. (2011). American options under stochastic volatility. *Operations Research*, *59*(4), 793–809.

Clark, P. K. (1973). A subordinated stochastic process model with finite variance for speculative prices. *Econometrica: Journal of the Econometric Society*, 135–155.

Cox, J. C., & Ross, S. A. (1976). The valuation of options for alternative stochastic processes. *Journal of Financial Economics*, *3*(1-2), 145–166.

Davydov, D., & Linetsky, V. (2001). Pricing and hedging path-dependent options under the CEV process. *Management science*, *47*(7), 949–965.

Duffie, D., Pan, J., & Singleton, K. (1998). *Transform analysis and option pricing for affine jump-diffusion* (tech. rep.). Working Paper, Graduate School of Business, Stanford University.

Engle, R. (1995). *Arch: Selected readings*. Oxford University Press.

Feng, L., & Linetsky, V. (2008). Pricing options in jump-diffusion models: An extrapolation approach. *Operations Research*, *56*, 304–325.

Fouque, J.-P., Papanicolaou, G., & Sircar, K. R. (2000). *Derivatives in financial markets with stochastic volatility*. Cambridge University Press.

Geman, H., Madan, D. B., & Yor, M. (2001). Time changes for Lévy processes. *Mathematical Finance*, *11*(1), 79–96.

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

Heyde, C. C. (1999). A risky asset model with strong dependence through fractal activity time. *Journal of Applied Probability*, *36*(4), 1234–1239.

Hull, J., & White, A. (1987). The pricing of options on assets with stochastic volatilities. *The Journal of Finance*, *42*(2), 281–300.

Jarrow, R., & Rosenfeld, E. (1984). Jump risks and the intertemporal capital asset pricing model. *Journal of Business*, *57*, 337–351.

Jorion, P. (1988). On jump processes in the foreign exchange and stock markets. *The Review of Financial Studies*, *1*(4), 427–445.

Kou, S. (2002). A jump-diffusion model for option pricing. *Management Science*, *48*(8), 1086–1101.

Kou, S., & Wang, H. (2004). Option pricing under a double exponential jump diffusion model. *Management Science*, *50*(9), 1178–1192.

Lange, R., Ralph, D., & Støre, K. (2020). Real-option valuation in multiple dimensions using poisson optional stopping times. *Journal of Financial and Quantitative Analysis*, *55*(2), 653–677.

Madan, D. B., & Seneta, E. (1990). The variance gamma (VG) model for share market returns. *Journal of Business*, 511–524.

Mandelbrot, B. (1963). New methods in statistical economics. *Journal of Political Economy*, *71*(5), 421–440.

Merton, R. C. (1976). Option pricing when underlying stock returns are discontinuous. *Journal of Financial Economics*, *3*(1-2), 125–144.

Rogers, L. C. G. (1997). Arbitrage with fractional Brownian motion. *Mathematical Finance*, *7*(1), 95–105.

Samorodnitsky, G., & Taqqu, M. (1994). Non-Gaussian stable processes: Stochastic models with infinite variance. *Chapman ft Hall, London*.

A Constructing a grid

The true value $V_\lambda(X, Y)$ requires $\{(X, Y)\}$ to operate in continuous space $\mathbb{R}^2$. For the discretized approximation to approach the true value, we need only choose a grid that is dense enough for this application. An advantage of POST is that we can choose to vary the density over the grid and give the area around points of interest a denser grid. For the time dimension this turned out not to be necessary. A linear grid between minimum and maximum values $t_{\min} = 0$ and $t_{\max} = T$ worked just as well as a grid which was denser around $t = T$. An obvious point of interest for American options in the stock price dimension would be the strike price. Note that approximating the integral that represents the jump probability in the infinitesimal generator (equation 19) places stricter requirements on the density of the grid. When the X grid points are spaced far apart, $[F_y(x_{j+1} - x_i) - F_y(x_j - x_i)]$ contains a large amount of probability mass for each $V(x_j)$, making the total sum a bad approximation of the integral. Especially around the edge of the grid, where the grid is spaced further apart, this poses a challenge. A somewhat methodical approach to construct a grid has to ensure the grid is dense around the point of interest and still dense enough when we move away from the point of interest. To somewhat standardize the procedure a few quantities are defined: Minimum grid value $x_{\min}$, maximum grid value $x_{\max}$ and the point x_{mid}, around which the grid should be dense. In the option valuation case reasonable values are $x_{\min} = 0.001$, $x_{\max} = 10 \cdot K$ and $x_{\text{mid}} = K$. Notice the difference between $x_{\min}$ and x_{mid} is much smaller than the difference between x_{mid} and $x_{\max}$, therefore defining there should be an equal number of grid points in both ranges would lead to a skewed grid. To alleviate this we construct a symmetrical grid around x_{mid}, ranging from $x_{\min}$ to $2x_{\text{mid}} - x_{\min}$. Finding a methodical way to construct this grid posed a challenge, as taking a grid where the difference is based on e^x for example yielded a too dense grid around x_{mid} and too far spaced apart at the edges. We found the inverse normal CDF a versatile solution to this. When we define normal distribution $Y \sim N(x_{\text{mid}}, K/2)$ and take our grid points such that the chance mass between each point is the same between each grid point we get a perfectly spaced out grid, that can be further calibrated by changing $\sigma_Y = K/2$. In the algorithm below the CDF of the normal distribution with parameters μ and σ is denoted by $F_{\mu,\sigma}(x)$ and its inverse as $F^{-1}_{\mu,\sigma}(x)$. Other distributions can also be used as the researcher sees fit. A predefined number of grid points is then used to construct a tail from $2x_{\text{mid}} - x_{\min}$ to $x_{\max}$, usually linearly. The fraction of the total amount of grid points is defined as n_{tail}. The complete algorithm to construct a 'normal' grid then reads:

$y_{\min} \leftarrow F_{x_{\text{mid}},K/2}(x_{\min})$

$y_{\max} \leftarrow F_{x_{\text{mid}},K/2}(2x_{\text{mid}} - x_{\min})$

$\delta_y \leftarrow (y_{\max} - y_{\text{start}})/(\lceil N_x \cdot (1 - n_{\text{tail}})\rceil - 1)$

$\mathbf{y} \leftarrow [y_{\min} : \delta_y : y_{\max}]$

$\mathbf{x}_{-\text{tail}} \leftarrow F^{-1}_{x_{\text{mid}},K/2}(\mathbf{y})$

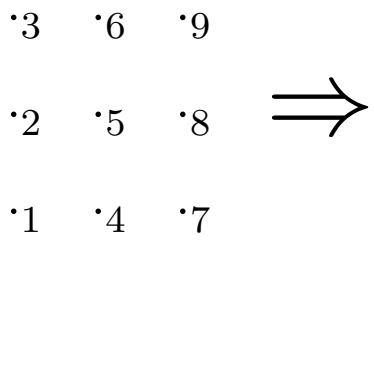
$\delta_{\text{tail}} \leftarrow (x_{\max} - \mathbf{x}_{-\text{tail}}(\text{end}))/\lfloor N_x \cdot n_{\text{tail}}\rfloor$

$\mathbf{x}_{\text{tail}} \leftarrow [\mathbf{x}_{-\text{tail}}(\text{end}) : \delta_{\text{tail}} : x_{\max}]$

$\mathbf{x} \leftarrow [\mathbf{x}_{-\text{tail}}, \mathbf{x}_{\text{tail}}]$

B Constructing L

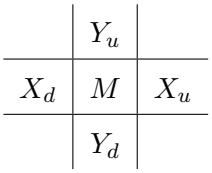
The constructed grid operates in two dimensions, every grid point has an x and y value. To use this grid in the POST algorithm, it needs to be translated to a one dimensional vector, where every element of the vector represents one grid point. It turns out that using a certain key to construct this vector makes constructing matrices needed in other parts of the algorithm easier. Take as graphical example a simple grid with $N_x = 3$ and $N_y = 3$, such that $X = [1, 2, 3]$ and $Y = [1, 2, 3]$. Construct the 'key' matrix in such a way that it cycles through all values of Y first, after which it moves to the next value of X. This becomes clear when we represent it visually and we number the different grid points 1 through 9 and write down the (x, y) values for each grid point.



point	(x, y)
1	$(1, 1)$
2	$(1, 2)$
3	$(1, 3)$
4	$(2, 1)$
5	$(2, 2)$
6	$(2, 3)$
7	$(3, 1)$
8	$(3, 2)$
9	$(3, 3)$

Recall that $\mathbf{L}$ is then constructed such that every row represents one grid point and that every column of that row connects it to another grid point to ensure the discretized equation of the infinitesimal generator holds. Once again visualising makes it easier to understand. Below a stencil and $\mathbf{L}$ corresponding to the constructed grid are shown. Note that the points such as $(3, 4)$ and $(4, 3)$ seem to interrupt the pattern. This is because point $3 = (1, 3)$ in our grid is a

boundary point. Y_u should refer to $(1,4)$ for point 3, which does not exist in our grid and is not point $4 = (2,1)$.



	1	2	3	4	5	6	7	8	9
1	M	Y_u		X_u					
2	Y_d	M	Y_u		X_u				
3		Y_d	M			X_u			
4	X_d			M	Y_u		X_u		
5		X_d		Y_d	M	Y_u		X_u	
6			X_d		Y_d	M			X_u
7				X_d			M	Y_u	
8					X_d		Y_d	M	Y_u
9						X_d		Y_d	M

Constructing $\mathbf{L}$ this way keeps the Y values of the stencils clustered together. This is especially important when constructing $\mathbf{L}_{\text{ex}}$. When we choose Y to represent stock price all entries of $\mathbf{L}$ that refer to grid points at the same time (same x value) are in submatrices on the diagonal. In this example all grid points referred to by the entries in $\mathbf{L}[1,2,3;1,2,3]$ have the same x value. This allows us to create a single matrix for jump possibilities and repeat this N_t times along the diagonal of $\mathbf{L}_{\text{ex}}$. Recall that every entry in $\mathbf{L}_{\text{ex}}$ represents the chance the stock price jumps from the row value to the column value. Say in this example the chance of a jump upwards once is 0.1, twice is 0.05, jumping downwards is 0.2, twice is 0.15 and the chance of not jumping far enough to reach the next grid point thus 0.5. When we assume jumps outside of the grid end up at the boundary, we need to add the lost jump probabilities to the boundary point. Take point $3 = (1,3)$ where no more upward jumps can occur. The possibility of staying at $(1,3)$ becomes $0.5 + 0.1 + 0.05 = 0.65$. $\mathbf{L}_{\text{ex}}$ then becomes

	1	2	3	4	5	6	7	8	9
1	0.85	0.1	0.05						
2	0.35	0.5	0.15						
3	0.15	0.2	0.65						
4				0.85	0.1	0.05			
5				0.35	0.5	0.15			
6				0.15	0.2	0.65			
7							0.85	0.1	0.05
8							0.35	0.5	0.15
9							0.15	0.2	0.65

Notice that for every time period the matrix is exactly the same. All we have to do to is construct the unique part of $\mathbf{L}_{\text{ex}}$ once and place it N_x times on the diagonal of a further sparse matrix.

C Adaptive stencil for negative drift

When a parameter with a negative sign enters $\mathbf{L}$ outside of the diagonal, it violates the convergence requirements. To ensure that a negative drift is handled in the correct way we can use a backwards difference discretization instead of a forward difference. This ensures the sign of the parameter (μ in the example below) is negative on the diagonal of $\mathbf{L}$ and positive everywhere else. A way to represent this when the sign of μ is unknown is shown below.

	$\max(\mu, 0)$	
	$-\mid \mu \mid$	
	$\max(-\mu, 0)$	

This stencil ensures a forward difference is applied when $\mu > 0$, as the bottom of the stencil becomes 0 and the top μ, and a backward difference is applied when $\mu < 0$. This ensures the convergence requirements are met.

D Boundary conditions for time dimension

At the left side of the grid (for t going to 0) no assumptions are necessary. The forward difference scheme ensures values for t smaller than the grid are not taken into consideration. The right side of the grid asks for a different approach. When the option nears expiry (t nears T), we observe three distinct phases. At $t < T$ the usual dynamics between asset value and option value follow their normal pattern. Then at $t = T$, the option value is fixed at its payoff, which is $(S-K)^+$ in the case of an American call. After expiry, the option is worthless. As the values of $t > T$ have no effect on the option value for $t \leq T$ (the option does not exist after it has expired) we terminate the grid at $t = T$ and set the option values at these points $V_\lambda^{(1)}(S,T) = (S-K)^+$. To ensure this value is fixed for the other iterations $V_\lambda^{(j)}(S,T) = V_\lambda^{(j-1)}(S,T) = (S-K)^+$ must hold. We derive the value of $\mathbf{L}$ for which this equation holds. We start at the algorithms characteristic equation:

$$[(r+\lambda)\mathbf{I} - \mathbf{L}]\mathbf{V}_\lambda^{(\mathbf{j})} = \mathbf{f} + \lambda \max(\mathbf{g}, \mathbf{V}_\lambda^{(j-1)}). \tag{24}$$

In the American put application $\mathbf{f} = \mathbf{0}$. At the grid points for which $t = T$ holds $\mathbf{g} = \mathbf{V}_\lambda^{(j-1)}$, simplifying the equation to:

$$[(r+\lambda)\mathbf{I} - \mathbf{L}]\mathbf{V}_\lambda^{(\mathbf{j})} = \lambda \mathbf{V}_\lambda^{(\mathbf{j-1})}. \tag{25}$$

From above equation we easily see that setting $\mathbf{L} = r\mathbf{I}$ ensures $\mathbf{V}_\lambda^{(j)} = \mathbf{V}_\lambda^{(j-1)} (= (S-K)^+)$.

$$[(r+\lambda)\mathbf{I} - r\mathbf{I}]\mathbf{V}_\lambda^{(j)} = [\lambda \mathbf{I}]\mathbf{V}_\lambda^{(j)} = \lambda \mathbf{V}_\lambda^{(j-1)}. \tag{26}$$

To enforce the boundary conditions at the right side of the grid we thus fill all rows of $\mathbf{L}$ that correspond with a grid point at the right side of the grid with r at the diagonal element and zeroes anywhere else.

E Boundary conditions for dimensions affecting stock price

The boundary conditions in our case are dictated by the way V_λ behaves for extreme values of the variables that it depends on. For American put options $V_\lambda(t, Y)$ depends on the stock price. At the bottom of our grid $S_t = S_0 e^Y$ approaches zero. As the pay-off of the option then approaches K, we assume $V(t, Y) = K$ for $Y = Y_{\min}$. In practice this means we fill in $V(t, Y - \mathrm{d}x) = K$ for the bottom of the stencil. At the top the grid S_t becomes very large. As S_t moves further away from K, the probability that the option remains out-of-the-money nears 1. We can safely assume $V_\lambda(t, Y) = 0$ for $Y > Y_{\max}$. For American put options with stochastic volatility $V_\lambda(X, Y, t)$ the value depends on asset price X and time-varying volatility Y. The boundary conditions for the asset price process do not change with time-varying volatility. The option value is not very sensitive to volatility, as such we extrapolate the option value using $V_\lambda(X, y_{\text{end}+1}, t) = 2V_\lambda(X, y_{\text{end}}, t) - V_\lambda(X, y_{\text{end}-1}, t)$. For the Asian call options we extrapolated the value at the boundaries for both A and X.

F Handling a non-uniform grid

To apply the difference schemes to varying $\mathrm{d}x$, we define the upwards and downwards differences at x_i as $\mathrm{d}x_{\mathrm{u}} = x_{i+1} - x_i$ and $\mathrm{d}x_{\mathrm{d}} = x_i - x_{i-1}$ respectively. We also define the average difference $\mathrm{d}x_{\text{ave}} = \frac{1}{2}(\mathrm{d}x_{\mathrm{u}} + \mathrm{d}x_{\mathrm{d}})$. For a forwards or backwards difference this change is trivial. The central difference for the second derivative is derived as follows:

$$\begin{aligned} f''(x) &\approx \frac{\frac{f(x+\mathrm{d}x_{\mathrm{u}})-f(x)}{\mathrm{d}x_{\mathrm{u}}} - \frac{f(x)-f(x-\mathrm{d}x_{\mathrm{d}})}{\mathrm{d}x_{\mathrm{d}}}}{\mathrm{d}x_{\text{ave}}}, \\ &= \frac{f(x+\mathrm{d}x_{\mathrm{u}})}{\mathrm{d}x_{\mathrm{u}}\mathrm{d}x_{\text{ave}}} + \frac{f(x-\mathrm{d}x_{\mathrm{d}})}{\mathrm{d}x_{\mathrm{d}}\mathrm{d}x_{\text{ave}}} - \frac{f(x)}{\mathrm{d}x_{\mathrm{d}}\mathrm{d}x_{\text{ave}}} - \frac{f(x)}{\mathrm{d}x_{\mathrm{u}}\mathrm{d}x_{\text{ave}}}. \end{aligned}$$

The discretized covariance term can be handled as follows:

$$f_{xy}(x,y) \approx \frac{f(x+\mathrm{d}x_\mathrm{u}, y+\mathrm{d}y_\mathrm{u}) - f(x+\mathrm{d}x_\mathrm{u}, y-\mathrm{d}y_\mathrm{d}) + f(x-\mathrm{d}x_\mathrm{d}, y+\mathrm{d}y_\mathrm{u}) - f(x-\mathrm{d}x_\mathrm{d}, y-\mathrm{d}y_\mathrm{d})}{(\mathrm{d}x_\mathrm{u}+\mathrm{d}x_\mathrm{d})(\mathrm{d}y_\mathrm{u}+\mathrm{d}y_\mathrm{d})}.$$

G Approximation of the jump integral

Transforming the jump expression from the infinitesimal generator to a workable quantity for the algorithm means finding an approximation that can be computed with the grid we use. Start by dividing the integral in two and noting that the second term is an integral over a PDF. As such we can reduce it to $V(y_i)$.

$$\int_{\mathbb{R}} [V(y_i+z) - V(y_i)] f_z(z)\mathrm{d}z = \int_{\mathbb{R}} V(y_i+z) f_z(z)\mathrm{d}z - \int_{\mathbb{R}} V(y_i) f_z(z)\mathrm{d}z, \tag{27}$$

$$= \int_{\mathbb{R}} V(y_i+z) f_z(z)\mathrm{d}z - V(y_i). \tag{28}$$

The remaining integral operates over z, whereas our grid ranges from y_1 to y_{N_y}. Of the two quantities we need to compute with z, $V_(y_i+z)$ and $f_z(z)$, only $V_(y_i+z)$ is restricted to our grid as $f_z(z)$ is the PDF of the chosen jump distribution which is computable for all possible values of z. We get the most accurate approximation of the integral when we use as many grid points as possible, which means the discretization of $V(y_i+z)$ should use all available grid points and y_i+z should range from y_1 to y_{N_y}. We thus transform the variables to $y_j = y_i+z$ and $z = y_j - y_i$, where $y_j = (y_1, ..., y_{N_y})$ is our grid. Splitting the integral in $N_y - 1$ parts allows us to estimate the parts using the CDF of $f_z(z)$.

$$\int_{\mathbb{R}} V(y_i+z) f_z(z)\mathrm{d}z - V(y_i) = \int_{\mathbb{R}} V(y_j) f_z(y_j-y_i)\mathrm{d}(y_j-y_i) - V(y_i), \tag{29}$$

$$= \sum_{j=1}^{N_y-1} \int_{y_j}^{y_j+1} V(y_j) f_z(y_j-y_i)\mathrm{d}(y_j-y_i) - V(y_i), \tag{30}$$

$$\approx \sum_{j=1}^{N_y-1} V(y_j) \int_{y_j}^{y_j+1} f_z(y_j-y_i)\mathrm{d}(y_j-y_i) - V(y_i), \tag{31}$$

$$= \sum_{j=1}^{N_y-1} V(y_j)[F_z(y_{j+1}-y_i) - F_z(y_j-y_i)] - V(y_i). \tag{32}$$

Note that we discarded the tails in above derivation. When we assume jumps outside of the grid end up at the boundaries of the grid these can be computed as $V(y_1)[F_z(y_1-y_i)]$ and $V(y_{N_j})[1-F_z(y_{N_j}-y_i)]$.

H Iterative procedure for refining grid and increasing POST iterations

A major complexity of the POST algorithm is that it requires two different aspects of the model to be tweaked to increase the accuracy of the outcome. On one side increasing POST iterations j and POST arrival rate λ increases the accuracy for a given grid and on the other hand choosing a finer grid increases the overall accuracy. There is a methodical way to tackle this and it involves the starting value $V_\lambda^{(0)}$. Usually the researcher would set this to g and start the algorithm. Setting it to a different value allows us to chop up the algorithm in different stages and speed up convergence. To get quicker convergence start with a low λ_1, run the algorithm for some iterations and once convergence slows down run the algorithm again with a higher λ_2 and setting $V_{\lambda_2}^{(0)} = V_{\lambda_1}^{(\text{end})}$. Instead of running the algorithm once with 1000 iterations and $\lambda = 50$, we can run it three times with $\lambda = 5, 25, 50$, with three times 100 iterations and get the same result. In our numerical applications we generally used the pairs $(\lambda, j) = (2, 10), (16, 20), (64, 40), (256, 80), (512, 200), (1024, 200)$. This proved to give consistent results, increasing λ or j did not provide more accuracy. One could even go so far as to use an iterative procedure to refine the grid. The same principle for refining λ applies. Start out by running the algorithm, refine the grid and use the previous end value as a starting point for the next run of the algorithm. This can be done with a different grid by interpolating the values of $V_\lambda^{(0)}$ at the new grid points. In our numerical applications this was not used, rather the amount of grid points was chosen based on the time needed to run the algorithm. Anywhere between 10 to 30 minutes was deemed acceptable.

I Time as a discrete dimension

With a third dimension comes a large extra computational burden. Especially the dimensions of L increase rapidly when the number of grid points increases. Recall the equation that enforces the POST algorithm. In every infinitesimal generator t only enters the equation only through $-1/\mathrm{d}t$, which allows us to separate the grid points and run the algorithm separately for every value of t, starting at $t = T$. This reduces the problem to N_t different $N_x \cdot N_y$ problems instead of one $N_t \cdot N_x \cdot N_y$ problem. Due to the dimensions of L this actually saves a large amount of time. To show this is computationally equivalent we zoom in on one specific grid point (x, y, t). Keep in mind that the construction with L is only meant to enforce the POST equation for every grid point at the same time. When we isolate $V_{\lambda_P}^{(j)}(x, y, t + \mathrm{d}t)$ we see that solving the problem for each time step individually is computationally equivalent to solving it in one go. Denote

$\mathrm{L}_{x,y,t}$ as the infinitesimal generator containing all terms and $\mathrm{L}_{x,y}$ as the infinitesimal generator without the term $1/\mathrm{d}t$. As $f(x,y,t)=0$ and $g(x,y)$ does not depend on t the following equation should hold:

$$
\begin{aligned}
(r+\lambda)V_\lambda^{(j)}(x,y,t) - \mathrm{L}_{x,y,t}V_\lambda^{(j)}(x,y,t) &= \max[g(x,y), V_\lambda^{(j-1)}(x,y,t)],\\
(r+\lambda)V_\lambda^{(j)}(x,y,t) - \mathrm{L}_{x,y}V_\lambda^{(j)}(x,y,t)&\\
+\frac{1}{\mathrm{d}t}(V_\lambda^{(j)}(x,y,t) - V_\lambda^{(j)}(x,y,t+\mathrm{d}t)) &= \max[g(x,y), V_\lambda^{(j-1)}(x,y,t)],\\
(r+\lambda+\frac{1}{\mathrm{d}t})V_\lambda^{(j)}(x,y,t) - \mathrm{L}_{x,y}V_\lambda^{(j)}(x,y,t) &= \frac{1}{\mathrm{d}t}V_\lambda^{(j)}(x,y,t+\mathrm{d}t) + \max[g(x,y), V_\lambda^{(j-1)}(x,y,t)].
\end{aligned}
$$

After $(1/\mathrm{d}t)V_\lambda^{(j)}(x,y,t+\mathrm{d}t)$ is isolated it becomes clear that the rest of the variables are no further interrelated. Starting at $t+\mathrm{d}t=T$ we only have known quantities on the right hand side, as $V(x,y,T)=g(x,y)$, independent of λ or j. This allows us to solve adjusted problem for each period by iterating. Note that it can be incorporated in existing code by setting $r_{adj}=r+1/\mathrm{d}t$ and $f_{adj}(x,y,t)=f(x,y,t)+(1/\mathrm{d}t)V_\lambda^{(J)}(x,y,t+\mathrm{d}t)$.

J Kou's analytical approximation

The approximation of an American put in Kou and Wang, 2004 revolves around finding the exercise boundary v_0 and computing the early exercise premium above that boundary. Both computations utilise the approximation of a European put from Kou, 2002, which in turn closely resembles the standard BSM specification. The standard normal CDF found in BSM is replaced by a CDF that also takes the double exponential jumps into account. Boundary v_0 is then obtained as the solution to an equation that represents the free boundary problem. With these quantities and some constants $\beta_{1,\dots,4}$ that depend on the parameters of the jump-diffusion process, the value of an American put can be computed as follows:

$$
\psi(v,t) = f(x) = \begin{cases} EuP(v,t) + Av^{-\beta_3} + Bv^{-\beta_4}, & \text{if } v \geq v_0 \\ K - v, & \text{if } v \leq v_0. \end{cases} \tag{33}
$$

Here $EuP(v,t)$ is Kou's approximation of a European put with maturity t and strike K, and with A and B constants:

$$
A = \frac{v_0^{\beta_3}}{\beta_4-\beta_3}\left\{\beta_4 K - (1+\beta_4)\left[v_0 + EuP\left(v_0,\ t\right)\right] + Ke^{-rt}P^{v_0}\left[S\left(t\right)\leq K\right]\right\} > 0, \tag{34}
$$

$$
B = \frac{v_0^{\beta_4}}{\beta_3-\beta_4}\left\{\beta_3 K - (1+\beta_3)\left[v_0 + EuP\left(v_0,\ t\right)\right] + Ke^{-rt}P^{v_0}\left[S\left(t\right)\leq K\right]\right\} > 0, \tag{35}
$$

with $P^{v_0}[S(t)\leq K]$ the probability that $S(t)\leq K$ given $S_0=v_0$.