



ELSEVIER

Contents lists available at ScienceDirect

Finance Research Letters

journal homepage: [www.elsevier.com/locate/frl](http://www.elsevier.com/locate/frl)



## Computing American option prices in the lognormal jump–diffusion framework with a Markov chain

Jean-Guy Simonato\*

Service de l'enseignement de la finance, HEC Montréal, 3000 Côte-Sainte-Catherine, Montréal (Québec), Canada H3T 2A7

### ARTICLE INFO

#### Article history:

Received 12 January 2011

Accepted 14 January 2011

Available online 22 January 2011

#### Jel classification:

C60

C61

C63

G13

#### Keywords:

American option

Jump–diffusion

Markov chain

### ABSTRACT

This note examines a numerical approach for computing American option prices in the lognormal jump–diffusion context. The approach uses the known transition density of the process to build a discrete-time, homogenous Markov chain to approximate the target jump–diffusion process. Numerical results showing the performance of the proposed method are examined.

© 2011 Elsevier Inc. All rights reserved.

## 1. Introduction

It is well known that the Black and Scholes (1973) model fails to capture key phenomena observed in financial markets such as the skewness and fat tails of stock return distributions and the possibility of discontinuities in stock prices. A typical approach used to address these features is the addition of Poisson distributed jumps to the geometric Brownian motion diffusion process assumed in Black–Scholes. Two well known models using such processes are Merton (1976), with lognormal jumps, and Kou (2002), with double exponential distributed jumps. As pointed in Kou (2008), such models do not capture all the characteristics of stock returns. For example, they cannot capture the well documented volatility clustering phenomenon. However, unlike many models coherent with this feature,

\* Fax: +1 514 340 5632.

E-mail address: [jean-guy.simonato@hec.ca](mailto:jean-guy.simonato@hec.ca)

the Merton (1976) and Kou (2002) jump–diffusion models provide closed form solutions for computing European option prices in a variety of settings. Such a characteristic is interesting in many practical situations.

Just as for the Black–Scholes case, there is no closed form solution for computing American options prices in the jump–diffusion context. For such derivatives, a numerical method is required. Several approaches are proposed in the literature: finite difference methods (Andersen and Andreasen, 2000), the Fourier transform approach (Chiarella and Zogas, 2009), tree methods (Amin, 1993) and the method of lines (Meyer, 1998). Despite this variety, to our knowledge, there is no numerical work based on the direct use of the stock price distribution. This note examines how a Markov chain approach, which uses the known transition density of the stock price to build an approximating Markov chain, can be used to compute prices for American options in the jump–diffusion context with lognormal jumps. Such a method is interesting since it is conceptually simple, easy to implement and produces good results.

Section 2 provides a brief description of the Merton's (1976) jump–diffusion process and its associated return distribution. Section 3 describes the Markov chain approach while Section 4 provides some numerical examples showing the performance of the method. Section 5 concludes the paper.

## 2. The jump–diffusion framework

In the Merton (1976) framework, the stock price follows the risk neutral process

$$\frac{dS_t}{S_t} = (r - \rho - \lambda k)dt + \sigma dZ + dJ$$

where  $S_t$  is the stock price,  $r$  is the annual continuously compounded risk-free rate,  $\rho$  is the annual continuously compounded dividend rate,  $Z$  is a standard Brownian motion,  $\sigma$  the constant volatility parameter associated to the Brownian motion,  $\lambda$  is the arrival rate of jumps and  $dJ$  is the jump portion of the process, independent of  $Z$ . Over an interval  $dt$ , the stock price can jump with probability  $\lambda dt$ . The jump portion  $dJ$  takes a value of 0 if there is no jump and  $Y - 1$  if there is jump with  $E(Y - 1) = k$ . It is assumed that the jump magnitude,  $Y$ , is lognormally distributed, i.e.  $\ln Y \sim N(\alpha_j, \sigma_j^2)$  which yields  $k = e^{\alpha_j} - 1$ .

In this context, the density of the logarithm of the stock price  $h$  years from now, conditional on the current stock price value, is available in closed form and is given by

$$\phi_J(\ln S_{t+h} | \ln S_t) = \sum_{i=0}^{\infty} \frac{e^{-\lambda h} (\lambda h)^i}{i!} \phi_N(\mu_i, \sigma_i^2)$$

where  $\phi_N(\mu_i, \sigma_i^2)$  is the normal density with  $\mu_i = \ln S_t + (\tilde{\alpha} - \frac{1}{2} \sigma^2)h + i(\alpha_j - 0.5\sigma_j^2)$  with  $\tilde{\alpha} = r - \rho - \lambda k$  and variance  $\sigma_i^2 = \sigma^2 h + i\sigma_j^2$ . The distribution function can be written as

$$\Phi_J(a | \ln S_t) = \sum_{i=0}^{\infty} \frac{e^{-\lambda h} (\lambda h)^i}{i!} N\left(\frac{a - \mu_i}{\sigma_i}\right) \tag{1}$$

where  $N(\cdot)$  is the standard normal distribution function. Although the above functions require computing an infinite sum, the magnitude of the elements quickly decay to negligible values for all practical purposes. Hence, only the first few elements are used in actual implementation. The next section shows how an approximating Markov chain can be built to compute option prices in the jump–diffusion context.

## 3. A Markov chain approach

In order to compute the prices of American options in the context of the above process, an homogeneous discrete-time Markov chain will be used. As shown in Duan and Simonato (2001), one can construct this chain in such a way that, as the number of states goes to infinity, it converges to the target

stochastic process over the time points  $0, h, 2h \dots$ . Option prices computed with it will converge to the theoretical option values.

The idea behind the construction of an  $m$  states approximating Markov chain is as follows. The real line, the support of the log of the stock price distribution, is first partitioned into  $m$  distinct cells. A numerical value is then assigned to each cell, yielding the  $m$  possible values to be taken by the log of stock price. Given one of these values, the probability of reaching another value is computed as the probability to land in a given cell  $h$  years from now. For the stock process assumed above, these probabilities can be conveniently computed with the distribution function. The  $m$  log of stock prices and computed probabilities are then stored in a vector and a matrix, allowing simple matrix multiplications to be used when computing the option prices.

More formally, denote the  $m$  log of stock prices to be  $\mathbf{p} = [p_1, p_2, \dots, p_m]'$ . The probability matrix describing the transition from one state to the other over an  $h$  year period is

$$\mathbf{Q} = \begin{bmatrix} q_{11} & \cdots & q_{1m} \\ \vdots & \ddots & \vdots \\ q_{m1} & \cdots & q_{mm} \end{bmatrix} \tag{2}$$

Here,  $m$  is an odd integer and  $p_{(m+1)/2} = \ln(S_0)$ . Using the vector  $\mathbf{p}$  and matrix  $\mathbf{Q}$  an American option price with a maturity  $T$  years and strike price  $K$  can be computed in  $n = T/h$  time steps with the following recursive system:

$$\mathbf{V}(\mathbf{p}, t) = \max [\mathbf{g}(\mathbf{p}, K), e^{-rh} \mathbf{QV}(\mathbf{p}, t + h)] \tag{3}$$

with

$$\mathbf{g}(\mathbf{p}, K) = \max [w(\exp(\mathbf{p}) - K\mathbf{1}), \mathbf{0}]$$

and where the recursion is initialized with  $\mathbf{V}(\mathbf{p}, T) = \mathbf{g}(\mathbf{p}, K)$ . Here,  $\mathbf{V}(\mathbf{p}, t)$  is the time- $t$  option price vector corresponding to vector  $\mathbf{p}$ ;  $\max[\cdot, \cdot]$  is a vector-valued function returning the maximum value on an element-by-element basis and is the option's payoff function with  $\mathbf{0}$  and  $\mathbf{1}$  denoting vectors of zeros and ones, and  $w$  indicates a call ( $w = 1$ ) or a put ( $w = -1$ ). The time-0 option price is the  $(m + 1)/2$ th element of  $\mathbf{V}(\mathbf{p}, 0)$ .

### 3.1. Constructing the chain

To construct the vector of log stock prices,  $\mathbf{p}$ , an overall interval covering the set of representative asset prices over a period of  $T$  years is defined. This interval is written as  $[\ln S_0 - I_p, \ln S_0 + I_p]$ . The quantity  $I_p$  is computed using the conditional standard deviation of the asset return over the life of the option contract multiplied by a scaling factor. Formally,

$$I_p = \delta(m) \times v \times \sqrt{T}$$

where  $v = \sqrt{\sigma^2 + \lambda\sigma_j^2 + \lambda\alpha_j^2}$  is the annualized standard deviation of the stock return (see Das and Sundaram (1999)). The quantity  $v \times \sqrt{T}$  is the standard deviation of the stock return over a period of  $T$  years. The scaling factor  $\delta(m)$  is an increasing function of  $m$ , satisfying some mild partition conditions stating that  $\delta(m)$  should grow at a rate smaller than  $m$ . In this study we opt for  $\delta(m) = \ln(m)$ . We then divide the above interval equally into  $m - 1$  parts to obtain  $m$  discrete values

$$p_i = \ln S_0 + \frac{2i - m - 1}{m - 1} I_p$$

with  $i \in \{1, \dots, m\}$  for the log of the asset price. Note that  $p_1 = \ln S_0 - I_p$  and  $p_m = \ln S_0 + I_p$  are the minimum and maximum values. In order to have  $\ln S_0$  among the state prices,  $m$  needs to be an odd integer which obtains  $p_{(m+1)/2} = \ln S_0$ .

To compute the elements of matrix  $\mathbf{Q}$ , we define  $m$  cells that are constructed as  $C_i = [c_i, c_{i+1})$  for  $i = \{1, \dots, m\}$  and where  $c_1 = -\infty$ ,  $c_i = \frac{p_i + p_{i-1}}{2}$  for  $i = 2$  to  $m$  and  $c_{m+1} = +\infty$ . Using the above cells and

states, the transition probabilities i.e. the probability of landing in cell  $C_j$  given a current price  $p_i$  are computed with

$$q_{ij} = \Phi_j(C_{j+1}|p_i) - \Phi_j(C_j|p_i) \tag{4}$$

where  $\Phi(\cdot)$  is the distribution function of the logarithm of the stock price given in Eq. (1).

### 3.2. Implementation issues

The above procedure can be implemented in few lines of codes with standard matrix based programming languages such as Matlab, Gauss or SAS. However, when implementing the above Markov chain approach, some care should be taken about computational issues.

For many values of  $(i, j)$ , the transition probabilities are different from zero in theory, but are negligible for all practical purposes. A threshold value can thus be defined and all  $q_{ij}$  below this value can be set to zero. Such a procedure typically yields transition matrices with a large number of elements equal to zero. The use of sparse matrix techniques can be used to save on computation time and memory use. Such techniques allow substantial memory requirement reductions realized by storing only the non-zero entries. Computation time is also improved by avoiding the multiplications of zero entries. Standard matrix programming languages such as Matlab, Gauss or SAS allow the use of sparse matrices with few changes with respect to standard matrix operations.

Alternatively, the computational efficiency can also be increased by looking at the structure of the transition probability matrix. For this model, the transition matrix has a well defined structure which is shown here for the case  $m = 5$ :

$$\begin{bmatrix} q_{1,1} & q_{1,2} & q_{1,3} & q_{1,4} & q_{1,5} \\ \sum_{i=1}^{5-1} q_{5,i} & q_{2,2} & q_{1,2} & q_{1,3} & \sum_{i=4}^5 q_{1,i} \\ \sum_{i=1}^{5-2} q_{5,i} & q_{5,4} & q_{2,2} & q_{1,2} & \sum_{i=3}^5 q_{1,i} \\ \sum_{i=1}^{5-3} q_{5,i} & q_{5,3} & q_{5,4} & q_{2,2} & \sum_{i=2}^5 q_{1,i} \\ q_{5,1} & q_{5,2} & q_{5,3} & q_{5,4} & q_{5,5} \end{bmatrix}$$

All the elements below the diagonal can be computed from the elements of the last line of the matrix. All the elements above the diagonal can be computed with the elements from the first line. Except for the first and last lines, the elements on the diagonal are all equal between each other.<sup>1</sup> The above and below diagonal dichotomy is induced by the asymmetric distribution of the log prices of jump–diffusion process. The possibility to compute all the elements of the upper or lower diagonal matrix with the first (last) line is induced by the partition of the state space in cells of identical length (except for the first and last ones).

With such a structure, all the probabilities from the transition matrix can be computed from  $2m + 1$  transition probabilities. It should also be noticed that the probabilities of line 1 are monotonically decreasing while those from the last line are monotonically increasing. It is therefore easy to find which elements of the transition matrix will be zero once a threshold value (the number below which a probability is considered to be zero) is defined. Appendix A gives a simple and compact pseudo-code taking advantage of this structure for the computation of the matrix by vector multiplication  $\mathbf{Q} \times \mathbf{V}(\cdot)$  in Eq. (3). This pseudo-code requires a minimal amount of memory and avoids the multiplications of zero-probabilities. This is the computational approach that is adopted here.

### 4. Numerical results

Table 1 shows European call option prices computed with the above procedure. The table reports option prices for increasing values of  $m$ , the number of elements in the Markov chain. To compute these prices, the number of time steps  $n$  is set equal to the number of days to maturity. The function

<sup>1</sup> For convenience, for lines 2 to  $m - 1$  of this matrix, the element  $q_{2,2}$  is placed on the diagonal in the above structure to show that all the elements are equal to this value.

**Table 1**  
European call option prices.

T	10/365	30/365	60/365	90/365	270/365
<i>Benchmark option prices</i>					
	1.0224	2.0474	3.0895	3.8847	7.1299
<i>Markov chain option prices</i>					
m = 501	1.0228	2.0495	3.0952	3.8954	7.1951
m = 1001	1.0225	2.0481	3.0913	3.8880	7.1500
m = 2001	1.0224	2.0476	3.0901	3.8857	7.1360
m = 3001	1.0224	2.0475	3.0898	3.8852	7.1329
m = 4001	1.0224	2.0475	3.0897	3.8850	7.1317
m = 5001	1.0224	2.0475	3.0896	3.8849	7.1311
m = 10,001	1.0224	2.0474	3.0896	3.8847	7.1302

This table reports European call option prices in the lognormal jump–diffusion context. Benchmark prices are computed with the Merton (1976) closed form formula. Parameter values:  $S_0 = 50$ ,  $K = 50$ ,  $r = 0.05$ ,  $\rho = 0$ ,  $\sigma = 0.2$ ,  $\lambda = 5$ ,  $\alpha_j = -0.1$ ,  $\sigma_j = 0.1$ . Markov chain prices are computed with  $n$  equal to the number of days to maturity,  $\delta(m) = \ln(m)$ , and by setting probabilities smaller than  $1 \times 10^{-10}$  to zero.

**Table 2**  
American call option prices.

$S_0$	80	90	100	110	120	
<i>Benchmark option prices</i>						
	0.9648	2.3063	5.3603	11.5079	20.1333	
<i>Markov chain option prices</i>						Time
m = 501	0.9710	2.3251	5.4076	11.5525	20.1506	0.119
m = 1001	0.9668	2.3121	5.3750	11.5214	20.1379	0.393
m = 2001	0.9654	2.3081	5.3647	11.5116	20.1340	1.273
m = 3001	0.9651	2.3072	5.3624	11.5094	20.1331	2.683
m = 4001	0.9649	2.3068	5.3615	11.5085	20.1327	4.490
m = 5001	0.9649	2.3066	5.3611	11.5080	20.1326	6.817
m = 10,001	0.9647	2.3062	5.3602	11.5073	20.1322	25.169

This table reports American call option prices in the lognormal jump–diffusion context. Benchmark prices are taken from Chiarella and Zogas (2009) and obtained using the Crank–Nicolson method with 10,000 time steps and 5000 space steps. Parameter values:  $T = 0.5$ ,  $K = 100$ ,  $r = 0.05$ ,  $\rho = 0.03$ ,  $\sigma^2 = 0.0136$ ,  $\lambda = 1$ ,  $\alpha_j = \ln(1, 04)$ ,  $\sigma_j^2 = 0.04$ . Markov chain prices are computed with  $n = 182$ ,  $\delta(m) = \ln(m)$ , and by setting probabilities smaller than  $1 \times 10^{-10}$  to zero. ‘Time’ is the average computing time in seconds.

$\delta(m)$  is set to  $\ln(m)$ , and probabilities smaller than  $1 \times 10^{-10}$  are set to zero. As  $m$  is increasing, we see the computed prices converging smoothly to the benchmark prices, which are available in closed form with the Merton (1976) formula. For short maturities, penny accuracy is reached with chains of 501 elements, which typically take fractions of seconds to compute on a standard laptop computer when using the procedure described in Appendix A. For options with longer maturities, penny accuracy is reached at 2001 or 3001 elements, which are typically computed in 2 or 3 seconds.

Table 2 reports the prices of American call options with a non-zero dividend rate, which can trigger the early exercise of these options. Benchmark values are taken from Chiarella and Zogas (2009), who report option prices computed with a finite difference approach using the Crank–Nicolson method with 10,000 time steps and 5000 space steps. Such benchmark prices should be very close to the true, but unknown, option prices. The option contracts in this table all have a maturity of  $T = 0.5$ . For the Markov chain prices, again, the function  $\delta(m)$  is set to  $\ln(m)$ , and probabilities smaller than  $1 \times 10^{-10}$  are set to zero. Early exercise is allowed on a daily frequency, i.e.  $n$  is set equal to 182. As the number of states is increased, the option prices computed with the Markov chain become closer to the benchmark prices. However, it should be noticed that the frequency of early exercise possibilities are different for the benchmark and the Markov chain prices. It is thus natural to find that some of the Markov chain prices converge to a value slightly lower than that of the benchmark. As for the European cases with a similar maturity, penny accuracy is reached with 2001 or 3001 states. The computing times are reasonable and are around 2 to 3 seconds for penny accuracy.

## 5. Conclusion

A Markov chain approach for pricing American options in the lognormal jump–diffusion framework is proposed. This approach uses the known transition density of the stock prices and presents a viable and simple alternative to the various approaches that have been proposed in the literature, such as tree methods, finite differences, Fourier transform, and the method of lines. The proposed approach is shown numerically to converge smoothly to benchmark values as the number of states of the Markov chain is increased. Further research could examine and compare the recursive integration approach suggested in Andricopoulos et al. (2003), which also uses the transition density of the stock price to compute derivatives prices.

## Appendix A. Pseudo-code for matrix by vector multiplication

This appendix describes the pseudo-code performing the operation  $\mathbf{Q} \times \mathbf{V}(\cdot)$  required in Eq. (3) without having to store all the elements of  $\mathbf{Q}$ . It also avoids multiplying the elements of  $\mathbf{Q}$  when they are zero. The following inputs are required by the pseudo-code:

- $q_{\text{first}}$ : an  $m \times 1$  vector =  $[q_{1,1}, q_{1,2}, \dots, q_{1,m}]'$
- $q_{\text{last}}$ : an  $m \times 1$  vector =  $[q_{m,1}, q_{m,2}, \dots, q_{m,m}]'$
- $v_{\text{sumfirst}}$ : an  $m \times 1$  vector =  $[\sum_{i=m}^m q_{1,i}, \sum_{i=m-1}^m q_{1,i}, \dots, \sum_{i=1}^m q_{1,i}]'$
- $v_{\text{sumlast}}$ : an  $m \times 1$  vector =  $[\sum_{i=1}^m q_{m,i}, \sum_{i=1}^{m-1} q_{m,i}, \dots, \sum_{i=1}^1 q_{m,i}]'$
- $nz_{\text{first}}$  and  $nz_{\text{last}}$ : the number of non-zero elements in  $q_{\text{first}}$  and  $q_{\text{last}}$
- $q_{ii}$ : scalar equal to  $q_{2,2}$

The pseudo-code below gives the output in the  $m \times 1$  vector  $\mathbf{u}$ :

```

/*Multiplying line 1 of matrix Q*/
for j=1 to nzfirst
    u(1)=u(1)+qfirst(j)*v(j)
end
/*Multiplying lines 2 to m-1 of matrix Q*/
for i=2 to m-1
    u(i)=u(i)+vsumlast(i)*v(1)
    if i>nzlast; cbeg=i-nzlast+1; else cbeg=2; end
    for j=cbeg to i-1
        u(i)=u(i)+qlast(m-i+j)*v(j)
    end
    u(i)=u(i)+qii*v(i)
    if i<=(m-nzfirst); cend=nzfirst+i; else cend=m-1; end
    for j=i+1 to cend
        u(i)=u(i)+qfirst(j-i+1)*v(j)
    end
    u(i)=u(i)+vsumfirst(i)*v(m)
end
/*Multiplying line m of matrix Q*/
for j=nzlast to m
    u(m)=u(m)+qlast(j)*v(j)
end

```

## References

- Amin, K., 1993. Jump diffusion option valuation in discrete time. *Journal of Finance* 48, 1833–1863.
- Andersen, L., Andreasen, J., 2000. Jump–diffusion processes: volatility smile fitting and numerical methods for option pricing. *Review of Derivatives Research* 4, 231–262.

- Andricopoulos, A., Widdicks, M., Duck, P., Newton, D., 2003. Universal option valuation using quadrature methods. *Journal of Financial Economics* 67, 447–471.
- Black, F., Scholes, M., 1973. The pricing of options and corporate liabilities. *Journal of Political Economy* 81, 637–659.
- Chiarella, C., Ziogas, A., 2009. American call option on jump–diffusion processes: a Fourier transform approach. *Applied Mathematical Finance* 16, 37–79.
- Das, S., Sundaram, R., 1999. Of smiles and smirks: a term structure perspective. *Journal of Financial and Quantitative Analysis* 34, 211–239.
- Duan, J.C., Simonato, J.G., 2001. American option pricing under GARCH by a Markov chain approximation. *Journal of Economic Dynamics and Control* 25, 1689–1718.
- Kou, S.G., 2002. A jump–diffusion model for option pricing. *Management Science* 48, 1086–1101.
- Kou, S., 2008. Jump–diffusion models for asset pricing in financial engineering. *Handbooks in Operations Research and Management Science*, vol. 15. North-Holland, Amsterdam, pp. 73–116.
- Merton, R.C., 1976. Option pricing when the underlying process for stock returns is discontinuous. *Journal of Financial Economics* 3, 124–144.
- Meyer, G., 1998. The numerical valuation of options with underlying jumps. *Acta Mathematica* 67, 69–82.