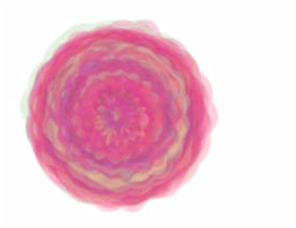

Okada Algorithm
Private Invention
Research Laboratory
Shizuoka City, Japan
Masashi Okada
okadaalgorithm@gmail.com
https://github.com/jirotubuyaki

A Vignette
Jdmbs version 1.3
2018-05-01

Jdmbs: An R Package for Monte Carlo Option Pricing Algorithms for Jump Diffusion Models with Correlational Companies

Abstract

Black-Scholes model is important to calculate option price in the stock market, and sometimes stock prices show jump phenomena. In order to handle it, a variety of jump diffusion models are studied. In this paper, we propose a new jump diffusion model which has correlation coefficients in order to calculate the option prices of related companies together. Its model can treat jump phenomena which happen among companies represented as a directed graph structure. Then, we simulate a implemented model in this package.

Introduction

In the early 1970's, Black-Scholes (BS) model[1] was proposed. This model can calculate option prices as market transactions of derivatives. BS model is illustrated as geometric Brownian motion in Stochastic differential equation. Option prices are calculated from geometric Brownian motion under a risk-neutral probability. The appearance of BS model expanded and grew option markets at a rapid pace. For the achievement, Scholes and Marton won the Novel prize. However BS model does not represent all aspects of characteristics of the real market. Therefore the expansions of BS models are studied and proposed. Especially the time-series of a stock price exhibits phenomena like price jumps. In response to it, Jump diffusion model[2][3] using Poison process to represent jump phenomena is researched. In this paper, we propose a new jumps model which models the correlations of companies. A jump phenomenon of one company affects the jumps of other correlational companies as obeying correlation coefficient, and its model can calculate the prices of companies together. In this package, the new model and the algorithm which calculates correlation coefficients are implemented. Finally, we explain how to use it and simulate it.

Background

Black Scholes Model

There are several types of options in the stock market. European call option can not execute until the duration of T is finished, and its strike price is K. Option prices are calculated under a risk-neutral probability. European call option price is given by

$$Price = E[max(X(T) - K, 0)], \tag{1}$$

where $E[x]$ denotes expected value of x. European put option price is given by

$$Price = E[max(K - X(T), 0)]. \tag{2}$$

Black-Scholes equation is given by

$$C = e^{-qt}SN(d_1) - e^{-rt}KN(d_2), \tag{3}$$

$$d_1 = \frac{ln(\frac{s}{k}) + (r - q + \frac{1}{2}\sigma^2)t}{\sigma\sqrt{t}}, \tag{4}$$

$$d_2 = d_1 - \sigma\sqrt{t}, \tag{5}$$

where C is a call option price and t is a duration, K is a strike price. σ is a volatility. r is a risk-free interest rate. N is a Gauss distribution.

Poison Process

The Poisson process presents random phenomena happened at any timings. It is widely used in order to model random points in both time and space. Poison process is given by

$$P(X(t+s) - X(t) = k) = e^{-\lambda s}\frac{(\lambda s)^k}{k!}, \tag{6}$$

where λ is the arrival intensity. k is a number something happen.

Mixed-Exponential Jump Diffusion Model

Under the mixed-exponential jump diffusion model (MEM), the dynamics of the asset price St are given by

$$\frac{dS(t+1)}{S(t)} = \mu dt + \sigma dW(t) + d(\sum_{i=1}^{N(t)} Y_i), \tag{7}$$

$$dJ_t = d(\sum_{i=1}^{N(t)} Y_i), \tag{8}$$

where r is the risk-free interest rate, σ is the volatility, $\{N(t) : t = 0 \cdots\}$ a Poisson process with rate λ, $\{W(t) : t = 0 \cdots\}$ is a standard Brownian motion.

Correlational Jumps Model

Standard jump diffusion model causes jumps in the one stock market and it does not affect other companies. In correlational Jumps model, one jump among companies affects other stock prices of companies obeying correlation coefficients. Therefore equations are given by

$$\begin{pmatrix} \frac{dS_1(t+1)}{S_1(t)} \\ \frac{dS_2(t+1)}{S_2(t)} \\ \vdots \\ \frac{dS_n(t+1)}{S_n(t)} \end{pmatrix} = \begin{pmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_n \end{pmatrix} dt + \begin{pmatrix} \sigma_1 dW_1 t) \\ \sigma_2 dW_2(t) \\ \vdots \\ \sigma_n dW_n(t) \end{pmatrix} + d \begin{pmatrix} J_{1t} \\ J_{2t} \\ \vdots \\ J_{nt} \end{pmatrix}, \tag{9}$$

$$d \begin{pmatrix} J_{1t} \\ J_{2t} \\ \vdots \\ J_{nt} \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{N(t)} (Y_i * correlation_{company_i 1}) \\ \sum_{i=1}^{N(t)} (Y_i * correlation_{company_i 2}) \\ \vdots \\ \sum_{i=1}^{N(t)} (Y_i * correlation_{company_i n}) \end{pmatrix}, \tag{10}$$

$$\begin{aligned} company_i &\sim U_i(a, b), \\ a &\in \{1, 2 \cdots\}, \\ b &\in \{1, 2 \cdots\}, \end{aligned} \tag{11}$$

where $company_i$ is a n_{th} company and U is a discrete uniform distribution. $correlation_{ij}$ is a correlation coefficient from company i to company j.

Correlational Companies Algorithm

In order to calculate correlation coefficients between all pair companies, all paths must be enumerated in a graph structure and a variety of algorithms to find paths are proposed. We propose algorithm for enumerating correlations in a given circulation graph. This program code produces a matrix of correlation coefficients between all pair companies.

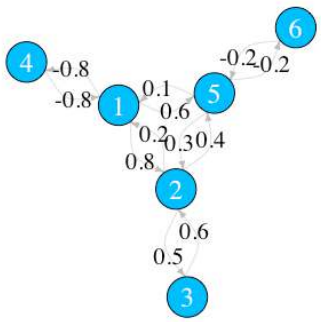

Figure 1: The correlation of companies

This package includes a Perl program in order to calculate the correlations of companies. Please change connect_companies parameters and use like below. output data is data.csv.

```
> perl path.pl
```

Table 1: Result of the correlation coefficients of the companies

	1	2	3	4	5	6
1	1	0.98	0.49	-0.8	0.92	-0.184
2	0.24	1	0.5	-0.192	0.52	-0.104
3	0.144	0.6	1	-0.1152	0.312	-0.0624
4	-0.8	-0.784	-0.392	1	-0.736	0.1472
5	0.16	0.38	0.19	-0.128	1	-0.2
6	-0.032	-0.076	-0.038	-0.0256	-0.2	1

Installation

Jdmbs is available through GitHub (https://github.com/jirotubuyaki/Jdmbs) or CRAN (https://CRAN.R-project.org/package=Jdmbs). If download from Github you can use devtools by the commands:

```
> library(devtools)
> install_github("jirotubuyaki/Jdmbs")
```

Once the packages are installed, it needs to be made accessible to the current R session by the commands:

```
> library(Jdmbs)
```

For online help facilities or the details of a particular command (such as the function normal_bs) you can type:

```
> help(package="Jdmbs")
```

Methods

This package has three methods. This is a normal model for Monte Carlo:

```
> price <- normal_bs(companies, simulation.length=180, monte_carlo=1000,
                     start_price, mu, sigma, K, color
                    )
```

Jump diffusion model for Monte Carlo:

```
> price <- jdm_bs(companies, simulation.length=180, monte_carlo=1000,
                  start_price, mu,sigma, event_times,jump, K, color
                 )
```

This is a proposed method for Monte Carlo. companies_data must be required:

```
> price <- jdm_new_bs(companies_data, companies, simulation.length=180,
                      monte_carlo=1000, start_price, mu,sigma,
                      event_times, jump, K, color
                     )
```

Let arguments be:

- companies_data: a matrix of a correlation coefficient of companies
- companies: an integer of a company number in order to simulate.
- simulation.length: an integer of a time duration of simulation.
- monte_carlo: an integer of an iteration number for monte carlo.
- start_price: a vector of company's initial stock prices.
- mu: a vector of drift parameters of geometric Brownian motion.
- sigma: a vector of volatility parameters of geometric Brownian motion.
- event_times: an integer of how many times jump in unit time.
- jump: a vector of jump parameter.
- K: a vector of option strike prices.
- color: a vector of colors in plot.

Let return be:

- price of a list of (call_price, put_price)

Example

It is a normal model for Monte Carlo:

```
> price <- normal_bs(1, simulation.length=50, monte_carlo=1000,
                     1000, 0.007, 0.03, 1500, "blue"
                    )
```

Jump diffusion for Monte Carlo:

```
> price <- jdm_bs(3 ,simulation.length=100,monte_carlo=80,
                  c(1000,500,500), c(0.002, 0.015, 0.01),
                  c(0.08,0.04,0.06), 3, c(0.1,0.1,0.1),
                  c(1300,600,700), c("red","blue","green")
                  )
```

It is a proposed method for Monte Carlo. companies_data must be required:

```
> price <- jdm_new_bs(matrix(c(0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9),
                      nrow=3, ncol=3),3, simulation.length=100,
                      monte_carlo=80, c(1000,500,500),
                      c(0.002, 0.012, 0.005),c(0.05,0.05,0.06),
                      3,c(0.1,0.1,0.1),
                      c(1500,1000,700),c("red","blue","green")
                     )
```

Conclusions

New algorithms for option prices were described and explained how to use it. This package can produce option prices with related companies. And several improvements are planed. Please send suggestions and report bugs to okadaalgorithm@gmail.com.

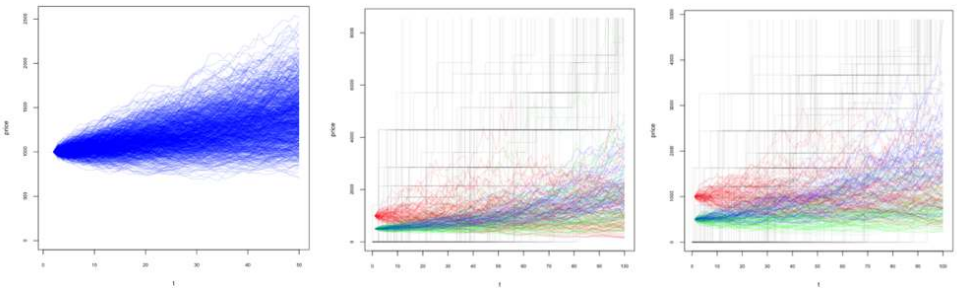

Figure 2: Simulation of geometric Brownian motion

Acknowledgments

This activity would not have been possible without the support of my family and friends. To my family, thank you for much encouragement for me and inspiring me to follow my dreams. I am especially grateful to my parents, who supported me all aspects.

References

[1] Scholes Black and Merton. The pricing of options and corporate liabilities. Journal of Political Economy, 3 Issue 3:637-654, 1973.

[2] Simon S Clift and Peter A Forsyth. Numerical Solution of Two Asset Jump Diffusion Models for Option Valuation. Applied Numerical Mathematics, pages 1-44, 2007.

[3] Steven Shreve. Stochastic Calculus for Finance II: Continuous-Time Models. Springer-Verlag, 2004.