Ch 5. Numerical Methods for Option Pricing

I. Monte Carlo Simulation for Multiple Variables

II. Confidence Interval and Variance Reduction

III. Finite Difference Method (有限差分法)

Appendix A. Solving Systems of Linear Equations

- This chapter formally explores the Monte Carlo simulation. In addition to the univariate Monte Carlo simulation, the multivariate Monte Carlo simulation is also introduced. Furthermore, the confidence interval for Monte Carlo estimates and various kinds of variance reduction techniques are discussed.

- Next, I will introduce another lattice model, called the finite difference method, to compute option prices through solving the partial differential equation of options numerically.

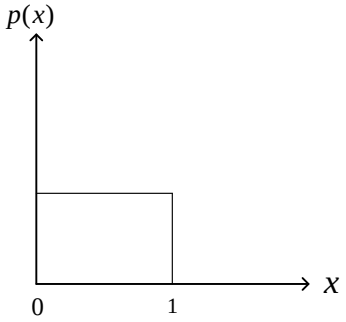
- From Chapter 2, Chapter 4, and this chapter, one can learn four basic methods to compute option prices, including the closed-form solution, the binomial tree model, the Monte Carlo simulation, and the finite difference method.

I. Monte Carlo Simulation for Multiple Variables

- Rand() or Rnd() draws uniformly distributed random samples.

  In the field of the financial engineering, it is often to apply the Monte Carlo simulation technique for pricing derivatives. Therefore, it is important to learn how to draw random samples from different kinds of distributions, e.g., the normal distribution, the binomial distribution, the Poisson distribution, etc. However, in most computer languages, like VBA or C, the random sample generators, usually with the function name Rand() or Rnd(), are to draw random samples from a (standard) uniform distribution as follows.

Figure 5-1 Illustration of the standard uniform distribution

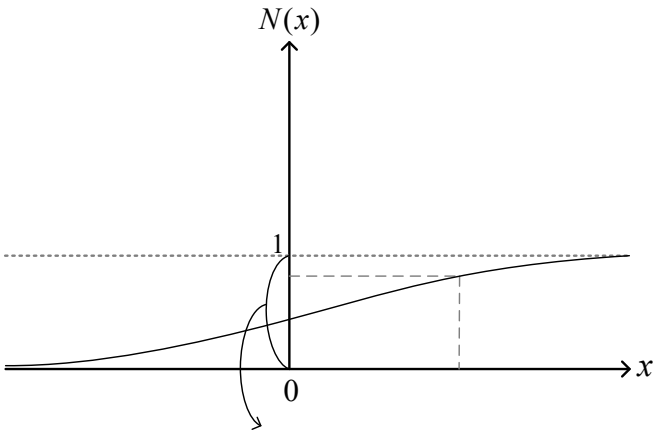


- Since the normal distribution is the most common used distribution, here we only learn how to transform **uniformly distributed random samples** to **normally distributed random samples**. If you are interested in the transformation from uniformly distributed random samples to random samples following other distributions, please refer to Chapter 7 in "Numerical Recipes in C," by Press, Flannery, Teukolsky, and Vetterling.

- Four methods to draw random samples from the standard normal distribution.

  ⊙ Method 1: In Excel, the combination of Norm.S.Inv(Rand()) is employed to transform uniformly distributed random samples to standard normal distributed random samples, where Rand() draws uniform random samples from $[0, 1]$ and Norm.S.Inv() is the function to calculate $N^{-1}()$.

  Figure 5-2



The function Rand() draws a random sample from [0,1]. Then we use the inverse function N^{-1}(rand()) to derive a simulated value of x from the standard normal distribution, where $N(x)$ is the cumulative distribution function of the standard normal distribution.

  ∗ In VBA, call "Application.WorksheetFunction.Norm_S_Inv(Rnd())" to generate normally distributed random samples. Note that the function Rnd() is provided by VBA rather than Excel.

  ⊙ Method 2: Repeatedly draw two samples $\tilde{X}_1$ and $\tilde{X}_2$ from uniform(0,1) until $\tilde{W} = (2\tilde{X}_1 - 1)^2 + (2\tilde{X}_2 - 1)^2 < 1$, then $\tilde{C}(2\tilde{X}_1 - 1)$ and $\tilde{C}(2\tilde{X}_2 - 1)$ are independently standard normal distributed random variables, where $\tilde{C} = \sqrt{\frac{-2(\ln \tilde{W})}{\tilde{W}}}$.

⊙ Method 3: Box-Muller method: Suppose $\tilde{X}_1$ and $\tilde{X}_2$ are drawn from uniform(0,1).

$\tilde{y}_1 = \sqrt{-2\ln \tilde{X}_1} \cdot \cos(2\pi \tilde{X}_2)$

$\tilde{y}_2 = \sqrt{-2\ln \tilde{X}_1} \cdot \sin(2\pi \tilde{X}_2)$

$\Rightarrow \tilde{y}_1$ and $\tilde{y}_2$ are independently standard normal distributed random variables.

∗ It is the most recommended method if the programming language that you use provide neither the function like the NORMSINV() in Excel nor the function to draw directly standard normal distributed random variables.

⊙ Method 4: Draw 12 random samples from uniform(0,1). Calculate the sum of these 12 random samples and then minus 6, and we can derive one random sample from the standard normal distribution. This is also called a dirty method.

∗ It is the least recommended method due to the inefficiency of this method.

- How to draw random samples from a bivariate normal distribution as follows

$$\begin{pmatrix} r_1 \\ r_2 \end{pmatrix} \sim ND\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{pmatrix}\right).$$

Step 1: Draw random samples $\widetilde{z}_1$ and $\widetilde{z}_2$ from $\begin{pmatrix} z_1 \\ z_2 \end{pmatrix} \sim ND\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\right)$.

(Since z_1 and z_2 follow independently standard normal distributions, in practice we draw $\widetilde{z}_1$ and $\widetilde{z}_2$ from $ND(0,1)$ individually.)

Step 2: $\begin{cases} \widetilde{r}_1 = \sigma_1 \cdot \widetilde{z}_1 \\ \widetilde{r}_2 = \sigma_2 \cdot (\widetilde{z}_1 \cdot \rho + \widetilde{z}_2 \cdot \sqrt{1-\rho^2}) \end{cases}$.

- Two methods to draw random samples from a multivariate normal distribution.

  ⊙ Method 1: Cholesky Decomposition Method

  Decompose the covariance matrix $C = A^T A$, where $A = \begin{bmatrix} \alpha & \beta \\ 0 & \phi \end{bmatrix}$

  $$\Rightarrow \begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix} = \begin{bmatrix} \alpha^2 & \alpha\beta \\ \alpha\beta & \beta^2+\phi^2 \end{bmatrix} \Rightarrow \begin{array}{l} \alpha = \sigma_1 \\ \beta = \sigma_2\rho \\ \phi = \sigma_2\sqrt{1-\rho^2} \end{array}$$

  $\Rightarrow [\widetilde{r}_1\ \widetilde{r}_2] = [\widetilde{z}_1\ \widetilde{z}_2] \begin{bmatrix} A \end{bmatrix}$.

  $*$ Based on the above result, it can be inferred that the aforementioned method for the bivariate normal distribution is a special case of this Cholesky decomposition method.

  $*$ Here the bivariate normal distribution is taken as an example. It is straightforward to extend this method to the n-variate case, i.e., $[\widetilde{r}_1\ \widetilde{r}_2\ \cdots\ \widetilde{r}_n] = [\widetilde{z}_1\ \widetilde{z}_2\ \cdots\ \widetilde{z}_n] \begin{bmatrix} A \end{bmatrix}$, where $A^T A$ equals the covariance matrix C for $r_1, r_2, \ldots, r_n$.

  $*$ The detailed algorithm for implementing the Cholesky decomposition method:

  Step 1: $a_{11} = \sqrt{c_{11}}$, $a_{1j} = \frac{c_{1j}}{a_{11}}, j = 2, \cdots, n$

  Step 2: $a_{ii} = \sqrt{c_{ii} - \sum_{k=1}^{i-1} a_{ki}^2}$

  Step 3: $a_{ij} = \frac{1}{a_{ii}} (c_{ij} - \sum_{k=1}^{i-1} a_{ki}a_{kj})$, for $j = i+1, i+2, ..., n$

  repeat Step 2 and Step 3 for $i = 2, 3, ..., n-1$

  Step 4: $a_{nn} = \sqrt{c_{nn} - \sum_{k=1}^{n-1} a_{kn}^2}$

⊙ Method 2: Eigenvalue Decomposition Method (The term eigenvalue is from the German word "Eigenwert," meaning "proper value" (固有值) or "characteristic value" (特徵值). "Eigen" is a prefix means "own" (自我特有的), referring to something characteristic or particular, e.g., a unique feature of a person or an object.)

$$C = E^T \Lambda E, \text{ where } \Lambda = \begin{bmatrix} \lambda_1 & & & 0 \\ & \lambda_2 & & \\ & & \ddots & \\ 0 & & & \lambda_n \end{bmatrix}, \text{ and } E^T E = I$$

λ_i's are the eigenvalues of C and the ith-row of E is the unit-length eigenvector of C corresponding to λ_i.

$$= E^T \Lambda^{\frac{1}{2}} \Lambda^{\frac{1}{2}} E, \text{ where } \Lambda^{\frac{1}{2}} = \begin{bmatrix} \sqrt{\lambda_1} & & & 0 \\ & \sqrt{\lambda_2} & & \\ & & \ddots & \\ 0 & & & \sqrt{\lambda_n} \end{bmatrix}$$

$$= B^T B, \text{ where } B = \Lambda^{\frac{1}{2}} E$$

$$\Rightarrow [\widetilde{r}_1 \ \widetilde{r}_2 \cdots \widetilde{r}_n] = [\widetilde{z}_1 \ \widetilde{z}_2 \cdots \widetilde{z}_n] \begin{bmatrix} B \end{bmatrix}$$

∗ The reason behind the Cholesky and Eigenvalue decomposition method:

$$\begin{aligned} \text{var}(\begin{bmatrix} r_1 \\ r_2 \end{bmatrix}) &= E\left[\begin{bmatrix} r_1 \\ r_2 \end{bmatrix} \begin{bmatrix} r_1 & r_2 \end{bmatrix} \right] \\ &= E\left[A^T \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \begin{bmatrix} z_1 & z_2 \end{bmatrix} A \right] \\ &= A^T E\left[\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \begin{bmatrix} z_1 & z_2 \end{bmatrix} \right] A \\ &= A^T I A = A^T A = C. \end{aligned}$$

∗ From the above demonstration, it is straightforward to infer that any decomposition for the covariance matrix with the form of $C = M^T M$ can be used to generate correlated random samples from the multivariate normal distribution.

II. Confidence Interval and Variance Reduction

- Confidence interval and standard error for the Monte Carlo simulation estimate:

  ⊙ Standard deviation is a simple measure of the variability or dispersion of a population, a data set, or a probability distribution. For N simulated random samples, the standard deviation of this set can be calculated through $\hat{\sigma} = \sqrt{\sum_{i=1}^{N}(x_i - \bar{x})^2/N}$.

  ⊙ Standard error s_E: The standard error of an estimation method is the **standard deviation of the sampling distribution** associated with the estimation method.

  ⊙ A sampling distribution is the probability distribution, under repeated sampling of the population, of a given estimation. For example, consider a very large normal distributed population. Assume we repeatedly draw sample sets with a given size from the population and calculate the sample mean for each sample set. A different sample set yields a different sample mean. The distribution of these means is the "sampling distribution of the sample mean" (for the given sample size).

  ⊙ For the estimates of the sample variance, skewness, kurtosis, etc., it is also possible to derive the "sampling distributions" of these estimates.

  ⊙ For the Monte Carlo simulation, since it is a method for estimating the mean, we can compute the standard error for the mean results based on the Monte Carlo simulation.

  ⊙ Method 1: The standard error of the mean is usually estimated by the sample standard deviation divided by the square root of the sample size: $s_E = \frac{\hat{\sigma}}{\sqrt{N}}$, where $\hat{\sigma}$ is the sample standard deviation, and N is the sample size.

  ⊙ Method 2: Since the Monte Carlo simulation method is a process to estimate the mean, we can repeat the Monte Carlo simulation method for several times, e.g., 20 to 30 times, to obtain a stable sampling distribution for the estimation of the mean. As a consequence, the standard deviation of these repetitions is the standard error for the Monte Carlo simulation method.

⊙ **Confidence interval comparison:**

Method 1: $s_{E,1} = \frac{\hat{\sigma}}{\sqrt{N}}$ for $N = 10000$ vs. Method 2: $s_{E,2}$ = standard deviation of M repetitions and each is with $N = 10000$ random samples:

The 95% confidence interval for the estimation based on $N = 10000$ random samples:

For Method 1, the 95% confidence interval is

[mean of 10000 random samples$-2 \times s_{E,1}$, mean of 10000 random samples$+2 \times s_{E,1}$].

For Method 2, the 95% confidence interval is

[mean of M repetitions$-2 \times s_{E,2}$, mean of M repetitions$+2 \times s_{E,2}$].

∗ According to the central limit theorem, the sampling distribution of the mean is asymptotically normal with $Z_{2.5\%} = -1.96$ and $Z_{97.5\%} = 1.96$. The use of $\pm 2 \times s_E$ is approximately correct and common in practice.

∗ The 95% confidence intervals generated from both methods are with a similar size (see "Standard Error.xlsx"). However, Method 2 provides a more accurate mean, so the 95% confidence interval of Method 2 is preferred. (Of course, Method 2 needs more computational time since it repeats the Monte Carlo simulation method based on $N = 10000$ random samples for M times.)

∗ Method 2 is more intuitive:

To examine the confidence interval is to estimate how accurate for your method based on 10000 random samples. Computing the s.d. of the sampling distribution, i.e., Method 2, is the most intuitive way to achieve the above goal.

∗ Sometimes, Method 1 does not work. See "Effect of Antithetic.xls" after understanding the antithetic variate approach. The possible reason may be that the random samples are no more independent after using the antithetic variate approach.

- Variance-reduction technique: approaches which can narrow the confidence intervals of the Monte Carlo simulation.

  (i) Antithetic variate approach (反向變異法): satisfying the zero mean and the symmetric feature of the standard normal distribution

  $$\begin{array}{c} \widetilde{z}_1,\ \widetilde{z}_2,\ \cdots,\ \widetilde{z}_{\frac{N}{2}},\ \widetilde{z}_{\frac{N}{2}+1},\ \cdots,\ \widetilde{z}_N \\ \qquad\qquad\qquad\qquad \| \qquad\quad \| \\ \qquad\qquad\qquad -\widetilde{z}_1 \cdots - \widetilde{z}_{\frac{N}{2}} \end{array}$$

  (ii) Moment Matching: matching the first two moments of the standard normal distribution

  Draw random samples $\underbrace{\widetilde{z}_1, \widetilde{z}_2, \cdots, \widetilde{z}_N}$ first.

  the mean equals m (e.g., -0.003) and the s.d. equals s (e.g., 1.046)

  Define $\widetilde{y}_i = \frac{\widetilde{z}_i - m}{s}$

  $\Rightarrow \underbrace{\widetilde{y}_1, \widetilde{y}_2, \cdots, \widetilde{y}_N}$

  the mean equals 0 and the s.d. equals 1

  (iii) Control variates

  Suppose the goal is to estimate $E[X]$. Find a random variable Y such that $E[Y] = \mu$. Define a new random variable $W \equiv X + \beta(Y - \mu)$, and consider to estimate $E[W]$.
  $\Rightarrow E[W] = E[X]$, and $\text{var}(W) = \text{var}(X) + \beta^2 \text{var}(Y) + 2\beta \text{cov}(X, Y)$

  If $\beta^2 \text{var}(Y) + 2\beta \text{cov}(X, Y) < 0$, then $\text{var}(W) < \text{var}(X)$. Under this criterion, we can estimate $E[W]$ instead of $E[X]$, that gives us the same expectation value but a smaller variance and thus a smaller confidence interval.

  > The equation of $\beta^2 \text{var}(Y) + 2\beta \text{cov}(X, Y)$ is minimized when $\beta = \beta^* = \frac{-\text{cov}(X,Y)}{\text{var}(Y)}$, and the minimized value of this equation becomes $\frac{-\text{cov}(X,Y)^2}{\text{var}(Y)}$
  > $\Rightarrow \text{var}(W) = \text{var}(X) - \frac{\text{cov}(X,Y)^2}{\text{var}(Y)} = (1 - \rho_{XY}^2)\text{var}(X)$
  > $\Rightarrow$ when ρ_{XY} is closer to ± 1, we can derive a smaller $\text{var}(W)$

  $\odot$ The first difficulty is to find the expectation of the random variable Y. (Note that we need the true value for $E[Y]$, so we cannot employ the Monte carlo simulation method to estimate $E[Y]$ here.)

⊙ The second difficulty is to decide β. In practice, because the slope estimation of a linear regression is $\frac{\text{cov}(X,Y)}{\text{var}(Y)}$ theoretically, β is commonly determined to be the negative of the slope-coefficient result of linearly regressing randomly-generated X over randomly-generated Y. However, the β generated by this method is dependent on the drawn samples of X and Y and thus itself is random such that the stability of the estimations of $E[W]$ and $\text{var}(W)$ may be affected.

⊙ Kemna and Vorst (1990), "A pricing method for options based on average asset values," *Journal of Banking and Finance* 14, pp. 113–129.

1. Suppose X is $e^{-rT}\max(\overline{S_A} - K, 0)$ and Y is $e^{-rT}\max(\overline{S_G} - K, 0)$, where $\overline{S_A}$ is the arithmetic average and $\overline{S_G}$ is the geometric average along the stock path.

2. Since $\overline{S_A}$ and $\overline{S_G}$ are highly positive correlated, we set $\beta = -1$ in the control variates method.

3. In addition, there is a Black-Scholes-like pricing formula for geometric average options (introduced in Ch.10), and we can employ this formula to derive $E[Y] = \mu$.

(iv) Empirical Martingale Simulation (EMS) (平賭過程配適模擬法): studying the relationship among simulated stock paths (Duan and Simonato (1998))

In the risk-neutral world, the discounted values of simulated asset prices should be a martingale process theoretically. However, it is not always the case in practice. Moreover, the violation of the martingale characteristic has the propagation feature, i.e., if the simulated asset prices do not follow the martingale characteristic at some time point t, the error will accumulate and thus the degree of violation of the martingale characteristic will becomes more serious after t.

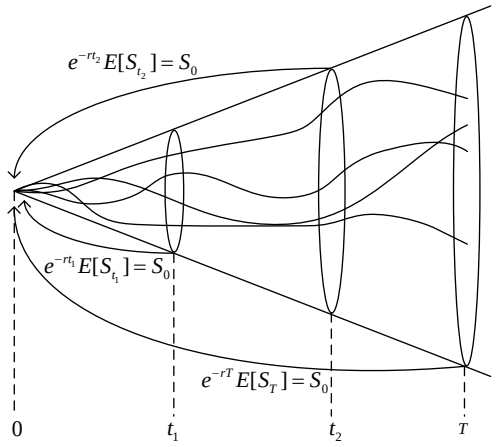
⊙ Martingale process (a series of fair games)
A stochastic process $X(t), t \geq 0$ is a martingale if $E[|X(t)|] < \infty$ for $t \geq 0$ and $E[X(t) \mid X(u), 0 \leq u \leq s] = X(s)$.

∗ By applying the tower rule of iterated conditional expectations, we can further derive $E[X(t)] = X(0)$.

⊙ Consider $X = e^{-rt}S(t)$, and the idea of the EMS method is to adjust the simulated stock prices such that $E[X(t)] = X(0)$, which is equivalent to $E[e^{-rt}S(t)] = e^{-r0}S(0) = S(0)$.

⊙ Like the original Monte Carlo simulation, in the EMS method, the arithmetic average of stock prices $S(t)$ of all simulated paths is used to estimate $E[S(t)]$. Thus, the EMS is to ensure that e^{-rt}[arithmetic average of $S(t)$] $= S(0)$ for each time point t.

Figure 5-3 Notion of the EMS method



$\odot$ Given N simulated stock price paths, the EMS method adjusts stock prices as follows, where $\hat{S}$ and S^* represent the stock prices before and after the adjustment.

$$S_i^*(t_j,N)=\frac{S_0}{\frac{1}{N}\sum_{i=1}^{N}e^{-rt_j}\hat{S}_i(t_j,N)}\hat{S}_i(t_j,N)$$

$\Big\|$ The subscript i denotes the i-th stock price path, and t_j denotes the j-th time point.

$$\Rightarrow S_0^*(t,N)=\frac{1}{N}\sum_{i=1}^{N}e^{-rt}S_i^*(t,N)\ \text{(for any time point } t)$$

$\Big\|$ The subscript 0 means to compute the average of the present value of the stock prices over the N simulated paths.

$$=\frac{1}{N}\sum_{i=1}^{N}e^{-rt}\frac{S_0}{\frac{1}{N}\sum_{i=1}^{N}e^{-rt}\hat{S}_i(t,N)}\hat{S}_i(t,N)$$
$$=S_0$$

$*$ If the simulated samples for the stock price process are perfect, i.e., they satisfy the martingale characteristic already such that $\frac{1}{N}\sum_{i=1}^{N}e^{-rt_j}\hat{S}_i(t_j,N)=E_0[e^{-rt_j}\hat{S}(t_j)]=S_0$, the above adjustment equation returns $S_i^*(t_j,N)=\hat{S}_i(t_j,N)$. Under this ideal condition, it is not necessary to adjust the original simulated samples for the stock price process. However, it is almost impossible to meet this perfect condition.

$*$ In fact, Duan and Simonato (1998) suggest a 4-step method to compute $S_i^*(t_j,N)$. The above method is identical to the 4-step method in Duan and Simonato (1998), but is more efficient and intuitive.

- "Common random samples" method: If you intend to calculate the option value under different variance-reduction techniques (or different parameter values), it is not suited to create different random samples for each calculation, because you cannot distinguish the difference between any two calculations being from the effect of different variance-reduction techniques (or different parameter values) or from the effect of using different random samples. The method of common random samples means to employ the same set of random samples to calculate the option prices for different variance-reduction techniques (or different parameter values) to avoid the above problem.

III. Finite Difference Method (有限差分法)

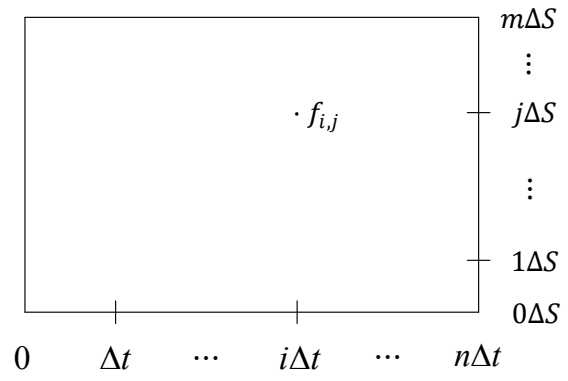
- Main idea:

  The finite difference method (FDM) is initially proposed to solve partial differential equations numerically, and its main idea is to use finite difference (差分) to approximate the partial differentiation (微分).

- General setting:

  Here we apply the FDM to solve partial differential equations for derivatives mentioned in Chapters 2 and 4, i.e., to solve $\frac{\partial f}{\partial t} + (r-q)S\frac{\partial f}{\partial S} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 f}{\partial S^2} = rf$.

Figure 5-4 Discretized stock-time plane for the FDM



∗ The stock-time (or S-t) plane is discretized into a grid of nodes. The examined time points are $0, \Delta t, 2\Delta t, \ldots, n\Delta t$, and the examined stock prices are $0, \Delta S, 2\Delta S, \ldots, m\Delta S$. The numerical solution of f means to find the values for all $f_{i,j}$, where i and j are the indexes for the time and the stock price, respectively. If the grid size is small enough, it is equivalent to derive a closed-form solution because given any values of S and t, one can identify a corresponding node and thus obtain its value of f.

$*$ Similar to the binomial tree model, the FDM proceeds from the maturity backward to $t = 0$. Therefore, the first step of the FDM is to determine the payoff at maturity. Taking the put option for example, $f_{n,j} = \max(K - j\Delta S, 0)$, for $j = 0, 1, ..., m$.

$*$ Next, the implicit and explicit FDMs are introduced.

(i) Implicit FDM:

For node(i, j),

$\frac{\partial f}{\partial S} = \frac{f_{i,j+1} - f_{i,j-1}}{2\Delta S}$,

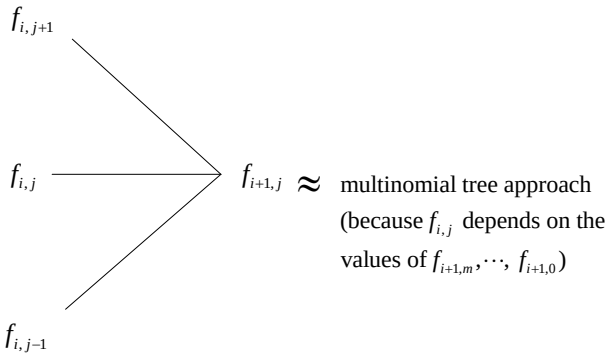
$\frac{\partial^2 f}{\partial S^2} = (\frac{f_{i,j+1} - f_{i,j}}{\Delta S} - \frac{f_{i,j} - f_{i,j-1}}{\Delta S})/\Delta S = \frac{f_{i,j+1} - 2f_{i,j} + f_{i,j-1}}{\Delta S^2}$,

$\frac{\partial f}{\partial t} = \frac{f_{i+1,j} - f_{i,j}}{\Delta t}$.

$\Rightarrow \frac{f_{i+1,j} - f_{i,j}}{\Delta t} + (r - q)(j\Delta S)\frac{f_{i,j+1} - f_{i,j-1}}{2\Delta S} + \frac{1}{2}\sigma^2(j\Delta S)^2\frac{f_{i,j+1} - 2f_{i,j} + f_{i,j-1}}{\Delta S^2} = r f_{i,j}$,

for $j = 1, 2, \ldots, m - 1$ and $i = 0, 1, \ldots, n - 1$.

Figure 5-5



$$\Rightarrow a_j f_{i,j-1} + b_j f_{i,j} + c_j f_{i,j+1} = f_{i+1,j}, \text{ and } \begin{cases} a_j = \frac{r-q}{2} j\Delta t - \frac{1}{2}\sigma^2 j^2 \Delta t \\ b_j = 1 + \sigma^2 j^2 \Delta t + r\Delta t \\ c_j = -\frac{r-q}{2} j\Delta t - \frac{1}{2}\sigma^2 j^2 \Delta t \end{cases}.$$

$\odot$ Note that for each pair of successive time points $i\Delta t$ and $(i + 1)\Delta t$, there are $(m - 1)$ equations and able to be expressed as a linear system in the following matrix form.

$$\begin{matrix} \\ m-1 \\ \text{equations} \\ \\ m+1 \\ \text{unknowns} \end{matrix} \begin{bmatrix} c_{m-1} & b_{m-1} & a_{m-1} & 0 & & & \cdots & & 0 \\ & & & \vdots & & & & & \\ & & & & & & & & \\ 0 & \cdots & 0 & c_j & b_j & a_j & 0 & \cdots & 0 \\ & & & \vdots & & & & & \\ 0 & & \cdots & & & 0 & c_1 & b_1 & a_1 \end{bmatrix} \begin{bmatrix} f_{i,m} \\ \\ \vdots \\ f_{i,j+1} \\ \\ f_{i,j} \\ \\ f_{i,j-1} \\ \vdots \\ \\ f_{i,0} \end{bmatrix} = \begin{bmatrix} f_{i+1,m-1} \\ \\ \vdots \\ f_{i+1,j} \\ \\ \vdots \\ \\ f_{i+1,1} \end{bmatrix}.$$

$\odot$ However, since there is only $m-1$ equations but we need to solve $m+1$ values of $f_{i,j}$, for $j = 0, 1, ..., m$, at $i\Delta t$, we further assume the values of the uppermost nodes $f_{0,m}, ..., f_{n-1,m}$ and the lowermost nodes $f_{0,0}, ..., f_{n-1,0}$ to satisfy some boundary conditions. Taking the call option as an example, $f_{0,m} = \cdots = f_{n-1,m} = m\Delta S - K$ (assuming those nodes to be extremely in the money), and $f_{0,0} = \cdots = f_{n-1,0} = 0$ (assuming those nodes to be extremely out of the money). As a consequence, the system of linear equations for any two successive time points becomes

$$\begin{matrix} \\ m-1 \\ \text{equations} \\ \\ m-1 \\ \text{unknowns} \end{matrix} \begin{bmatrix} b_{m-1} & a_{m-1} & 0 & & & \cdots & & 0 \\ & & \vdots & & & & & \\ & & & & & & & \\ 0 & \cdots & c_j & b_j & a_j & 0 & \cdots & 0 \\ & & \vdots & & & & & \\ 0 & & \cdots & & & 0 & c_1 & b_1 \end{bmatrix} \begin{bmatrix} f_{i,m-1} \\ \\ \vdots \\ f_{i,j} \\ \\ \vdots \\ \\ f_{i,1} \end{bmatrix} = \begin{bmatrix} f_{i+1,m-1} - c_{m-1} f_{i,m} \\ \\ \vdots \\ f_{i+1,j} \\ \\ \vdots \\ \\ f_{i+1,1} - a_1 f_{i,0} \end{bmatrix}.$$

$\Rightarrow A_{(m-1)\times(m-1)} \mathrm{x}_{(m-1)\times 1} = \mathrm{b}_{(m-1)\times 1}$

$\Rightarrow \mathrm{x} = A^{-1}\mathrm{b}$

$*$ Note that the matrix A is independent of time, i.e., independent of i. Therefore, it is suggested to solve A^{-1} only once and store the result of A^{-1}, which can then be used for every iteration of the backward induction.

$*$ In VBA, one can employ the functions of Application.WorksheetFunction.Mmult() and Application.WorksheetFunction.Minverse() to calculate the matrix multiplication and inverse, respectively.

$\odot$ Backward induction algorithm for the (implicit) FDM:

1. By setting $f_{n,0}, \ldots, f_{n,m}$ to be the payoffs at maturity of the examined derivatives, e.g., $f_{n,j} = \max(j\Delta S - K, 0)$ if a call option is considered.

2. Solve the above system of linear equations for any two successive time points backward from $n\Delta t$ to $0\Delta t$. If the American-style option is considered, it needs to check $f_{i,j} = \max(f_{i,j}$, exercise value at node $(i, j))$.

3. The value of $f(S_0, 0)$ is the option value today.

$*$ It is worth noting that the FDM generates not only the option value today, i.e, $f(S_0, 0)$, but also the values of $f(S, t)$ for any node on the S-t plane.

$*$ The implicit FDM is more robust than the explicit FDM introduced next, i.e., the implicit FDM does not need an extremely small Δt for obtaining convergent results.

(ii) Explicit FDM (special assumption: $\frac{\partial f}{\partial S}$ and $\frac{\partial^2 f}{\partial S^2}$ at node(i, j) are assumed to be the same as those at node$(i + 1, j)$. Thus, it can be inferred that this method needs a smaller Δt to minimize the error caused by the special assumption.):

For node(i, j),

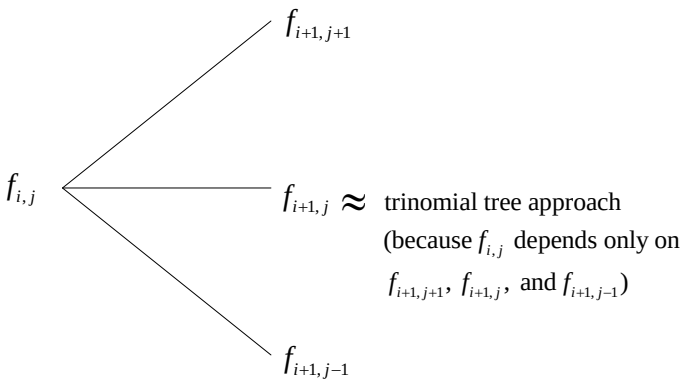
$\frac{\partial f}{\partial S} = \frac{f_{i+1,j+1} - f_{i+1,j-1}}{2\Delta S}$,

$\frac{\partial^2 f}{\partial S^2} = \frac{f_{i+1,j+1} - 2f_{i+1,j} + f_{i+1,j-1}}{\Delta S^2}$,

$\frac{\partial f}{\partial t} = \frac{f_{i+1,j} - f_{i,j}}{\Delta t}$.

$\Rightarrow \frac{f_{i+1,j} - f_{i,j}}{\Delta t} + (r - q)(j\Delta S)\frac{f_{i+1,j+1} - f_{i+1,j-1}}{2\Delta S} + \frac{1}{2}\sigma^2(j\Delta S)^2\frac{f_{i+1,j+1} - 2f_{i+1,j} + f_{i+1,j-1}}{\Delta S^2} = r f_{i,j}$,
for $j = 1, 2, \cdots, m - 1$ and $i = 0, 1, \cdots, n - 1$.

Figure 5-6



$\Rightarrow f_{i,j} = a_j^* f_{i+1,j-1} + b_j^* f_{i+1,j} + c_j^* f_{i+1,j+1}$, and $\begin{cases} a_j^* = \frac{1}{1+r\Delta t}(-\frac{1}{2}(r-q)j\Delta t + \frac{1}{2}\sigma^2 j^2 \Delta t) \\ b_j^* = \frac{1}{1+r\Delta t}(1 - \sigma^2 j^2 \Delta t) \\ c_j^* = \frac{1}{1+r\Delta t}(\frac{1}{2}(r-q)j\Delta t + \frac{1}{2}\sigma^2 j^2 \Delta t) \end{cases}$.

$\odot$ The above $(m-1)$ equations can be expressed in the following matrix form.

$$\begin{bmatrix} c_{m-1}^* & b_{m-1}^* & a_{m-1}^* & 0 & & & \cdots & & 0 \\ & & & \vdots & & & & & \\ & & & & & & & & \\ 0 & \cdots & 0 & c_j^* & b_j^* & a_j^* & 0 & \cdots & 0 \\ & & & \vdots & & & & & \\ 0 & & \cdots & & & 0 & c_1^* & b_1^* & a_1^* \end{bmatrix} \begin{bmatrix} f_{i+1,m} \\ \\ \vdots \\ f_{i+1,j+1} \\ \\ f_{i+1,j} \\ \\ f_{i+1,j-1} \\ \vdots \\ \\ f_{i+1,0} \end{bmatrix} = \begin{bmatrix} f_{i,m-1} \\ \\ \vdots \\ f_{i,j} \\ \\ \vdots \\ \\ f_{i,1} \end{bmatrix}.$$

$\Rightarrow A_{(m-1)\times(m+1)} \mathrm{b}_{(m+1)\times 1} = \mathrm{x}_{(m-1)\times 1}$

$*$ Note that in the above system, the $m-1$ unknowns are $f_{i,1}, \ldots, f_{i,m-1}$. Therefore, to solve these unknowns, a matrix multiplication (rather than solving the system) is enough.

$*$ However, we still need the boundary conditions for the uppermost nodes $f_{0,m}, \ldots, f_{n-1,m}$ and the lowermost nodes $f_{0,0}, \ldots, f_{n-1,0}$ in the backward induction process.

$*$ The explicit FDM is similar to the trinomial tree model and thus one could interpret the coefficients, a_j^*, b_j^*, and c_j^*, as the probabilities corresponding to each branch times the discount factor $\frac{1}{1+r\Delta t}$.

$*$ Since the explicit FDM is similar to the trinomial tree model, it is not appropriate that the coefficients, a_j^*, b_j^*, and c_j^*, are smaller than 0. To avoid this problem, a small enough value of Δt can be determined in advance by ensuring all a_j^*, b_j^*, and c_j^* being nonnegative first.

- For both implicit and explicit FDMs, the value of $f_{i,j}$ sometimes may be negative (see the example in Table 20.5 in Hull (2011)). Since the option provides the right to the holder, the value of option should be positive. So, we can set $f_{i,j} = 0$ when $f_{i,j} < 0$ to minimize this type of pricing error.

- The convergence criterion of the finite difference method depends on the relationship between Δt and ΔS. It is known that for a simpler PDE of $\frac{\partial f}{\partial t} = \alpha^2 \frac{\partial^2 f}{\partial S^2}$, the finite difference method converges as long as $\alpha^2 \frac{\Delta t}{(\Delta S)^2} \leq \frac{1}{2}$. That means when ΔS is small, Δt should be much smaller to ensure the convergence of the solution.

- For our option pricing PDE, $\alpha^2 = \frac{1}{2}\sigma^2 S^2$, and we have two more terms, $rS\frac{\partial f}{\partial S}$ and rf. Although the option pricing PDE does not follow the above simpler form, it is still necessary to maintain a proper relationship between Δt and ΔS to make sure the convergence of the FDM.

Appendix A. Solving Systems of Linear Equations

- Gaussian elimination: solving a system of linear eqautions

$$\begin{array}{ll} \mathrm{E}_1\colon & x_1 - x_2 + 2x_3 - x_4 = -8 \\ \mathrm{E}_2\colon & 2x_1 - 2x_2 + 3x_3 - 3x_4 = -20 \\ \mathrm{E}_3\colon & x_1 + x_2 + x_3 = -2 \\ \mathrm{E}_4\colon & x_1 - x_2 + 4x_3 + 3x_4 = 4 \end{array} \Rightarrow \underbrace{\begin{bmatrix} 1 & -1 & 2 & -1 \\ 2 & -2 & 3 & -3 \\ 1 & 1 & 1 & 0 \\ 1 & -1 & 4 & 3 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}}_{\mathrm{x}} = \underbrace{\begin{bmatrix} -8 \\ -20 \\ -2 \\ 4 \end{bmatrix}}_{\mathrm{b}}$$

$$\left[\begin{array}{cccc:c} 1 & -1 & 2 & -1 & -8 \\ 2 & -2 & 3 & -3 & -20 \\ 1 & 1 & 1 & 0 & -2 \\ 1 & -1 & 4 & 3 & 4 \end{array}\right]$$

$$\begin{array}{rcl} & \mathrm{E}_1 \times -2 + \mathrm{E}_2 & \rightarrow \mathrm{E}_2 \\ \Downarrow & \mathrm{E}_1 \times -1 + \mathrm{E}_3 & \rightarrow \mathrm{E}_3 \\ & \mathrm{E}_1 \times -1 + \mathrm{E}_4 & \rightarrow \mathrm{E}_4 \end{array}$$

$$\left[\begin{array}{cccc:c} 1 & -1 & 2 & -1 & -8 \\ 0 & 0 & -1 & -1 & -4 \\ 0 & 2 & -1 & 1 & 6 \\ 0 & 0 & 2 & 4 & 12 \end{array}\right]$$

$\uparrow$

Find a nonzero pivoting element in the second column. If there is no nonzero element for a_{22}, a_{32}, and a_{42}, it could be the case of no solution or infinitely many solutions. (It is better to choose the largest $|a_{ij}|$ to be the pivoting element. That choice can reduce the round off error.)

$$\Downarrow \mathrm{E}_2 \rightleftarrows \mathrm{E}_3$$

$$\left[\begin{array}{cccc:c} 1 & -1 & 2 & -1 & -8 \\ 0 & 2 & -1 & 1 & 6 \\ 0 & 0 & -1 & -1 & -4 \\ 0 & 0 & 2 & 4 & 12 \end{array}\right]$$

$$\Downarrow \mathrm{E}_3 \times 2 + \mathrm{E}_4 \rightarrow \mathrm{E}_4$$

$$\left[\begin{array}{cccc:c} 1 & -1 & 2 & -1 & -8 \\ 0 & 2 & -1 & 1 & 6 \\ 0 & 0 & -1 & -1 & -4 \\ 0 & 0 & 0 & 2 & 4 \end{array}\right] \underset{\text{back subsitution}}{\Longrightarrow} \begin{array}{l} x_4 = 2 \\ x_3 = 2 \\ x_2 = 3 \\ x_1 = -7 \end{array}$$

- Gauss-Jordan elimination (This method can solve both $A\mathrm{x} = \mathrm{b}$ and A^{-1})

$$\Rightarrow \left[\begin{array}{ccc:ccc} 1 & 2 & -1 & 1 & 0 & 0 \\ 2 & 1 & 0 & 0 & 1 & 0 \\ -1 & 1 & 2 & 0 & 0 & 1 \end{array}\right]$$

$$\begin{array}{r} \mathrm{E}_1 \times -2 + \mathrm{E}_2 \to \mathrm{E}_2 \\ \mathrm{E}_1 \times \ \ 1 + \mathrm{E}_3 \to \mathrm{E}_3 \\ \Longrightarrow \end{array} \left[\begin{array}{ccc:ccc} 1 & 2 & -1 & 1 & 0 & 0 \\ 0 & -3 & 2 & -2 & 1 & 0 \\ 0 & -3 & 1 & 1 & 0 & 1 \end{array}\right]$$

$$\begin{array}{r} \mathrm{E}_2 \times -\frac{1}{3} \to \mathrm{E}_2 \\ \Longrightarrow \end{array} \left[\begin{array}{ccc:ccc} 1 & 2 & -1 & 1 & 0 & 0 \\ 0 & 1 & -\frac{2}{3} & \frac{2}{3} & -\frac{1}{3} & 0 \\ 0 & 3 & 1 & 1 & 0 & 1 \end{array}\right]$$

$$\begin{array}{r} \mathrm{E}_2 \times -2 + \mathrm{E}_1 \to \mathrm{E}_1 \\ \mathrm{E}_2 \times -3 + \mathrm{E}_3 \to \mathrm{E}_3 \\ \Longrightarrow \end{array} \left[\begin{array}{ccc:ccc} 1 & 0 & \frac{1}{3} & \frac{1}{3} & \frac{2}{3} & 0 \\ 0 & 1 & -\frac{2}{3} & \frac{2}{3} & -\frac{1}{3} & 0 \\ 0 & 0 & 3 & -1 & 1 & 1 \end{array}\right]$$

$$\begin{array}{r} \mathrm{E}_3 \times \frac{1}{3} \to \mathrm{E}_3 \\ \Longrightarrow \end{array} \left[\begin{array}{ccc:ccc} 1 & 0 & \frac{1}{3} & \frac{1}{3} & \frac{2}{3} & 0 \\ 0 & 1 & -\frac{2}{3} & \frac{2}{3} & -\frac{1}{3} & 0 \\ 0 & 0 & 1 & -\frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{array}\right]$$

$$\begin{array}{r} \mathrm{E}_3 \times -\frac{1}{3} + \mathrm{E}_1 \to \mathrm{E}_1 \\ \mathrm{E}_3 \times -\frac{2}{3} + \mathrm{E}_2 \to \mathrm{E}_2 \\ \Longrightarrow \end{array} \left[\begin{array}{ccc:ccc} 1 & 0 & 0 & -\frac{2}{9} & \frac{5}{9} & -\frac{1}{9} \\ 0 & 1 & 0 & \frac{4}{9} & -\frac{1}{9} & \frac{2}{9} \\ 0 & 0 & 1 & -\frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{array}\right]$$

$$\| \qquad\qquad A^{-1} = \left[\begin{array}{ccc} -\frac{2}{9} & \frac{5}{9} & -\frac{1}{9} \\ \frac{4}{9} & -\frac{1}{9} & \frac{2}{9} \\ -\frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{array}\right]$$

$\odot$ For solving $A\mathrm{x} = \mathrm{b}$, in addition to the performing the Gaussian (or Gauss-Jordon) elimination, we can calculate A^{-1} by the Gauss-Jordon elimination first and then apply $\mathrm{x} = A^{-1}\mathrm{b}$ to obtain the solution.