

Monte Carlo Methods for Derivative Security Pricing

- The main references for the material covered here are:
 1. Darrell Duffie (1996): *Dynamic Asset Pricing Theory*.
 2. Robert C. Merton (1992): *Continuous-Time Finance*.
 3. Dixit and Pindyck (1994): *Investment under Uncertainty*.
 4. Boyle, Broadie, and Glasserman (1997): “Monte Carlo methods for security pricing,” *Journal of Economic Dynamics and Control*.
 5. Giorgio Pauletto (2000): “Parallel Monte Carlo Methods for Derivative Security Pricing,” manuscript, University of Geneva.
 6. Tan and Boyle (2000): “Applications of randomized low discrepancy sequences to the valuation of complex securities,” *Journal of Economic Dynamics and Control*.
- We will concentrate in this lecture on valuing options, but similar techniques can be used to value more sophisticated securities, and also estimate sensitivities, to do risk analysis, value investments, etc.
- An option is also called a derivative security, and is a security the payoff of which depends on one or several other underlying securities.
- The prices of these underlying securities are the state variables in the problem, and are often modelled as continuous-time stochastic processes.
- Assuming that not arbitrage exists (there is no self-financing trading strategy) one can show that the price of such an option is the discounted expected value of the payoffs under a risk neutrality measure (this does not mean that agents are risk neutral, but rather that risk-neutrality is without loss of generality for purposes of pricing derivative securities).
- In this framework, pricing an option that can be written as an expectation of a random variable lends itself naturally to a numerical procedure that estimates this expected value through simulation.
- We will focus on the pricing of a *European Call* on a stock. This option gives the owner the right, but not the obligation, to buy the stock at a given exercise price K on a given expiration date T . The difference with an *American Call* is that the latter allows the owner to exercise at any point before T .

- The option price process Y is as yet unknown except for the fact that $Y_T = (S_T - K)^+ \equiv \max(S_T - K, 0)$, which follows from the fact that the option is rationally exercised if and only if $S_T > K$.
- S_T is the price process of the underlying security (stock). As we explained before it usually follows a continuous time stochastic process. The most common ones are *Brownian motions* or *Wiener processes*.
- A Wiener process has three important properties:
 1. It is a Markov process. This means that the probability distribution for all future values of the process depends only on its current value, and is unaffected by past values of the process or by any other current information. This implies that only current information is useful for forecasting the future path of the process. Stock prices are often modelled as Markov processes, on the grounds that public information is quickly incorporated in the current price of the stock, so that the past pattern of prices has no forecasting value. This is called the weak form of market efficiency.
 2. This process has independent increments. This means that the probability distribution for the change in the process over any time interval is independent of any other time interval. We can think of these processes as the continuous-time version of a random walk.
 3. Changes in the process over any finite interval of time are normally distributed, with a variance that increases linearly with the time interval.
- These properties are not very restrictive. Notice that the Wiener process is non-stationary.
- At times we might want to work with two or more Wiener processes, and we will be interested in their covariances. Suppose that $z_1(t)$ and $z_2(t)$ are Wiener processes. Then we can write $E(dz_1, dz_2) = \rho_{12} dt$, where ρ_{12} is the coefficient of correlation between the two processes. Because a Wiener process has a variance and standard deviation per unit of time of 1 (this means that $E[(dz)^2]/dt = 1$), then ρ_{12} is also the covariance per unit of time for the two processes.

- Sometimes it can be useful to characterize a generalization of the Wiener process and the Brownian motion with a drift, where

$$dx = a(x,t)dt + b(x,t)dz, \quad (1)$$

where dz is the increment of a Wiener process, and $a(x,t)$ and $b(x,t)$ are known (nonrandom) functions. Notice here that the drift and variance coefficients are functions of the current state and time.

- The continuous time stochastic process above is called an *Ito* process.
- A related process is the geometric Brownian motion. Consider a security with price process

$$S_t = x \exp[\alpha t + \sigma B_t], \text{ for } t \geq 0 \quad (2)$$

where $x > 0$, α , and σ are constants. This process is often called lognormal, because for any t , $\log(S_t) = \log(x) + \alpha t + \sigma B_t$ is normally distributed.

- From the properties of this type of processes it follows that S is an Ito process and that

$$dS_t = \mu S_t dt + \sigma S_t dB_t; S_0 = x, \quad (3)$$

where $\mu = \alpha + \sigma^2/2$. Again from the properties of these processes, at any time t , S has the conditional expected rate of change μS_t and conditional rate of change of variance $\sigma^2 S_t^2$. So that, per dollar invested in this security at time t , one may think of μ as the *instantaneous* expected rate of return, and σ as the *instantaneous* standard deviation of the rate of return.

- The coefficient σ is also known as the *volatility* of S . The geometric Brownian motion is a natural two-parameter model of a security-price process because of these simple interpretations of μ and σ .
- For completeness we want to define another security, a *bond*, with the price process β defined by

$$\beta_t = \beta_0 e^{rt}, \text{ for } t \geq 0, \quad (4)$$

for some constants $\beta_0 > 0$ and r .

- The obvious interpretation of r is the continually compounding interest rate, that is, the exponential rate at which riskless deposits accumulate with interest. β is trivially an Ito process

$$d\beta_t = r \beta_t dt. \quad (5)$$

- With all this we can write the *Black-Scholes* formula: If there is not arbitrage, then, for all $t < T$, $Y_t = C(S_t, t)$ where

$$C(x, t) = x \Phi(z) - e^{-r(T-t)} K \Phi(z - \sigma\sqrt{T-t}), \quad (6)$$

with

$$z = \frac{\log(x/K) + (r + \sigma^2/2)(T-t)}{\sigma\sqrt{T-t}} \quad (7)$$

where Φ is the cumulative standard normal distribution function.

- The Monte Carlo method is well suited once we agree that security pricing can be represented by expectations (Duffie 1996). The approach consists of the following steps:
 1. Simulate sample paths of the underlying state variables (underlying asset prices and interest rates) over the relevant time horizon.
 2. Evaluate the discounted cash flows of a security on each sample path, as determined by the structure of the security in question. Compute C_j .
 3. Average the discounted cash flows over sample paths. Compute

$$\hat{C} = \frac{1}{N} \sum_{j=1}^N C_j. \quad (8)$$

4. Compute the standard deviation of the simulation

$$\hat{\sigma}_{\hat{C}} = \sqrt{\frac{1}{N-1} \sum_{j=1}^N (C_j - \hat{C})^2}. \quad (9)$$

- These methods compute a multi-dimensional integral, the expected value of the discounted payouts over the space of sample paths. The increase in the complexity of derivative securities in recent years has led to a need to evaluate high-dimensional integrals
- Monte Carlo becomes increasingly attractive compared to other methods of numerical integration as the dimension of the problem increases. For example is you want to integrate $f(x)$ in the unit hypercube, the value of the integral is the average value of the function over n points selected at random in the hypercube.

- By the SLLN this estimate converges to the true value of the integrand as n tends to infinity. And the CLT assures us that the standard error of the estimate tends to zero as $1/\sqrt{n}$. \hat{C} is normally distributed and we can construct a confidence interval around it.
- Thus the error convergence rate is independent of the dimension of the problem, this is the dominant advantage over quadrature methods. One drawback is that for very complex problems a large number of replications may be required to obtain precise results. We know we can also use deterministic low-discrepancy sequences and variance reduction techniques to improve upon the crude Monte Carlo.
- Now we are ready to describe the problem for which we have code. Think of the problem of pricing a multiasset option (option that depends on several underlying assets). Here we consider the pricing of an European call option on the maximum of n risky assets.
- The underlying assets have prices $S_1(t), S_2(t), \dots, S_n(t)$ at time $t = 0, \dots, T$ and the respective strike prices are K_1, K_2, \dots, K_n .

- We also assume the diffusion process

$$\frac{dS_i}{S_i} = \mu_i + \sigma_i dZ_i \quad i = 1, 2, \dots, n, \quad (10)$$

where μ_i and σ_i denote respectively the expected rate of return and volatility, and dZ_i is the Wiener process for asset i . These processes can be correlated and ρ_{ij} denotes the correlation coefficient between dZ_i and dZ_j . Notice that equation (10) has the same shape as equation (3).

- The price of the call at maturity time T is

$$C(T) = \max(\max(S_1(T) - K_1, S_2(T) - K_2, \dots, S_n(T) - K_n), 0), \quad (11)$$

and what we look for is the value of this option at time 0, $C(0)$. To price this option using MC methods follows the steps:

1. Decompose the correlation matrix with Cholesky: $\Sigma = LL'$. Meaning that you find the upper right triangular matrix that decomposes your positive-definite matrix of correlations.
2. For as many as N times generate an n dimensional vector of unit normal random values z .

3. Transform $\tilde{z} = Lz$.
4. Compute the discounted cash flows of the derivative C_j .
5. Average the discounted cash flows over sample paths. Compute

$$\hat{C} = \frac{1}{N} \sum_{j=1}^N C_j. \quad (12)$$

6. Compute the standard deviation of the simulation

$$\hat{\sigma}_{\hat{C}} = \sqrt{\frac{1}{N-1} \sum_{j=1}^N (C_j - \hat{C})^2}. \quad (13)$$