Black-Scholes Formula

https://github.com/HST0077/HYOTC

Review

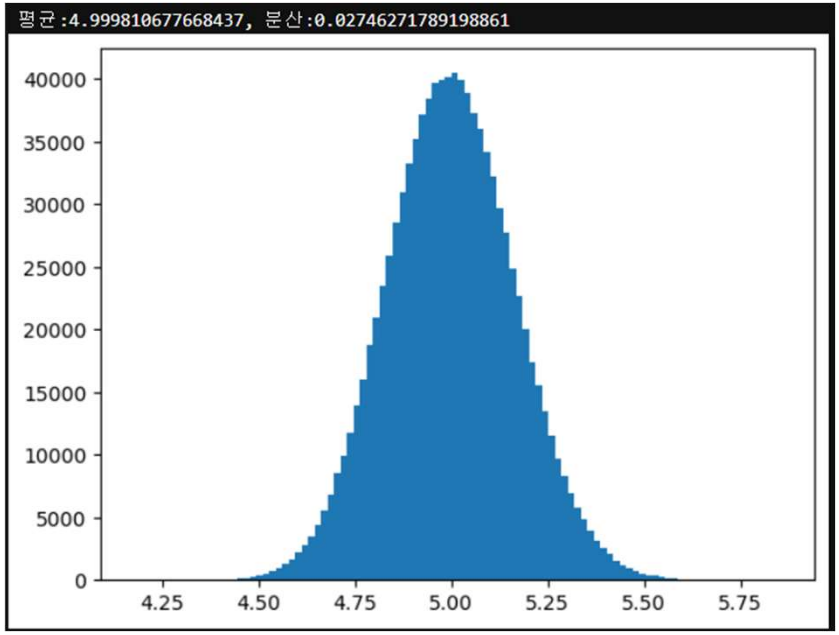
Stochastic Calculus

$\Delta W \sim N(0, \Delta)$, $\int_0^5 (dW_t)^2 = ?$

```python
import cupy as cp
from scipy.stats import norm
import matplotlib.pyplot as plt
import numpy as np

T=5 # num of years
N=T*365 # 일 단위로 환산한 만기까지의 기간
sim=1000000 # simulation 회수
s=cp.sqrt(1/365) # 하루를 년으로 환산하여 표준편차화
dW=cp.random.normal(0,s,[N,sim]) # [N,sim] 크기 정규난수 생성
dW2=dW**2
sumW=cp.sum(dW2,axis=0) # 각 열별로 누적합을 구함
plt.hist(sumW.get(),bins=100)
mu,sig=norm.fit(sumW.get()) # get()을 이용해 CPU로 변환
print('평균:{0}, 분산:{1}'.format(mu,sig**2))
```

```
평균:4.999810677668437, 분산:0.02746271789198861
```

Random number의 제곱의 합이 수렴하는 현상을 바탕으로 확률적분 연구가 확산됨



Ito lemma

$$\boldsymbol{\int_0^T (dW_t)^2 = T \rightarrow (dW)^2 \cong dt}$$

$$d \ln S_t = \frac{1}{S_t}(dS_t) - \frac{1}{2}\frac{1}{(S_t)^2}(dS_t)^2 + \cdots$$

$$\cong \frac{1}{S_t}(rS_t dt + \sigma S_t dW_t) - \frac{1}{2}\frac{1}{(S_t)^2}(\sigma^2 S_t^2 dt)$$

$$= \left(r - \frac{1}{2}\sigma^2\right)dt + \sigma dW_t$$

SDE for Geometric Brownian Motion

$$dN_t = rN_t dt + \sigma N_t dW_t$$

$$d\ln N_t = \frac{1}{N_t}(dN_t) - \frac{1}{2}\frac{1}{(N_t)^2}(dN_t)^2 + \cdots \cong rdt + \sigma dW_t - \frac{1}{2}\sigma^2 dt = \left(r - \frac{1}{2}\sigma^2\right)dt + \sigma dW_t$$

$$\int_0^t d\ln N_s = \left(r - \frac{1}{2}\sigma^2\right)t + \sigma\int_0^t dW_s$$

$$\therefore N_t = N_0 e^{\left(r-\frac{1}{2}\sigma^2\right)t+\sigma W_t}$$

Main Idea for Option Pricing

Relative Valuation

Backward looking side

•Cost based pricing
예) homogeneous product, perfect competition하에서의 핸드폰의 가격

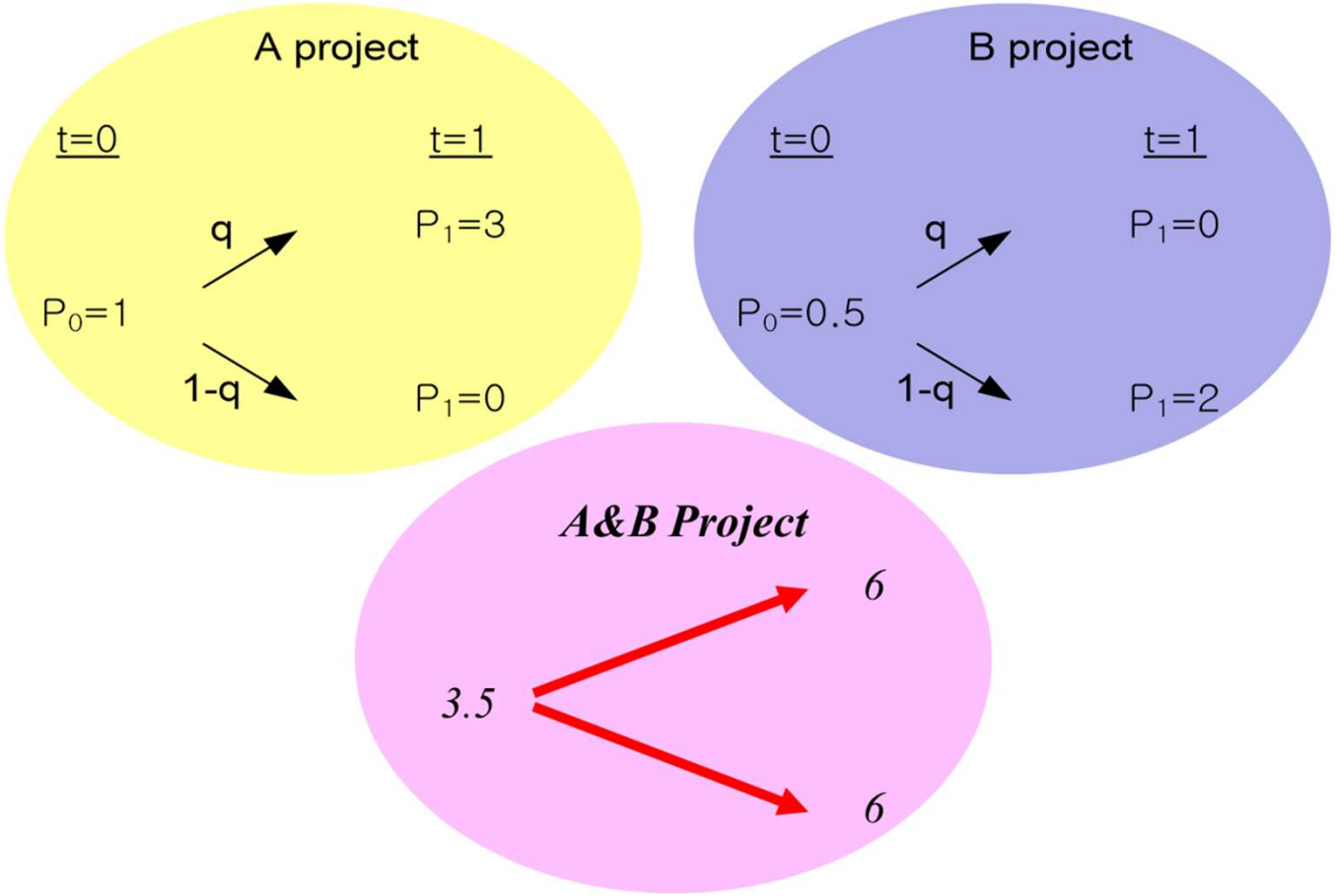
생산자의 입장에서의 공급의 하한이 됨

Forward looking side

•Discounted Cash Flow
예) 현금흐름의 크기,시점, 할인율을 추정하여 기업가격

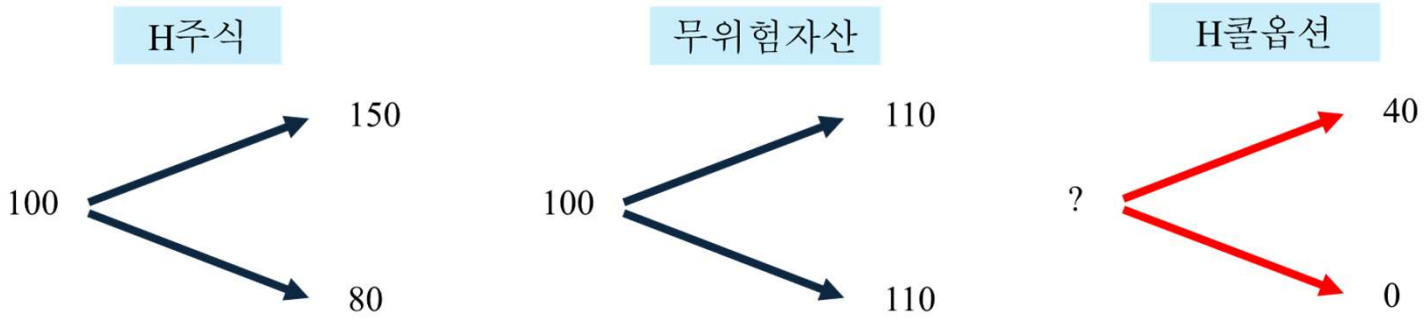
미래의 현금흐름과 할인율에 대한 예측이 불가능할 경우 벤치마킹을 통해 상대적인 평가를 하게 된다.

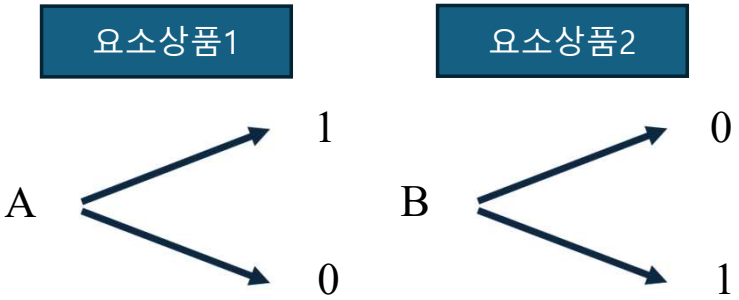
Example of Relative Valuation



Call option pricing

Q. 현재 무위험 이자율은 10%, H주식의 현재가는 100이라고 한다. 1년 후에 H주식의 주가는 150이나 80일 경우만 존재한다. 행사가 110인 H 콜옵션의 현재가는 얼마여야 하는가?





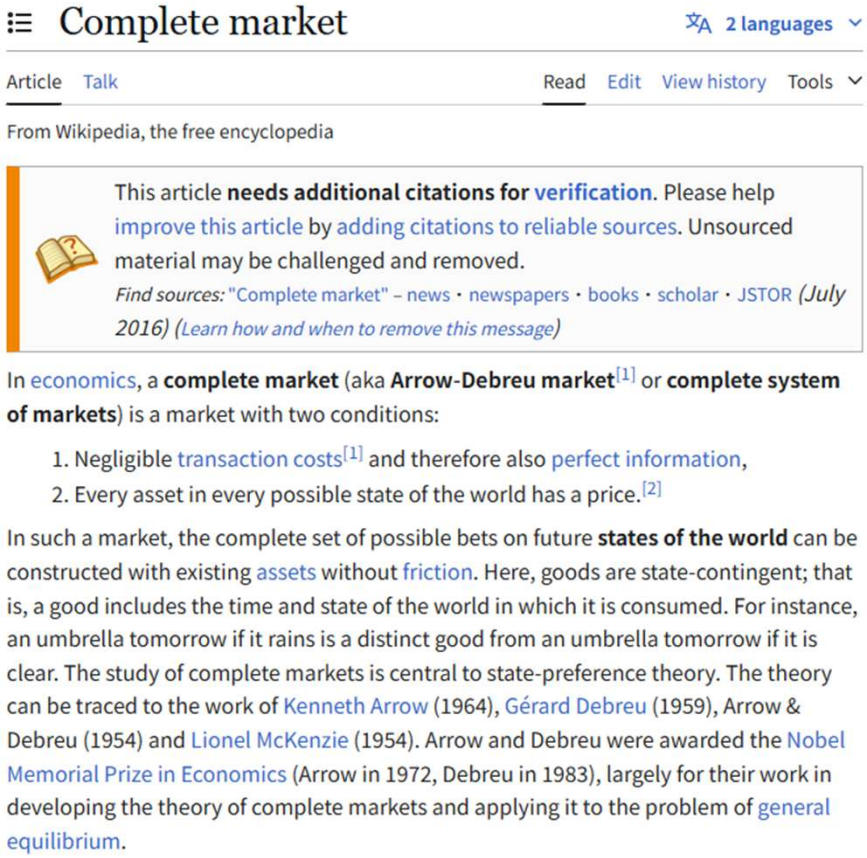

```
from sympy import symbols, Eq, solve

# 1. 변수 선언
A, B = symbols('A B')

# 2. 방정식 정의
eq1 = Eq(150*A + 80*B, 100)
eq2 = Eq(110*A + 110*B, 100)

# 3. 연립방정식 풀이
solution = solve((eq1, eq2), (A, B))

solution
```

```
{A: 30/77, B: 40/77}
```

H콜옵션의 가격:

40A=15.58

Complete market?



Complete market

2 languages

Article Talk Read Edit View history Tools

From Wikipedia, the free encyclopedia

This article **needs additional citations for verification**. Please help improve this article by adding citations to reliable sources. Unsourced material may be challenged and removed.

Find sources: "Complete market" – news · newspapers · books · scholar · JSTOR *(July 2016)* *(Learn how and when to remove this message)*

In economics, a **complete market** (aka **Arrow-Debreu market**[1] or **complete system of markets**) is a market with two conditions:

1. Negligible transaction costs[1] and therefore also perfect information,
2. Every asset in every possible state of the world has a price.[2]

In such a market, the complete set of possible bets on future **states of the world** can be constructed with existing assets without friction. Here, goods are state-contingent; that is, a good includes the time and state of the world in which it is consumed. For instance, an umbrella tomorrow if it rains is a distinct good from an umbrella tomorrow if it is clear. The study of complete markets is central to state-preference theory. The theory can be traced to the work of Kenneth Arrow (1964), Gérard Debreu (1959), Arrow & Debreu (1954) and Lionel McKenzie (1954). Arrow and Debreu were awarded the Nobel Memorial Prize in Economics (Arrow in 1972, Debreu in 1983), largely for their work in developing the theory of complete markets and applying it to the problem of general equilibrium.

시장에서 거래되는 자산들의 조합으로 어떠한 Cash Flow라도 복제할 수 있는 시장을 완비시장(complete market)이라고 한다.

→ 경제적 상황에 따른 각각의 자산을 하나의 상태별 지급 벡터(payoff vector) 로 표시한다고 할 때,이들 자산 벡터들이 서로 선형독립(linearly independent) 이고, 그들의 선형결합(span) 이 모든 가능한 상태공간을 덮는다면, 어떠한 형태의 Cash Flow라도 자산들의 조합으로 복제할 수 있다.

Black Scholes Formula

From SDE to PDE

SDE for Stock Price

$$dS_t = rS_t dt + \sigma S_t dW_t$$

SDE for Option Price

$$V(S_t, t)$$

$$dV$$

$$= \frac{\partial V}{\partial S} dS + \frac{\partial V}{\partial t} dt + \frac{1}{2} \frac{\partial^2 V}{\partial S^2} (dS)^2 + \frac{1}{2} \frac{\partial^2 V}{\partial t^2} (dt)^2 + \frac{\partial^2 V}{\partial S \partial t} (dS)(dt)$$

$$\cong \frac{\partial V}{\partial S} (rSdt + \sigma SdW) + \frac{\partial V}{\partial t} dt + \frac{1}{2} \frac{\partial^2 V}{\partial S^2} (\sigma^2 S^2 dt)$$

$$= \left(rS \frac{\partial V}{\partial S} + \frac{\partial V}{\partial t} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} \right) dt + \sigma S \frac{\partial V}{\partial S} dW$$

From SDE to PDE (Cont'd)

SDE for Stock Price

$$dS = rSdt + \sigma S dW$$

SDE for Option Price

$$dV = \left(rS\frac{\partial V}{\partial S} + \frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} \right) dt + \sigma S \frac{\partial V}{\partial S} dW$$

$$P \equiv V - \frac{\partial V}{\partial S} S \rightarrow dP \cong dV - \frac{\partial V}{\partial S} dS$$

$$dP \cong dV - \frac{\partial V}{\partial S} dS = \left(\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} \right) dt$$

위험항이 제거 되어 PDE가 되었고, 무위험포트폴리오이기 때문에 무위험 이자율로 성장해야 함

$$dP = Prdt$$

From SDE to PDE (Cont'd)

$$dP \cong dV - \frac{\partial V}{\partial S} dS = \left(\frac{\partial V}{\partial t} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} \right) dt \quad \Rightarrow \quad dP = Prdt$$

$$dP \cong dV - \frac{\partial V}{\partial S} dS = \left(\frac{\partial V}{\partial t} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} \right) dt = r \left(V - \frac{\partial V}{\partial S} S \right) dt$$

$$\Downarrow$$

$$\frac{\partial V}{\partial t} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 V}{\partial t^2} + rS \frac{\partial V}{\partial S} - rV = 0$$

Call Option Price Formula

$$V(S_T, T) = \max(S_T - K, 0), \quad \frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial t^2} + rS\frac{\partial V}{\partial S} - rV = 0$$

$$N(x) \equiv \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{t^2}{2}} dt$$

$$V = S_0 N(d_1) - K e^{-rT} N(d_2), d_1 \equiv \frac{\ln\left(\frac{S_0}{K}\right) + \left(r + \frac{1}{2}\sigma^2\right)T}{\sigma\sqrt{T}}, d_2 \equiv d_1 - \sigma\sqrt{T}$$

Call Option Price Formula _ Python sympy

$$V = S_0 N(d_1) - K e^{-r}\ N(d_2), d_1 \equiv \frac{\ln\left(\frac{S_0}{K}\right) + \left(r + \frac{1}{2}\sigma^2\right)T}{\sigma\sqrt{T}}, d_2 \equiv d_1 - \sigma\sqrt{T}$$

$$N(x) \equiv \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{t^2}{2}} dt, \quad \text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt, \quad N(x) = \frac{1}{2}\left(1 + \text{erf}\left(\frac{x}{\sqrt{2}}\right)\right)$$

```
# error function
import sympy as sp

x=sp.symbols('x')
sp.erf(x)
```

$\text{erf}(x)$

```
import sympy as sp

# 표준정규분포 누적확률밀도함수 (CDF)
N = lambda x: (1 / 2) * (1 + sp.erf(x / sp.sqrt(2)))

N(0),N(sp.oo)
```

```
(0.500000000000000, 1.00000000000000)
```

```
# 변수 정의
S0, K, r, sigma, T = sp.symbols('S0 K r sigma T')
d1 = (sp.ln(S0 / K) + (r + sigma**2 / 2) * T) / (sigma * sp.sqrt(T))
d2 = d1 - sigma * sp.sqrt(T)

# Black-Scholes 공식
C = S0 * N(d1) - K * sp.exp(-r * T) * N(d2)
```

Call Option Price Formula _ Python numpy

$$V = S_0 N(d_1) - K e^{-rT} N(d_2), d_1 \equiv \frac{\ln\left(\frac{S_0}{K}\right) + \left(r + \frac{1}{2}\sigma^2\right)T}{\sigma\sqrt{T}}, d_2 \equiv d_1 - \sigma\sqrt{T}$$

```python
from math import log, sqrt, pi, exp
from scipy.stats import norm
import numpy as np

def d1(S, K, T, r, sigma):
    return (log(S / K) + (r + sigma ** 2 / 2) * T) / (sigma * sqrt(T))

def d2(S, K, T, r, sigma):
    return d1(S, K, T, r, sigma) - sigma * sqrt(T)

def bs_call(S, K, T, r, sigma):
    return S * norm.cdf(d1(S, K, T, r, sigma)) - K * exp(-r * T) * norm.cdf(d2(S, K, T, r, sigma))
```

Call Option Price

Q. 현재 무위험 이자율은 3%, H주식의 현재가는 100,변동성이 30%라고 한다. 6개월 만기 행사가가 110인 유럽형 콜옵션의 공정가격은 얼마일까?

```
def bs_call(S, K, T, r, sigma):
    return S * norm.cdf(d1(S, K, T, r, sigma)) - K * exp(-r * T) * norm.cdf(d2(S, K, T, r, sigma))
```

```
print('H콜옵션의 가격:',bs_call(100,110,6/12,0.03,0.3))
```

```
H콜옵션의 가격: 5.239505678484882
```

Put Option Price

Q. 블랙숄즈 공식을 이용하여 기초자산의 현재가 100, 행사가 100, 잔여만기 1년인 풋옵션의 현재가를 추정하여라. (단, 무위험 이자율 5% p.a., 기초자산의 변동성 30% p.a.)

$$C = S_0 N(d_1) - Ke^{-r}\ N(d_2), \qquad P = Ke^{-rT} N(-d_2) - S_0 N(-d_1)$$

$$P = Ke^{-rT}\big(1 - N(d_2)\big) - S_0\big(1 - N(d_1)\big) = Ke^{-rT} - S_0 + C$$

```
def bs_put(S, K, T, r, sigma):
    return K * exp(-r * T) - S + bs_call(S, K, T, r, sigma)
```

```
print('풋옵션의 현재가:',bs_put(100,100,1,0.05,0.3))
```

```
풋옵션의 현재가: 9.354197236057225
```

Greeks of Options

Q. 기초자산 현재가 100, 행사가 100, 잔여만기 1년인 콜옵션의 현재가와 델타, 감마, 1% 베가, 1day 세타, 1bp 르호 값을 구하여라. (단, 무위험이자율은 5% p.a., 기초자산의 변동성은 30% p.a.)

```
def bs_call(S, K, T, r, sigma):
    return S * norm.cdf(d1(S, K, T, r, sigma)) - K * exp(-r * T) * norm.cdf(d2(S, K, T, r, sigma))

def call_delta(S, K, T, r, sigma):
    return norm.cdf(d1(S, K, T, r, sigma))

def call_gamma(S, K, T, r, sigma):
    return norm.pdf(d1(S, K, T, r, sigma)) / (S * sigma * sqrt(T))

def call_vega(S, K, T, r, sigma): # 1% 변동성 민감도
    return 0.01 * (S * norm.pdf(d1(S, K, T, r, sigma)) * sqrt(T))

def call_theta(S, K, T, r, sigma): # 1 day 감소에 대한 민감도
    return (1/365) * (- (S * norm.pdf(d1(S, K, T, r, sigma)) * sigma) / (2 * sqrt(T)) #
                      - r * K * exp(-r * T) * norm.cdf(d2(S, K, T, r, sigma)))

def call_rho(S, K, T, r, sigma): # 1bp 변화에 대한 민감도
    return 0.0001 * ((K * T * exp(-r * T) * norm.cdf(d2(S, K, T, r, sigma))))
```

```
print('콜옵션의 현재가:',bs_call(100,100,1,0.05,0.3))
print('콜옵션의 델타:',call_delta(100,100,1,0.05,0.3))
print('콜옵션의 감마:',call_gamma(100,100,1,0.05,0.3))
print('콜옵션의 1% 베가:',call_vega(100,100,1,0.05,0.3))
print('콜옵션의 1day 세타:',call_theta(100,100,1,0.05,0.3))
print('콜옵션의 1bp 르호:',call_rho(100,100,1,0.05,0.3))
```

```
콜옵션의 현재가: 14.231254785985819
콜옵션의 델타: 0.6242517279060125
콜옵션의 감마: 0.012647764437231512
콜옵션의 1% 베가: 0.3794329331169454
콜옵션의 1day 세타: -0.0221950408136574
콜옵션의 1bp 르호: 0.004819391800461543
```

Monte Carlo Simulation for Call option

Stock Price Path

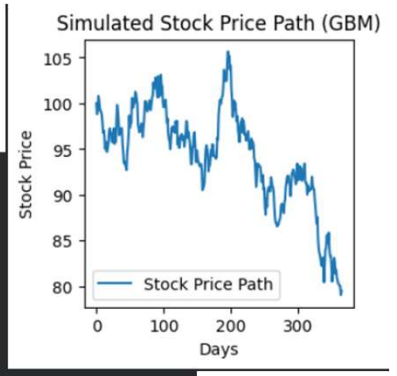
Q. 1년을 365일로 가정하고, 매일 종가 데이터를 GBM모형에 의해 하나의 path로 생성해 보아라. (단, 현재 주가 100, 무위험 이자율 5% p.a., 기초자산의 변동성 30% p.a.)

$$S_{t+\Delta} = S_t e^{\left(r-\frac{1}{2}\sigma^2\right)\Delta+\sigma dW_t}, dW_t \equiv \sqrt{dt}W$$

$$d\ln S_t = \left(r - \frac{1}{2}\sigma^2\right)dt + \sigma dW_t$$

```
# 파라미터 설정
S0 = 100    # 초기 주가
r = 0.05    # 무위험 이자율
sigma = 0.2 # 변동성
T = 1       # 기간 (년)
N = 365     # 시뮬레이션 일수 (하루 단위)
dt = T / N  # 시간 간격

# 주가 경로 생성
np.random.seed(111)
W = np.random.randn(N)  # 표준 정규분포 난수
S = np.zeros(N+1)
S[0] = S0

for t in range(1, N+1):
    S[t] = S[t-1] * np.exp((r - 0.5 * sigma**2) * dt
                           + sigma * np.sqrt(dt) * W[t-1])
```

```
# 주가 경로 생성
np.random.seed(111)
W = np.random.randn(N)  # 표준 정규분포 난수
S = np.zeros(N+1)

# log(S)벡터 만들기
lnS=(r - 0.5 * sigma**2) * dt + sigma * np.sqrt(dt) * W
lnS=np.insert(lnS,0,np.log(S0)) #초기값 추가

# 누적합 구하기
S=np.exp(np.cumsum(lnS))
```



Option Pricing using MC simulation

Q. 몬테카를로 시뮬레이션을 이용하여 기초자산의 현재가 100, 행사가 100, 잔여만기 1년인 콜옵션의 현재가를 추정하여라. (단, 무위험 이자율 5% p.a., 기초자산의 변동성 30% p.a.)

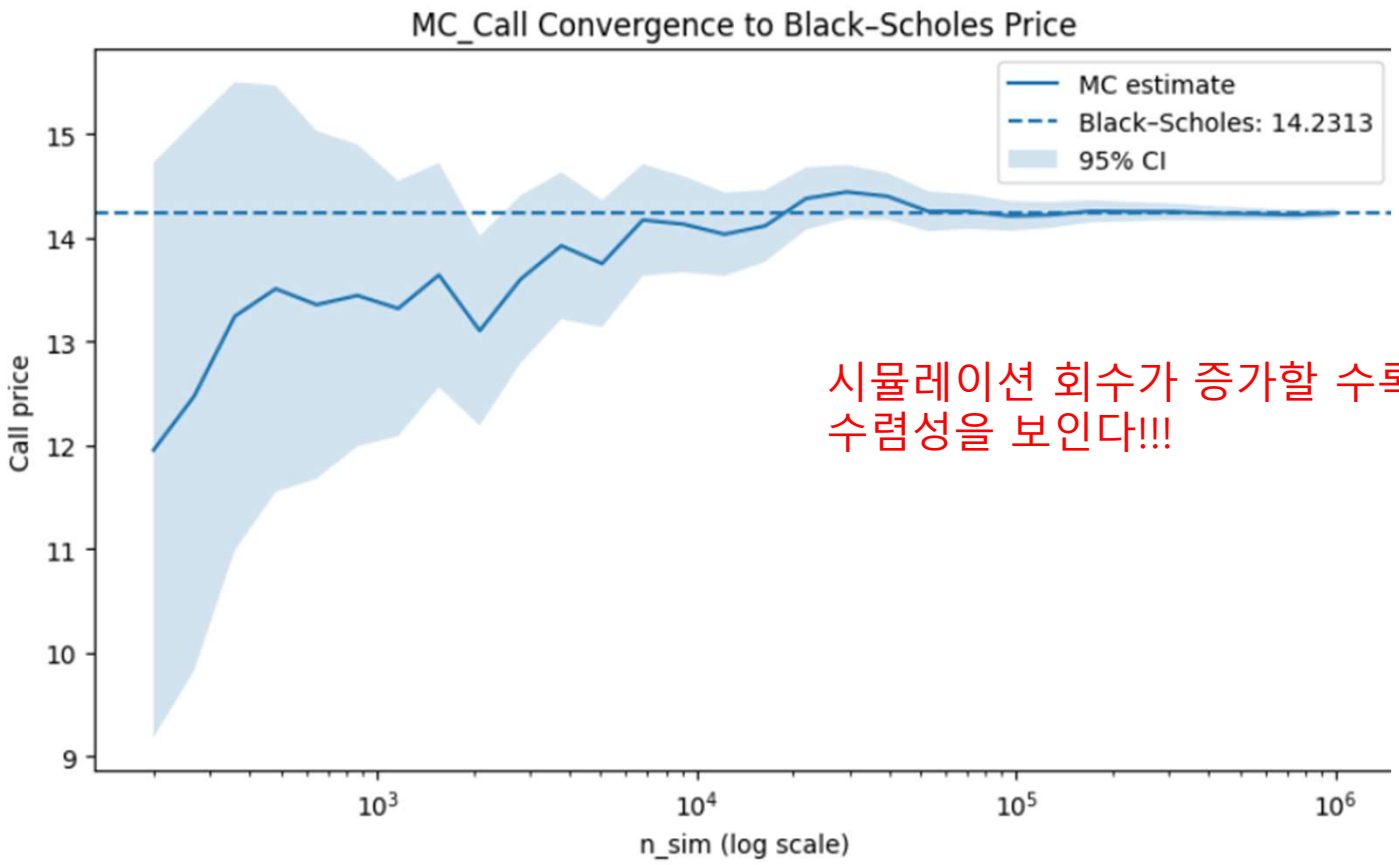
$$S_T = S_0 e^{\left(r-\frac{1}{2}\sigma^2\right)T+\sigma dW_T}, dW_T \equiv \sqrt{T}W$$

```
# 난수 생성 (표준 정규분포)
Z = np.random.randn(n_sim)

# 몬테카를로 시뮬레이션 (기하 브라운 운동 사용)
ST = S0 * np.exp((r - 0.5 * sigma**2) * T + sigma * np.sqrt(T) * Z)
```

```
# 콜옵션의 페이오프 계산
payoff = np.maximum(ST - K, 0)

# 현재가치 할인 적용 (할인율: e^(-rT))
call_price = np.exp(-r * T) * np.mean(payoff)

# 결과 출력
print(f"몬테카를로 시뮬레이션 기반 콜옵션 가격: {call_price:.4f}")
print(f"블랙숄즈 공식 기반 콜옵션 가격: {bs_call(100,100,1,0.05,0.3):.4f}")
```

```
몬테카를로 시뮬레이션 기반 콜옵션 가격: 14.2468
블랙숄즈 공식 기반 콜옵션 가격: 14.2313
```

Option Pricing using MC simulation (Cont'd)



시뮬레이션 회수가 증가할 수록 콜옵션 가격은 수렴성을 보인다!!!

Option Greeks using MC simulation

Q. 몬테카를로 시뮬레이션을 이용하여 기초자산의 현재가 100, 행사가 100, 잔여만기 1년인 콜옵션의 민감도(Greeks)들을 추정하여라. (단, 무위험 이자율 5% p.a., 기초자산의 변동성 30% p.a.)

```python
# Delta (Δ)
S,K,T,r,sigma=100,100,1,0.05,0.3
eps=S*0.01
C_plus = MC_Call(S + eps, K, T, r, sigma)
C_minus = MC_Call(S - eps, K, T, r, sigma)
Delta = (C_plus - C_minus) / (2 * eps)

# 결과 출력
print(f"몬테카를로 시뮬레이션 기반 콜옵션 델타: {Delta:.4f}")
print(f"블랙숄즈 공식 기반 콜옵션 델타: {call_delta(100,100,1,0.05,0.3):.4f}")
```

```
몬테카를로 시뮬레이션 기반 콜옵션 델타: 0.6475
블랙숄즈 공식 기반 콜옵션 델타: 0.6243
```

```python
# Gamma (Γ)
eps=S*0.01
C_0 = MC_Call(S, K, T, r, sigma)
Gamma = (C_plus - 2 * C_0 + C_minus) / (eps ** 2)
# 결과 출력
print(f"몬테카를로 시뮬레이션 기반 콜옵션 감마: {Gamma:.4f}")
print(f"블랙숄즈 공식 기반 콜옵션 감마: {call_gamma(100,100,1,0.05,0.3):.4f}")
```

```
몬테카를로 시뮬레이션 기반 콜옵션 감마: 0.0632
블랙숄즈 공식 기반 콜옵션 감마: 0.0126
```

```python
# Vega (V)
eps=sigma*0.01
C_sigma_plus = MC_Call(S, K, T, r, sigma + eps)
C_sigma_minus = MC_Call(S, K, T, r, sigma - eps)
Vega = (C_sigma_plus - C_sigma_minus) / (2 * eps)
Vega = 0.01*Vega # 1% 베가로 변환
# 결과 출력
print(f"몬테카를로 시뮬레이션 기반 콜옵션 베가: {Vega:.4f}")
print(f"블랙숄즈 공식 기반 콜옵션 베가: {call_vega(100,100,1,0.05,0.3):.4f}")
```

```
몬테카를로 시뮬레이션 기반 콜옵션 베가: 0.5234
블랙숄즈 공식 기반 콜옵션 베가: 0.3794
```

```python
# 1day Theta (Θ)
eps=0.01
C_t = MC_Call(S, K, T - eps, r, sigma)
Theta = (1/365)*(C_t - C_0) / eps
# 결과 출력
print(f"몬테카를로 시뮬레이션 기반 콜옵션 세타: {Theta:.4f}")
print(f"블랙숄즈 공식 기반 콜옵션 세타: {call_theta(100,100,1,0.05,0.3):.4f}")
```

```
몬테카를로 시뮬레이션 기반 콜옵션 세타: -0.0385
블랙숄즈 공식 기반 콜옵션 세타: -0.0222
```

```python
# 1bp Rho (ρ)
eps=0.01
C_r_plus = MC_Call(S, K, T, r + eps, sigma)
C_r_minus = MC_Call(S, K, T, r - eps, sigma)
Rho = 0.0001* (C_r_plus - C_r_minus) / (2 * eps)
# 결과 출력
print(f"몬테카를로 시뮬레이션 기반 콜옵션 르호: {Rho:.4f}")
print(f"블랙숄즈 공식 기반 콜옵션 르호: {call_rho(100,100,1,0.05,0.3):.4f}")
```

```
몬테카를로 시뮬레이션 기반 콜옵션 르호: 0.0051
블랙숄즈 공식 기반 콜옵션 르호: 0.0048
```