

Final: ELS Pricing

(Due: Oct. 24, 2020, 11:59 PM)

You have to do all work on your own. You are not supposed to discuss with other students. If any dishonest act are found, you may face severe penalty.

Choose any ELS written on two underlying assets. To find an example, you may visit homepage of security investment banks.

1. What is the name of ELS?
2. Explain payoff structure.
3. Specify parameters precisely. For example, if you consider volatility surface instead of constant volatility,
4. Price the ELS using ADI FDM or OS FDM. You have to describe the whole procedure of pricing scheme.
5. Simulate paths of underlying assets and calculate the price. Compare two prices with the price the bank suggests.
6. Calculate Greeks and explain price movement according to Greeks
7. You may add hedging analysis if you can.
8. Program codes should be handed in with documents together.
9. Zip all files into one and upload.

1. What is the name of ELS?

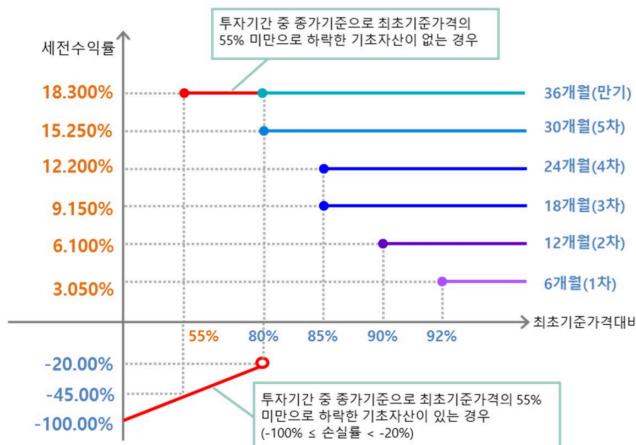
어떤 ELS를 쓸까 많은 고민을 해보았다. NH투자증권, 미래에셋대우투자증권, 한국투자증권, 유안타투자증권, 신영증권 사이트 모두를 방문해 보았는데, 상품에 대해 가장 상세한 설명을 해주는 증권사가 신영증권이었다. 다른 증권사의 경우에는 우리나라 금융자산과 외국의 금융자산을 조합해서 **환리스크**가 존재하거나, **월별 쿠폰 지급** 등의 정형화되어 있지 않은 상품들이 많았다.

그에 비해 유안타증권은 우리나라 자산들로 구성된 2 star ELS가 많이 존재했으며, 그래서 여러 상품들을 보고 마음에 드는 상품을 고를 수 있었다.

내가 고른 ELS 상품은 "유안타 MY ELS 제4623호"이며, 기초자산은 (KOSPI200, POSCO보통주)이다.

2. Explain payoff structure.

예상수익구조



(1) 자동조기상환조건

구분	내용	수익률(세전)
1 차	1차 자동조기상환평가일에 모든 기초자산의 자동조기상환평가가격이 각 최초기준가격의 92% 이상인 경우	3.05% (연 6.10%)
2 차	2차 자동조기상환평가일에 모든 기초자산의 자동조기상환평가가격이 각 최초기준가격의 90% 이상인 경우	6.10% (연 6.10%)
3 차	3차 자동조기상환평가일에 모든 기초자산의 자동조기상환평가가격이 각 최초기준가격의 85% 이상인 경우	9.15% (연 6.10%)
4 차	4차 자동조기상환평가일에 모든 기초자산의 자동조기상환평가가격이 각 최초기준가격의 85% 이상인 경우	12.20% (연 6.10%)
5 차	5차 자동조기상환평가일에 모든 기초자산의 자동조기상환평가가격이 각 최초기준가격의 80% 이상인 경우	15.25% (연 6.10%)

(2) 만기상환조건

구분	내용	수익률(세전)
만기상환 ①	(1)의 요건을 충족하지 못하였고, 만기평가일에 모든 기초자산의 만기평가가격이 각 최초기준가격 × 80% 이상인 경우	18.30% (연 6.10%)
만기상환 ②	(2)-(3)의 요건을 충족하지 못하였고, 최초기준가격결정일(불포함)부터 만기평가일(포함)까지 모든 기초자산 중 어느 하나라도 각 최초기준가격 × 55%(증가 기준) 미만으로 하락한 적이 있는 경우	18.30% (연 6.10%)
만기상환 ③	(2)-(3)의 요건을 충족하지 못하였고, 최초기준가격결정일(불포함)부터 만기평가일(포함)까지 모든 기초자산 중 어느 하나라도 각 최초기준가격 × 55%(증가 기준) 미만으로 하락한 적이 있는 경우 → 원금손실 발생 (만기평가일에 하락률이 큰 기초자산 기준)	기준증권의 ((만기평가가격/최초기준가격)-1) × 100%

〈만기〉

$$80\% \quad | \quad 1 + Cr_0 \\ | + Cr_0 \\ min(S_x(t_0)/S_x(t_f), S_y(t_0)/S_y(t_f))$$

$$1 + Cr_0 \\ min(S_x(t_0)/S_x(t_f), S_y(t_0)/S_y(t_f))$$

〈배위어터치 X〉

행사가격: 6개월마다 92%, 90%, 85%, 85%, 80%, 80%

평균: 55%

3. Specify parameters precisely. For example, if you consider volatility surface instead of constant volatility,

항 목	내 용
기초자산가격 변동성	6개월 변동성 - KOSPI200 지수 : 17,980.25777% - POSCO 보통주 : 30,508.83903%
기초자산 일별수익률 간의 상관계수	6개월 상관계수 - KOSPI200 지수와 POSCO 보통주 : 0.5831793444

* 산출기준: EWMA모형을 이용한 변동성 및 상관계수 적용

최초기준가격	2020. 10. 08 각 기초자산의 종가

일별시세						
날짜	종가	전일비	시가	고가	저가	거래량
2020.10.16	202,000	▼ 3,000	206,500	207,500	201,000	268,787
2020.10.15	205,000	▲ 4,500	200,000	208,000	200,000	309,713
2020.10.14	200,500	▼ 4,500	207,000	207,000	200,000	300,335
2020.10.13	205,000	▼ 2,000	206,500	207,500	204,500	190,587
2020.10.12	207,000	▼ 1,000	208,500	208,500	204,000	269,502
2020.10.08	208,000	▲ 3,000	208,000	213,000	206,500	554,481

일별시세					
날짜	체결가	전일비	등락률	거래량(천주)	거래대금(백만)
2020.10.16	311.83	▼ 2.09	-0.67%	126,440	6,249,577
2020.10.15	313.92	▼ 2.65	-0.84%	124,410	6,366,830
2020.10.14	316.57	▼ 3.01	-0.94%	129,341	6,719,033
2020.10.13	319.58	▲ 0.43	+0.13%	126,741	6,568,609
2020.10.12	319.15	▲ 1.68	+0.53%	140,937	6,923,381
2020.10.08	317.47	▲ 0.12	+0.04%	168,692	8,471,910

$$\chi_0 = 317.41$$

$$y_0 = 208000$$

$$\tau_x = 0.19802577$$

$$\tau_y = 0.3050883903$$

$$\rho = 0.5831993444$$

$$D_x = 0.02$$

$$D_y = 0.03$$

$$r = 0.0063$$

임의로 정함.

4. Price the ELS using ADI FDM or OS FDM. You have to describe the whole procedure of pricing scheme.

- 방법론 : "OS FDM" 사용 (기존은 틀라 없음.)

- 순서

```

tmp = np.arange(1, I)
if method == 'ADI':
    a1 = 1 + dt/2*(r + (r-D_1) * tmp + (vol_1**2) * (tmp**2))
    b1 = -dt/4*(vol_1**2)*(tmp**2)
    c1 = -dt/2*((r-D_1)*tmp + (vol_1**2)/2 * (tmp**2))
    a2 = 1 + dt/2*(r + (r-D_2) * tmp + (vol_2**2) * (tmp**2))
    b2 = -dt/4*(vol_2**2)*(tmp**2)
    c2 = -dt/2*((r-D_2)*tmp + (vol_2**2)/2 * (tmp**2))
elif method == 'OS':
    a1 = 1 + dt * (r/2 + (r-D_1)*tmp + (tmp**2)*(vol_1**2))
    a2 = 1 + dt * (r/2 + (r-D_2)*tmp + (tmp**2)*(vol_2**2))
    b1 = -dt/2 * (tmp**2) * (vol_1**2)
    b2 = -dt/2 * (tmp**2) * (vol_2**2)
    c1 = -dt * ((r-D_1)*tmp + (vol_1**2)/2*(tmp**2))
    c2 = -dt * ((r-D_2)*tmp + (vol_2**2)/2*(tmp**2))
else:
    print('InputError: use method "ADI" or "OS"')
    return -1
a1_prime = a1.copy()
b1_prime = b1.copy()
c1_prime = c1.copy()
a2_prime = a2.copy()
b2_prime = b2.copy()
c2_prime = c2.copy()
a1_prime[0] = a1[0] + 2*b1[0]
c1_prime[0] = c1[0] - b1[0]
a2_prime[0] = a2[0] + 2*b2[0]
c2_prime[0] = c2[0] - b2[0]
# a_prime[I-2] = a[I-2] + 2*c[I-2]
# b_prime[I-2] = b[I-2] - c[I-2]
a1_prime[I-2] = a1[I-2] + 2*c1[I-2]
b1_prime[I-2] = b1[I-2] - c1[I-2]
a2_prime[I-2] = a2[I-2] + 2*c2[I-2]
b2_prime[I-2] = b2[I-2] - c2[I-2]
```

① a, b, c 정의 및 a', b', c' 정의

$$a_x = 1 + k \left(\frac{r}{2} + \frac{(r-a)x_i}{h} + \frac{(\tau_i \chi_i)^2}{h^2} \right)$$

$$b_x = -\frac{k(\tau_i \chi_i)^2}{2h^2}$$

$$c_x = -k \left(\frac{(r-D)x_i}{h} + \frac{(\tau_i \chi_i)^2}{2h^2} \right)$$

a_y, b_y, c_y 도 같은 방식으로 구할 수 있음.

(여기 $k = \frac{\tau_i \chi_i}{h}$ 는 인테리벌 부분인 것을 이용하면, 구조상으로는 $\text{np.arange}(1, I)$ 을 배울 수 있음)

a', b', c' 은 a, b, c 를 나머지는 동일하게,

다음식으로 수정하여 구할 수 있음.

(여러식을 x, y 모두에 적용 시켜면됨)

$$a'_i = a_i + 2b_i$$

$$c'_i = c_i - b_i$$

$$a'_{Nx-1} = a_{Nx-1} + 2c_{Nx-1}$$

$$b'_{Nx-1} = b_{Nx-1} - c_{Nx-1}$$

→ 교수님 PPT에는 다른, typing error로 빠여짐.

② U , W 정의.

U 는 비리스크를 터치하지 않은 경우.

W 는 비리스크를 터치한 경우이다.

③ 초기 Setting ($T=0$ 시점)

위 payoff 조건에 맞게 초기시점 setting을 해야된다.

i) U (비리스크를 터치하지 않은 경우)

1) 기초자산 가격 증가가 비리스크보다 높을 경우

$$U = 1 + \text{Coupon rate} (18.3\%)$$

2) 등락 하더라도 비리스크보다 높을 경우

$$U = \min(\text{Kospi 200 증가율}, \text{삼성전자 보통주 증가율})$$

ii) W (비리스크를 터치한 경우)

1) 기초자산 가격 증가가 행사가격보다 높을 경우

$$W = 1 + \text{Coupon rate} (18.3\%)$$

2) 등락 하더라도 행사가격보다 높을 경우.

$$W = \min(\text{Kospi 200 증가율}, \text{삼성전자 보통주 증가율})$$

④ 초기 Boundary Condition 설정.

Boundary conditions are given by

$$u(\tau, 0, y) = u_{0,j}^n = 2u_{1,j}^n - u_{2,j}^n, \quad u(\tau, x_{max}, y) = u_{Nx,j}^n = 2u_{Nx-1,j}^n - u_{Nx-2,j}^n$$

$$u(\tau, x, 0) = u_{i,0}^n = 2u_{i,1}^n - u_{i,2}^n, \quad u(\tau, x, y_{max}) = u_{i,Ny}^n = 2u_{i,Ny-1}^n - u_{i,Ny-2}^n$$

⑤ $T = 1 \dots N_t$ 시점의 U, W 찾기.

For $n = 1:N_t$

For $j = 1:Ny - 1$

For $i = 1:Nx - 1$

Set a_i, b_i, c_i and d_n

Solve Eq. (9.1B) using Thomas Algorithm

For $i = 1:Nx - 1$

For $j = 1:Ny - 1$

Set a_j, b_j, c_j and d_n

Solve Eq. (9.2B) using Thomas Algorithm

$\rightarrow U(\text{상승})$ 뿐만
d \neq 초기!

$\rightarrow X(\text{증가})$ 뿐만
d \neq 초기!

(1) first step : $n \rightarrow n^*$ 시점

$$d_i^n = u_{i,j}^n + \frac{k\rho\sigma_1\sigma_2 x_i y_j}{2 \cdot 4 h^2} (u_{i+1,j+1}^n + u_{i-1,j-1}^n - u_{i-1,j+1}^n - u_{i+1,j-1}^n)$$



Tri-diagonal 풀기 \Rightarrow Thomas algorithm.

* U^*, W^* Boundary Condition,
Barrier hit 경우 쓰워져야 한다!

(2) Second Step : $n^* \rightarrow n+1$ 시점

$$\bullet d_{n+1}^{n*} = u_{i,j}^{n*} + \frac{k\rho\sigma_1\sigma_2 x_i y_j}{2 \cdot 4 h^2} (u_{i+1,j+1}^{n*} + u_{i-1,j-1}^{n*} - u_{i-1,j+1}^{n*} - u_{i+1,j-1}^{n*})$$

```

u2 = np.zeros((I+1, I+1))
w2 = np.zeros((I+1, I+1))
for n in range(0, N):
    print(n)
    # First Stage
    for j in range(1, I): # Y
        if method == 'ADI':
            d = u[n, 1:-1, j] + dt * (r-D_2) * j/2 * (u[n, 1:-1, j+1] - u[n, 1:-1, j])
            + dt*(vol_2**2)/4*(j*j) * (u[n, 1:-1, j+1]-2*u[n, 1:-1, j]+u[n, 1:-1, j-1])
            + dt*corr*vol_1*vol_2* tmp * j / 8*(u[n, 2:, j+1] + u[n, :-2, j-1]
            - u[n, :-2, j+1] - u[n, 2:, j-1])
            d2 = w[n, 1:-1, j] + dt * (r-D_2) * j/2 * (w[n, 1:-1, j+1] - w[n, 1:-1, j])
            + dt*(vol_2**2)/4*(j*j) * (w[n, 1:-1, j+1]-2*w[n, 1:-1, j]+w[n, 1:-1, j-1])
            + dt*corr*vol_1*vol_2* tmp * j / 8*(w[n, 2:, j+1] + w[n, :-2, j-1]
            - w[n, :-2, j+1] - w[n, 2:, j-1])
        elif method == 'OS':
            d = u[n, 1:-1, j] + dt*corr*vol_1*vol_2* tmp * j / 8
            *(u[n, 2:, j+1] + u[n, :-2, j-1] - u[n, :-2, j+1] - u[n, 2:, j-1])
            d2 = w[n, 1:-1, j] + dt*corr*vol_1*vol_2* tmp * j / 8
            *(w[n, 2:, j+1] + w[n, :-2, j-1] - w[n, :-2, j+1] - w[n, 2:, j-1])
        u[1:I, j] = Thomas_algol(al_prime, bl_prime, cl_prime, d)
        w[1:I, j] = Thomas_algol(al_prime, bl_prime, cl_prime, d2)
    
```

```

# u_star, w_star boundary condition
u2[:, 0] = 2*u2[:, 1]-u2[:, 2]
u2[:, -1] = 2*u2[:, -2]-u2[:, -3]
u2[:, 1:] = 2*u2[:, 1:] - u2[:, 1]
u2[-1, :] = 2*u2[-2, :] - u2[-3, :]
w2[:, 0] = 2*w2[:, 1]-w2[:, 2]
w2[:, -1] = 2*w2[:, -2]-w2[:, -3]
w2[:, 1:] = 2*w2[:, 1:] - w2[:, 2]
w2[-1, :] = 2*w2[-2, :] - w2[-3, :]

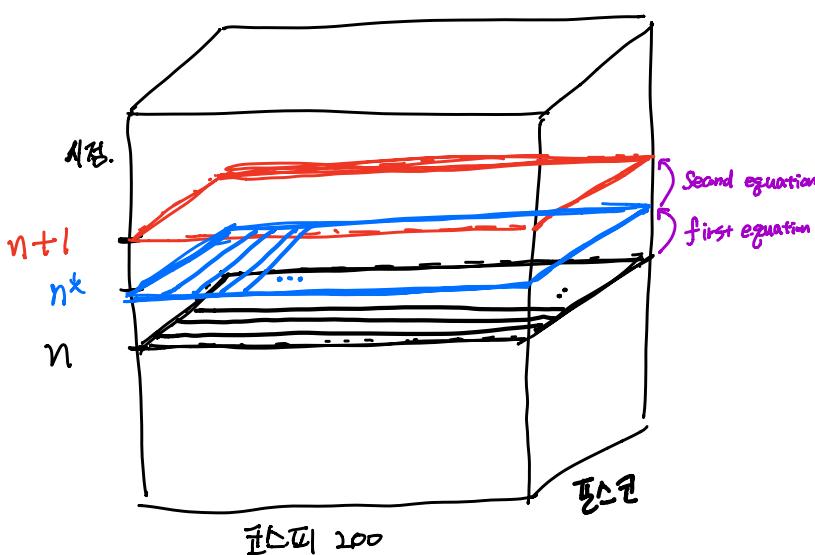
# u star barrier hit
u2[:, 1:Btag+1] = w2[:, 1:Btag+1]
u2[:, Btag+1:] = w2[:, 1:Btag+1, :]

```

```

# Second stage
for i in range(1, I): # X
    if method == 'ADI':
        d = u2[i, 1:-1] + dt * (r-D_1) * i/2 * (u2[i, 2:] - u2[i, 1:-1])
        + dt*(vol_1**2)/4*(i*i) * (u2[i, 2:]-2*u2[i, 1:-1]+u2[i, :-2])
        + dt*corr*vol_1*vol_2*tmp*i/8*(u2[i+1, 2:] + u2[i-1, :-2]
        - u2[i+1, :-2] - u2[i-1, 2:])
        d2 = w2[i, 1:-1] + dt * (r-D_1) * i/2 * (w2[i, 2:] - w2[i, 1:-1])
        + dt*(vol_1**2)/4*(i*i) * (w2[i, 2:]-2*w2[i, 1:-1]+w2[i, :-2])
        + dt*corr*vol_1*vol_2*tmp*i/8*(w2[i+1, 2:] + w2[i-1, :-2]
        - w2[i+1, :-2] - w2[i-1, 2:])
    elif method == 'OS':
        d = u2[i, 1:-1] + dt*corr*vol_1*vol_2*tmp*i/8
        *(u2[i+1, 2:] + u2[i-1, :-2] - u2[i+1, :-2] - u2[i-1, 2:])
        d2 = w2[i, 1:-1] + dt*corr*vol_1*vol_2*tmp*i/8
        *(w2[i+1, 2:] + w2[i-1, :-2] - w2[i+1, :-2] - w2[i-1, 2:])
    u[n+1, i, 1:-1] = Thomas_algol(a2_prime, b2_prime, c2_prime, d)
    w[n+1, i, 1:-1] = Thomas_algol(a2_prime, b2_prime, c2_prime, d2)

```



위 과정을 기하학적으로 표현한 것.

⑥ 현재가격(n)이 early redemption date인 경우

```
# fix over strike price at the early redemption date.
if n*dt in dates:
    dummy = int(round(S0_point * K[len(K)-1 - round((n+1)*dt*2)])) # dummy : strike price index
    print(int(round(S0_point * K[len(K)-1 - round((n+1)*dt*2)]))) # dummy : strike price index
    u[n+1, :, dummy : I+1] = 1 + cr[round(len(cr) - round((n+1)*dt, 1)*2)-1]
    w[n+1, :, dummy : I+1] = 1 + cr[round(len(cr) - round((n+1)*dt, 1)*2)-1]
    u[n+1, dummy : I+1, :] = 1 + cr[round(len(cr) - round((n+1)*dt, 1)*2)-1]
    w[n+1, dummy : I+1, :] = 1 + cr[round(len(cr) - round((n+1)*dt, 1)*2)-1]
    u[n+1, dummy :, dummy:] = 1 + cr[round(len(cr) - round((n+1)*dt, 1)*2)-1]
    w[n+1, dummy :, dummy:] = 1 + cr[round(len(cr) - round((n+1)*dt, 1)*2)-1]
```

2011 맞는 쿠폰을 지급!

⑦ Barrier 를 친 경우.

U 를 W 로 업데이트한다.

```
# barrier
u[n+1, :, 1:Btag+1] = w[n+1, :, 1:Btag+1]
u[n+1, 1:Btag+1, :] = w[n+1, 1:Btag+1, :]
```

⑧ Boundary Condition (Linear) 적용.

```
# boundary condition
u[n+1, 0, :] = 2 * u[n+1, 1, :] - u[n+1, 2, :]
u[n+1, :, 0] = 2 * u[n+1, :, 1] - u[n+1, :, 2]
u[n+1, :, I] = 2 * u[n+1, :, I-1] - u[n+1, :, I-2]
u[n+1, I, :] = 2 * u[n+1, I-1, :] - u[n+1, I-2, :]
w[n+1, 0, :] = 2 * w[n+1, 1, :] - w[n+1, 2, :]
w[n+1, :, 0] = 2 * w[n+1, :, 1] - w[n+1, :, 2]
w[n+1, :, I] = 2 * w[n+1, :, I-1] - w[n+1, :, I-2]
w[n+1, I, :] = 2 * w[n+1, I-1, :] - w[n+1, I-2, :]
```

⑨ 결과

S0에서 ELS가격 $\Rightarrow 9325.635$ 원.

유안타 증권의 추산가격 $\Rightarrow 8399$ 원

5. Simulate paths of underlying assets and calculate the price. Compare two prices with the price the bank suggests.

```

def ELS_TwoStar_Simulation(s0_1, s0_2, mu, vol_1, vol_2, corr,
                           r, D_1, D_2, cr, K, B, N, dates, T):
    s1 = np.zeros(N+1)
    s2 = np.zeros(N+1)
    x1 = np.zeros(N+1)
    x2 = np.zeros(N+1)
    dt = T/N
    x1[0] = 1
    x2[0] = 1
    s1[0] = s0_1
    s2[0] = s0_2
    cnt = 0
    total_iter = 50000
    sprice = 0
    for i in range(total_iter):
        if(i % (total_iter//10) == 0):
            print('iter: {}, {}% complete'.format(i, i/total_iter * 100))
        KI = 0 # Knock-In

        # 하나의 시나리오 만들어내기.
        for j in range(N):
            e = np.random.randn(2)
            s1[j+1] = s1[j] * np.exp((r-D_1-0.5*vol_1**2)*dt + np.sqrt(dt)*(vol_1*e[0]))
            s2[j+1] = s2[j] * np.exp((r-D_2-0.5*vol_2**2)*dt
                                      + np.sqrt(dt)*(corr*vol_2*e[0]+np.sqrt(1-corr**2)*vol_2*e[1]))

            x1[j+1] = s1[j+1]/s0_1
            x2[j+1] = s2[j+1]/s0_2

        # 시나리오의 상황을 체크한다.
        # 1. Barrier를 Knock-In 한적 있는가?
        if min(x1) <= B or min(x2) <= B:
            KI = 1
            cnt = cnt+1

        # 2. 첫번째 조기상환 기간에 조기상환 되었는가?
        if x1[N//6] >= K[0] and x2[N//6] >= K[0]:
            price = (1+cr[0]) * np.exp(-r*dates[0])
        # 3. 두번째 조기상환 기간에 조기상환 되었는가?
        elif x1[N//6*2] >= K[1] and x2[N//6*2] >= K[1]:
            price = (1+cr[1]) * np.exp(-r*dates[1])
        # 4. 세번째 조기상환 기간에 조기상환 되었는가?
        elif x1[N//6*3] >= K[2] and x2[N//6*3] >= K[2]:
            price = (1+cr[2]) * np.exp(-r*dates[2])
        # 5. 네번째 조기상환 기간에 조기상환 되었는가?
        elif x1[N//6*4] >= K[3] and x2[N//6*4] >= K[3]:
            price = (1+cr[3]) * np.exp(-r*dates[3])
        # 6. 다섯번째 조기상환 기간에 조기상환 되었는가?
        elif x1[N//6*5] >= K[4] and x2[N//6*5] >= K[4]:
            price = (1+cr[4]) * np.exp(-r*dates[4])
        # 7. 만기 상환관련 이슈. 만기애 가격이 행사가격보다 높은가?
        elif x1[N] >= K[5] and x2[N] >= K[5]:
            price = (1+cr[5]) * np.exp(-r*dates[5])
        # 8. 두 번째 만기 관련 이슈. 배리어를 친 적은 있는가? 그렇다면, 행사가격보다 낮기도 하기 때문에.. 낮은가격!
        elif KI == 1:
            price = min(x1[N], x2[N]) * np.exp(-r*dates[5])
        else: # 배리어를 친 적은 없고, 만기 가격이 (배리어 ~ 행사가격) 사이인 경우.
            price = (1+cr[5]) * np.exp(-r*dates[5])
        sprice = sprice + price
    sprice = sprice / total_iter
    prob = cnt / total_iter
    print('Simulation price and probability of KI: %.4f, %.4f \n\n' %(sprice, prob))

```

(시뮬레이션 결과.)

9213 원.

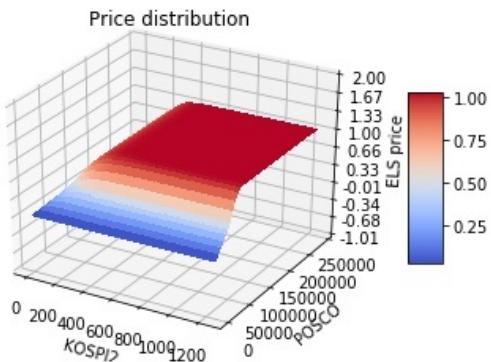
OS FDM과는 비슷하게 나왔지만,
주상가격과는 차이가 있다. 결과적으로

이론가격이 충분가격과 같지 나오지 않는다. 기억의 차이는,
증권사에서 변동성과 이자율을 상수로 사용하지 않아서
발생하는 것이다.

또한, 지주상가격을 매기면 당시의 Kospi200, 삼성전자
가격은 정확히 모르기 때문에 어떤 가격이 $S_1(t), S_2(t)$ 인지가
불확실해 발생하는 차이도 있을 것이다.
주상가격은 삼성화재 당시의 posco, Kospi200 가격을 반대면,
증거 원리를滿足할 수 있을 것이다.

6. Calculate Greeks and explain price movement according to Greeks

< Price graph >



< Greeks > $\rightarrow S_x(0), S_y(0)$ 기준!

① X (KOSPI 200)

- delta : 5.5604
- gamma : -0.19366
- Vega : -2649.06476

② Y (POSCO)

- delta : 0.069
- gamma : -5.56965
- Vega : -5316.127

③ Cross-gamma : 0.00020573

④ rho : -1153.1331 \rightarrow 이자율이 올라가면, ELS가격은 떨어짐.

⑤ theta : -1293.1483 \rightarrow 일자가 경과해지 않아서 사실상 크게 중요하지 않아보인다.

두 자산 모두 (+) 델타, (-) 감마, (-) 베가를
가지고 있다. 즉, ↘ 형태를 가진다
(위 price graph 참고)
기초자산 가격이 한수록 ELS의 가격도 올라가며,
변화점도는 감소로 돌아온다. ($\because (-)$ gamma)
또한, 상관관계가 높아서 ($\rho = 0.5831993444$)
POSCO 가격이 올랐을 때, KOSPI 200도 아마 올라갈 가능성이
있어서 생각보다 ELS 가격 변동폭이 클 것 같다.

\rightarrow yet 변화에 따른 차익과 손실이 (+) 지만
값이 크지 않다. (반대도 마찬가지)

7. You may add hedging analysis if you can.
8. Program codes should be handed in with documents together. OK.
9. Zip all files into one and upload. OK.

단지 잘 모르지만, 아마 기초자산 가격이 오르면, 그만큼
기초자산에 Short 포지션을 취해서 데이터증권 상태를 유지해야
할 것이다. Gamma가 높아지면 기초자산 가격이 변동될 때 put short 포지션은
취해야 할 것이다.. 갑연해지는 기초자산 가격이 변동될 때 put short 포지션은
취하고 드릴 것이다.