Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра математики факультету інформатики

Триноміальні моделі фінансового ринку: обчислення справедливої ціни

Курсова робота за спеціальністю „Системний аналіз” 124

Керівник курсової роботи

к.ф-м.н., доц. Щестюк Н.Ю.

____

(підпис)

“____” __________ 2021 р.

Виконала студентка Паук В.М.

“____” _________ 2021 р.

Київ 2021

Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра інформатики факультету інформатики

ЗАТВЕРДЖУЮ

Зав.кафедри математики,

Доцент, д. ф.-м. н. Б.В. Олійник

____

(підпис)

“____”__________________________2021 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

на курсову роботу

студентці Паук Вікторії Михайлівні

факультету інформатики МП 1 курсу

Тема: Триноміальні моделі фінансового ринку: обчислення справедливої ціни

1 Триноміальна модель. Порівняння з біноміальною

2 Визначення справедливої ціни для біноміальної моделі

3 Визначення справедливої ціни для триноміальної моделі

Дата видачі „____”_________ 2020 р.

Керівник _______________

(підпис)

Завдання отримала __________

(підпис)

Календарний план виконання роботи:

№ п/п	Назва етапу курсової роботи	Термін виконання етапу	Примітка
1.	Отримання завдання на курсову роботу.	22.10.2020	
2.	Огляд технічної літератури за темою роботи.	23.10.2020	
3.	Програмування практичної частини	14.03.2021	
4.	Написання пояснювальної роботи.	17.04.2021	
6.	Створення слайдів для доповіді та написання доповіді.	8.10.2021	
8.	Коригування роботи.	10.05.2021	
8.	Остаточне оформлення пояснювальної роботи та слайдів.	10.05.2021	
10.	Здача курсової роботи	11.05.2021	

Студент ___________________________________

Керівник __________________________________

“______”______________

Вступ

Базовою моделлю фінансового ринку для дискретного часу є біноміальна модель, що була запропонована Джоном Коксом, Стефеном Россом та Марком Рубінштейном у 1979 році [5]. Суть цієї моделі полягає в тому, що формується біноміальне дерево, яке описує рух акцій і за цим деревом знаходиться справедлива ціна.

Проте, існує альтернативна триноміальна модель для опису руху ризикових активів. Триноміальна модель має свої особливості, які є недостатньо визначеними.

Метою роботи є побудова триноміальної моделі фінансового ринку за реальними даними.

В першому розділі ми порівнюємо загалом чим відрізняється біноміальна та триноміальна моделі.

В другому розділі детальніше розглянемо біноміальну модель та наведемо формули для розрахунку справедливої ціни, а також ознайомимось з підходом зворотнього обходу дерева.

В третьому розділі розглянемо триноміальні дерева та на базі другого розділу наведемо формули для розрахунку опціону.

Ця робота буде корисна як з теоретичної точки зору, так і може стати в нагоді для інвесторів.

Триноміальні моделі. Порівняння з біноміальною моделлю

Одним із найбільш поширених і корисних методів оцінки опціонів є побудова біноміальних дерев. Біноміальні дерева - діаграми, що демонструють різні варіанти змін цін на акції протягом строку дії опціону. Цей метод заснований на припущенні, що ціна акції підкоряється законам випадкового блукання. На кожному проміжку часу є ймовірність, що ціна акції збільшиться або зменшиться на деяку відносну величину. Якщо величина часового кроку прагне до нуля, то можна допустити, що ціни акцій мають логонормальний розподіл, що лежить в основі моделі Блека-Шоулза.

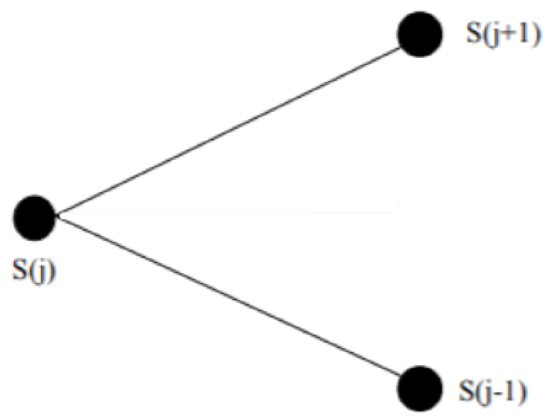


Рис.1 Одноперіодне біноміальне дерево

Модель побудови біноміальних дерев була найбільш поширеною деякий час, але останні 15 років вона дещо зменшила свою популярність. Більш вдосконалена модель - побудова триноміальних дерев. Оскільки в біноміальних деревах ціна опціона може тільки підвищуватись або зменшуватись, триноміальна модель допускає сталу ціну протягом деякого часового проміжку.

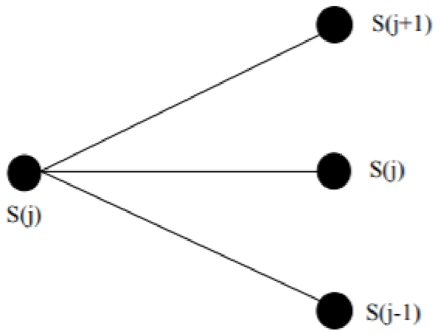
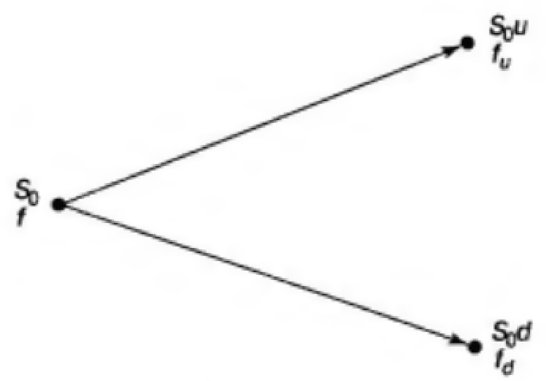


Рис.2 Одноперіодне триноміальне дерево

Визначення справедливої ціни для біноміальної моделі

Розглянемо детальніше як працює біноміальна модель. Нехай ціна акції рівна S0 і поточна вартість фондового опціону f. Припустимо, що за час дії опціону T ціна акції може або піднятися до величини S0u, u>1, або впасти до S0d, d<1. Якщо ціна акції збільшується до S0u, то вважається, що опціон приносить прибуток fu, якщо ж ціна знижується до S0d,то опціон приносить прибуток fd[2].



Уявимо собі інвестиційний портфель, який складається з довгої позиції на пакет з Δ акцій і короткої позиції по одному опціону. Обчислимо Δ .

$S_0 u\,\Delta - f_u = S_0 d\Delta - f_d$(1)

$\Delta = (f_u - f_d)/(S_0 u - S_0 d)$(2)

Нехай безризикова процентна ставка r. Тоді вартість портфеля дорівнює

$(S_0 u\,\Delta - f_u)e^{-rT}$(3)

Вартість створення портфеля дорівнює

$S_0\Delta - f$(4)

Звідси слідує, що

$$S_0\Delta - f = (S_0 u\Delta - f_u)e^{-rT}(5)$$

$$f = S_0\Delta(1 - ue^{-rT}) + f_u e^{-rT}(6)$$

Підставивши дельта і спростивши вираз, отримаємо

$$f = e^{-rT}(pf_u + (1 - p)f_d)(7)$$

де

$$p = (e^{rT} - d)/(u - d)(8)$$

Якщо розглядати біноміальні дерева вищих ступенів, то довжина кроку буде не Т, а Δt і формули видозмінюються таким чином

$$f = e^{-r\Delta t}(pf_u + (1 - p)f_d)(9)$$

$$p = (e^{r\Delta t} - d)/(u - d)(10)$$

Повторне застосування формул приводить до таких рівнянь:

$$f_u = e^{-r\Delta t}(pf_{uu} + (1 - p)f_{ud})(11)$$

$$f_d = e^{-r\Delta t}(pf_{ud} + (1 - p)f_{dd})(12)$$

Підставивши формули 11-12 в формулу 9, отримаємо

$$f = e^{-2r\Delta t}(p^2 f_{uu} + 2p(1 - p)f_{ud} + (1 - p)^2 f_{dd})(13)$$

Ця формула використовується для двоперіодних біноміальних дерев.

Вартість опціону обраховується за допомогою зворотнього обходу дерева. Ми знаємо вартість опціону в деякий момент Т, тоді в момент $T - \Delta t$

вартість опціону буде дорівнювати очікуваній вартості опціону в момент T, застосувавши ставку дисконту r[2].

В загальному випадку алгоритм знаходження справедливої ціни виглядає наступним чином[5]:

1 Побудова біноміального дерева

2 Знаходження значення опціону кінцевої ноди

(Для call формула має вигляд $max(S_n - K, 0)$, а для put $max(K - S_n, 0)$)

3 Знаходження значення опціону попередніх нод

$C_{t-\Delta t,i} = e^{-r\Delta t}(pC_{t,i} + (1 - p)C_{t,i+1})$, де $C_{t,i}$- значення опціону для і-ої ноди в момент часу t.

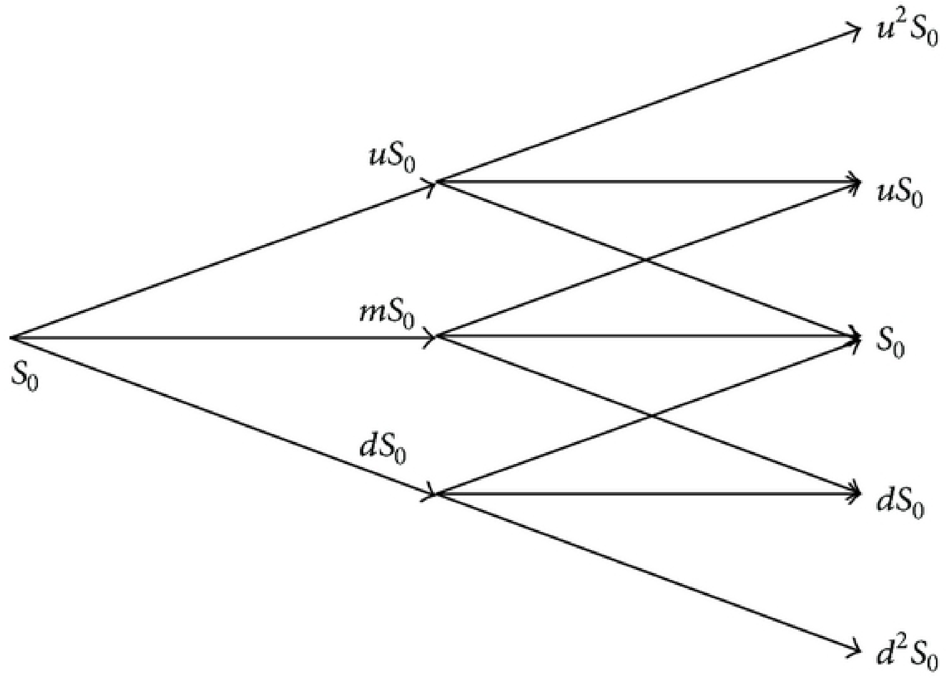
Реалізувавши даний алгоритм на програмі Python, розглянемо приклад обрахунку біноміального дерева для 10 періодів, де S0=100, sigma=0.1, gamma=0.05, K=100.

```
In [16]: binom_tree("C",10,1,100,0.1,0.05,100)[0]
Out[16]: 6.69921527613769
```

```
In [18]: binom_tree("P",10,1,100,0.1,0.05,100)[0]
Out[18]: 1.822157726209201
```

Визначення справедливої ціни для триноміальної моделі

Тепер розглянемо триноміальні дерева.



Оскільки тепер у нас є три варіанти розвитку подій, то маємо три ймовірності, які обчислюються за наступними формулами[1]:

$$p_u = \left(\frac{e^{r\Delta t/2} - e^{-\sigma\sqrt{\Delta t/2}}}{e^{\sigma\sqrt{\Delta t/2}} - e^{-\sigma\sqrt{\Delta t/2}}}\right)^2$$

$$p_d = \left(\frac{e^{\sigma\sqrt{\Delta t/2}} - e^{r\Delta t/2}}{e^{\sigma\sqrt{\Delta t/2}} - e^{-\sigma\sqrt{\Delta t/2}}}\right)^2$$

$$p_m = 1 - p_u - p_d$$

Причому

$u = e^{\sigma\sqrt{2\Delta t}}$

$d = e^{-\sigma\sqrt{2\Delta t}}$[4]

Обчислення за триноміальним деревом аналогічні до обчислень за біноміальним деревом. Обхід починається з кінця триноміального дерева до початку.

Вартість продовження володіння опціоном обчислюється як[]:

$e^{-r\Delta t}(p_u f_u + p_m f_m + p_d f_d)$,

де f_u- вартість опціону, що відповідає верхньому рівню, f_d- нижньому, а f_m-середньому.

В загальному випадку алгоритм знаходження справедливої ціни виглядає наступним чином:

1 Побудова триноміального дерева

2 Знаходження значення опціону кінцевої ноди

(Для call формула має вигляд $max(S_n - K, 0)$, а для put $max(K - S_n, 0)$)

3 Знаходження значення опціону попередніх нод

$C_{t-\Delta t,i} = e^{-r\Delta t}(p_u C_{t,i} + p_m C_{t,i+1} + p_d C_{t,i+2})$, де $C_{t,i}$- значення опціону для i-ої ноди в момент часу t[3].

Реалізувавши модель на Python отримали наступні значення:

```
In [60]: trinomial_call(100, 101, 0.1, 0.1, 5, 3)

Out[60]: 90.90029058373088
```

```
In [62]: trinomial_put(100, 101, 0.1, 0.1, 5, 3)

Out[62]: 0.023188707673861483
```

Висновок

Отже, ми розглянули підходи з галузі фінансової математики, що дозволяють визначити справедливу ціну.

В першому розділі ми розглянули різницю між триноміальним та біноміальним деревом. Різниця полягає в тому, що в триноміальному дереві розглядається можливість сталості ціни акцій, окрім підвищення чи пониження.

В другому розділі ми дослідили базову модель, що застосовується в фінансовій математиці для визначення опціону.

В третьому розділі розглянули визначення справедливої ціни, маючи побудоване триноміальне дерево.

Джерела:

1
https://warwick.ac.uk/fac/sci/maths/people/staff/oleg_zaboronski/fm/trinomial_tree_2008.pdf

2 Hull, John C. (2002). *Options, Futures and Other Derivatives* (5th ed.). Prentice Hall.

3
https://warwick.ac.uk/fac/sci/maths/people/staff/oleg_zaboronski/fm/trinomial_tree_2009.pdf

4 http://publications.lib.chalmers.se/records/fulltext/238499/238499.pdf

5 https://en.wikipedia.org/wiki/Binomial_options_pricing_model

Додаток:

```
import numpy as np
def binom_tree(type,N,T,S0,sigma,r,K):
    dt=T/N
    u=np.exp(sigma*np.sqrt(dt))
    d=1/u
    p=(np.exp(r*dt)-d)/(u-d)

    tree=np.zeros([N+1,N+1])

    for i in range(N+1):
        for j in range(N+1):
            tree[j,i]=S0*(d**j)*(u**(i-j))

    option=np.zeros([N+1,N+1])
    if type=='C':
        option[:,N]=np.maximum(np.zeros(N+1),tree[:,N]-K)
    elif type=='P':
        option[:,N]=np.maximum(np.zeros(N+1),K-tree[:,N])

    for i in np.arange(N-1,-1,-1):
        for j in np.arange(0,i+1):
            option[j,i]=np.exp(-r*dt)*(p*option[j,i+1]+(1-p)*option[j+1,i+1])
    return [option[0,0],price_tree]



def trinomial_call(S0, K, r, sigma, T, N=2000):
    t = float(T) / N
```

```
    u = np.exp(2 * sigma * np.sqrt(t))
    d = 1.0 / u

    f1 =  np.asarray([0.0 for i in range(2*N + 1)])
    f2 = np.asarray([(S0 * u**j) for j in range(-N, N+1)])
    f3 =np.asarray( [float(K) for i in range(2*N + 1)])


    p_u = ( ( np.exp(r*t/2.0) - np.exp(-sigma * np.sqrt(t/2.0)) ) / ( np.exp(sigma *
np.sqrt(t/2.0)) - d ) )**2
    p_d = ( ( np.exp(sigma * np.sqrt(t/2.0)) - np.exp(r*t/2.0) ) / ( np.exp(sigma *
np.sqrt(t/2.0)) - np.exp(-sigma * np.sqrt(t/2.0)) ) )**2
    p_m = 1 - p_u - p_d

    f1[:] = np.maximum(f2-f3, 0.0)

    for i in range(N-1, -1, -1):
        f1[:-1] = np.exp(-r * t) * (p_u * f1[1:] + p_d * f1[:-1] + p_m*f1[-1])

    return f1[0]



def trinomial_put(S0, K, r, sigma, T, N=2000):
    t = float(T) / N
    u = np.exp(2 * sigma * np.sqrt(t))
    d = 1.0 / u

    f1 =  np.asarray([0.0 for i in range(2*N + 1)])
```

```
    f2 = np.asarray([(S0 * u**j) for j in range(-N, N+1)])
    f3 =np.asarray( [float(K) for i in range(2*N + 1)])

    p_u = ( ( np.exp(r*t/2.0) - np.exp(-sigma * np.sqrt(t/2.0)) ) / ( np.exp(sigma * np.sqrt(t/2.0)) - d ) )**2
    p_d = ( ( np.exp(sigma * np.sqrt(t/2.0)) - np.exp(r*t/2.0) ) / ( np.exp(sigma * np.sqrt(t/2.0)) - np.exp(-sigma * np.sqrt(t/2.0)) ) )**2
    p_m = 1 - p_u - p_d

    f1[:] = np.maximum(-f2+f3, 0.0)

    for i in range(N-1, -1, -1):
        f1[:-1] = np.exp(-r * t) * (p_u * f1[1:] + p_d * f1[:-1] + p_m*f1[-1])

    return f1[0]
```