Міністерство освіти і науки України
Національний університет «Києво-Могилянська академія»
Факультет інформатики
Кафедра математики

Магістерська робота

освітній ступінь – магістр

на тему: «**СУБДИФУЗІЙНЕ МОДЕЛЮВАННЯ ДИНАМІКИ НЕЛІКВІДНИХ ФІНАНСОВИХ РИНКІВ»**

Виконав: студент 2-го року навчання,

освітньо-наукової програми «113 Прикладна математика»

Сердюк Федір Олексійович

Керівник: Щестюк Н.Ю., кандидат фізико-математичних наук, доцент

Рецензент:_____

Кваліфікаційна робота захищена з оцінкою _____

Секретар ЕК _____

«___» _____20___р.

Київ – 2025

Національний університет «Києво-Могилянська академія»

Факультет інформатики

Кафедра математики

Освітній ступінь магістра

Спеціальність 113 Прикладна математика

Освітньо-наукова програма “Прикладна математика”

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____

_____

«___»_____20__ року

ЗАВДАННЯ

ДЛЯ МАГІСТЕРСЬКОЇ РОБОТИ СТУДЕНТУ

Сердюку Федору Олексійовичу

1. Тема роботи: «Субдифузійне моделювання динаміки неліквідних фінансових ринків»

   керівник роботи: Щестюк Наталія Юріївна, кандидат фізико-математичних наук, доцент

   затверджені наказом вищого навчального закладу

   від «___»_____20__ року № ___
2. Строк подання студентом роботи:
3. План роботи:
    1. Вступ і аналіз класичної моделі Блека–Шоулза
    2. Теоретичне обґрунтування субдифузійних підходів
    3. Побудова субдифузійних моделей з IG-субординатором
    4. Уточнення моделей за допомогою локальної волатильності
    5. Реалізація та тестування моделей на реальних даних
    6. Висновки
    7. Список використаних джерел
    8. Додатки

ГРАФІК ПІДГОТОВКИ КВАЛІФІКАЦІЙНОЇ/МАГІСТЕРСЬКОЇ РОБОТИ ДО ЗАХИСТУ

№ з/п	ПЕРЕЛІК РОБІТ	Термін виконання	Дата ознайомлення наукового керівника	Підпис наукового керівника	Примітки
1.	Вибір теми, затвердження її на засіданні кафедри та закріплення наукового керівника Узгодження календарного графіка підготовки кваліфікаційної роботи. Ознайомлення студента з критеріями оцінювання кваліфікаційної роботи (п. 8.5).	жовтень			
2.	Вивчення джерел літератури, матеріалів архівів, періодичних видань, збір та узагальнення фактів, даних	жовтень – листопад			
3.	Складання плану каліф. роботи та узгодження з науковим керівником	листопад			
4.	Написання розділів роботи	листопад – березень			
5.	Проміжний контроль виконання роботи	лютий			
6.	Написання кваліфікаційної роботи в цілому, ознайомлення з її першим варіантом наукового керівника	січень – березень			
	Розділ 1 (постановка проблеми, теоретичні основи, огляд літературних джерел)	квітень			
	Розділ 2 (аналітично-дослідницька частина) (експериментальна частина для природничих і біологічних наук)	квітень			
	Розділ 3 (проектно-рекомендаційна частина) (аналіз результатів експерименту для природничих і біологічних наук)	квітень			
7.	Повне завершення написання кваліфікаційної роботи, оформлення її згідно з вимогами й подання на відгук науковому керівнику	квітень – травень			
8.	Подання кваліфікаційної роботи для перевірки письмових робіт студентів НаУКМА на відповідність вимогам академічної доброчесності,	травень			
9.	Подання на зовнішню рецензію	травень			
10.	Підготовка до захисту кваліфікаційної роботи на засіданні кафедри: написання доповіді та виготовлення ілюстративного матеріалу	до 29 травня			
11.	Попередній захист кваліфікаційної роботи на засіданні кафедри	до 29 травня			
12.	Подання кваліфікаційної роботи на кафедру з усіма супроводжувальними документами	до 8 червня			
13.	Публічний захист кваліфікаційної роботи перед екзаменаційною комісією	13 червня			

Графік узгоджено «___»_____2025р
Науковий керівник _____(ПІБ)
Виконавець магістерської роботи _____(ПІБ)

ЗМІСТ

АНОТАЦІЯ

Кваліфікаційна робота присвячена дослідженню динаміки неліквідних фінансових ринків за допомогою субдифузійного моделювання. У якості субординатора було використано стандартний інверсний гаусівський процес IG(0,1), що дозволяє враховувати нетривіальну часову структуру та затримки, які характерні для ринків із низькою ліквідністю.

До оцінювання опціонів було застосовано два методи: перший базується на класичному підході Магдіазера; другий — на фрактальному рівнянні Дюпіра отриманому шляхом введення дробової похідної Капуто–Джербашяна по часовій змінній. Створено програмний продукт для обчислення справедливої ціни європейських опціонів двома методами. Проведено порівняння отриманих результатів з моделлю Блека–Шоулза на основі реальних даних компанії AMD із використанням сталої та локальної волатильності.

ВСТУП

Актуальність теми:

Математичне моделювання фінансових ринків є важливим і активно досліджуваним напрямом прикладної математики, що має широкий спектр застосувань у фінансовій аналітиці, ризик-менеджменті та ціноутворенні фінансових інструментів. Воно забезпечує формалізовану основу для опису, аналізу та прогнозування поведінки ринків, дозволяючи приймати обґрунтовані рішення в умовах невизначеності. Однією з центральних задач фінансової математики є побудова моделей для оцінки справедливої вартості деривативів, зокрема опціонів.

Класичною та найбільш відомою моделлю в цьому контексті є модель Блека–Шоулза, яка стала фундаментом сучасної теорії ціноутворення опціонів. Вона ґрунтується на припущеннях про безперервність торгівлі, досконалу ліквідність ринку та броунівську природу динаміки цін активів. Проте в реальних ринкових умовах ці припущення часто не виконуються. Особливо це стосується ринків із низькою ліквідністю, де спостерігаються затримки в русі ціни, а також періоди малої або нульової торговельної активності. У таких ситуаціях традиційні моделі, побудовані на базі стандартних броунівських процесів, виявляються неспроможними для адекватного опису ринкової динаміки.

Це зумовлює необхідність розробки альтернативних підходів до моделювання, які враховують складну часову структуру процесів та нетривіальну поведінку цін у середовищі з обмеженою ліквідністю. Одним із таких перспективних підходів є субдифузійне моделювання, що базується на використанні стохастичних часових змінних — субординаторів — та відповідних до них обернених процесів. Такий підхід дозволяє гнучко модифікувати часову шкалу моделі, відображаючи ефекти «застою» або уповільнення динаміки, що є характерними для реальних фінансових ринків(див. [1], [3], [4], [5]).

Крім того, важливим кроком до вдосконалення моделей ціноутворення є відмова від припущення про сталу волатильність. На практиці волатильність змінюється залежно від ринкової ситуації, тому доцільно враховувати її залежність від часу та ціни активу. У зв'язку з цим виникає ідея поєднання субдифузійних підходів з локальною волатильністю[3], що відкриває нові можливості для побудови більш точних і гнучких моделей.

Об'єкт дослідження:

Фінансові ринки з обмеженою ліквідністю та механізми ціноутворення похідних фінансових інструментів, зокрема європейських кол-опціонів.

Предмет дослідження:

Субдифузійні IG(0, 1) моделі динаміки цін активів з локальною волатильністю та їх застосування до ціноутворення опціонів на неліквідних ринках.

Мета роботи:

Побудувати, реалізувати й проаналізувати субдифузійні підходи до ціноутворення європейських кол-опціонів на неліквідних ринках та порівняти їх точність із класичною моделлю Блека–Шоулза.

Завдання:

1. Проаналізувати класичну модель Блека–Шоулза та її обмеження в умовах неліквідних ринків.
2. Розглянути теоретичні передумови субдифузійних моделей.
3. Побудувати субдифузійну модель для ризикових активів з інверсно-гауссівським субординатором.
4. Для оцінювання опціонів розглянути класичний підхід Магдіазера та підхід на основі модифікованого рівняння Дюпіра, замінивши класичну похідну за часом на дробову похідну Капуто–Джербашяна. Розробити чисельні схеми для їх розв'язання. Для їх реалізації створити програмний продукт на мові Python.

5. Використати локальну волатильність для уточнення субдифузійних моделей, щоб покращити їхню відповідність реальній ринковій динаміці.
6. Провести чисельні експерименти на реальних ринкових даних, здійснити оцінку точності побудованих моделей та порівняти отримані результати з класичною моделлю Блека–Шоулза.

Наукова новизна отриманих результатів:

У роботі реалізовано два підходи до субдифузійного моделювання цін опціонів з урахуванням ефекту неліквідності ринку. Обидва підходи беруть до уваги локальну волатильність. Для випадку IG(0,1) сформульовано твердження про вигляд фрактального рівняння Дюпіра у якому похідна Капуто–Джербашяна перетворюється на похідну Капуто, що відкриває можливості застосування відомих чисельних методів. Проведено реалізацію моделей із використанням символьних і чисельних методів, створено програмний продукт і показано переваги субдифузійних моделей над класичними для реальних фінансових даних.

Практичне значення отриманих результатів:

Розроблені моделі можуть бути використані для більш точної оцінки вартості опціонів на ринках із низькою ліквідністю. Врахування субдифузійних ефектів і локальної волатильності дозволяє побудувати більш адекватні моделі ціноутворення, які враховують реальні ринкові умови. Створене програмне забезпечення може слугувати інструментом для подальших досліджень у фінансовій математиці, а також бути корисним у прикладних задачах фінансового аналізу та ризик-менеджменту.

Методи дослідження:

У роботі застосовано аналітичні підходи та методи наближених обчислень. Реалізацію моделей виконано мовою програмування Python із використанням бібліотек SymPy (для символьних обчислень), NumPy та SciPy (для чисельного

аналізу), CuPy (для прискорення обчислень на графічному процесорі), а також Joblib (для паралельних обчислень і кешування результатів).

Апробація результатів магістерської роботи:

Матеріали кваліфікаційної роботи доповідались на науковій конференції «XIII Всеукраїнська наукова конференція молодих математиків» 9 травня 2025 року[9]. Крім того, за результатами дослідження підготовлено статтю, яку подано до публікації у збірнику матеріалів міжнародної конференції «*International Conference of the Mathematical and Statistical Methods for Actuarial Sciences and Finance (MAF 2025)»[10].*

Структура роботи:

Магістерська кваліфікаційна робота складається зі вступу, трьох розділів основної частини, висновків, списку використаних джерел та додатків.

У першому розділі подано теоретичні основи математичного моделювання фінансових ринків, зокрема розглянуто класичну модель Блека–Шоулза.

Другий розділ присвячено субдифузійному підходу до моделювання динаміки активів. Розглянуто введення змінного часу за допомогою субординатора та два підходи для оцінювання опціонів:

- класичним методом (Спосіб 1);
- через фрактальне рівняння Дюпіра (Спосіб 2).

У третьому розділі реалізовано чисельні обчислення та проведено порівняльний аналіз моделей для реальних фінансових даних компанії AMD. Показано вплив вибору сталої та локальної волатильності на точність ціноутворення опціонів.

РОЗДІЛ 1. Теоретичні основи моделювання фінансових ринків

Однією з базових моделей математичного фінансового ринку є модель Блека–Шоулза, яка широко використовується для ціноутворення опціонів. У цьому розділі розглянуто класичний підхід до моделювання фінансових активів, що лежить в основі цієї моделі, а також базові припущення щодо структури ринку та динаміки його елементів.

Припустимо, що фінансовий ринок складається з трьох базових інструментів:

- **безризикового активу** $B(t)$, наприклад, банківського депозиту чи облігації;
- **ризикового активу** $S(t)$, як-от акції;
- **деривативу** — зокрема європейського опціону на купівлю або продаж активу.

Безризиковий актив:

Одним із ключових припущень моделі Блека–Шоулза є наявність на ринку безризикового активу — фінансового інструменту, який забезпечує стабільну дохідність і не зазнає коливань вартості.
У реальних умовах такими активами вважають, наприклад, банківські депозити або державні облігації з фіксованою ставкою.

Вважається, що ціна безризикового активу B_t зростає експоненціально зі сталою процентною ставкою r, відповідно до диференціального рівняння

$$\frac{dB_t}{dt} = rB_t, B_0 = 1$$

розв’язком якого є

$$B_t = e^{rt}.$$

Ризиковий актив (акція):

Динаміка ціни ризикового активу S_t у моделі Блека–Шоулза описується стохастичним диференціальним рівнянням (СДР) першого порядку

$$\frac{dS_t}{S_t} = \left(r + \frac{\sigma^2}{2}\right)dt + \sigma dW_t, S_0 > 0,$$

де:

- r — безризикова процентна ставка,
- σ— волатильність активу,
- W_t — стандартний броунівський рух.
- S_0 — початкова ціна активу.

Розв'язок цього рівняння знаходиться шляхом застосування леми Ітo до логарифму ціни активу, що дозволяє отримати

$$S_t = S_0 e^{rt+\sigma W_t},$$

де S_0 — початкова ціна активу.

Дериватив (європейський кол-опціон):

Європейський кол-опціон є одним із найпоширеніших типів фінансових деривативів. Він надає інвестору право (але не зобов'язання) придбати базовий актив за фіксованою ціною виконання K у визначений момент часу T. Якщо на дату погашення ціна активу S_T перевищує ціну виконання, опціон виконується і приносить прибуток $S_T - K$. Якщо ж на момент погашення ціна активу не перевищує страйкову ціну, виконання опціону є недоцільним, і його вартість у цей момент дорівнює нулю. Таким чином, виплата такого опціону описується функцією

$$(S_T - K)^+ = \max(S_T - K, 0).$$

Виходячи з рівняння для динаміки ціни ризикового активу та побудови хеджованого портфеля, справедлива вартість європейського кол-опціону $C(S,t)$ задовольняє наступне частинне диференціальне рівняння

$$\frac{dC}{dt} + rS\frac{dC}{dS} + \frac{1}{2}\sigma^2 S^2 \frac{d^2C}{dS^2} = rC.$$

Розв’язання цього рівняння з граничною умовою приводить до формули Блека–Шоулза

$$C = S_0\Phi(d_1) - Ke^{-rT}\Phi(d_2),$$

де

$$d_1 = \frac{\log\left(\frac{S_0}{K}\right) + rT + \frac{1}{2}\sigma^2 T}{\sigma\sqrt{T}}, d_2 = \frac{\log\left(\frac{S_0}{K}\right) + rT - \frac{1}{2}\sigma^2 T}{\sigma\sqrt{T}},$$

а $\Phi(\cdot)$ стандартна функція кумулятивного нормального розподілу.

Таким чином, модель Блека–Шоулза є найважливішою у теорії фінансової математики та широко використовується для оцінки вартості опціонів. Проте вона базується на низці припущень — зокрема про сталу волатильність та безперервну динаміку ціни активу, — які не завжди виконуються в умовах реального, особливо неліквідного, ринку.

РОЗДІЛ 2. Субдифузійні підходи до моделювання фінансових ринків

2.1 Введення субординатора та змінного часу

Класичні моделі фінансової математики, зокрема модель Блека–Шоулза, базуються на припущенні про високу ліквідність ринку та безперервну змінність цін фінансових активів. У таких моделях вважається, що учасники можуть укладати угоди у будь-який момент, а ціна активу змінюється постійно, відповідно до геометричного броунівського руху.

Проте на практиці часто спостерігаються періоди “застою”, коли ціна активу залишається незмінною протягом тривалого часу. Це характерно для неліквідних ринків, де угоди укладаються рідко, або для активів з обмеженим обігом, зокрема у позабіржовому середовищі. Прикладом такої поведінки є динаміка акцій Ryanair Holdings, зображена на рисунку 2.1, де видно тривалі інтервали стабільної ціни між окремими стрибками.

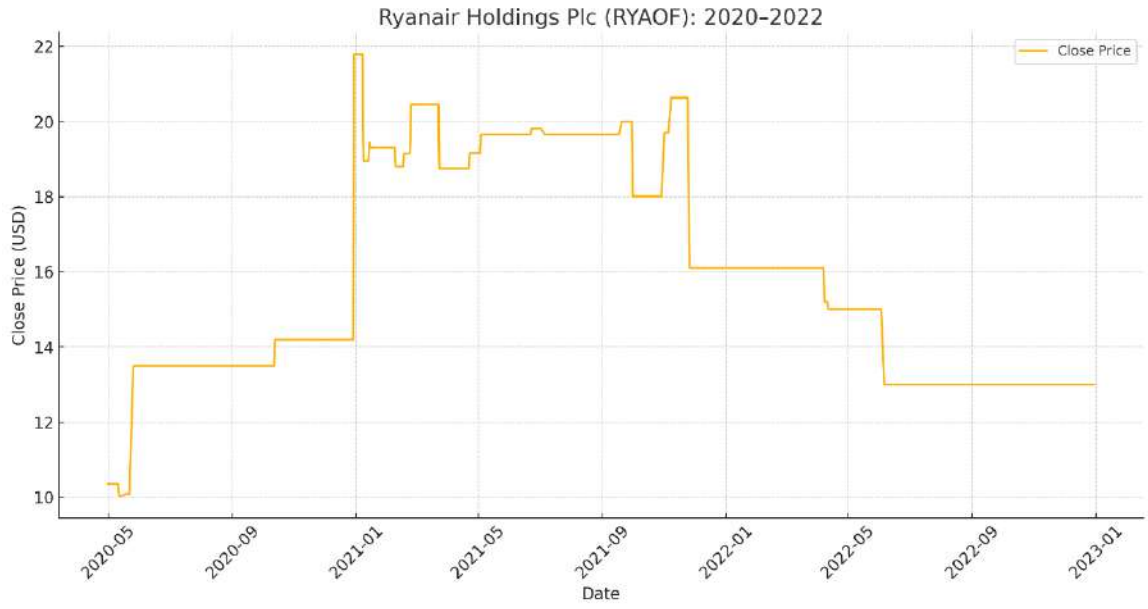


Рис. 2.1. Поведінка ціни активу на реальному ринку (Ryanair Holdings Plc, 2020–2022)

Подібна поведінка має прямі аналогії у статистичній фізиці. Наприклад, для опису повільного транспорту носіїв заряду в аморфних напівпровідниках або руху частинок у неоднорідних середовищах використовують моделі субдифузії.

У таких системах частинка може потрапити в пастку і залишатися у ній тривалий час, що уповільнює загальну динаміку процесу. Подібним чином, у фінансах можна спостерігати “зависання” ціни активу — коли вона тривалий час залишається сталою, незважаючи на те, що календарний час минає. У фінансових моделях подібні ефекти доцільно враховувати через введення випадкового часу, який дозволяє описати затримки в динаміці активу.

У класичних підходах час вважається лінійним і рівномірним, тоді як на практиці між подіями на ринку можуть виникати тривалі паузи — періоди, коли жодних угод не відбувається. Це суперечить припущенням броунівського руху. Щоб краще описати подібні ефекти, пропонується заміна фізичного часу t на операційний (випадковий) час, що зростає лише тоді, коли відбувається фактична зміна на ринку. Така заміна дозволяє вбудувати саму структуру ринку з періодами бездіяльності у часову вісь моделі.

Одним зі способів реалізації цієї ідеї є введення нового стохастичного часу([6], [7]), який визначається через допоміжний зростаючий випадковий процес $G(\tau)$, представлений на рисунку 2.2., що називається субординатором. Такий процес виконує роль випадкової часової шкали, яка зростає лише в моменти появи подій.

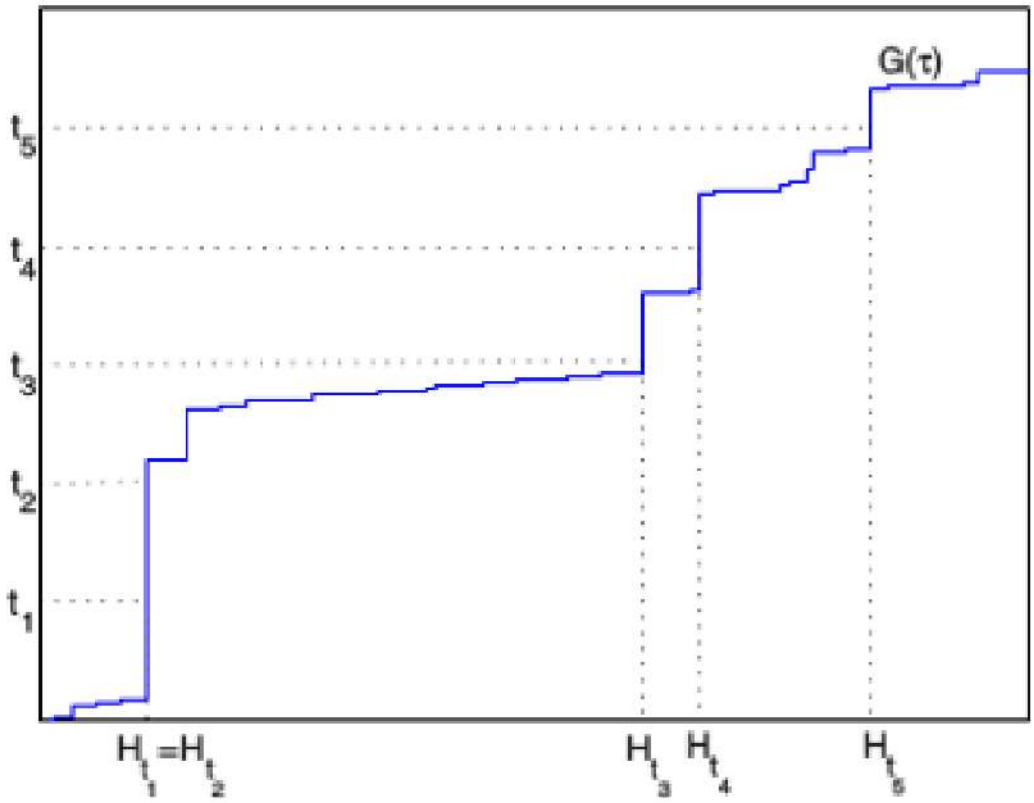


Рис. 2.2. Стохастичний процес G(t)

На його основі визначається **обернений субординатор** (або hitting time)

$$H(t) = \inf(\tau > 0 : G(\tau) > t)),$$

який вказує на перший момент, коли значення процесу $G(\tau)$ перевищує фіксовану календарну мітку часу t. Таким чином, якщо на ринку певний час нічого не відбувається, процес $H(t)$ просто не змінюється — він «чекає», поки субординатор не наздожене календарний час. Це дозволяє природним чином врахувати затримки або фази бездіяльності у динаміці активів.

У цій новій часовій шкалі динаміка безризикового активу моделюється як

$$B_{H_t} = B_0 e^{rH_t}.$$

Аналогічно модифікується і ризиковий актив

$$S_{H_t} = S_0 e^{\mu H_t + \sigma W_{H_t}}.$$

2.2 Оцінювання опціонів класичним методом (Спосіб 1)

Одним із ефективних підходів до субдифузійного моделювання цін опціонів є метод запропонований Магдіазером[1]. Він ґрунтується на заміні календарного часу у класичній моделі Блека–Шоулза на випадкову часову шкалу, представлену оберненим субординатором.

Справедлива ціна європейського кол-опціону в такій моделі визначається наступною формулою

$$C_{H_t}(S,K,T,\sigma) = \int_0^\infty C(S,K,x,\sigma)h_\Psi(x,T)dx,$$

де $C(S,K,x,\sigma)$ — ціна опціону за моделлю Блека–Шоулза, але з часовим параметром x, $h_\Psi(x,T)$ — щільність ймовірності hitting time (оберненого субординатора) H_t. Такий підхід дозволяє врахувати часову випадковість, що виникає через періоди інерції чи застою на ринку.

Субординатор G_t у цьому підході є неспадним процесом Леві, прирости якого мають інверсно-гауссівський розподіл $\varrho(\delta t,\ \gamma)$. Його щільність задається формулою[7]

$$g(x,t) = \frac{\delta t}{\sqrt{2\pi x^3}} e^{\delta\gamma t-(\delta^2 t^2/x+\gamma^2 x)/2},\ x>0.$$

Обернений до нього процес H_t, хоча й не є процесом Леві, має монотонно зростаючі траєкторії([2], [7]). Його щільність імовірності визначається виразом

$$h_\Psi(x,t) = \frac{\delta}{\pi} e^{\delta\gamma x-\frac{\gamma^2}{2}} \int_0^\infty \frac{e^{-ty}}{y+\frac{\gamma^2}{2}} \left(\gamma\, sin\left(\delta x\sqrt{2y}\right)+\sqrt{2y}\cos\left(\delta x\sqrt{2y}\right)\right)dy.$$

Математичне сподівання та дисперсія hitting time H_t мають асимптотичні оцінки[7]

$$E(H_t) \sim \begin{cases} \left(\frac{\gamma}{\delta}\right) t, \ \gamma > 0 \\ \left(\frac{1}{\delta}\sqrt{\frac{2t}{\pi}}\right) t, \ \gamma = 0 \end{cases}, Var(H_t) \sim \left(\frac{\gamma}{\delta}\right)^2 t^2.$$

У дослідженні також розглядається так званий стандартний випадок, коли параметри процесу мають $\delta = 1, \gamma = 0$. У цьому випадку щільність $h_\Psi(x,t)$ набуває вигляду

$$h_\Psi = \sqrt{\frac{2}{\pi t}} e^{-\frac{x^2}{2t}},$$

а формула для вартості опціону спрощується до

$$C_{H_t}(S,K,T,\sigma) = \int_0^\infty C(S,K,x,\sigma) \sqrt{\frac{2}{\pi t}} e^{-\frac{x^2}{2t}} dx.$$

Цей аналітичний вираз дає змогу застосовувати методи символьних обчислень для визначення справедливої вартості опціону.

Обчислення було реалізовано в середовищі Python із використанням бібліотеки SymPy, яка забезпечує символьну побудову інтегралів та обчислення значень без необхідності чисельного апроксимаційного інтегрування. Для кожного моменту часу $T = \frac{i}{252}$, де $i = 1,2,\ldots,252$, інтеграл формувався аналітично, після чого до нього підставлялися фіксовані параметри моделі.

З метою прискорення обчислень, побудований обчислювальний цикл було реалізовано з використанням паралельної обробки (бібліотека joblib), що дало змогу ефективно отримати серію оцінених значень справедливої вартості опціону для повного діапазону календарних термінів. Результати цього моделювання використовуються у наступному розділі для емпіричного порівняння з іншими підходами до ціноутворення.

2.3 Оцінювання опціонів через фрактальне рівняння Дюпіра (Спосіб 2)

Другий підхід до субдифузійного моделювання ціни опціонів ґрунтується на модифікації класичного рівняння Дюпіра[3]. Це рівняння широко використовується для визначення вартості опціонів у випадку, коли волатильність залежить від ціни базового активу та часу. Його класичний вигляд наступний

$$\frac{\partial}{\partial T}C(T,k) = -r\frac{\partial}{\partial k}C(T,k) + \frac{\sigma_t^2}{2}\frac{\partial^2}{\partial k^2}C(T,k).$$

На реальних фінансових ринках ціни активів часто демонструють затримки та періоди низької активності. Щоб врахувати ці ефекти, у класичному рівнянні Дюпіра похідна за часом замінюється дробовим інтегрально-диференційним оператором Капуто–Джербашяна. Цей оператор визначається наступним чином

$${}^{\Psi}Du(t) = \int_0^t \frac{\partial}{\partial t}u(t-s)v(s)ds\,;\ \Psi(x) = \int_0^{+\infty}(1-e^{-sz})\tilde{v}(dz).$$

У загальному випадку, коли субординатором є інверсно-гауссівський процес із параметрами γ та δ, відповідне ядро $v(s)$ має вигляд[4]

$$v(s) = \frac{\delta\gamma}{2\sqrt{\pi}}\Gamma\left(-\frac{1}{2}, \frac{s\gamma^2}{2}\right).$$

Таким чином, ціна європейського кол-опціону $C(T,k)$ визначається розв'язком наступного дробового рівняння

$${}^{\Psi}DC(T,k) = -r\frac{\partial}{\partial k}C_H(T,k) + \frac{\sigma_H^2}{2}\frac{\partial^2}{\partial k^2}C_H(T,k).$$

У розгорнутому вигляді рівняння має вигляд

$$\int_0^T \delta\frac{\partial}{\partial t}C_H(T-s,k)\left(\gamma\left(\Phi\left(\gamma\sqrt{s}\right)-1\right) + \frac{e^{-\frac{s\gamma^2}{2}}}{\sqrt{2\pi s}}\right)ds =$$

$$= -\frac{r}{2}\frac{\partial}{\partial k}C_H(T,k) + \frac{\sigma_H^2}{4}\frac{\partial^2}{\partial k^2}C_H(T,k).$$

У цій роботі розглядається стандартний випадок, коли $\gamma = 0, \delta = 1$. У цьому випадку рівняння набуває вигляду:

$$\int_0^T \delta \frac{\partial}{\partial t} C_H(T-s,k)\left(\frac{1}{\sqrt{2\pi s}}\right) ds = -\frac{r}{2}\frac{\partial}{\partial k} C_H(T,k) + \frac{\sigma^2}{4}\ \frac{\partial^2}{\partial k^2} C_H(T,k).$$

Це дозволяє спростити оператор до дробової похідної Капуто порядку $\alpha = \frac{1}{2}$. Отже, для параметрів IG(0, 1) справедливо наступне твердження.

Твердження. *Для стандартного випадку IG(0, 1) фрактальне рівняння Дюпіра набуває вигляду:*

$$\frac{\sqrt{2}}{2}\mathcal{D}_{1/2} C_H(T,k) = -\frac{r}{2}\frac{\partial}{\partial k} C_H(T,k) + \frac{\sigma_H^2}{4}\ \frac{\partial^2}{\partial k^2} C_H(T,k).$$

Для чисельного розв'язання застосовується апроксимація похідної Капуто(наприклад [3], [8]). У загальному вигляді вона записується як

$$\mathcal{D}_\alpha C_H(k,j) = \frac{(\Delta_t)^{-\alpha}}{\Gamma(1-\alpha)}\big(C_H(k,j) - C_H(k-1,j)\big) +$$

$$+\frac{(\Delta_t)^{-\alpha}}{\Gamma(1-\alpha)}\sum_{m=0}^{k-2}(k-m)^{-\alpha}(C_H(m+1,j) - C_H(m,j)).$$

У випадку $\alpha = \frac{1}{2}$, похідну можна подати в матричному вигляді

$$\frac{\sqrt{2}}{2}\mathcal{D}_{\frac{1}{2}} C_H(k,.)^\top = \frac{\sqrt{2}}{2}\frac{(\Delta_t)^{-\frac{1}{2}}}{\sqrt{\pi}}\big(C_H(k,.) - C_H(k-1,.)\big)^\top +$$

$$+\frac{\sqrt{2}}{2}\big(D(k)C_H(0\colon k-1,.)\big)^\top \mathbf{1}_{k-1},$$

де $\mathbf{1}_{k-1}$ - вектор одиниць, а матриця D(k) має розмірність $(k-1)\times k$ та елементи

$$D(k) = \frac{(\Delta_t)^{-\frac{1}{2}}}{\Gamma\left(\frac{1}{2}\right)}\begin{pmatrix} 0 & \cdots & \cdots & 0 & -(2)^{-\frac{1}{2}} & (2)^{-\frac{1}{2}} \\ \vdots & \ddots & 0 & -(3)^{-\frac{1}{2}} & (3)^{-\frac{1}{2}} & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ -(k)^{-\frac{1}{2}} & (k)^{-\frac{1}{2}} & \cdots & \cdots & \cdots & 0 \end{pmatrix}.$$

Для наближення похідних за страйком K застосовуються центральні різниці. Формули наближення мають вигляд[3]

$$\frac{\partial C_H(k,j)}{\partial K} \approx \frac{C_H(k,j+1) - C_H(k,k-1)}{2\Delta_K},$$

$$\frac{\partial C_H(k,0)}{\partial K} = \frac{C_H(k,1) - C_H(k,0)}{\Delta_K},$$

$$\frac{\partial C_H(k,n_K+1)}{\partial K} = \frac{C_H(k,n_K+1) - C_H(k,n_K)}{\Delta_K}$$

$$\frac{\partial^2 C_H(k,j)}{\partial K^2} \approx \frac{C_H(k,j+1) - 2C_H(k,j) + C_H(k,j-1)}{\Delta_K^2},$$

$$\frac{\partial^2 C_H(k,0)}{\partial K^2} = \frac{\partial^2 C_H(k,1)}{\partial K^2}, \frac{\partial^2 C_H(k,n_K+1)}{\partial K^2} = \frac{\partial^2 C_H(k,n_K)}{\partial K^2},$$

де $n_K = [\frac{K_{max}-K_0}{\Delta_K}]$. Крайові умови реалізуються шляхом дублювання значень похідних у найближчих внутрішніх точках.

У матричній формі похідні записуються як

$$\frac{\partial C_H(k,j)}{\partial K} = \overline{K} R_1 C_H(k,.)^\top, \frac{\partial^2 C_H(k,j)}{\partial K^2} = \overline{K}^2 R_2 C_H(k,.)^\top,$$

де $\overline{K} = diag(K_j)$, а матриці

$$R_1 = \frac{1}{\Delta_K}\begin{pmatrix} -1 & 1 & 0 & \cdots & 0 \\ -\frac{1}{2} & 0 & \frac{1}{2} & \cdots & 0 \\ 0 & -\frac{1}{2} & 0 & \frac{1}{2} & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & -1 & 1 \end{pmatrix}, \; R_2 = \frac{1}{\Delta_K^2}\begin{pmatrix} 1 & -2 & 1 & 0 & \cdots & 0 \\ 1 & -2 & 1 & 0 & \cdots & 0 \\ 0 & 1 & -2 & 1 & 0 & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 & -2 & 1 \\ 0 & \cdots & 0 & 1 & -2 & 1 \end{pmatrix}.$$

Після підстановки всіх складових маємо матричне рівняння виду

$$\overline{K} R_1 C_H(k,.)^\top \left(-\frac{r}{2}\right) + \overline{K}^2 R_2 C_H(k,.)^\top \left(\frac{\sigma_H^2}{4}\right) =$$

$$= \frac{\sqrt{2}}{2}\frac{(\Delta_t)^{-\frac{1}{2}}}{\sqrt{\pi}}\left(C_H(k,.) - C_H(k-1,.)\right)^\top + \frac{\sqrt{2}}{2}\left(D(k)C_H(0:k-1,.)\right)^\top \mathbf{1}_{k-1}.$$

Розв’язання цього рівняння здійснюється за допомогою ітераційної схеми

$$C_H(k,.)^\top = \left(\bar{K} R_1 \frac{r}{2} - \bar{K}^2 R_2 \frac{\sigma_H^2}{4} + I_{n_k+1} \frac{\sqrt{2}}{2} \frac{(\Delta_t)^{-\frac{1}{2}}}{\sqrt{\pi}} \right)^{-1} \times$$

$$\times \left(\frac{\sqrt{2}}{2} \frac{(\Delta_t)^{-\frac{1}{2}}}{\sqrt{\pi}} (D(k) C_H(k-1,.))^\top \mathbf{1}_{k-1} \right).$$

Оскільки ця чисельна схема ґрунтується на матричних операціях (добутках, транспонуванні, додаванні, інверсіях), її зручно реалізовувати на графічному процесорі (GPU). Це забезпечує значне прискорення обчислень і дає змогу ефективно обробляти великі обсяги даних при моделюванні субдифузійної динаміки в опціонному ціноутворенні. У цій роботі обчислення було реалізовано з використанням бібліотеки CuPy, яка забезпечує API, сумісний із NumPy, для виконання обчислень безпосередньо на GPU.

РОЗДІЛ 3. Застосування моделей до реальних даних

3.1 Стала волатильність

Для оцінки ефективності представлених моделей було проведено чисельне ціноутворення європейських кол-опціонів на акції компанії AMD, зафіксоване станом на 13.02.2025. Було використано такі параметри:

- Страйкова ціна опціону: $K = 111$,
- Поточна ціна акції: $S = 111$,
- Безризикова ставка: $r = 5\%$,
- Річна стала волатильність: $\sigma = 47.37\%$, розрахована на основі історичних ринкових даних.

Ціни опціонів були розраховані для кількох дат експірації з використанням трьох моделей:

- класичної моделі Блека–Шоулза,
- класичний субдифузійний підхід (Спосіб 1),
- субдифузійний підхід на основі модифікованого рівняння Дюпіра (Спосіб 2).

У таблиці 3.1 представлено порівняння отриманих результатів з реальними ринковими цінами опціонів, згідно з даними ресурсу Nasdaq.com.

Таб. 3.1 Табличне порівняння змодельованих цін з реальними ринковими(Стала волатильність)

Exp. Date	**Субдифузійна модель(Спосіб 1)**	**Субдифузійна модель(Спосіб 2)**	**Стандартна модель Блека-Шоулза**	**Реальна ціна**
2025-03-21	10.39	11.43	7.00	6.7
2025-04-17	11.99	13.33	9.31	8.58
2025-05-16	13.27	14.94	11.38	10.95
2025-06-20	14.44	16.53	13.47	12.75
2025-07-18	15.22	17.63	14.96	13.75
2025-08-15	15.90	18.64	16.33	15.45

Exp. Date	Субдифузійна модель(Спосіб 1)	Субдифузійна модель(Спосіб 2)	Стандартна модель Блека-Шоулза	Реальна ціна
2025-09-19	16.65	19.80	17.91	16.9
2025-10-17	17.18	20.66	19.09	18
2025-11-21	17.80	21.67	20.48	19.5
2025-12-19	18.25	22.43	21.54	20.4
2026-01-16	18.70	23.16	22.55	21.5

На рисунку 3.1. показано порівняння змодельованих цін з ринковими:

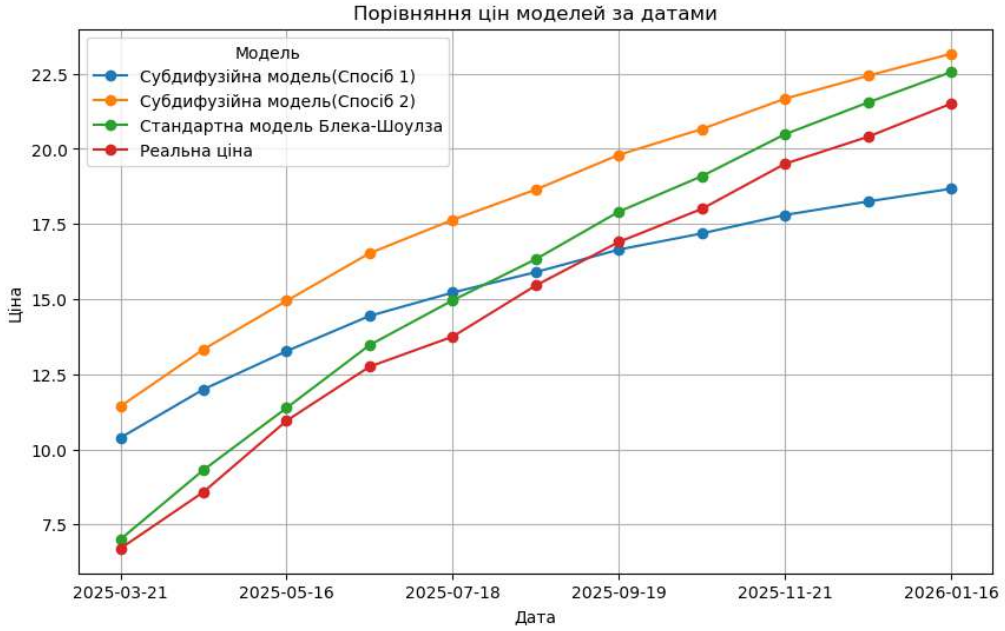


Рис. 3.1 Графічне порівняння змодельованих цін з реальними ринковими(Стала волатильність)

Модель Блека–Шоулза надає найкраще наближення до реальних цін у більшості випадків. Спосіб 1 показує точніше наближення у середньостроковому періоді (на горизонті близько пів року до експірації), але в коротко- та довгостроковому періодах дає гірші результати. Спосіб 2 систематично переоцінює ціни опціонів.

Для більш глибокої оцінки ефективності моделей було обчислено класичні метрики точності: MAE (середня абсолютна похибка), MSE (середньоквадратична похибка), RMSE (середньокоренева квадратична похибка) та ARPE (відносна похибка у відсотках). Розрахунки проведено окремо для середньострокового періоду (з 2025-07-18 по 2025-10-17) та всього періоду і представлені в таблиці 3.2.

Таб. 3.2. Метрики точності(Стала волатильність)

Період	**Модель**	**MAE**	**MSE**	**RMSE**	**ARPE**
2025-07-18 – 2025-10-17	Блек-Шоулз	1.044	1.105	1.051	0.066
	Субдифузійна модель 1	0.745	0.769	0.877	0.049
	Субдифузійна модель 2	3.157	10.179	3.190	0.202
Весь період	Блек-Шоулз	0.867	0.829	0.911	0.059
	Субдифузійна модель 1	1.888	4.735	2.176	0.165
	Субдифузійна модель 2	3.248	11.577	3.403	0.274

Значення метрик підтверджують попередні спостереження: класична модель Блека–Шоулза загалом забезпечує найменші похибки та найкраще наближення до реальних ринкових цін. Субдифузійна модель 1 демонструє покращену точність у середньостроковому інтервалі.

Для подальшої оцінки якості моделей було проведено статистичні тести на нормальність та автокореляцію залишків, отриманих як різниця між змодельованими та реальними ринковими цінами опціонів.

Тести на нормальність залишків:

- Блек-Шоулз:
    - Shapiro–Wilk p-value: 0.2104
    - Jarque–Bera p-value: 0.5264
- Субдифузійна модель 1:
    - Shapiro–Wilk p-value: 0.7144

  - Jarque–Bera p-value: 0.6941
- Субдифузійна модель 2:
  - Shapiro–Wilk p-value: 0.5873
  - Jarque–Bera p-value: 0.7010

Тести на автокореляцію залишків (Durbin–Watson):

- Блек-Шоулз: 0.085
- Субдифузійна модель 1: 0.096
- Субдифузійна модель 2: 0.013

За результатами тестів на нормальність, усі моделі демонструють залишки, які не суперечать гіпотезі про нормальний розподіл (p-value > 0.05), що свідчить про коректну побудову моделей з точки зору розподілу помилок. Проте значення статистики Durbin–Watson в усіх випадках значно нижчі за 2, що вказує на наявність сильної позитивної автокореляції залишків. Це означає, що моделі не повністю враховують часову структуру динаміки опціонних цін і можуть бути вдосконалені в цьому напрямку.

Для перевірки припущення про нормальність залишків було побудовано Q-Q графіки для кожної з моделей(рисунки 3.2., 3.3., 3.4.):

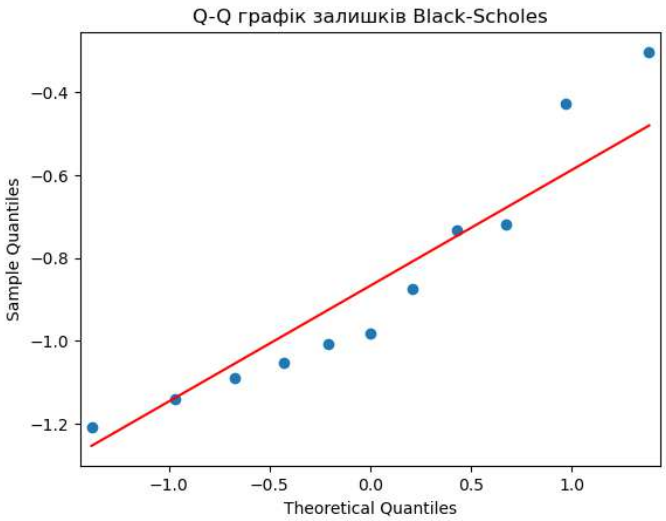


Рис. 3.2. Q-Q графік для Блека-Шоулза(Стала волатильність)

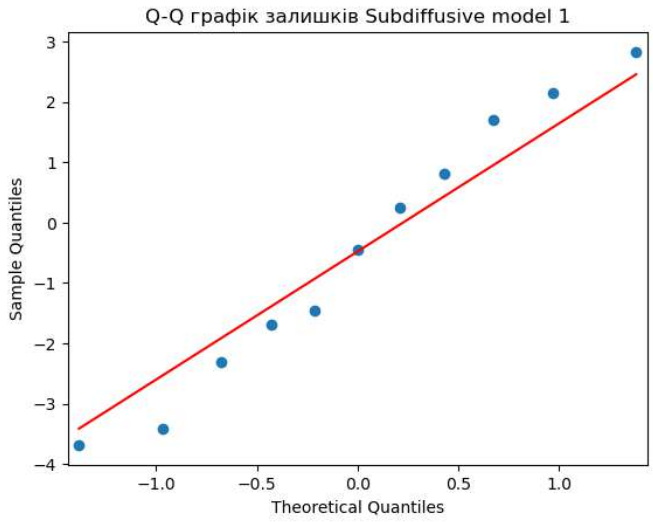


Рис. 3.3. Q-Q графік для Способу 1(Стала волатильність)

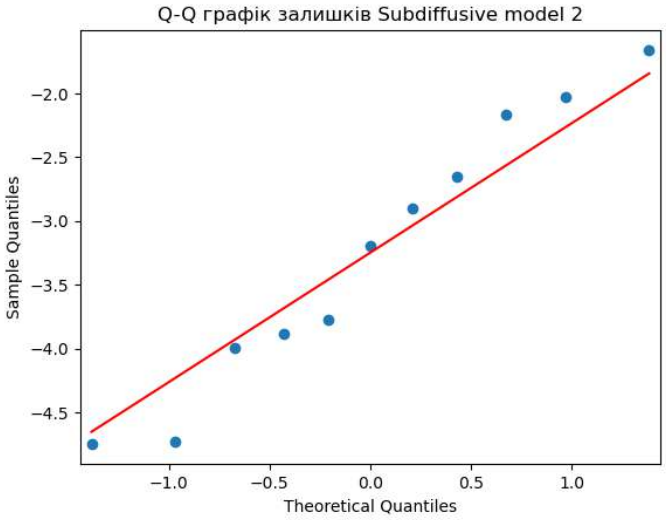


Рис. 3.4. Q-Q графік для Способу 2(Стала волатильність)

Візуальний аналіз показує, що залишки всіх моделей приблизно слідують нормальному розподілу, з незначними відхиленням. Це узгоджується з результатами статистичних тестів (Shapiro–Wilk і Jarque–Bera), які не дають підстав відхилити гіпотезу нормальності залишків. Таким чином, припущення про нормальність залишків вважається обґрунтованим.

3.2 Локальна волатильність

Для покращення точності моделювання цін опціонів було розглянуто підхід із використанням локальної волатильності. Вона враховує зміну волатильності залежно від часу до експірації, що дає змогу краще відобразити реальну ринкову

динаміку. У моделі використовується така функціональна форма локальної волатильності

$$\sigma_{local}(0) = \sigma,\ \sigma_{local}(i) = \sigma_{local}(i-1)(\beta_0 + \beta_1 i^{\gamma} + \beta_1 i^{2\gamma}), i = 1, \dots, n_t.$$

Параметри були отримані шляхом оптимізаційної процедури

$$\beta_0 = 0.99, \beta_1 = -1.1 \times 10^{-11}, \gamma = 1.63.$$

Чисельні розрахунки проводились для тих самих умов, що й раніше: страйк $K = 111$, поточна ціна акції $S = 111$, безризикова ставка $r = 5\%$. Таблиця 3.3 демонструє змодельовані ціни опціонів для кожної з моделей з локальною волатильністю порівняно з реальними ринковими цінами, які було взято на сайті Nasdaq.com та класичною моделлю Блека–Шоулза:

Таб. 3.3 Табличне порівняння змодельованих цін з реальними ринковими(Локальна волатильність)

Exp. Date	**Субдифузійна модель локальна волатильність(Спосіб 1)**	**Субдифузійна модель локальна волатильність(Спосіб 2)**	**Стандартна модель Блека-Шоулза**	**Реальна ціна**
2025-03-21	10.39	10.93	7.00	6.7
2025-04-17	11.99	12.71	9.31	8.58
2025-05-16	13.26	14.22	11.38	10.95
2025-06-20	14.43	15.70	13.47	12.75
2025-07-18	15.20	16.73	14.96	13.75
2025-08-15	15.86	17.66	16.33	15.45
2025-09-19	16.56	18.73	17.91	16.9
2025-10-17	17.04	19.52	19.09	18
2025-11-21	17.54	20.45	20.48	19.5
2025-12-19	17.86	21.15	21.54	20.4
2026-01-16	18.11	21.82	22.55	21.5

На рисунку 3.5. зображено динаміку змодельованих цін опціонів відповідно до кожного з підходів на всьому часовому горизонті, а також їхнє порівняння з реальними ринковими цінами. Це візуальне представлення дозволяє оцінити точність моделей на різних інтервалах часу до експірації.:

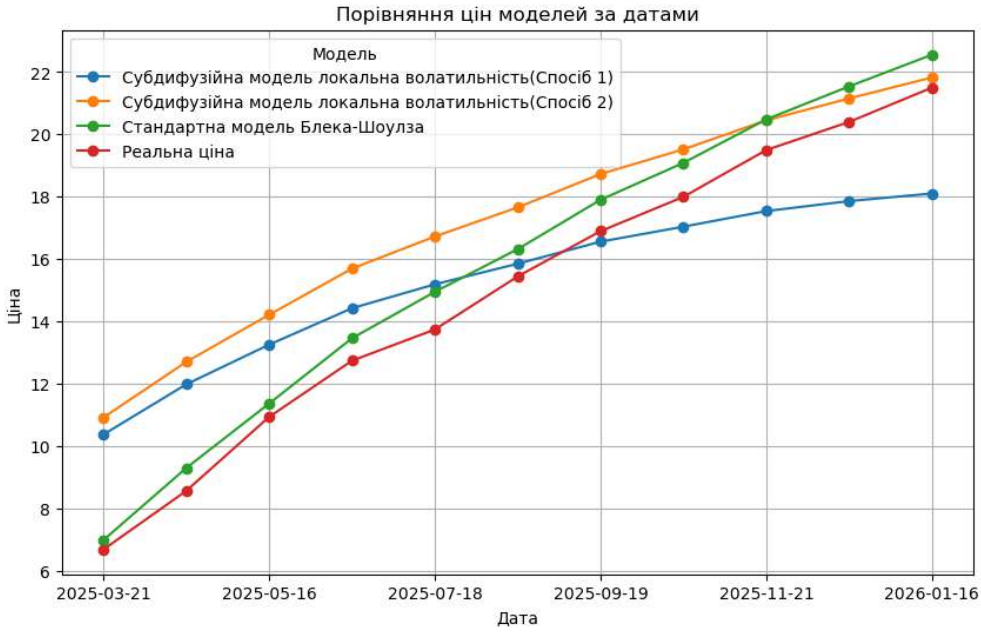


Рис. 3.5. Графічне порівняння змодельованих цін з реальними ринковими(Локальна волатильність)

Субдифузійна модель (Спосіб 2), яка раніше демонструвала найбільші відхилення, тепер наближається до реальних ринкових цін і на довгостроковому проміжку, коли час до експірації наближається до року, забезпечує кращі результати, ніж модель Блека–Шоулза. Спосіб 1, як і у випадку сталої волатильності, дає точніше наближення на середньостроковому горизонті (близько пів року до експірації), однак у короткостроковому та довгостроковому періодах дещо гірше узгоджується з ринковими цінами.

Для більш детального аналізу ефективності моделей було обчислено стандартні метрики точності. Оцінювання проводилось окремо для середньострокового, довгострокового та повного періоду і представлено в таблиці 3.4.:

Таб. 3.4. Метрики точності(Локальна волатильність)

Період	**Модель**	**MAE**	**MSE**	**RMSE**	**ARPE**
2025-07-18 – 2025-10-17	Блек-Шоулз	1.044	1.105	1.051	0.066
	Субдифузійна модель 1	0.787	0.822	0.907	0.051
	Субдифузійна модель 2	2.136	4.860	2.204	0.138
2025-10-17 – 2026-01-16	Блек-Шоулз	1.066	1.140	1.068	0.054
	Субдифузійна модель 1	2.210	5.670	2.381	0.109
	Субдифузійна модель 2	0.888	0.974	0.987	0.046
Весь період	Блек-Шоулз	0.867	0.829	0.911	0.059
	Субдифузійна модель 1	2.012	5.316	2.306	0.171
	Субдифузійна модель 2	2.287	6.856	2.618	0.209

Значення метрик підтверджують зроблені раніше спостереження: модель Блека–Шоулза в середньому демонструє менші похибки, однак поступається субдифузійним підходам на певних часових горизонтах. Зокрема, субдифузійна модель 2 з локальною волатильністю досягає найкращої точності на довгостроковому інтервалі, тоді як модель 1 зберігає точніше узгодження з ринковими цінами у середньостроковому періоді — подібно до результатів при використанні сталої волатильності.

Для подальшої оцінки якості моделей було проведено статистичні тести на нормальність та автокореляцію залишків:

Тести на нормальність залишків:

- Блек-Шоулз:
  - Shapiro–Wilk p-value: 0.2104
  - Jarque–Bera p-value: 0.5264
- Субдифузійна модель 1:
  - Shapiro–Wilk p-value: 0.7548
  - Jarque–Bera p-value: 0.7032

- Субдифузійна модель 2:
    - Shapiro–Wilk p-value: 0.6442
    - Jarque–Bera p-value: 0.7034

Тести на автокореляцію залишків (Durbin–Watson):

- Black-Scholes: 0.085
- Субдифузійна модель 1: 0.100
- Субдифузійна модель 2: 0.030

За результатами тестів на нормальність, усі моделі демонструють залишки, які не суперечать гіпотезі про нормальний розподіл (усі p-value перевищують порогове значення 0.05). Це свідчить про відповідність моделей із точки зору розподілу похибок.

Водночас низькі значення статистики Durbin–Watson вказують на наявність сильної позитивної автокореляції залишків у всіх трьох випадках. Це свідчить про те, що жодна з моделей не забезпечує повного врахування часової структури у динаміці ринкових опціонних цін, що відкриває перспективи для подальшого вдосконалення моделей у цьому напрямку.

Для перевірки припущення про нормальність залишків було побудовано Q-Q графіки для кожної з моделей(рисунки 3.6., 3.7., 3.8):

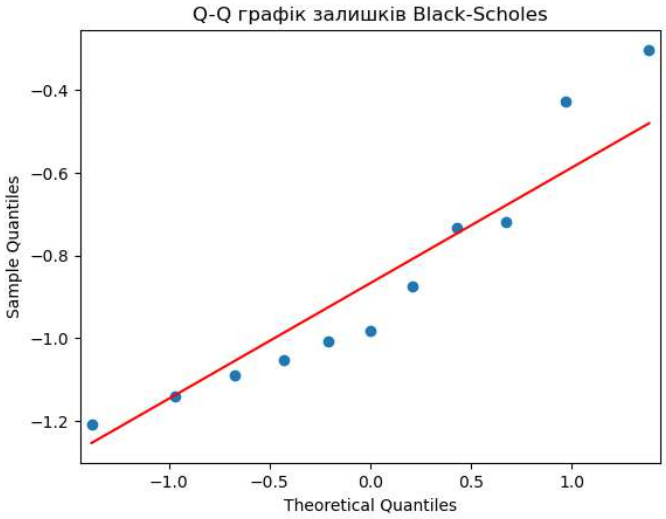


Рис. 3.6. Q-Q графік для Блека-Шоулза(Локальна волатильність)

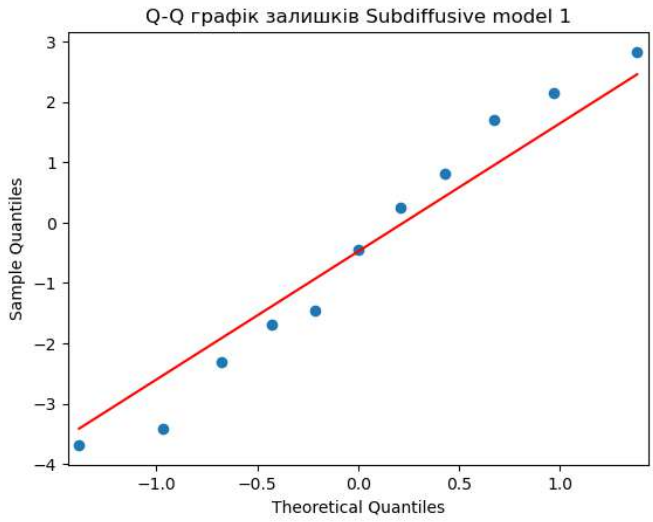


Рис. 3.7. Q-Q графік для Способу 1(Локальна волатильність)

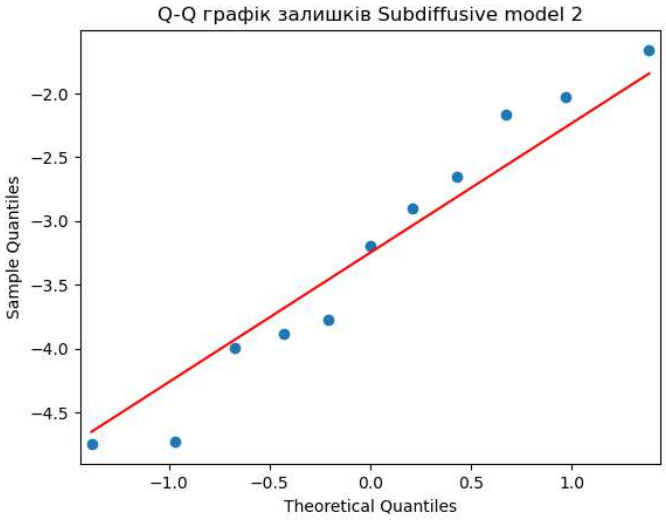


Рис. 3.8. Q-Q графік для Способу 2(Локальна волатильність)

Візуальна оцінка Q-Q графіків свідчить, що залишки змодельованих цін здебільшого відповідають нормальному розподілу, з незначними відхиленнями від прямої. Це підтверджує результати статистичних тестів (Shapiro–Wilk і Jarque–Bera), які не виявили істотних відхилень від нормальності. Отже, припущення про нормальний розподіл залишків є прийнятним для всіх розглянутих моделей.

ВИСНОВКИ

У цій роботі було досліджено підходи до моделювання динаміки цін фінансових активів на неліквідних ринках за допомогою субдифузійних процесів. У якості субординатора було використано стандартний інверсний гаусівський процес (IG(0,1)), що дозволяє враховувати нетривіальну часову структуру та затримки, які характерні для ринків із низькою ліквідністю. Такий підхід є актуальним у контексті пошуку альтернатив до класичних моделей, що недостатньо точно описують реальну ринкову динаміку у складних умовах.

1. У рамках дослідження було застосовано два підходи до розрахунку справедливої вартості європейських кол-опціонів:
   - перший підхід базується на модифікації класичної моделі Блека–Шоулза шляхом заміни детермінованого часу на випадкову величину — інверсного гаусівського субординатора IG(0,1). Справедлива ціна європейського колопціону в такій моделі визначається інтегралом добутку класичної формули Блека–Шоулза та щільності оберненого субординатора;
   - другий передбачає використання фракційного рівняння Дюпіра з урахуванням локальної волатильності, у якому класична похідна за часом замінена на похідну Капуто–Джербашяна.
   - Для випадку IG(0,1) сформульовано твердження про вигляд фрактального рівняння Дюпіра у якому похідна Капуто–Джербашяна перетворюється на похідну Капуто, що відкриває можливості застосування відомих чисельних методів.
2. Субдифузійні моделі було протестовано на прикладі реальних ринкових даних для опціонів компанії AMD. Отримані результати показали, що субдифузійна модель з інтегральним підходом забезпечує кращу точність на середньостроковому горизонті, тоді як модель на основі модифікованого рівняння Дюпіра демонструє вищу точність на довгострокових інтервалах лише за умови врахування локальної волатильності. Обидва підходи в окремих умовах перевершують класичну модель Блека–Шоулза і додаткове

включення локальної волатильності дозволило суттєво покращити відповідність змодельованих цін до реальних ринкових даних.

ЛІТЕРАТУРА

[1] Magdziarz M. Black-Scholes formula in subdiffusive regime. — J. Stat. Phys. — 2009. — T. 136. — C. 553–564. MR2529684. https://doi.org/10.1007/s10955-009-9791-4

[2] Kumar, A., Wylomanska A, Poloczanski, R., Sundar., S.: Fractional Brownian motion time-changed by gamma and inverse gamma process. Physica A-statistical Mechanics and Its Applications, 468, 648–667 (2017).

[3] Donatien, H., Leonenko, N. N.: Option pricing in illiquid markets: A fractional jump–diffusion approach. Journal of Computational and Applied Mathematics,381, 112995, (2021).

[4] Shchestyuk, N., & Tyshchenko, S. Subdiffusive option price model with inverse Gaussian subordinator. — Modern Stochastics: Theory and Applications, 12(2), 136–152, 2025.

[5] Shchestyuk, N., Drin, S., Tyshchenko, S., 2024. Risk evaluating for subdiffusive option price model with gamma subordinator, Mathematical and Statistical Methods for Actuarial Sciences and Finance: Conference proceedings: International Conference of the Mathematical and Statistical Methods for Actuarial Sciences and Finance (MAF 2024), Le Havre, France, Springer, Cham, 286--291.

[6] Vellaisamy, P., and Kumar, A.: The Hitting Time of an Inverse Gaussian Process. ArXiv. 1105.1468 (2011).

[7] Wylomańska, A., Kumar, A., Poloczánski, R., Vellaisamy, P.: Inverse Gaussian and its inverse process as the subordinators of fractional Brownian motion. Phys Review, 94, 042128 (2016)

[8] Deng, W. Finite element method for the space and time fractional Fokker-Planck equation. SIAM Journal on Numerical Analysis, 47, 204–226 (2008).

[9] Сердюк Ф. О., Случинський Д. Ю. Моделювання динаміки неліквідних фінансових ринків // XIII Всеукраїнська наукова конференція молодих математиків. — Київ: Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», 2025.

[10] Shchestyuk, N., Sluchinsky, D., Serduk, F., 2025. Subdiffusive models with local volatility for illiquid markets, Mathematical and Statistical Methods for Actuarial Sciences and Finance: Conference proceedings: International Conference of the Mathematical and Statistical Methods for Actuarial Sciences and Finance (MAF 2025), Springer, Cham (подано до друку).

Додаток А Сердюк Ф. О., Случинський Д. Ю. Моделювання динаміки неліквідних фінансових ринків // XIII Всеукраїнська наукова конференція молодих математиків. — Київ: Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», 2025.

МОДЕЛЮВАННЯ ДИНАМІКИ НЕЛІКВІДНИХ ФІНАНСОВИХ РИНКІВ

Ф. О. СЕРДЮК, Д. Ю. СЛУЧИНСЬКИЙ

Оцінка опціонів на ринках з обмеженою ліквідністю є критично важливою для управління ризиками. Проте традиційні моделі, зокрема модель Блека- Шоуза, часто вироджуються за умов неліквідності. Це зумовлює необхідність використовувати інші моделі, які враховують зупинку зміни ціни та нестабільну волатильність. Наприклад, субдифузійні моделі, що ґрунтуються на введенні процесу нового часу H_t.

У роботі розглянуто субдифузійні моделі ціноутворення опціонів [1] з урахуванням локальної волатильності σ_{H_t} для випадку оберненого гаусівського субординатора з параметрами (0, 1). Для цього випадку пропонуються два методи знаходження справедливої ціни опціонів. Перший метод використовує мартингальний підхід, запропонований Магдіазр [2]:

$$C_{H_t}(S_{H_t}, K, T, \sigma_{H_t}) = \int_0^\infty C(S_{H_t}, K, x, \sigma^2_{H_t}) \cdot \frac{\sqrt{2}}{\sqrt{\pi T}} e^{-\frac{x^2}{2T}}\, dx, \qquad (1)$$

де:

- S_{H_t} — ціна активу в момент часу H_t,
- K — ціна страйку,
- T — час до реалізації опціону.

Другий - базується на фрактальному рівнянні Дюпері [3].

Для цих підходів було реалізовано чисельні методи і знайдено оцінку премій опціонів для реальних ринкових даних з Nasdaq. Проведені статистичні та емпіричні тести для оцінювання якості моделі.

Література

[1] Shchestyuk, N., & Tyshchenko, S. *Subdiffusive option price model with inverse Gaussian subordinator. — Modern Stochastics: Theory and Applications*, *12*(2), 136–152, 2025.

[2] Magdziarz M. *Black-Scholes formula in subdiffusive regime.* — J. Stat. Phys. — 2009. — T. 136. — C. 553–564. MR2529684. https://doi.org/10.1007/s10955-009-9791-4

[3] Donatien, H., & Leonenko, N. N. *Option pricing in illiquid markets: A fractional jump-diffusion approach. — Journal of Computational and Applied Mathematics*, *381*, Article 112995, 2021.

Національний університет «Києво-Могилянська академія»
Email address: f.serdiuk@ukma.edu.ua

Національний університет «Києво-Могилянська академія»
Email address: d.sluchynskyi@ukma.edu.ua

Додаток Б Shchestyuk, N., Sluchinsky, D., Serduk, F., 2025. Subdiffusive models with local volatility for illiquid markets, Mathematical and Statistical Methods for Actuarial Sciences and Finance: Conference proceedings: International Conference of the Mathematical and Statistical Methods for Actuarial Sciences and Finance (MAF 2025), Springer, Cham (подано до друку).

Subdiffusive models with local volatility for illiquid markets

Nataliya Shchestyuk[1], Dmytro Sluchinsky[1], and Fedir Serduk[1]

National University of Kyiv-Mohyla Academy, Department of Mathematics, 04070 Kyiv, Ukraine

Abstract. The paper focuses on subdiffusive models with local volatility that extend classical approaches to capture illiquidity effects and trapping events through stochastic time changes. The main option pricing tool in this paper is a fractional extension of Dupire equation. We apply the equation for subdiffusive in Brownian setting in the case of the inverted tempered stable subordinator and demonstrate numerical results for real financial data.

Keywords: Option pricing, subdiffusion, local volatility, tempered stable subordinator.

1 Intoduction

Illiquid market refers to the state of a stock, bond, or other assets that cannot easily and readily be sold or exchanged for cash without a substantial loss in value. It happens for crisis periods that negatively affect financial activity or for emerging markets in which the number of participants, and thus the number of transactions, is rather low. As a result, illiquid assets tend to have lower trading volume, wider bid-ask spreads, greater price volatility and long periods without any trading. To account for the market's ability to accelerate or slow down during periods of varying trading intensity, fractal models of activity over time have been proposed(see for example [5], [2], [7], etc.). In the context of this model, the "activity time" process can be interpreted as time over which market prices evolve, and it is often associated with trading volume or the flow of new price-sensitive information. However these models and any other models based on Brownian motion are perpetually moving, they are not adapted for modeling periods with motionless stock returns.

Nevertheless, in statistical physics, there are subdiffusion processes that are well suited to describe the dynamic of underlying returns with trapping events [10]. Magdyarz was the first to propose a subdiffusion extension of the Bachelier and Black-Scholes models (see [9], [8]). He considered the standard diffusive process $S_t = S_{H_t}$, which is time-changed by some stochastic process H_t called the inverse subordinator ("hitting time"). In general, for a given subordinator G_t, the inverse subordinator H_t is defined as

$$H_t = inf\{\tau > 0 : G_\tau \geq t\}, \tag{1}$$

and interpreted as the first time at which G_t hits the barrier t. The subordinator G_t in (1) is generally Lévy process with Laplace transform in the following form:

$$E\left(e^{-uG_t}\right) = e^{-t\Psi(u)},$$

where $\Psi(u)$ is called Lévy exponent, which can be written in the following driftless form

$$\Psi(u) = \int_0^{+\infty} \left(1 - e^{-ux}\right) \tilde{\nu}(dx),$$

$\tilde{\nu}(dx)$ is an appropriate Lévy measure. By construction, the inverted process may be constant, and any process subordinated by H_t exhibits motionless periods. Magdziarz constructed the corresponding martingale measure, showed that the model is incomplete and derived the formulas for European put and call option prices using martingale method ([9], [8]). However the approach has two drawbacks. Firstly, following the martingale method, the option price can be evaluated by computing an integral that includes the hitting time density. Thus, the density should admit a closed form. An alternative is to calculate the price using Monte-Carlo simulation. These solutions are explored in [8] in a Brownian setting for α-stable subordinator, in [12] for inverse Gaussian subordinator, and in [13] for Gamma subordinator. Unfortunately, in the general case, the closed form of the hitting time density is unknown, and we need to look for other ways. Secondly, Magdziarz considered subdiffusive models in Brownian setting with constant volatility. In fact, the realized volatility of an underlying asset actually varies with time and with the underlying itself.

In the paper, we consider a subdiffusive model in the Brownian setting with local volatility. To price options we use the Dupire fractal equation, that does not require a closed form for hitting time and takes into account local volatility. It is a forward partial differential equation in which the derivative with respect to time is replaced by a Dzerbayshan–Caputo (D–C) derivative. In [3] was proved that option prices for subdiffusion can be found as solutions of the fractal Dupire equation. The form of the D–C derivative depends upon the chosen Lévy subordinator. In [3] it was detailed for α-stable and Poisson subordinators in the jump-diffusion setting. We detail this for a *Tempered Stable Subordinator* $TS(\alpha, \lambda, c)$. It is a type of Lévy subordinator with Lévy exponent

$$\Psi(x) = c\left((\lambda + x)^\alpha - \lambda^\alpha\right), \tag{2}$$

where $0 < \alpha < 1$ is stability parameter, $\lambda > 0$ is tempering parameter and $c > 0$. The corresponding process H_t is called an inverse tempered stable subordinator (ITS). Its analytical properties are fully detailed in [6] and [1].

The paper is organized as follows. In the second section we consider the subdiffusive model and present the form of fractal Dupire equation for option pricing in case of the tempered stable subordinator. The third section is devoted to the numerical approach and pricing of options for real financial data.

2 Subdiffusive option pricing via fractal Dupire equation

We assume that the subdiffusive market consists of at least three components: a risk-free bond, a risky asset and an option. The idea of subdiffusion models is to replace the calendar time t in the risk-free bond motion and the classical geometrical Brownian motion (GBM) by some stochastic process H_t that represents the hitting time, defined as (1). Thus, the bond B_t and stock S_t are the solutions of the time-changed stochastic differential equations:

$$\frac{dB_{H_t}}{B_{H_t}} = rdH_t, B_0 = 1,$$

$$\frac{dS_{H_t}}{S_{H_t}} = \left(\mu + \frac{\sigma_{H_t}^2}{2}\right) dH_t + \sigma_{H_t} dB_{H_t}, \quad t > 0,$$

where $\sigma_t = \sigma_{H_t}$ is local volatility.

To model option pricing, we use the fractional approach that was introduced in [3] in jump-diffusion setting. We denote the option price as $C_H(T,k)$, where T is the time to maturity and the logarithmic strike price as $k = \log K$. We then replace the standard time derivative in the classical Dupire equation with a convolution-type derivative. Thus, the Fractal Dupire equation can be written in the form

$${}^{\Psi}D_T C_H(T,k) = -r\frac{\partial}{\partial k}C_H(T,k) + \frac{1}{2}\sigma^2(T,S_{H_t})\frac{\partial^2}{\partial k^2}C_H(T,k), \tag{3}$$

where ${}^{\Psi}Du(t)$ is the convolution-type derivative, called the Dzerbayshan–Caputo (D–C) derivative and defined as

$${}^{\Psi}Du(t) = \int_0^t \frac{\partial}{\partial t}u(t-s)\nu(s)ds. \tag{4}$$

The Lévy measure $\tilde{\nu}(dx)$ for the subordinator G_t with Lévy exponent Ψ can be expressed throw the inverse Laplace transform:

$$\tilde{\nu}(dx) = \mathcal{L}^{-1}\left[\frac{d}{dx}\Psi(x)\right](x).$$

The tail of the Lévy measure, or kernel $\nu(s)$, is obtained as

$$\nu(s)\,ds = \int_s^{\infty} \tilde{\nu}(dx)\,ds.$$

We apply the equation (3) to diffusion in Brownian setting with the inverse tempered stable subordinator defined by Lévy exponent (2).

Proposition 1. *The European call option price $C_H(T,k) = C_H(S,K,T,\sigma_H)$ for a time-changed version of the Brownian setting with local volatility in case of*

an inverse tempered stable subordinator H_t is a solution of the fractional Dupire equation:

$$c\frac{\alpha\lambda^\alpha}{\Gamma(1-\alpha)}\int_0^T \frac{\partial}{\partial T}u(T-s)\Gamma(\alpha,\lambda s)\,ds = -r\frac{\partial}{\partial k}C_H(T,k)+\frac{1}{2}\sigma^2(T,S_{H_t})\frac{\partial^2}{\partial k^2}C_H(T,k \tag{5}$$

with the initial condition $C_H(0,K)=(S_{H_0}-K)^+$. The European put option price is a solution of the same fractional Dupire equation but with initial conditions $P_H(0,K)=(K-P_{H_0})^+$.

Proof. Using (2) we compute

$$\frac{d\Psi(x)}{dx} = c\cdot\alpha(\lambda+x)^{\alpha-1},$$

and obtain Lévy measure:

$$\tilde{v}(dx) = c\frac{\alpha}{\Gamma(1-\alpha)}x^{-\alpha-1}e^{-\lambda x}\,I_{x>0}dx.$$

To compute the kernel $\nu(s)$, we evaluate the tail integral:

$$\nu(s)\,ds = \int_s^\infty \tilde{v}(dx) = c\frac{\alpha}{\Gamma(1-\alpha)}\int_s^\infty x^{-\alpha-1}e^{-\lambda x}dx.$$

Using the change of variables $x=\frac{t}{\lambda}\Rightarrow dx=\frac{dt}{\lambda}$ we have

$$\nu(s)\,ds = c\frac{\alpha\lambda^\alpha}{\Gamma(1-\alpha)}\int_{\lambda s}^\infty t^{-\alpha-1}e^{-t}dt = c\frac{\alpha\lambda^\alpha}{\Gamma(1-\alpha)}\Gamma(-\alpha,\lambda s)\,ds,$$

where $\Gamma(-\alpha,\lambda s)$ denotes the upper incomplete gamma function. Then D–C derivative (4) for $TS(\alpha,\lambda,c)$ subordinator can be written as

$${}^{\Psi}Du(t) = c\frac{\alpha\lambda^\alpha}{\Gamma(1-\alpha)}\int_0^T \frac{\partial}{\partial T}u(T-s)\Gamma(\alpha,\lambda s)\,ds. \tag{6}$$

Thus, if we substitute (6) in (3), the proposition follows.

Remark 1. The issues of the existence of arbitrage opportunities and the completeness of the TSS subdiffusion market were investigated in [15], [14], where was proved that subdiffusive TSS market is arbitrage-free and incomplete.

Remark 2. The presented option pricing subdiffusive model with $TS(\alpha,\lambda,c)$ subordinator has an interesting interpretation of its parameters in light of the market price of risk. Thanks to the theorem about the incompleteness of the subdiffusive market, the existence of a market price of duration risk naturally arises (see [15]). Once S_t is calibrated to liquid market prices, the market price of duration risk will be reflected in the martingale density parameters, and these same features will be thus reflected in the (risk-adjusted) frequency and duration of trade pauses incorporated in hitting time H_t.

In [15] was shown the parameter λ expresses belief in the speed at which the granular price evolution of the asset will revert to a fully "liquid" state, which is well approximated by a standard L'evy-driven diffusion. So, λ captures the risk of latency to liquidity, while absolute trade duration risk defined by α. Therefore, $TS(\alpha, \lambda, c)$ models offer a unified framework in which ultra short-term option pricing accounts for the microstructural duration effect, whereas options at typical maturities are priced according to the usual L'evy paradigm: the cut-off between the two states is encoded in λ.

Remark 3. Since inverse Gaussian $IG(\delta, \gamma)$ subordinator can be considered as tempered stable L'evy subordinator $TS(1/2, \gamma^2, \delta)$, we can refer to the paper of [12], where the fair option price for IG market was explored as a solution of the PIDE:

$$\int_0^T \delta \frac{\partial}{\partial t} C_H(T-s,k) \left(\gamma(\Phi\left(\gamma\sqrt{s}\right) - 1) + \frac{e^{-\frac{s\gamma^2}{2}}}{\sqrt{2\pi s}} \right) ds$$
$$= -\frac{r}{2}\frac{\partial}{\partial k} C_H(T,k) + \frac{\sigma^2}{4}\frac{\partial^2}{\partial k^2} C_H(T,k),$$

with the initial condition $C_H(0,K) = (S_{H_0} - K)^+$, and $k = lnK$. Here parameter γ captures the risk of latency to liquidity, while absolute trade duration risk remains constant and is defined by $1/2$ (see [12], [15]).

Remark 4. If $\gamma = 0$ for $TS(1/2, \gamma^2, \delta)$, we get a regime where prices exhibit maximum staleness at all temporal scales. Moreover, in this case D-C derivative becames Caputo derivative. In fact,

$$^{\Psi}Du(t) = 2\delta \int_0^t \frac{\partial}{\partial t} C_H(t-s,k) \cdot \frac{1}{\sqrt{2\pi s}}\, ds.$$

Using the substitution $u = t - s \Rightarrow ds = -du$, we get:

$$^{\Psi}Du(t) = \frac{2\delta}{\sqrt{2\pi}} \Gamma\left(\tfrac{1}{2}\right) \frac{1}{\Gamma\left(1-\frac{1}{2}\right)} \int_0^t \frac{\frac{\partial}{\partial u} C_H(u,k)}{(t-u)^{1/2}} du = \frac{2\delta}{\sqrt{2}} D_{1/2} C_H(T,k),$$

where $D_{1/2}C_H(T,k)$ is Caputo derivative. This leads to the fractional Dupire-type equation:

$$\frac{\delta}{\sqrt{2}} D_{1/2} C_H(t) = -\frac{r}{2}\frac{\partial}{\partial k} C_H(t,k) + \frac{\sigma^2(t, S_{H_t})}{4}\frac{\partial^2}{\partial k^2} C_H(t,k). \tag{7}$$

3 Numerical approach and illustration

In the section a numerical approach involving the discretization of the integro-differential equation (7) is applied. The Caputo derivative in the left side of (7)

admits the numerical representation by the finite difference sum [4]:

$$\mathcal{D}_\alpha C_H(k,j) = \frac{(\Delta t)^{-\alpha}}{\Gamma(1-\alpha)}\left(C_H(k,j) - C_H(k-1,j)\right) + \\ + \frac{(\Delta t)^{-\alpha}}{\Gamma(1-\alpha)}\sum_{m=0}^{k-2}(k-m)^{-\alpha}\left(C_H(m+1,j) - C_H(m,j)\right).$$

The matrix form of Caputo derivative for the case $\alpha = \frac{1}{2}$ can be written as

$$\frac{1}{\sqrt{2}}\mathcal{D}_{1/2}C_H(k,\cdot)^T = \frac{1}{\sqrt{2}}\frac{(\Delta t)^{-\frac{1}{2}}}{\sqrt{\pi}}\left(C_H(k,\cdot) - C_H(k-1,\cdot)\right)^T + \\ + \frac{1}{\sqrt{2}}\left(D(k)C_H(0:k-1,\cdot)\right)^T \mathbf{1}_{k-1},$$

where $D(k)$ is a special matrix of dimension $(k-1)\times k$ with appropriate coefficients:

$$D(k) = \frac{(\Delta t)^{-\frac{1}{2}}}{\Gamma\left(\frac{1}{2}\right)}\begin{pmatrix} 0 & \dots & \dots & 0 & -(2)^{-\frac{1}{2}} & (2)^{-\frac{1}{2}} \\ \vdots & \ddots & 0 & -(3)^{-\frac{1}{2}} & (3)^{-\frac{1}{2}} & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ -(k)^{-\frac{1}{2}} & (k)^{-\frac{1}{2}} & 0 & \dots & \dots & 0 \end{pmatrix}.$$

The first and second order derivatives with respect to k on the right side of (7) can be calculated by a central finite difference approximation:

$$\frac{\partial C_H(k,j)}{\partial k} \approx \frac{C_H(k,j+1) - C_H(k,k-1)}{2\Delta_K}, \quad \frac{\partial C_H(k,0)}{\partial k} = \frac{C_H(k,1) - C_H(k,0)}{\Delta_K}, \\ \frac{\partial C_H(k,n_K+1)}{\partial k} = \frac{C_H(k,n_K+1) - C_H(k,n_K)}{\Delta_K},$$

$$\frac{\partial^2 C_H(k,j)}{\partial k^2} \approx \frac{C_H(k,j+1) - 2C_H(k,j) + C_H(k,j-1)}{\Delta_K^2}, \\ \frac{\partial^2 C_H(k,0)}{\partial k^2} = \frac{\partial^2 C_H(k,1)}{\partial k^2}, \frac{\partial^2 C_H(k,n_K+1)}{\partial k^2} = \frac{\partial^2 C_H(k,n_K)}{\partial k^2},$$

where $n_K = \left[\frac{K_{max}-K_0}{\Delta_K}\right]$. These derivatives are represented in matrix form:

$$\frac{\partial C_H(k,j)}{\partial k} = \bar{K}R_1 C_H(k,\cdot)^T, \quad \frac{\partial^2 C_H(k,j)}{\partial k^2} = \bar{K}^2 R_2 C_H(k,\cdot)^T,$$

where

$$R_1 = \frac{1}{\Delta_K}\begin{pmatrix} -1 & 1 & 0 & \dots & 0 \\ -\frac{1}{2} & 0 & \frac{1}{2} & \dots & 0 \\ 0 & -\frac{1}{2} & 0 & \frac{1}{2} & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & 0 & -1 & 1 \end{pmatrix}, \quad R_2 = \frac{1}{\Delta_K^2}\begin{pmatrix} 1 & -2 & 1 & 0 & \dots & 0 \\ 1 & -2 & 1 & 0 & \dots & 0 \\ 0 & 1 & -2 & 1 & 0 & \dots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & 1 & -2 & 1 \end{pmatrix}.$$

Therefore, the final numerical iterative scheme for solving (7) in the case $\delta = 1$ can be written in the form:

$$C_H(k,\cdot)^T = \left(\bar{K} R_1 \frac{r}{2} - \bar{K}^2 R_2 \frac{\sigma_H^2}{4} + I_{n_k+1} \frac{1}{\sqrt{2}} \frac{(\Delta t)^{-\frac{1}{2}}}{\sqrt{\pi}} \right)^{-1} \times$$
$$\times \left(\frac{1}{\sqrt{2}} \frac{(\Delta t)^{-\frac{1}{2}}}{\sqrt{\pi}} \left(D(k) C_H(k-1,\cdot) \right)^T \mathbf{1}_{k-1} \right).$$

Numerical experiments were performed to compute European call option prices on AMD stock for the date 13/02/2025 with the following parameters: strike price $K = 111$, spot price $S = 111$, and interest rate $r = 5\%$. The annual constant volatility from historical data equals $\sigma = 47.37\%$. Local volatility σ_{local} was constructed iteratively using the recurrence relation:

$$\sigma_{\text{local}}(i) = \sigma_{\text{local}}(i-1) \cdot \left(\beta_0 + \beta_1 i^{\gamma} + \beta_1 i^{2\gamma} \right), \quad \text{for } i = 1, \ldots, n_t,$$

with initial value $\sigma_{\text{local}}(0) = \sigma$ and parameters $\beta_0 = 0.99$, $\beta_1 = -1.10 \times 10^{-11}$, $\gamma = 1.63$, which was found by optimization procedure.

Three models were evaluated:

- Classical Black-Scholes model;
- Subdiffusive model with constant volatility;
- Subdiffusive model with local volatility.

Figure 1 presents the comparison of call option model prices and actual market prices (premiums) for varying times of maturity.

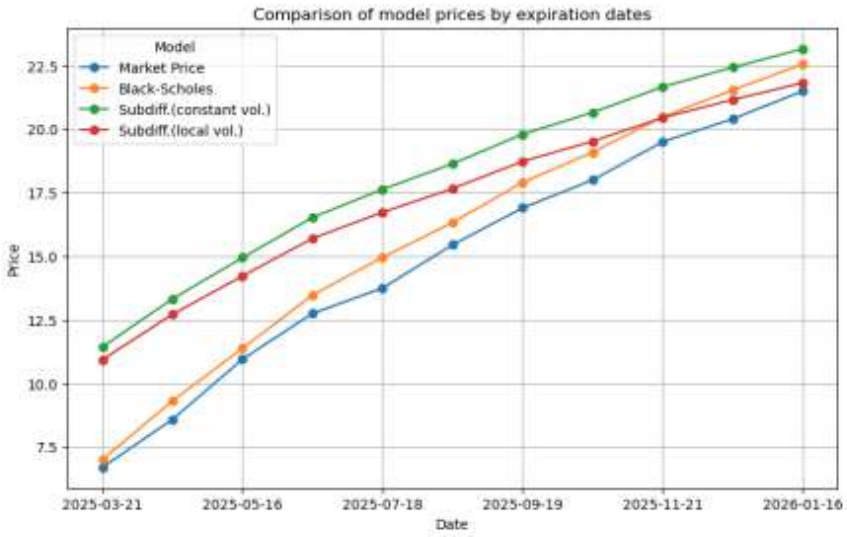


Fig. 1: Comparison of model call option prices with actual market premiums

As we can see from the graphics in Fig. 1, the Black-Scholes option pricing model shows better results in the short-term period, while the subdiffusive model

is more effective in the long-term perspective. To assess and compare the predictive performance of the models, for two time periods the following accuracy metrics were calculated: **MAE** – Mean Absolute Error, **MSE** – Mean Squared Error, **RMSE** – Root Mean Squared Error, **ARPE** – Average Relative Pricing Error.

The results are presented in the Table 1.

Table 1: Accuracy metrics by time interval and model

Interval	Model	MAE	MSE	RMSE	ARPE
2025-03-21 to 2025-09-19	Black-Scholes	0.753	0.652	0.807	0.061
	Subdiffusive (const)	3.888	15.533	3.941	0.369
	Subdiffusive (local)	3.087	10.218	3.197	0.302
2025-10-17 to 2026-01-16	Black-Scholes	1.066	1.140	1.068	0.054
	Subdiffusive (const)	2.128	4.656	2.158	0.109
	Subdiffusive (local)	0.888	0.974	0.987	0.046

4 Conclusion

The paper develops a sub-diffusive model with local volatility time-changed by inverse TSS subordinator. The main pricing tool for European call and put options in this paper is a fractional extension of Dupire's equation in Brownian setting with local volatility. We apply the equation for TSS and demonstrate numerical results for real historical financial data.

The study finds that subdiffusive models are more accurate and sensitive, especially in capturing market behaviors like periods of price motionless for illiquid markets. The results of comparing the studied model with the classical one show that the classical B-S model demonstrates better results in the short-term period, while the sub-diffusive models is more effective in the long-term perspective. Moreover, the accuracy of sub-diffusive model with local volatility prevails over sub-diffusive model with constant volatility

Due to the proposed approaches and numerical algorithm, the investor gets tools that allow him to take into account the market's illiquidity and a local volatility.

References

1. Alrawashdeh, M. S., Kelly, J. F., Meerschaert, M. M., and Scheffler, H. P.: Applications of inverse tempered stable subordinators. *Computers and Mathematics with Applications,* **73**, 892–905 (2017).
2. Casteli, F., Leonenko, N., Shchestyuk, N.: Student-like models for risky asset with dependence. *Stochastic Analysis and Applications,* **35**, 452–464 (2017).

3. Donatien, H., Leonenko, N. N.: Option pricing in illiquid markets: A fractional jump–diffusion approach. *Journal of Computational and Applied Mathematics,* **381**, 112995, (2021).
4. Deng, W.: Finite element method for the space and time fractional Fokker-Planck equation. *SIAM Journal on Numer. Anal.,* **47**, 204–226 (2008).
5. Heyde, C.: A risky asset model with strong dependence through fractal activity time. *J. Appl. Probab.,* **36**, 1234–1239 (1999).
6. Kumar, A. and Vellaisamy, P.: Inverse tempered stable subordinators. *J. Statistics and Probability Letters,* **103**,134–141 (2015).
7. Leonenko N., Liu A., Shchestyuk, N. Student models for a risky asset with dependence: Option pricing and Greeks. *Austrian Journal of Statistics,* **54 (1)** , 138–165 (2025).
8. Magdziarz, M.: Black–Scholes formula in subdiffusive regime. Journal of Stat. Phys.**136**, 553–564 (2009).
9. Magdziarz, M., Orze l, S., and Weron, A.: Option Pricing in Subdiffusive Bachelier Model. Journal of Stat. Phys. **145**, 187–202 (2011).
10. Metzler, R., Klafter, J.: The random walk's guide to anomalous diffusion: a fractional dynamics approach. Physics Reports **339(1)**, 1–77 (2000).
11. Shchestyuk, N., Tyshchenko, S.: Option Pricing and Stochastic Optimization. In *Malyarenko, A., Ni, Y., Rancic, M., Silvestrov, S. (eds) Stochastic Processes, Statistical Methods, and Engineering Mathematics. SPAS 2019. Springer Proceedings in Mathematics and Statistics*, **408** 651–669 (Springer, Cham) (2022).
12. Shchestyuk, N., Tyshchenko, S. Subdiffusive option price model with inverse Gaussian subordinator. Modern Stochastics: Theory and Applications, Vol.12, No 2, 136–152.(2025).
13. Shchestyuk, N., Drin, S., Tyshchenko, S., 2024. Risk evaluating for subdiffusive option price model with gamma subordinator, Mathematical and Statistical Methods for Actuarial Sciences and Finance: Conference proceedings: International Conference of the Mathematical and Statistical Methods for Actuarial Sciences and Finance (MAF 2024), Le Havre, France, Springer, Cham, 286–291.
14. Toaldo, B.: Convolution-type derivatives, hitting-times of subordinators and time-changed C_0-semigroups. *Potential Analysis*, **42**, 115–140, (2015).
15. Torricelli, L.: Trade duration risk in subdiffusive financial models. *Physica A: Statistical Mechanics and its Applications*, **541**,123694, (2020).

Додаток В Код програми

Загальний блок

Підключення бібліотек

In [4]:

```
import pandas as pd
import math
import statistics
from sympy.stats import cdf, Normal
import numpy as np
from scipy.special import gamma
import matplotlib.pyplot as plt
from scipy.stats import norm
from scipy.integrate import quad
from joblib import Parallel, delayed
from sklearn.metrics import mean_absolute_error, mean_squared_error
from scipy.stats import shapiro, jarque_bera, t
import statsmodels.api as sm
from statsmodels.stats.stattools import durbin_watson
from sympy import *
from tqdm.notebook import tqdm
import cupy as cp
```

Завантаження файлів.

In [6]:

```
historic_df = pd.read_csv('HistoricalData_1739451966876.csv')
print(historic_df.head())
```

```
       Date  Close/Last    Volume     Open      High       Low
0  02/12/2025      111.72  25319730  $109.52  $111.8356  $109.0649
1  02/11/2025      111.10  35033760  $108.98    $113.07    $108.94
2  02/10/2025      110.48  34905360  $108.44    $111.40    $108.15
3  02/07/2025      107.56  46082500  $109.13    $109.92    $106.79
4  02/06/2025      110.16  50426590  $110.93    $112.56    $109.02
```

D_k

In [7]:

```
def fractional_derivative_matrix(delta_t, alpha, k):
    """
    Create the fractional derivative matrix D(k) for fractional order alpha.

    Parameters:
    - delta_t: Step size in the k-grid.
    - alpha: Fractional order of the derivative.
    - k: Number of points in the k-grid.

    Returns:
    - D_k: Fractional derivative matrix.
    """
    coef = (delta_t**(-alpha)) / gamma(1 - alpha)
    coef_gpu   = float(coef)
    D_k = cp.zeros((k-1, k), dtype=cp.float64)
```

```
    for i in range(k-1):
        D_k[i, k-i-2] = -((i+2)**(-alpha))
        D_k[i, k-i-1] = (i+2)**(-alpha)

    return coef_gpu * D_k
```

R_k

In [9]:

```
def finite_difference_matrix(n_k, Delta_K):
    """
    Create the finite difference matrix R_k for first derivative approximation.

    Parameters:
    - n_k: Number of points in the k-grid.
    - delta_t: Step size in the t-grid.

    Returns:
    - R_k: Finite difference matrix for the first derivative.
    """
    R_k = np.zeros((n_k + 1, n_k + 1))
    for i in range(1, n_k):
        R_k[i, i - 1] = -1/2
        R_k[i, i + 1] = 1/2

    R_k[0, 0] = -1
    R_k[0, 1] = 1
    R_k[n_k, n_k - 1] = -1
    R_k[n_k, n_k] = 1
    R_k = R_k / Delta_K
    return R_k
```

R_2k

In [11]:

```
def second_derivative_matrix(n_k, Delta_K):
    """
    Create the finite difference matrix R2_k for second derivative approximation.

    Parameters:
    - n_k: Number of points in the k-grid.
    - delta_t: Step size in the t-grid.

    Returns:
    - R2_k: Finite difference matrix for the second derivative.
    """
    R2_k = np.zeros((n_k + 1, n_k + 1))
    for i in range(1, n_k):
        R2_k[i, i - 1] = 1
        R2_k[i, i] = -2
        R2_k[i, i + 1] = 1

    R2_k[0, 0] = 1
    R2_k[0, 1] = -2
    R2_k[0, 2] = 1

    R2_k[n_k, n_k - 2] = 1
```

```
    R2_k[n_k, n_k - 1] = -2
    R2_k[n_k, n_k] = 1
    R2_k /= Delta_K**2
    return R2_k
```

Початкові параметри

In [8]:

```
S = pd.read_excel('data.xlsx', sheet_name='S').loc[0, 'PriceUSD']          #Початкова ціна з реальних даних
historic_df['Date'] = pd.to_datetime(historic_df['Date'])                  #Перетворення в формат дат

K = 111                                                                    #Страйк ціна
k_min = 50                                                                   #Мінімальна страйк ціна
k_max = 150                                                                  #Максимальна страйк ціна
A_0 = 111                                                                  #Початкова ціна


n_t = 252                                                                  #Кількість проміжків на реальних даних
delta_t = 1/252                                                              #Різниця між проміжками в роках
print(n_t,delta_t)

alpha=0.5                                                                  #Параметр субординатора
K_j = np.arange(k_min, k_max + 0.5, 0.5).tolist()                                  #Створеня списку страйкових цін з люфтом в 0.5
Delta_K = K_j[1] - K_j[0]                                                    #Обрахунок разниці між сусідніми(0.5)
n_k = len(K_j)-1                                                             #Кількість страйкових цін

K_plot_BS = K                                                              #Параметр для побудови Блека Шоуза
K_plot_dif = int((K-k_min)/Delta_K)                                                #Параметр для побудови субдифузії
```

```
252 0.003968253968253968
```

Розрахунок річної волатильності та процентної ставки:

In [10]:

```
OptionPrice = pd.read_excel('data.xlsx', sheet_name='Option')
print(OptionPrice.head())
hni = []
for i in range(historic_df.shape[0]):
    if i == 0:
        continue
    hn = math.log(historic_df.loc[i, 'Close/Last']/historic_df.loc[i-1, 'Close/Last'])
    hni.append(hn)
hn_av = sum(hni)/len(hni)
st_dev = statistics.stdev(hni)
print('Середнє з hn: ')
print(hn_av)
print('Стандартне квадратичне відхилення: ')
print(st_dev)

sigma = st_dev * math.sqrt(historic_df.shape[0])
vol_per = sigma*100
print("Річна волатильність у відсотках: ")
print(vol_per)
# r = (hn_av + math.pow(sigma,2)/2)
r = 0.05
print("Процентна ставка: ")
print(r)
```

```
   Exp. Date  Last  Change   Bid   Ask Volume Open Int.  Strike Last.1  \
0 2025-02-21  0.05   -0.01  0.05  0.06     12      597     133   22.1
```

```
1 2025-02-21  0.05  -0.01   0.05  0.06    663     5880    134   23.9
2 2025-02-21  0.05  -0.01   0.04  0.05    512    17088    135  23.55
3 2025-02-21  0.03     --  0.02  0.03   1028    20719    140  27.79
4 2025-02-21  0.01  -0.02   0.01  0.02   1053    21076    145  34.05

  Change.1  Bid.1  Ask.1 Volume.1 Open Int..1
0      --   21.2  21.45      --         --
1      --  22.15  22.45      --          1
2   -0.21   23.2   23.4     117       3430
3      --   28.1   28.4      --        631
4    0.23  33.15  33.45       8       1755
Середнє з hn:
0.0017152829379411961
Стандартне квадратичне відхилення:
0.02990096736454434
Річна волатильність у відсотках:
47.372041351753246
Процентна ставка:
0.05
```

Стала сігма

C_H

In [18]:

```
def compute_C_H(n_t, n_k, R_k, R2_k, K_j,alpha,r,sigma,A_0,delta_t):
    """
    Compute C_H(t, k) iteratively over time steps.

    Parameters:
    - n_t: Number of time steps.
    - n_k: Number of strike price steps.
    - delta_t: Step size in the t-grid.
    - alpha: Fractional order of the derivative.
    - R_k: First derivative matrix.
    - R2_k: Second derivative matrix.
    - D_k: Fractional derivative matrix.

    Returns:
    - C_H: Matrix of C_H values over time and k.
    """
    I_nk_1 = np.identity(n_k + 1)
    coef = (delta_t**(-alpha)) / gamma(1 - alpha) * 1/np.sqrt(2)


    C_H = np.zeros((n_t+1, n_k + 1))
    K_1_d = np.diag(K_j)
    K_2_d = np.diag(np.array(K_j) ** 2)


    for j in range(len(K_j)):
        C_H[0, j] = np.maximum(A_0 - K_j[j], 0)

    I_nk_1_gpu = cp.array(I_nk_1,    dtype=cp.float64)
    K_1_d_gpu  = cp.array(K_1_d,     dtype=cp.float64)
    R_k_gpu    = cp.array(R_k,       dtype=cp.float64)
    K_2_d_gpu  = cp.array(K_2_d,     dtype=cp.float64)
```

```
    R2_k_gpu   = cp.array(R2_k,     dtype=cp.float64)
    coef_gpu   = float(coef)

    C_H_gpu = cp.array(C_H, dtype=cp.float64)

    for k in tqdm(range(1, n_t+1), desc="Time steps on GPU"):
        M = (coef_gpu * I_nk_1_gpu
             + (r/2) * K_1_d_gpu @ R_k_gpu
             - ((sigma**2) / 4) * (K_2_d_gpu @ R2_k_gpu))

        invM = cp.linalg.inv(M)

        I_k_1    = cp.ones((k-1,1), dtype=cp.float64)
        C_H_k1   = C_H_gpu[k-1,:].reshape(-1,1)
        D_k_gpu      = fractional_derivative_matrix(delta_t, alpha, k)

        multiplier2 = (C_H_k1 * coef_gpu
                       - (1/cp.sqrt(2)) * (D_k_gpu @ C_H_gpu[:k,:]).T @ I_k_1)

        C_H_gpu[k,:] = (invM @ multiplier2).T

    C_H_result = cp.asnumpy(C_H_gpu)
    return C_H_result
```

Розрахунок Блека Шоуза

In [21]:

```
N = Normal('N', 0.0, 1.0)

def d1(S, K, T, r, sigma):
    return (ln(S / K) + (r + 1/2 * sigma ** 2) * T) / (sigma * sqrt(T))

def d2(S, K, T, r, sigma):
    return (ln(S / K) + (r - 1/2 * sigma ** 2) * T) / (sigma * sqrt(T))

def C(S, K, T, r, sigma):
    return S * cdf(N)(d1(S, K, T, r, sigma)) - K * exp(-r*T)* cdf(N)(d2(S, K, T, r, sigma))
```

In [22]:

```
def compute_values(i):
    try:
        t = i / 252.0
        c_val = C(A_0, K, t, r, sigma).evalf()
        return i, c_val
    except Exception as e:
        print(f"Error at i={i}: {e}")
        return i, None, None

results = Parallel(n_jobs=-1, verbose=10)(
    delayed(compute_values)(i) for i in range(1, 253)
)

C_val_BS = [item[1] for item in results]
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 24 concurrent workers.
[Parallel(n_jobs=-1)]: Done   2 tasks      | elapsed:    1.3s
[Parallel(n_jobs=-1)]: Done  13 tasks      | elapsed:    1.3s
[Parallel(n_jobs=-1)]: Done  24 tasks      | elapsed:    1.4s
[Parallel(n_jobs=-1)]: Done  37 tasks      | elapsed:    1.4s
[Parallel(n_jobs=-1)]: Done  50 tasks      | elapsed:    1.4s
```

```
[Parallel(n_jobs=-1)]: Batch computation too fast (0.17446639472742925s.) Setting batch_size=2.
[Parallel(n_jobs=-1)]: Done  65 tasks      | elapsed:    1.4s
[Parallel(n_jobs=-1)]: Done  80 tasks      | elapsed:    1.4s
[Parallel(n_jobs=-1)]: Done  97 tasks      | elapsed:    1.4s
[Parallel(n_jobs=-1)]: Done 114 tasks      | elapsed:    1.4s
[Parallel(n_jobs=-1)]: Batch computation too fast (0.025600671768188477s.) Setting batch_size=4.
[Parallel(n_jobs=-1)]: Done 147 tasks      | elapsed:    1.4s
[Parallel(n_jobs=-1)]: Done 184 tasks      | elapsed:    1.4s
[Parallel(n_jobs=-1)]: Done 231 out of 252 | elapsed:    1.5s remaining:    0.0s
[Parallel(n_jobs=-1)]: Done 252 out of 252 | elapsed:    1.5s finished
```

Розрахунок Субдифузивної моделі через інтеграл(Спосіб 1)

In [25]:

```
def h(x, T):
    return sqrt(2 / (pi * T)) * E ** (-(x ** 2)/(2*T))

def IG_sub_C(x, S, K, T, r, sigma):
    return Integral(C(S, K, x, r, sigma)*h(x, T),(x, 0, oo))
```

In [26]:

```
x = symbols('x')

def compute_values(i):
    try:
        t = i / 252.0
        C_sub_val1 = IG_sub_C(x, A_0, K, t, r, sigma).evalf()
        return i, C_sub_val1
    except Exception as e:
        print(f"Error at i={i}: {e}")
        return i, None, None

results = Parallel(n_jobs=-1, verbose=10)(
    delayed(compute_values)(i) for i in range(1, 253)
)

C_sub_val1 = [item[1] for item in results]
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 24 concurrent workers.
[Parallel(n_jobs=-1)]: Done   2 tasks      | elapsed:   20.8s
[Parallel(n_jobs=-1)]: Done  13 tasks      | elapsed:   21.6s
[Parallel(n_jobs=-1)]: Done  24 tasks      | elapsed:   25.4s
[Parallel(n_jobs=-1)]: Done  37 tasks      | elapsed:   43.8s
[Parallel(n_jobs=-1)]: Done  50 tasks      | elapsed:  1.1min
[Parallel(n_jobs=-1)]: Done  65 tasks      | elapsed:  1.3min
[Parallel(n_jobs=-1)]: Done  80 tasks      | elapsed:  1.4min
[Parallel(n_jobs=-1)]: Done  97 tasks      | elapsed:  1.5min
[Parallel(n_jobs=-1)]: Done 114 tasks      | elapsed:  1.6min
[Parallel(n_jobs=-1)]: Done 133 tasks      | elapsed:  1.8min
[Parallel(n_jobs=-1)]: Done 152 tasks      | elapsed:  1.9min
[Parallel(n_jobs=-1)]: Done 173 tasks      | elapsed:  2.1min
[Parallel(n_jobs=-1)]: Done 194 tasks      | elapsed:  2.2min
[Parallel(n_jobs=-1)]: Done 231 out of 252 | elapsed:  2.5min remaining:   13.4s
[Parallel(n_jobs=-1)]: Done 252 out of 252 | elapsed:  2.6min finished
```

Розрахунок Субдифузивної моделі через наближення(Спосіб 2)

In [28]:

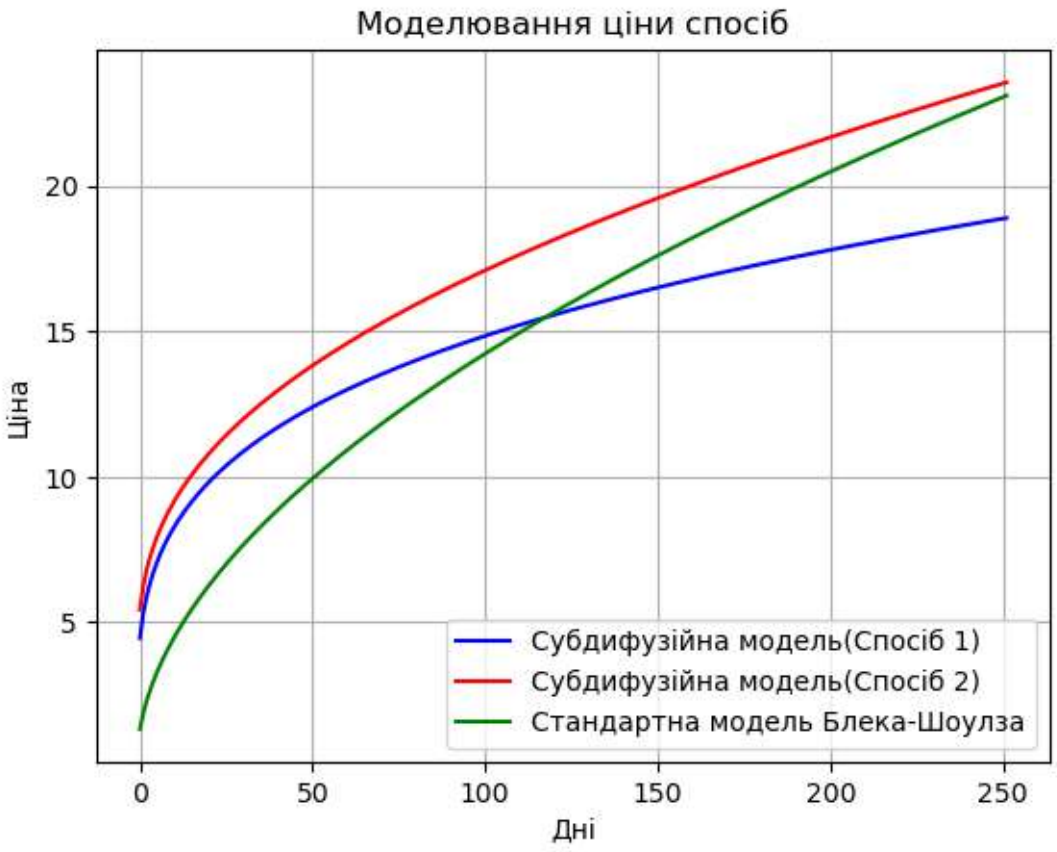
```
R_k = finite_difference_matrix(n_k, Delta_K)
R2_k = second_derivative_matrix(n_k, Delta_K)
C_sub_val2 = compute_C_H(n_t, n_k, R_k, R2_k, K_j,alpha,r,sigma,A_0,delta_t)
```

```
Time steps on GPU:   0%|          | 0/252 [00:00<?, ?it/s]
```

Побудова графіків

In [30]:

```
plt.plot(range(n_t), C_sub_val1[0:], color = 'blue', label='Субдифузійна модель(Спосіб 1)')
plt.plot(range(n_t), C_sub_val2[1:,K_plot_dif], color = 'red', label='Субдифузійна модель(Спосіб 2)')
plt.plot(range(n_t), C_val_BS[0:], color = 'green', label='Стандартна модель Блека-Шоулза')
plt.title('Моделювання ціни спосіб')
plt.xlabel('Дні')
plt.ylabel('Ціна')
plt.legend()
plt.grid(True)
plt.show()
```



Реальні дані.

In [32]:

```
OptionPrice = pd.read_excel('data.xlsx', sheet_name='Option')
print(OptionPrice.head())
```

```
start_date = pd.read_excel('data.xlsx', sheet_name='S').loc[0, 'Date'].strftime('%Y-%m-%d')
```

```
print(start_date)

range_strikes = (OptionPrice['Strike'] >= K - 1) & (OptionPrice['Strike'] <= K + 1)
real_price = OptionPrice[range_strikes][['Strike', 'Exp. Date', 'Last']]

real_price['Exp. Date'] = pd.to_datetime(real_price['Exp. Date']).dt.strftime('%Y-%m-%d')

real_price['busday diff'] = real_price['Exp. Date'].apply(lambda x: np.busday_count(start_date, x))
real_price['busday diff'] = real_price['busday diff'] - 1
real_price

K_merge = real_price['Strike'].max()
K_merge
```

```
  Exp. Date  Last  Change   Bid   Ask Volume Open Int.  Strike Last.1  \
0 2025-02-21  0.05  -0.01   0.05  0.06     12      597    133   22.1
1 2025-02-21  0.05  -0.01   0.05  0.06    663     5880    134   23.9
2 2025-02-21  0.05  -0.01   0.04  0.05    512    17088    135  23.55
3 2025-02-21  0.03     --  0.02  0.03   1028    20719    140  27.79
4 2025-02-21  0.01  -0.02   0.01  0.02   1053    21076    145  34.05

  Change.1  Bid.1  Ask.1 Volume.1 Open Int..1
0      --   21.2  21.45      --         --
1      --  22.15  22.45      --          1
2   -0.21   23.2   23.4     117       3430
3      --   28.1   28.4      --        631
4    0.23  33.15  33.45       8       1755
2025-02-13
```

Out[32]:

```
110
```

Збереження даних

In [34]:

```
df = pd.DataFrame({
   'Субдифузійна модель(Спосіб 1)': C_sub_val1,
   'Субдифузійна модель(Спосіб 2)': C_sub_val2[1:253,K_plot_dif],
   'Стандартна модель Блека-Шоулза': C_val_BS
})


file_path = 'models_result.xlsx'
df.to_excel(file_path, index=False)
```

Завантаження

In [36]:

```
C_models_result = pd.read_excel('models_result.xlsx')
C_models_result = C_models_result.reset_index()
print(C_models_result)

oprion_dates = np.sort(OptionPrice['Exp. Date'].unique())
oprion_dates = pd.DataFrame(oprion_dates, columns = ['Exp. Date'])
oprion_dates['Exp. Date'] = pd.to_datetime(oprion_dates['Exp. Date']).dt.strftime('%Y-%m-%d')
oprion_dates['busday diff'] = oprion_dates['Exp. Date'].apply(lambda x: np.busday_count(start_date, x))
oprion_dates['busday diff'] = oprion_dates['busday diff'] - 1
oprion_dates
```

```
   index  Субдифузійна модель(Спосіб 1)  Субдифузійна модель(Спосіб 2)  \
0      0                       4.459847                       5.433990
```

```
1      1              5.330819              6.267608
2      2              5.919180              6.828747
3      3              6.376631              7.268299
4      4              6.756304              7.636654
..   ...                   ...                   ...
247  247             18.811093             23.410490
248  248             18.831071             23.445829
249  249             18.850989             23.481102
250  250             18.870849             23.516309
251  251             18.890650             23.551450

    Стандартна модель Блека-Шоулза
0                    1.332322
1                    1.890423
2                    2.321082
3                    2.685748
4                    3.008226
..                        ...
247                 22.898473
248                 22.947427
249                 22.996289
250                 23.045059
251                 23.093738

[252 rows x 4 columns]
```

Out[36]:

	Exp. Date	busday diff
0	2025-02-21	5
1	2025-03-21	25
2	2025-04-17	44
3	2025-05-16	65
4	2025-06-20	90
5	2025-07-18	110
6	2025-08-15	130
7	2025-09-19	155
8	2025-10-17	175
9	2025-11-21	200

	Exp. Date	busday diff
10	2025-12-19	220
11	2026-01-16	240

Об'єднання

In [38]:

```
C_dates = pd.merge(oprion_dates, C_models_result, left_on='busday diff', right_on='index',
how='left').drop(columns=['busday diff', 'index'])
real_price = OptionPrice[OptionPrice['Strike'] == K_merge][['Exp. Date','Last']]
real_price['Exp. Date'] = pd.to_datetime(real_price['Exp. Date']).dt.strftime('%Y-%m-%d')
C_dates = pd.merge(C_dates, real_price, left_on='Exp. Date', right_on='Exp. Date',
how='left').rename(columns={'Last': 'Реальна ціна'})
C_dates = C_dates.set_index('Exp. Date')
C_dates = C_dates.dropna()
C_dates
```

Out[38]:

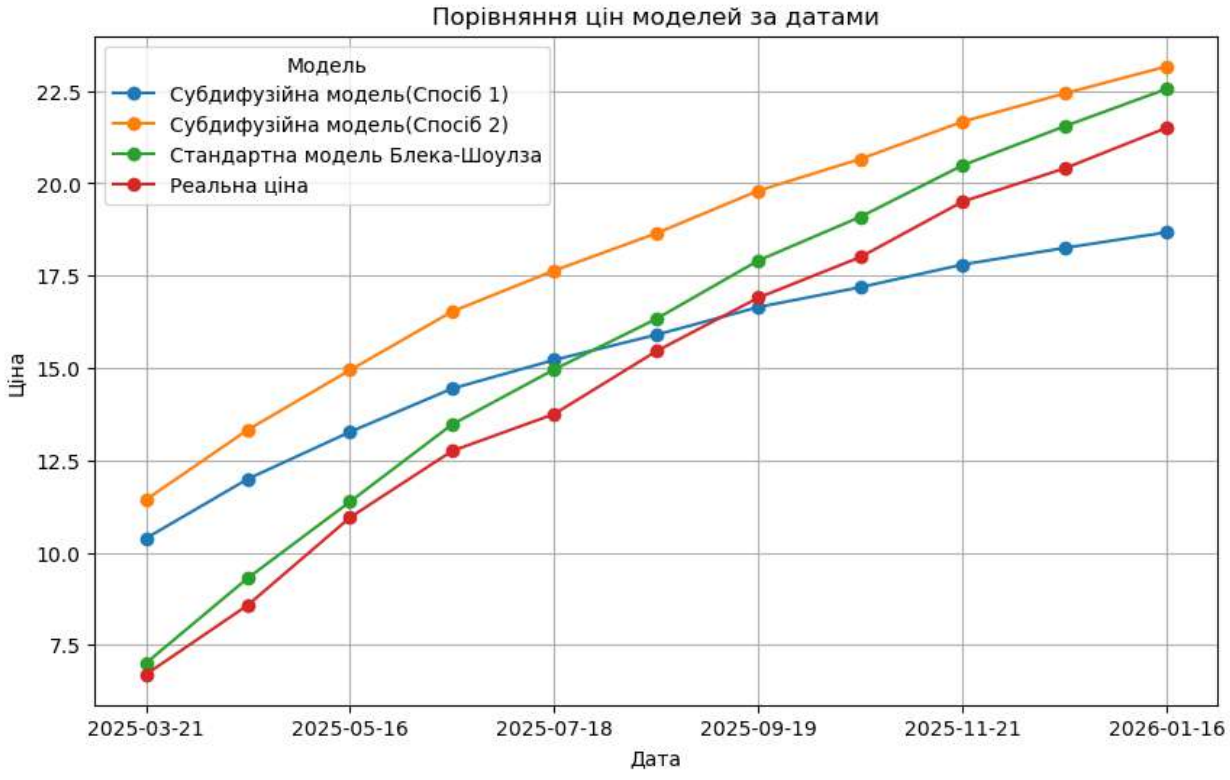
Exp. Date	Субдифузійна модель(Спосіб 1)	Субдифузійна модель(Спосіб 2)	Стандартна модель Блека-Шоулза	Реальна ціна
2025-03-21	10.385482	11.430188	7.003773	6.7
2025-04-17	11.994422	13.325633	9.312592	8.58
2025-05-16	13.266494	14.939495	11.377206	10.95
2025-06-20	14.438533	16.525806	13.469069	12.75
2025-07-18	15.215347	17.633399	14.956644	13.75
2025-08-15	15.895237	18.641515	16.325099	15.45
2025-09-19	16.645103	19.797098	17.906533	16.9

Exp. Date	Субдифузійна модель(Спосіб 1)	Субдифузійна модель(Спосіб 2)	Стандартна модель Блека-Шоулза	Реальна ціна
2025-10-17	17.183575	20.656254	19.088826	18
2025-11-21	17.796601	21.665294	20.482221	19.5
2025-12-19	18.247682	22.429442	21.539649	20.4
2026-01-16	18.669556	23.161211	22.553153	21.5

Графік порівняння

In [40]:

```
C_dates.plot(marker='o', figsize=(10, 6))
plt.title("Порівняння цін моделей за датами")
plt.xlabel("Дата")
plt.ylabel("Ціна")
plt.grid(True)
plt.legend(title='Модель')
plt.show()
```



Аналіз 1

1. Метрики точності
2. Тести на нормальність залишків
3. Тести на автокореляції залишків

In [42]:

```
real_data = np.array(C_dates['Реальна ціна'], dtype=object).astype(float)

# Модель Black-Scholes (BS)
pred_BS = np.array(C_dates['Стандартна модель Блека-Шоулза'], dtype=object).astype(float)

# субдифузійна модель 1 (SD1)
pred_SD1 = np.array(C_dates['Субдифузійна модель(Спосіб 1)'], dtype=object).astype(float)

# субдифузійна модель 2 (SD2)
pred_SD2 = np.array(C_dates['Субдифузійна модель(Спосіб 2)'], dtype=object).astype(float)

n = real_data.shape[0]
cut = 4

idx_middle = slice(4, 8)
idx_all = slice(0, n)

datasets = {
    "середньостроковий період": idx_middle,
    "весь період": idx_all
}

# ================================
# 1. Обчислення метрик точності
# ================================

def print_error_metrics(true, pred, label):
    """
    Обчислює та виводить основні метрики помилки:
    - MAE  (Mean Absolute Error)
    - MSE  (Mean Squared Error)
    - RMSE (Root Mean Squared Error)
    - ARPE (Average Relative Pricing Error)
    """
    mae = mean_absolute_error(true, pred)
    mse = mean_squared_error(true, pred)
    rmse = np.sqrt(mse)
    arpe = np.mean(np.abs(true - pred) / true)
    print(f"{label}:")
    print(f"  MAE  = {mae:.3f}")
    print(f"  MSE  = {mse:.3f}")
    print(f"  RMSE = {rmse:.3f}")
    print(f"  ARPE = {arpe:.3f}\n")

def print_normality_tests(resid, label):
    shapiro_test = shapiro(resid)
    jb_test = jarque_bera(resid)
    print(f"{label}:")
    print(f"  Shapiro-Wilk p-value: {shapiro_test.pvalue:.4f}")
    print(f"  Jarque-Bera p-value: {jb_test[1]:.4f}\n")
```

```
for name, idx in datasets.items():
    print(f"=== Метрики для {name} ===\n")

    y_true_BS  = real_data[idx]
    y_pred_BS  = pred_BS[idx]
    y_pred_SD1 = pred_SD1[idx]
    y_pred_SD2 = pred_SD2[idx]

    print("Метрики точності:")
    print_error_metrics(y_true_BS, y_pred_BS, "Black-Scholes")
    print_error_metrics(y_true_BS, y_pred_SD1, "Subdiffusive model 1")
    print_error_metrics(y_true_BS, y_pred_SD2, "Subdiffusive model 2")

    # ====================================
    # 2. Аналіз залишків (residual analysis)
    # ====================================

    resid_BS = y_true_BS - y_pred_BS
    resid_SD1 = y_true_BS - y_pred_SD1
    resid_SD2 = y_true_BS - y_pred_SD2


    print("Тести на нормальність залишків:")
    print_normality_tests(resid_BS, "Black-Scholes")
    print_normality_tests(resid_SD1, "Subdiffusive model 1")
    print_normality_tests(resid_SD2, "Subdiffusive model 2")

    # ====================================
    # 3. Аналіз автокореляції залишків
    # ====================================

    dw_BS = durbin_watson(resid_BS)
    dw_SD1 = durbin_watson(resid_SD1)
    dw_SD2 = durbin_watson(resid_SD2)

    print("Тести на автокореляції залишків :")
    print("Durbin-Watson для Black-Scholes:", dw_BS)
    print("Durbin-Watson для Subdiffusive model 1:", dw_SD1)
    print("Durbin-Watson для Subdiffusive model 2:", dw_SD2)
    print("\n")
```

```
=== Метрики для середньостроковий період ===

Метрики точності:
Black-Scholes:
  MAE  = 1.044
  MSE  = 1.105
  RMSE = 1.051
  ARPE = 0.066

Subdiffusive model 1:
  MAE  = 0.745
  MSE  = 0.769
  RMSE = 0.877
  ARPE = 0.049

Subdiffusive model 2:
```

MAE = 3.157
MSE = 10.179
RMSE = 3.190
ARPE = 0.202

Тести на нормальність залишків:
Black-Scholes:
Shapiro-Wilk p-value: 0.9977
Jarque-Bera p-value: 0.8823

Subdiffusive model 1:
Shapiro-Wilk p-value: 0.8921
Jarque-Bera p-value: 0.8437

Subdiffusive model 2:
Shapiro-Wilk p-value: 0.6541
Jarque-Bera p-value: 0.8007

Тести на автокореляції залишків :
Durbin-Watson для Black-Scholes: 0.030306682898762 96
Durbin-Watson для Subdiffusive model 1: 0.5999759276002898
Durbin-Watson для Subdiffusive model 2: 0.015310918474656578

=== Метрики для весь період ===

Метрики точності:
Black-Scholes:
MAE = 0.867
MSE = 0.829
RMSE = 0.911
ARPE = 0.059

Subdiffusive model 1:
MAE = 1.888
MSE = 4.735
RMSE = 2.176
ARPE = 0.165

Subdiffusive model 2:
MAE = 3.248
MSE = 11.577
RMSE = 3.403
ARPE = 0.274

Тести на нормальність залишків:
Black-Scholes:
Shapiro-Wilk p-value: 0.2104
Jarque-Bera p-value: 0.5264

Subdiffusive model 1:
Shapiro-Wilk p-value: 0.7144
Jarque-Bera p-value: 0.6941

Subdiffusive model 2:
Shapiro-Wilk p-value: 0.5873
Jarque-Bera p-value: 0.7010

Тести на автокореляції залишків :
Durbin-Watson для Black-Scholes: 0.08525399663326999
Durbin-Watson для Subdiffusive model 1: 0.09633682778535661
Durbin-Watson для Subdiffusive model 2: 0.01293832580705588

Аналіз 2

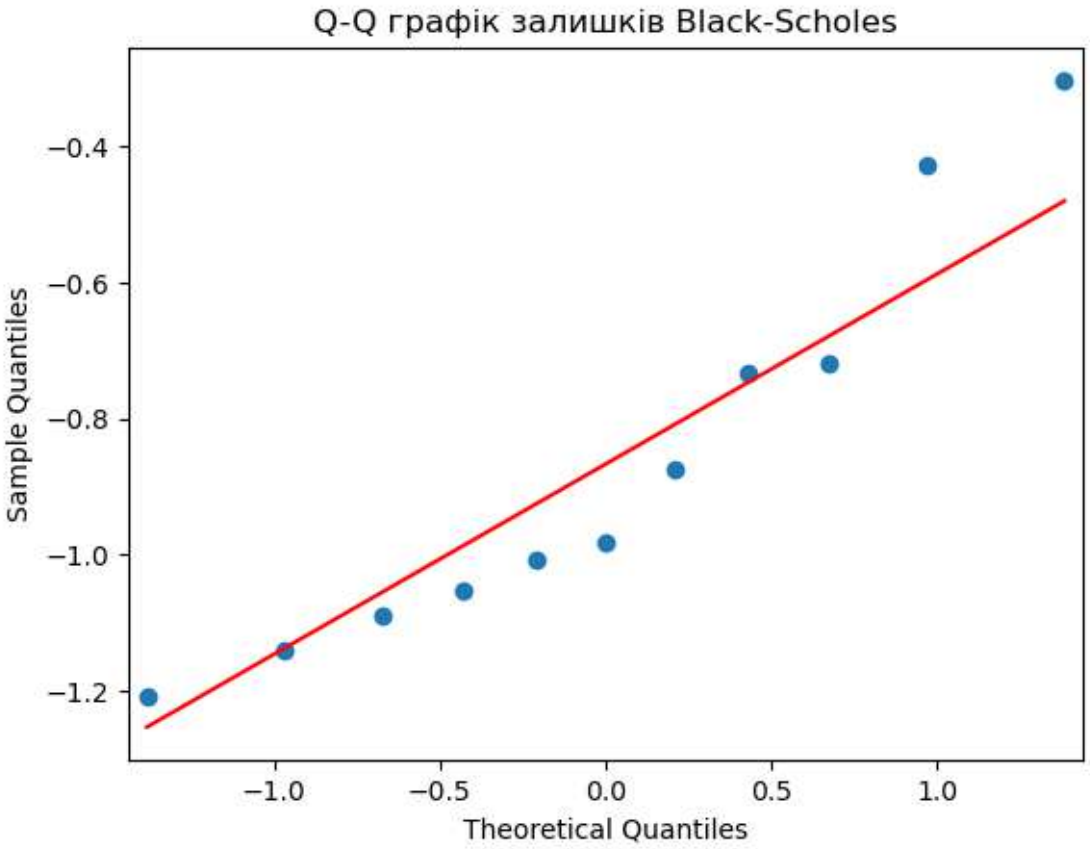
1. Q_Q plot

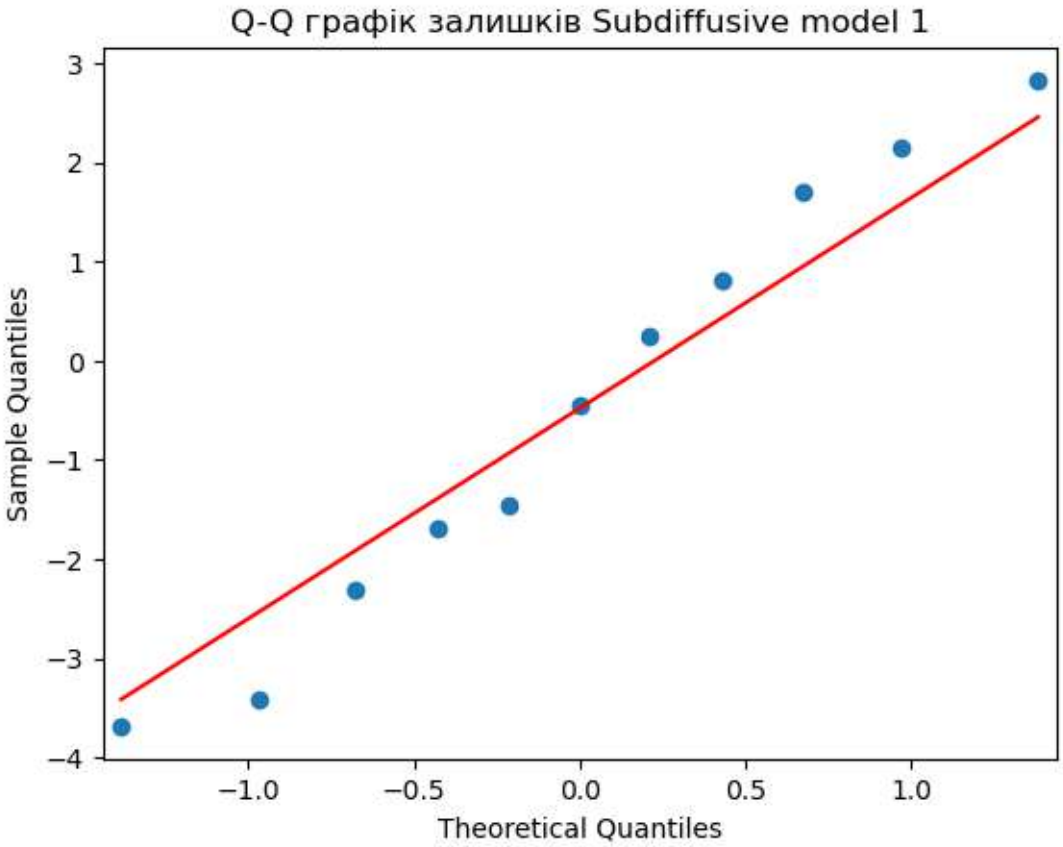
In [44]:

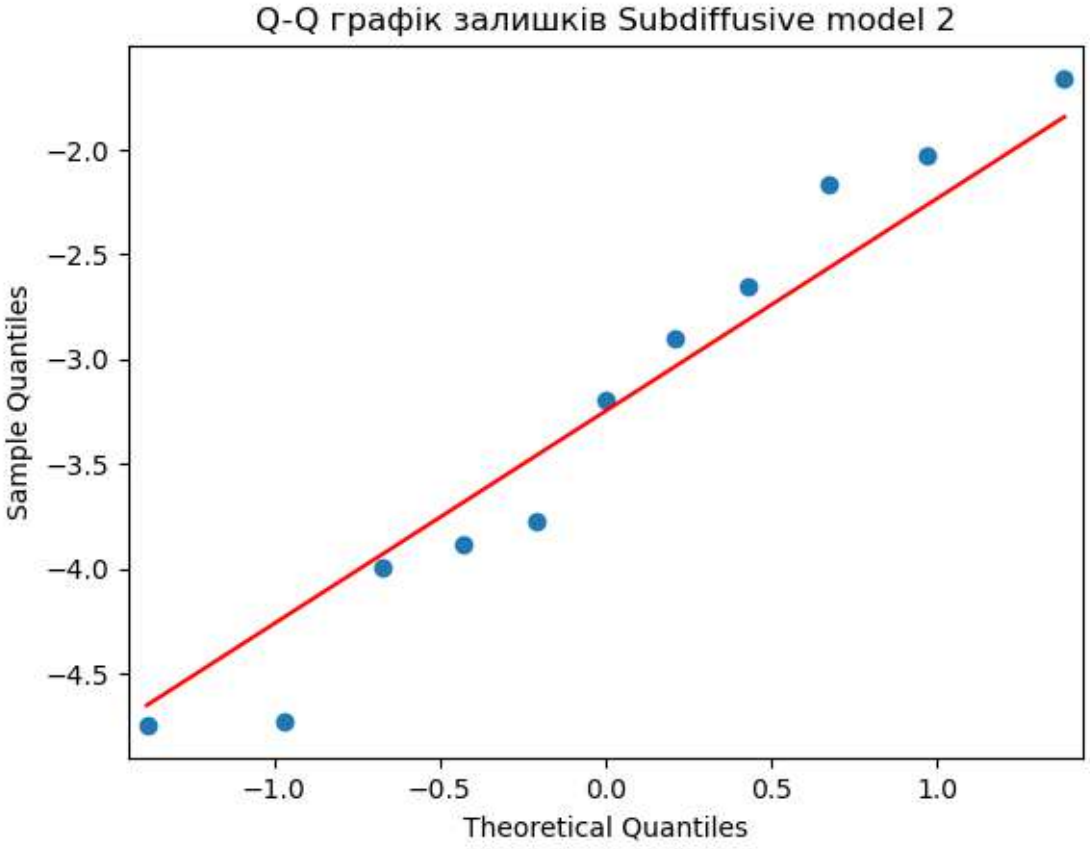
```
sm.qqplot(resid_BS, line='s')
plt.title("Q-Q графік залишків Black-Scholes")
plt.show()

sm.qqplot(resid_SD1, line='s')
plt.title("Q-Q графік залишків Subdiffusive model 1")
plt.show()

sm.qqplot(resid_SD2, line='s')
plt.title("Q-Q графік залишків Subdiffusive model 2")
plt.show()
```







Локальна волатильність

Сворення локальної волатильності

In [12]:

```
def sqrt_func(x, a, b):
    return a * np.sqrt(x) + b

def volatility(sigma,n_t):
    sigma_local = []
    sigma_local.append(sigma)
    for i in range(1,n_t):
        beta_0, beta_1, gamma =  9.99999674e-01, -1.09660059e-11,  1.62917518
        sigma_local.append(sigma_local[i-1]*(beta_0 + beta_1 * i ** gamma + beta_1 * i ** (2 * gamma)))


    return sigma_local

sigma_local = volatility(sigma,n_t)
```

C_H local

In [49]:

```
def compute_C_H_local(n_t, n_k, R_k, R2_k, K_j,alpha,r,sigma_local,A_0,delta_t):
    """
    Compute C_H(t, k) iteratively over time steps.

    Parameters:
    - n_t: Number of time steps.
    - n_k: Number of strike price steps.
    - delta_t: Step size in the t-grid.
    - alpha: Fractional order of the derivative.
    - R_k: First derivative matrix.
    - R2_k: Second derivative matrix.
    - D_k: Fractional derivative matrix.

    Returns:
    - C_H: Matrix of C_H values over time and k.
    """
    I_nk_1 = np.identity(n_k + 1)
    coef = (delta_t**(-alpha)) / gamma(1 - alpha) * 1/np.sqrt(2)


    C_H = np.zeros((n_t+1, n_k + 1))
    K_1_d = np.diag(K_j)
    K_2_d = np.diag(np.array(K_j) ** 2)

    for j in range(len(K_j)):
        C_H[0, j] = np.maximum(A_0 - K_j[j], 0)


    I_nk_1_gpu = cp.array(I_nk_1,    dtype=cp.float64)
    K_1_d_gpu  = cp.array(K_1_d,     dtype=cp.float64)
    R_k_gpu    = cp.array(R_k,       dtype=cp.float64)
    K_2_d_gpu  = cp.array(K_2_d,     dtype=cp.float64)
    R2_k_gpu   = cp.array(R2_k,      dtype=cp.float64)
    coef_gpu   = float(coef)
```

```
    C_H_gpu = cp.array(C_H, dtype=cp.float64)

    for k in tqdm(range(1, n_t+1), desc="Time steps on GPU"):
        M = (coef_gpu * I_nk_1_gpu
             + (r/2) * K_1_d_gpu @ R_k_gpu
             - ((sigma_local[i-1]**2) / 4) * (K_2_d_gpu @ R2_k_gpu))

        invM = cp.linalg.inv(M)

        I_k_1    = cp.ones((k-1,1), dtype=cp.float64)
        C_H_k1   = C_H_gpu[k-1,:].reshape(-1,1)
        D_k_gpu      = fractional_derivative_matrix(delta_t, alpha, k)

        multiplier2 = (C_H_k1 * coef_gpu
                       - (1/cp.sqrt(2)) * (D_k_gpu @ C_H_gpu[:k,:]).T @ I_k_1)

        C_H_gpu[k,:] = (invM @ multiplier2).T

    C_H_result = cp.asnumpy(C_H_gpu)
    return C_H_result
```

Розрахунок Блека Шоуза

In [52]:

```
N = Normal('N', 0.0, 1.0)

def d1(S, K, T, r, sigma):
    return (ln(S / K) + (r + 1/2 * sigma ** 2) * T) / (sigma * sqrt(T))

def d2(S, K, T, r, sigma):
    return (ln(S / K) + (r - 1/2 * sigma ** 2) * T) / (sigma * sqrt(T))

def C(S, K, T, r, sigma):
    return S * cdf(N)(d1(S, K, T, r, sigma)) - K * exp(-r*T)* cdf(N)(d2(S, K, T, r, sigma))
```

In [53]:

```
def compute_values(i):
    try:
        t = i / 252.0
        c_val = C(A_0, K, t, r, sigma).evalf()
        return i, c_val
    except Exception as e:
        print(f"Error at i={i}: {e}")
        return i, None, None

results = Parallel(n_jobs=-1, verbose=10)(
    delayed(compute_values)(i) for i in range(1, 253)
)

C_val_BS = [item[1] for item in results]
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 24 concurrent workers.
[Parallel(n_jobs=-1)]: Batch computation too fast (0.010439872741699219s.) Setting batch_size=2.
[Parallel(n_jobs=-1)]: Done   2 tasks      | elapsed:    0.0s
[Parallel(n_jobs=-1)]: Done  13 tasks      | elapsed:    0.0s
[Parallel(n_jobs=-1)]: Done  24 tasks      | elapsed:    0.0s
[Parallel(n_jobs=-1)]: Done  37 tasks      | elapsed:    0.0s
[Parallel(n_jobs=-1)]: Batch computation too fast (0.021210908889770508s.) Setting batch_size=4.
[Parallel(n_jobs=-1)]: Done  52 tasks      | elapsed:    0.0s
[Parallel(n_jobs=-1)]: Done  82 tasks      | elapsed:    0.0s
```

```
[Parallel(n_jobs=-1)]: Done 112 tasks      | elapsed:    0.0s
[Parallel(n_jobs=-1)]: Done 252 out of 252 | elapsed:    0.0s finished
```

Розрахунок Субдифузивної моделі через інтеграл(Спосіб 1)

In [56]:

```
def h(x, T):
    return sqrt(2 / (pi * T)) * E ** (-(x ** 2)/(2*T))

def IG_sub_C(x, S, K, T, r, sigma):
    return Integral(C(S, K, x, r, sigma)*h(x, T),(x, 0, oo))
```

In [57]:

```
x = symbols('x')

def compute_values(i):
    try:
        t = i / 252.0
        C_sub_val1_local = IG_sub_C(x, A_0, K, t, r, sigma_local[i-1]).evalf()
        return i, C_sub_val1_local
    except Exception as e:
        print(f"Error at i={i}: {e}")
        return i, None, None

results = Parallel(n_jobs=-1, verbose=10)(
    delayed(compute_values)(i) for i in range(1, 253)
)

C_sub_val1_local = [item[1] for item in results]
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 24 concurrent workers.
[Parallel(n_jobs=-1)]: Done   2 tasks      | elapsed:   21.3s
[Parallel(n_jobs=-1)]: Done  13 tasks      | elapsed:   22.1s
[Parallel(n_jobs=-1)]: Done  24 tasks      | elapsed:   25.6s
[Parallel(n_jobs=-1)]: Done  37 tasks      | elapsed:   45.3s
[Parallel(n_jobs=-1)]: Done  50 tasks      | elapsed:  1.1min
[Parallel(n_jobs=-1)]: Done  65 tasks      | elapsed:  1.3min
[Parallel(n_jobs=-1)]: Done  80 tasks      | elapsed:  1.4min
[Parallel(n_jobs=-1)]: Done  97 tasks      | elapsed:  1.6min
[Parallel(n_jobs=-1)]: Done 114 tasks      | elapsed:  1.7min
[Parallel(n_jobs=-1)]: Done 133 tasks      | elapsed:  1.8min
[Parallel(n_jobs=-1)]: Done 152 tasks      | elapsed:  1.9min
[Parallel(n_jobs=-1)]: Done 173 tasks      | elapsed:  2.1min
[Parallel(n_jobs=-1)]: Done 194 tasks      | elapsed:  2.2min
[Parallel(n_jobs=-1)]: Done 231 out of 252 | elapsed:  2.5min remaining:   13.4s
[Parallel(n_jobs=-1)]: Done 252 out of 252 | elapsed:  2.6min finished
```

Розрахунок Субдифузивної моделі через наближення(Спосіб 2)

In [59]:

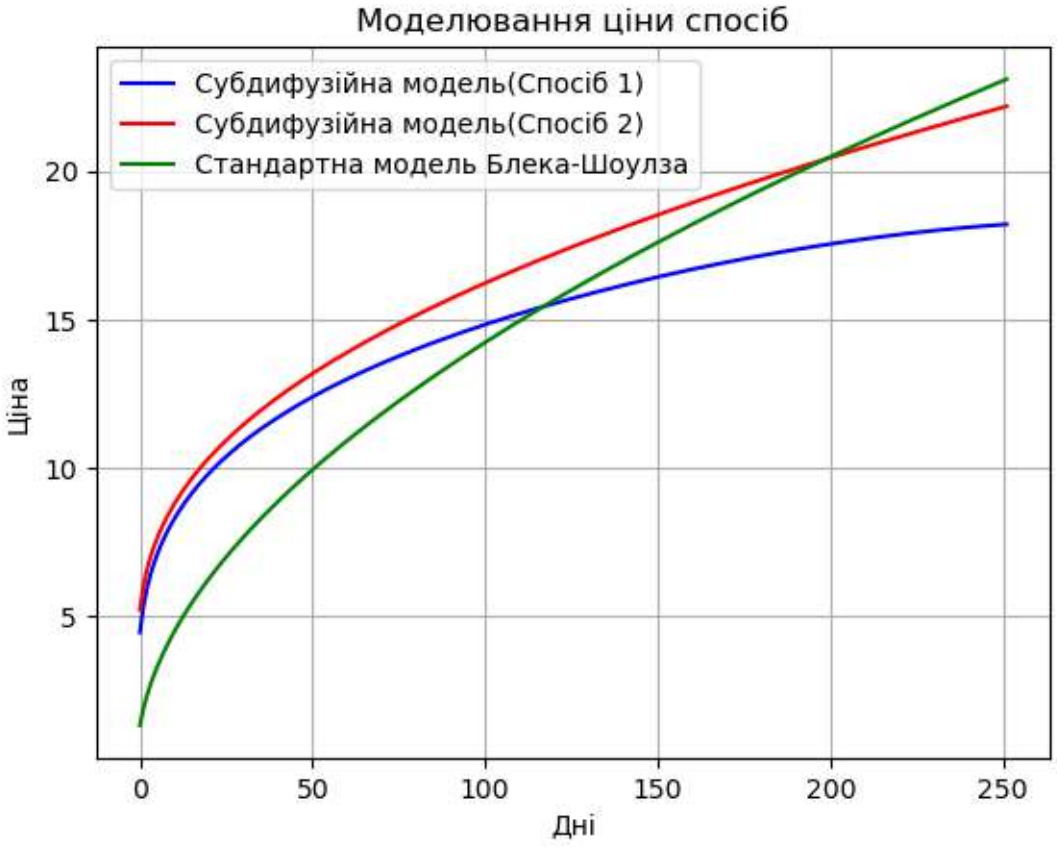
```
R_k = finite_difference_matrix(n_k, Delta_K)
R2_k = second_derivative_matrix(n_k, Delta_K)
C_sub_val2_local = compute_C_H_local(n_t, n_k, R_k, R2_k, K_j,alpha,r,sigma_local,A_0,delta_t)
```

```
Time steps on GPU:   0%|          | 0/252 [00:00<?, ?it/s]
```

Побудова графіків

In [61]:

```
plt.plot(range(n_t), C_sub_val1_local[0:], color = 'blue', label='Субдифузійна модель(Спосіб 1)')
plt.plot(range(n_t), C_sub_val2_local[1:,K_plot_dif], color = 'red', label='Субдифузійна модель(Спосіб 2)')
plt.plot(range(n_t), C_val_BS[0:], color = 'green', label='Стандартна модель Блека-Шоулза')
plt.title('Моделювання ціни спосіб')
plt.xlabel('Дні')
plt.ylabel('Ціна')
plt.legend()
plt.grid(True)
plt.show()
```



Реальні дані.

In [63]:

```
OptionPrice = pd.read_excel('data.xlsx', sheet_name='Option')
print(OptionPrice.head())

start_date = pd.read_excel('data.xlsx', sheet_name='S').loc[0, 'Date'].strftime('%Y-%m-%d')
print(start_date)

range_strikes = (OptionPrice['Strike'] >= K - 1) & (OptionPrice['Strike'] <= K + 1)
real_price = OptionPrice[range_strikes][['Strike', 'Exp. Date', 'Last']]

real_price['Exp. Date'] = pd.to_datetime(real_price['Exp. Date']).dt.strftime('%Y-%m-%d')

real_price['busday diff'] = real_price['Exp. Date'].apply(lambda x: np.busday_count(start_date, x))
real_price['busday diff'] = real_price['busday diff'] - 1
real_price
```

```
K_merge = real_price['Strike'].max()
K_merge
```

```
  Exp. Date  Last  Change   Bid   Ask Volume Open Int.  Strike Last.1  \
0 2025-02-21  0.05  -0.01  0.05  0.06     12      597    133   22.1
1 2025-02-21  0.05  -0.01  0.05  0.06    663     5880    134   23.9
2 2025-02-21  0.05  -0.01  0.04  0.05    512    17088    135  23.55
3 2025-02-21  0.03     --  0.02  0.03   1028    20719    140  27.79
4 2025-02-21  0.01  -0.02  0.01  0.02   1053    21076    145  34.05

  Change.1  Bid.1  Ask.1 Volume.1 Open Int..1
0      --   21.2  21.45      --         --
1      --  22.15  22.45      --          1
2   -0.21   23.2   23.4     117       3430
3      --   28.1   28.4      --        631
4    0.23  33.15  33.45       8       1755
2025-02-13
```

Out[63]:

```
110
```

Збереження даних

In [65]:

```
df = pd.DataFrame({
   'Субдифузійна модель локальна волатильність(Спосіб 1)': C_sub_val1_local,
   'Субдифузійна модель локальна волатильність(Спосіб 2)': C_sub_val2_local[1:253,K_plot_dif],
   'Стандартна модель Блека-Шоулза': C_val_BS
})


file_path = 'models_result_local.xlsx'
df.to_excel(file_path, index=False)
```

Завантаження

In [67]:

```
C_models_result_local = pd.read_excel('models_result_local.xlsx')
C_models_result_local = C_models_result_local.reset_index()
print(C_models_result_local)

oprion_dates = np.sort(OptionPrice['Exp. Date'].unique())
oprion_dates = pd.DataFrame(oprion_dates, columns = ['Exp. Date'])
oprion_dates['Exp. Date'] = pd.to_datetime(oprion_dates['Exp. Date']).dt.strftime('%Y-%m-%d')
oprion_dates['busday diff'] = oprion_dates['Exp. Date'].apply(lambda x: np.busday_count(start_date, x))
oprion_dates['busday diff'] = oprion_dates['busday diff'] - 1
oprion_dates
```

```
     index  Субдифузійна модель локальна волатильність(Спосіб 1)  \
0        0                                          4.459847
1        1                                          5.330817
2        2                                          5.919177
3        3                                          6.376625
4        4                                          6.756296
..     ...                                               ...
247    247                                         18.172737
248    248                                         18.181313
249    249                                         18.189677
250    250                                         18.197828
251    251                                         18.205765
```

```
    Субдифузійна модель локальна волатильність(Спосіб 2)  \
0                                    5.219254
1                                    6.019153
2                                    6.556867
3                                    6.977550
4                                    7.329692
..                                        ...
247                                 22.053074
248                                 22.085452
249                                 22.117768
250                                 22.150021
251                                 22.182211

    Стандартна модель Блека-Шоулза
0                    1.332322
1                    1.890423
2                    2.321082
3                    2.685748
4                    3.008226
..                        ...
247                 22.898473
248                 22.947427
249                 22.996289
250                 23.045059
251                 23.093738

[252 rows x 4 columns]
```

Out[67]:

	Exp. Date	busday diff
0	2025-02-21	5
1	2025-03-21	25
2	2025-04-17	44
3	2025-05-16	65
4	2025-06-20	90
5	2025-07-18	110
6	2025-08-15	130
7	2025-09-19	155
8	2025-10-17	175

	Exp. Date	busday diff
9	2025-11-21	200
10	2025-12-19	220
11	2026-01-16	240

Об'єднання

In [69]:

```
C_dates_local = pd.merge(oprion_dates, C_models_result_local, left_on='busday diff', right_on='index',
how='left').drop(columns=['busday diff', 'index'])
real_price_local = OptionPrice[OptionPrice['Strike'] == K_merge][['Exp. Date','Last']]
real_price_local['Exp. Date'] = pd.to_datetime(real_price_local['Exp. Date']).dt.strftime('%Y-%m-%d')
C_dates_local = pd.merge(C_dates_local, real_price_local, left_on='Exp. Date', right_on='Exp. Date',
how='left').rename(columns={'Last': 'Реальна ціна'})
C_dates_local = C_dates_local.set_index('Exp. Date')
C_dates_local = C_dates_local.dropna()
C_dates_local
```

Out[69]:

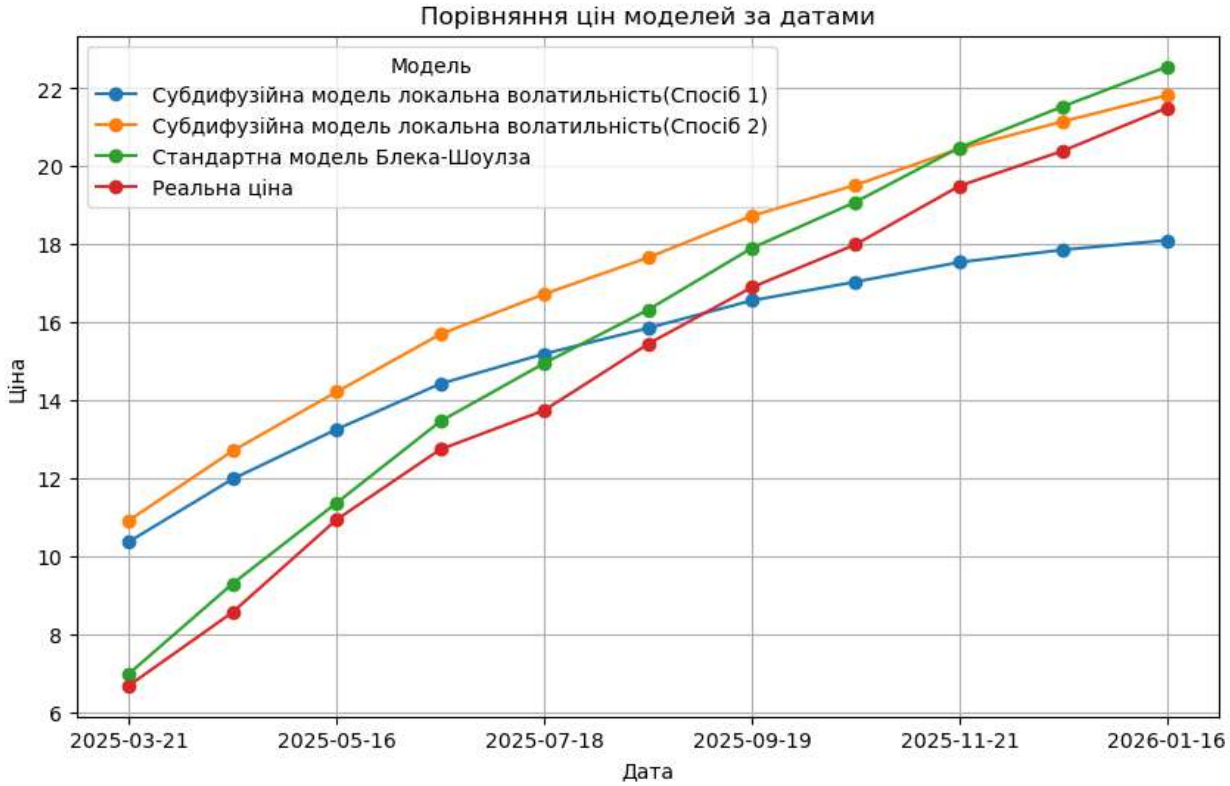
Exp. Date	Субдифузійна модель локальна волатильність(Спосіб 1)	Субдифузійна модель локальна волатильність(Спосіб 2)	Стандартна модель Блека-Шоулза	Реальна ціна
2025-03-21	10.385379	10.931051	7.003773	6.7
2025-04-17	11.993970	12.713298	9.312592	8.58
2025-05-16	13.264568	14.222938	11.377206	10.95
2025-06-20	14.431019	15.700565	13.469069	12.75
2025-07-18	15.197391	16.728914	14.956644	13.75
2025-08-15	15.857818	17.662652	16.325099	15.45

Exp. Date	Субдифузійна модель локальна волатильність(Спосіб 1)	Субдифузійна модель локальна волатильність(Спосіб 2)	Стандартна модель Блека-Шоулза	Реальна ціна
2025-09-19	16.563668	18.730457	17.906533	16.9
2025-10-17	17.044159	19.522670	19.088826	18
2025-11-21	17.544770	20.451315	20.482221	19.5
2025-12-19	17.864000	21.153333	21.539649	20.4
2026-01-16	18.106834	21.824617	22.553153	21.5

Графік порівняння

In [71]:

```
C_dates_local.plot(marker='o', figsize=(10, 6))
plt.title("Порівняння цін моделей за датами")
plt.xlabel("Дата")
plt.ylabel("Ціна")
plt.grid(True)
plt.legend(title='Модель')
plt.show()
```



Аналіз 1

1. Метрики точності
2. Тести на нормальність залишків
3. Тести на автокореляції залишків

In [73]:

```
real_data_l = np.array(C_dates_local['Реальна ціна'], dtype=object).astype(float)

pred_BS_l = np.array(C_dates_local['Стандартна модель Блека-Шоулза'], dtype=object).astype(float)

pred_SD1_l = np.array(C_dates_local['Субдифузійна модель локальна волатильність(Спосіб 1)'],
dtype=object).astype(float)

pred_SD2_l = np.array(C_dates_local['Субдифузійна модель локальна волатильність(Спосіб 2)'],
dtype=object).astype(float)

n = real_data.shape[0]
cut = 4

idx_middle = slice(4, 8)
idx_long  = slice(n - cut, n)

idx_all = slice(0, n)

datasets = {
   "середньостроковий період": idx_middle,
   "довгостроковий період":    idx_long,
   "весь період": idx_all
}
```

```
# ================================
# 1. Обчислення метрик точності
# ================================

def print_error_metrics(true, pred, label):
    """
    Обчислює та виводить основні метрики помилки:
    - MAE  (Mean Absolute Error)
    - MSE  (Mean Squared Error)
    - RMSE (Root Mean Squared Error)
    - ARPE (Average Relative Pricing Error)
    """
    mae = mean_absolute_error(true, pred)
    mse = mean_squared_error(true, pred)
    rmse = np.sqrt(mse)
    arpe = np.mean(np.abs(true - pred) / true)
    print(f"{label}:")
    print(f"  MAE  = {mae:.3f}")
    print(f"  MSE  = {mse:.3f}")
    print(f"  RMSE = {rmse:.3f}")
    print(f"  ARPE = {arpe:.3f}\n")

def print_normality_tests(resid, label):
    shapiro_test = shapiro(resid)
    jb_test = jarque_bera(resid)
    print(f"{label}:")
    print(f"  Shapiro-Wilk p-value: {shapiro_test.pvalue:.4f}")
    print(f"  Jarque-Bera p-value: {jb_test[1]:.4f}\n")



for name, idx in datasets.items():
    print(f"=== Метрики для {name} ===\n")
    y_true_BS_l  = real_data_l[idx]
    y_pred_BS_l  = pred_BS_l[idx]
    y_pred_SD1_l = pred_SD1_l[idx]
    y_pred_SD2_l = pred_SD2_l[idx]

    print("Метрики точності:")
    print_error_metrics(y_true_BS_l, y_pred_BS_l, "Black-Scholes")
    print_error_metrics(y_true_BS_l, y_pred_SD1_l, "Subdiffusive model 1")
    print_error_metrics(y_true_BS_l, y_pred_SD2_l, "Subdiffusive model 2")

    # ====================================
    # 2. Аналіз залишків (residual analysis)
    # ====================================

    # Обчислюємо залишки
    resid_BS_l = y_true_BS_l - y_pred_BS_l
    resid_SD1_l = y_true_BS_l - y_pred_SD1_l
    resid_SD2_l = y_true_BS_l - y_pred_SD2_l



    print("Тести на нормальність залишків:")
    print_normality_tests(resid_BS_l, "Black-Scholes")
    print_normality_tests(resid_SD1_l, "Subdiffusive model 1")
    print_normality_tests(resid_SD2_l, "Subdiffusive model 2")
```

```
    # ====================================
    # 3. Аналіз автокореляції залишків
    # ====================================

    dw_BS_l = durbin_watson(resid_BS_l)
    dw_SD1_l = durbin_watson(resid_SD1_l)
    dw_SD2_l = durbin_watson(resid_SD2_l)

    print("Тести на автокореляції залишків :")
    print("Durbin-Watson для Black-Scholes:", dw_BS_l)
    print("Durbin-Watson для Subdiffusive model 1:", dw_SD1_l)
    print("Durbin-Watson для Subdiffusive model 2:", dw_SD2_l)
    print("\n")
```

```
=== Метрики для середньостроковий період ===

Метрики точності:
Black-Scholes:
  MAE  = 1.044
  MSE  = 1.105
  RMSE = 1.051
  ARPE = 0.066

Subdiffusive model 1:
  MAE  = 0.787
  MSE  = 0.822
  RMSE = 0.907
  ARPE = 0.051

Subdiffusive model 2:
  MAE  = 2.136
  MSE  = 4.860
  RMSE = 2.204
  ARPE = 0.138

Тести на нормальність залишків:
Black-Scholes:
  Shapiro-Wilk p-value: 0.9977
  Jarque-Bera p-value: 0.8823

Subdiffusive model 1:
  Shapiro-Wilk p-value: 0.9157
  Jarque-Bera p-value: 0.8482

Subdiffusive model 2:
  Shapiro-Wilk p-value: 0.7496
  Jarque-Bera p-value: 0.8188

Тести на автокореляції залишків :
Durbin-Watson для Black-Scholes: 0.03030668289876296
Durbin-Watson для Subdiffusive model 1: 0.6138254172507788
Durbin-Watson для Subdiffusive model 2: 0.042593230613345856


=== Метрики для довгостроковий період ===

Метрики точності:
Black-Scholes:
```

MAE = 1.066
MSE = 1.140
RMSE = 1.068
ARPE = 0.054

Subdiffusive model 1:
MAE = 2.210
MSE = 5.670
RMSE = 2.381
ARPE = 0.109

Subdiffusive model 2:
MAE = 0.888
MSE = 0.974
RMSE = 0.987
ARPE = 0.046

Тести на нормальність залишків:
Black-Scholes:
Shapiro-Wilk p-value: 0.9679
Jarque-Bera p-value: 0.8762

Subdiffusive model 1:
Shapiro-Wilk p-value: 0.9950
Jarque-Bera p-value: 0.8836

Subdiffusive model 2:
Shapiro-Wilk p-value: 0.9458
Jarque-Bera p-value: 0.8897

Тести на автокореляції залишків :
Durbin-Watson для Black-Scholes: 0.009571626914347185
Durbin-Watson для Subdiffusive model 1: 0.09129963478371175
Durbin-Watson для Subdiffusive model 2: 0.14101210090382274

=== Метрики для весь період ===

Метрики точності:
Black-Scholes:
MAE = 0.867
MSE = 0.829
RMSE = 0.911
ARPE = 0.059

Subdiffusive model 1:
MAE = 2.012
MSE = 5.316
RMSE = 2.306
ARPE = 0.171

Subdiffusive model 2:
MAE = 2.287
MSE = 6.856
RMSE = 2.618
ARPE = 0.209

Тести на нормальність залишків:

```
Black-Scholes:
  Shapiro-Wilk p-value: 0.2104
  Jarque-Bera p-value: 0.5264

Subdiffusive model 1:
  Shapiro-Wilk p-value: 0.7548
  Jarque-Bera p-value: 0.7032

Subdiffusive model 2:
  Shapiro-Wilk p-value: 0.6442
  Jarque-Bera p-value: 0.7034

Тести на автокореляції залишків :
Durbin-Watson для Black-Scholes: 0.08525399663326999
Durbin-Watson для Subdiffusive model 1: 0.09965416308099205
Durbin-Watson для Subdiffusive model 2: 0.0295927973236136
```

Аналіз 2

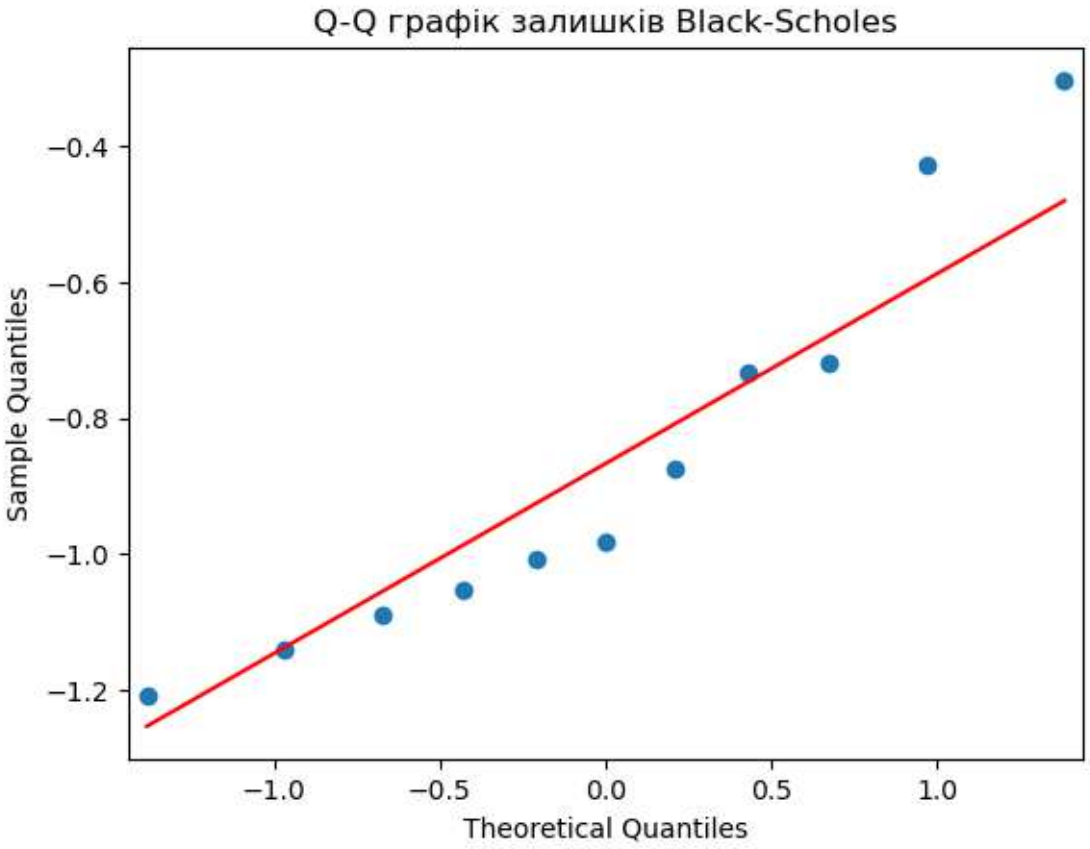
1. Q_Q plot

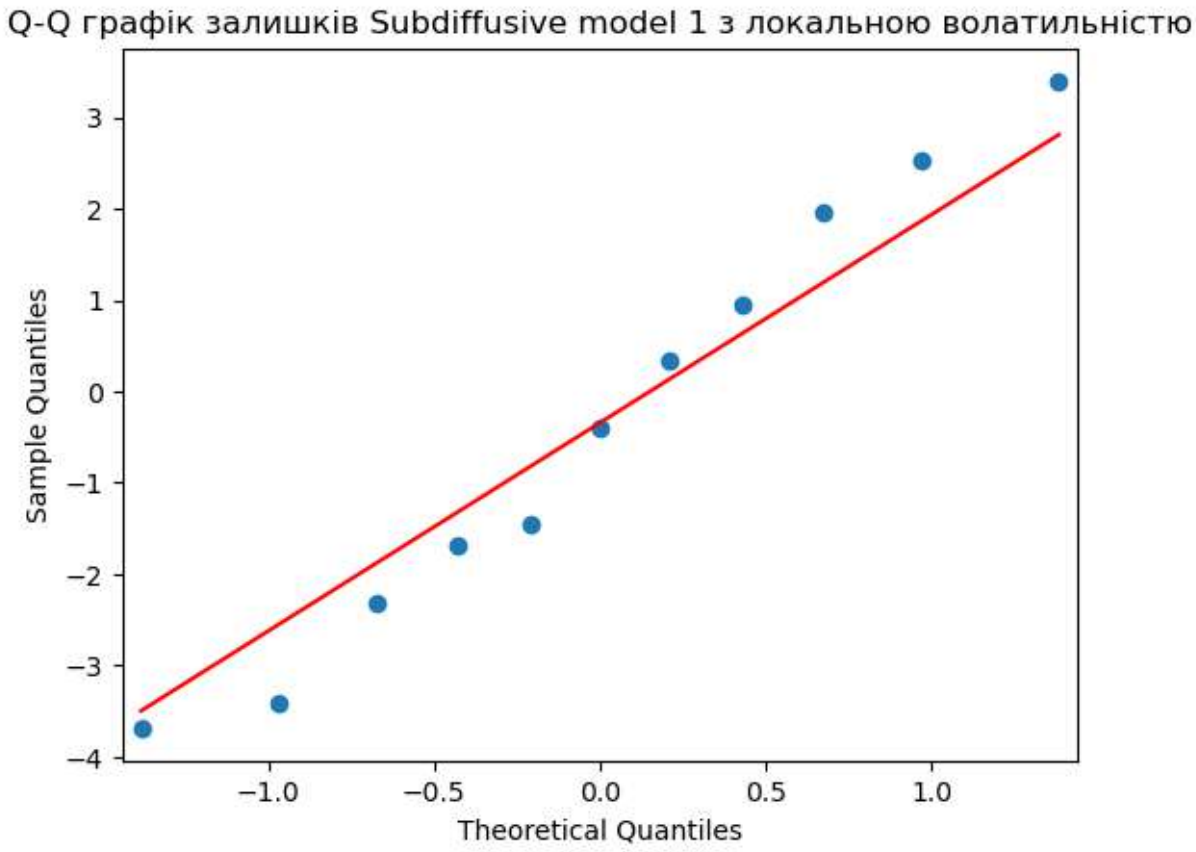
In [75]:

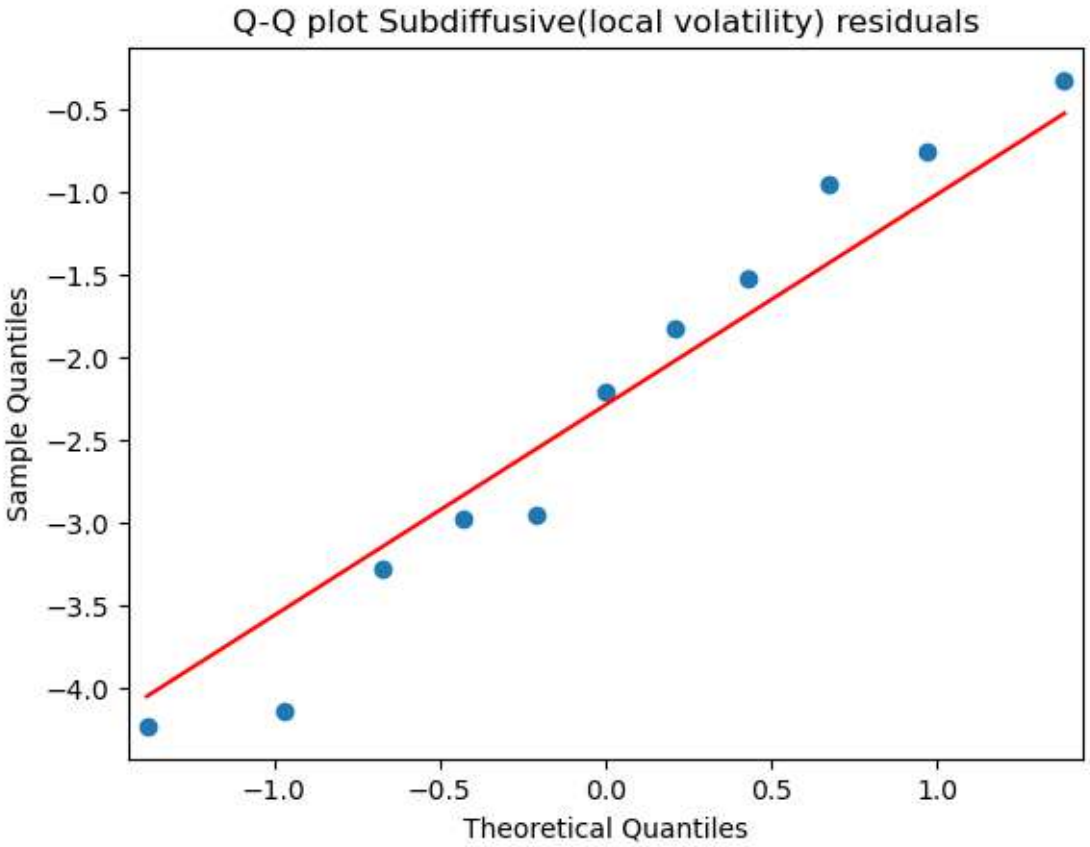
```
sm.qqplot(resid_BS_l, line='s')
plt.title("Q-Q графік залишків Black-Scholes")
plt.show()

sm.qqplot(resid_SD1_l, line='s')
plt.title("Q-Q графік залишків Subdiffusive model 1 з локальною волатильністю")
plt.show()

sm.qqplot(resid_SD2_l, line='s')
plt.title("Q-Q plot Subdiffusive(local volatility) residuals")
plt.show()
```


Q-Q графік залишків Black-Scholes
Sample Quantiles
Theoretical Quantiles
−0.4
−0.6
−0.8
−1.0
−1.2
−1.0
−0.5
0.0
0.5
1.0


Q-Q графік залишків Subdiffusive model 1 з локальною волатильністю
Sample Quantiles
Theoretical Quantiles
3
2
1
0
−1
−2
−3
−4
−1.0
−0.5
0.0
0.5
1.0



Загальне порівняння

Загальна таблиця

In [78]:

```
duplicate_cols = [col for col in C_dates.columns if col in C_dates_local.columns and col != 'Exp. Date']

C_dates_clean = C_dates.drop(columns=duplicate_cols)

C_all = pd.merge(C_dates_local, C_dates_clean, on='Exp. Date', how='left')

C_all
```

Out[78]:

	Субдифузійна модель локальна волатильність(Спосіб 1)	Субдифузійна модель локальна волатильність(Спосіб 2)	Стандартна модель Блека-Шоулза	Реальна ціна	Субдифузійна модель(Спосіб 1)	Субдифузійна модель(Спосіб 2)
Exp. Date						
2025-03-21	10.385379	10.931051	7.003773	6.7	10.385482	11.430188

Exp. Date	Субдифузійна модель локальна волатильність(Спосіб 1)	Субдифузійна модель локальна волатильність(Спосіб 2)	Стандартна модель Блека-Шоулза	Реальна ціна	Субдифузійна модель(Спосіб 1)	Субдифузійна модель(Спосіб 2)
2025-04-17	11.993970	12.713298	9.312592	8.58	11.994422	13.325633
2025-05-16	13.264568	14.222938	11.377206	10.95	13.266494	14.939495
2025-06-20	14.431019	15.700565	13.469069	12.75	14.438533	16.525806
2025-07-18	15.197391	16.728914	14.956644	13.75	15.215347	17.633399
2025-08-15	15.857818	17.662652	16.325099	15.45	15.895237	18.641515
2025-09-19	16.563668	18.730457	17.906533	16.9	16.645103	19.797098
2025-10-17	17.044159	19.522670	19.088826	18	17.183575	20.656254
2025-11-21	17.544770	20.451315	20.482221	19.5	17.796601	21.665294
2025-12-19	17.864000	21.153333	21.539649	20.4	18.247682	22.429442
2026-01-16	18.106834	21.824617	22.553153	21.5	18.669556	23.161211

Графік порівняння

In [80]:

```
C_all.plot(marker='o', figsize=(10, 6))
plt.title("Порівняння цін моделей за датами")
plt.xlabel("Дата")
plt.ylabel("Ціна")
plt.grid(True)
plt.legend(title='Модель')
plt.show()
```