DOI: https://doi.org/10.36910/6775-2524-0560-2025-60-24
УДК 004.42:519.2
Папка Олег Степанович, к. е. н.
https://orcid.org/0000-0002-8962-5247
Олійник Роман Володимирович, к. т. н.
https://orcid.org/0009-0008-4967-4926
Львівський торговельно-економічний університет, м. Львів, Україна

ПОРІВНЯЛЬНИЙ АНАЛІЗ РЕАЛІЗАЦІЇ КЛАСИЧНИХ СТАТИСТИЧНИХ МОДЕЛЕЙ ПРОГНОЗУВАННЯ ОДНОВИМІРНИХ ЧАСОВИХ РЯДІВ У PYTHON, R ТА JULIA

Папка О. С., Олійник Р. В. Порівняльний аналіз реалізації класичних статистичних моделей прогнозування одновимірних часових рядів у Python, R та Julia. В роботі здійснено порівняльний аналіз реалізації класичних статистичних моделей прогнозування одновимірних часових рядів у середовищах програмування Python, R та Julia. У дослідженні розглянуто наївні та сезонно-наївні методи, експоненційне згладжування ETS, а також автоматичний підбір параметрів авторегресійних інтегрованих моделей ковзного середнього (Auto-ARIMA). Для експериментів використано два набори даних: щоденні котирування нафти марки Brent та щомісячні показники середньої заробітної плати в Україні. Оцінювання точності прогнозів здійснено за трьома метриками — RMSE, MASE та sMAPE — з урахуванням різних горизонтів прогнозування. Дослідження показало, що результати моделювання залежать не лише від математичної природи алгоритмів, але й від специфіки їхньої реалізації у відповідних екосистемах. Встановлено, що Python і R забезпечують повноцінну підтримку всіх розглянутих моделей, тоді як у Julia на момент проведення дослідження відсутня стабільна реалізація ETS та Auto-ARIMA, через що можливості порівняння були частково обмежені. Отримані результати мають значення як для оцінювання зрілості програмних екосистем у сфері часових рядів, так і для практичного вибору інструментів у прикладних задачах прогнозування.
Ключові слова: прогнозування часових рядів, Python, R, Julia, ETS, Auto-ARIMA, RMSE, MASE, SMAPE.

Papka O., Oliinyk R. Comparative Analysis of the Implementation of Classical Statistical Models for Univariate Time Series Forecasting in Python, R, and Julia. This study presents a comparative analysis of the implementation of classical statistical models for univariate time series forecasting in the programming environments Python, R, and Julia. The research examines naïve and seasonal naïve methods, exponential smoothing (ETS), as well as the automatic parameter selection of autoregressive integrated moving average models (Auto-ARIMA). Two datasets were used for the experiments: daily Brent crude oil prices and monthly average wage indicators in Ukraine. Forecast accuracy was evaluated using three metrics — RMSE, MASE, and sMAPE — across different forecasting horizons. The results demonstrate that modeling outcomes depend not only on the mathematical nature of the algorithms but also on the specifics of their implementation in the respective ecosystems. It was found that Python and R provide full support for all the models under consideration, whereas Julia, at the time of the study, lacked stable implementations of ETS and Auto-ARIMA, which partially limited the possibilities for comparison. The findings are relevant both for assessing the maturity of software ecosystems in the field of time series analysis and for guiding the practical selection of tools in applied forecasting tasks.
Keywords: time series forecasting, Python, R, Julia, ETS, Auto-ARIMA, RMSE, MASE, sMAPE.

Постановка проблеми. Прогнозування часових рядів належить до ключових напрямів сучасної прикладної статистики та економетрики і широко застосовується у фінансовому аналізі, макроекономічному прогнозуванні, плануванні виробництва, управлінні запасами тощо. Часовий ряд є впорядкованою у часі послідовністю спостережень певного показника, і саме ця впорядкованість зумовлює потребу у використанні спеціалізованих методів, здатних враховувати залежності між значеннями у різні моменти часу. В сучасних умовах, коли обсяг даних постійно зростає, а горизонти прогнозування стають дедалі складнішими, результативність таких задач визначається не лише властивостями обраної математичної моделі, але й особливостями інструментального середовища, у якому ця модель реалізується.

Вибір програмного середовища впливає на всі етапи аналітичного процесу: від первинної обробки даних та формування навчальних і тестових вибірок до побудови прогнозу, його верифікації за узгодженими метриками та представлення результатів у графічній формі. На практиці це означає, що для ефективного застосування моделей необхідними є не лише адекватні алгоритми, але й розвинена інфраструктура, яка включає наявність стабільних бібліотек, зрозумілий синтаксис, підтримку інтеграції з іншими інструментами, достатню швидкодію та можливість відтворення експериментів на різних платформах.

Впродовж останніх років найбільш розвиненими інструментами для прогнозування часових рядів залишаються мови програмування R та Python, які сформували потужні екосистеми бібліотек для реалізації як класичних статистичних моделей, так і сучасних методів машинного навчання. Водночас дедалі більший інтерес наукової та інженерної спільноти привертає мова Julia,

що поєднує високу продуктивність компільованих мов із синтаксичною гнучкістю динамічних середовищ і позиціонується як перспективна платформа для наукових обчислень та моделювання.

Попри теоретичну еквівалентність багатьох моделей, їх програмна реалізація у різних мовах може суттєво відрізнятися через особливості чисельної оптимізації, методи оцінювання параметрів, стандартні налаштування та рівень стабільності бібліотек. У результаті, використання однакових даних та однакових моделей не гарантує повної тотожності прогнозів у різних середовищах. Важливим чинником також є зрілість екосистеми кожної мови: наявність чи відсутність готових функцій, зручність налаштування обчислювальних процедур, простота побудови робочих процесів, інтеграція з інструментами візуалізації та підтримка з боку спільноти розробників.

Таким чином, постає необхідність у комплексному порівнянні можливостей Python, R та Julia в задачах прогнозування одновимірних часових рядів. Важливим є не лише оцінити точність прогнозів за допомогою уніфікованих метрик на основі узгоджених даних, але й проаналізувати особливості реалізації моделей, зручність інфраструктури та технічні обмежень кожного середовища. Результати такого дослідження становлять інтерес як для теоретиків, що працюють над удосконаленням методів моделювання часових рядів, так і для практиків, які інтегрують інструменти прогнозування у бізнес-процеси.

Аналіз останніх досліджень і публікацій. Використання класичних статистичних методів для прогнозування часових рядів на сьогоднішній день залишається важливим напрямом досліджень. Незважаючи на активний розвиток сучасних методів машинного навчання, ARIMA, експоненційне згладжування та похідні моделі продовжують широко використовуватися як у фундаментальних, так і в прикладних дослідженнях.

У середовищі R ключовий внесок належить Р. Дж. Гайндману та Й. Хандакару (Hyndman R. J., Khandakar Y.), які описали автоматизовану ідентифікацію та оцінювання ARIMA (алгоритм Hyndman–Khandakar) і реалізували широку лінійку ETS-моделей у пакеті forecast [1]. Методичну основу та практичні рецепти використання цих підходів структуровано подано Р. Дж. Гайндманом та Г. Афанасопулосом (Hyndman R.J., Athanasopoulos G.) у підручнику «Forecasting: Principles and Practice». Дана робота забезпечує зв'язок між теорією, програмною реалізацією та правильною валідацією прогнозів [5].

У Python на даний момент найпоширенішою з подібних бібліотек є бібліотека statsmodels, архітектуру та принципи якої детально описали С. Сіболд і Й. Перктольд (Seabold S., Perktold J.). Вона охоплює ARMA/ARIMA та експоненційне згладжування, надаючи засоби оцінювання параметрів та діагностики [2]. Паралельно розвиваються фреймворки уніфікації інтерфейсів і процедур експериментів; зокрема, команда sktime запропонувала відтворюваний робочий процес для оцінювання (у т. ч. ймовірнісних) прогнозів, що спрощує порівняльні дослідження і забезпечує узгоджені протоколи оцінювання ефективності [6].

Екосистема Julia активно розвиває засоби класичного прогнозування на основі стан-просторових моделей. Р. Сааведра (Saavedra R.) зі співавторами описують пакет StateSpaceModels.jl - цілісну платформу для моделювання, оцінювання та прогнозування у просторі станів [3]. Паралельно з'являються спеціалізовані програмні реалізації для окремих класів процесів: Дж. Е.. Вера-Вальдес (Vera-Valdés J.E.) описує LongMemory.jl для моделей з довгою пам'яттю з підтримкою прогнозування [14], а М. Штаппер (Stapper M.) демонструє CountTimeSeries.jl для цілочисельних рядових моделей та їхнього прогнозування [15].

Щодо емпіричних практик, відкриті приклади сучасного застосування ARIMA підтверджують релевантність використання класичних моделей у розв'язанні суспільно-економічних задач. Так, К. Перейра (Pereira da Veiga C.) зі співавторами демонструють повний цикл побудови та інтерпретації ARIMA в аналізі макроекономічних індикаторів у сфері охорони здоров'я [7], а Р. Оспіна (Ospina R.) зі співавторами подають огляд методології ARIMA під час пандемії COVID-19 разом із аналізом конкретних випадків та рекомендаціями щодо практичного оцінювання [8].

Вагомим джерелом стандартів емпіричної перевірки служать результати М-конкурсів. Праця С. Макридакіса, Е. Спіліотіса та В. Ассімакопулоса (Makridakis S., Spiliotis E., Assimakopoulos V.) з описом M4 (100 тис. рядів і 61 метод) зафіксувала конкурентоспроможність простих статистичних методів і комбінацій [9], а підсумкова праця авторів окреслила висновки

щодо ролі комбінацій і прогнозних інтервалів у практичних застосуваннях [10]. Ці джерела задають коректні стандарти порівняння реалізацій ARIMA/ETS у різних мовах програмування.

Вітчизняні науковці також приділяють увагу застосуванню ARMA/ARIMA та пов'язаних з ними методів. Зокрема, Д. Круковець і О. Верченко показали ефективність комбінованого ARMA-підходу для короткострокового прогнозування базової інфляції в Україні [11]; Ю. Андрусенко та І. Лановенко систематизували основні моделі прогнозування часових рядів у прикладних задачах [12]; А. Яровий і В. Сєдін продемонстрували використання ARIMA для аналізу динаміки злочинності [13].

Також у наукових дослідженнях особливого значення набуває поняття відтворюваності. Використання контейнеризації дозволяє стандартизувати середовища виконання та забезпечити прозорість комп'ютерних експериментів. К. Беттігер (Boettiger C.) у своїй роботі окреслив роль Docker у відтворюваних дослідженнях [16], К. Беттігер і Д. Еддельбуттель (Boettiger C., Eddelbuettel D.) представили Rocker - контейнеризовані середовища R [17], а Д. Нюст (Nüst D.) зі співавторами формулювали «десять простих правил» для написання відтворюваних Dockerfile [18].

Метою роботи є здійснення порівняльного аналізу реалізації статистичних методів прогнозування одновимірних часових рядів у середовищах Python, R та Julia з використанням однакових наборів даних, уніфікованих процедур попередньої обробки, узгоджених горизонтів прогнозу та єдиних метрик оцінювання точності. Завдання дослідження полягає не лише у кількісному порівнянні якості прогнозів, але й у виявленні особливостей, пов'язаних із програмною реалізацією моделей у різних мовах, включаючи доступність бібліотек, стабільність їх функціонування та зручність організації відтворюваних аналітичних процесів. Особливу увагу приділено аналізу сильних і слабких сторін кожного середовища у практичному застосуванні.

Виклад основного матеріалу дослідження. Проведення дослідження передбачало створення уніфікованого програмного середовища, яке забезпечує відтворюваність результатів та мінімізацію впливу відмінностей між платформами. У роботі застосовано підхід контейнеризації на основі технології Docker, який дав змогу ізолювати програмне оточення для кожної з мов програмування та забезпечити ідентичність виконання на різних апаратних конфігураціях. Структура проєкту була організована з урахуванням чіткого розподілу функцій між модулями, що містять вхідні дані, програмний код, результати обчислень та допоміжні конфігураційні файли.

Для Python використовувався базовий образ python:3.12-slim із додатково встановленими бібліотеками для статистичного моделювання та візуалізації, включаючи statsmodels, pmdarima, pandas та matplotlib. Середовище R було побудоване на основі офіційного образу rocker/verse:4.4.0, який містить широкий набір статистичних пакетів, у тому числі пакет forecast, що реалізує функції ets() та auto.arima(). Для Julia застосовувався образ julia:1.11 із попередньо встановленими бібліотеками CSV, DataFrames, ARFIMA та StateSpaceModels.jl.

Структура середовища була організована так, щоб вхідні дані, програмний код, результати обчислень і візуалізація даних зберігалися в окремих каталогах, що спрощувало управління експериментом. Запуск розрахунків відбувався за допомогою уніфікованих команд Docker, які забезпечували однакові умови для кожного середовища. Таке рішення дозволило організувати єдиний дослідницький процес, у межах якого Python, R та Julia виконувалися незалежно, але за однаковими правилами, щоб будь-які відмінності у бібліотеках чи налаштуваннях не могли спотворити результати порівняння.

У дослідженні використовувалися два набори даних. Перший - часовий ряд щоденних котирувань нафти Brent за період 2018–2025 років, другий - щомісячні показники середньої заробітної плати в Україні з 2003 по 2025 рік. Обидва набори містили два стовпці: дату спостереження та числове значення. Для часового ряду Brent характерною є відсутність поодиноких значень, які усувалися шляхом побудови повного календарного індексу та видалення відсутніх значень, що, в свою чергу, дозволило зберегти рівномірність часових відрізків. Наприклад, у Python це було реалізовано наступним чином:

```
full_idx = pd.date_range(df["date"].min(), df["date"].max(), freq="D")
df = df.set_index("date").reindex(full_idx).rename_axis("date").reset_index()
df = df.dropna()
```

Для набору даних показників середньої заробітної плати було обрано горизонт прогнозування 12 місяців, який відповідає одному календарному року та дозволяє оцінити моделі в умовах повного сезонного циклу. Для набору даних Brent горизонт прогнозування становив 30 днів, що є прийнятним для тестування моделей у задачах короткострокового прогнозування фінансових часових рядів із потенційно високою волатильністю.

Набір моделей включав як прості базові підходи (наївний та сезонно-наївний), так і складні методи — експоненційне згладжування ETS та автоматичний підбір параметрів ARIMA (Auto-ARIMA).У Python моделі реалізовувалися за допомогою бібліотек statsmodels і pmdarima, у R — засобами пакета forecast. У Julia передбачалося використання пакетів StateSpaceModels.jl та ARFIMA.jl для відтворення ETS та ARIMA-моделей, поряд із реалізацією наївних підходів. Таким чином, дослідження було організовано так, щоб усі три середовища мали можливість продемонструвати як базові, так і більш гнучкі методи прогнозування, а їх результати могли бути співставлені за єдиними метриками.

Для оцінювання точності прогнозів було застосовано три узгоджені метрики, що відображають різні аспекти похибки. Корінь середньоквадратичної помилки (RMSE) дозволяє оцінити абсолютні відхилення прогнозу від фактичних значень у тих самих одиницях, що й вихідний показник. Середня абсолютна масштабована помилка (MASE) є метрикою, яка забезпечує можливість порівняння якості прогнозів між різними рядами, оскільки нормується відносно простої наївної моделі. Симетрична середня абсолютна відсоткова помилка (sMAPE) застосовується для вимірювання відсоткових похибок і враховує симетричність оцінювання, що робить її стійкою до варіацій масштабу показника. Для забезпечення відтворюваності всі формули реалізовувалися безпосередньо у скриптах кожної мови. Наприклад, у Julia обчислення sMAPE виконувалося так:

```
function smape(y_true, y_pred)
    return mean(2 .* abs.(y_true .- y_pred) ./ (abs.(y_true) .+ abs.(y_pred))) * 100
end
```

Результати обчислень зберігалися у вигляді CSV-файлів, в яких містилась інформація про назву моделі, значення метрик, довжину прогнозного горизонту та мітку часу. Такий підхід спрощував подальший аналіз та уможливлював автоматизоване завантаження результатів для візуалізації. Окремі скрипти у кожному середовищі відповідали за побудову графіків, які зберігалися у форматі PNG у підкаталогах plots. Таким чином, результати дослідження можна було оцінювати не лише за числовими показниками, але й за формою прогнозних траєкторій.

Важливим аспектом організації експериментів була уніфікація обробки даних. Зокрема, як вже було зазначено раніше, для ряду Brent застосовувалася побудова повного календарного індексу, яка гарантувала однакову часову шкалу в усіх середовищах. Для заробітної плати, де дані мали чітку місячну періодичність, таких перетворень не потребувалося, однак у всіх випадках значення показників було приведено до числового формату з плаваючою точкою. Це виключило можливі розбіжності унаслідок різних способів парсингу дат чи округлення чисел у Python, R та Julia.

Для обох часових рядів були отримані прогнози за всіма розглянутими методами. Результати оцінювання точності прогнозів наведено у таблицях 1 і 2.

Таблиця 1. Метрики RMSE, MASE та sMAPE для прогнозування ціни на нафту марки Brent (горизонт 30 днів)

Модель	RMSE	MASE	sMAPE
Python			
Naive	8.4351	6.0710	10.8451
Seasonal Naive	4.8940	3.0797	5.5827
ETS	8.5080	6.1213	10.9289
Auto-ARIMA	8.4555	6.0832	10.8654
R			
Naive	8.4351	6.0710	10.8451
Seasonal Naive	7.6113	5.4940	9.8718
ETS	8.4351	6.0710	10.8451
Auto-ARIMA	8.4555	6.0832	10.8655
Julia			

Naive	8.4351	6.0710	10.8451
Seasonal Naive	7.6113	5.4940	9.8718

Таблиця 2. Метрики RMSE, MASE та sMAPE для прогнозування значення середньої заробітної плати (горизонт 12 місяців)

Модель	RMSE	MASE	sMAPE
Python			
Naive	3188.2470	9.2877	16.5860
Seasonal Naive	3453.8442	11.1511	20.5709
ETS	1933.8230	5.8243	10.0910
Auto-ARIMA	1038.0210	3.0382	5.1987
R			
Naive	3188.2470	9.2877	16.5860
Seasonal Naive	3453.8442	11.1511	20.5709
ETS	1925.5644	5.8008	10.0484
Auto-ARIMA	2974.6375	9.3110	16.7757
Julia			
Naive	3188.2470	9.2877	16.5860
Seasonal Naive	3453.8442	11.1511	20.5709

Результати дослідження вказують на низку важливих закономірностей. Для часового ряду Brent, що характеризується коротким горизонтом прогнозування та вираженою повторюваністю коливань, найнижчі значення похибок забезпечив сезонний наївний метод. Зокрема, у середовищі Python значення RMSE становило 4.8940, а симетричної середньої абсолютної відносної похибки — 5.5827, що було істотно нижчим порівняно з іншими моделями. Подібна ситуація спостерігається і у R та Julia, де сезонно-наївні прогнози відтворювали характерні тижневі коливання з мінімальними відхиленнями від фактичних значень. Це свідчить про те, що для короткострокових задач з регулярною циклічністю застосування простих методів може бути не менш ефективним, ніж використання більш складних статистичних моделей.

Для ряду середньої заробітної плати ситуація є дещо іншою - дані мають чітку річну сезонність та виражений зростаючий тренд, тому базові підходи не забезпечили задовільної точності. Найкращі результати у Python продемонструвала модель Auto-ARIMA, для якої значення RMSE становило 1038.0210, а sMAPE — 5.1987. Це істотно краще порівняно як з наївними методами, так і з експоненційним згладжуванням. У R результати Auto-ARIMA були менш точними: RMSE дорівнювало 2974.6375, а sMAPE — 16.7757, тоді як модель ETS у цьому середовищі продемонструвала нижчі значення похибок. Це підтверджує думку про те, що навіть за однакових математичних формулювань результати можуть значно відрізнятися через специфіку реалізації алгоритмів у різних програмних екосистемах.

Для обидвох рядів наївні підходи у Python, R та Julia були успішно реалізовані і дали ідентичні результати, однак при спробах відтворити моделі ETS та Auto-ARIMA на Julia виникли суттєві труднощі. В процесі дослідження цьому питанню було приділено значну увагу, оскільки мова програмування Julia останніми роками позиціонується як високопродуктивне середовище для наукових обчислень.

Пакет StateSpaceModels.jl**,** який задекларовано як базове рішення для експоненційного згладжування та ARIMA-процедур, на момент дослідження мав обмеження. Для моделей ETS підтримувалася лише адитивна специфікація, без можливості побудови мультиплікативних сезонних компонентів, що звужувало діапазон практичного застосування. Реалізація функцій auto_ets та auto_arima виявилася чутливою до версій інтерпретатора Julia та залежностей, що призводило до нестабільності результатів і несумісності з відтворюваним робочим потоком на основі Docker. У кількох випадках під час запуску автоматичного підбору параметрів виникали помилки оптимізації або повернення некоректних об'єктів прогнозу, що унеможливлювало інтеграцію цих моделей у загальну систему.

Застосування альтернативних бібліотек також не дало позитивного результату. Пакет ARFIMA.jl орієнтований насамперед на симуляцію процесів типу ARFIMA та не забезпечує зручних процедур для автоматизованого прогнозування. Інші експериментальні розробки, зокрема

порти функціональності з мови R, не мають актуальної підтримки й несумісні з сучасними версіями Julia.

В результаті для Julia ми обмежилися реалізацією лише наївних підходів. Попри свою простоту, ці методи продемонстрували результати, що збігалися з аналогічними обчисленнями у Python та R. Водночас відсутність можливості застосувати повний спектр класичних моделей у Julia стала важливим обмеженням цього дослідження і засвідчила те, що на сьогодні ця мова програмування ще не може розглядатися як повноцінна альтернатива для прогнозування часових рядів.

Важливою складовою роботи було дослідження інструментів візуалізації результатів прогнозування. Графічне представлення прогнозів дозволяє наочно оцінити відповідність моделей динаміці фактичних даних та відмінності між середовищами. На рисунках 1–3 подано візуалізацію прогнозів для часового ряду показників середньої заробітної плати, отримані в Python, R та Julia відповідно.

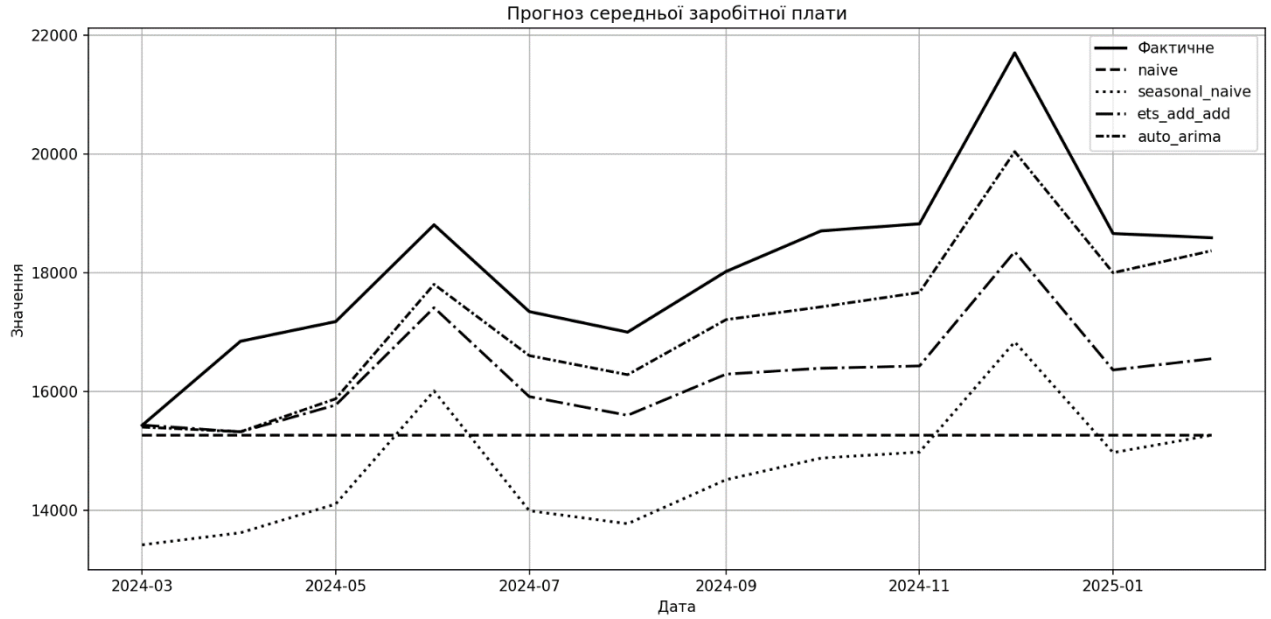

Рис. 1. Прогноз показників середньої заробітної плати в Україні в Python

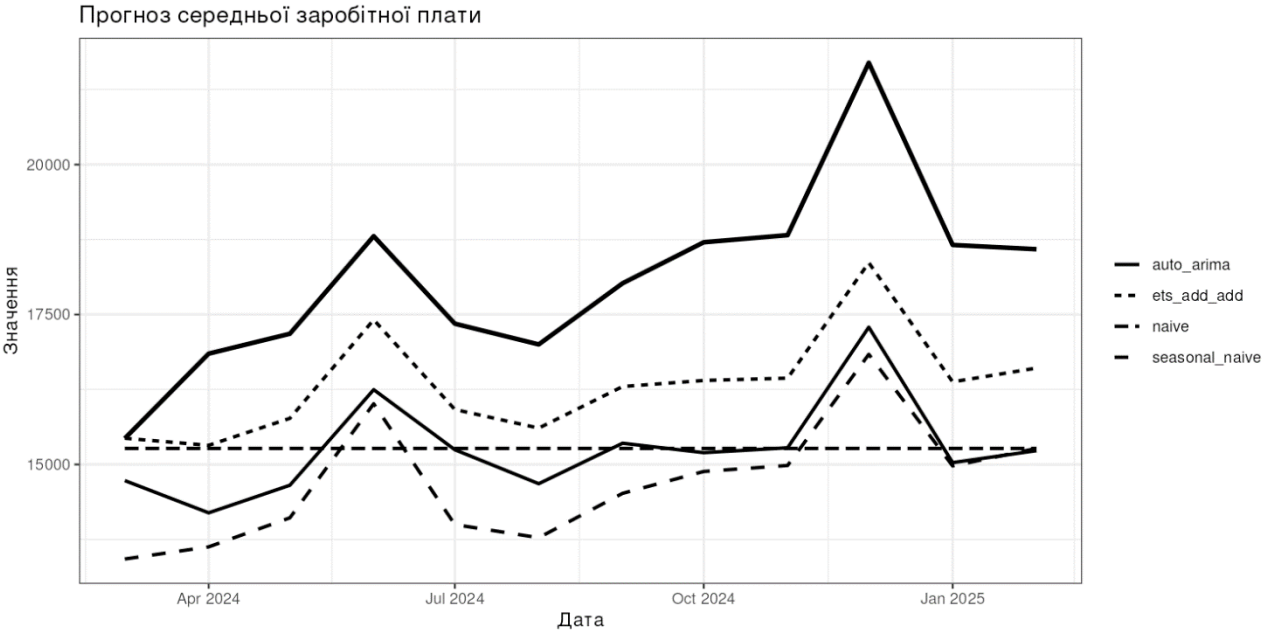

Рис. 2. Прогноз показників середньої заробітної плати в Україні в R

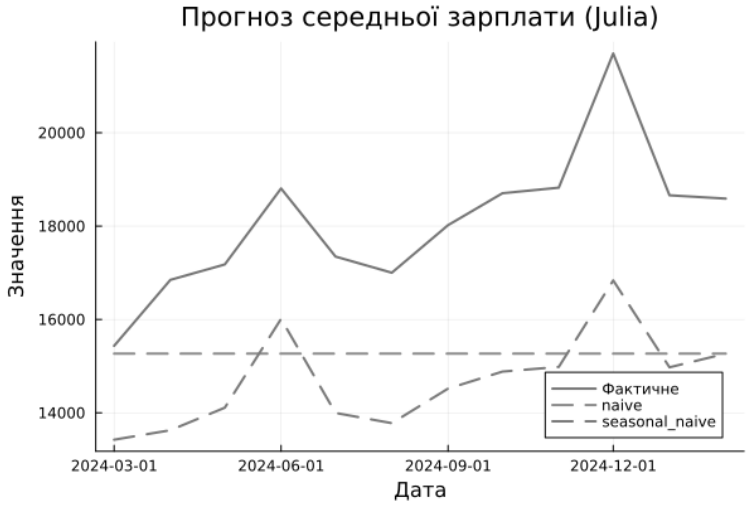

Рис. 3. Прогноз показників середньої заробітної плати в Україні в Julia

У Python побудова графіків здійснювалася за допомогою бібліотеки matplotlib, яка є стандартом де-факто для візуалізації у наукових дослідженнях. Вона забезпечує гнучке налаштування параметрів відображення, включаючи зміну масштабів осей, форматування дат і зручне поєднання фактичних та прогнозних кривих. У середовищі R аналогічні задачі виконувалися засобами пакета ggplot2, що входить до складу tidyverse і пропонує декларативний підхід до побудови графіків. Його перевагою є висока якість графічного відтворення та зручність створення складних візуалізацій на основі простих правил комбінування геометричних об'єктів. У Julia для побудови графіків використовувався пакет Plots.jl, який забезпечує базову функціональність і підтримує декілька бекендів візуалізації. Хоча можливості Plots.jl поступаються більш зрілим екосистемам Python та R у частині гнучкості та розвиненості додаткових функцій, його використання дозволило створити порівнянні за структурою графіки для цілей даного дослідження.

Процес виконання дослідження супроводжувався систематичними зверненнями до офіційної документації, прикладів коду та обговорень у спільнотах. Варто зазначити, що екосистема Python вирізняється зрілою інфраструктурою: структурованими довідниками, регулярними мінорними оновленнями бібліотек для часових рядів та наявністю типових сценаріїв, які забезпечують швидке розв'язання практичних завдань. Підтримка здійснюється як через офіційну документацію та трекери GitHub, так і через велику мережу обговорень (дискусійні майданчики, Q&A), що значно полегшує вирішення нетипових кейсів. У R виразно простежується усталена статистична традиція: пакети для прогнозування мають продумані налаштування за замовчуванням, детальну документацію та публікуються з дотриманням регламентованих процедур на CRAN. Екосистема Julia демонструє інтенсивний темп розвитку: оперативні мажорні та мінорні оновлення супроводжуються активною взаємодією розробників у GitHub/Discourse. Загалом можна констатувати, що рівень супроводу кожної з екосистем знаходиться на високому рівні.

Висновки та перспективи подальших досліджень. У ході дослідження було здійснено порівняння реалізації моделей прогнозування часових рядів у трьох середовищах програмування — Python, R та Julia. Результати показали, що навіть за використання однакових вихідних даних та узгоджених методик моделювання точність прогнозів і стабільність результатів суттєво залежать від зрілості екосистеми та реалізації алгоритмів у відповідному середовищі.

У середовищі Python усі розглянуті моделі були реалізовані без обмежень. Було підтверджено, що Python забезпечує коректні і стабільні результати як для наївних підходів, так і для ETS та Auto-ARIMA, при цьому саме автоматичний підбір параметрів ARIMA продемонстрував найкращі результати у задачі довгострокового прогнозування заробітних плат. Водночас для коротких прогнозних горизонтів на фінансових рядах складні моделі не перевершували простіші сезонні наївні підходи.

Середовище R також забезпечило повну реалізацію методів, що розглядалися. У порівнянні з Python воно показало близькі результати для наївних моделей і ETS, проте метод Auto-ARIMA виявився менш ефективним на даних із вираженим трендом і сезонністю. Це підтверджує, що навіть за формально однакових методів точність прогнозів може залежати від особливостей реалізації у різних програмних екосистемах.

Julia, попри високий потенціал і продуктивність, на момент дослідження виявила суттєві обмеження в реалізації обраних методів. У цьому середовищі вдалося реалізувати лише наївні та сезонні наївні підходи, які показали однакові результати з Python і R. Через нестабільність доступних реалізацій ETS та Auto-ARIMA ці методи не були включені до експериментальної частини. Це свідчить, що Julia поки що поступається Python і R у зрілості екосистеми для прогнозування часових рядів.

Загалом проведене дослідження підтвердило, що вибір середовища програмування є не менш важливим, ніж вибір самої моделі. Python та R на сьогодні можна вважати найбільш зрілими та універсальними інструментами у сфері статистичного прогнозування часових рядів, тоді як Julia перебуває на етапі становлення й потребує подальшого вдосконалення бібліотек. Подальші дослідження доцільно зосередити на розширенні спектра моделей, включенні сучасних методів машинного навчання та розвитку інструментарію Julia.

Список бібліографічного опису:

1. Hyndman R. J., Khandakar Y. Automatic Time Series Forecasting: The forecast package for R. Journal of Statistical Software. 2008. Vol. 27, No. 3. P. 1–22. URL: https://www.jstatsoft.org/v27/i03 (дата звернення: 15.08.2025)
2. Seabold S., Perktold J. Statsmodels: Econometric and Statistical Modeling with Python. Proceedings of the 9th Python in Science Conference (SciPy 2010). 2010. P. 57–61. URL: https://proceedings.scipy.org/articles/Majora-92bf1922-011.pdf (дата звернення: 16.08.2025)
3. Saavedra R., Bodin G., Souto M. StateSpaceModels.jl: a Julia Package for Time-Series Analysis in a State-Space Framework. arXiv preprint. 2019. URL: https://arxiv.org/pdf/1908.01757 (дата звернення: 15.08.2025)
4. Hyndman R. J., Koehler A. B. Another look at measures of forecast accuracy. International Journal of Forecasting. 2006. Vol. 22, No. 4. P. 679–688. URL: https://robjhyndman.com/papers/mase.pdf (дата звернення: 16.08.2025)
5. Hyndman R. J., Athanasopoulos G. Forecasting: Principles and Practice (3rd ed.). Melbourne: OTexts, 2021. URL: https://otexts.com/fpp3/ (дата звернення: 19.08.2025)
6. Heidrich B., Stephan A., et al. Evaluating Probabilistic Forecasters with sktime and evalprob4cast. Proceedings of the 23rd Python in Science Conference (SciPy 2024). 2024. URL: https://proceedings.scipy.org/articles/VPNX1595 (дата звернення: 17.08.2025)
7. Pereira da Veiga C., Pereira da Veiga C. R., Girotto F. M., Marconatto D. A. B., Su Z. Implementation of the ARIMA model for prediction of economic variables: evidence from the health sector in Brazil. Humanities and Social Sciences Communications. 2024. 11: Article 30 23. URL: https://www.nature.com/articles/s41599-024-03023-3 (дата звернення: 15.08.2025)
8. Ospina R., Gondim J. A. M., Leiva V., Castro C. An Overview of Forecast Analysis with ARIMA Models during the COVID-19 Pandemic: Methodology and Case Study in Brazil. Mathematics. 2023. Vol. 11, No. 14: 3069. URL: https://www.mdpi.com/2227-7390/11/14/3069 (дата звернення: 16.08.2025)
9. Makridakis S., Spiliotis E., Assimakopoulos V. The M4 Competition: 100,000 time series and 61 forecasting methods. International Journal of Forecasting. 2020. Vol. 36, No. 1. P. 54–74. URL: https://www.researchgate.net/publication/334578434_The_M4_Competition_100000_time_series_and_61_forecasting_methods (дата звернення: 15.08.2025)
10. Makridakis S., Spiliotis E., Assimakopoulos V. The M4 Competition: Results, findings, conclusion and way forward. International Journal of Forecasting. 2018. Vol. 34, No. 4. P. 802–808. URL: https://www.researchgate.net/profile/Spyros_Makridakis/publication/325901666_The_M4_Competition_Results_findings_conclusion_and_way_forward/links/5b2c9aa4aca2720785d66b5e/The-M4-Competition-Results-findings-conclusion-and-way-forward.pdf (дата звернення: 15.08.2025)
11. Krukovets D., Verchenko O. Short-Run Forecasting of Core Inflation in Ukraine: A Combined ARMA Approach. Visnyk of the National Bank of Ukraine. 2019. No. 248. P. 11–20. URL: https://journal.bank.gov.ua/uploads/articles/248_2_Krukovets_Verchenko.pdf (дата звернення: 17.08.2025)
12. Андрусенко Ю. О., Лановенко І. М. Аналіз основних моделей прогнозування часових рядів. Збірник наукових праць Харківського національного університету Повітряних Сил. 2020. № 2(64). С. 62–68. URL: https://journal-hnups.com.ua/index.php/concern/article/view/358/292 (дата звернення: 19.08.2025)
13. Яровий А. А., Сєдін В. М. Використання ARIMA-моделей для прогнозування рівня злочинності. Український журнал інформаційних технологій (Lviv Polytechnic). 2024. Т. 6, № 4. С. 649–656. URL: https://science.lpnu.ua/ujit/all-volumes-and-issues/volume-6-number-4-2024/vykorystannia-arima-modelei-dlia-prohnozuvannia (дата звернення: 17.08.2025)
14. Vera-Valdés J. E. LongMemory.jl: Generating, Estimating, and Forecasting Long Memory Models in Julia. Journal of Open Source Software. 2025. Vol. 10, No. 108: 7708. URL: https://www.theoj.org/joss-papers/joss.07708/10.21105.joss.07708.pdf (дата звернення: 16.08.2025)

15. Stapper M. Count Data Time Series Modelling in Julia — The CountTimeSeries.jl Package and Applications. Entropy. 2021. Vol. 23, No. 6: 666. URL: https://www.mdpi.com/1099-4300/23/6/666 (дата звернення: 16.08.2025)
16. Boettiger C. An introduction to Docker for reproducible research, with examples from the R environment. ACM SIGOPS Operating Systems Review. 2015. Vol. 49, No. 1. P. 71–79. URL: https://arxiv.org/abs/1410.0846 (дата звернення: 19.08.2025)
17. Boettiger C., Eddelbuettel D. An Introduction to Rocker: Docker Containers for R. The R Journal. 2017. Vol. 9, No. 2. P. 527–536. URL: https://journal.r-project.org/archive/2017/RJ-2017-065/index.html (дата звернення: 17.08.2025)
18. Nüst D., Sochat V., Marwick B., Eglen S. J., Head T., Hirst T., Evans B. D. Ten simple rules for writing Dockerfiles for reproducible data science. PLOS Computational Biology. 2020. Vol. 16, No. 11: e1008316. URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC7654784/ (дата звернення: 19.08.2025)

References:

1. Hyndman R. J., Khandakar Y. Automatic Time Series Forecasting: The forecast package for R. Journal of Statistical Software. 2008. Vol. 27, No. 3. P. 1–22. URL: https://www.jstatsoft.org/v27/i03 (access date: 15.08.2025)
2. Seabold S., Perktold J. Statsmodels: Econometric and Statistical Modeling with Python. Proceedings of the 9th Python in Science Conference (SciPy 2010). 2010. P. 57–61. URL: https://proceedings.scipy.org/articles/Majora-92bf1922-011.pdf (access date: 16.08.2025)
3. Saavedra R., Bodin G., Souto M. StateSpaceModels.jl: a Julia Package for Time-Series Analysis in a State-Space Framework. arXiv preprint. 2019. URL: https://arxiv.org/pdf/1908.01757 (access date: 15.08.2025)
4. Hyndman R. J., Koehler A. B. Another look at measures of forecast accuracy. International Journal of Forecasting. 2006. Vol. 22, No. 4. P. 679–688. URL: https://robjhyndman.com/papers/mase.pdf (access date: 16.08.2025)
5. Hyndman R. J., Athanasopoulos G. Forecasting: Principles and Practice (3rd ed.). Melbourne: OTexts, 2021. URL: https://otexts.com/fpp3/ (access date: 19.08.2025)
6. Heidrich B., Stephan A., et al. Evaluating Probabilistic Forecasters with sktime and evalprob4cast. Proceedings of the 23rd Python in Science Conference (SciPy 2024). 2024. URL: https://proceedings.scipy.org/articles/VPNX1595 (access date: 17.08.2025)
7. Pereira da Veiga C., Pereira da Veiga C. R., Girotto F. M., Marconatto D. A. B., Su Z. Implementation of the ARIMA model for prediction of economic variables: evidence from the health sector in Brazil. Humanities and Social Sciences Communications. 2024. 11: Article 30 23. URL: https://www.nature.com/articles/s41599-024-03023-3 (access date: 15.08.2025)
8. Ospina R., Gondim J. A. M., Leiva V., Castro C. An Overview of Forecast Analysis with ARIMA Models during the COVID-19 Pandemic: Methodology and Case Study in Brazil. Mathematics. 2023. Vol. 11, No. 14: 3069. URL: https://www.mdpi.com/2227-7390/11/14/3069 (access date: 16.08.2025)
9. Makridakis S., Spiliotis E., Assimakopoulos V. The M4 Competition: 100,000 time series and 61 forecasting methods. International Journal of Forecasting. 2020. Vol. 36, No. 1. P. 54–74. URL: https://www.researchgate.net/publication/334578434_The_M4_Competition_100000_time_series_and_61_forecasting_methods (access date: 15.08.2025)
10. Makridakis S., Spiliotis E., Assimakopoulos V. The M4 Competition: Results, findings, conclusion and way forward. International Journal of Forecasting. 2018. Vol. 34, No. 4. P. 802–808. URL: https://www.researchgate.net/profile/Spyros_Makridakis/publication/325901666_The_M4_Competition_Results_findings_conclusion_and_way_forward/links/5b2c9aa4aca2720785d66b5e/The-M4-Competition-Results-findings-conclusion-and-way-forward.pdf (access date: 15.08.2025)
11. Krukovets D., Verchenko O. Short-Run Forecasting of Core Inflation in Ukraine: A Combined ARMA Approach. Visnyk of the National Bank of Ukraine. 2019. No. 248. P. 11–20. URL: https://journal.bank.gov.ua/uploads/articles/248_2_Krukovets_Verchenko.pdf (access date: 17.08.2025)
12. Andrusenko Yu. O., Lanovenko I. M. Analysis of the main models of time series forecasting. Collection of Scientific Works of Kharkiv National Air Force University. 2020. No. 2(64). P. 62–68. URL: https://journal-hnups.com.ua/index.php/concern/article/view/358/292 (access date: 19.08.2025).
13. Yarovyi A. A., Siedin V. M. Application of ARIMA models for crime rate forecasting. Ukrainian Journal of Information Technology (Lviv Polytechnic). 2024. Vol. 6, No. 4. P. 649–656. URL: https://science.lpnu.ua/ujit/all-volumes-and-issues/volume-6-number-4-2024/vykorystannia-arima-modelei-dlia-prohnozuvannia (access date: 17.08.2025).
14. Vera-Valdés J. E. LongMemory.jl: Generating, Estimating, and Forecasting Long Memory Models in Julia. Journal of Open Source Software. 2025. Vol. 10, No. 108: 7708. URL: https://www.theoj.org/joss-papers/joss.07708/10.21105.joss.07708.pdf (access date: 16.08.2025)
15. Stapper M. Count Data Time Series Modelling in Julia — The CountTimeSeries.jl Package and Applications. Entropy. 2021. Vol. 23, No. 6: 666. URL: https://www.mdpi.com/1099-4300/23/6/666 (access date: 16.08.2025)
16. Boettiger C. An introduction to Docker for reproducible research, with examples from the R environment. ACM SIGOPS Operating Systems Review. 2015. Vol. 49, No. 1. P. 71–79. URL: https://arxiv.org/abs/1410.0846 (access date: 19.08.2025)
17. Boettiger C., Eddelbuettel D. An Introduction to Rocker: Docker Containers for R. The R Journal. 2017. Vol. 9, No. 2. P. 527–536. URL: https://journal.r-project.org/archive/2017/RJ-2017-065/index.html (access date: 17.08.2025)
18. Nüst D., Sochat V., Marwick B., Eglen S. J., Head T., Hirst T., Evans B. D. Ten simple rules for writing Dockerfiles for reproducible data science. PLOS Computational Biology. 2020. Vol. 16, No. 11: e1008316. URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC7654784/ (access date: 19.08.2025)