

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
«ДНІПРОВСЬКА ПОЛІТЕХНІКА»



ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
Кафедра програмного забезпечення комп'ютерних систем
Кафедра інформаційних технологій та комп'ютерної інженерії

**МОДЕЛЮВАННЯ ТА ОПТИМІЗАЦІЯ КОМП'ЮТЕРНИХ І
КІБЕРФІЗИЧНИХ СИСТЕМ І МЕРЕЖ**

Методичні рекомендації до виконання практичних робіт
для здобувачів ступеня доктора філософії
спеціальності F7 Комп'ютерна інженерія

Дніпро
НТУ «ДП»
2025

Моделювання та оптимізація комп'ютерних і кіберфізичних систем і мереж [Електронний ресурс]: методичні рекомендації до виконання практичних робіт для здобувачів ступеня доктора філософії спеціальності F7 Комп'ютерна інженерія / уклад.: І.С. Лактіонов, В.Ю. Каштан, В.В. Гнатушенко, В.І. Олевський ; М-во освіти і науки України, Нац. техн. ун-т «Дніпровська політехніка». – Дніпро : НТУ «ДП», 2025. – 29 с.

Укладачі:

І.С. Лактіонов, д-р техн. наук, проф.,

В.Ю. Каштан В.Ю., канд. техн. наук, доц.,

В.В. Гнатушенко , д-р техн. наук, проф.,

В.І. Олевський, д-р техн. наук, проф.

Затверджено науково-методичною комісією спеціальності F7 Комп'ютерна інженерія (протокол № 5 від 19.05.2025) за поданням кафедри інформаційних технологій та комп'ютерної інженерії (протокол № 13 від 11.04.2025).

Уміщено загальні теоретичні відомості щодо сучасних методів математичного, імітаційного та комп'ютерного моделювання, багатокритеріальної оптимізації та побудови цифрових двійників; опис функціональних можливостей і особливостей використання сучасних бібліотек Python для наукових досліджень; методики та типові приклади виконання практичних робіт; рекомендації щодо порядку виконання завдань і оформлення звітів; контрольні питання та перелік рекомендованих літературних джерел.

Відповідальний за випуск завідувач кафедри інформаційних технологій та комп'ютерної інженерії В.В. Гнатушенко, д-р техн. наук, проф.

ЗМІСТ

ВСТУП.....	4
ПРАКТИЧНА РОБОТА № 1. ДОСЛІДЖЕННЯ МЕТОДІВ ПОБУДОВИ ТА АНАЛІЗУ МАТЕМАТИЧНИХ МОДЕЛЕЙ СКЛАДНИХ КОМП'ЮТЕРНИХ І КІБЕРФІЗИЧНИХ СИСТЕМ ЗАСОБАМИ PYTHON.....	6
ПРАКТИЧНА РОБОТА № 2. ДОСЛІДЖЕННЯ МЕТОДІВ ІМІТАЦІЙНОГО МОДЕЛЮВАННЯ КОМП'ЮТЕРНИХ І КІБЕРФІЗИЧНИХ СИСТЕМ.....	11
ПРАКТИЧНА РОБОТА № 3. ДОСЛІДЖЕННЯ ПІДХОДІВ ДО БАГАТОКРИТЕРІАЛЬНОЇ ОПТИМІЗАЦІЇ МОДЕЛЕЙ СКЛАДНИХ КОМП'ЮТЕРНИХ І КІБЕРФІЗИЧНИХ СИСТЕМ.....	16
ПРАКТИЧНА РОБОТА № 4. ДОСЛІДЖЕННЯ МЕТОДІВ ПОБУДОВИ І ОПТИМІЗАЦІЇ ЦИФРОВИХ ДВІЙНИКІВ ЗАСОБАМИ КОМП'ЮТЕРНОГО ЕКСПЕРИМЕНТУ.....	21
КРИТЕРІЇ ОЦІНЮВАННЯ ПРАКТИЧНИХ РОБІТ.....	27

ВСТУП

Практичні роботи з дисципліни «Моделювання та оптимізація комп'ютерних і кіберфізичних систем і мереж» є обов'язковою складовою підготовки здобувачів третього (освітньо-наукового) рівня вищої освіти за спеціальністю F7 «Комп'ютерна інженерія». Їх виконання спрямоване на закріплення теоретичних знань, отриманих під час лекційних занять, та формування практичних компетентностей із застосування сучасних методів математичного й комп'ютерного моделювання, оптимізації та проведення наукових комп'ютерних експериментів.

Стрімкий розвиток комп'ютерних і кіберфізичних систем, цифрових двійників, технологій штучного інтелекту та інтелектуального аналізу даних зумовлює необхідність широкого використання моделей як основного інструменту дослідження складних технічних об'єктів. Сучасні методи моделювання дозволяють досліджувати поведінку систем у різних режимах функціонування, прогнозувати їхні характеристики, оптимізувати параметри та оцінювати ефективність прийнятих технічних рішень без проведення великої кількості дороговартісних натурних експериментів.

Особливістю цієї дисципліни є її дослідницька спрямованість. Практичні роботи орієнтовані не лише на освоєння сучасних комп'ютеризованих засобів моделювання, а й на формування навичок планування та проведення комп'ютерних експериментів, аналізу результатів моделювання, оцінювання адекватності моделей, застосування сучасних методів оптимізації та використання технологій штучного інтелекту під час розв'язання актуальних задач комп'ютерної інженерії.

Усі практичні роботи виконуються мовою програмування Python із використанням сучасних бібліотек наукових обчислень і аналізу даних, зокрема NumPy, SciPy, SymPy, Pandas, Matplotlib, SimPy, NetworkX, Scikit-learn, Optuna та DEAP. Практична робота № 4 додатково передбачає застосування середовища Node-red для створення і тестування цифрового двійника. Такий підхід

забезпечує використання відкритого програмного забезпечення, що широко застосовується у світовій науковій спільноті для моделювання, аналізу та оптимізації складних комп'ютерних і кіберфізичних систем.

Практичні роботи побудовано за принципом поступового ускладнення дослідницьких завдань. Починаючи з побудови математичних моделей та їх аналізу, здобувачі переходять до імітаційного моделювання, застосування сучасних методів оптимізації, а завершальним етапом є проведення повноцінного комп'ютерного експерименту, статистичний аналіз його результатів і розроблення спрощеного цифрового двійника досліджуваної системи.

Навчально-методичне забезпечення практикуму розроблено з урахуванням сучасних міжнародних тенденцій розвитку методів математичного моделювання, цифрових двійників та інтелектуальної оптимізації, а також на основі багаторічного досвіду науково-дослідної діяльності авторів, участі у наукових проєктах, виконання прикладних наукових досліджень, підготовки публікацій у провідних міжнародних наукових виданнях та впровадження отриманих результатів у практику.

Методичні рекомендації містять 4 практичні роботи, кожна з яких включає мету роботи, короткі теоретичні відомості, порядок виконання, приклади програмної реалізації, індивідуальні дослідницькі завдання, контрольні запитання та рекомендовану літературу.

Виконання запропонованих практичних робіт сприятиме формуванню у здобувачів компетентностей, необхідних для проведення самостійних наукових досліджень, розроблення математичних і комп'ютерних моделей, оптимізації складних систем та обґрунтування результатів дисертаційних досліджень.

ПРАКТИЧНА РОБОТА № 1

ДОСЛІДЖЕННЯ МЕТОДІВ ПОБУДОВИ ТА АНАЛІЗУ

МАТЕМАТИЧНИХ МОДЕЛЕЙ СКЛАДНИХ КОМП'ЮТЕРНИХ І

КІБЕРФІЗИЧНИХ СИСТЕМ ЗАСОБАМИ PYTHON

Мета роботи: сформувати практичні навички побудови, дослідження та аналізу математичних моделей складних комп'ютерних і кіберфізичних систем із використанням бібліотек NumPy, SciPy та SymPy.

У результаті виконання цієї практичної роботи здобувач повинен навчитися:

- формалізувати прикладні задачі комп'ютерної інженерії;
- будувати математичні моделі технічних систем;
- реалізовувати математичні моделі мовою Python;
- виконувати чисельні експерименти;
- аналізувати отримані результати;
- оцінювати адекватність моделей.

Необхідне програмне забезпечення: MS Office, Jupyter Notebook, JupyterLab, Google Colab, Visual Studio Code. Бібліотеки Python: numpy, scipy, sympy, matplotlib, pandas.

Короткі теоретичні відомості. Під математичною моделлю розуміють формалізований опис об'єкта, процесу або явища, який встановлює взаємозв'язок між його параметрами за допомогою математичного апарату. Залежно від характеру досліджуваної задачі математичні моделі можуть бути представлені у вигляді: алгебраїчних і трансцендентних рівнянь; систем звичайних або диференціальних рівнянь у частинних похідних; матричних рівнянь; графових моделей; стохастичних моделей; моделей масового обслуговування; моделей теорії керування; моделей, заснованих на методах машинного навчання та штучного інтелекту.

У комп'ютерній інженерії математичне моделювання широко застосовується для дослідження комп'ютерних мереж, кіберфізичних систем,

систем Інтернету речей, розподілених обчислень, систем реального часу, інтелектуальних виробничих комплексів та інших складних технічних об'єктів.

Процес математичного моделювання є ітераційним і складається з декількох взаємопов'язаних етапів.

Етап 1. Аналіз предметної області: на цьому етапі визначаються об'єкт дослідження, його структура, функціональне призначення, основні процеси, що протікають у системі, а також фактори, які можуть впливати на її функціонування.

Етап 2. Постановка задачі моделювання: на цьому етапі формулюються мета дослідження, критерії ефективності, вхідні та вихідні параметри моделі, а також встановлюються припущення й обмеження, які дозволяють спростити дослідження без суттєвої втрати адекватності.

Етап 3. Формалізація моделі: формалізація полягає у встановленні математичних залежностей між параметрами системи. Вибір математичного апарату залежить від фізичної природи процесів, необхідної точності та доступної інформації про систему.

Етап 4. Комп'ютерна реалізація моделі: побудована математична модель реалізується за допомогою відповідних програмних засобів.

Етап 5. Верифікація моделі: верифікація полягає у перевірці правильності програмної реалізації математичної моделі. На цьому етапі визначається відповідність програмного коду математичним рівнянням, перевіряється відсутність алгоритмічних помилок та оцінюється коректність виконання обчислень.

Етап 6. Валідація моделі: на цьому етапі реалізується оцінювання ступеня відповідності результатів моделювання реальним процесам або експериментальним даним. Якщо експериментальні результати відсутні, порівняння може виконуватися з аналітичними розрахунками або результатами, наведеними у науковій літературі.

Етап 7. Комп'ютерний експеримент: після підтвердження адекватності моделі проводиться серія комп'ютерних експериментів із варіюванням основних

параметрів системи. Отримані результати використовуються для аналізу поведінки досліджуваної системи, оцінювання її стійкості, дослідження впливу окремих факторів та визначення оптимальних режимів функціонування.

У загальному випадку математична модель комп'ютерної або кіберфізичної системи може бути подана у такому вигляді:

$$Y = F(X, P, t),$$

де Y – вектор вихідних характеристик; X – вектор вхідних параметрів системи; P – вектор параметрів моделі; t – час; F – набір функцій і операторів, що описують поведінку системи.

У цій практичній роботі для реалізації математичних моделей використовується мова програмування Python, яка є одним із найбільш поширених інструментів у сучасних наукових дослідженнях. Основні бібліотеки, що застосовуються під час виконання роботи:

- NumPy забезпечує можливість роботи з багатовимірними масивами та лінійною алгеброю;
- SciPy забезпечує можливість роботи з чисельними методами, алгоритмами оптимізації, інтегрування, статистика;
- SymPy забезпечує можливість роботи з символьними математичними обчисленнями;
- Pandas забезпечує можливість роботи з механізмами обробки та аналізу експериментальних даних;
- Matplotlib забезпечує можливість побудови графіків і візуалізації результатів моделювання.

Завдання та хід виконання роботи:

1. Оберіть об'єкт дослідження відповідно до тематики власної дисертаційної роботи або одного із запропонованих варіантів (ІоТ-мережа, комп'ютерна мережа, мікрокомп'ютерна система із вбудованим програмним забезпеченням, кластерна система, кіберфізична система тощо).

2. Визначте мету моделювання, вхідні та вихідні параметри системи, основні припущення і обмеження моделі, а також критерії ефективності функціонування системи.

3. Синтезуйте математичну модель досліджуваної системи. Залежно від об'єкта дослідження модель може бути представлена у вигляді: системи алгебраїчних рівнянь; системи диференціальних рівнянь; матричної моделі тощо. Вибір типу моделі має бути обґрунтованим.

4. Алгоритмізуйте та реалізуйте мовою Python побудовану модель із використанням спеціалізованих бібліотек (NumPy, SciPy, SymPy та інше). Програмна реалізація повинна забезпечувати: зміну параметрів моделі; автоматичне виконання серії експериментів; збереження результатів; побудову графіків та таблиць.

5. Виконайте серію комп'ютерних експериментів із дослідження моделі для декількох наборів її параметрів. Оцініть режими роботи системи та характер зміни основних показників ефективності.

6. Обґрунтуйте адекватність розробленої моделі шляхом перевірки виконання основних математичних співвідношень, аналізу отриманих результатів на фізичну та логічну узгодженість, а також порівняння результатів моделювання з аналітичними розрахунками або літературними даними (за наявності).

7. Виконайте комплексний аналіз отриманих результатів. У цьому випадку необхідно визначити закономірності функціонування досліджуваної системи, оцінити ефективність розробленої моделі, сформулювати рекомендації щодо вдосконалення системи або визначити напрями її подальшої оптимізації.

Типовий приклад: код і результат реалізації базового прикладу, який реалізує модель комп'ютеризованої системи керування мікрокліматом теплиць, наведено у файлі PR_1.ipynb, що розміщений на сторінці курсу на ДО Moodle.

Типовий зміст звіту до практичної роботи:

1. Титульний лист.
2. Мета та задачі практичної роботи.

3. Основні результати, отримані під час виконання практичної роботи відповідно до завдань.

4. Загальні висновки з роботи, що містять результати критичного аналізу на якісному й кількісному рівнях.

Контрольні питання:

1. Якими є основні етапи побудови математичної моделі?
2. У чому полягає відмінність між верифікацією та валідацією моделі?
3. Яким є функціональне призначення бібліотек NumPy, SciPy і SymPy?
4. Які основні джерела похибок під час комп'ютерного моделювання найбільше погіршують показник адекватності?
5. За якими критеріями оцінюється адекватність математичної моделі?
6. Якими перевагами, обмеженнями та недоліками характеризується комп'ютерний експеримент порівняно з натурним?

Список рекомендованої літератури:

1. McKinney W. Python for Data Analysis: 3rd ed. Sebastopol: O'Reilly Media, 2022. 579 p.
2. Кравченко І.В., Микитенко В.І., Тимчик Г.С. Комп'ютерне моделювання: системи і процеси: підручник. Київ: НТУУ «КПІ», 2022. 215 с.
3. Моделювання та оптимізація систем: підручник / В.М. Дубовой, Р.Н. Кветний, О.І. Михальов, А.В. Усов. Вінниця: ПП «Едельвейс», 2017. 804 с.
4. Вовна О.В., Лактіонов І.С., Лебедев В.А. Комп'ютерно-інтегрований моніторинг та керування в промислових теплицях: поточні результати і перспективи досліджень: монографія. Покровськ: ДВНЗ «ДонНТУ», 2020. 255 с.

ПРАКТИЧНА РОБОТА № 2

ДОСЛІДЖЕННЯ МЕТОДІВ ІМІТАЦІЙНОГО МОДЕЛЮВАННЯ КОМП'ЮТЕРНИХ І КІБЕРФІЗИЧНИХ СИСТЕМ

Мета роботи: сформувати практичні навички створення імітаційних моделей складних комп'ютерних і кіберфізичних систем, проведення серії комп'ютерних експериментів, аналізу характеристик функціонування систем та оцінювання їх ефективності із використанням сучасних бібліотек Python.

У результаті виконання цієї практичної роботи здобувач повинен навчитися:

- обґрунтовувати вибір методів імітаційного моделювання для дослідження складних комп'ютерних і кіберфізичних систем;
- формувати структуру імітаційної моделі відповідно до особливостей предметної області;
- реалізовувати дискретно-подієві моделі засобами Python;
- виконувати серії комп'ютерних експериментів із варіюванням параметрів системи;
- аналізувати часові характеристики функціонування систем;
- оцінювати ефективність архітектурних рішень на підставі результатів моделювання;
- формулювати науково обґрунтовані висновки щодо вдосконалення та подальшого розвитку досліджуваної системи.

Необхідне програмне забезпечення: MS Office, Jupyter Notebook, JupyterLab, Google Colab, Visual Studio Code. Бібліотеки Python: NumPy, SimPy, NetworkX, Pandas, Matplotlib.

Короткі теоретичні відомості. Імітаційне моделювання є одним із найбільш поширених методів дослідження складних комп'ютерних і кіберфізичних систем, поведінка яких не може бути достатньо повно описана лише аналітичними математичними моделями. На відміну від математичного моделювання, яке ґрунтується на побудові систем рівнянь або функціональних

залежностей, імітаційне моделювання відтворює логіку функціонування реальної системи шляхом покрокового відображення подій, взаємодії її компонентів і зміни станів у часі.

Сучасні комп'ютерні і кіберфізичні системи характеризуються високим рівнем складності, наявністю великої кількості взаємопов'язаних компонентів, паралельністю та ітераційністю процесів, випадковими подіями, а також стохастичними дестабілізуючими впливами. У таких умовах застосування імітаційного моделювання дозволяє досліджувати поведінку системи без необхідності створення дорогих фізичних прототипів або проведення натурних експериментів. До основних переваг імітаційного моделювання належать: можливість дослідження складних нелінійних систем; підтримка випадкових процесів і стохастичних моделей; можливість багаторазового проведення комп'ютерних експериментів; дослідження критичних ситуацій; аналіз альтернативних архітектур систем; оцінювання ефективності нових алгоритмів ще до їх практичного впровадження; високий ступінь відтворюваності результатів.

У сучасній комп'ютерній інженерії найбільшого поширення набули такі різновиди імітаційного моделювання: дискретно-подійне моделювання, агентне моделювання, системна динаміка, мережеве моделювання та гібридне моделювання.

У цій практичній роботі використовується дискретно-подієве моделювання, яке передбачає опис функціонування системи у вигляді послідовності подій. Зміна стану системи відбувається лише у моменти виникнення певних подій (надходження запиту, передавання пакета, завершення обчислення, спрацювання виконавчого механізму тощо).

До основних етапів імітаційного експерименту можна віднести:

Етап 1. Постановка задачі дослідження: на цьому етапі формулюється мета комп'ютерного експерименту, визначаються показники ефективності функціонування системи та параметри, що підлягають дослідженню.

Етап 2. Побудова моделі: під час цього етапу визначаються компоненти

системи, взаємозв'язки між ними, логіка функціонування та послідовність подій.

Етап 3. Реалізація імітаційної моделі: на цьому етапі запропонована модель реалізується програмними засобами, у даному випадку мовою Python із використанням бібліотеки SimPy.

Етап 4. Планування комп'ютерного експерименту: формуються серії експериментів із зміною впливових параметрів.

Етап 5. Проведення експериментів: на цьому етапі виконується багаторазове моделювання для різних сценаріїв функціонування системи.

Етап 6. Аналіз результатів: під час цього етапу обчислюються статистичні показники, оцінюється продуктивність, виявляються обмеження системи та визначаються напрями її подальшої оптимізації.

Завдання та хід виконання роботи:

1. Використайте математичну модель, розроблену під час виконання практичної роботи № 1, як основу для побудови імітаційної моделі досліджуваної комп'ютерної або кіберфізичної системи. За необхідності уточніть структуру моделі, склад її компонентів, перелік вхідних і вихідних параметрів, а також критерії оцінювання ефективності функціонування відповідно до поставленої дослідницької задачі.

2. Розробіть модель функціонування системи, яка має відображати її основні компоненти, ресурси, процеси, події, інформаційні потоки та взаємозв'язки між окремими елементами. Доцільно подати концептуальну модель у вигляді структурної схеми, блок-схеми, UML-діаграми або іншого формалізованого графічного представлення.

3. На основі запропонованої моделі реалізуйте програмну імітаційну модель мовою Python із використанням бібліотеки SimPy. Під час реалізації моделі представте окремі компоненти системи у вигляді взаємопов'язаних процесів і подій. За необхідності використовуйте бібліотеку NetworkX для моделювання мережевої топології, NumPy – для генерування випадкових величин і формування експериментальних вибірок, Pandas – для накопичення та оброблення результатів експериментів, Matplotlib – для візуалізації отриманих

результатів.

3. Розробіть програмний модуль автоматизованого проведення комп'ютерних експериментів, який забезпечує багаторазове виконання імітаційної моделі зі зміною значень щонайменше трьох параметрів системи. Вибір параметрів має бути обґрунтований предметною областю дослідження та техніко-функціональними характеристиками об'єкта дослідження.

4. Проведіть серію комп'ютерних експериментів для різних режимів функціонування системи. Для кожного експерименту визначте основні показники ефективності, що відповідають специфіці досліджуваної системи.

5. Виконайте статистичний аналіз результатів комп'ютерних експериментів. Побудуйте таблиці та графіки, визначте основні статистичні характеристики досліджуваних показників та проаналізуйте вплив зміни параметрів моделі на ефективність функціонування системи.

6. Здійсніть критичний аналіз отриманих результатів та сформулюйте науково обґрунтовані висновки щодо адекватності розробленої імітаційної моделі. На підставі проведених досліджень запропонуйте рекомендації щодо вдосконалення архітектури, алгоритмів функціонування або режимів роботи комп'ютерної або кіберфізичної системи, а також визначте можливі напрями її подальшого розвитку.

Типовий приклад: код і результат реалізації базового прикладу, який реалізує імітаційну модель комп'ютеризованої системи дискретно-подієвого керування мікрокліматом теплиць, наведено у файлі PR_2.ipynb, що розміщений на сторінці курсу на ДО Moodle.

Типовий зміст звіту до практичної роботи:

1. Титульний лист.
2. Мета та задачі практичної роботи.
3. Основні результати, отримані під час виконання практичної роботи відповідно до завдань.
4. Загальні висновки з роботи, що містять результати критичного аналізу на якісному й кількісному рівнях.

Контрольні питання:

1. У чому полягає концептуальна відмінність між математичним та імітаційним моделюванням?
2. Якими перевагами характеризується підхід на основі дискретно-подієвого моделювання під час дослідження комп'ютерних і кіберфізичних систем?
3. Якими є основні компоненти імітаційної моделі?
4. Які показники найчастіше використовуються для оцінювання ефективності функціонування імітаційних моделей комп'ютерних та/або кіберфізичних систем?
5. У чому полягає основне функціональне призначення бібліотек SimPy, NetworkX, NumPy, Pandas та Matplotlib?
6. Якими є основні етапи планування комп'ютерного експерименту в імітаційному моделюванні?
7. Якими перевагами та обмеженнями характеризується імітаційне моделювання порівняно з натурним експериментом?

Список рекомендованої літератури:

1. McKinney W. Python for Data Analysis: 3rd ed. Sebastopol: O'Reilly Media, 2022. 579 p.
2. Моделювання та оптимізація систем: підручник / В.М. Дубовой, Р.Н. Кветний, О.І. Михальов, А.В. Усов. Вінниця: ПП «Едельвейс», 2017. 804 с.
3. Diachenko G., Laktionov I., Vovna O., Aleksieiev O., Moroz D. Computer Model of an IoT Decision-Making Network for Detecting the Probability of Crop Diseases. *IoT*. 2025. Vol. 6 (1). P. 1–23. doi.org/10.3390/iot6010008.

ПРАКТИЧНА РОБОТА № 3

ДОСЛІДЖЕННЯ ПІДХОДІВ ДО БАГАТОКРИТЕРІАЛЬНОЇ ОПТИМІЗАЦІЇ МОДЕЛЕЙ СКЛАДНИХ КОМП'ЮТЕРНИХ І КІБЕРФІЗИЧНИХ СИСТЕМ

Мета роботи: сформувати практичні навички застосування сучасних методів багатокритеріальної оптимізації моделей складних комп'ютерних і кіберфізичних систем, дослідження компромісних рішень та використання сучасних програмних засобів Python для формулювання науково обґрунтованих рішень.

У результаті виконання роботи здобувач повинен навчитися:

- формулювати багатокритеріальні задачі оптимізації комп'ютерних і кіберфізичних систем;
- визначати критерії ефективності функціонування систем та їх взаємозв'язок;
- обирати параметри оптимізації відповідно до предметної області дослідження;
- використовувати сучасні алгоритми оптимізації для пошуку оптимальних параметрів моделей;
- будувати множину Парето та аналізувати компромісні рішення;
- оцінювати ефективність отриманих рішень за сукупністю критеріїв;
- формулювати практичні рекомендації щодо вдосконалення досліджуваної системи.

Необхідне програмне забезпечення: MS Office, Jupyter Notebook, JupyterLab, Google Colab, Visual Studio Code. Бібліотеки Python: NumPy, SciPy, Pandas, Matplotlib, Plotly, Optuna, DEAP, Scikit-learn.

Короткі теоретичні відомості. Оптимізація полягає у знаходженні таких значень параметрів системи, зокрема комп'ютерної та/або кіберфізичної, за яких забезпечується найкраще значення певного критерію або сукупності критеріїв ефективності. У більшості сучасних комп'ютерних і кіберфізичних систем існує

не один, а декілька критеріїв ефективності, які часто конкурують між собою. Наприклад, підвищення продуктивності може супроводжуватися зростанням енергоспоживання, а мінімізація часу відгуку – підвищенням вартості апаратних ресурсів.

У такому випадку задачу формулюють як задачу багатокритеріальної оптимізації. У загальному вигляді вона записується як:

$$\text{optimum}\{\min, \max\}(f_1(x_1, x_2, \dots, x_n), f_2(x_1, x_2, \dots, x_n), \dots, f_m(x_1, x_2, \dots, x_n)),$$

де f_i – критерії оптимізації; x_j – вектор змінних параметрів системи.

Під час аналізу та синтезу архітектур комп'ютерних та кіберфізичних систем найчастіше використовують такі критерії оптимізації: продуктивність, часова затримка, пропускна здатність, енергоспоживання, надійність, коефіцієнт завантаження, економічні показники та інше.

У сучасній науково-технічній практиці комп'ютерної інженерії широкого використання набули такі методи оптимізації: градієнтні методи, метод Ньютона, генетичні алгоритми, PSO, Байєсова оптимізація, NSGA-II та інші.

Варто також зазначити, що для багатокритеріальних задач не існує єдиного глобального оптимуму. В такому випадку визначається множина Парето. Рішення називається Парето-оптимальним, якщо неможливо покращити хоча б один критерій без погіршення іншого. Саме множина Парето використовується під час прийняття інженерних рішень.

Після знаходження оптимального рішення необхідно дослідити стійкість моделі. Для цього аналізують вплив зміни окремих параметрів на результати оптимізації. Найчастіше використовуються такі підходи: локальний аналіз чутливості, глобальний аналіз чутливості, сценарний аналіз, метод Монте-Карло.

Під час виконання цієї практичної роботи мовою Python рекомендується застосовувати такі бібліотеки: NumPy – для чисельних обчислень; SciPy Optimize – для реалізації класичних методів оптимізації; Optuna – для автоматичного пошуку параметрів; DEAP – для реалізації генетичних алгоритмів; Pandas – для

аналізу результатів; Matplotlib та Plotly – для візуалізації результатів; Scikit-learn – для аналізу чутливості та реалізації допоміжних алгоритмів машинного навчання.

Завдання та хід виконання роботи:

1. Під час виконання цієї практичної роботи використовуйте математичну та імітаційну моделі, розроблені під час виконання практичних робіт №1 та №2, як основу для постановки задачі багатокритеріальної оптимізації. Визначте перелік параметрів, що підлягають оптимізації, а також множину критеріїв ефективності відповідно до техніко-функціональних характеристик досліджуваної системи.

2. Формалізуйте математичну постановку задачі багатокритеріальної оптимізації, а саме визначте: змінні оптимізації, допустимі області зміни параметрів, критерії оптимізації, систему обмежень та цільову функцію. Під час вибору критеріїв обґрунтуйте можливі суперечності між ними.

3. Реалізуйте програмний модуль багатокритеріальної оптимізації мовою Python. Під час програмної реалізації алгоритму використовуйте одну або декілька бібліотек із рекомендованих: SciPy Optimize, Optuna, DEAP. Вибір бібліотек та, відповідно, алгоритмів оптимізації має бути обґрунтований на підставі аналізу техніко-функціональних характеристик досліджуваної системи та обраних критеріїв оптимізації. Також програмна реалізація повинна забезпечувати: автоматичну зміну параметрів моделі, запуск імітаційної моделі, накопичення результатів експериментів, автоматичне оцінювання критеріїв оптимізації, збереження результатів оптимізації.

4. Проведіть серію багатокритеріальних оптимізаційних експериментів. Під час дослідження необхідно виконати оптимізацію щонайменше за двома незалежними параметрами моделі. Наприклад: кількість сенсорних вузлів, швидкість обробки контролера, пропускна здатність мережі, кількість виконавчих пристроїв, параметри регулятора, інтервал дискретизації системи та інше. Фінальний перелік параметрів моделі обирається на підставі конкретних характеристик досліджуваної системи. Також для кожної серії експериментів необхідно оцінити: продуктивність системи, середній час реагування, пропускну

здатність та енергоспоживання.

5. Побудуйте множину Парето-оптимальних рішень. За результатами багатокритеріальної оптимізації: визначте Парето-фронт, побудуйте відповідні графіки, виконайте порівняння отриманих альтернатив, проаналізуйте компроміс між окремими критеріями оптимізації. Обґрунтуйте вибір найбільш доцільного рішення відповідно до поставленої дослідницької задачі.

6. Виконайте аналіз чутливості моделі. Для кожного оптимального рішення виконайте дослідження впливу зміни основних параметрів системи на результати оптимізації. Під час такого дослідження необхідно: визначити найбільш впливові параметри, оцінити стійкість отриманих оптимальних рішень, визначити області стабільного функціонування системи.

7. Проведіть комплексний аналіз результатів багатокритеріальної оптимізації. За результатами виконаних експериментів необхідно: порівняти початковий та оптимізований режими роботи системи, оцінити приріст ефективності за кожним критерієм, визначити можливі обмеження використаного алгоритму оптимізації, сформулювати рекомендації щодо вдосконалення архітектури або алгоритмів функціонування досліджуваної комп'ютерної чи кіберфізичної системи, визначити перспективи подальших досліджень.

Типовий приклад: код і результат реалізації базового прикладу, який реалізує багатокритеріальну оптимізацію моделі системи керування мікрокліматом теплиці із застосуванням методів SciPy, Optuna та NSGA-II, забезпечуючи пошук оптимальних параметрів функціонування системи та аналіз компромісних Парето-оптимальних рішень, наведено у файлі PR_3.ipynb, що розміщений на сторінці курсу на ДО Moodle.

Типовий зміст звіту до практичної роботи:

1. Титульний лист.
2. Мета та задачі практичної роботи.
3. Основні результати, отримані під час виконання практичної роботи відповідно до завдань.

4. Загальні висновки з роботи, що містять результати критичного аналізу на якісному й кількісному рівнях.

Контрольні питання:

1. Що розуміють під багатокритеріальною оптимізацією та в яких випадках її застосування є доцільнішим за однокритеріальну оптимізацію?

2. Які основні критерії ефективності можуть використовуватися під час оптимізації комп'ютерних і кіберфізичних систем та яким чином вони можуть суперечити один одному?

3. У чому полягають принципові відмінності між класичними методами оптимізації (SciPy), Байєсівською оптимізацією (Optuna) та еволюційними алгоритмами (NSGA-II)?

4. Які властивості характеризують Парето-оптимальні рішення?

5. Яким чином здійснюється постановка багатокритеріальної задачі оптимізації та які основні етапи її розв'язання?

6. Якими способами можна оцінити якість отриманих оптимальних рішень та обґрунтувати доцільність їх практичного використання під час проєктування комп'ютерних і кіберфізичних систем?

Список рекомендованої літератури:

1. McKinney W. Python for Data Analysis: 3rd ed. Sebastopol: O'Reilly Media, 2022. 579 p.

2. Моделирование та оптимизация систем: учебник / В.М. Дубовой, Р.Н. Кветний, О.И. Михальов, А.В. Усов. Вінниця: ПП «Едельвейс», 2017. 804 с.

3. Wang H., Laktionov I., Rodríguez Díaz F., Sánchez-Molina J.A., Li M. An optimized approach to hourly temperature and humidity setpoint generation for reducing tomato disease and saving power cost in greenhouses. *Comp. and Electronics in Agric.* 2024. 226. P. 1–16. doi.org/10.1016/j.compag.2024.109413.

ПРАКТИЧНА РОБОТА № 4

ДОСЛІДЖЕННЯ МЕТОДІВ ПОБУДОВИ І ОПТИМІЗАЦІЇ ЦИФРОВИХ ДВІЙНИКІВ ЗАСОБАМИ КОМП'ЮТЕРНОГО ЕКСПЕРИМЕНТУ

Мета роботи: сформувати практичні навички із розробки та дослідження цифрових двійників комп'ютерних і кіберфізичних систем засобами комп'ютерного експерименту із використанням мови програмування Python.

У результаті виконання цієї практичної роботи здобувач повинен навчитися:

- формалізувати структуру цифрового двійника комп'ютерної або кіберфізичної системи;
- визначати склад фізичної, інформаційної та аналітичної компонент цифрового двійника;
- реалізовувати цифровий двійник мовою Python із релевантних сучасних бібліотек;
- виконувати комп'ютерні експерименти з цифровим двійником;
- оцінювати адекватність і точність цифрового двійника;
- виконувати прогнозування режимів функціонування системи на основі цифрового двійника;
- формулювати практичні рекомендації щодо використання цифрових двійників у комп'ютерній інженерії.

Необхідне програмне забезпечення: MS Office, Jupyter Notebook, JupyterLab, Google Colab, Visual Studio Code, Node-Red. Бібліотеки Python: NumPy, SciPy, Pandas, Matplotlib, Plotly, SimPy, Scikit-learn, Optuna.

Короткі теоретичні відомості. Цифровий двійник – це динамічна цифрова модель фізичного об'єкта, процесу або системи, яка забезпечує синхронізацію із реальним об'єктом, відображення його поточного стану, прогнозування поведінки та підтримку прийняття рішень у реальному часі.

У комп'ютерній інженерії цифрові двійники застосовуються під час проектування: комп'ютерних мереж, кіберфізичних систем, IoT та ПoT-

платформ, робототехнічних комплексів, систем автоматизованого керування, систем реального часу та інше.

Типова архітектура цифрового двійника містить:

- фізичний об’єкт – реальна система або її макетний зразок;
- сенсорний рівень – засоби збору даних;
- інфокомунікаційний рівень – канали передавання даних;
- цифрова модель – алгоритмізована математична та/або імітаційна модель;
- аналітичний рівень – програмні компоненти обробки, аналітики та оптимізації;
- рівень візуалізації та підтримки прийняття рішень.

Основні властивості цифрового двійника є такими: динамічність, адаптивність, синхронізація з об’єктом, прогнозування, підтримка оптимізації, а також можливість проведення віртуальних експериментів.

Загалом процес створення цифрового двійника включає такі етапи: аналіз предметної області, вибір параметрів моніторингу, побудова математичної та/або імітаційної моделі, інтеграція експериментальних даних, калібрування параметрів, верифікація, валідація, оптимізація.

У загальному випадку цифровий двійник може бути формалізований таким чином:

$$Y(t) = F(X(t), P, U(t), E(t)),$$

де $Y(t)$ – вектор вихідних характеристик; $X(t)$ – вектор станів системи; P – сукупність параметрів моделі; $U(t)$ – впливи керування; $E(t)$ – зовнішні дестабілізуючі впливи.

До основних процедур розробки та дослідження цифрового двійника, що розглядаються в цій практичній роботі відносяться: калібрування, валідація та комп’ютерний сценарний експеримент.

Калібрування полягає у визначенні параметрів P , що мінімізують різницю

між результатами моделювання та експериментальними даними:

$$J(P) = \sum_{i=1}^n (Y_{exp_i} - Y_{model_i}(P))^2,$$

де $J(P)$ – цільова функція, що характеризує сумарну квадратичну похибку між експериментальними даними та результатами моделювання; P – сукупність параметрів моделі, що підлягають калібруванню; Y_{exp_i} – експериментально виміряне значення досліджуваного параметра під час i^{zo} спостереження; $Y_{model_i}(P)$ – значення відповідного параметра, отримане за допомогою цифрового двійника для i^{zo} спостереження за заданого вектора параметрів P ; i – номер експериментального спостереження; n – загальна кількість експериментальних спостережень.

Під час реалізації процедури калібрування мовою Python можуть бути використані такі підходи: метод найменших квадратів, градієнтні методи, Байєсова оптимізація, генетичні алгоритми, NSGA-II.

Валідація цифрового двійника реалізується шляхом оцінки адекватності за такими метриками: RMSE, MAE, MAPE, R^2 .

Комп'ютерний експеримент із дослідження цифрових двійників дозволяє реалізувати: сценарний аналіз, аналіз відмов, аналіз чутливості, оптимізацію режимів, предикативну аналітику та інше.

Під час виконання цієї практичної роботи мовою Python рекомендується застосовувати такі бібліотеки: NumPy – для чисельних обчислень; SciPy – для інтегрування та оптимізація; Pandas – для роботи з експериментальними даними; Matplotlib / Plotly – для візуалізації результатів; SimPy – для дискретно-подієвого моделювання; Scikit-learn – для аналізу даних та оцінювання точності; Optuna – для оптимізації параметрів.

Завдання та хід виконання роботи:

1. Використайте результати практичних робіт №1 – №3 як основу для побудови цифрового двійника досліджуваної комп'ютерної або кіберфізичної

системи. За необхідності уточніть математичну модель, імітаційну модель та результати оптимізації відповідно до особливостей обраного об'єкта дослідження.

2. Розробіть концептуальну архітектуру цифрового двійника. Визначте взаємозв'язки між фізичним об'єктом, джерелами експериментальних даних, математичною моделлю, імітаційною моделлю, модулем оптимізації, засобами синхронізації даних та модулем прогнозування. Побудуйте структурну схему цифрового двійника.

3. Інтегруйте математичну модель (ПР №1), імітаційну модель (ПР №2) та результати багатокритеріальної оптимізації (ПР №3) у єдиний програмний модуль цифрового двійника мовою Python. Забезпечте можливість змінювати параметри моделі, запускати комп'ютерні експерименти та автоматично накопичувати результати.

4. Підготуйте набір експериментальних даних, що характеризують функціонування фізичної системи. За відсутності реальних експериментальних даних виконайте їх генерацію засобами об'єктивної симуляції. Реалізуйте механізм синхронізації цифрового двійника з експериментальними даними.

5. Виконайте калібрування параметрів цифрового двійника шляхом мінімізації різниці між експериментальними та модельними даними. Для цього використовуйте бібліотеки SciPy Optimize, Optuna або інші програмні засоби оптимізації. Визначте оптимальні параметри цифрового двійника та оцініть їх фізичну інтерпретацію.

6. Проведіть верифікацію та валідацію цифрового двійника. Оцініть його адекватність за допомогою показників RMSE, MAE, MSE, MAPE, R^2 або інших статистичних критеріїв. Побудуйте графіки порівняння експериментальних і модельних даних та виконайте аналіз отриманих результатів.

7. Проведіть серію комп'ютерних експериментів із цифровим двійником у середовищі Node-Red. Дослідіть поведінку системи за різних сценаріїв функціонування, змін зовнішніх впливів, режимів навантаження або конфігурацій системи. Виконайте прогнозування основних показників

ефективності та проаналізуйте вплив зміни параметрів на функціонування досліджуваної системи.

8. Проведіть комплексний аналіз отриманих результатів. Порівняйте характеристики базової математичної моделі, імітаційної моделі, оптимізованої моделі та цифрового двійника. Оцініть переваги використання цифрового двійника під час моделювання, прогнозування та підтримки прийняття рішень.

9. Сформулюйте рекомендації щодо подальшого вдосконалення цифрового двійника, визначте основні обмеження розробленої моделі, перспективи її інтеграції з реальними комп'ютерними або кіберфізичними системами, а також напрями подальших наукових досліджень.

Типовий приклад: код і результат реалізації базового прикладу, який реалізує цифрового двійника системи керування мікрокліматом теплиці, забезпечуючи генерацію експериментальних даних засобами об'єктивної симуляції, калібрування математичної моделі, оцінювання її адекватності, прогнозування режимів функціонування та оптимізацію температурної уставки засобами Python, наведено у файлі PR_4.ipynb, що розміщений на сторінці курсу на ДО Moodle.

Типовий зміст звіту до практичної роботи:

1. Титульний лист.
2. Мета та задачі практичної роботи.
3. Основні результати, отримані під час виконання практичної роботи відповідно до завдань.
4. Загальні висновки з роботи, що містять результати критичного аналізу на якісному й кількісному рівнях.

Контрольні питання:

1. Які основні компоненти цифрового двійника забезпечують взаємодію між фізичною системою й її цифровою моделлю?
2. Чим цифровий двійник відрізняється від математичної та імітаційної моделей комп'ютерної або кіберфізичної системи?
3. Якими є основні етапи побудови цифрового двійника? Яку роль у цьому

процесі відіграють математичне моделювання, комп'ютерний експеримент і багатокритеріальна оптимізація?

4. У чому полягають процеси калібрування, верифікації та валідації цифрового двійника? Які показники використовуються для оцінювання його адекватності?

5. Які методи синхронізації цифрового двійника з експериментальними або сенсорними даними застосовуються в сучасних комп'ютерних і кіберфізичних системах?

6. Яким чином цифровий двійник може використовуватися для прогнозування режимів функціонування, підтримки прийняття рішень та оптимізації параметрів складних технічних систем?

7. Які переваги, обмеження та потенційні джерела похибок характерні для цифрових двійників під час моделювання комп'ютерних і кіберфізичних систем?

8. Які перспективи розвитку технології цифрових двійників у сфері комп'ютерної інженерії Ви вбачаєте?

Список рекомендованої літератури:

1. McKinney W. Python for Data Analysis: 3rd ed. Sebastopol: O'Reilly Media, 2022. 579 p.

2. Моделювання та оптимізація систем: підручник / В.М. Дубовой, Р.Н. Кветний, О.І. Михальов, А.В. Усов. Вінниця: ПП «Едельвейс», 2017. 804 с.

3. Node-RED: Documentation. URL: <https://nodered.org/docs/> (дата звернення 17.03.2025).

КРИТЕРІЇ ОЦІНЮВАННЯ ПРАКТИЧНИХ РОБІТ

Навчальні досягнення здобувачів вищої освіти за результатами вивчення курсу оцінюватимуться за шкалою, що наведена нижче:

Рейтингова шкала	Інституційна шкала
90–100	відмінно
74–89	добре
60–73	задовільно
0–59	незадовільно

Здобувачі вищої освіти можуть отримати підсумкову оцінку з навчальної дисципліни на підставі поточного оцінювання знань за умови, якщо набрана кількість балів складатиме не менше як 60 балів.

Максимальне оцінювання:

Теоретична частина	Практична частина	Разом
36	64	100

При своєчасному (протягом двох тижнів від дати проведення заняття) та правильному виконанні завдань відповідної практичної роботи в повному обсязі здобувач отримує оцінку 16 балів, із яких:

10 балів – виконання та оформлення звіту з практичної роботи;

6 балів – захист практичної роботи.

За відсутності об'єктивних причин у разі несвоєчасного виконання практичної роботи передбачено -1 бал за кожен прострочений тиждень.

Критерії оцінювання практичних робіт:

Бали за виконання та оформлення	
10	Роботу виконано в повному обсязі та правильно, усі передбачені етапи реалізовано, отримані результати наведено, описано й проаналізовано. Звіт містить необхідні матеріали та оформлений відповідно до зазначених рекомендацій.
8–9	Роботу виконано переважно правильно та в достатньому обсязі, отримано основні передбачені результати. Наявні окремі неточності або незначні помилки в реалізації чи оформленні звіту, які не мають істотного впливу на загальний результат роботи.
6–7	Роботу виконано переважно правильно, однак наявні окремі помилки або неточності. Основні результати отримано та представлено, але відсутній або недостатньо обґрунтований їх аналіз, не розкрито окремі аспекти виконання роботи.

4–5	Роботу виконано частково: реалізовано лише окремі етапи завдання, наявні помилки в налаштуванні, реалізації або виконанні окремих операцій. Результати перевірки представлено недостатньо або фрагментарно, звіт потребує доопрацювання.
1–3	Виконано лише окремі етапи, звіт неповний.
0	Роботу не виконано або поданий результат не відповідає поставленому завданню та не дає змоги оцінити досягнення передбачених результатів.
Бали за захист	
6	Здобувач упевнено та повно пояснює виконані етапи роботи, обґрунтовує використані методи та отримані результати, правильно й аргументовано відповідає на запитання.
4–5	Здобувач загалом правильно пояснює виконані етапи роботи та отримані результати, розуміє використані методи, але допускає окремі неточності або неповно відповідає на окремі запитання.
2–3	Здобувач розуміє основний зміст роботи, але має труднощі з поясненням окремих етапів, методів або отриманих результатів, а також допускає принципові неточності у відповідях на запитання.
1	Здобувач має загальне уявлення про виконану роботу, може частково пояснити її зміст або окремі етапи, але не може обґрунтувати отримані результати та відповісти на більшість запитань.
0	Здобувач не може пояснити виконання роботи, не розуміє її основного змісту та не відповідає на основні запитання.

Навчальне видання

Лактіонов Іван Сергійович

Каштан Віта Юріївна

Гнатушенко Володимир Володимирович

Олевський Віктор Ісаакович

**МОДЕЛЮВАННЯ ТА ОПТИМІЗАЦІЯ КОМП'ЮТЕРНИХ І
КІБЕРФІЗИЧНИХ СИСТЕМ І МЕРЕЖ**

Методичні рекомендації до виконання практичних робіт
для здобувачів ступеня доктора філософії
спеціальності F7 Комп'ютерна інженерія

Видано в авторській редакції.

Електронний ресурс.

Підписано до видання 12.06.2025. Авт. арк. 1,1.

Національний технічний університет «Дніпровська політехніка».

49005, м. Дніпро, просп. Дмитра Яворницького, 19.