МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Тернопільський національний технічний університет
імені Івана Пулюя

Кафедра програмної інженерії


# Конспект лекцій

з дисципліни

## «Ідентифікація складних систем і об’єктів»

СЕН ID: **6687**

для здобувачів третього (освітньо-наукового) рівня вищої освіти
ОНП «Інженерія програмного забезпечення»

Тернопіль — 2025

УДК 004.4(075.8)
ББК 32.973.202я73
Б83

**Укладач:**
В.М. Бревус, PhD

**Рецензент:**
О.П. Ясній, д.т.н., професор

Розглянуто й затверджено на засіданні кафедри програмної інженерії.
Протокол № 1 від 01.09.2025.

Розглянуто й затверджено на засіданні методичної комісії факультету комп'ютерно-інформаційних систем та програмної інженерії Тернопільського національного технічного університету імені Івана Пулюя.
Протокол № 1 від 01.09.2025.

Б83 Конспект лекцій з дисципліни «Ідентифікація складних систем і об'єктів» для здобувачів третього (освітньо-наукового) рівня вищої освіти ОНП «Інженерія програмного забезпечення» / Укладач: Бревус В.М. – Тернопіль: ТНТУ імені Івана Пулюя, 2025 – 74 с.

**Відповідальний за випуск:** М.Р. Петрик, докт. фіз.-мат. наук, професор

# АНОТАЦІЯ

У конспекті лекцій розглянуто основні концепції та методи ідентифікації складних систем і об’єктів. Викладено теоретичні основи моделювання, аналізу та оцінювання параметрів динамічних систем. Особливу увагу приділено сучасним підходам до ідентифікації, таким як методи найменших квадратів, максимальної правдоподібності та нейромережеві технології.

Конспект лекцій призначений для здобувачів третього (освітньо-наукового) рівня вищої освіти, які навчаються за освітньо-науковою програмою «Інженерія програмного забезпечення». Він також буде корисним для аспірантів, науковців та інженерів, що спеціалізуються в галузі системного аналізу, керування та обробки даних.

**Ключові слова:** ідентифікація систем, складні системи, моделювання, оцінювання параметрів, системний аналіз, нейронні мережі.

# ABSTRACT

The lecture notes cover the fundamental concepts and methods for the identification of complex systems and objects. The theoretical foundations of modeling, analysis, and parameter estimation for dynamic systems are presented. Special attention is given to modern identification approaches, such as the least squares method, maximum likelihood, and neural network technologies.

The lecture notes are intended for students of the third (educational and scientific) level of higher education enrolled in the “Software Engineering” educational and scientific program. It will also be useful for postgraduate students, researchers, and engineers specializing in system analysis, control, and data processing.

**Keywords:** system identification, complex systems, modeling, parameter estimation, system analysis, neural networks.

# Зміст

# Вступ

Цей курс присвячений вивченню методів ідентифікації складних систем та об'єктів. Ідентифікація систем є ключовою дисципліною в інженерії та наукових дослідженнях, що дозволяє створювати математичні моделі динамічних процесів на основі експериментальних даних.

Сучасний світ характеризується стрімким розвитком технологій, зростанням складності технічних, соціальних, біологічних та екологічних систем. Моделювання та ідентифікація систем є фундаментальними інструментами для аналізу, синтезу та управління об'єктами різної природи. Особливу увагу приділяється питанням побудови адекватних моделей, що відображають реальні процеси, а також методам отримання інформації про структуру та параметри системи на основі експериментальних даних.

Важливою особливістю сучасних досліджень є фокус на складних системах, які мають велику кількість взаємодіючих елементів, нелінійні зв'язки, самоорганізацію, емерджентність та адаптивність. Складні системи — це такі, в яких поведінка цілого не зводиться до властивостей окремих частин, а виникає з їх взаємодії. Прикладами складних систем є біологічні організми, екосистеми, соціальні мережі, економічні ринки, технічні комплекси.

Ідентифікація складних систем вимагає використання сучасних математичних методів, алгоритмів машинного навчання, статистичних підходів та комп'ютерного моделювання. Особливу роль відіграють питання збору, обробки та аналізу великих обсягів даних, а також розробка моделей, здатних враховувати невизначеність, стохастичність та багаторівневу структуру об'єктів.

Протягом курсу ми розглянемо:

- Основні поняття теорії систем та моделювання.
- Класичні та сучасні підходи до ідентифікації.

- Методи оцінки параметрів лінійних та нелінійних моделей.
- Практичні аспекти планування експерименту та валідації моделей.
- Особливості моделювання та ідентифікації складних систем.
- Приклади застосування методів у різних галузях: техніка, біологія, економіка, соціальні науки.

Метою курсу є формування у здобувачів PhD теоретичних знань та практичних навичок, необхідних для самостійного проведення досліджень, пов'язаних з моделюванням та ідентифікацією складних систем у різних предметних областях.

Слухачі отримають досвід роботи з сучасними програмними засобами, навчаться застосовувати математичні та комп'ютерні методи для аналізу реальних об'єктів, а також розвиватимуть навички критичного мислення, необхідні для вирішення міждисциплінарних задач.

Завдяки інтеграції теоретичних знань і практичних кейсів, курс сприятиме розвитку компетентностей у сфері системного аналізу, моделювання, ідентифікації та управління складними об'єктами.

# Розділ 1

# Предмет та задачі ідентифікації складних систем

**Мистецтво розшифровки реальності**

## З чого все починається? Системи та моделі

### Про що ми взагалі говоримо?

Уявіть, що ви натрапили на дивну чорну скриньку. Ви не знаєте, що в неї всередині, але у вас є кілька кнопок, які можна натискати, і екран, на якому щось з'являється. Ви натискаєте червону кнопку (це ваш **вхід**, або **input**) і бачите, як на екрані з'являється цифра 5 (це **вихід**, або **output**). Ви натискаєте її знову — з'являється 10. Натискаєте синю кнопку — бачите 3. Що, в біса, відбувається всередині?

Ось це і є фундаментальна гра, в яку грають наука та інженерія. Ми маємо справу зі світом, повним таких «чорних скриньок». Деякі з них — це планети, що обертаються навколо Сонця, інші — хімічні реактори, економіка країни, або навіть звичайний тостер на вашій кухні.

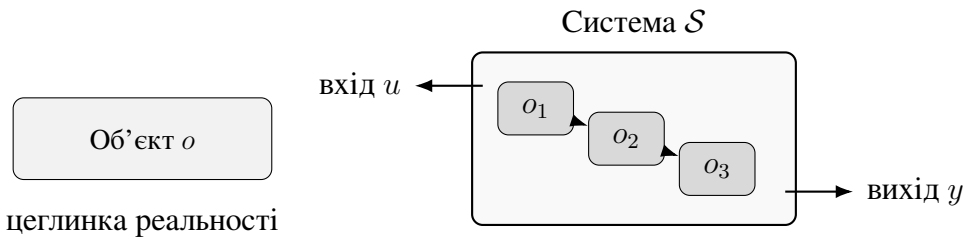


Рис. 1.1: Схематичне зображення об'єкта та системи: взаємодія елементів, вхід та вихід.

Коли ми говоримо про **об'єкт**, ми маємо на увазі якусь «річ», яку ми можемо спостерігати. А коли кілька таких речей взаємодіють разом, щоб робити щось цікаве, ми називаємо це **системою**. Автомобіль — це система. Його двигун — це підсистема, а поршень у двигуні — це об'єкт. Система — це щось більше, ніж просто сума її частин. Ви не можете зрозуміти, як працює автомобіль, дивлячись лише на купу його деталей, розкладених на підлозі [1]. Див. рис.1.1.

## Навіщо взагалі потрібні моделі? Або чому карти корисні

Навіщо нам заглядати всередину чорної скриньки? Бо ми хочемо зрозуміти її правила. Ми хочемо створити **модель** — свого роду карту, що описує, як ця скринька працює. Карта — це не територія, але вона страшенно корисна, якщо ви хочете кудись дістатися. Так само і з моделями. Вони не є самою реальністю, але вони дозволяють нам робити дивовижні речі [2, 3].

- **Зазирнути в майбутнє (Прогнозування):** Якщо я буду тримати важіль тостера натиснутим 30 секунд, наскільки темним буде мій тост? Модель може дати відповідь.
- **Стати ляльководом (Керування):** Як мені спроєктувати тостер, щоб він *завжди* робив ідеальні тости, незалежно від типу хліба? Для цього потрібна модель його поведінки.
- **Грати в «а що, як...» без ризику (Симуляція):** А що, як я підключу тостер до розетки з іншою напругою? Чи не вибухне він? Краще спершу перевірити це на моделі!
- **Розповісти історію (Пояснення):** Модель розповідає нам історію про те,

як працює тостер: струм тече по дроту, дріт нагрівається, тепло підсмажує хліб. Це допомагає нам зрозуміти суть процесу.

## Велика детективна гра: Ідентифікація систем

Отже, як нам розшифрувати правила нашої чорної скриньки? Цей процес і є **ідентифікацією систем**. Це робота детектива. У нас є докази — дані про те, що ми подавали на вхід і що отримували на виході, — і ми намагаємося відновити картину «злочину», тобто зрозуміти внутрішні закони системи. Див. рис.1.2.

Формально, **ідентифікація системи** — це процес побудови математичних моделей динамічних систем на основі експериментальних даних [4]. Це наукова діяльність з виведення внутрішніх механізмів, зв'язків і принципів функціонування системи, часто в умовах, коли все навколо шумить, заважає і намагається нас заплутати.

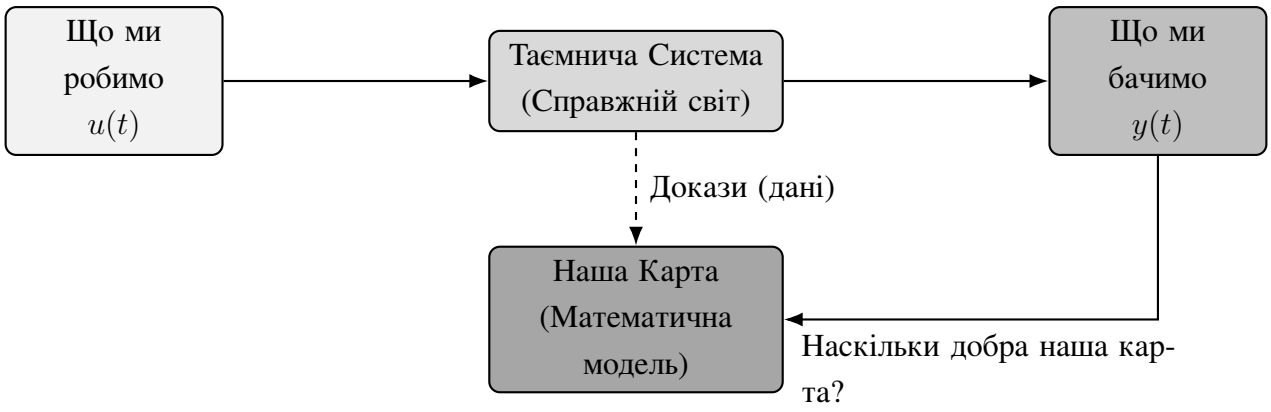


Рис. 1.2: Процес ідентифікації: як ми будуємо модель на основі даних про вхід та вихід системи.

## Наш детективний набір: кроки ідентифікації

Процес ідентифікації — це не пряма дорога, а скоріше цикл, де ми постійно повертаємося назад і вдосконалюємо свої ідеї. Ось основні кроки нашого розслідування:

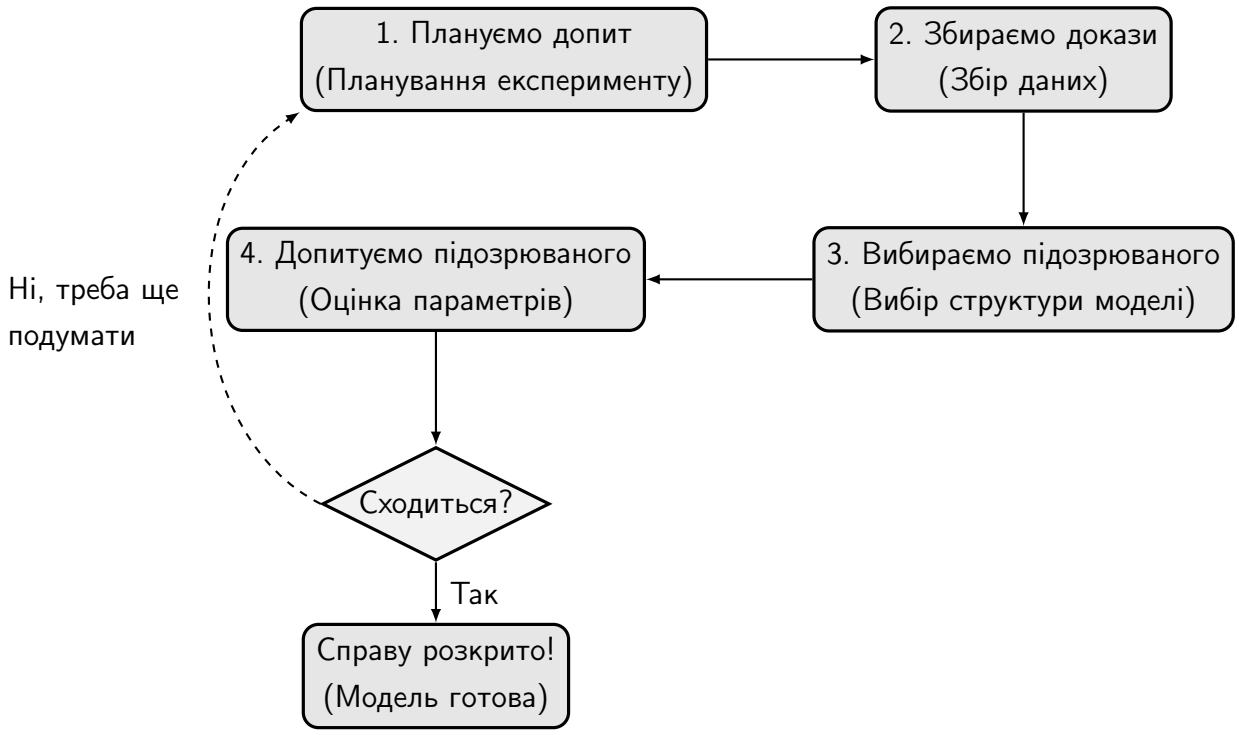


Рис. 1.3: Етапи ідентифікації системи: цикл планування, збору даних, вибору моделі, оцінки параметрів та перевірки.

## Спектр знань: від «білої» до «чорної» скриньки

Коли ми вибираємо «підозрюваного» (структуру моделі), ми знаходимося десь на спектрі між повним знанням і повним невіглаством.

- **«Біла скринька» (White-box):** Ми — як годинникар, що знає, як працює кожна шестерня. Ми знаємо всі фізичні закони, що керують системою (наприклад, закони Ньютона для механічної системи). Нам залишається лише виміряти точні розміри цих «шестерень» (параметри).
- **«Чорна скринька» (Black-box):** Ми поняття не маємо, що всередині. Ми просто шукаємо будь-яку математичну формулу, яка добре описує зв'язок між входом і виходом. Це може бути поліном, нейронна мережа чи щось інше. Ми не питаємо «чому?», ми питаємо «що працює?».
- **«Сіра скринька» (Grey-box):** Це найцікавіший і найпоширеніший випадок. У нас є певна інсайдерська інформація. Ми знаємо, що в тостері є нагрівальний елемент і таймер, але ми не знаємо їхніх точних характеристик. Це суміш фізики та детективної роботи з даними.

# Формулювання задачі

Давайте трохи формалізуємо нашу детективну роботу.

1. **Дано:** У нас є докази — спостереження за входом $u(t)$ та виходом $y(t)$ для моментів часу $t = 1, \dots, N$.

2. **Обрати клас моделей (підозрюваних):** Наприклад, ми припускаємо, що поточний вихід залежить від попередніх виходів та входів. Це модель класу ARX (AutoRegressive with eXternal input):

$$y(t) + a_1 y(t-1) + \cdots = b_1 u(t-1) + \cdots + e(t) \tag{1.1}$$

   де $e(t)$ — це непередбачуваний шум, «туман війни».

3. **Визначити критерій успіху:** Як ми зрозуміємо, що наша модель добра? Ми хочемо, щоб помилка між реальним виходом $y(t)$ і тим, що прогнозує наша модель $\hat{y}(t|\theta)$, була якомога меншою. Наприклад, мінімізуємо суму квадратів помилок:

$$J(\theta) = \sum_{t=1}^{N} (y(t) - \hat{y}(t|\theta))^2 \to \min_{\theta} \tag{1.2}$$

4. **Знайти найкращі параметри $\theta^*$:** Це і є «допит». Ми знаходимо такі значення параметрів $\theta = \{a_1, \dots, b_1, \dots\}$, які мінімізують нашу функцію втрат.

## Приклад даних для ідентифікації

Осъ як можуть виглядати наші «докази» — вхідний сигнал, яким ми «штовхаємо» систему, і вихідний, який ми спостерігаємо.

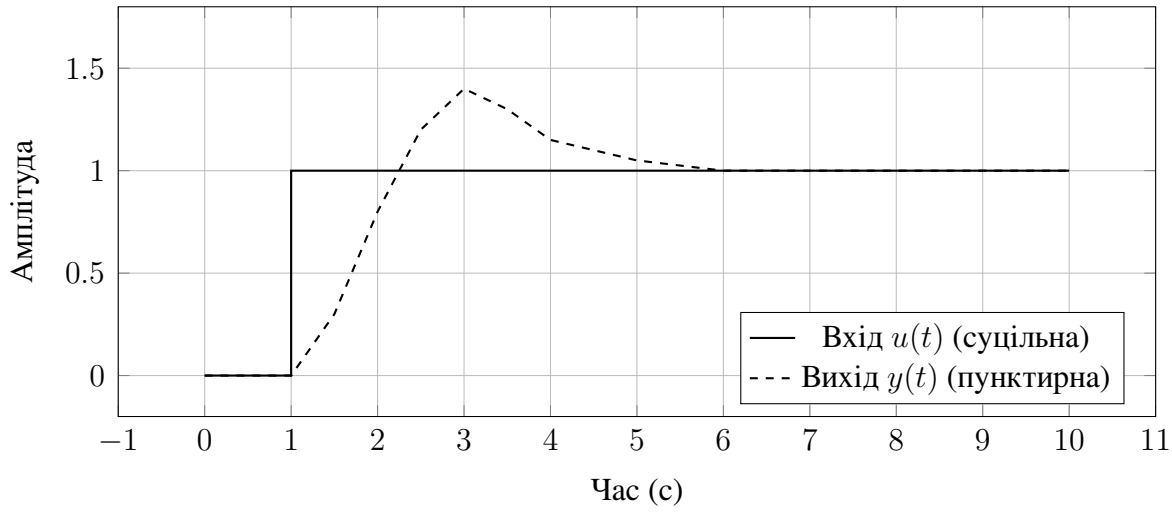


Рис. 1.4: Приклад експериментальних даних для ідентифікації: вхідний (суцільна) та вихідний (пунктирна) сигнали.

# Коли все стає по-справжньому цікавим: світ складних систем

Досі ми говорили про системи, які, хоч і бувають заплутаними, але все ж поводяться більш-менш... пристойно. Годинник — складний, але не *комплексний*. Це як швейцарський годинник: багато деталей, але їхня взаємодія повністю передбачувана. Якщо ви знаєте, як працює кожна шестерня, ви знаєте, як працює годинник.

Але що, як ми подивимося на зграю птахів? Або на мурашник? Або на фондовий ринок? Тут починається справжня магія. Це — **складні системи** (complex systems). І вони грають за зовсім іншими правилами [5].

Уявіть собі, що ви намагаєтеся змоделювати дорожній затор. Ви можете ідеально описати кожен автомобіль: його швидкість, потужність двигуна, час реакції водія. Але чи допоможе це вам передбачити, де і коли виникне затор? Не зовсім. Затор — це не властивість одного автомобіля. Це щось, що *виникає* із взаємодії багатьох автомобілів.

## Ключові ідеї складних систем

Складні системи мають кілька дивовижних властивостей, які змушують нас думати інакше.

- **Емерджентність (Emergence):** Це коли ціле стає чимось більшим, ніж просто сума його частин. Жодна окрема мураха не знає, як побудувати мурашник, але разом вони створюють дивовижні архітектурні споруди. Жоден окремий нейрон у вашому мозку не має свідомості, але разом вони створюють... вас! Емерджентність — це коли з простих локальних правил народжується складна глобальна поведінка. Це магія, що виникає з натовпу.

- **Самоорганізація (Self-organization):** Хто керує зграєю птахів? Хто каже їм, куди повертати? Відповідь: ніхто. Немає головного птаха-командира. Порядок виникає сам по собі, із простих правил, яких дотримується кожен птах (наприклад: «не врізайся в сусіда», «лети приблизно в тому ж напрямку, що й сусіди»). Це і є самоорганізація — порядок із хаосу без центрального контролю.

- **Нелінійність (Nonlinearity):** У простих системах, якщо ви штовхнете щось вдвічі сильніше, воно і поїде вдвічі далі. У складних системах все не так. Маленький поштовх може не викликати жодного ефекту... а може спричинити лавину. Одна сніжинка може скотитися з гори без наслідків, а може запустити величезну лавину. Це ефект метелика. У таких системах причина і наслідок не завжди пропорційні, що робить довгострокове прогнозування майже неможливим.

- **Зворотні зв'язки (Feedback Loops):** Елементи в складній системі постійно впливають один на одного. Уявіть собі мікрофон біля динаміка. Звук з динаміка потрапляє в мікрофон, підсилюється, знову виходить з динаміка, і так по колу, доки не почуєте оглушливий писк. Це *позитивний зворотний зв'язок* — він підсилює сам себе. А є *негативний зворотний зв'язок*, як у термостата: стало занадто спекотно — він вимкнув опалення; стало холодно — увімкнув. Він стабілізує систему. Складні системи сповнені таких петель, які роблять їхню поведінку ще більш заплутаною та цікавою.

Моделювання таких систем вимагає зовсім інших інструментів. Замість того, щоб писати одне велике рівняння для всієї системи, ми часто створюємо «агентів» (наприклад, модель одного птаха чи автомобіля) і задаємо їм прості

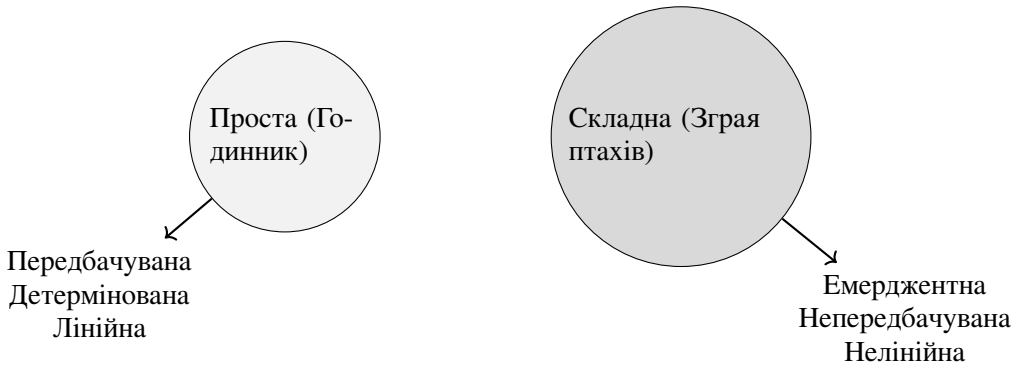


Рис. 1.5: Різниця між простою (або ускладненою) та складною системою.

правила взаємодії. А потім ми запускаємо симуляцію і з подивом спостерігаємо, яка дивовижна поведінка виникає з цього «цифрового мурашника». Це називається агент-орієнтованим моделюванням.

# Підсумок: Чому складність важлива?

Уявіть собі світ, де все складається з безлічі взаємодіючих частин — від клітин у вашому тілі до людей у місті, від атомів у хмарі до комп'ютерів в інтернеті. Складні системи — це саме такі світи, де прості правила на локальному рівні породжують дивовижну, несподівану поведінку на глобальному рівні [6].

**Взаємодії:** У складних системах компоненти постійно взаємодіють між собою та з середовищем. Це як розмова у великій компанії — кожен впливає на всіх, і навіть невелика зміна може породити нову хвилю подій.

**Емерджентність:** «Ціле — більше, ніж сума частин». Жодна окрема клітина не знає, як виглядає організм, але разом вони створюють життя. Так само, з простих взаємодій виникають торнадо, свідомість, економічні кризи.

**Динаміка і хаос:** Складні системи змінюються з часом, часто непередбачувано. Маленька зміна може призвести до великих наслідків (ефект метелика). Дві майже однакові системи можуть швидко стати зовсім різними.

**Самоорганізація:** Порядок може виникати без диригента. Зграя птахів, мурашник, місто — все це приклади самоорганізації, коли глобальна структура виникає з локальних взаємодій.

**Адаптація:** Складні системи здатні вчитися, змінюватися, пристосовуватися до нових умов. Вони можуть бути стійкими до змін, а іноді навіть ставати кращими після потрясінь.

**Міждисциплінарність:** Складність — це не лише про фізику чи біологію. Це про все: економіку, соціальні мережі, екологію, технології. Скрізь, де багато взаємодій, виникає складність.

**Методи:** Щоб зрозуміти складні системи, нам потрібні нові інструменти — моделювання, симуляції, аналіз даних. Комп'ютери допомагають нам бачити те, що не видно неозброєним оком.

> «В наступному столітті наука про складність стане ключовою»
> — Stephen Hawking

**Висновок:** Складність — це не проблема, а джерело краси, творчості та несподіванок у світі. Вивчаючи складні системи, ми краще розуміємо себе, природу і суспільство.

*Джерело:* https://complexityexplained.github.io/

# Контрольні запитання

1. Що таке «система» та «об'єкт»? Наведіть приклад системи та поясніть, чому її не можна звести до суми її частин.

2. Поясніть метафору «чорної скриньки». Чому вона є центральною для ідентифікації систем?

3. Назвіть та коротко опишіть чотири основні причини, чому моделі є корисними в науці та інженерії.

4. Що таке ідентифікація систем? Опишіть її як «детективну гру».

5. Перелічіть та поясніть основні етапи циклу ідентифікації системи. Чому це ітеративний процес?

6. У чому полягає різниця між моделями «білої», «сірої» та «чорної» скриньки? Наведіть приклад для кожного типу.

7. Які ключові елементи входять до формальної постановки задачі ідентифікації?

8. Чим відрізняється складна (complex) система від ускладненої (complicated)? Наведіть приклад обох.

9. Що таке емерджентність? Наведіть приклад емерджентної поведінки, що виникає з простих локальних правил.

10. Поясніть концепцію самоорганізації. Хто керує зграєю птахів або колонією мурах?

11. Що означає нелінійність у контексті складних систем? Як це пов'язано з «ефектом метелика»?

12. Поясніть роль позитивних та негативних зворотних зв'язків у системах. Наведіть приклад для кожного типу.

# Розділ 2

# Принципи ідентифікації та підготовка даних

**Мистецтво не обдурити себе: як обрати правильну модель?**

## Детектив у лабіринті дзеркал: проблема перенавчання

Уявіть, що ви — детектив, який розслідує справу. Ви зібрали купу доказів: відбитки пальців, свідчення свідків, записи з камер. Тепер вам потрібно скласти з цього цілісну картину. Чому ми взагалі цим займаємось? Як влучно зауважив Джошуа Епштейн, ми будуємо моделі не лише для прогнозування, а й для того, щоб пояснити світ, структурувати наше мислення та зрозуміти, що ми знаємо, а чого — ні [2].

Ви можете створити **дуже просту теорію**. Наприклад: «Злочин скоїв дворецький». Ця версія проста, зрозуміла, але ігнорує купу доказів, які їй суперечать. Вона, скоріш за все, помилкова, бо надто спрощена. У нашому світі це називається **недонавчання (underfitting)** або високе **зміщення (bias)**. Ваша модель настільки упереджена своєю простотою, що не бачить реальності.

Але є й інша крайність. Ви можете створити **неймовірно складну теорію**, яка пояснює абсолютно кожну дрібницю. «Дворецький — це лише прикриття для міжнародної змови за участю ілюмінатів, які залишили на місці злочину таємне послання у вигляді пташиного посліду, а те, що свідок чхнув о 14:32, було насправді секретним сигналом». Така теорія ідеально «пасує» до всіх зі-

браних фактів, навіть до випадкових і неважливих. Але чи повірите ви в неї? Чи допоможе вона розкрити наступний злочин? Навряд чи.

Це і є **перенавчання (overfitting)**. Ви настільки сильно вчепилися в деталі наявних даних, що прийняли випадковий «шум» за закономірність. Ваша модель ідеально описує минуле, але абсолютно безпорадна у прогнозуванні майбутнього. Вона втратила зв'язок із реальністю і живе у світі власних фантазій.

Приклад недонавчання, оптимальної моделі та перенавчання

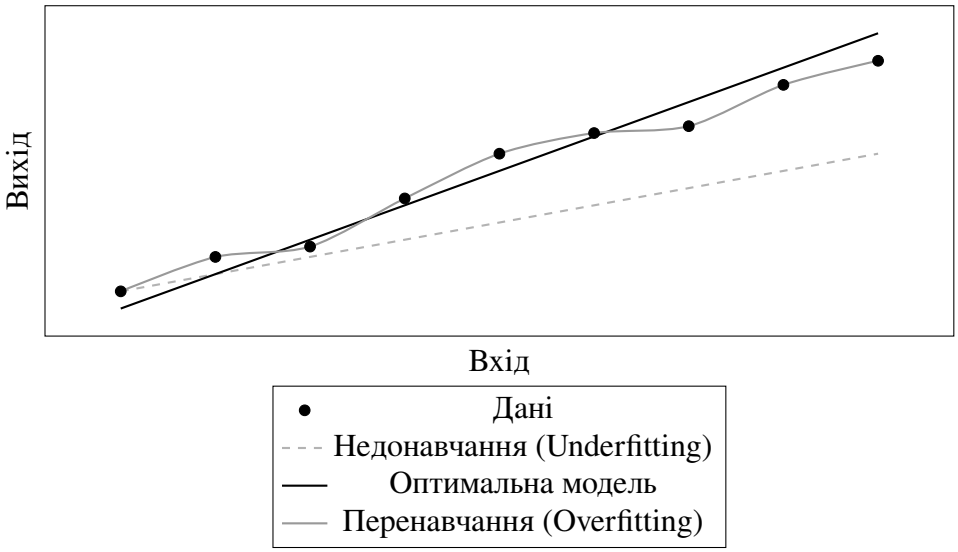


Рис. 2.1: Візуалізація проблеми вибору моделі. Червона лінія — занадто проста модель. Зелена — оптимальна. Помаранчева — ідеально проходить через навчальні точки, але, ймовірно, покаже поганий результат на нових, невидимих даних.

На ілюстрації 2.1 ми бачимо три спроби описати набір даних. Пряма червона лінія — це приклад недонавчання. Вона занадто проста і не враховує загальну тенденцію в даних, тому її помилка буде великою. Помаранчева лінія, навпаки, є прикладом перенавчання. Вона ідеально проходить через кожну точку даних, ніби намагаючись пояснити кожне випадкове відхилення. Така модель, ймовірно, зазнає краху при спробі спрогнозувати нові дані, оскільки вона вивчила шум, а не справжню залежність.

Зелена лінія представляє золоту середину. Вона вловлює основну структуру даних, ігноруючи при цьому незначний шум. Це і є мета ідентифікації — знайти модель, яка не є ані надто простою, ані надто складною. Цей баланс є ключовим для створення моделей, які є не тільки точними, але й надійними, тобто здатними до узагальнення.

Цей танець між простотою та складністю — фундаментальна дилема, відома як **компроміс між зміщенням та дисперсією (Bias-Variance Tradeoff)**. Проста модель має велике зміщення (вона систематично помиляється), але низьку дисперсію (вона стійка до шуму в даних). Складна модель, навпаки, має низьке зміщення, але високу дисперсію — вона реагує на будь-який чих у даних, ніби це важливий сигнал.


Рис. 2.2: Золота середина детектива: занадто проста модель ігнорує факти (високий bias), занадто складна — вигадує їх (висока variance). Істина десь посередині.

На графіку 2.2 показано, як ці дві помилки — через зміщення та через дисперсію — взаємодіють. Коли ми збільшуємо складність моделі, помилка через зміщення (червона лінія) падає, оскільки модель стає гнучкішою і краще відповідає даним. Однак, починаючи з певного моменту, помилка через дисперсію (синя лінія) починає стрімко зростати. Це означає, що модель стає занадто чутливою до шуму в навчальних даних.

Загальна помилка (чорна лінія) є сумою цих двох помилок. Наше завдання — знайти «долину» на цьому графіку, тобто оптимальну складність моделі, де загальна помилка є мінімальною. Це і є точка, де ми досягли найкращого компромісу між точністю опису наявних даних і здатністю до узагальнення для майбутніх, ще не бачених даних. Саме тут знаходиться модель, яка є «достатньо хорошою», не будучи при цьому ні наївно простою, ні параноїдально складною.

Отже, наша перша заповідь: **шукай найпростіше пояснення, яке все ще працює**. Це принцип, відомий як **«лезо Оккама»**. Якщо у вас є дві теорії, які однаково добре пояснюють факти, обирайте ту, що простіша. Чому? Бо прості моделі легше зрозуміти, вони стійкіші і, що найголовніше, краще узагальнюють — тобто краще прогнозують майбутнє.

# Як оцінити «підозрюваного»? Критерії якості

Добре, ми хочемо просту, але точну модель. Але як нам виміряти, наскільки вона «точна»? Нам потрібен спосіб оцінювати наших «підозрюваних» (моделі) об'єктивно. Найпростіший спосіб — подивитися, наскільки сильно наша модель помиляється. Ми беремо реальні дані про вихід системи $y(t)$ і порівнюємо їх з тим, що напрогнозувала наша модель $\hat{y}(t)$. Різниця між ними — це помилка $e(t) = y(t) - \hat{y}(t)$.

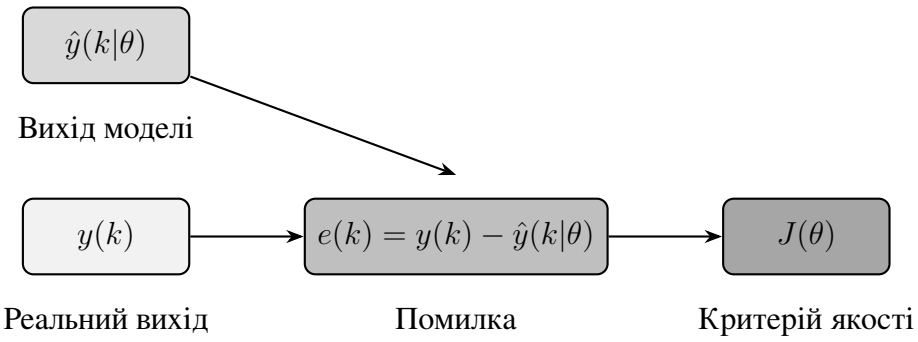


Рис. 2.3: Схема формування критерію якості моделі. Помилка $e(k)$ є різницею між реальним виходом системи $y(k)$ та прогнозом моделі $\hat{y}(k|\theta)$. На основі цієї помилки розраховується критерій якості $J(\theta)$.

Ця схема на рис. 2.3 ілюструє фундаментальну ідею. Ми порівнюємо те, що система зробила насправді ($y(k)$), з тим, що, за нашими розрахунками, вона мала б зробити ($\hat{y}(k|\theta)$). Різниця між цими двома величинами — це помилка, або нев'язка, $e(k)$. Ця помилка є мірою нашого невігластва. Якби наша модель була ідеальною, ця помилка дорівнювала б нулю.

Але одна помилка в один момент часу нам ні про що не говорить. Нам потрібна загальна оцінка якості моделі за весь час спостереження. Для цього ми об'єднуємо всі ці маленькі помилки в один великий критерій якості $J(\theta)$. Найпростіший і найпоширеніший спосіб зробити це — взяти середнє значення квадратів помилок.

Щоб отримати загальну оцінку, ми можемо взяти всі ці помилки, піднести кожну до квадрату (щоб позбутися знаків мінус і сильніше «штрафувати» за великі помилки) і усереднити. Це називається **середньоквадратична помилка (MSE — Mean Squared Error)**.

$$MSE = \frac{1}{N}\sum_{t=1}^{N}(y(t) - \hat{y}(t))^2 \tag{2.1}$$

Здається, все просто: чим менша MSE, тим краща модель. Але тут є пастка! Складна модель, яка вивчила весь шум, матиме дуже низьку MSE на тих даних, на яких ми її «тренували». Це як студент, що визубрив відповіді на екзаменаційні білети, але не зрозумів предмету.

Тому розумні люди придумали хитріші критерії, які враховують не лише помилку, а й складність моделі. Найвідоміші з них — це **інформаційний критерій Акаіке (AIC)** та **Байєсівський інформаційний критерій (BIC)**. Ці підходи, детально описані в класичних роботах з ідентифікації систем [3, 4], працюють за таким принципом:

$$extCriterion = (\text{Model error}) + (\text{Penalty for complexity}) \tag{2.2}$$

$$AIC = N\ln(MSE) + 2p \tag{2.3}$$

$$BIC = N\ln(MSE) + p\ln(N) \tag{2.4}$$

Тут $p$ — це кількість параметрів у вашій моделі (тобто її «складність»). Тепер, якщо модель стає складнішою (зростає $p$), вона отримує «штраф». Це змушує нас шукати баланс: ми хочемо модель, яка добре пояснює дані, але не є надто заплутаною. Це як обирати інструмент: можна забити цвях мікроскопом, але молоток простіший і ефективніший.

# Підготовка допиту: планування експерименту та дані

Найкраща модель у світі буде марною, якщо вона побудована на поганих даних. Це фундаментальний принцип: **«сміття на вході — сміття на виході» (Garbage In, Garbage Out)**. Тому, перш ніж кидатися в бій, детектив повинен ретельно

підготуватися.

## Як «допитувати» систему?

Щоб зрозуміти, як працює чорна скринька, її треба правильно «поштовхати». Сигнал, який ми подаємо на вхід системи, має бути достатньо інформативним.

- **Ступінчаста дія (Step input):** Це найпростіше — просто увімкнути щось і дивитися. Це як один раз натиснути на кнопку. Ми побачимо реакцію, але цього може бути недостатньо, щоб розкрити всі таємниці системи.
- **Псевдовипадкова послідовність (PRBS):** Це набагато цікавіше. Уявіть, що ви хаотично, але за певним планом, клацаєте перемикачем між «увімкнено» і «вимкнено». Такий сигнал «торсає» систему на різних частотах і змушує її розкрити набагато більше своїх секретів. Це найпопулярніший інструмент для «допиту» систем.


Рис. 2.4: PRBS-сигнал, що перемикається між двома рівнями ($A$ та $-A$) у псевдовипадковому порядку. Такий сигнал забезпечує хороше збудження системи у широкому діапазоні частот.

Правильно спланований експеримент є ключем до успішної ідентифікації. Якщо ми хочемо зрозуміти, як система реагує на різні впливи, ми повинні подати на її вхід сигнал, який є достатньо «багатим» і «збуджуючим». Простий ступінчастий сигнал може розкрити лише базові характеристики, такі як час наростання та статичне підсилення. Однак для повного розуміння динаміки системи цього недостатньо.

Сигнал типу псевдовипадкової бінарної послідовності (PRBS), показаний на рис. 2.4, є набагато кращим інструментом для «допиту» системи. Він випадково перемикається між двома рівнями, що дозволяє збуджувати систему в широкому діапазоні частот. Це схоже на те, як лікар стукає молоточком по різних частинах тіла, щоб перевірити рефлекси. Кожен «удар» PRBS-сигналу дає нам нову інформацію про внутрішню будову нашої «чорної скриньки».

### Прибирання на місці злочину: обробка даних

Коли ми зібрали дані, вони рідко бувають ідеальними. Перш ніж будувати модель, їх треба «причесати».

- **Подивіться на дані!** Перший крок — завжди будувати графіки. Це допоможе побачити дивні речі: тренди, викиди, періодичні коливання.
- **Видалення тренду:** Якщо температура в кімнаті повільно зростала під час експерименту, це може вплинути на вимірювання. Цей повільний дрейф, або **тренд**, не є частиною динаміки системи, і його треба видалити.
- **Обробка викидів (outliers):** Іноді в даних з'являються абсолютно дикі значення — наприклад, через збій датчика. Такі «аномалії» можуть сильно спотворити нашу модель. Їх потрібно знайти і акуратно обробити (видалити або замінити).
- **Фільтрація:** Якщо дані дуже «шумні», ми можемо трохи згладити їх за допомогою фільтра, щоб краще побачити основний сигнал. Але тут треба бути обережним, щоб разом із шумом не видалити і корисну інформацію.

## Туман війни: що робити з шумом?

У реальному світі наші вимірювання ніколи не бувають ідеальними. Завжди є якийсь «туман» — випадкові коливання, перешкоди, неточності. Ми називаємо це **шумом**. Ігнорувати його — все одно що вести розслідування в тумані, вдаючи, що його не існує.

Найпростіший вид шуму — це **«білий шум»**. Уявіть собі тихий шиплячий звук радіо, коли воно не налаштоване на жодну станцію. Цей шум абсолютно

випадковий і непередбачуваний. Його значення в будь-який момент часу не залежить від попередніх. Це найпростіший випадок, і багато методів ідентифікації розраховані саме на нього.

Але часто шум буває хитрішим. Наприклад, він може бути **корельованим (кольоровим)**. Це якби в радіоефірі, крім шипіння, був присутній тихий, монотонний гул. Цей гул має структуру. Його значення зараз залежить від того, яким воно було секунду тому. Такий шум може виникати через інерційність датчиків, повільні зміни температури в лабораторії або періодичні перешкоди від електромережі.

Якщо ми проігноруємо такий «кольоровий» шум і будемо вважати його білим, наша модель може почати «вбирати» в себе структуру цього шуму, вважаючи його частиною поведінки системи. Це призведе до спотворених, **зміщених** оцінок параметрів. Щоб цього уникнути, нам потрібно моделювати не лише саму систему, а й шум! Це призводить до більш просунутих моделей (ARMAX, Box-Jenkins), які мають окремий блок для опису «туману війни».

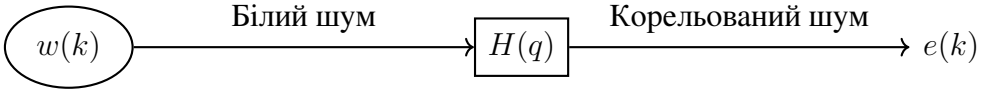


Рис. 2.5: Модель корельованого шуму. Випадковий білий шум $w(k)$ проходить через деякий уявний фільтр $H(q)$ і перетворюється на корельований (кольоровий) шум $e(k)$, який і впливає на наші вимірювання.

Нарешті, ми повинні поговорити про «туман війни» — шум. Жодні реальні вимірювання не є ідеальними. Завжди присутні випадкові флуктуації, які не є частиною поведінки самої системи. Найпростіший тип шуму — це «білий шум», який є абсолютно непередбачуваним. Однак у багатьох випадках шум має певну структуру, тобто є корельованим або «кольоровим».

На рис. 2.5 показано, як можна уявити собі такий корельований шум. Ми можемо думати про нього як про результат проходження ідеального білого шуму $w(k)$ через деякий невідомий нам фільтр $H(q)$. Цей фільтр і надає шуму його «колір» та структуру. Ігнорування цієї структури може призвести до систематичних помилок в оцінках параметрів моделі, оскільки алгоритм може помилково прийняти властивості шуму за властивості самої системи. Тому для отримання точних результатів часто необхідно моделювати не тільки динаміку системи, але і властивості шуму, що на неї діє.

# Висновки: Заповіді хорошого детектива

1. **Не обманюй себе:** Завжди пам'ятай про ризик перенавчання. Найскладніша теорія — не завжди найкраща. Шукай баланс між простотою і точністю. **(Лезо Оккама)**

2. **Використовуй об'єктивні критерії:** Не довіряй лише інтуїції. Використовуй критерії, як-от AIC або BIC, щоб порівнювати моделі, враховуючи їхню складність.

3. **Поважай шум:** Шум — це не просто перешкода, це частина реальності. Зрозумій його природу, і ти отримаєш набагато кращі результати.

4. **Дані — це все:** Якість твоєї моделі ніколи не перевищить якості твоїх даних. Ретельно плануй експеримент і готуй дані.

   І наостанок, пам'ятайте знамениту цитату статистика Джорджа Бокса:

   «Всі моделі неправильні, але деякі з них корисні» [7].

Наша мета — не знайти абсолютну істину, а створити корисну карту реальності. І мистецтво полягає в тому, щоб не заблукати, слідуючи цій карті.

# Контрольні запитання

1. Поясніть на прикладі «детектива» різницю між недонавчанням (underfitting) та перенавчанням (overfitting).

2. Що таке компроміс між зміщенням та дисперсією (Bias-Variance Tradeoff)? Як він пов'язаний зі складністю моделі?

3. Сформулюйте принцип «леза Оккама» в контексті ідентифікації систем. Чому простіші моделі часто є кращими?

4. Як розраховується середньоквадратична помилка (MSE)? Чому мінімізація лише MSE може призвести до перенавчання?

5. Як інформаційні критерії, такі як AIC та BIC, допомагають уникнути перенавчання? У чому їхня основна ідея?

6. Чому планування експерименту є критично важливим для ідентифікації? Порівняйте інформативність ступінчастої дії та PRBS-сигналу.

7. Назвіть та коротко опишіть три основні етапи попередньої обробки даних перед ідентифікацією.

8. Що таке «сміття на вході — сміття на виході» (Garbage In, Garbage Out) і як цей принцип стосується збору даних?

9. У чому різниця між «білим» та «кольоровим» (корельованим) шумом?

10. Чому ігнорування корельованого шуму може призвести до зміщених (неправильних) оцінок параметрів моделі?

11. Поясніть відомий вислів Джорджа Бокса: «Всі моделі неправильні, але деякі з них корисні».

12. Які чотири «заповіді хорошого детектива» можна вивести з принципів ідентифікації систем?

# Розділ 3

# Моделі динамічних систем

**Від класичних моделей до відкриття законів: Інструментарій ідентифікації**

## Зоопарк моделей: Як класифікувати те, що ми бачимо?

Перш ніж почати полювання, мисливець повинен знати, на якого звіра він полює. Чи це швидкий і непередбачуваний гепард, чи повільний, але могутній слон? У світі ідентифікації систем нашим «звіром» є модель, і їх існує цілий зоопарк. Нерозуміння їхніх відмінностей — це прямий шлях до того, щоб намагатися зловити метелика сіткою для ведмедів. Давайте наведемо лад у цьому звіринці.

Моделі можна класифікувати за кількома ключовими ознаками, як показано на рис. 3.1.

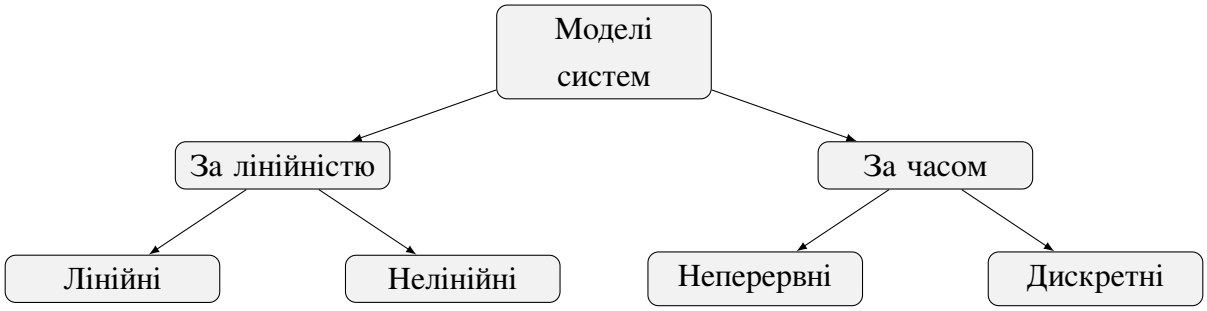


Рис. 3.1: Основні осі класифікації динамічних моделей.

- **Лінійні vs. Нелінійні.** Це, мабуть, найважливіший поділ. **Лінійна модель** — це світ, де все просто і пропорційно. **Нелінійні моделі** — це реальний

світ у всій його складності.

- **Неперервний час vs. Дискретний час.** Оскільки наші комп'ютери працюють у дискретному часі, більшість сучасної ідентифікації відбувається саме в цьому світі.
- **Стаціонарні vs. Нестаціонарні.** Чи залишаються закони, що керують системою, незмінними в часі?
- **Параметричні vs. Непараметричні.** У **параметричних** методах ми заздалегідь обираємо структуру моделі. У **непараметричних** — не робимо жорстких припущень.

У цьому курсі ми почнемо з **лінійних стаціонарних (LTI) дискретних моделей**, бо це «золотий стандарт», що пропонує найкращий компроміс між простотою та адекватністю для багатьох задач.

# Робоча конячка ідентифікації: Модель ARX

Серед усього розмаїття лінійних моделей є одна, що стоїть окремо завдяки своїй простоті та фундаментальності. Це модель **ARX** (AutoRegressive with eXternal input).

Математично це записується так:

$$y(t) + a_1 y(t-1) + \cdots + a_{n_a} y(t-n_a) = \begin{array}{l} b_1 u(t-n_k) + \dots \\ + b_{n_b} u(t-n_k-n_b+1) + e(t) \end{array} \tag{3.1}$$

де $a_i, b_j$ — параметри, $n_a, n_b$ — порядки моделі, $n_k$ — запізнення, а $e(t)$ — білий шум. Використовуючи оператор зсуву назад $q^{-1}$, це можна записати компактно:

$$A(q)y(t) = B(q)u(t-n_k) + e(t) \tag{3.2}$$

Вся магія ARX полягає в тому, що задача знаходження параметрів $\theta = [a_1, \dots, b_1, \dots]^T$ зводиться до простої лінійної регресії $\hat{y}(t|\theta) = \varphi^T(t)\theta$, яка має швидкий та унікальний розв'язок методом найменших квадратів.

- **Переваги:** Простота, швидкість, гарантований глобальний оптимум.

- **Недоліки:** Головний недолік — припущення про білий шум. Якщо реальний шум корельований, модель ARX дає **зміщені оцінки**, оскільки намагається одним і тим же знаменником $A(q)$ описати і динаміку системи, і структуру шуму.

Але що робити, якщо система складна, нелінійна, або ми не хочемо обмежувати себе структурою ARX? Тут починається найцікавіше.

# Полювання за законами природи: як змусити дані говорити?

У попередній бесіді ми були детективами, що обирають між готовими версіями злочину. Тепер ми підемо далі. Замість того, щоб обирати з кількох підозрюваних, ми змусимо самі докази — дані — розповісти нам, що сталося.

Уявіть собі величезну **«бібліотеку» всіх можливих фізичних законів**. Це нескінченні стелажі з математичними термінами: поліноми $(x, x^2, \dots)$, тригонометричні функції $(\sin(x), \cos(x), \dots)$, тощо. Ідея, що лежить в основі сучасних методів, таких як SINDy (Sparse Identification of Nonlinear Dynamics) [8], полягає в тому, що **закони природи, як правило, прості (рідкісні)**. Вони використовують лише декілька «слів» з цієї величезної бібліотеки.

Наша стратегія:

1. **Створюємо «словник» кандидатів.** Ми беремо наші дані $x(t)$ і будуємо з них матрицю $\boldsymbol{\Theta}(\mathbf{x})$, де кожен стовпець — це один із можливих термінів з нашої бібліотеки $(x, x^2, \sin(x), \dots)$.

2. **Шукаємо «найрідкісніше» пояснення.** Ми шукаємо таку комбінацію стовпців, яка найкраще описує похідну $\dot{\mathbf{x}}$, але використовує **якомога менше** термінів.

Це зводиться до розв'язання рівняння $\dot{\mathbf{x}} = \boldsymbol{\Theta}(\mathbf{x})\boldsymbol{\Xi}$, де $\boldsymbol{\Xi}$ — вектор коефіцієнтів. Але ми шукаємо не будь-який розв'язок, а **рідкісний** (sparse), де більшість коефіцієнтів дорівнюють нулю. Для цього використовують методи **рідкісної регресії**, такі як послідовне породове відсікання або LASSO.

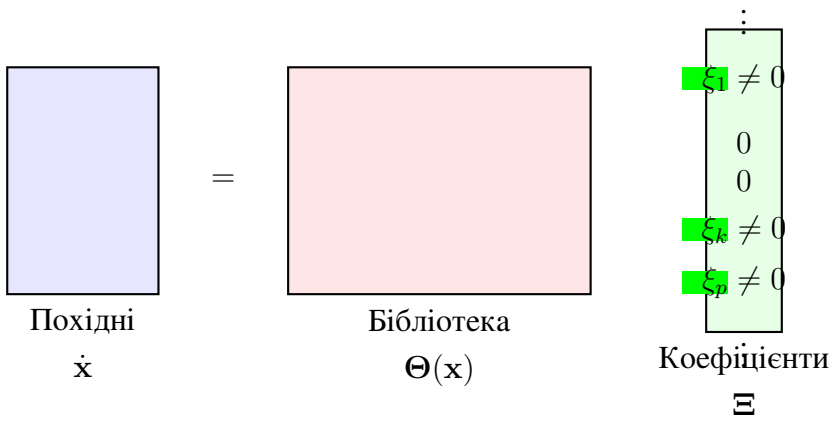


Рис. 3.2: Схематичне представлення рівняння SINDy. Ми шукаємо рідкісний вектор коефіцієнтів $\boldsymbol{\Xi}$.

## Мистецтво бачити головне: зменшення розмірності

А що робити, якщо даних дуже багато, як при описі руху косяка риб? Відстежувати кожну рибу — безглуздо. Потрібно знайти головні патерни руху. Це і є ідея **зменшення розмірності**.

Найпотужнішим інструментом для цього є **сингулярний розклад матриць (SVD)**. SVD — це як рентген для даних. Він розкладає матрицю даних $\mathbf{X}$ на три: $\mathbf{X} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$.

- $\mathbf{U}$ містить просторові патерни (моди).
- $\boldsymbol{\Sigma}$ — діагональна матриця з сингулярними числами, що показують «важливість» кожної моди.
- $\mathbf{V}$ описує, як ці моди змінюються в часі.

Дуже часто виявляється, що лише кілька перших сингулярних чисел є великими, а решта — крихітні. Це означає, що вся складна динаміка може бути описана лише кількома головними модами. Ми можемо відкинути решту (шум) і працювати у просторі значно меншої розмірності.

Отже, наш сучасний робочий процес:

1. Збираємо дані зі складної системи.
2. Застосовуємо SVD, щоб знайти прихований простір низької розмірності.

3. Проектуємо дані на цей простий підпростір.

4. **Тільки тепер** застосовуємо SINDy, щоб знайти рівняння в цих нових, спрощених координатах.

# Висновки: Від підгонки до відкриття

Ми пройшли шлях від класичних методів до сучасних підходів, що змінюють правила гри.

1. **ARX — надійний початок.** Для багатьох лінійних систем модель ARX є чудовою відправною точкою завдяки своїй простоті та швидкості.

2. **SINDy — крок до відкриття.** Коли ми стикаємося з нелінійністю або не хочемо обмежувати себе фіксованою структурою, методи на кшталт SINDy дозволяють нам не просто підганяти параметри, а відкривати саму структуру рівнянь з даних.

3. **SVD — наш мікроскоп.** Для складних, багатовимірних систем SVD дозволяє відфільтрувати шум і побачити приховану низькорозмірну динаміку, роблячи нерозв'язні задачі доступними для аналізу.

Це і є сучасне мистецтво ідентифікації — володіти як класичними інструментами для стандартних задач, так і потужними новими методами для дослідження незвіданих територій складних систем.

# Контрольні запитання

1. Назвіть чотири основні осі класифікації динамічних моделей.

2. Поясніть структуру моделі ARX. Що означають компоненти AR та X у її назві?

3. Яка головна перевага моделі ARX, що робить її «робочою конячкою» ідентифікації?

4. У якій ситуації модель ARX дає зміщені (некоректні) оцінки параметрів і чому?

5. Яка фундаментальна ідея лежить в основі методу SINDy? Поясніть метафору «бібліотеки законів» та принципу «рідкісності».

6. Чим підхід SINDy до моделювання відрізняється від класичних параметричних методів, як-от ARX?

7. Для чого використовується метод сингулярного розкладу матриць (SVD) при аналізі складних систем?

8. Що представляють собою матриці $\mathbf{U}$, $\mathbf{\Sigma}$ та $\mathbf{V}^T$ у розкладі SVD? Який фізичний сенс мають сингулярні числа?

9. Як за допомогою сингулярних чисел можна відокремити значущу динаміку системи від шуму?

10. Опишіть сучасний підхід до ідентифікації нелінійних систем, що поєднує SVD та SINDy. З яких кроків він складається?

11. Порівняйте SINDy та ARX: який метод ви б обрали для простої лінійної системи, а який — для дослідження невідомої нелінійної динаміки?

12. Поясніть перехід від «підгонки параметрів» до «відкриття законів» у сучасній ідентифікації систем.

# Розділ 4

# Непараметрична ідентифікація

**Полювання без карти: Як знайти структуру в хаосі даних**

## Коли структура невідома: Навігація за даними

У попередній лекції ми озброїлися моделлю ARX — надійною, як швейцарський ніж, але все ж таки обмеженою у своїх можливостях. Ми припускали, що знаємо структуру системи, і нам залишалося лише знайти параметри. Але що робити, коли ми опинилися в диких джунглях, де правила невідомі? Що, якщо система нелінійна, її порядок незрозумілий, а шум поводиться не так, як ми очікували?

Намагатися підігнати просту модель під складну систему — це все одно, що намагатися пояснити квантову фізику за допомогою законів Ньютона. Результат буде не просто неточним, а фундаментально хибним. Тут на сцену виходить **непараметрична ідентифікація** — мистецтво слухати дані та дозволяти їм самим розповісти свою історію, не нав'язуючи їм наших упереджень. Це не пошук конкретних параметрів, а скоріше розвідка, що має на меті отримати карту місцевості перед тим, як планувати маршрут.

Основна ідея полягає в тому, щоб отримати візуальні, графічні характеристики системи, які допоможуть нам зрозуміти її ключові властивості без необхідності записувати єдине рівняння.

# Перший погляд: Аналіз перехідної характеристики

Найпростіший спосіб “пнути” систему і подивитися, що станеться — це подати на неї **одиничний ступінчастий сигнал** (step input). Це еквівалентно тому, щоб раптово увімкнути рубильник. Реакція системи на такий вплив називається **перехідною характеристикою** (step response). Це один із найпотужніших інструментів експрес-діагностики.

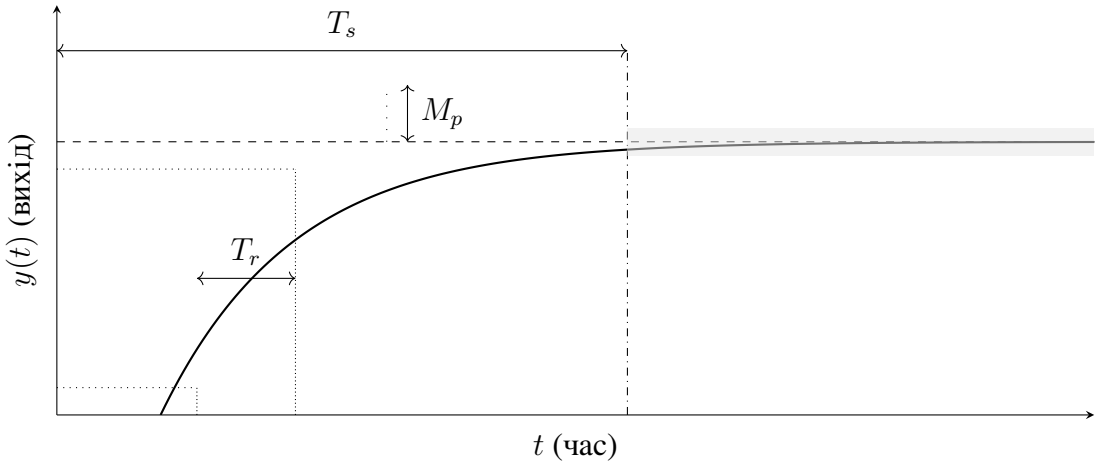


Рис. 4.1: Аналіз типової перехідної характеристики.

Дивлячись на графік перехідної характеристики (рис. 4.1), ми можемо миттєво оцінити:

- **Статичний коефіцієнт передачі ($K$):** Це усталене значення виходу. Він показує, наскільки сильно вихід реагує на вхід у довгостроковій перспективі.
- **Час запізнення ($\tau_d$):** Скільки часу проходить, перш ніж система взагалі почне реагувати.
- **Час наростання ($T_r$):** Як швидко система рухається від 10% до 90% свого фінального значення. Це показник її жвавості.
- **Час усталення ($T_s$):** Час, після якого вихід залишається в межах невеликого відсотка (зазвичай 2-5%) від свого фінального значення.
- **Перерегулювання ($M_p$):** Наскільки сильно вихід "проскакує"фінальне значення. Велике перерегулювання часто вказує на коливальну природу системи та потенційну близькість до нестабільності.

Аналіз перехідної характеристики — це дешево, швидко і неймовірно інформативно. Він не дасть нам точної моделі, але допоможе відповісти на ключові питання: система швидка чи повільна? Коливальна чи аперіодична? Чи є значне запізнення? Це перший крок до розуміння її "характеру".

# Пошук зв'язків у часі: Кореляційний аналіз

Якщо перехідна характеристика — це активний експеримент, то **кореляційний аналіз** — це пасивне спостереження. Ми просто слухаємо вхідні та вихідні сигнали і намагаємося знайти в них приховані закономірності.

**Взаємна кореляційна функція (ВКФ)** між входом $u(t)$ та виходом $y(t)$ показує, наскільки вони схожі, якщо зсунути один відносно одного на час $\tau$.

$$R_{uy}(\tau) = \mathbb{E}[u(t)y(t+\tau)] \tag{4.1}$$

Ключова ідея полягає в тому, що якщо система має затримку $\tau_d$, то вихід буде максимально схожий на вхід, що був $\tau_d$ секунд тому. Тому пік ВКФ часто вказує на час затримки в системі.

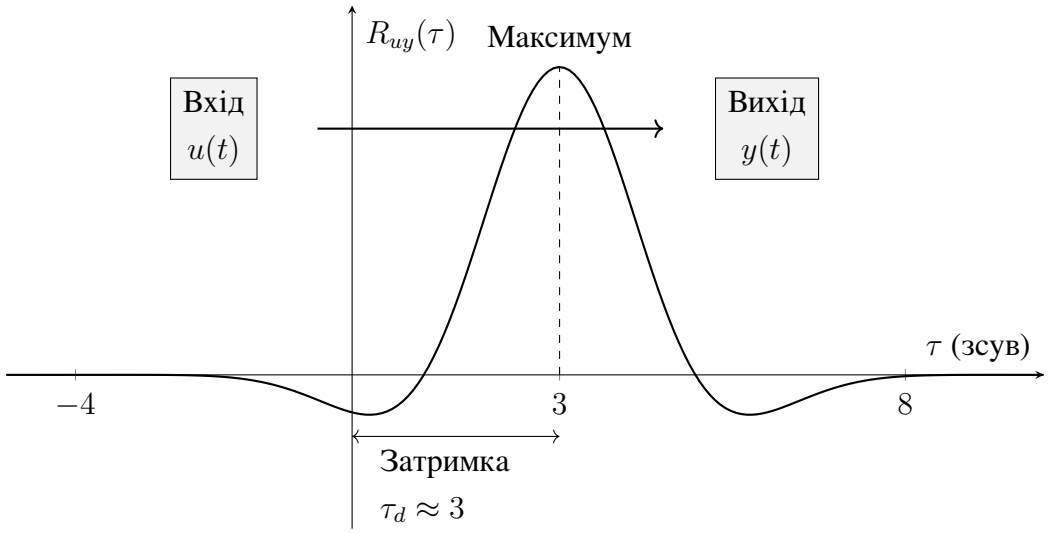


Рис. 4.2: Використання взаємної кореляційної функції для оцінки затримки.

Крім того, форма ВКФ може багато розповісти про динаміку системи. Якщо вона швидко згасає, система має коротку "пам'ять". Якщо вона коливається, це вказує на резонансні частоти.

# Погляд крізь призму частот: Спектральний аналіз

Часовий аналіз — це лише один бік медалі. Інший, не менш важливий — це **частотний аналіз**. Ідея полягає в тому, щоб розкласти складні сигнали на суму простих синусоїд і подивитися, як система реагує на кожну з них. Це дозволяє отримати **амплітудну та фазову частотні характеристики (АЧХ та ФЧХ)**.

- **АЧХ** показує, як система підсилює або послаблює коливання на різних частотах. Наприклад, автомобільна підвіска повинна добре послаблювати швидкі коливання (від нерівностей дороги), але пропускати повільні (від поворотів).
- **ФЧХ** показує, наскільки вихідний сигнал запізнюється відносно вхідного на кожній частоті.

Ці характеристики можна отримати за допомогою **швидкого перетворення Фур'є (FFT)** для вхідного та вихідного сигналів:

$$\hat{G}(j\omega) = \frac{Y(j\omega)}{U(j\omega)} \tag{4.2}$$

де $Y(j\omega)$ та $U(j\omega)$ — це перетворення Фур'є від виходу та входу відповідно.

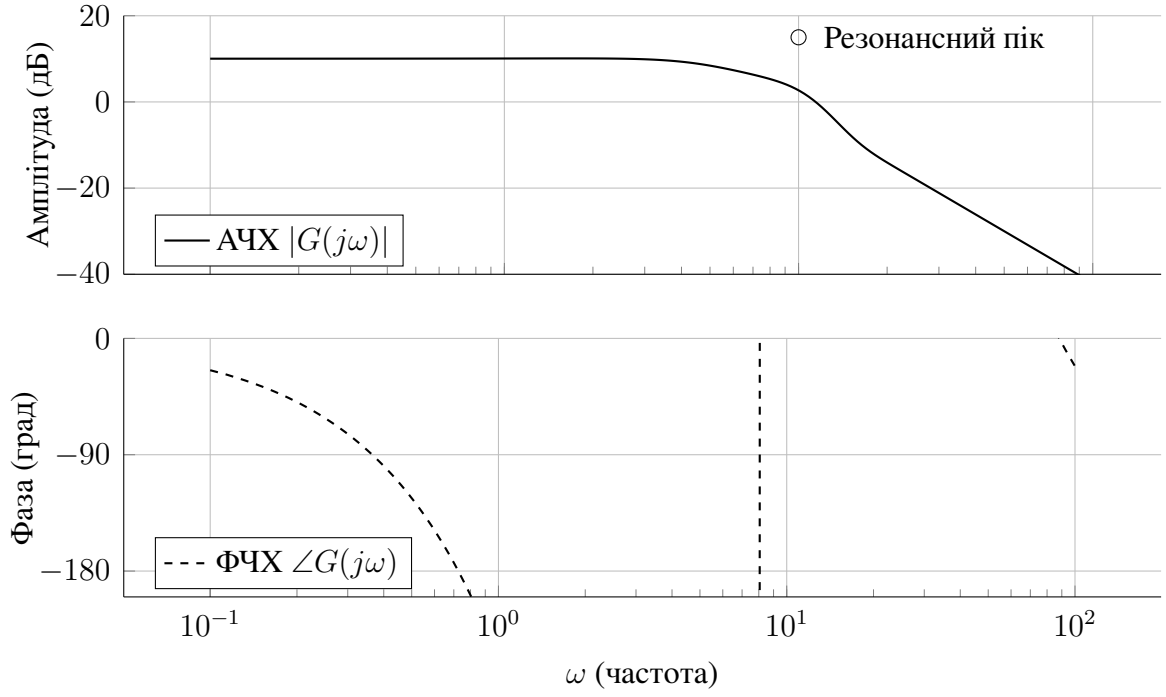


Рис. 4.3: Приклад діаграми Боде (АЧХ та ФЧХ), що показує резонансний пік.

Спектральний аналіз особливо корисний для:

- **Виявлення резонансів:** Піки на АЧХ вказують на частоти, на яких система особливо "любить"коливатися.
- **Визначення смуги пропускання:** Діапазон частот, у якому система ефективно працює.
- **Оцінки порядку системи:** Швидкість спаду АЧХ на високих частотах часто пов'язана з відносним порядком системи.

# Відкриття законів з даних: Sparse Identification of Nonlinear Dynamics (SINDy)

Непараметричні методи дають нам чудову картину "що відбувається але не "чому". Вони не дають нам рівнянь. А що, якби ми могли змусити дані самих написати ці рівняння?

Це здається фантастикою, але саме цю ідею реалізує метод **SINDy (Sparse Identification of Nonlinear Dynamics)**, запропонований Стівеном Брантоном та Нейтаном Кутцем. Це революційний підхід, що поєднує ідеї машинного навчання та класичної динаміки.

Ідея геніальна у своїй простоті. Припустимо, ми маємо систему, що описується диференціальним рівнянням:

$$\dot{\mathbf{x}} = f(\mathbf{x}) \tag{4.3}$$

Ми не знаємо, що таке $f(\mathbf{x})$, але ми можемо виміряти стан системи $\mathbf{x}(t)$ та його похідну $\dot{\mathbf{x}}(t)$ (наприклад, чисельно).

Далі ми робимо сміливе припущення: функція $f(\mathbf{x})$ є **розрідженою** (sparse) у бібліотеці можливих функцій-кандидатів. Тобто, вона складається з дуже невеликої кількості елементів. Наприклад, рівняння маятника $\ddot{\theta} = -\frac{g}{L}\sin(\theta)$ є розрідженим у бібліотеці $\{\theta, \theta^2, \theta^3, \sin(\theta), \cos(\theta), \dots\}$.

Алгоритм SINDy виглядає так:

1. **Збір даних:** Вимірюємо часові ряди стану системи $\mathbf{x}(t)$ та обчислюємо похідні $\dot{\mathbf{x}}(t)$.

2. **Побудова бібліотеки:** Створюємо величезну матрицю $\mathbf{\Theta}(\mathbf{x})$ з функцій-кандидатів. Це можуть бути поліноми, тригонометричні функції, або будь-що інше, що підказує нам фізична інтуїція. Наприклад, для системи з двома змінними $x_1, x_2$, бібліотека може виглядати так:

$$\mathbf{\Theta}(\mathbf{x}) = \begin{bmatrix} 1 & x_1 & x_2 & x_1^2 & x_1x_2 & x_2^2 & \sin(x_1) & \dots \end{bmatrix} \tag{4.4}$$

3. **Розв'язання регресії:** Тепер наша задача зводиться до лінійної системи:

$$\dot{\mathbf{X}} = \mathbf{\Theta}(\mathbf{X})\mathbf{\Xi} \tag{4.5}$$

де $\mathbf{\Xi}$ — матриця коефіцієнтів, що визначають, які члени бібліотеки входять до рівняння.

4. **Магія розрідження:** Замість звичайного методу найменших квадратів, ми використовуємо **розріджену регресію** (наприклад, LASSO), яка "обнуляє"більшість коефіцієнтів у $\mathbf{\Xi}$, залишаючи лише найважливіші.

В результаті ми отримуємо не просто "чорну скриньку а просту, інтерпретовану модель у вигляді диференціального рівняння, яке можна аналізувати, узагальнювати та використовувати для передбачення. SINDy автоматично знаходить баланс між точністю та складністю, відкриваючи закони, що керують системою, прямо з даних.

Це потужний зсув парадигми: від ручного моделювання, заснованого на перших принципах, до автоматизованого відкриття моделей, керованого даними. І це саме те, що потрібно для навігації у світі складних систем, де наші інтуїтивні уявлення часто виявляються безсилими.

# Контрольні запитання

1. У чому полягає головна відмінність непараметричної ідентифікації від параметричної? В якій ситуації доцільно використовувати непараметричні методи?

2. Що таке перехідна характеристика системи і як її експериментально отримати?

3. Назвіть та поясніть щонайменше чотири ключові показники, які можна визначити з аналізу перехідної характеристики (наприклад, час наростання, перерегулювання).

4. Для чого використовується кореляційний аналіз в ідентифікації систем? Що може показати пік взаємної кореляційної функції між входом та виходом?

5. Що таке амплітудна (АЧХ) та фазова (ФЧХ) частотні характеристики? Яку інформацію про поведінку системи вони надають?

6. Як за допомогою спектрального аналізу (діаграми Боде) можна виявити резонанси в системі та визначити її смугу пропускання?

7. Яка основна ідея лежить в основі методу SINDy? Поясніть поняття «розрідженості» (sparsity) в контексті диференціальних рівнянь.

8. Опишіть основні кроки алгоритму SINDy. Яку роль у ньому відіграє «бібліотека функцій-кандидатів»?

9. Чим розв'язання задачі регресії в SINDy (з використанням, наприклад, LASSO) відрізняється від класичного методу найменших квадратів і чому це принципово важливо?

10. Який зсув парадигми в моделюванні пропонує SINDy порівняно з традиційними підходами, заснованими на перших принципах?

11. Порівняйте інформативність аналізу перехідної характеристики та спектрального аналізу. Чи доповнюють вони один одного?

12. Уявіть, що ви отримали дані з невідомої системи. В якій послідовності ви б застосували непараметричні методи, описані в лекції, для її дослідження?

# Розділ 5

# Параметрична ідентифікація лінійних систем

**Побудова моделей за допомогою регресії та вибору**

## Від феноменології до механізму: Пошук фундаментальних законів

У попередній лекції ми діяли як натуралісти, спостерігаючи за поведінкою системи здалеку. Ми каталогізували її реакції, вимірювали кореляції та аналізували частотний спектр. Це був етап **феноменології** — опису явищ. Ми отримали відповідь на питання "що відбувається?". Тепер наше завдання — здійснити стрибок від опису до пояснення, від феноменології до **механізму**. Ми прагнемо відповісти на питання "чому це відбувається так, а не інакше?".

Цей перехід вимагає від нас побудови **параметричної моделі** — математичного апарату, що інкапсулює гіпотезу про внутрішню структуру системи. Якщо непараметричні методи давали нам карту місцевості, то параметрична модель — це закони фізики, що керують рухом на цій місцевості. Ми переходимо від споглядання до конструювання.

Наріжним каменем цього процесу є **регресія** — потужний математичний інструмент для знаходження залежностей у даних. Ідея, що лежить в основі методу SINDy, полягає в тому, що динаміку будь-якої системи, хоч би якою складною вона була, можна представити як лінійну комбінацію невеликої кількості базисних функцій. Це припущення про **розрідженість** (sparsity) є ключовим.

Наше завдання — не просто знайти параметри, а знайти найпростішу, найекономнішу модель, що адекватно описує дані. Це є втіленням принципу **бритви Оккама**: "Не слід множити сутності без необхідності".

# Лінійна регресія: Абетка побудови моделей

Розглянемо фундаментальну задачу. Ми маємо набір даних, що складається з вхідних векторів (регресорів) $\mathbf{x}_k \in \mathbb{R}^n$ та відповідних їм вихідних значень (відгуків) $y_k \in \mathbb{R}$. Ми хочемо знайти таку модель, яка б найкращим чином пов'язувала входи з виходом. Найпростіша гіпотеза — це лінійна залежність:

$$y_k \approx \beta_1 x_{k1} + \beta_2 x_{k2} + \cdots + \beta_n x_{kn} \tag{5.1}$$

де $\beta_j$ — невідомі параметри, що визначають "вагу"кожного вхідного фактора.

Для всього набору даних, що складається з $m$ спостережень, цю систему можна записати у матричній формі, яка є мовою сучасної науки про дані:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon} \tag{5.2}$$

де:

- $\mathbf{y} \in \mathbb{R}^m$ — вектор-стовпець виміряних виходів.
- $\mathbf{X} \in \mathbb{R}^{m \times n}$ — матриця регресорів (або матриця даних), де кожен рядок — це окремий експеримент, а кожен стовпець — окремий фактор (регресор).
- $\boldsymbol{\beta} \in \mathbb{R}^n$ — вектор невідомих параметрів, які ми прагнемо знайти.
- $\boldsymbol{\epsilon} \in \mathbb{R}^m$ — вектор завад або похибок моделі, що відображає все, що наша модель не враховує.

Це рівняння є центральним. У контексті ідентифікації систем, матриця $\mathbf{X}$ може містити минулі значення входів та виходів (як у моделі ARX), а у контексті SINDy — значення нелінійних функцій-кандидатів. Задача залишається тією ж: знайти вектор параметрів $\boldsymbol{\beta}$.

## Метод найменших квадратів (МНК)

Як визначити "найкращий"вектор $\boldsymbol{\beta}$? Класичний підхід, запропонований Гауссом, полягає у мінімізації енергії похибки, тобто суми квадратів відхилень прогнозів моделі від реальних даних. Ми шукаємо такий $\boldsymbol{\beta}$, що мінімізує норму-2 вектора похибок:

$$J(\boldsymbol{\beta}) = ||\boldsymbol{\epsilon}||_2^2 = ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||_2^2 \to \min_{\boldsymbol{\beta}} \tag{5.3}$$

Це задача опуклої оптимізації, і її розв'язок, на щастя, має елегантну аналітичну форму. Щоб його знайти, потрібно взяти градієнт функціоналу $J$ по $\boldsymbol{\beta}$ і прирівняти його до нуля:

$$\nabla_{\boldsymbol{\beta}} J = -2\mathbf{X}^T(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) = 0 \tag{5.4}$$

Звідси ми отримуємо систему так званих **нормальних рівнянь**:

$$(\mathbf{X}^T\mathbf{X})\boldsymbol{\beta} = \mathbf{X}^T\mathbf{y} \tag{5.5}$$

Якщо матриця $\mathbf{X}^T\mathbf{X}$ є невиродженою (тобто її стовпці лінійно незалежні), то існує єдиний розв'язок, який називається оцінкою МНК:

$$\hat{\boldsymbol{\beta}}_{LS} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \tag{5.6}$$

Вираз $\mathbf{X}^\dagger = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$ називається **псевдооберненою матрицею Мура-Пенроуза**. Геометрично, розв'язок МНК — це проєкція вектора даних $\mathbf{y}$ на простір, що натягнутий на стовпці матриці $\mathbf{X}$.

## Обмеження та виклики

Класичний МНК є потужним, але не всемогутнім. Його ефективність залежить від низки припущень, які в реальному світі часто порушуються:

1. **Погано обумовлені матриці.** Якщо стовпці матриці $\mathbf{X}$ є майже лінійно залежними (мультиколінеарність), матриця $\mathbf{X}^T\mathbf{X}$ стає погано обумовленою. Це означає, що навіть невеликі шуми в даних $\mathbf{y}$ можуть призвести до величезних змін в оцінці $\hat{\boldsymbol{\beta}}$. Модель стає надзвичайно чутливою до шуму.

2. **Надмірна кількість регресорів ($n > m$).** Якщо кількість потенційних ре-

гресорів (наприклад, функцій-кандидатів у SINDy) перевищує кількість спостережень, система (5.2) стає недовизначеною. Існує нескінченна кількість точних розв'язків, і МНК не може обрати один з них.

3. **Відсутність розрідженості.** МНК знаходить "щільний"розв'язок, де майже всі коефіцієнти $\beta_j$ є ненульовими. Це суперечить нашому бажанню знайти просту, інтерпретовану модель. Ми отримуємо точність за рахунок складності та втрачаємо здатність до узагальнення.

Ці проблеми особливо гостро постають при ідентифікації складних нелінійних систем, де ми створюємо величезні бібліотеки функцій-кандидатів. Нам потрібні інструменти, що можуть впоратися з цими викликами.

# Регуляризація: Введення апріорних знань для боротьби з перенавчанням

Ідея **регуляризації** полягає у модифікації критерію оптимізації, додаючи до нього штрафний член, який обмежує "складність"моделі. Замість того, щоб мінімізувати лише помилку, ми мінімізуємо комбінацію помилки та штрафу:

$$J_{reg}(\boldsymbol{\beta}) = ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||_2^2 + \lambda P(\boldsymbol{\beta}) \tag{5.7}$$

де $P(\boldsymbol{\beta})$ — це штрафна функція (регуляризатор), а $\lambda \geq 0$ — гіперпараметр, що контролює баланс між точністю на даних та складністю моделі. Вибір $\lambda$ — це мистецтво, що вимагає компромісу.

## Гребенева регресія (Ridge, L2-регуляризація)

Один з найпоширеніших підходів — це **гребенева регресія**, яка використовує L2-норму вектора параметрів як штраф:

$$J_{ridge}(\boldsymbol{\beta}) = ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||_2^2 + \lambda||\boldsymbol{\beta}||_2^2 \tag{5.8}$$

Цей штраф не любить великих значень коефіцієнтів $\beta_j$. Він "стягує"їх до нуля. Розв'язок цієї задачі також має аналітичну форму:

$$\hat{\boldsymbol{\beta}}_{ridge} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y} \tag{5.9}$$

Додавання діагонального члена $\lambda\mathbf{I}$ робить матрицю гарантовано оберненою, навіть якщо $\mathbf{X}^T\mathbf{X}$ була виродженою. Це вирішує проблему поганої обумовленості. Однак, гребенева регресія не обнуляє коефіцієнти, а лише зменшує їх. Вона не сприяє розрідженості.

## LASSO (L1-регуляризація)

Щоб отримати розріджені моделі, нам потрібен інший тип штрафу. **LASSO (Least Absolute Shrinkage and Selection Operator)** використовує L1-норму:

$$J_{LASSO}(\boldsymbol{\beta}) = ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||_2^2 + \lambda||\boldsymbol{\beta}||_1 \tag{5.10}$$

де $||\boldsymbol{\beta}||_1 = \sum_j |\beta_j|$. Через наявність модуля, L1-норма не є диференційовною в нулі, що створює "гострі кути"в поверхні вартості. Саме ці кути і змушують оптимізатор обнуляти багато коефіцієнтів $\beta_j$. LASSO одночасно виконує дві функції: **регуляризацію** та **відбір ознак** (feature selection).

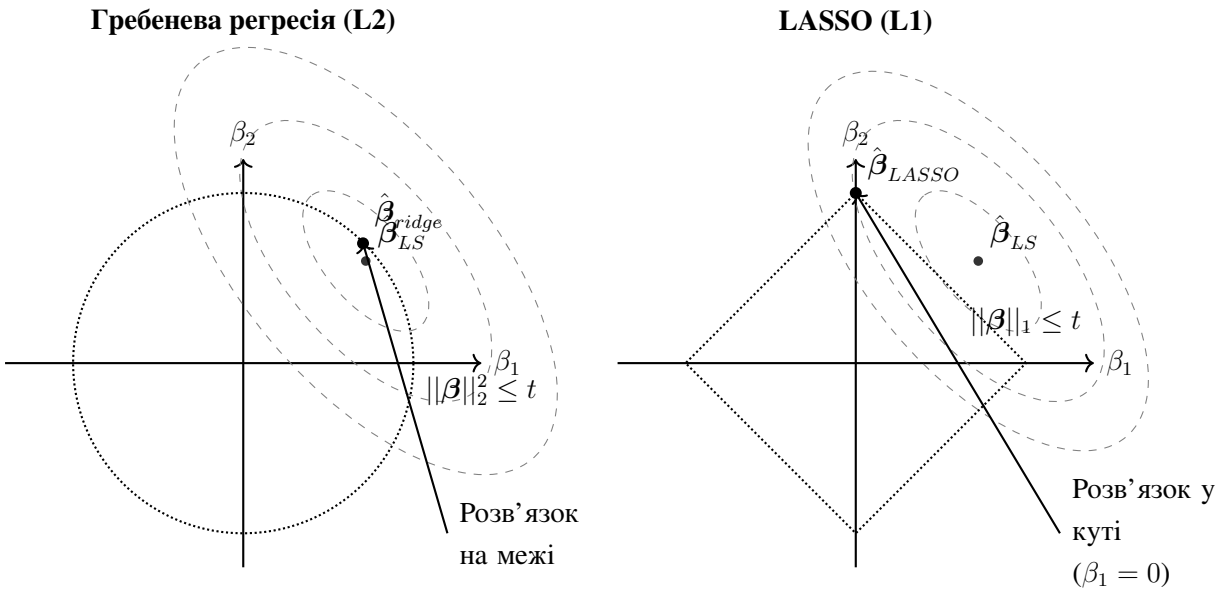


Рис. 5.1: Геометрична інтерпретація регуляризації. Еліпси — це лінії рівня функціоналу помилки МНК. Області, обмежені синім, — це області обмежень для L2 (коло) та L1 (ромб). Оптимальний розв'язок знаходиться в точці дотику. Для L1-регуляризації дотик часто відбувається на осі, що відповідає обнуленню одного з коефіцієнтів.

На відміну від МНК та Ridge, LASSO не має простого аналітичного розв'язку і вимагає ітераційних алгоритмів оптимізації (наприклад, координатного спуску).

# Алгоритми розрідженої регресії: Пошук голки в копиці сіна

Саме здатність LASSO до відбору ознак робить його ідеальним кандидатом для реалізації ідеї SINDy. Однак, LASSO має свої недоліки. Наприклад, він може бути нестабільним при високій кореляції регресорів. Тому були розроблені більш просунуті алгоритми.

Один з найефективніших підходів для SINDy — це **порогова послідовна регресія (Sequentially Thresholded Least-Squares)**. Алгоритм працює ітеративно:

1. **Крок 1: Оцінка.** Розв'язати звичайну задачу МНК для знаходження повного вектора коефіцієнтів $\boldsymbol{\beta}$.

2. **Крок 2: Поріг.** Визначити невеликі коефіцієнти, які, ймовірно, є шумом. Всі коефіцієнти $|\beta_j|$, що менші за певний поріг $\lambda$, примусово обнуляються.

3. **Крок 3: Повторення.** Створити нову, зменшену задачу регресії, використовуючи лише ті стовпці матриці $\mathbf{X}$, що відповідають ненульовим коефіцієнтам. Повернутися до кроку 1 з цією новою задачею.

Цей процес повторюється, доки набір активних коефіцієнтів не перестане змінюватися. Цей алгоритм є жадібним, але на практиці він демонструє надзвичайну ефективність у знаходженні розріджених розв'язків, що лежать в основі складних систем.

# Вибір моделі: Компроміс між точністю та складністю

Ми зіткнулися з фундаментальною дилемою: як обрати правильну модель? Або, в термінах регуляризації, як обрати оптимальне значення гіперпараметра $\lambda$?

- **Мале $\lambda$:** Модель буде дуже точною на навчальних даних, але складною і, ймовірно, перенавченою. Вона добре пояснить минуле, але погано передбачить майбутнє.

- **Велике $\lambda$:** Модель буде дуже простою (багато нульових коефіцієнтів), але може бути недостатньо точною, втрачаючи важливі аспекти динаміки (недонавчання).

Це класичний **компроміс між зміщенням та дисперсією (bias-variance tradeoff)**. Прості моделі мають високе зміщення (систематичну помилку), але низьку дисперсію (стабільність). Складні моделі — навпаки.

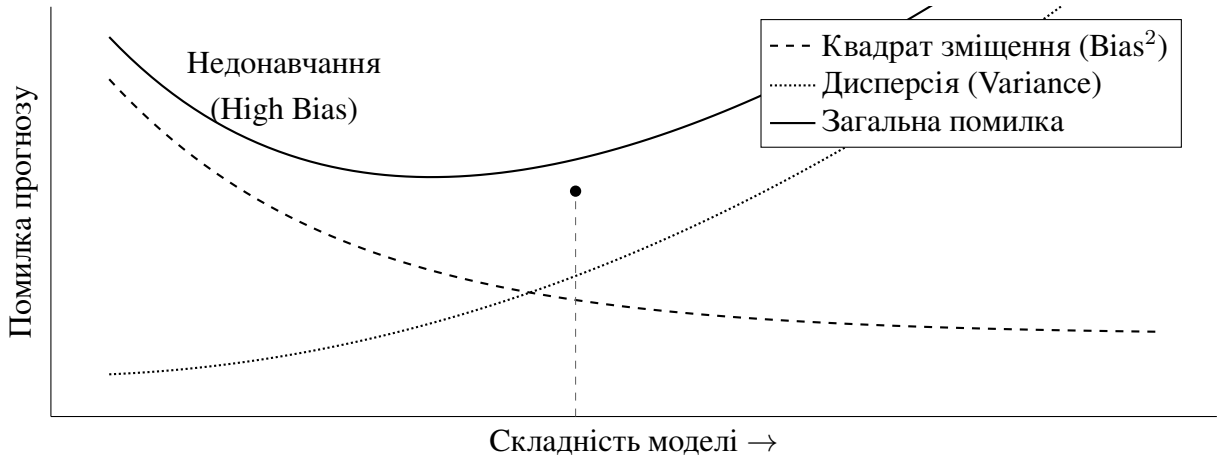


Рис. 5.2: Компроміс між зміщенням та дисперсією. Сумарна помилка на нових даних є мінімальною при оптимальній складності моделі.

Для вирішення цієї дилеми існують два основних підходи:

## Перехресна валідація (Cross-Validation)

Ідея **перехресної валідації** проста і потужна: ми не повинні оцінювати якість моделі на тих самих даних, на яких ми її навчали. Натомість, ми розбиваємо наші дані на дві частини: **навчальну** (training set) та **тестову** (validation/test set).

1. Для кожного можливого значення $\lambda$ ми навчаємо модель на навчальній вибірці.

2. Потім ми оцінюємо помилку цієї моделі на тестовій вибірці, яку модель "не бачила".

3. Ми обираємо те значення $\lambda$, яке дає найменшу помилку на тестових даних.

Цей підхід імітує здатність моделі до узагальнення на нових, невідомих даних.

## Інформаційні критерії

Альтернативний підхід полягає у використанні **інформаційних критеріїв**, які модифікують функціонал помилки, явно штрафуючи за кількість параметрів у моделі. Найвідоміші з них:

- **Інформаційний критерій Акаіке (AIC):**

$$AIC = 2k - 2\ln(\hat{L}) \tag{5.11}$$

де $k$ — кількість параметрів, а $\hat{L}$ — максимальне значення функції правдоподібності.

- **Байєсівський інформаційний критерій (BIC):**

$$BIC = k\ln(m) - 2\ln(\hat{L}) \tag{5.12}$$

де $m$ — кількість спостережень. BIC штрафує за складність сильніше, ніж AIC, і має тенденцію обирати простіші моделі.

Ми обираємо модель, що мінімізує значення відповідного критерію.

# Висновки

Параметрична ідентифікація — це не просто механічне застосування формули МНК. Це мистецтво побудови та вибору моделей. Ми перейшли від простої ідеї підгонки параметрів до значно більш потужної концепції: пошуку найбільш економного, розрідженого опису реальності, що прихований у даних.

Методи регуляризації, такі як LASSO, та алгоритми розрідженої регресії дають нам інструменти для автоматизації цього пошуку, реалізуючи принцип бритви Оккама в алгоритмічній формі. Перехресна валідація та інформаційні критерії слугують нам компасом у цьому пошуку, допомагаючи знайти оптимальний баланс між точністю та простотою.

Озброївшись цими інструментами, ми готові перейти до аналізу більш складних систем, де структура не є очевидною, а шум є невід'ємною частиною процесу. Ми навчилися не просто апроксимувати дані, а видобувати з них знання.

# Контрольні запитання

1. У чому полягає перехід від феноменологічного до механістичного моделювання? Яку роль у цьому відіграє параметрична ідентифікація?

2. Поясніть принцип “бритви Оккама” та концепцію розрідженості (sparsity) при побудові моделей. Чому прості моделі часто є кращими?

3. Запишіть загальне рівняння лінійної регресії у матричній формі та поясніть, що представляють собою вектор виходів $\mathbf{y}$, матриця регресорів $\mathbf{X}$ та вектор параметрів $\boldsymbol{\beta}$.

4. Який критерій мінімізує метод найменших квадратів (МНК)? Виведіть нормальні рівняння та їх розв’язок для оцінки параметрів.

5. Назвіть щонайменше дві проблеми, що виникають при застосуванні класичного МНК до складних задач ідентифікації (наприклад, з великою кількістю регресорів).

6. Що таке регуляризація і яку фундаментальну проблему вона допомагає вирішити?

7. Порівняйте гребеневу регресію (Ridge, L2) та LASSO (L1). Який тип регуляризації сприяє розрідженості моделі і чому?

8. Поясніть геометричну інтерпретацію L1 та L2 регуляризації. Чому LASSO схильне обнуляти коефіцієнти?

9. Опишіть кроки алгоритму порогової послідовної регресії (Sequentially Thresholded Least-Squares).

10. Поясніть суть компромісу між зміщенням та дисперсією (bias-variance tradeoff) у контексті вибору складності моделі.

11. Як працює метод перехресної валідації (cross-validation) для вибору оптимального гіперпараметра регуляризації $\lambda$?

12. Яка роль інформаційних критеріїв (AIC, BIC) у виборі моделі? Чим відрізняється “штраф” за складність у BIC порівняно з AIC?

# Розділ 6

# Ідентифікація нелінійних динамічних систем з даних

**Алгоритмічний підхід до пошуку законів природи**

## Від регресії до відкриття: Нова парадигма

У попередній лекції ми збудували міцний фундамент. Ми засвоїли, що задача параметричної ідентифікації зводиться до лінійної регресії, і навчилися керувати складністю моделі за допомогою регуляризації та перехресної валідації. Ми озброїлися "бритвою Оккама"в алгоритмічній формі — методами розрідженої регресії, такими як LASSO та порогова ітеративна згортка, що дозволяють знаходити найпростіші пояснення для складних даних. Це був перехід від простої апроксимації до осмисленого вибору моделі.

Тепер ми готові зробити наступний, вирішальний крок. Ми перейдемо від ідентифікації моделей із заданою структурою (як-от ARX) до амбітної мети **відкриття** фундаментальних законів, що керують системою, безпосередньо з експериментальних даних. Якщо раніше ми питали "які параметри найкраще описують мою модель? то тепер наше питання звучить так: "**Яка модель (яке диференціальне рівняння) найкраще описує мої дані?**".

Ми прагнемо автоматизувати процес, який протягом століть виконували вчені, від Кеплера до Ньютона: спостерігати за рухом, вимірювати дані та виводити з них математичні закони. Сучасні обчислювальні потужності та методи розрідженої регресії дозволяють нам реалізувати цю мрію для широкого класу

нелінійних систем.

# SINDy: Sparse Identification of Nonlinear Dynamics

Центральною ідеєю цієї лекції є алгоритм **SINDy (Sparse Identification of Nonlinear Dynamics)**. Його філософія елегантна і потужна: динаміка більшості фізичних систем, хоч і може здаватися складною, насправді визначається невеликою кількістю ключових взаємодій. Це означає, що праві частини диференціальних рівнянь, що описують систему, є **розрідженими** у просторі можливих функцій.

Розглянемо загальну нелінійну динамічну систему у просторі станів:

$$\frac{d}{dt}\mathbf{x}(t) = \mathbf{f}(\mathbf{x}(t)) \tag{6.1}$$

де $\mathbf{x}(t) = [x_1(t), x_2(t), \ldots, x_n(t)]^T$ — вектор стану системи. Наша мета — знайти аналітичний вигляд невідомої векторної функції $\mathbf{f}(\mathbf{x})$.

Алгоритм SINDy розв'язує цю задачу за три кроки.

## Крок 1: Збір даних та обчислення похідних

Перш за все, нам потрібні дані. Ми маємо виміряти часові ряди для всіх змінних стану системи. Результатом є матриця даних історії станів:

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}(t_1)^T \\ \mathbf{x}(t_2)^T \\ \vdots \\ \mathbf{x}(t_m)^T \end{bmatrix} = \begin{bmatrix} x_1(t_1) & x_2(t_1) & \ldots & x_n(t_1) \\ x_1(t_2) & x_2(t_2) & \ldots & x_n(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ x_1(t_m) & x_2(t_m) & \ldots & x_n(t_m) \end{bmatrix} \tag{6.2}$$

Далі, нам потрібно оцінити похідні за часом для кожної змінної стану, $\dot{\mathbf{x}}(t)$. Це критично важливий і чутливий до шуму етап. Якщо дані зашумлені, просте використання скінченних різниць може дати катастрофічні результати, оскільки диференціювання підсилює високочастотний шум. Існують більш надійні методи:

- **Поліноміальна інтерполяція (фільтри Савицького-Голея):** Дані у ков-

зному вікні апроксимуються поліномом, похідна якого потім обчислюється аналітично.

- **Регресія з використанням сплайнів або гауссових процесів.**
- **Диференціювання у частотній області** з використанням перетворення Фур'є та відповідних фільтрів.

Після цього кроку ми маємо матрицю похідних $\dot{\mathbf{X}}$, що має таку ж розмірність, як і $\mathbf{X}$.

## Крок 2: Побудова бібліотеки функцій-кандидатів

Це ключовий крок, де ми інкапсулюємо наші апріорні знання про систему. Ми припускаємо, що кожна компонента векторної функції $\mathbf{f}(\mathbf{x})$ є лінійною комбінацією функцій з деякої бібліотеки $\boldsymbol{\Theta}(\mathbf{X})$. Ця бібліотека може містити будь-які функції, які, на нашу думку, можуть брати участь у динаміці. Для прикладу, для двовимірної системи $\mathbf{x} = [x_1, x_2]^T$, бібліотека може включати:

- **Константний член:** $1$
- **Лінійні члени:** $x_1, x_2$
- **Поліноміальні члени:** $x_1^2, x_1x_2, x_2^2, x_1^3, \ldots$
- **Тригонометричні функції:** $\sin(x_1), \cos(x_1), \sin(x_2), \ldots$
- **Інші нелінійності**, специфічні для задачі.

Застосувавши ці функції до кожного рядка матриці даних $\mathbf{X}$, ми будуємо величезну матрицю регресорів $\boldsymbol{\Theta}(\mathbf{X})$:

$$\begin{aligned}\boldsymbol{\Theta}(\mathbf{X}) &= \begin{bmatrix} | & | & & | \\ \mathbf{1} & \mathbf{X} & \mathbf{X}_{P_2} & \ldots \\ | & | & & | \end{bmatrix} \\ &= \begin{bmatrix} 1 & x_1(t_1) & x_2(t_1) & x_1^2(t_1) & x_1(t_1)x_2(t_1) & \ldots \\ 1 & x_1(t_2) & x_2(t_2) & x_1^2(t_2) & x_1(t_2)x_2(t_2) & \ldots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}\end{aligned} \tag{6.3}$$

Тепер наша задача ідентифікації (6.1) перетворюється на знайому нам з минулої лекції лінійну регресійну задачу для кожного стовпця матриці $\dot{\mathbf{X}}$:

$$\dot{\mathbf{X}} \approx \mathbf{\Theta}(\mathbf{X})\mathbf{\Xi} \tag{6.4}$$

де $\mathbf{\Xi} = [\boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \ldots, \boldsymbol{\xi}_n]$ — матриця невідомих коефіцієнтів. Кожен стовпець $\boldsymbol{\xi}_k$ містить коефіцієнти, що визначають праву частину рівняння для $\dot{x}_k$. Наше припущення про розрідженість динаміки означає, що кожен вектор $\boldsymbol{\xi}_k$ має бути розрідженим, тобто містити лише кілька ненульових елементів.

## Крок 3: Розріджена регресія

Ми маємо справу з класичною задачею, яку розглядали раніше: знайти розріджений розв'язок для системи лінійних рівнянь (6.4). Оскільки задача є однаковою для кожної змінної стану, розглянемо її для однієї компоненти $\dot{x}_k$:

$$\dot{\mathbf{x}}_k \approx \mathbf{\Theta}(\mathbf{X})\boldsymbol{\xi}_k \tag{6.5}$$

Тут $\dot{\mathbf{x}}_k$ — $k$-й стовпець матриці $\dot{\mathbf{X}}$. Ми можемо застосувати будь-який з алгоритмів розрідженої регресії, які ми вивчили:

- **LASSO:**
$$\boldsymbol{\xi}_k = \arg\min_{\boldsymbol{\xi}} ||\dot{\mathbf{x}}_k - \mathbf{\Theta}(\mathbf{X})\boldsymbol{\xi}||_2^2 + \lambda||\boldsymbol{\xi}||_1 \tag{6.6}$$
- **Порогова послідовна регресія (Sequentially Thresholded Least-Squares, STLSQ):** Цей алгоритм, як правило, показує кращі результати для задач SINDy. Він ітеративно застосовує МНК та жорсткий поріг для обнулення малих коефіцієнтів.
    1. Ініціалізувати $\boldsymbol{\xi}_k$ за допомогою МНК: $\boldsymbol{\xi}_k = \mathbf{\Theta}(\mathbf{X})^\dagger \dot{\mathbf{x}}_k$.
    2. **Цикл до збіжності:**
        (а) Знайти індекси малих коефіцієнтів: $I_{small} = \{j : |\xi_{kj}| < \lambda\}$.
        (б) Обнулити малі коефіцієнти: $\xi_{kj} = 0$ для всіх $j \in I_{small}$.
        (в) Створити нову, зменшену матрицю $\mathbf{\Theta}_{big}$, що містить лише стовпці, які відповідають великим коефіцієнтам.
        (г) Розв'язати МНК на зменшеній задачі: $\boldsymbol{\xi}_{big} = \mathbf{\Theta}_{big}^\dagger \dot{\mathbf{x}}_k$.

(д) Оновити відповідні елементи в $\boldsymbol{\xi}_k$.

Після виконання цієї процедури для всіх змінних стану $k = 1, \ldots, n$, ми отримуємо розріджену матрицю коефіцієнтів $\boldsymbol{\Xi}$. Ненульові елементи цієї матриці і визначають знайдену модель динамічної системи.

# Приклад: Відкриття рівнянь осцилятора Лотки-Вольтерри

Розглянемо класичну модель "хижак-жертва":

$$\dot{x} = \alpha x - \beta xy \quad \text{(жертви)} \tag{6.7}$$

$$\dot{y} = \delta xy - \gamma y \quad \text{(хижаки)} \tag{6.8}$$

Припустимо, ми не знаємо цих рівнянь, але маємо дані часових рядів $x(t)$ та $y(t)$.

1. **Дані:** Збираємо дані $x(t_i), y(t_i)$ та чисельно обчислюємо $\dot{x}(t_i), \dot{y}(t_i)$.

2. **Бібліотека:** Будуємо поліноміальну бібліотеку до 3-го ступеня: $\boldsymbol{\Theta} = [1, x, y, x^2, xy, y^2, x^3, \ldots]$.

3. **Регресія:** Формуємо лінійну систему:

$$\begin{bmatrix} \dot{\mathbf{x}} & \dot{\mathbf{y}} \end{bmatrix} \approx \begin{bmatrix} \mathbf{1} & \mathbf{x} & \mathbf{y} & \mathbf{x}^2 & \mathbf{xy} & \mathbf{y}^2 & \ldots \end{bmatrix} \begin{bmatrix} \xi_{11} & \xi_{12} \\ \xi_{21} & \xi_{22} \\ \xi_{31} & \xi_{32} \\ \xi_{41} & \xi_{42} \\ \xi_{51} & \xi_{52} \\ \xi_{61} & \xi_{62} \\ \vdots & \vdots \end{bmatrix} \tag{6.9}$$

4. **Результат:** Застосовуючи розріджену регресію, ми очікуємо отримати розріджену матрицю коефіцієнтів $\boldsymbol{\Xi}$, де ненульовими будуть лише елементи, що відповідають членам $x$ та $xy$ для рівняння $\dot{x}$, і членам $xy$ та $y$

для рівняння $\dot{y}$. Наприклад:

$$\boldsymbol{\Xi} \approx \begin{bmatrix} 0 & 0 \\ \alpha & 0 \\ 0 & -\gamma \\ 0 & 0 \\ -\beta & \delta \\ 0 & 0 \\ \vdots & \vdots \end{bmatrix} \tag{6.10}$$

Таким чином, SINDy автоматично "відкриває"правильну структуру рівнянь з даних.

# Практичні аспекти та виклики

## Вибір гіперпараметра $\lambda$

Вибір порогу $\lambda$ є критичним і втілює компроміс між точністю та складністю, який ми обговорювали.

- **Мале $\lambda$:** Призводить до менш розріджених, більш складних моделей, які можуть бути точнішими на навчальних даних, але ризикують перенавчитися.
- **Велике $\lambda$:** Створює дуже розріджені, прості моделі, які можуть втратити важливі, але слабкі динамічні ефекти (недонавчання).

Оптимальне значення $\lambda$ часто шукають за допомогою **перехресної валідації** або скануючи діапазон значень і аналізуючи отримані моделі за допомогою **інформаційних критеріїв (AIC/BIC)**. Часто будують так звані **Pareto-діаграми**, що показують залежність між помилкою моделі та кількістю активних членів.

## Побудова бібліотеки

Ефективність SINDy сильно залежить від того, чи містить бібліотека $\boldsymbol{\Theta}$ "правильні"функції. Якщо ключовий терм відсутній у бібліотеці, алгоритм не зможе його знайти.

- **Масштабування:** Поліноміальні члени різних степенів можуть мати дуже різні масштаби ($x$ vs $x^5$). Це створює погану обумовленість матриці $\mathbf{\Theta}$. Важливо нормалізувати стовпці бібліотеки перед регресією.

- **Експертні знання:** Якщо є фізичні міркування (наприклад, закони збереження), їх можна і треба використовувати для конструювання бібліотеки.

- **Автоматизовані бібліотеки:** Існують підходи, що намагаються автоматично генерувати складні функції, наприклад, за допомогою генетичного програмування.

### Робастність до шуму

Шум у даних є головним ворогом ідентифікації. Як зазначалося, диференціювання підсилює шум. Крім робастних методів диференціювання, існують модифікації самого алгоритму SINDy:

- **Ensemble SINDy:** Алгоритм запускається багато разів на різних підвибірках даних (бутстрепінг). Обираються ті члени моделі, які з'являються у більшості реалізацій.

- **Інтегральна форма SINDy:** Замість диференціювання даних, рівняння (6.4) інтегрується за часом. Це перетворює чутливу операцію диференціювання на стабілізуючу операцію інтегрування, що значно підвищує робастність до шуму.

## Висновки

Ми здійснили подорож від фундаментальних ідей регресії до потужної методології автоматизованого наукового відкриття. SINDy та подібні до нього підходи, що базуються на розрідженості, змінюють наш погляд на моделювання. Вони дозволяють нам ставити машині запитання не лише про параметри, а й про саму структуру законів природи.

Ключові ідеї, які слід винести з цієї лекції:

1. **Принцип розрідженості:** Складні системи часто керуються простими законами, що виражаються через невелику кількість взаємодій.

2. **Переформулювання задачі:** Проблема ідентифікації нелінійної динаміки може бути точно переформульована як задача розрідженої лінійної регресії.

3. **Бібліотека функцій:** Наші апріорні знання та гіпотези про світ кодуються у вигляді бібліотеки функцій-кандидатів.

4. **Алгоритмічний інструментарій:** Методи розрідженої регресії (LASSO, STLSQ) є обчислювальним ядром, що дозволяє знаходити розріджені моделі.

Звісно, цей підхід не є "срібною кулею". Його успіх залежить від якості даних, вдалого вибору бібліотеки та правильного налаштування гіперпараметрів. Проте, він відкриває захоплюючі перспективи для аналізу складних систем у фізиці, біології, економіці та інженерії, дозволяючи нам видобувати прості, інтерпретовані та узагальнюючі моделі з потоків даних.

# Контрольні запитання

1. Який фундаментальний зсув парадигми відбувається при переході від класичної параметричної ідентифікації до підходів на кшталт SINDy?

2. Сформулюйте ключове припущення про розрідженість (sparsity), на якому базується алгоритм SINDy. Чому це припущення є обґрунтованим для багатьох фізичних систем?

3. Опишіть три основні етапи алгоритму SINDy.

4. Чому крок обчислення похідних з експериментальних даних є настільки критичним і чутливим до шуму? Назвіть один надійний метод для чисельного диференціювання.

5. Яку роль відіграє бібліотека функцій-кандидатів $\boldsymbol{\Theta}(\mathbf{X})$? Які апріорні знання вона втілює?

6. Поясніть, як задача ідентифікації нелінійної системи $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$ перетворюється на задачу лінійної регресії $\dot{\mathbf{X}} \approx \boldsymbol{\Theta}(\mathbf{X})\boldsymbol{\Xi}$.

7. Чому звичайний метод найменших квадратів не підходить для розв'язання фінального рівняння в SINDy? Який тип регресії використовується натомість?

8. Опишіть ітеративний процес роботи алгоритму порогової послідовної регресії (STLSQ).

9. На прикладі системи Лотки-Вольтерри поясніть, як розріджена матриця коефіцієнтів $\Xi$ дозволяє відновити структуру вихідних диференціальних рівнянь.

10. Яку роль відіграє гіперпараметр (поріг) $\lambda$ в алгоритмах розрідженої регресії? Що станеться, якщо обрати занадто мале або занадто велике значення $\lambda$?

11. Назвіть два підходи до вибору оптимального значення гіперпараметра $\lambda$.

12. Які практичні проблеми можуть виникнути при побудові бібліотеки функцій? Як погане масштабування стовпців може вплинути на результат?

13. Назвіть один із методів підвищення стійкості SINDy до шуму в даних.

# Розділ 7

# Валідація та аналіз якості моделей

***Про мистецтво вибору — як відрізнити корисну модель від химери в океані даних***

## Від достатку до вибору: нова проблема

У попередній лекції ми озброїлися потужним інструментом — алгоритмом SINDy, що дозволяє нам, подібно до сучасних алхіміків, перетворювати сирі дані на золото математичних рівнянь. Ми навчилися будувати бібліотеки потенційних законів природи та, за допомогою розрідженої регресії, виокремлювати з них ті, що найімовірніше керують системою. Проте, цей успіх породжує нову, не менш складну проблему. Змінюючи гіперпараметр розрідження $\lambda$, ми можемо згенерувати ціле сімейство моделей — від найпростіших, що містять лише один-два члени, до надзвичайно складних, що намагаються врахувати кожну флуктуацію в даних.

Ми стоїмо перед фундаментальною дилемою, яку влучно сформулював статистик Джордж Бокс: *«Всі моделі є невірними, але деякі є корисними»*. Наша задача — не знайти міфічну "істинну"модель, бо дані завжди зашумлені, а вимірювання неповні. Наша мета — обрати **найкориснішу** модель: ту, що найкраще балансує між точністю опису, простотою інтерпретації та здатністю до узагальнення.

Ця лекція присвячена методології та філософії валідації — мистецтву та

науці вибору моделі. Ми перейдемо від процесу "відкриття"до процесу "перевірки що є не менш важливим етапом у циклі наукового пізнання.

# Фундаментальний інструментарій валідації

## Розділення даних: наріжний камінь чесної оцінки

Найперший і найважливіший принцип валідації — ніколи не оцінювати якість моделі на тих самих даних, на яких вона навчалася. Це було б самообманом. Модель, що ідеально відтворює навчальні дані, може виявитися абсолютно безпорадною при зіткненні з новою інформацією. Це явище називається **перенавчанням (overfitting)**.

Щоб уникнути цього, ми розділяємо наявний набір даних щонайменше на дві незалежні частини:

- **Навчальна вибірка (Training set):** Використовується виключно для ідентифікації моделі — тобто для запуску алгоритму SINDy та знаходження коефіцієнтів $\Xi$.
- **Тестова (валідаційна) вибірка (Test set):** Ця частина даних "ховається"від алгоритму на етапі навчання. Вона служить для незалежної, неупередженої перевірки якості отриманої моделі.

Типове співвідношення для розділення — 70-80% для навчання та 20-30% для тестування.

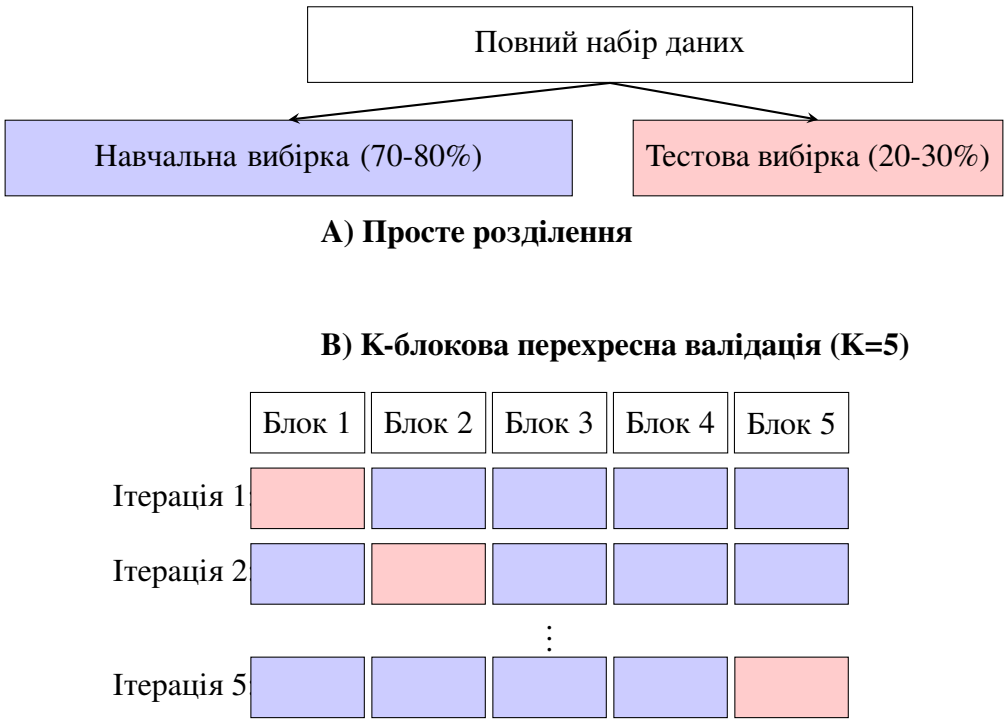


Рис. 7.1: Візуалізація методів розділення даних. (A) Просте розділення на навчальну та тестову вибірки. (B) 5-блокова перехресна валідація, де на кожній ітерації один блок використовується для тестування (червоний), а решта — для навчання (синій).

## Перехресна валідація: максимум інформації з обмежених даних

Часто даних буває недостатньо, щоб дозволити собі розкіш відкласти значну частину для тестування. У таких випадках застосовують більш складний підхід — **перехресну валідацію (Cross-Validation)**. Найпоширенішим її варіантом є **k-fold Cross-Validation**:

1. Весь набір даних випадковим чином розбивається на $k$ рівних частин (наприклад, $k = 5$ або $k = 10$).

2. Процес навчання та валідації повторюється $k$ разів.

3. На кожній ітерації одна з $k$ частин використовується як тестова вибірка, а решта $k - 1$ частин — як навчальна.

4. Метрики якості (про них нижче) обчислюються для кожної з $k$ ітерацій, а потім усереднюються.

Цей підхід дає значно надійнішу оцінку якості моделі, оскільки кожна частина даних встигає побувати і в ролі навчальної, і в ролі тестової вибірки.

# Кількісна оцінка: Метрики якості

Щоб порівнювати моделі, нам потрібні об'єктивні критерії. Основний спосіб перевірки — це **симуляція**. Ми беремо ідентифіковану модель (систему диференціальних рівнянь), задаємо їй початкові умови з тестової вибірки ($\mathbf{x}(t_0)$) і інтегруємо її в часі, отримуючи модельний прогноз $\hat{\mathbf{x}}(t)$. Потім цей прогноз порівнюється з реальними даними $\mathbf{x}(t)$ з тестової вибірки.

## Коефіцієнт відповідності (Fit Percent)

Це одна з найпростіших метрик, що

# Інформаційні критерії: елегантність та ощадливість

Як формалізувати компроміс між складністю та точністю? Інформаційні критерії пропонують елегантний спосіб зробити це, штрафуючи модель за кількість параметрів.

## Інформаційний критерій Акаіке (AIC)

$$\text{AIC} = 2p - 2\ln(L)$$

де $p$ — кількість параметрів у моделі (ненульових коефіцієнтів у матриці $\Xi$), а $L$ — максимальне значення функції правдоподібності для моделі. Для випадку гауссового шуму AIC можна записати як:

$$\text{AIC} \approx m\ln(\text{MSE}) + 2p$$

де $m$ — кількість точок даних.

### Байєсівський інформаційний критерій (BIC)

$$\text{BIC} = p\ln(m) - 2\ln(L) \approx m\ln(\text{MSE}) + p\ln(m)$$

BIC накладає більший штраф на складність моделі, ніж AIC, особливо для великих наборів даних (коли $\ln(m) > 2$). Тому BIC схильний обирати простіші (більш розріджені) моделі.

При використанні цих критеріїв ми обчислюємо їх значення для кожної моделі з нашого сімейства (отриманого для різних $\lambda$) і обираємо ту, для якої значення AIC або BIC є **мінімальним**.

## Парето-фронт: карта компромісів

Замість того, щоб покладатися на один-єдиний критерій, надзвичайно корисно візуалізувати компроміс між складністю та точністю. Для цього будують так звану **Парето-діаграму**.

- По осі X відкладається складність моделі (кількість активних членів у бібліотеці).
- По осі Y відкладається помилка моделі (наприклад, MSE на тестовій вибірці).

Кожна точка на графіку відповідає одній моделі, отриманій за допомогою SINDy з певним порогом $\lambda$. Сукупність цих точок утворює хмару. Нас цікавить її нижня межа, так званий **Парето-фронт**.

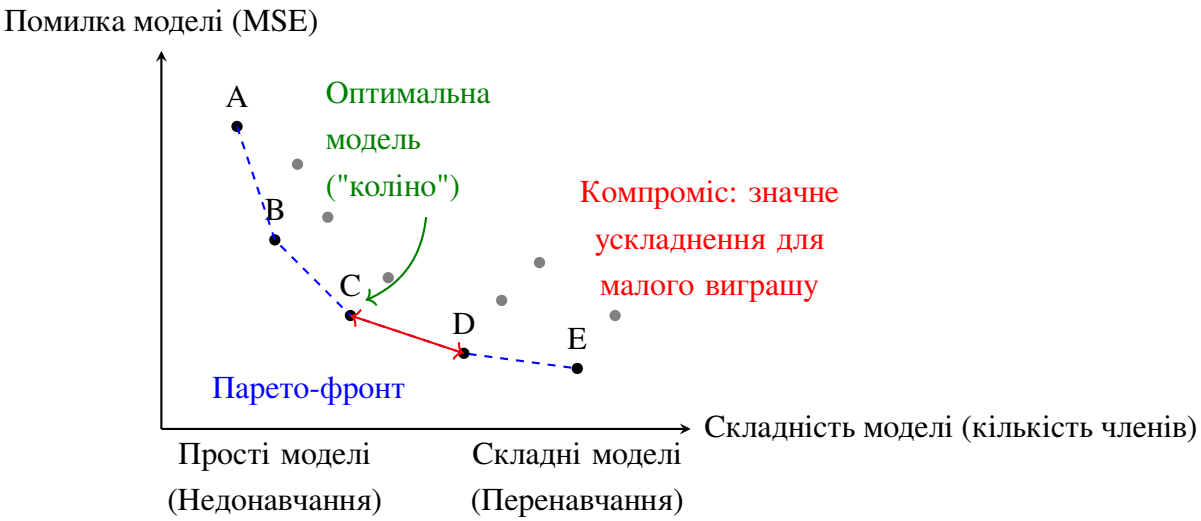


Рис. 7.2: Парето-фронт для вибору моделі. Діаграма ілюструє компроміс між помилкою моделі та її складністю. Моделі на фронті (A-E) є оптимальними. Вибір часто падає на "коліно"кривої (модель C), де досягається найкращий баланс між точністю та простотою.

Моделі, що лежать на цьому фронті, є "оптимальними"в тому сенсі, що неможливо покращити їхню точність без збільшення складності, і неможливо зменшити їхню складність без втрати точності. Вибір конкретної моделі на цьому фронті — це вже стратегічне рішення дослідника, що залежить від мети. Часто шукають "коліно"на цьому графіку — точку, де подальше ускладнення моделі дає лише незначне зменшення помилки.

# Стійкість та узагальнення

## Стійкість до шуму

Реальні дані завжди зашумлені. Хороша модель повинна бути стійкою — тобто, її структура не повинна кардинально змінюватися при невеликих змінах у даних. Один зі способів це перевірити — **бутстрепінг (bootstrapping)**:

1. Багато разів (сотні, тисячі) створювати нові навчальні вибірки шляхом випадкового вибору з початкового набору даних (з поверненням).
2. На кожній такій вибірці запускати SINDy.
3. Аналізувати, які члени бібліотеки обираються найчастіше.

Якщо певний терм (наприклад, $x^2$) з'являється у 99% реалізацій, ми можемо бути впевнені в його важливості. Якщо інший терм з'являється лише у 10% випадків, ймовірно, він є артефактом шуму.

### Здатність до прогнозування

Найсуворіший тест для будь-якої динамічної моделі — це її здатність прогнозувати майбутнє. Модель може добре відтворювати дані на короткому інтервалі часу, але її помилки можуть експоненційно зростати при довгостроковому прогнозуванні. Тому критично важливою є перевірка моделі на її **прогнозуючій здатності** на даних, які вона ніколи не бачила, і на часових горизонтах, що значно перевищують ті, на яких вона навчалася.

## Висновки

Валідація — це не формальний останній крок, а невід'ємна, ітеративна частина процесу моделювання. Вона перетворює "полювання"за рівняннями на строгий науковий процес. Ми не просто знаходимо модель, ми ретельно досліджуємо її властивості, сильні та слабкі сторони.

Ключові ідеї, які слід винести з цієї лекції:

1. **Чесність оцінки:** Завжди використовуйте незалежні тестові дані.
2. **Баланс:** Якість моделі — це компроміс між точністю та складністю. Парето-фронт є найкращим інструментом для візуалізації цього компромісу.
3. **Діагностика:** Аналіз залишків дозволяє зазирнути "під капот"моделі та зрозуміти, що саме вона не змогла пояснити.
4. **Формалізація простоти:** Інформаційні критерії (AIC, BIC) дають кількісну оцінку принципу "бритви Оккама".
5. **Стійкість:** Надійна модель повинна бути стійкою до шуму та зберігати свою структуру при варіаціях даних.

Озброївшись методами ідентифікації з попередньої лекції та методами валідації з цієї, ми завершуємо цикл побудови моделі з даних: від вимірювань

до вибору остаточного, корисного та інтерпретованого рівняння, що описує світ.

## Контрольні запитання

1. Чому, згідно з цитатою Джорджа Бокса, ми шукаємо «корисну», а не «істинну» модель?

2. Поясніть фундаментальний принцип валідації щодо розділення даних. Що таке перенавчання (overfitting) і як розділення на навчальну та тестову вибірки допомагає його уникнути?

3. У яких випадках доцільно використовувати k-блокову перехресну валідацію замість простого розділення даних? Опишіть її алгоритм.

4. Що таке інформаційні критерії AIC та BIC? Поясніть, як вони допомагають формалізувати компроміс між точністю та складністю моделі.

5. Чим відрізняється штраф за складність в критеріях AIC та BIC? Який з них схильний обирати простіші моделі і чому?

6. Що таке Парето-фронт у контексті вибору моделі? Які осі використовуються для його побудови і що представляють точки на цьому фронті?

7. Що означає знайти «коліно» на Парето-фронті і чому це часто є хорошою стратегією для вибору моделі?

8. Поясніть метод бутстрепінгу (bootstrapping) для оцінки стійкості моделі до шуму. Як частота появи термів у рівнянні вказує на їхню важливість?

9. Чому здатність до довгострокового прогнозування вважається найсуворішим тестом для динамічної моделі?

10. Назвіть п'ять ключових ідей, які слід винести з лекції про валідацію моделей.

11. Уявіть, що за допомогою SINDy ви згенерували 10 різних моделей для одного набору даних. Опишіть ваш покроковий план дій для вибору найкращої з них, використовуючи методи з цієї лекції.

12. Чи може модель з мінімальним значенням AIC/BIC не бути найкращою з точки зору прогнозування на довгостроковому горизонті? Поясніть свою думку.

# Словник ключової термінології

**Адаптивність (Adaptivity)** Здатність системи змінювати свою поведінку у відповідь на зовнішні впливи.

**Бритва Оккама (Occam's Razor)** Принцип, що рекомендує обирати найпростішу модель, яка пояснює дані.

**Валідація (Validation)** Перевірка якості моделі на незалежних даних.

**Взаємодія (Interaction)** Вплив елементів системи один на одного.

**Вихід (Output)** Реакція системи на вхідний сигнал.

**Вхід (Input)** Зовнішній сигнал або дія, що подається на систему.

**Динамічний процес (Dynamic Process)** Процес, що змінюється у часі.

**Дискретний час (Discrete Time)** Представлення процесу у вигляді послідовності моментів часу.

**Диференціальне рівняння (Differential Equation)** Математичне рівняння, що описує зміну змінної у часі.

**Експериментальні дані (Experimental Data)** Дані, отримані шляхом спостереження або експерименту.

**Емерджентність (Emergence)** Властивість системи, коли ціле має характеристики, що не властиві окремим елементам.

**Зміщення (Bias)** Систематична помилка моделі, що виникає через її спрощення.

**Зворотний зв'язок (Feedback)** Механізм, за якого вихід системи впливає на її подальшу поведінку.

**Ідентифікація системи (System Identification)** Процес побудови математичної моделі системи на основі експериментальних даних.

**Критерій якості (Quality Criterion)** Математична функція, що оцінює ступінь відповідності моделі реальним даним.

**Лінійна модель (Linear Model)** Модель, у якій вихід є лінійною функцією входу.

**Лінійна регресія (Linear Regression)** Метод знаходження залежності між змінними шляхом підбору лінійної функції.

**LASSO** Метод розрідженої регресії для вибору важливих ознак у моделі.

**Механізм (Mechanism)** Причинно-наслідковий зв'язок, що пояснює поведінку системи.

**Модель (Model)** Формалізоване представлення системи, що дозволяє аналізувати, прогнозувати або керувати її поведінкою.

**Недонавчання (Underfitting)** Ситуація, коли модель недостатньо складна для опису даних.

**Нелінійність (Nonlinearity)** Властивість системи, коли зміна виходу не пропорційна зміні входу.

**Непараметричний (Nonparametric)** Метод, що не передбачає фіксованої структури моделі.

**Непараметрична ідентифікація (Nonparametric Identification)** Визначення властивостей системи без фіксованої структури моделі.

**Об'єкт (Object)** Окрема частина реальності, яку можна спостерігати або досліджувати.

**Оцінка параметрів (Parameter Estimation)** Визначення числових значень параметрів моделі.

**Перехідна характеристика (Step Response)** Реакція системи на одиничний стрибок входу.

**Підсистема (Subsystem)** Частина системи, що виконує окрему функцію.

**Порогова згортка (Thresholding)** Метод вибору значущих параметрів у моделі.

**Параметрична ідентифікація (Parametric Identification)** Визначення параметрів моделі із заданою структурою.

**Параметричний (Parametric)** Метод, що передбачає фіксовану структуру моделі.

**Параметр моделі (Model Parameter)** Кількісна характеристика моделі, що визначає її поведінку.

**Регресор (Regressor)** Змінна, що використовується для прогнозування виходу моделі.

**Регуляризація (Regularization)** Метод запобігання перенавчанню шляхом обмеження складності моделі.

**Розрідженість (Sparsity)** Властивість моделі, коли більшість параметрів дорівнюють нулю.

**Самоорганізація (Self-organization)** Процес виникнення впорядкованої структури без зовнішнього керування.

**SINDy** Алгоритм розрідженої ідентифікації нелінійної динаміки.

**Стаціонарність (Stationarity)** Властивість системи, коли її характеристики не змінюються у часі.

**Стохастичність (Stochasticity)** Властивість системи, коли її поведінка містить випадкові елементи.

**Структура (Structure)** Організація елементів системи та їх зв'язків.

**Система (System)** Сукупність взаємодіючих елементів, що утворюють єдине ціле для досягнення певної мети.

**Тестова вибірка (Test Set)** Частина даних, що використовується для перевірки якості моделі.

**Узагальнення (Generalization)** Здатність моделі коректно працювати на нових, невідомих даних.

**Феноменологія (Phenomenology)** Опис явищ без пояснення їх причин.

**Хаос (Chaos)** Нелінійна поведінка системи, що призводить до непередбачуваних результатів.

**Шум (Noise)** Випадкові або неконтрольовані впливи, що спотворюють вимірювання.

**Якість моделі (Model Quality)** Ступінь відповідності моделі реальним даним та її здатність до узагальнення.

# Список джерел


[1] R. L. Ackoff, *Ackoff's Best: His Classic Writings on Management*. Wiley, 1999.

[2] J. M. Epstein, «Why Model?» *Journal of Artificial Societies and Social Simulation*, т. 11, № 4, с. 12, 2008, ISSN: 1460-7425. https://www.jasss.org/11/4/12.html.

[3] T. Söderström та P. Stoica, *System Identification*. New York: Prentice Hall, 1989, 612 с., ISBN: 978-0-13-881236-2.

[4] L. Ljung, *System Identification: Theory for the User* (Prentice Hall information and system sciences series). Upper Saddle River, N.J: Prentice Hall PTR, 1999, 609 с., ISBN: 978-0-13-656695-3.

[5] H. Sayama, *Introduction to the Modeling and Analysis of Complex Systems*. Open SUNY Textbooks, 2015, 498 с., ISBN: 978-1-942341-09-3. дата зверн. 10 серп. 2025. https://open.umn.edu/opentextbooks/textbooks/233.

[6] M. De Domenico та H. Sayama, «Complexity Explained,» за уч. Open Science Framework, 30 серп. 2022, Publisher: Open Science Framework. DOI: 10.17605/OSF.IO/TQGNW дата зверн. 8 серп. 2025. https://complexityexplained.github.io/.

[7] G. E. P. Box та N. R. Draper, *Empirical Model-Building and Response Surfaces*. New York: John Wiley & Sons, 1987.

[8] S. L. Brunton та J. N. Kutz, *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge New York, NY Port Melbourne New Delhi Singapore: Cambridge University Press, 2019, 492 с., ISBN: 978-1-108-42209-3. https://www.databookuw.com/.