МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені ІВАНА ФРАНКА
Факультет прикладної математики та інформатики
Кафедра дискретного аналізу та інтелектуальних систем

Моделювання еволюційних систем

Реферат на тему:

“Керування хаосом”

Виконала:
студентка групи ПМІ-43
Шувар С.О.

Прийняв:
доц. Вовк В. Д.

Львів – 2022

Зміст

Вступ

Згідно міфології стародавні греки вважали, що хаос - це невпорядкований стан неоформленої матерії, яка, нібито, існувала до впорядкованого Всесвіту. Але навіть, якщо сьогодні запитати звичайну людину, що не є спеціалістом в даній області, що вона розуміє під поєднанням «керування хаосом», із дуже високою ймовірністю, вона відповість, що це підсильне тільки чомусь божественному і величному. Але насправді ідея керування хаосом привернула увагу дослідників ще у 1990 році і на сьогоднішній день це уже не ідея, а реальність. Керування хаосом активно використовується у біології, екології, хімії, медицині, фізиці, економіці, і ще у величезній кількості галузей. Але, які саме можливості нам відкриває вміння керувати хаосом?

Одного разу мені випала можливість спостерігати за тим, як учні Львівської академії танцю вчились виконувати один із найскладніших трюків балетного мистецтва - фуете. Суть полягає в тому, щоб виконувати повороти на одній нозі, раз за оберт піднімаючись на пятачок пуанту. Скажу чесно, процес був точно не найприємніший. Учні постійно спотикались, нерідко підвертали ноги та падали. Під кінець уроку лише кількох з них вдалось досягти чогось хоть трішки схожого до бажаного результату. Цей досвід наштовхнув мене на роздуми з приводу того, що так як, балансуючи на одній нозі, учням важко втримати стан рівноваги, то систему в якій перебуває уже доросла артистка балету під час виконання фуете можна вважати хаотичною. А так як при цьому вона намагається не просто зберегти положення рівноваги, а ще й отримати перевагу із свого нестабільного положення, а саме перетворити його у чудовий рух, з впевненістю можемо заявити, що в даний момент часу балерина керує хаосом. Сумніваюсь, що ця інформація принесла б хоч крихту користі танцівникам, але даний приклад є непоганим підґрунтям для того щоб зрозуміти, що можливість керувати хаосом не лише може допомогти уникнути різних непередбачуваніх обставин, але і стати можливістю для отримання переваг від того, що система є власне хаотичною.

У даній роботі буде розглянуто основну ідею керування хаосом, буде висвітлено алгоритм OGY - як фундаментальний алгоритм керування хаосом, та проілюстровано роботу даного алгоритму на конкретному прикладі. В загальному буде розглянуто інші алгоритми керування, а також висвітлена класифікація задач керування хаосом. На останок, буде наведено кілька прикладів використання цих алгоритмів у різних сферах.

Розділ 1 Попередні відомості

У цьому розділі буде розглянута деяка базова інформація про хаотичні динамічні системи.

1.1 Хаотичні динамічні системи

У поєднанні “керування хаосом” мова йде про керування хаотичними динамічними системами, тому перш ніж перейти до розгляду алгоритмів керування важливо зрозуміти, що саме ховається під цим терміном.

У загальному вигляді рівняння динамічної системи записується у вигляді:

$$\dot{x} = f(x, u) \tag{1.1}$$

$$x_{i+1} = f(x_i, u_i) \tag{1.2}$$

де x - вектор стану, а u - вхідний вектор системи.

Якщо функція f- нелінійна по x і/або u - система також нелінійна. Рівняння (1.1) - рівняння неперервної системи в часі, рівняння (1.2) - рівняння дискретної системи в часі. Неперервну систему можна перевести до дискретної використовуючи січення Пуанкаре.

Хаотична система - особливий вид нелінійної системи, що характеризується хаотичною поведінкою. Однак цю поведінку неможливо розпізнати, просто дивлячись на рівняння системи, незалежно від того, на скільки воно є простим. Розпізнати хаотичну поведінку можна лише аналізуючи реакцію системи, яка може бути дуже складною та схожою на шум. Через його складність не існує загальноприйнятого визначення хаотичної системи. Зупинимось на визначенні, яке дав Роберт Деваней, а саме він виділив три властивості хаотичних динамічних систем:

1. Хаотична система має чутливу залежність від початкових умов.
2. Хаотична система є топологічно транзитивною.
3. Періодичні орбіти та/або точки є щільними в хаотичному атракторі.

Топологічна транзитивність означає, що система нерозкладна, тобто її не можна розбити на дві підсистеми.

Першу і останню властивості розглянемо детально.

1.1.1 Залежність від початкових умов.

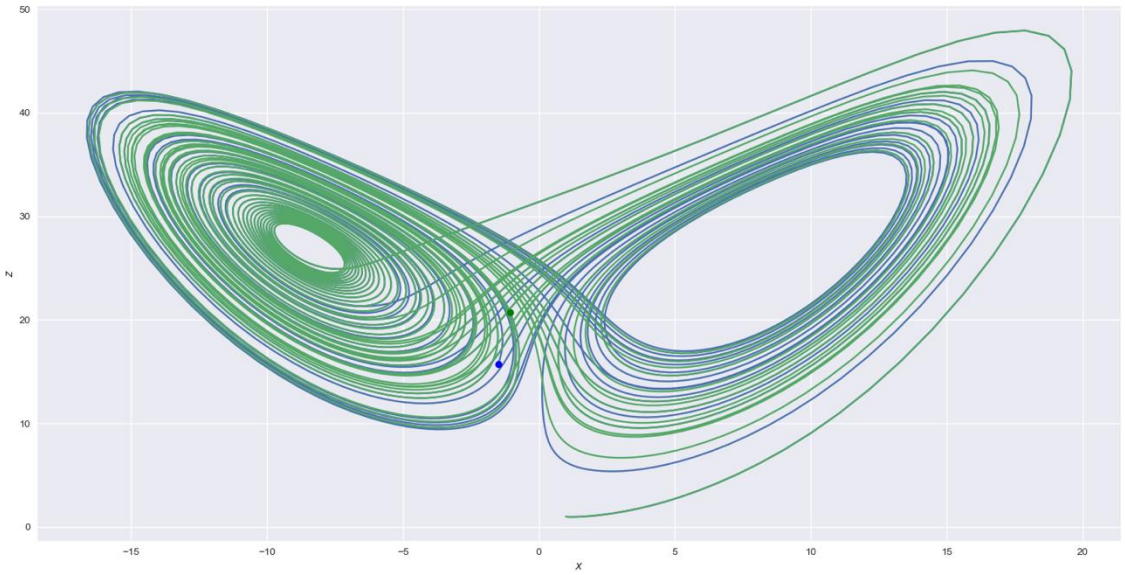
Чутлива залежність від початкових умов означає, що хаотична система є непередбачуваною, оскільки її реакція сильно залежатиме від початкових умов. Невелике збурення початкових умов може призвести до зовсім іншої реакції. Переконатись у цьому допоможе приклад.

Розглянемо систему Лоренца, яка, як відомо, поводиться хаотично для певних значень параметрів і початкових умов:

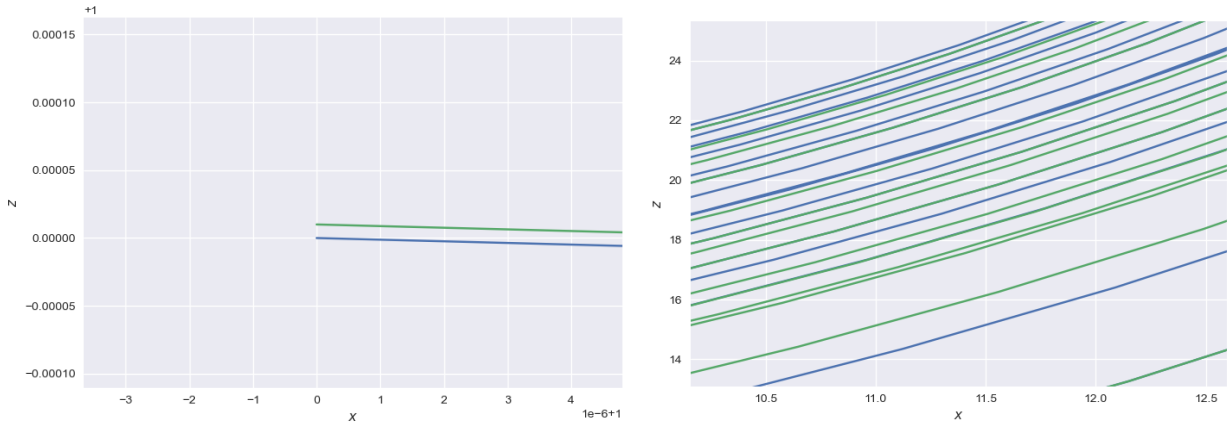
$$\begin{cases} \dot{x} = \sigma(y - x) \\ \dot{y} = x(r - z) - y \\ \dot{z} = xy - bz \end{cases} \tag{1.3}$$

де x – швидкість руху повітря, y, z – змінні, котрі характеризують горизонтальний та вертикальний розподіли температури, σ, r, b - параметри, які описують властивості атмосферного середовища. Атрактор Лоренца – це сукупність хаотичних розв'язків системи Лоренца. Зокрема, система поводиться хаотично при значеннях параметрів $\sigma = 10, r = 28, b = \frac{8}{3}$.

Для того щоб переконатись, що атрактор Лоренца дійсно сильно залежить від початкових умов розглянемо рис. (1.1), на якому зображено дві проекції дивного атрактора Лоренца з вищезгаданими значеннями параметрів на площину (xz) при 40 ітераціях з початковою координатою (1; 1; 1) та (1; 1; 1.00001) відповідно. Жирні синя та зелена точки демонструють на скільки далеко один від одного знаходяться розв'язки системи по завершенню 40 ітерацій.



а)



б) *в)*

Рис. 1.1 (а – дивний атрактор Лоренца з початковою умовою (1; 1; 1) та (1; 1; 1.00001); б – цей же атрактор приближеній для демонстрації різниці початкових умов; в – невелике зближення цього ж атрактора для демонстрації хаотичності траєкторій, (див додаток 1, скрипт 1))

Залежність від початкових координат показана теж на рис. (1.2)**.** Тут по осі ординат на верхньому графіку відкладені значення x_i системи за обох початкових умов по завершенню i-тої ітерації. Для більшої наглядності нижній графік ілюструє цю різницю.

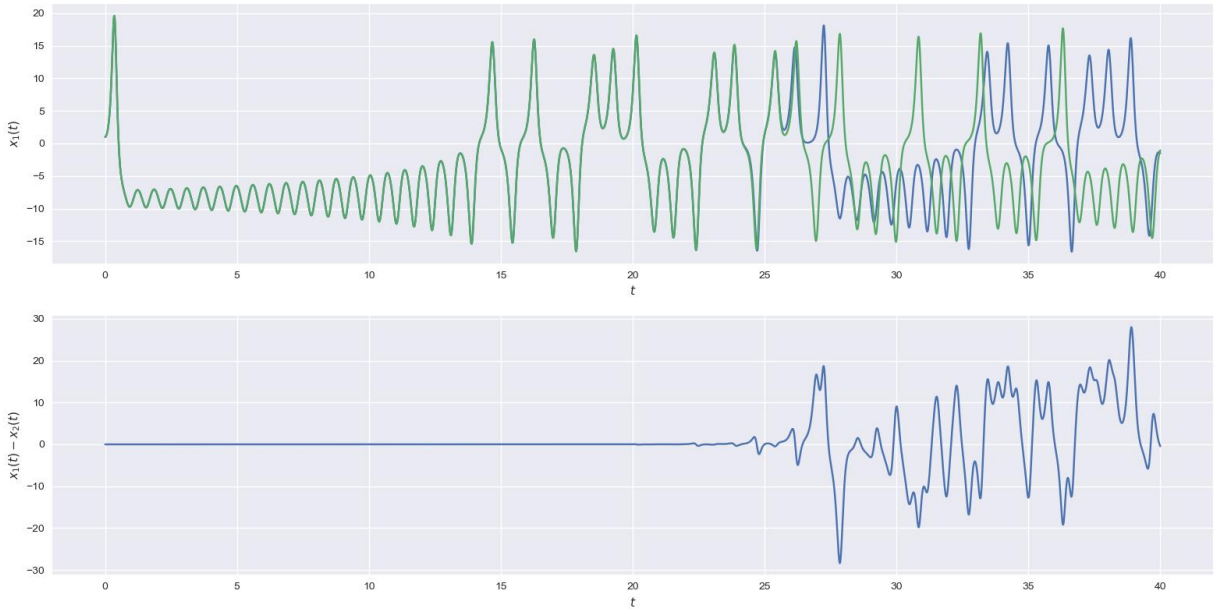


Рис 1.2 (Динаміка зміни x_i дивного атрактора Лоренца з початковою умовою (1; 1; 1) та (1; 1; 1.00001) , (див додаток 1, скрипт 1))

Перше, що зразу ж привертає увагу це те, на скільки нестабільною є компонента x. Це може бути неочевидним, з рис. (1.1).

Але висновок, який випливає з цих графіків, це те, що для двох розв'язків початкові координати яких трішки відрізняються за компонентою z, розв'язки спочатку майже ідентичні: якщо дивитись неозброєним оком, різниця між двома розв'язками дорівнює нулю. Але незабаром (приблизно через 20 ітерацій) різниця між двома розв`язками має приблизно таку ж величину, як і самі розв`язки, а це означає, що система є сильно чутливою до початкових умов.

1.1.2 Щільність хаотичного атрактору.

Хаотичну систему можна проаналізувати лише спостерігаючи за її розв'язками для великого набору різних початкових умов, оскільки одне рішення, аперіодичне чи ні, не може повністю описати хаотичну систему. Хоча типова аперіодична реакція хаосу виглядає як шум, її структурована поведінка чітко показує протилежне. Така структура називається ергодичною поведінкою. Коли система блукає простором, у певний час t_1 вона перетинає точку x_1. Через час t_2 після t_1 вона знову потрапить у деякий ε окіл точки x_1. Те саме станеться через час t_3 після моменту часу t_2 і т.д. Загалом, чим менший цей окіл , тим більше $t_2, t_3, \ldots$ буде. Але хоча розв'язок дуже наблизиться до x_1, він ніколи не стане йому повністю рівним (при $\varepsilon \to 0$, $t_i \to \infty$). Ця ергодична поведінка пояснює так званий дивний атрактор, який виникне, - певний підпростір у просторі, в середині якого буде блукати рішення системи. Більш того, розглянувши січення Пуанкаре даної системи нестійкі траєкторії дивного атрактора утворюють фрактальний «пил» з точок, які при $t \to \infty$ все щільніше заповнюють криву лінію.

Знову звернемось до прикладу дивного атрактора Лоренца. На жаль, щільність атрактора неможливо показати використовуючи ЕОМ з обмеженими можливостями, але рис. (1.3) демонструє, що зі збільшенням кількості ітерацій та за різних початкових умов усі траєкторії дивного атрактора Лоренца дійсно обмежуються певним підпростором та зі зростанням кількості ітерацій справді більш щільно заповнюють цей підпростір. На малюнку різними кольорами представлена плинність ітерацій.

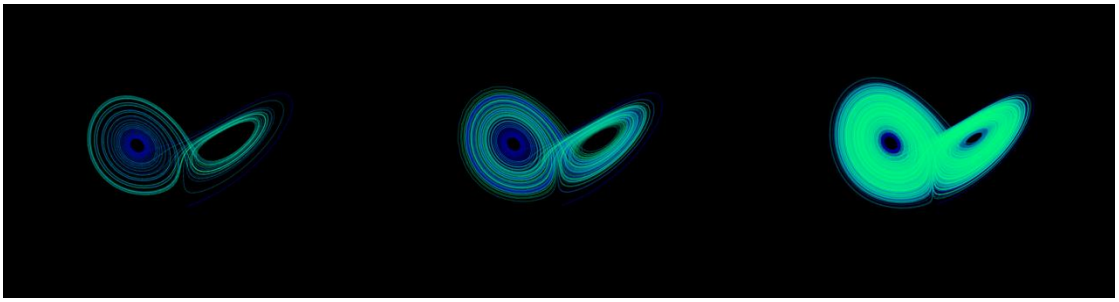


Рис. 1.3 (Заповнення підпростору територіями атрактора Лоренца при 50, 150, та 1000 ітераціях, (див додаток 1, скрипт 2))

Розв’язок не збігається ні до стабільної точки, ні до періодичної орбіти, а також x, y, z не прямують до нескінченності. Натомість, система блукає в певному підпросторі, ніколи не перетинаючи себе, створюючи аперіодичні траєкторії. Цей дивний атрактор показує ергодичну і, отже, хаотичну поведінку системи. Однак слід зазначити, що для деяких початкових умов розв’язок стає періодичним, а для інших початкових умов $x, y, z \rightarrow \pm\infty$.

Розділ 2 Теорія Керування Хаосом та алгоритм OGY

2.1 Ідея керування хаосом

У попередньому розділі було розглянуто такі важливі характеристики хаотичних динамічних систем як сильна чутливість до початкових умов та ергодичність, але в основі теорії керування хаосом лежить ще одна фундаментальна властивість хаотичних динамічних систем. А саме, у скелеті хаотичного атрактора існує не лише нескінченна кількість хаотичних траєкторій але і нескінченна кількість періодичних орбіт, кожна з яких є нестабільною. На рис. (2.1) продемонстрировано, що система Лоренца проводиться періодично, наприклад, при значенні параметра $r = 350$ та $r = 160$ відповідно.

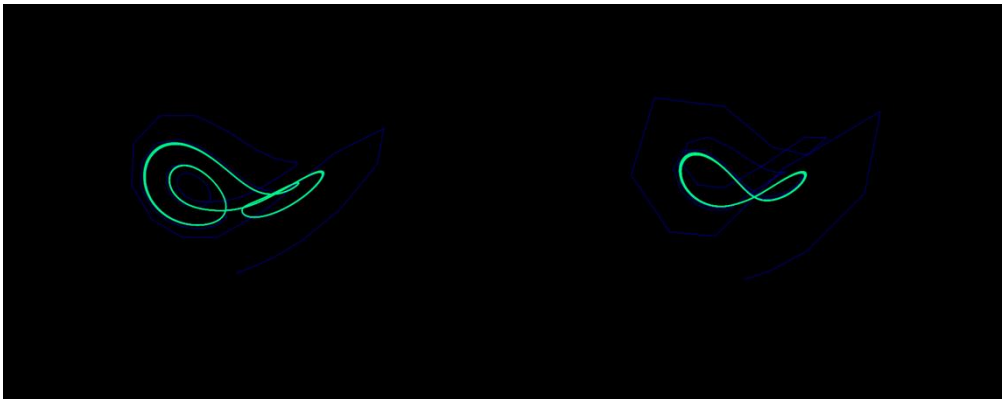


Рис. 2.1 (Періодичні орбіти системи Лоренца при значенні параметра $r = 350$ *та* $r = 160$ *(див. додаток 1, скрипт 2)*

Саме із цих трьох властивостей випливає основна ідея можливості керування хаосом. Суть полягає в тому, що **траєкторія ергодично наближається до бажаної періодичної орбіти, вбудованої в атрактор, тому застосувавши невеликі збурення таку орбіту можна стабілізувати**. Цей факт наштовхнув на думку про те, що критична чутливість хаотичної системи до змін (збурень) у її початкових умовах може бути насправді дуже бажаною в практичних експериментальних ситуаціях.

Ідея контролю хаосу була висловлена 1990 році в Університеті Меріленда вченими Едвардом Оттом, Селсо Гребогі та Джеймсом Йорком. У своїй роботі вони виклали ідеї контролю хаосу та запропонували метод стабілізації нестабільної періодичної орбіти, як доказ принципу.

Основна ідея полягає в тому, щоб стабілізувати одну (або більше) з орбіт хаотичного атрактора шляхом застосування невеликих збурень. Для застосування цих збурень один із параметрів системи повинен бути

доступним, тобто цей параметр можна регулювати під час роботи системи. Таким чином, цей параметр стає входом системи. Оскільки хаос поводиться ергодично, в якийсь момент часу розв'язок надійде в окіл певної точки орбіти, де дійсна лінеаризація. За допомогою цієї лінеаризації простий метод розміщення полюсів може бути використаний для розрахунку керуючого зусилля, щоб спрямувати систему до цієї точки і, отже, до орбіти.

Висловивши цю загальну тезу, вони проілюстрували її за допомогою конкретного методу, так званого методу OGY (Отта, Гребогі та Йорка), для досягнення стабілізації вибраної нестабільної періодичної орбіти. У методі OGY невеликі, спеціально підібрані удари застосовуються до системи один раз за цикл, щоб підтримувати її поблизу бажаної нестабільної періодичної орбіти.

2.2 Метод OGY

Як було сказано раніше, основна ідея алгоритму OGY полягає в тому, що існує велика кількість нестабільних періодичних орбіт, вбудованих у хаотичний атрактор. Збурюючи певний доступний параметр, можна стабілізувати одну (або більше) з цих орбіт. Але як це працює? У цьому розділі буде представлено математику алгоритму OGY. Важливо зазначити те, що керування OGY застосовне лише до систем дискретних у часі. У разі неперервної системи, цю систему слід спочатку перетворити на систему дискретного часу, використовуючи, наприклад, січення Пуанкаре. Таким чином n-вимірну систему можна записати у вигляді:

$$x_{i+1} = f(x_i, p) \tag{2.1}$$

Де $x_i \in \mathbb{R}^n$ є n-вимірним станом системи на ітерації i, а p - доступний параметр, згаданий у розділі 2.1. Для параметра p визначено номінальне значення $\bar{p}$, для якого система поводиться періодично. Таким чином значення $|p - \bar{p}|$ і є шуканою величиною збурення. Оскільки збурення є малим, значення цієї різниці - обмежене:

$$|p - \bar{p}| < \delta \tag{2.2}$$

Візьмемо $p = \bar{p}$. Так як в цьому випадку траєкторія є періодичною, то зафіксуємо точку $x_{i+T}^* = x_i^*$, де T - період. Спочатку розглянемо випадок, коли довжина цього періоду $T = 1$, маємо $x_{i+1}^* = x_i^* = x^*$, де x^* – вбудована нестабільна фіксована точка в середині хаотичного атрактора. Оскільки система ергодична, стан x_i наблизиться до цієї точки в якийсь момент часу, так як $p = \bar{p}$. У такому випадку, систему (2.1) можна лінеаризувати навколо x^*:

$$x_{i+1} - x^* = A(x_i - x^*) + B(p - \bar{p}) \tag{2.3}$$

де A - якобіан, а B - вплив «входу» p:

$$A = \frac{df}{dx}(x^*, \bar{p}) = D_x f(x, p) \tag{2.4}$$

$$B = \frac{df}{dp}(x^*, \bar{p}) = D_p f(x, p) \tag{2.5}$$

Важливо зазначити, що система є стабільною якщо всі власні значення λ_i матриці A менші 1.

Згідно алгоритму розподілу полюсів (pole placement), система є керованою, якщо у матриці контролю:

$$P = [B, AB, A^2B, \dots, A^{n-1}B] \quad , \tag{2.6}$$

ранг є рівний n, тобто вимірності системи.

Якщо система контрольована, то можна визначити зворотний зв’язок стану, тобто:

$$p - \bar{p} = -K^T(x_i - x^*) \quad , \tag{2.7}$$

таким чином:

$$x_{i+1} - x^* = A(x_i - x^*) + B(p - \bar{p}) = \\ = A(x_i - x^*) - BK^T(x_i - x^*) = \\ = (A - BK^T)(x_i - x^*) \tag{2.8}$$

Тепер матриця $(A - BK^T)$ є новою системною матрицею, тому стабільність керованої системи визначається власними значеннями $(A - BK^T)$ і залежатиме від K. Таким чином, власні значення, або полюси, системи можна розмістити будь-де, вибравши відповідне K. Вибравши ці полюси відповідно до $|\lambda_i| < 1$ система стане стабільною.

Таким чином, стабілізація лінійної системи перетворюється на знаходження відповідності вибраним полюсам. Це можливо зробити, наприклад, завдяки формулі Аккермана.

Маючи це K, ми уже маємо величину шуканого збурення. Все, що залишилось зробити, так це визначити максимальне значення δ, таке, що шукана різниця не може його перевищувати. У такому випадку присвоюємо збуренню значення 0.

Алгоритм OGY для орбіт (або точок) з періодом T більшим одиниці є дещо складніший. Припустимо, що існує орбіта періоду T, вбудована в хаотичний атрактор, що означає $x^*_{i+T} = x^*_i$ для будь-якого i. Отже, для кожної ітерації (або перетину з картою Пуанкаре) можна лінеаризувати систему, знаходячи T різних матриць A_i (якобіан) і векторів B_i :

$$A_i = A_{i+T} = \frac{df}{dx}(x^*_i, \bar{p}) = D_x f(x, p) \tag{2.9}$$

$$B_i = B_{i+T} = \frac{df}{dp}(x^*_i, \bar{p}) = D_p f(x, p) \tag{2.10}$$

оцінюється при $x = x^*_i$ і $p = \bar{p}$. На кожній ітерації i лінеаризація тепер визначається як:

$$x_{i+1} - x^*_{i+1} = A_i(x_i - x^*_i) + B_i(p_i - \bar{p}) \tag{2.11}$$

Крім того, на нестійкій орбіті періоду T (з $p = \bar{p}$) відомо, що

$$x^*_{i+T} = f(x^*_{i-1+T}, \bar{p}), x^*_{i-1+T} = f(x^*_{i-2+T}, \bar{p}), \dots, x^*_{i+1} = f(x^*_i, \bar{p}) \tag{2.12}$$

Щоб визначити стабільність цієї орбіти, можна дослідити поширення невеликої похибки ε. Це можна зробити за допомогою розширення Тейлора

$$f(x_i + \varepsilon, \bar{p}) = f(x_i, \bar{p}\,) + D_x f(x_i, \bar{p})\varepsilon + O(\varepsilon^2) \approx f(x_i, \bar{p}) + D_x f(x_i, \bar{p})\varepsilon \quad (2.13)$$

Після T ітерацій, використовуючи $D_x f(x_i) = A_i$, що пізніше призводить до:

$$f(x_1 + \varepsilon, \bar{p}) \approx f(x_1, \bar{p}) + D_x f(x_1, \bar{p})\varepsilon = x_2 + A_1\varepsilon \quad (2.14а)$$

$$f(x_2 + A_1\varepsilon, \bar{p}) \approx f(x_2, \bar{p}) + D_x f(x_2, \bar{p})A_1\varepsilon = x_3 + A_2A_1\varepsilon \quad (2.14б)$$

$$\vdots$$

$$f(x_T + A_{T-1} \dots A_1\varepsilon, \bar{p}) \approx x_1 + (A_TA_{T-1} \dots A_2A_1)\varepsilon \quad (2.14в)$$

Похибка після одного періоду обертання, за наближенням, дорівнює $(A_TA_{T-1} \dots A_2A_1)\varepsilon$.

Зрозуміло, що ця похибка є меншою ε при $(A_TA_{T-1} \dots A_2A_1)$ менше одиниці. Іншими словами, стійкість орбіти періоду T визначається власними значеннями λ матриці $(A_TA_{T-1} \dots A_2A_1)$.

Проте власні вектори відрізняються. Там, де стабільність орбіти однакова на кожній ітерації, напрямок, у якому це відбувається, звичайно, інший. Тепер припустимо, що орбіта має s стабільних і u нестабільних власних значень. Тоді на кожній ітерації i є стабільні власні вектори $\{v_{i,1}, \dots, v_{i,s}\}$ і u нестабільні власні вектори $\{v_{i,1}, \dots, v_{i,u}\}$, які утворюються власними векторами $(A_iA_{i-1} \dots A_{i-T+1})$. Тоді ідея OGY полягає в тому, щоб контролювати u нестабільних напрямків. Система збурюється u разів, так що x_{i+u} приземляється на лінеаризований стійкий різновид періодичної орбіти через точку x^*_{i+u}. Іншими словами, після u ітерацій відхилення має лежати на лінеаризованому стабільному підпросторі, охопленому стабільними власними векторами $\{v_{i+u,1}, \dots, v_{i+u,s}\}$:

$$x_{i+u} - x^*_{i+u} = a_1v_{i+u,1} + a_2v_{i+u,2} + \cdots + a_sv_{i+u,s} \quad (2.15)$$

Використання лінеаризації (2.11) $x_{i+u} - x^*_{i+u}$ апроксимується:

$$x_{i+2} - x_{i+2}^* = A_{i+1}(x_{i+1} - x_{i+1}^*) + B_{i+1}(p_{i+1} - \bar{p}) \quad (2.16а)$$
$$= A_{i+1}A_i(x_i - x_i^*) + A_{i+1}B_i(p_i - \bar{p}) +$$
$$+B_{i+1}(p_{i+1} - \bar{p})$$

$$x_{i+u} - x_{i+u}^* = A_{i+(u-1)} \dots A_i(x_i - x_i^*) \quad (2.16б)$$
$$+ A_{i+(u-1)} \dots A_{i+1}B_i(p_i - \bar{p})$$
$$+ \ A_{i+(u-1)} \dots A_{i+2}B_{i+1}(p_{i+1} - \bar{p}) + \cdots$$
$$+ A_{i+(u-1)}B_{i+(u-2)}\left(p_{i+(u-2)} - \bar{p}\right)$$
$$+ \ B_{i+(u-1)}\left(p_{i+(u-1)} - \bar{p}\right)$$

$$= \ \Phi_{i,0}(x_i - x_i^*) + \Phi_{i,1}B_i(p_i - \bar{p})$$
$$+ \Phi_{i,2}B_{i+1}(p_{i+1} - \bar{p}) + \cdots$$
$$+ \Phi_{i,u-1}B_{i+(u-2)}\left(p_{i+(u-2)} - \bar{p}\right)$$
$$+ \ B_{i+(u-1)}(p_{i+(u-1)} - \bar{p})$$

Тут $\Phi_{i,j}$ визначається як:

$$\Phi_{i,j} = A_{i+u-1}A_{i+u-2} \dots A_{i+j+1}A_{i+j} \quad (2.17)$$

Крім того, визначається матриця C_iна кожній i-тій ітерації:

$$C_i = \left[\Phi_{i,1}B_i \vdots \Phi_{i,2}B_{i+1} \vdots \cdots \vdots \Phi_{i,u-1}B_{i+u-2} \vdots v_{i+u,1} \vdots v_{i+u,2} \vdots \cdots \right. \quad (2.18)$$
$$\left. \vdots v_{i+u,s}\right]$$

Система є керованою, якщо C_i неодинична. Керуюче зусилля отримаємо аналогічно:

$$p_i - \bar{p} = -K_i^T(x_i - x_i^*) \quad (2.19)$$

2.3 Застосування OGY до мапи Хенона

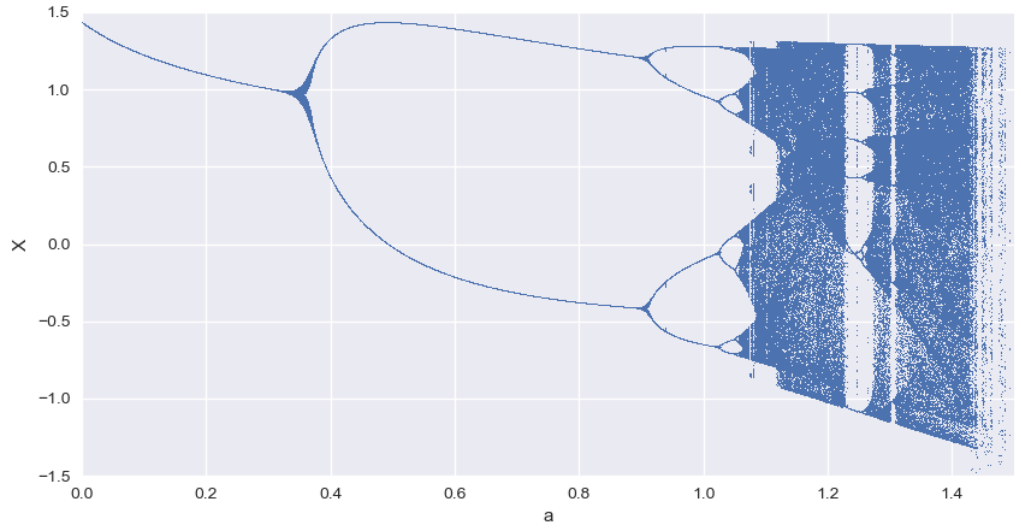
2.3.1 Мапа Хенона

Особливим видом хаотичної системи є так звана мапа Хенона, названа на честь її винахідника. Мапа Хенона є суто теоретичною картою, яка не відображає фізичної системи. Тим не менш, ця дискретна часова система, або

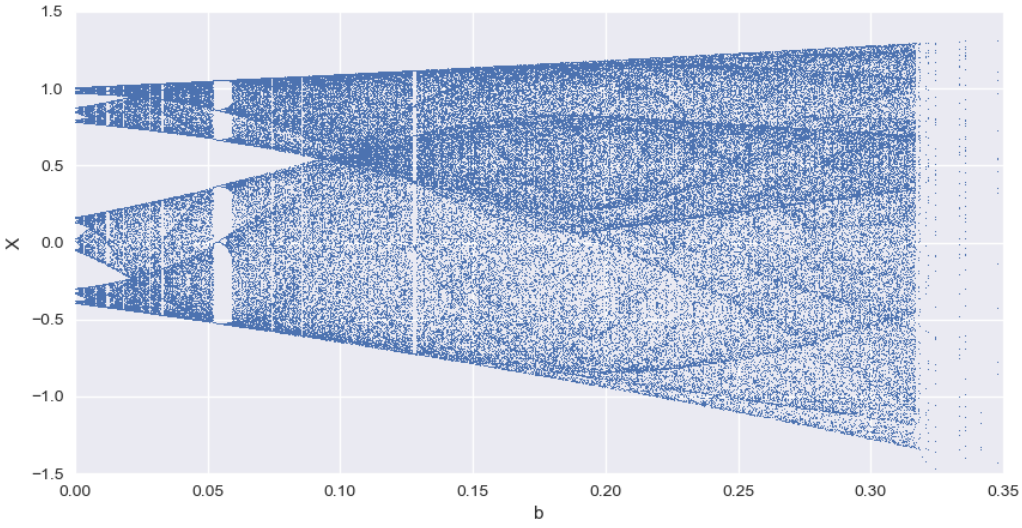
карта, описує основні характеристики хаотичної поведінки і має дуже чіткий дивний атрактор. Рівняння задається системою:

$$x_{i+1} = f_1(x_i, y_i) = y_i + 1 - ax_i^2 \qquad (2.20)$$
$$y_{i+1} = f_2(x_i, y_i) = bx_i$$

Типові значення параметрів, при яких виникає хаос - це $a = 1.4$ і $b = 0.3$. Це ілюструють біфуркаційні діаграми карти Хенона, показані на рис. (2.2) (перша для фіксованого b, друга для фіксованого a).

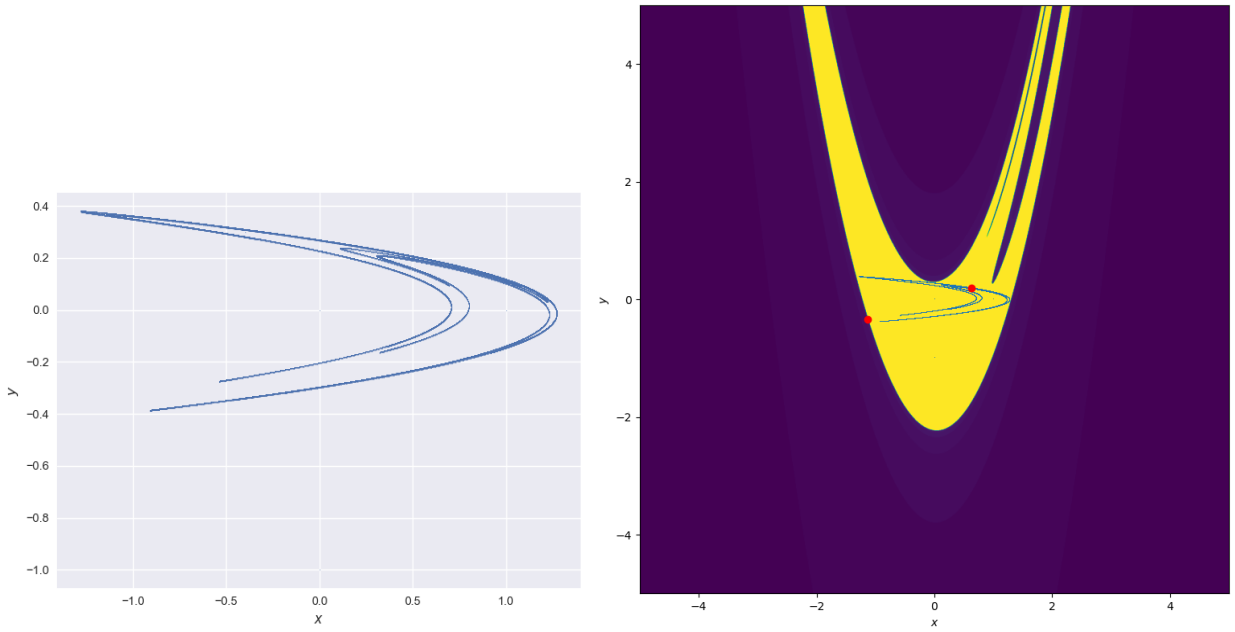


а)



б)

Рис. 2.2 (Біфуркаційні діаграми мапи Хенона а) b = 0.3, a – змінна б) b – змінна, a = 1.4. (див додаток 1, скрипт 3))



а) б)

Рис. 2.3 (а) дивний атрактор мапи Хенона (a = 1.4; b = 0.3)
б) басейн притягання мапи Хенона (див додаток 1, скрипт 4 та 5))

Дивний атрактор карти Хенона з цими значеннями параметрів показаний на малюнку (2.3 а), де y_i є функцією x_i. Після усунення перехідної поведінки рішення карти завжди блукатиме на цьому атракторі. Але деякі початкові умови ніколи не збігаються до атрактора ($y_i \to \pm\infty$). Коли досліджується довгострокова поведінка ($t \to \infty$) кожної початкової умови, виходить басейн притягання, який показаний на малюнку (2.3 б). Усі початкові умови всередині жовтої V-подібної частини будуть сходитися до атрактора, тоді як для інших точок $y_i \to \pm\infty$.

2.3.2 Застосування алгоритму

Згідно до алгоритму описаного у розділі 2.2 знайдемо фіксовані точки:

$$x_{i+1} = x_i = x^*$$
$$y_{i+1} = y_i = y^*$$

прийнявши період $T = 1$.

$$\left.\begin{matrix} x^* = y^* + 1 - ax^{*2} \\ y^* = bx^* \end{matrix}\right\} \Rightarrow x^* = bx^* + 1 - ax^{*2}$$
$$\Longrightarrow ax^{*2} + (1-b)x^* - 1 = 0$$

Звідси,

$$x^* = \frac{-(1-b) \pm \sqrt{(1-b)^2 + 4a}}{2a}$$

Для $a = 1.4$ та $b = 0.3$, отримали:

$$\begin{pmatrix} x^* \\ y^* \end{pmatrix} = \begin{pmatrix} -1.1314 \\ -0.3394 \end{pmatrix} \quad \text{та} \quad \begin{pmatrix} x^* \\ y^* \end{pmatrix} = \begin{pmatrix} 0.6314 \\ 0.1894 \end{pmatrix},$$

Ці точки відмічені на рис. (2.3 б**)**. Так як перша точка лежить на краю басейну притягання і тому не підходить для контролю OGY, при подальших обрахунках буде використовуватись лише друга фіксована точка.

Лінеаризація карти Хенона за допомогою (2.3) дає

$$A = D_{x,y} f(x, y, a, b) = \begin{bmatrix} -2ax & 1 \\ b & 0 \end{bmatrix}$$

$$B_a = D_a f(x, y, a, b) = \begin{bmatrix} -x^2 \\ 0 \end{bmatrix}$$

$$B_b = D_b f(x, y, a, b) = \begin{bmatrix} 0 \\ x \end{bmatrix}$$

Перевіряємо, чи дана система є керованою, склавши відповідні кожному з параметрів матриці контролю:

$$P_a = [B_a, AB_a] = \begin{bmatrix} -x^2 & 2ax^3 \\ 0 & -bx^2 \end{bmatrix}$$

$$P_b = [B_b, AB_b] = \begin{bmatrix} 0 & x \\ x & 0 \end{bmatrix}$$

Обидві матриці мають повний ранг, якщо $x \neq 0$. Так як система контрольована застосовуємо правило зворотного зв'язку.

$$p - \bar{p} = -K^T(x_i - x^*) \quad ,$$

Матрицю K^Tможна з легкістю обчислити використовуючи алгоритм Аккермана.

Успіх OGY залежить лише від вибору максимального збурення параметра, яке назвемо δa_{max} або δb_{max}. Відповідний код наведено в додатку 1 скрипт 6 . На малюнках 2.4 показано вихід карти Хенона для різних варіантів доступного параметра a. Бачимо, що система веде себе хаотично, аж поки не застувалось збурення. Дійсно переконуємось, що чим більше це значення δb_{max}, тим довше шукається окіл.

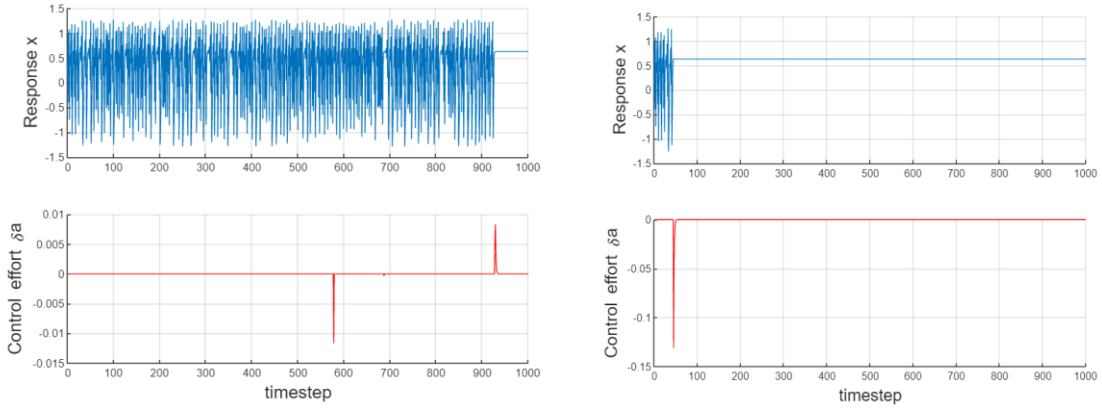


Рис. 2.4 (Робота алгоритму OGY за різних значень δb_{max}: 0.015 та 0.15 відповідно (див додаток 1, скрипт 6))

Також нерідко може виникнути помилкове зусилля керування, коли стан знаходиться далеко від фіксованої точки. У цьому випадку зусилля керування δa або δb не призведе до сталого стану. Крім того, через поведінку системи можливо, що система відійде від фіксованої точки на ітерації $i + 1$ далі ніж на i-тій ітерації, що призведе до хибного зусилля керування. Це явище трапляється частіше, коли δa_{max} і δb_{max} є відносно великими. Тоді алгоритм спробує контролювати точки, які знаходяться далі від фіксованої точки, де

лінеаризація є не завжди точною. Це призводить до значно більшої кількості спроб контролю, в тому числі більшої кількості помилкових.

2.4 Аналіз алгоритму OGY

Отож, основне спостереження за контролем хаосу полягає в тому, що будь-яка неперервна або дискретна хаотична система, з параметром керування p, має велику кількість періодичних орбіт вбудованих в його хаотичний атрактор. Алгоритм керування OGY керує хаотичною системою в напрямку однієї або кількох із цих орбіт, збурюючи параметр p на незначну величину. Оскільки хаотичний атрактор є ергодичним, рішення в якийсь момент часу потрапить в окіл розглянутої орбіти, де можна застосувати лінеаризацію та простий метод керування розміщенням полюсів.

Виділимо 2 основні обмеження OGY:

1. Орбіти, якими керує OGY, обмежені орбітами, вбудованими в атрактор. OGY лише перетворює хаос на періодичний розв'язок, але який цей розв'язок не можна визначити заздалегідь, а його форму не можна змінити за допомогою закону керування.
2. Також у попередньому розділі було розглянуто вплив δp_{max}. Загалом, малі значення δp_{max} призведуть до тривалого часу досягнення контролю. Окрім меншого часу для досягнення контролю, вищі значення δp_{max} призведуть до більшої кількості помилкових спроб керування, оскільки не кожна точка, де застосовується керування, знаходиться в межах лінійної близькості до орбіти.

Однак варто сказати що OGY потребує дуже невеликих зусиль контролю, що, безперечно, є перевагою, наприклад для систем, де параметри не можуть бути сильно змінені через різні обмеження. Ще одна перевага полягає в тому, що алгоритм не потребує ніякої математичної моделі.

Розділ 3 Інші алгоритми контролю

У роботі Отта, Гребогі і Йорка параметр варіювався з урахуванням поточного стану системи, тобто за допомогою зворотного зв'язку. У наступних публікаціях цей ефект було підтверджено експериментально. Парадоксальність висновку про те, що хаос не можна передбачити, але ним можна керувати, викликав вибуховий інтерес дослідників і лавину публікацій. На даний момент можна виділити три основні підгрупи методів: Open loop control, Feedback control та Time delayed feedback. Варто зазначити, що прицип роботи Feedback control та Time delayed feedback полягає у стабілізації однієї або кількох орбіт за допомогою збурень параметру системи, натомість у методі Open loop control стабілізація відбувається шляхом застосування зовнішньої сили до системи.

3.1 Feedback Control (контроль зі зворотнім зв’язком)

Розглянутий у попередньому розділі алгоритм OGY належить до групи методів feedback control. Принцип роботи цих методів полягає у стабілізації однієї або кількох орбіт за допомогою коливань параметру системи на основі попереднього значення цього параметру. У такому методі визначають функцію (або коефіцієнт, на який домножити), через яку проганятиметься заданий параметр, щоб пізніше вихід цієї функції додавати або віднімати від входу n-ного стану системи.

3.2 Time Delayed Feedback (метод зворотного зв'язку з запізненням)

Метод зворотного зв'язку з запізненням запропонував литовський фізик Кестуттіс Пірагас у 1992 році. Метод за принципом роботи дуже подібний на метод контролю зі зворотним зв'язком, але ціль контролю із затримкою зворотного зв'язку - це стабілізація T-періодичної орбіти нелінійної системи. Дану ціль можна досягти за допомогою складання правила зворотного зв’язку із затримкою T, яка буде дорівнювати періодичності орбіти та побудови схеми системи контролю зворотного зв'язку. Метод використовує властивість високої чутливості параметрів до збурення.

Тут можна розглядати контрольовану систему, як так званий «чорний ящик», що означає, що нам не потрібно знати форми стабілізуючої орбіти, так само, як і рівняння системи.

3.3 Open Loop Control (Програмний метод)

Принцип роботи методу Open loop control полягає в тому, щоб стабілізувати одну або декілька орбіт у фазовому просторі. На відміну від Feedback control стабілізація тут відбувається за допомогою коливань якогось зовнішнього контролюючого сигналу, який подається на вхід у систему. Цей сигнал може бути, або якоюсь фізичною дією на систему, тобто зовнішня сила або поле, або виражати зміну деякого параметра керованої системи. З плюсів цього методу варто виділити те, що він досить просто реалізується, бо при використанні цього методу не потрібно проводити будь яких вимірів. В основному цей метод використовується під час управління над швидкими процесами, що проходять на молекулярному або атомному рівні, так як для них неможливо виміряти стан системи.

Розділ 4. Керування хаотичними системами: задачі та застосування

4.1 Задачі керування хаотичними системами

Насправді, до цього мова йшла лише про одну із задач керування хаотичними системами, а саме про задачу стабілізації нестійкої періодичної орбіти. Це завдання виникає при придушенні шумів чи вібрацій різних конструкцій, усунення небажаних шумів у системах зв'язку, електроніці, тощо.

Крім завдання стабілізації нестійкої періодичної орбіти, виділяють іще три основні задачі керування хаотичними динамічними системами.

Другий клас завдань керування відповідає завданням порушення, або генерації хаотичних коливань. Такі завдання виникають, коли хаотичний рух є бажаним типом поведінки системи. Класичними прикладами є генератори псевдовипадкових чисел. Також доведено, що хаотизація процесів може дати дуже відчутний ефект у хімічних технологіях і біотехнологіях, при обробці сипких матеріалів.

Третій важливий клас - задачі керування синхронізацією. Синхронізація має важливі застосування у вібраційній техніці, у техніці зв'язку, у біології та біотехнологіях, тощо. Яскравим прикладом того, як інженерська помилка в області саме задач синхронізації ледь не призвела до трагедії, може бути ситуація, що сталась у день відкриття Міленівського мосту в Лондоні. Так як міст є підвісним, то він коливається. У день відкриття на ньому перебували багато людей, і всі вони крокували по мосту, через ці коливання, хаотичні кроки людей стабілізувались у такт мосту, що розхитувало його ще більше. Зараз на мості розмістили спеціальні придушувачі коливань.

Останнім типом завдань управління хаотичними процесами є модифікація атракторів, наприклад, перетворення хаотичних коливань в періодичні та назад. Це використовується в лазерних та хімічних технологіях, у техніці телекомунікацій, у біології та медицині.

4.2 Використання методів керування хаотичними системами

У перші роки після впровадження поняття детермінованого хаосу в наукову літературу, система хаотичної поведінки вважалася екзотичним явищем, цікавим лише математикам, та не мала нічого спільного із практичним використанням. Однак з часом хаотична динаміка була виявлена у величезній кількості різних систем, а саме, у механіці, системах зв'язку, лазерній фізиці та радіофізиці, хімії та біохімії, біології, в економіці та медицині.

Розвиток теорії керування хаосом виявив цілий ряд реальних практичних завдань, де хаотичні режими дійсно можуть виникнути, будучи іноді шкідливими, а іноді - корисними. Зараз методи керування хаосом уже активно використовуються у багатьох сферах. Прикладами можуть бути:

1. У біологічних та екологічних сферах було показано, що навіть простий State Feedback може швидко стабілізувати динаміку популяцій і використовуватись для контролю над паразитами та комахами.
2. Динаміка багатьох економічних систем описується нелінійними моделями, тобто вони можуть володіти хаотичною поведінкою. Тож цілюю керування є подолання хаосу, що призводить до передбачуваності бізнес циклів.
3. Методи керування хаосом активно використовуються і у медицині.
    - Одним із найбільш захоплюючих застосувань керування хаосом використовується під час лікування аритмії, а саме, лягло в основу створення швидкодіючих кардіостимуляторів з зворотнім зв’язком.
    - Також ці методи дозволили розробити алгоритми для зменшення рівня хаотичності коливань при сезонних епідеміях. Показано, що хаос може бути подоланий, якщо використовувати сталий, досить високий темп вакцинації.
    - Методи теорії керованого хаосу можуть допомогти справитися з труднощами управління вмісту цукру в крові, привести до правильного підбору індивідуальних процедур лікування цукрового діабету та його автоматизації.
4. У фізиці основними прикладами застосування цих методів є використання при керуванні турбулентністю, керуванні лазерами та керуванні хаосом в плазмі.

Розділ 5 Підсумки

У даній роботі було висвітлено 4 основні задачі керування хаотичними системами: задача стабілізації нестійкої періодичної орбіти, задача генерації хаотичних коливань, задача керування синхронізацією та модифікація атракторів. Було розглянуто три алгоритми вирішення задач стабілізації нестійкої періодичної орбіти, та детально розглянуто принцип роботи одного з методів контроль зі зворотнім зв'язком, а саме алгоритму-першопрохідця - методу контролю OGY, а також проілюстровано його застосування до мапи Хенона.

Ми переконались, що керування хаосом є дійсно можливе, його можна досягти шляхом невеликого збурення параметра хаотичної динамічної системи для того, щоб хаотична орбіта наблизилась до нестабільної періодичної орбіти. Було продемонстровано, що цього дійсно можна досягти, так як будь-яка хаотична система є сильно чутливою до початкових умов, має ергодичну поведінку та складається з нескінченної кількості як хаотичних, так і періодичних орбіт. Звичайно метод розглянутий в даній роботі дещо відрізняється від початкової ідеї запропонованої його винахідниками, але попри це є не поганим підґрунтям для розуміння основної ідеї керування хаотичними динамічними системами.

На завершення, у роботі було наведено приклади застосування розглянутих методів керування, що вкотре підтверджує, актуальність даної тематики уже тепер та у майбутньому.

Висновок

Я вважаю, що основною задачею науковця-математика, програміста є створення та імплементація різних алгоритмів, метою яких є покращення інших сфер науки та техніки: усунення монотонної роботи, оцифрація ручних обрахунків та вимірювань, а також вирішення проблем непередбачуваних обставин. Теорія керування хаотичними систем - це відносно нова сфера досліджень, початок розвитку якої приписують на 1990 рік. Попри те, що спочатку дослідження були лише теоретичними та не мали практичного застосування, зараз ці алгоритми контролю хаосу у динамічних системах уже змінюють Світ та знайшли застосування у численних галузях науки та техніки, а це безперечно доводить як цінність даної галузі, так і потребу ще глибше і глибше досліджувати та покращувати її.

Після публікації Отто, Гребогі та Йорком своєї праці, дана тематика почала розвиватись із величезною швидкістю. Щороку публікуються сотні нових статей зорієнтованих на створення нових алгоритмів, покращення уже існуючих, а також імплементація цих алгоритмів до вже існуючих проблем. Попри цю динаміку розвитку, варто зазаначити, що навіть досі існує велика кількість прогалин у цій області, так само ще як немає загально сформульованої математичної теорії керування хаосом.

На останок, хотілось би зазначити, у цьому рефераті я лише трішечки змогла заглибитись у теорію керування хаосом, але безсумнівно робота над ним, так само, як і вивчення курсу “Моделювання еволюційних систем”, зробило свій внесок у моє світосприйняття. А саме, такі поняття як, на приклад, трикутник Серпінського, фрактальні структури в цілому, “ефект метелика”, зображення дивного атрактора Лоренца, тощо, з якими кожен, мабуть, стикався раніше, змогли набути уже не філософського трактування, а радше їхнього фундаментального - математичного.

Додаток 1

Скрипт 1. Атрактор Лоренца, проекція на xz та x

(Python 3.8)

```
import numpy as np
from scipy.integrate import solve_ivp
import matplotlib.pyplot as plt

def lorenz(t, xyz):
  x, y, z = xyz
  s, r, b = 10, 28, 8 / 3.  # parameters Lorentz used
  return [s * (y - x), x * (r - z) - y, x * y - b * z]

a, b = 0, 40
t = np.linspace(a, b, 4000)

sol1 = solve_ivp(lorenz, [a, b], [1, 1, 1], t_eval=t)
sol2 = solve_ivp(lorenz, [a, b], [1, 1, 1.00001], t_eval=t)

plt.style.use('seaborn')
plt.plot(sol1.y[0], sol1.y[2])
plt.plot(sol2.y[0], sol2.y[2])
plt.plot(sol1.y[0][-1], sol1.y[2][-1], 'o', color="blue")
plt.plot(sol2.y[0][-1], sol2.y[2][-1], 'o', color="green")
plt.xlabel("$x$")
plt.ylabel("$z$")
plt.style.use('seaborn')
plt.show()
plt.savefig("lorenz_xz.png")
plt.close()

plt.subplot(211)
plt.plot(sol1.t, sol1.y[0])
plt.plot(sol2.t, sol2.y[0])
plt.xlabel("$t$")
plt.ylabel("$x_1(t)$")
plt.subplot(212)
plt.plot(sol1.t, sol1.y[0] - sol2.y[0])
plt.ylabel("$x_1(t) - x_2(t)$")
plt.xlabel("$t$")
plt.style.use('seaborn')
plt.show()
plt.savefig("lorenz_x.png")
```

Скрипт 2. Атрактор Лоренца, тривимірна проекція

(Python 3.8)

```
import numpy as np
from scipy.integrate import solve_ivp
```

```
import matplotlib.pyplot as plt

WIDTH, HEIGHT, DPI = 1000, 750, 100

def lorenz(t, xyz):
  x, y, z = xyz
  s, r, b = 10, 28, 8 / 3.  # parameters Lorentz used
  return [s * (y - x), x * (r - z) - y, x * y - b * z]

a, b, n = 0, 150, 10000
t = np.linspace(a, b, n)

sol1 = solve_ivp(lorenz, [a, b], [1, 1, 1], t_eval=t, dense_output=True)
x, y, z = sol1.sol(t)

fig = plt.figure(facecolor='k', figsize=(WIDTH/DPI, HEIGHT/DPI))
ax = fig.gca(projection='3d')
ax.set_facecolor('k')
fig.subplots_adjust(left=0, right=1, bottom=0, top=1)

s = 10
cmap = plt.cm.winter
for i in range(0,n-s,s):
  ax.plot(x[i:i+s+1], y[i:i+s+1], z[i:i+s+1], color=cmap(i/n), alpha=0.4)

ax.set axis off()
plt.savefig('lorenz.png', dpi=DPI)
plt.show()
```

Скрипт 3. Біфуркаційні діаграми мапи Хенона

(Python 3.8)

```
import numpy as np
import matplotlib.pyplot as plt

def henon_attractor(x, y, a=1.4, b=0.03):
  dx = 1 - a * x ** 2 + y
  dy = b * x
  return dx, dy


def logistic_eq(r, x, y):
  dx, dy = henon_attractor(x, y, a=1.4, b=r)
  return dx

def bifurcation_diagram(seed, n_skip, n_iter, step=0.0001, r_min=0):
  print("Starting with x0 seed {0}, skip plotting first {1} iterations, "
        "then plot next {2} iterations.".format(seed, n_skip, n_iter))
  R, X, Y = [], [], []
  r_range = np.linspace(r_min, 4, int(1 / step))
  for r in r_range:
      x = seed
      y = seed
```

```
        for i in range(n_iter + n_skip + 1):
            if i >= n_skip:
                R.append(r)
                X.append(x)
                Y.append(y)
            x, y = henon_attractor(x, y, a=r, b=0.3)

    plt.style.use('seaborn')
    plt.figure(figsize=(10, 5))
    plt.plot(R, X, ls='', marker=',')

    plt.ylim(-1.5, 1.5)
    plt.xlim(r_min, 1.5)
    plt.xlabel('a')
    plt.ylabel('X')
    plt.show()

bifurcation_diagram(1.5, 100, 200)
```

Скрипт 4. Атрактор мапи Хенона

(Python 3.8)

```
import numpy as np
import matplotlib.pyplot as plt

def henon_attractor(x, y, a=1.4, b=0.3):
    x_next = 1 - a * x ** 2 + y
    y_next = b * x
    return x_next, y_next

steps = 100000
X = np.zeros(steps + 1)
Y = np.zeros(steps + 1)

X[0], Y[0] = 0, -1
for i in range(steps):
    x_next, y_next = henon_attractor(X[i], Y[i])
    X[i + 1] = x_next
    Y[i + 1] = y_next

plt.style.use('seaborn')
plt.plot(X, Y, '^', markersize=0.8)
plt.xlabel("$x$")
plt.ylabel("$y$")
plt.show()
plt.close()
```

Скрипт 5. Басейн притягання мапи Хенона

(Python 3.8)

```
import numpy as np
import matplotlib.pyplot as plt
import copy

def henon_boundary(max_iterations, a, b):
  x_range = 2000
  y_range = 2000

  x_list = np.arange(-5, 5, 10 / x_range)
  y_list = np.arange(5, -5, -10 / y_range)
  array = np.meshgrid(x_list, y_list)

  x2 = np.zeros(x_range)
  y2 = np.zeros(y_range)
  iterations_until_divergence = np.meshgrid(x2, y2)

  for i in iterations_until_divergence:
      for j in i:
          j += max_iterations

  not_already_diverged = array[0] < 1000
  for k in range(max_iterations):
      array_copied = copy.deepcopy(array[0])

      array[0] = 1 - a * array[0] ** 2 + array[1]
      array[1] = b * array_copied

      r = (array[0] ** 2 + array[1] ** 2) ** 0.5
      diverging = r > 10
      diverging_now = diverging & not_already_diverged
      iterations_until_divergence[0][diverging_now] = k
      not_already_diverged = np.invert(diverging_now) & not_already_diverged

      array[0][diverging] = 0
      array[1][diverging] = 0
   return iterations_until_divergence[0]

def henon_attractor(x, y, a=1.4, b=0.3):
  x_next = 1 - a * x ** 2 + y
  y_next = b * x
  return x_next, y_next

steps = 100000
X = np.zeros(steps + 1)
Y = np.zeros(steps + 1)
X[0], Y[0] = 0, -1
for i in range(steps):
  x_next, y_next = henon_attractor(X[i], Y[i])
  X[i + 1] = x_next
  Y[i + 1] = y_next

plt.plot(X, Y, ',', alpha = 0.8, markersize=0.3)
henon_boundary = henon_boundary(70, a=1.4, b=0.3)
```

```
plt.imshow(henon_boundary, extent=[-5, 5, -5, 5], alpha=1)
plt.xlabel("$x$")
plt.ylabel("$y$")
plt.plot(-1.1314, -0.3394, 'o', color="red")
plt.plot(0.6314, 0.1894, 'o', color="red")
plt.show()
plt.savefig('Henon_boundary.png', dpi=300)
plt.close()
```

Скрипт 6. Застосування алгоритму OGY до мапи Хенона *(MATLAB)*

```
a = 1.4; b = 0.3;
iter = 100000;
% Parameters
p = a;
dp_grens = 0.15;
fp = max(roots([a (1-b) -1])) % Theoretical fixed point
% Linearize:
A = [-2*a*fp , 1;
b , 0];
B = [0 ; fp];
lam = eig(A); % Eigenvalues of A
% Setting unstable eigenvalues to zero
i = 1;
while i <= length(lam);
if abs(lam(i)) >= 1
k(i) = 0; i=i+1;
else k(i) = lam(i); i=i+1;
End
End
% Pole placement
C = ctrb(A,B);
if size(B,1) == rank(C) % Controllability demand
K = acker(A,B,k);
K = K';
else error("Matrices A and B are not controllable!")
End
% Application to 1000 iterations
Xfp = [fp;b*fp]
X = [0;0]; % Initial condition
regelstap = [0];
a = p;
for i = 1:1000;
dp = -K'*(X(:,i)-Xfp);
if abs(dp) <= dp_grens
a = p + dp;
regelstap = [regelstap , dp];
else dp = 0; a = p + dp;
regelstap = [regelstap , dp];
End
X(1,i+1) = X(2,i)+1-a*X(1,i)^2;
X(2,i+1) = b*X(1,i);
End
```

```
tijd_X = 0:(length(X)-1);
% Visualization
figure(1)
subplot(2,1,1)
hold on, grid
plot(tijd_X,X(1,:),'-','MarkerSize',5)
ylabel('Response x','FontSize',15)
subplot(2,1,2)
hold on, grid
plot(tijd_X,regelstap,'r-','MarkerSize',5)
xlabel('timestep','FontSize',15),
ylabel('Control effort \deltaa','FontSize',15);
```

Список літератури

1. Andrievskii , B. R., & Fradkov, A. L. (2003, November 5). *Control of Chaos: Methods and Applications. I. Methods*. http://engineering.nyu.edu/mechatronics/Control_Lab/bck/VKapila/Chaotic%20Ref/Porfiri's/Biblio/andrievskii02.pdf

2. Andrievskii , B. R., & Fradkov, A. L. (2004, November 4). *Control of Chaos: Methods and Applications. II. Applications*. https://www.ipme.ru/ipme/labs/ccs/andr/af_at04.pdf

3. BOCCALETTI, S., GREBOGI, C., LAI , Y.-C., MANCINI, H., & MAZA, D. (2000, June). *THE CONTROL OF CHAOS: THEORY AND APPLICATIONS*. https://www.researchgate.net/publication/2435257_The_Control_Of_Chaos_Theory_And_Applications

4. Witvoet, J. (2005, March 31). *Control of Chaotic Dynamical Systems using OGY*. https://is.muni.cz/el/1431/jaro2016/M6201/um/OGY.pdf

5. Jentoft, & Li. (2008, October 19). *Stabilizing the H´enon Map with the OGY Algorithm*. https://is.muni.cz/do/rect/el/estud/prif/js12/dynamika/publikace/controlling_chaos.pdf

6. Fradkov , A. L. (n.d.). *CONTROL OF CHAOTIC SYSTEMS*. https://www.eolss.net/sample-chapters/C18/E6-43-22-03.pdf

7. Вовк, В. Д. (2015). *Моделювання еволюційних систем*