令和 5 年度 博士論文

題目

Numerical Approaches for Nonlinear Singular Systems —Hidden Dynamics and Slow–Fast Dynamics—

報告者

天羽 晟矢

徳島大学 大学院 先端技術科学教育部
博士後期課程 システム創生工学専攻

徳島大学 大学院 先端技術科学教育部 システム創生工学専攻 博士論文	
受付年月日	令和 6 年 1 月 15 日
指導教員	上田 哲史
審査担当教員	西尾 芳文
	松浦 健二
	上田 哲史

Contents

Chapter 1

Introduction

The time evolution of dynamical systems changes with system parameters, often generating equilibrium points and periodic solutions. These states have stability, which can be categorized as stable or unstable. By discussing the stability of states within a system, it can be determined which stable state a given initial state will converge to. When system parameters change, certain equilibrium states may alter their stability, a phenomenon known as bifurcation. Identifying parameters where bifurcation occurs is crucial for understanding the global behavior of a system. By computing the set of parameters where bifurcation phenomena occur, it is possible to avoid or induce specific responses.

Since the 19th century, analytical and numerical solutions for dynamical systems have been discussed. However, the visualization of specific system responses became more active only after the development of computer science. When nonlinear dynamical systems exhibit strong nonlinearity, their solutions cannot be analytically determined. Therefore, numerical computation is often the primary approach for almost all dynamical systems. Despite active research in recent years, some areas still lack well-established numerical methods. This includes hidden dynamical systems and slow–fast dynamical systems.

Hidden dynamical systems appear to have equilibrium states typical of regular dynamical systems. However, there can be attractors not observable unless initial values are set in locations far from or within an extremely small range of these attractors, calledd hidden attractors. Although typical methods of dynamical systems apply to hidden attractor computations, the difficulty lies in discovering their basins of attraction. Such attractors can occur in systems like

Chua's circuit or certain Lorentz equations and may also be found in discrete dynamical systems like types of Hénon maps.

Slow–fast dynamical systems include subsystems with extremely different time constants, defined as very small, introducing singular perturbations. These perturbations can lead to oscillatory states corresponding to Hard oscillations, rendering standard numerical methods for regular dynamical systems useless. Furthermore, these systems can display periodic solution responses, known as canards, which are unusual for regular dynamical systems and stem from the singular perturbations, making the application of traditional numerical methods challenging. Historically, the van der Pol oscillator and FitzHugh-Nagumo equations correspond to slow–fast systems. Despite being well-known, specific bifurcation structures in related systems are rarely demonstrated due to numerical difficulties.

This thesis aims to examine effective numerical solutions targeting hidden dynamics and slow–fast dynamics, revealing specific system responses and bifurcation structures. Chapter 2 revisits the mathematical definition of regular dynamical systems and explains basic numerical and bifurcation computation methods, covering both continuous-time autonomous and discrete-time autonomous systems. Chapter 3 discusses efficient implementation methods in Python and C++ for the bifurcation computation methods shown in Chapter 2, introducing new implementation approaches for bifurcation computation programs using automatic differentiation and Python-specific notation. Chapter 4 uses the numerical methods from Chapters 2 and 3 to discuss the bifurcation structure and hidden dynamics of the generalized Hénon map. This chapter explains hidden dynamics using only the classical computational methods of dynamical systems and visualizes the system's bifurcation structure and hidden attractors. In Chapter 5, the discussion moves to the multivibrator, an electronic circuit with multiple modes, where mode switching is explained through bifurcation. However, the numerical methods shown in Chapters 2 and 3 are challenging to apply to slow–fast systems. This chapter also explains numerical methods for calculating canards and their occurrence parameters, demonstrating specific bifurcation structures and the presence of canards. Finally, Chapter 6 summarizes our research, outstanding issues, and future works.

Chapter 2

Mathematical preliminaries

In this chapter, we discuss the mathematical preliminaries for the dynamical systems used in this study, covering both continuous-time autonomous systems and discrete-time autonomous systems. Additionally, we derive the bifurcation conditions necessary for calculating the bifurcation points of these systems.

2.1 Discrete dynamical systems

Consider the n-dimensional discrete autonomous dynamical system:

$$\boldsymbol{x}_{k+1} = T(\boldsymbol{x}_k, \lambda), \;\; T : \boldsymbol{R}^n \times \boldsymbol{R} \to \boldsymbol{R}^n, \, \boldsymbol{x}_k \in \boldsymbol{R}^n, \, \lambda \in \boldsymbol{R}, \tag{2.1}$$

where, $k \in \boldsymbol{Z}$, $\boldsymbol{x}_k$ is the state and λ is a parameter. The map T is assumed to be of class C^∞. Assume that the initial value $\boldsymbol{x}_0$ is given at $k = 0$. The dynamics are obtained as

$$\begin{aligned} \boldsymbol{x}_1 &= T(\boldsymbol{x}_0, \lambda) \\ \boldsymbol{x}_2 &= T(T(\boldsymbol{x}_0, \lambda), \lambda) = T^2(\boldsymbol{x}_0, \lambda) \\ \boldsymbol{x}_3 &= T(T(T(\boldsymbol{x}_0, \lambda), \lambda), \lambda) = T^3(\boldsymbol{x}_0, \lambda) \\ &\vdots, \end{aligned} \tag{2.2}$$

and we denote the composite map as $T^\ell = \overbrace{T \circ T \circ \cdots \circ T}^{\ell}$ for a positive integer ℓ. The state $\boldsymbol{x}_0$ satisfies the condition

$$T^\ell(\boldsymbol{x}_0, \lambda) = \boldsymbol{x}_0 \tag{2.3}$$

is called a periodic point. In particular, $\boldsymbol{x}_0$ is a fixed point for $\ell = 1$.

The variation of the parameter λ often leads to bifurcation phenomena in dynamical systems. Bifurcation phenomena are classified into two types: global bifurcation and local bifurcation. In this study, we only discuss with local bifurcations.

A dynamical system's local bifurcation refers to a situation where a small change in the parameters of the system causes a sudden qualitative or topological change in its behavior. Local bifurcations occur when a periodic point $\boldsymbol{x}_0$ satisfies the following condition:

$$\chi(\boldsymbol{x}_0, \lambda) = \det\left(\frac{\partial T^\ell}{\partial \boldsymbol{x}}(\boldsymbol{x}_0, \lambda) - \mu I^n\right) = 0, \tag{2.4}$$

where, μ is a characteristic constant, I^n is $n \times n$ identity matrix. Therefore, by combining equations (2.3) and (2.4), we obtain the conditions for bifurcation. There are several types of local bifurcations, and in this study, we deal with period-doubling bifurcations, tangent bifurcations, and Neimark-Sacker bifurcations.

Due to their algebraic complexity, these equations are difficult to solve analytically. In this study, we use Newton's method to solve these objective functions. The Newton method has the advantage of quadratic convergence properties. However, it requires the preparation of the Jacobian matrix of the objective function. The following subsections describe each bifurcation condition and the method for obtaining their respective Jacobian matrices.

2.1.1 Period-doubling bifurcation

Period-doubling bifurcation occur when $\mu = -1$. Therefore, the bifurcation condition is:

$$\begin{cases} T^\ell(\boldsymbol{x}_0, \lambda) - \boldsymbol{x}_0 = \boldsymbol{0} \\ \chi(\boldsymbol{x}_0, \lambda) = \det\left(\dfrac{\partial T^\ell}{\partial \boldsymbol{x}}(\boldsymbol{x}_0, \lambda) + I^n\right) = 0 \end{cases}. \tag{2.5}$$

The jacobian matrix for Newton's method is:

$$\begin{pmatrix} \dfrac{\partial T^\ell}{\partial \boldsymbol{x}_0}(\boldsymbol{x}_0, \lambda) & \dfrac{\partial T^\ell}{\partial \lambda}(\boldsymbol{x}_0, \lambda) \\ \dfrac{\partial \chi}{\partial \boldsymbol{x}_0}(\boldsymbol{x}_0, \lambda) & \dfrac{\partial \chi}{\partial \lambda}(\boldsymbol{x}_0, \lambda) \end{pmatrix}. \tag{2.6}$$

Here, preparing the Jacobian matrix requires differentiation with respect to the state and parameters of T^ℓ. There are two methods to obtain these: solving variational equations and using composite mapping. Both methods will be introduced here, but the former is simpler for typical discrete-time dynamical systems. Note that the method using composite mapping can be applied to hybrid dynamical systems, although its implementation is more challenging.

First, we discuss the method using variational equations. If $\boldsymbol{x}_0$ is the initial value and the solution that returns to $\boldsymbol{x}_0$ after ℓ time steps is denoted by $\boldsymbol{\varphi}$, then we have:

$$\begin{aligned} \boldsymbol{x}_k &= \boldsymbol{\varphi}(k, \boldsymbol{x}_0, \lambda), \\ \boldsymbol{x}_{k+\ell} &= \boldsymbol{\varphi}(k+\ell, \boldsymbol{x}_0, \lambda) = \boldsymbol{x}_k, \\ \boldsymbol{\varphi}(0, \boldsymbol{x}_0, \lambda) &= \boldsymbol{x}_0. \end{aligned} \tag{2.7}$$

From $\boldsymbol{\varphi}(k+1, \boldsymbol{x}_0, \lambda) = T(\boldsymbol{\varphi}(k, \boldsymbol{x}_0, \lambda), \lambda)$, we derive:

$$\begin{aligned} \frac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0}(k+1, \boldsymbol{x}_0, \lambda) &= \frac{\partial T}{\partial \boldsymbol{x}}(\boldsymbol{\varphi}(k, \boldsymbol{x}_0, \lambda), \lambda) \frac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0}(k, \boldsymbol{x}_0, \lambda), \\ \frac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0}(0, \boldsymbol{x}_0, \lambda) &= I^n. \end{aligned} \tag{2.8}$$

Here, I is the identity matrix in $\boldsymbol{R}^n \times \boldsymbol{R}^n$. Similarly, the variational equation with respect to the

parameter is:

$$\begin{aligned}\frac{\partial \boldsymbol{\varphi}}{\partial \lambda}(k+1, \boldsymbol{x}_0, \lambda) &= \frac{\partial T}{\partial \boldsymbol{x}}(\boldsymbol{\varphi}(k, \boldsymbol{x}_0, \lambda), \lambda) \frac{\partial \boldsymbol{\varphi}}{\partial \lambda}(k, \boldsymbol{x}_0, \lambda) + \frac{\partial T}{\partial \lambda}(\boldsymbol{\varphi}(k, \boldsymbol{x}_0, \lambda), \lambda), \\ \frac{\partial \boldsymbol{\varphi}}{\partial \lambda}(0, \boldsymbol{x}_0, \lambda) &= \mathbf{0}.\end{aligned} \tag{2.9}$$

Differentiating Eq. (2.8) with respect to $\boldsymbol{x}_0$:

$$\begin{aligned}\frac{\partial^2 \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0^2}(k+1, \boldsymbol{x}_0, \lambda) &= \frac{\partial^2 T}{\partial \boldsymbol{x}^2}(\boldsymbol{\varphi}(k, \boldsymbol{x}_0, \lambda), \lambda) \frac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0}(k, \boldsymbol{x}_0, \lambda)^2 \\ &\quad + \frac{\partial T}{\partial \boldsymbol{x}}(\boldsymbol{\varphi}(k, \boldsymbol{x}_0, \lambda), \lambda) \frac{\partial^2 \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0^2}(k, \boldsymbol{x}_0, \lambda), \\ \frac{\partial^2 \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0^2}(0, \boldsymbol{x}_0, \lambda) &= \boldsymbol{O}.\end{aligned} \tag{2.10}$$

Here, $\boldsymbol{O}$ is the zero matrix in $\boldsymbol{R}^n \times \boldsymbol{R}^n \times \boldsymbol{R}^n$. Similarly, differentiating Eq. (2.9) with respect to $\boldsymbol{x}_0$:

$$\begin{aligned}\frac{\partial^2 \boldsymbol{\varphi}}{\partial \lambda \partial \boldsymbol{x}_0}(k+1, \boldsymbol{x}_0, \lambda) &= \frac{\partial^2 T}{\partial \boldsymbol{x}^2}(\boldsymbol{\varphi}(k, \boldsymbol{x}_0, \lambda), \lambda) \frac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0}(k, \boldsymbol{x}_0, \lambda) \frac{\partial \boldsymbol{\varphi}}{\partial \lambda}(k, \boldsymbol{x}_0, \lambda) \\ &\quad + \frac{\partial T}{\partial \boldsymbol{x}}(\boldsymbol{\varphi}(k, \boldsymbol{x}_0, \lambda), \lambda) \frac{\partial^2 \boldsymbol{\varphi}}{\partial \lambda \partial \boldsymbol{x}_0}(k, \boldsymbol{x}_0, \lambda) \\ &\quad + \frac{\partial^2 T}{\partial \lambda \partial \boldsymbol{x}}(\boldsymbol{\varphi}(k, \boldsymbol{x}_0, \lambda), \lambda) \frac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0}(k, \boldsymbol{x}_0, \lambda), \\ \frac{\partial^2 \boldsymbol{\varphi}}{\partial \lambda \partial \boldsymbol{x}_0}(0, \boldsymbol{x}_0, \lambda) &= O.\end{aligned} \tag{2.11}$$

Here, O is the zero matrix in $\boldsymbol{R}^n \times \boldsymbol{R}^n$. Since $\boldsymbol{\varphi}(k, \boldsymbol{x}_0, \lambda) = \boldsymbol{x}_k = T^k(\boldsymbol{x}_0, \lambda)$,

$$\frac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0}(k, \boldsymbol{x}_0, \lambda) = \frac{\partial T^k}{\partial \boldsymbol{x}_0}, \tag{2.12}$$

and the solutions for the other three variational equations are similar. Thus, by iteratively solving Eq. (2.1) and Eqs. from (2.8) to (2.11) ℓ times, we find the solution to the variational equations, which is then used in the Jacobian matrix for the Newton method.

Next, derivatives of the composite map T_λ^ℓ are derived as:

$$\begin{aligned}
\frac{\partial T_\lambda^\ell}{\partial \boldsymbol{x}} &= \left.\frac{\partial T_\lambda}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}_{\ell-1}} \left.\frac{\partial T_\lambda}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}_{\ell-2}} \cdots \left.\frac{\partial T_\lambda}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}_1} \left.\frac{\partial T_\lambda}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}_0}, \qquad (2.13)\\
\frac{\partial T_\lambda^\ell}{\partial \lambda} &= \left.\frac{\partial T_\lambda}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}_{\ell-1}} \frac{\partial T_\lambda^{\ell-1}}{\partial \lambda} + \left.\frac{\partial T_\lambda}{\partial \lambda}\right|_{\boldsymbol{x}=\boldsymbol{x}_{\ell-1}},\\
\frac{\partial T_\lambda^j}{\partial \lambda} &= \left.\frac{\partial T_\lambda}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}_{j-1}} \frac{\partial T_\lambda^{j-1}}{\partial \lambda} + \left.\frac{\partial T_\lambda}{\partial \lambda}\right|_{\boldsymbol{x}=\boldsymbol{x}_{j-1}}, \qquad (2.14)\\
\frac{\partial T_\lambda^1}{\partial \lambda} &= \left.\frac{\partial T_\lambda}{\partial \lambda}\right|_{\boldsymbol{x}=\boldsymbol{x}_0},
\end{aligned}$$

$$\begin{aligned}
\frac{\partial^2 T_\lambda^\ell}{\partial \boldsymbol{x}^2} = {} & \left.\frac{\partial^2 T_\lambda}{\partial \boldsymbol{x}^2}\right|_{\boldsymbol{x}=\boldsymbol{x}_{\ell-1}} \prod_{k=0}^{\ell-2} \left.\frac{\partial T_\lambda}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}_{\ell-2-k}} \prod_{k=0}^{\ell-2} \left.\frac{\partial T_\lambda}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}_{\ell-2-k}}\\
& + \cdots + \prod_{k=0}^{j+1} \left.\frac{\partial T_\lambda}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}_{\ell-1-k}} \left.\frac{\partial^2 T_\lambda}{\partial \boldsymbol{x}^2}\right|_{\boldsymbol{x}=\boldsymbol{x}_j} \prod_{k=0}^{j-1} \left.\frac{\partial T_\lambda}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}_{j-1-k}} \prod_{k=0}^{j-1} \left.\frac{\partial T_\lambda}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}_{j-1-k}} \qquad (2.15)\\
& + \cdots + \prod_{k=0}^{\ell-2} \left.\frac{\partial T_\lambda}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}_{\ell-1-k}} \left.\frac{\partial^2 T_\lambda}{\partial \boldsymbol{x}^2}\right|_{\boldsymbol{x}=\boldsymbol{x}_0},
\end{aligned}$$

$$\begin{aligned}
\frac{\partial T_\lambda^\ell}{\partial \boldsymbol{x}\partial \lambda} &= \left.\frac{\partial^2 T_\lambda}{\partial \boldsymbol{x}^2}\right|_{\boldsymbol{x}=\boldsymbol{x}_{\ell-1}} \prod_{k=0}^{\ell-2} \left.\frac{\partial T_\lambda}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}_{\ell-2-k}} \frac{\partial T_\lambda^{\ell-1}}{\partial \lambda} + \left.\frac{\partial T_\lambda}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}_{\ell-1}} \frac{\partial T_\lambda^{\ell-1}}{\partial \boldsymbol{x}\partial \lambda} + \left.\frac{\partial T_\lambda}{\partial \boldsymbol{x}\partial \lambda}\right|_{\boldsymbol{x}=\boldsymbol{x}_{\ell-1}} \prod_{k=0}^{\ell-2} \left.\frac{\partial T_\lambda}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}_{\ell-2-k}},\\
\frac{\partial T_\lambda^j}{\partial \boldsymbol{x}\partial \lambda} &= \left.\frac{\partial^2 T_\lambda}{\partial \boldsymbol{x}^2}\right|_{\boldsymbol{x}=\boldsymbol{x}_{j-1}} \prod_{k=0}^{j-2} \left.\frac{\partial T_\lambda}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}_{j-2-k}} \frac{\partial T_\lambda^{j-1}}{\partial \lambda} + \left.\frac{\partial T_\lambda}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}_{j-1}} \frac{\partial T_\lambda^{j-1}}{\partial \boldsymbol{x}\partial \lambda} + \left.\frac{\partial T_\lambda}{\partial \boldsymbol{x}\partial \lambda}\right|_{\boldsymbol{x}=\boldsymbol{x}_{j-1}} \prod_{k=0}^{j-2} \left.\frac{\partial T_\lambda}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}_{j-2-k}}, \qquad (2.16)\\
\frac{\partial T_\lambda^1}{\partial \boldsymbol{x}\partial \lambda} &= \left.\frac{\partial^2 T_\lambda}{\partial \boldsymbol{x}\partial \lambda}\right|_{\boldsymbol{x}=\boldsymbol{x}_0},
\end{aligned}$$

where, $j = \ell-1, \ell-2, \ldots, 3, 2$, $\boldsymbol{x}_k = T_\lambda^k(\boldsymbol{x}_0)$, $T_\lambda(\boldsymbol{x}_0) = T(\boldsymbol{x}_0, \lambda)$. Refer the Ref.[1] to compute tensor product in second variational equations. Since the derivative of the composite map is constructed based on the concept of Ref.[2], bifurcation analysis can be performed for switching systems by appropriately defining the map piecewise.

Next, we determine the differentiation of the determinant χ. If we define the determinant of

a matrix A as follows,

$$|A| = \begin{vmatrix} a_{11} & a_{12} & \cdots & a_{1k} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2k} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nk} & \cdots & a_{nn} \end{vmatrix}, \tag{2.17}$$

the value of the determinant can be expressed as,

$$|A| = \sum_{\sigma \in S_n} \operatorname{sgn}(\sigma) a_{\sigma(1)1} a_{\sigma(2)2} \cdots a_{\sigma(n)n}. \tag{2.18}$$

Here, S_n is the set of all permutations of n elements, σ is a permutation, and $\operatorname{sgn}(\sigma)$ is the sign function that returns 1 or -1 for even or odd permutations, respectively.

The differentiation of $|A|$ with respect to x_i using the multilinearity of the determinant is,

$$\frac{\partial |A|}{\partial x_i} = \sum_{k=1}^{n} \begin{vmatrix} a_{11} & \cdots & a_{1k-1} & \frac{\partial a_{1k}}{\partial x_i} & a_{1k+1} & \cdots & a_{1n} \\ a_{21} & \cdots & a_{2k-1} & \frac{\partial a_{2k}}{\partial x_i} & a_{2k+1} & \cdots & a_{2n} \\ \vdots & & \vdots & & \vdots & & \vdots \\ a_{n1} & \cdots & a_{nk-1} & \frac{\partial a_{nk}}{\partial x_i} & a_{nk+1} & \cdots & a_{nn} \end{vmatrix}. \tag{2.19}$$

Prior to numerical calculation, one should calculate $\frac{\partial^2 T^\ell}{\partial \boldsymbol{x}^2}(\boldsymbol{x}_0, \lambda)$ and $\frac{\partial^2 T^\ell}{\partial \boldsymbol{x} \partial \lambda}(\boldsymbol{x}_0, \lambda)$, and then compute the sum of the determinants of matrices obtained by replacing each column element of $\frac{\partial T^\ell}{\partial \boldsymbol{x}}(\boldsymbol{x}_0, \lambda) - \mu \boldsymbol{I}$ with their respective derivatives.

2.1.2 Tangent bifurcation

Tangent bifurcation occur when $\mu = 1$. Therefore, the bifurcation condition is:

$$\begin{cases} T^{\ell}(\boldsymbol{x}_0, \lambda) - \boldsymbol{x}_0 = \boldsymbol{0} \\ \chi(\boldsymbol{x}_0, \lambda) = \det\left(\dfrac{\partial T^{\ell}}{\partial \boldsymbol{x}}(\boldsymbol{x}_0, \lambda) - I^n\right) = 0 \end{cases} . \tag{2.20}$$

The method for obtaining the Jacobian matrix for the Newton method is same to that used for period-doubling bifurcations. However, we note that since $\mu = 1$, the sign of the identity matrix in the characteristic equation has changed.

2.1.3 Neimark-Sacker bifurcation

Neimark-Sacker bifurcation (NS bifurcation) occurs when $\mu = e^{j\theta}$. As a result, the objective function for the Newton method includes complex numbers, and therefore the same objective function as used for period-doubling and tangent bifurcations cannot be applied. Here, two conditions for NS bifurcation are introduced.

First, we introduce a method that separates the real and imaginary parts of the bifurcation condition to increase the number of objective functions[3]. The NS bifurcation conditions, when separating the real and imaginary parts of the second equation in formula (2.4), are as follows:

$$\begin{cases} T^{\ell}(\boldsymbol{x}_0, \lambda) - \boldsymbol{x}_0 = \boldsymbol{0} \\ \mathrm{Re}(\chi(\boldsymbol{x}_0, \lambda, \theta)) = 0 \\ \mathrm{Im}(\chi(\boldsymbol{x}_0, \lambda, \theta)) = 0 \end{cases} \tag{2.21}$$

To solve these, find solutions for $(\boldsymbol{x}_0, \lambda, \theta)$. The Jacobian matrix required for the Newton method

is:

$$\begin{pmatrix} \dfrac{\partial T^\ell}{\partial \boldsymbol{x}}(\boldsymbol{x}_0, \lambda) & \dfrac{\partial T^\ell}{\partial \lambda}(\boldsymbol{x}_0, \lambda) & \dfrac{\partial T^\ell}{\partial \theta}(\boldsymbol{x}_0, \lambda) \\ \mathrm{Re}\left(\dfrac{\partial \chi}{\partial \boldsymbol{x}}(\boldsymbol{x}_0, \lambda, \theta)\right) & \mathrm{Re}\left(\dfrac{\partial \chi}{\partial \lambda}(\boldsymbol{x}_0, \lambda, \theta)\right) & \mathrm{Re}\left(\dfrac{\partial \chi}{\partial \theta}(\boldsymbol{x}_0, \lambda, \theta)\right) \\ \mathrm{Im}\left(\dfrac{\partial \chi}{\partial \boldsymbol{x}}(\boldsymbol{x}_0, \lambda, \theta)\right) & \mathrm{Im}\left(\dfrac{\partial \chi}{\partial \lambda}(\boldsymbol{x}_0, \lambda, \theta)\right) & \mathrm{Im}\left(\dfrac{\partial \chi}{\partial \theta}(\boldsymbol{x}_0, \lambda, \theta)\right) \end{pmatrix}. \tag{2.22}$$

$\partial T^\ell / \partial \theta$ is $\mathbf{0}$ since T does not depend on θ. Also,

$$\frac{\partial}{\partial \theta}\left(\frac{\partial T^\ell}{\partial \boldsymbol{x}}(\boldsymbol{x}_0, \lambda) - e^{j\theta} I^n\right) = (\sin\theta - j\cos\theta) I^n. \tag{2.23}$$

Note that the second and third rows of the Jacobian matrix involve swapping the operations of differentiation and extracting real and imaginary parts, Re and Im. This is trivial because a complex function $f(z) \in \boldsymbol{C}, z = x + jy$ can be expressed as the sum of real functions $u, v \in \boldsymbol{R}$, where $f(z) = u(x) + jv(y)$.

Next, we show the bifurcation condition using the bialternate product. The real-imaginary part separation algorithm can be challenging to implement in programming languages that do not handle complex numbers, especially in deriving the Jacobian matrix. Moreover, the increase in the number of objective functions for the Newton method also raises implementation complexity. By using the bialternate product, the number of objective functions remains the same as in the case of period-doubling or tangent bifurcations, and there is no need to handle the implicit variable θ.

Assume $A, B \in \boldsymbol{R}^{n\times n}$. The bialternate product[4] denoted $(A \odot B)$ is defined as:

$$(A \odot B)_{(p,q),(r,s)} = \frac{1}{2}\left\{\begin{vmatrix} a_{pr} & a_{ps} \\ b_{qr} & b_{qs} \end{vmatrix} + \begin{vmatrix} b_{pr} & b_{ps} \\ a_{qr} & a_{qs} \end{vmatrix}\right\} \tag{2.24}$$

whose rows are labeled by the multi-index (p, q) $(p = 2, 3, \ldots, n;\ q = 1, 2, \ldots, p-1)$, and columns are labeld by the multi-index (r, s) $(r = 2, 3, \ldots, n;\ s = 1, 2, \ldots, r-1)$. The size of bialternate product is $m \times m$, where $m = n(n-1)/2$. For example, if $n = 3$, then we get $m = 3$

and

$$(A \odot B) = \begin{pmatrix} (A \odot B)_{(2,1),(2,1)} & (A \odot B)_{(2,1),(3,1)} & (A \odot B)_{(2,1),(3,2)} \\ (A \odot B)_{(3,1),(2,1)} & (A \odot B)_{(3,1),(3,1)} & (A \odot B)_{(3,1),(3,2)} \\ (A \odot B)_{(3,2),(2,1)} & (A \odot B)_{(3,2),(3,1)} & (A \odot B)_{(3,2),(3,2)}. \end{pmatrix}$$

Bialternate product holds the following properties:

1. $A \odot A$ has eigenvalues $\lambda_i \lambda_j$.

2. $2A \odot I^n$ has eigenvalues $\lambda_i + \lambda_j$.

Therefore, we have Neimark-Sacker bifurcation condition as:

$$\begin{cases} T^{\ell}(\boldsymbol{x}_0, \lambda) - \boldsymbol{x}_0 = \mathbf{0} \\ \chi(\boldsymbol{x}, \lambda) = \det\left(\dfrac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0} \odot \dfrac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0} - I^m \right) = 0, \end{cases} \tag{2.25}$$

where I^m is identity matrix of $R^{m \times m}$. This has the type of bialternate product

$$(\boldsymbol{A} \odot \boldsymbol{A})_{(p,q),(r,s)} = \begin{vmatrix} a_{pr} & a_{ps} \\ a_{qr} & a_{qs} \end{vmatrix}. \tag{2.26}$$

Derivatives of each elements are drived by:

$$\frac{\partial}{\partial x_i}(\boldsymbol{A} \odot \boldsymbol{A})_{(p,q),(r,s)} = \begin{vmatrix} \dfrac{\partial a_{pr}}{\partial x_i} & a_{ps} \\ \dfrac{\partial a_{qr}}{\partial x_i} & a_{qs} \end{vmatrix} + \begin{vmatrix} a_{pr} & \dfrac{\partial a_{ps}}{\partial x_i} \\ a_{qr} & \dfrac{\partial a_{qs}}{\partial x_i} \end{vmatrix} \tag{2.27}$$

2.2 Continuous dynamical systems

Consider the initial value problem of an continuous autonomous dynamical system:

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{f}(\boldsymbol{x}, \lambda), \quad \text{with} \quad \boldsymbol{x}(0) = \boldsymbol{x}_0, \tag{2.28}$$

where, $\boldsymbol{x} \in \boldsymbol{R}^n$ represents the state variables, and $\lambda \in \boldsymbol{R}$ denotes the parameters. The function $\boldsymbol{f} : \boldsymbol{R}^n \to \boldsymbol{R}^n$ is assumed to be of class C^∞.

2.2.1 Equilibrium point and its bifurcations

An equilibrium point is a point $\boldsymbol{x}^*$ that satisfies:

$$\boldsymbol{f}(t, \boldsymbol{x}^*, \lambda) = \boldsymbol{0}. \tag{2.29}$$

Bifurcation of equilibrium points occurs when the real part of the eigenvalue μ is zero. These are respectively referred to as Tangent Bifurcation when $\mathrm{Im}(\mu) = 0$, and Hopf Bifurcation when $\mathrm{Im}(\mu) \neq 0$. Therefore, the bifurcation condition for equilibrium points is:

$$\begin{cases} \boldsymbol{f}(t, \boldsymbol{x}^*, \lambda) = \boldsymbol{0} \\ \mathrm{Re}(\chi(\mu)) = \mathrm{Re}\left(\det\left(\left.\dfrac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}^*} - \mu I^n\right)\right) = 0 \end{cases}. \tag{2.30}$$

This needs to be solved for $(\boldsymbol{x}^*, \lambda)$. The Jacobian matrix required for the Newton method is:

$$\begin{pmatrix} \dfrac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}} & \dfrac{\partial \boldsymbol{f}}{\partial \lambda} \\ \mathrm{Re}\left(\dfrac{\partial \chi}{\partial \boldsymbol{x}}\right) & \mathrm{Re}\left(\dfrac{\partial \chi}{\partial \lambda}\right) \end{pmatrix}. \tag{2.31}$$

In languages that cannot handle complex numbers, it is easier to use bifurcation conditions using the bialternate product, similar to NS bifurcation in discrete dynamical systems. For Hopf bifurcation, since the real parts of conjugate complex numbers satisfy $\mu_i + \mu_j = 0$, the bifurcation condition is:

$$\det\left(2\left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}^*} \bigcirc I^n\right) = 0. \tag{2.32}$$

The eigenvalue sum property of $2A \bigcirc I^n$ is particularly useful and has a special name, "Biproduct".

The elements of $2A \odot I^n$ are:

$$(2A \odot I^n)_{(p,q),(r,s)} = \begin{vmatrix} a_{pr} & a_{ps} \\ \delta_{qr} & \delta_{qs} \end{vmatrix} + \begin{vmatrix} \delta_{pr} & \delta_{ps} \\ a_{qr} & a_{qs} \end{vmatrix}, \tag{2.33}$$

where δ_{ij} is the Kronecker delta. Expanding this, we get:

$$(2A \odot I^n)_{(p,q),(r,s)} = \begin{cases} -a_{ps} & \text{if} \quad r = q, \\ a_{pr} & \text{if} \quad r \neq p \quad \text{and} \quad s = q, \\ a_{pp} + a_{qq} & \text{if} \quad r = p \quad \text{and} \quad s = q, \\ a_{qs} & \text{if} \quad r = p \quad \text{and} \quad s \neq q, \\ -a_{qr} & \text{if} \quad s = p, \\ 0 & \text{otherwise.} \end{cases} \tag{2.34}$$

The derivatives can be found by directly differentiating these elements.

2.2.2 Fixed point

When a solution function $\boldsymbol{\varphi}(t, \boldsymbol{x}_0, \lambda)$ satisfies $\boldsymbol{\varphi}(\tau, \boldsymbol{x_0}, \lambda) = \boldsymbol{x}_0$ with a period τ, it is called a periodic solution. Poincaré section is used for stability analysis of periodic solutions:

$$\Pi = \{\boldsymbol{x} \in \boldsymbol{R}^n \mid q(\boldsymbol{x}) = 0\}. \tag{2.35}$$

If a periodic solution exists in the system, defined as

$$\begin{aligned} T : \Pi &\to \Pi \\ \boldsymbol{x}_0 &\mapsto T(\boldsymbol{x}_0) = \boldsymbol{\varphi}(\tau, \boldsymbol{x}_0, \lambda), \end{aligned} \tag{2.36}$$

then the stability of the periodic solution can be reduced to the stability of fixed points of the discrete dynamical system $\boldsymbol{x}_{k+1} = T(\boldsymbol{x}_k)$ on the Poincaré section Π.

The fixed-point conditions are given by:

$$\begin{cases} \boldsymbol{\varphi}(\tau, \boldsymbol{x}_0, \lambda) - \boldsymbol{x}_0 & = 0 \\ q(\boldsymbol{\varphi}(\tau, \boldsymbol{x}_0, \lambda)) & = 0 \end{cases} \tag{2.37}$$

For periodic solutions in autonomous systems, the period τ changes with the parameters, so the second equation is necessary. It represents the condition that when the initial value $\boldsymbol{x}_0$ departs from Π, the solution reaches the Poincaré section again. When using Eq. (2.37) as the objective function for Newton's method for $(\boldsymbol{x}_0, \tau)$.

2.2.3 Bifurcations of a fixed point

A bifurcation occurs when the characteristic constants of the Poincaré map obtained from Eq. (2.37) satisfy $|\mu_i| = 1$, where $\mu_i, i = 1, \ldots, n$ is the characteristic constants of the Poincaré map. The bifurcation condition is given by[5]:

$$\chi(\boldsymbol{x}, \lambda) = \det\left(\frac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0} - \mu I^n\right) = 0. \tag{2.38}$$

the bifurcation point is obtained by solving Eq.(2.37) and (2.38) for $(\boldsymbol{x}_0, \tau, \lambda)$ simultaneously. The Jacobian matrix for bifurcation calculation is given by:

$$\begin{pmatrix} \dfrac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0} - I^n & \boldsymbol{f}(\boldsymbol{\varphi}) & \dfrac{\partial \boldsymbol{\varphi}}{\partial \lambda} \\ \dfrac{\partial q}{\partial \boldsymbol{x}} \dfrac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0} & \dfrac{\partial q}{\partial \boldsymbol{x}} \boldsymbol{f}(\boldsymbol{\varphi}) & \dfrac{\partial q}{\partial \boldsymbol{x}} \dfrac{\partial \boldsymbol{\varphi}}{\partial \lambda} \\ \dfrac{\partial \chi}{\partial \boldsymbol{x}_0} & \dfrac{\partial \chi}{\partial \tau} & \dfrac{\partial \chi}{\partial \lambda} \end{pmatrix}, \tag{2.39}$$

we describe $\boldsymbol{\varphi}(\tau, \boldsymbol{x}_0, \lambda)$ as just $\boldsymbol{\varphi}$ for simplicity.

The initial value and parameter derivatives of $\boldsymbol{\varphi}$ appearing in Eq. (2.39) are solutions of

variational equations[6]. The solution function $\boldsymbol{\varphi}$ satisfies:

$$\begin{aligned} \boldsymbol{x}(t) &= \boldsymbol{\varphi}(t, \boldsymbol{x}_0, \lambda), \\ \boldsymbol{x}(t+\tau) &= \boldsymbol{\varphi}(t+\tau, \boldsymbol{x}_0, \lambda) = \boldsymbol{x}(t), \\ \boldsymbol{x}(0) &= \boldsymbol{x}_0. \end{aligned} \tag{2.40}$$

From equations (2.28) and (2.40),

$$\frac{d\boldsymbol{\varphi}}{dt}(t, \boldsymbol{x}_0, \lambda) = \boldsymbol{f}(t, \boldsymbol{\varphi}(t, \boldsymbol{x}_0, \lambda), \lambda). \tag{2.41}$$

Differentiating Eq. (2.41) with respect to $\boldsymbol{x}_0$ yields,

$$\frac{\partial}{\partial \boldsymbol{x}_0}\frac{d\boldsymbol{\varphi}}{dt}(t, \boldsymbol{x}_0, \lambda) = \frac{\partial}{\partial \boldsymbol{x}_0}\boldsymbol{f}(t, \boldsymbol{\varphi}(t, \boldsymbol{x}_0, \lambda), \lambda). \tag{2.42}$$

Assuming Eq. (2.28) is sufficiently smooth, the order of differentiation can be swapped. Applying the derivative of a composite function,

$$\frac{d}{dt}\frac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}(t, \boldsymbol{\varphi}(t, \boldsymbol{x}_0, \lambda), \lambda)\frac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0}. \tag{2.43}$$

Similarly, differentiating Eq. (2.41) with respect to λ and rearranging (using the differentiation of composite and multivariable functions),

$$\frac{d}{dt}\frac{\partial \boldsymbol{\varphi}}{\partial \lambda} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}(t, \boldsymbol{\varphi}(t, \boldsymbol{x}_0, \lambda), \lambda)\frac{\partial \boldsymbol{\varphi}}{\partial \lambda} + \frac{\partial \boldsymbol{f}}{\partial \lambda}(t, \boldsymbol{\varphi}(t, \boldsymbol{x}_0, \lambda), \lambda). \tag{2.44}$$

This provides the first variational equations for the initial value and parameter. Next, differentiating Eq. (2.43) with respect to $\boldsymbol{x}_0$,

$$\frac{d}{dt}\frac{\partial^2 \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0^2} = \frac{\partial^2 \boldsymbol{f}}{\partial \boldsymbol{x}^2}(t, \boldsymbol{\varphi}(t, \boldsymbol{x}_0, \lambda), \lambda)\frac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0}^2 + \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}(t, \boldsymbol{\varphi}(t, \boldsymbol{x}_0, \lambda), \lambda)\frac{\partial^2 \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0^2}. \tag{2.45}$$

Also, differentiating Eq. (2.44) with respect to $\boldsymbol{x}_0$,

$$\begin{aligned}\frac{d}{dt}\frac{\partial^2\boldsymbol{\varphi}}{\partial\lambda\partial\boldsymbol{x}_0} = &\frac{\partial^2\boldsymbol{f}}{\partial\boldsymbol{x}^2}(t,\boldsymbol{\varphi}(t,\boldsymbol{x}_0,\lambda),\lambda)\frac{\partial\boldsymbol{\varphi}}{\partial\boldsymbol{x}_0}\frac{\partial\boldsymbol{\varphi}}{\partial\lambda}\\ &+\frac{\partial\boldsymbol{f}}{\partial\boldsymbol{x}}(t,\boldsymbol{\varphi}(t,\boldsymbol{x}_0,\lambda),\lambda)\frac{\partial^2\boldsymbol{\varphi}}{\partial\lambda\partial\boldsymbol{x}_0}+\frac{\partial^2\boldsymbol{f}}{\partial\lambda\partial\boldsymbol{x}}(t,\boldsymbol{\varphi}(t,\boldsymbol{x}_0,\lambda),\lambda)\frac{\partial\boldsymbol{\varphi}}{\partial\boldsymbol{x}_0}\end{aligned}. \tag{2.46}$$

The differentiation of Eq. (2.43) with respect to λ is complex and not recommended. These variational equations, together with Eq. (2.28), can be numerically integrated simultaneously from time 0 to τ to obtain derivatives with respect to the initial value and parameter.

The second variational equation for period τ can be derived from Eq. (2.41) as:

$$\frac{d}{d\tau}\frac{\partial\boldsymbol{\varphi}}{\partial\boldsymbol{x}_0} = \frac{\partial\boldsymbol{f}}{\partial\boldsymbol{x}}(\tau,\boldsymbol{\varphi}(\tau,\boldsymbol{x}_0,\lambda),\lambda)\frac{\partial\boldsymbol{\varphi}}{\partial\boldsymbol{x}_0}. \tag{2.47}$$

In other words, the variational equation for τ does not need to be integrated numerically; it can be easily obtained from the matrix product of previously calculated $\partial\boldsymbol{f}/\partial\boldsymbol{x}$ and $\partial\boldsymbol{\varphi}/\partial\boldsymbol{x}_0$.

Period-doubling bifurcation

Period doubling bifurcation occurs when $\mu = -1$. We get the period doubling bifurcation condition as:

$$\begin{cases}\boldsymbol{\varphi}(\tau,\boldsymbol{x}_0,\lambda)-\boldsymbol{x}_0 & =0\\ q(\boldsymbol{\varphi}(\tau,\boldsymbol{x}_0,\lambda)) & =0\\ \chi(\boldsymbol{x},\lambda)=\det\left(\dfrac{\partial\boldsymbol{\varphi}}{\partial\boldsymbol{x}_0}+I^n\right) & =0.\end{cases} \tag{2.48}$$

Tangent bifurcation

Tangent bifurcation occurs when $\mu = 1$. However, due to the condition that the fixed point $\boldsymbol{x}_0$ exists on the Poincaré section, one of the characteristic constants is always equal to 1. In other words, the condition $\mu = 1$ in Eq. (2.38) is always satisfied, making it inappropriate as a tangent bifurcation condition. Therefore, the derivative of Eq.(2.38) respect to μ is used as the tangent

bifurcation condition:

$$\begin{cases} \boldsymbol{\varphi}(\tau, \boldsymbol{x}_0, \lambda) - \boldsymbol{x}_0 & = 0 \\ q(\boldsymbol{\varphi}(\tau, \boldsymbol{x}_0, \lambda)) & = 0 \\ \dfrac{d\chi}{d\mu}(\boldsymbol{x}, \lambda) & = 0. \end{cases} \tag{2.49}$$

The third equation of Eq. (2.49) is derived as:

$$\frac{d\chi}{d\mu} = \underbrace{\begin{vmatrix} -1 & a_{12} & \dots & a_{1n} \\ 0 & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & a_{n2} & \dots & a_{nn} \end{vmatrix}}_{①} + \underbrace{\begin{vmatrix} a_{11} & 0 & \dots & a_{1n} \\ a_{21} & -1 & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & 0 & \dots & a_{nn} \end{vmatrix}}_{②} + \cdots + \underbrace{\begin{vmatrix} a_{11} & a_{12} & \dots & 0 \\ a_{21} & a_{22} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & -1 \end{vmatrix}}_{ⓝ} = 0 \tag{2.50}$$

where, $A = \{a_{ij}\} = \partial\varphi/\partial\boldsymbol{x}_0$. Newton's method require the derivative of Eq.(2.50):

$$\frac{\partial}{\partial x_i}\frac{\partial\chi}{\partial\mu} = \frac{\partial ①}{\partial x_i} + \frac{\partial ②}{\partial x_i} + \cdots + \frac{\partial ⓝ}{\partial x_i}. \tag{2.51}$$

For example, the first term of Eq.(2.51) is given by:

$$\frac{\partial ①}{\partial x_i} = \begin{vmatrix} 0 & \dfrac{\partial a_{12}}{\partial x_i} & \dots & \dfrac{\partial a_{1n}}{\partial x_i} \\ 0 & \dfrac{\partial a_{22}}{\partial x_i} & \dots & \dfrac{\partial a_{2n}}{\partial x_i} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \dfrac{\partial a_{n2}}{\partial x_i} & \dots & \dfrac{\partial a_{nn}}{\partial x_i} \end{vmatrix} + \begin{vmatrix} -1 & 0 & \dots & \dfrac{\partial a_{1n}}{\partial x_i} \\ 0 & 0 & \dots & \dfrac{\partial a_{2n}}{\partial x_i} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \dfrac{\partial a_{nn}}{\partial x_i} \end{vmatrix} + \cdots + \begin{vmatrix} -1 & \dfrac{\partial a_{12}}{\partial x_i} & \dots & 0 \\ 0 & \dfrac{\partial a_{22}}{\partial x_i} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \dfrac{\partial a_{n2}}{\partial x_i} & \dots & 0 \end{vmatrix} \tag{2.52}$$

Neimark-Sacker bifurcation

The NS bifurcation condition for continuous systems can be expressed using the Bialternate product, as same as discrete dynamical systems:

$$\begin{cases} \boldsymbol{\varphi}(\tau, \boldsymbol{x}_0, \lambda) - \boldsymbol{x}_0 & = 0 \\ q(\boldsymbol{\varphi}(\tau, \boldsymbol{x}_0, \lambda)) & = 0 \\ \chi(\boldsymbol{x}, \lambda) = \det\left(\dfrac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0} \odot \dfrac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0} - \boldsymbol{I}^m\right) & = 0 \end{cases} \tag{2.53}$$

We note that the real-imaginary part separation algorithm is also applicable for continuous systems, however, since it would duplicate the description for discrete systems, it is not described here.

Chapter 3

Computation of bifurcations: Implementation techniques

In this chapter, we provide techniques for implementing a bifurcation computation program using the bifurcation analysis methods discussed in Chapter 2. This includes interface design, automatic derivation of expressions using symbolic computation libraries, methods for setting objective functions, and discusses specific implementation techniques in Python and C++.

3.1 Introduction

Bifurcation computation software[7][8] has primarily used old implementations in C, FORTRAN or MATLAB. These programs are difficult to modify or understand. We will use the following computing environment:

- Shell: Unix shell (zsh, bash, etc.)
- Language: Python

- Library: `numpy`, `scipy`, `sympy`

The Python language is an interpreted language, which makes it easy to write programs. Additionally, a lot of libraries written in C and FORTRAN are available, making Python an excellent language for general-purpose numerical computations. In this chapter, we will explain techniques using Python, including the implementation of the Newton method, symbolic differentiation, and the representation of variational equations. By properly preparing function objects corresponding to Newton's method, it is possible to easily switch between multiple objective functions. Additionally, the use of symbolic differentiation eliminates the need for manual calculation of the system's derivatives. By automatically determining the derivatives, users hardly need to prepare for bifurcation calculations. Furthermore, using the `numpy` library allows for "very simple" descriptions of complex variational equations.

In some parts of this chapter, techniques for C++ will also be provided. The environment for using C++ is:

- Shell: Unix shell (zsh, bash, etc.)
- Language: C++
- Library: `Eigen-3.4-rc1`, `nlohmann-3.10.1`

Bifurcation calculations require numerical integrations, so C++ may be used for performance considerations. `eigen` is a linear algebra library that can be used by including only the header file. Similarly, `nlohmann` is a library for handling `json` in C++, and it can also be used by including only the header file.

Please refer to the author's GitHub page for detailed implementations on the bifurcation calculation program:

```
https://github.com/aw02m
```

In addition to the Python-based bifurcation calculation program `nonautonomous_bif_python`, bifurcation calculation programs written in C++ such as `autonomous_bif` and `discrete_bif` are also available. The implementation details can be checked on these repositories.

3.2 Newton's method

The Newton method is an algorithm for iteratively solving algebraic equations. The only requirement for the target equation is the differentiability in the root-finding region, and the convergence of this method is very fast, with a quadratic convergence rate. However, compared to other rich root-finding algorithms, the convergence region of the Newton method is quite narrow. Nevertheless, when the calculation succeeds, the results can be more reliable than those obtained using functions like `hybrid` method of `scipy.root`.

Consider the following nonlinear algebraic equation:

$$\boldsymbol{F}(\boldsymbol{u}) = \boldsymbol{0},\ \boldsymbol{u} \in \boldsymbol{R}^n,\ \boldsymbol{F} : \boldsymbol{R}^n \to \boldsymbol{R}^n. \tag{3.1}$$

Let $\boldsymbol{u}_k$ be the kth approximation of $\boldsymbol{u}$. The Taylor expansion around $\boldsymbol{u}_k$ is,

$$\boldsymbol{F}(\boldsymbol{u}) = \boldsymbol{F}(\boldsymbol{u}_k) + \left.\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{u}}\right|_{\boldsymbol{u}=\boldsymbol{u}_k} (\boldsymbol{u} - \boldsymbol{u}_k) + \cdots \tag{3.2}$$

Ignoring the nonlinear terms and using $\boldsymbol{F}(\boldsymbol{u}) = \boldsymbol{0}$,

$$\boldsymbol{F}(\boldsymbol{u}_k) + \left.\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{u}}\right|_{\boldsymbol{u}=\boldsymbol{u}_k} (\boldsymbol{u} - \boldsymbol{u}_k) = \boldsymbol{0}. \tag{3.3}$$

Replacing the solution $\boldsymbol{u}$ with the $k+1$th approximation, we can rearrange it into the following form:

$$\left.\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{u}}\right|_{\boldsymbol{u}=\boldsymbol{u}_k} (\boldsymbol{u}_{k+1} - \boldsymbol{u}_k) = -\boldsymbol{F}(\boldsymbol{u}_k). \tag{3.4}$$

This is in the form of the linear equation $A\boldsymbol{u} = \boldsymbol{b}$. By solving for $(\boldsymbol{u}_{k+1} - \boldsymbol{u}_k)$ using methods such as Gaussian elimination, we can find the difference between the $k+1$th and kth approximate solutions.

Since Newton's method is a simple iterative method, we have designed a custom function,

`newton(func, x0, args=())`, based on the input and output of the `scipy.optimize.root()` function. Certainly, `scipy.optimize.root()` is also available. However, it is important to note that highly convergent methods may converge to false solutions. Therefore, cautions are needed to select initial values and to verify the obtained solutions.

Listing 3.1 shows the implementation of the `newton()` function. `func` represents a multi-valued function that returns the objective function `F` and its Jacobian matrix `J`, `x0` is the initial value given to Newton's method, and `args` is the argument passed to `func`. `NewtonResult` is a class that stores the results of the calculation. It has the following three attributes: `x` is the solution obtained by Newton's method, `success` is a flag that indicates whether the calculation was successful, `eigvals` is the eigenvalues of the Jacobian matrix at the solution.

Listing 3.1: An example of a Newton's method

```
def newton(func, x0, args=(), tol=1e-8, max_iter=16):
  result = NewtonResult()
  for i in range(max_iter):
    F, J, eigvals = func(x0, *args)
    result.eigvals = eigvals
    dx = np.linalg.solve(J, -F)
    x = dx + x0
    if all(elem < tol for elem in abs(dx)):
      result.x = x
      result.success = True
      break
    else:  # For the next step
      pass
    x0 = x
  else:
    result.x = x0
  return result
```

3.3 Symbolic derivation of system derivatives

The most common cause of errors when using a bifurcation calculation package is mistakes in calculating or filling in the Jacobian or Hessian of the system before performing the calculations. Python provides the `sympy` package as a symbolic algebra system, which automates the pre-

derivation of these terms simply by providing the software with differential equations. While symbolic differentiation[9] can also be used in torch and other packages, in the case of bifurcation calculations, the algebraic structures like Jacobians do not change during the calculation, so it is more efficient to write them out as a script. Listing 3.2 shows the derivation of derivative functions using `sympy`. Thus, preparing for the proposed system involves only describing the differential equations to be solved in `sympy` format[9]. Formula explession of the system is printed as Python style just by `print()`. Additionally, `sympy` expression objects can be output in the C language format by using the `ccode()` function.

Listing 3.2: Automatic algebraic derivation with sympy

```
import sympy as sp
def func(x, p, t):
    return sp.Matrix([f_1, f_2, ..., f_n])
sym_x = sp.MatrixSymbol("x", xdim, 1)
sym_p = sp.MatrixSymbol("p", pdim, 1)
sym_t = sp.Symbol("t")
f = func(sym_x, sym_p, sym_t)
dfdx = sp.derive_by_array([f[i] for i in range(xdim)],
                          [sym_x[i] for i in range(xdim)]).transpose()
dfdl = [sp.diff(f, sym_p[i]) for i in range(pdim)]
d2fdx2 = [sp.diff(dfdx, sym_x[i]) for i in range(xdim)]
d2fdxdl = [sp.diff(dfdx, sym_p[i]) for i in range(pdim)]
```

3.4 Computation of variational equations

In the chapter 2, we mentioned that solving bifurcation problems in discrete and continuous systems requires finding the solutions to variational equations for the Newton method. While the description of the first variational equations for initial values or parameters is relatively simple due to their lower order, the description of the second variational equations is complex because they involve tensors. Therefore, we use the matrix multiplication function `numpy.matmul()` implemented in Python's `numpy` package to succinctly describe variational equations. `numpy.matmul()` is frequently used and is provided as the `@` operator. By using the `@` operator, there is no need for any `for` loops or `if-else` branching to describe variational

equations.

In this section, we will discuss the variational equations of continuous-time systems. In continuous-time systems, numerical integration algorithms are needed not only to determine the time evolution of the system but also to solve the variational equations. In this section, we will use the Runge-Kutta method, which is implemented in `scipy`, one of the Python packages.

The integration method `scipy.integrate.solve_ivp` has the interface of:

```
solve_ivp(func, t_span, x0, method='RK45', ...) -> OdeSolution
```

`OdeSolution` is the returned class of the system solution and have the trajectory data. To solve the variational equation, `func` should return a list of right-hand term of the system and variational equations.

First, prepare a list with right-hand term of the system as

```
f = [f_1, f_2, ..., f_n]
```

This gives the integrator the system trajectories flow. Next, we need to solve variational equations simultaneously. The right-hand term of Eq. (2.43) is described as

```
dfdx @ dphidx
```

and add to the list `f` as

```
f.extend((dfdx @ dphidx).T.flatten())
```

Since `solve_ivp` interface requires a list of vector field of the system, the solution of the variational equation should be flattened. Similarly, we add other variational equations by

```
f.extend(dfdx @ dphidl + dfdl)
f.extend((dfdx @ d2phidx2
        + (d2fdx2 @ dphidx).T @ dphidx).transpose(0,2,1).flatten())
f.extend((dfdx @ d2phidxdl + ((d2fdx2 @ dphidx).T @ dphidl).T
        + (d2fdxdl @ dphidx)).T.flatten())
```

As same as the first variational equation for initial value, second variational equations require to be flatten. Now, we have a list of flow of the system and variational equations. Let `func` the function that returns the list `f`, and we have solutions by solving `func` as `solve_ivp(func, ...)`.

3.5 Domain specific technique

3.5.1 JSON interfaces

In a bifurcation calculation program, the input varies depending on the system being solved and the type of bifurcation. Therefore, it is desirable to save inputs and outputs as separate files. Additionally, the format for storing the results of bifurcation calculations should be determined in advance.

In this study, we use `json` as the interface for the bifurcation calculation program. `json` is a data format capable of handling not only single values but also lists, and its ease of accessing elements through keys is attractive. Furthermore, `json` is used in various fields, ensuring active library maintenance and making it ideal for designing interfaces with scalability in mind.

List 3.3 shows an example of `json` input. The elements shown here are the bare minimum required, and it is possible to extend them as needed. It is also possible to save the results of bifurcation calculations as multidimensional lists. In this case, since the data format is the same, the `json` file of bifurcation calculation results can be reused as input data.

Listing 3.3: JSON I/O interfaces

```
{"x0": [ <FIXED POINT COORDINATE> ],
 "params": [ <PARAMETER> ],
 "period": <PERIOD TIME>,
 "inc_param": <INDEX NUMBER OF INCREMENTAL PARAMETER>,
 "var_param": <INDEX NUMBER OF VARIABLE PARAMETER>,
 "delta_inc": <INCREMENTAL PARAMETER STEP>,
 "tol": <TOLERANCE OF NEWTON'S METHOD>}
```

3.5.2 Derivative of determinant

In the section 2.1.1, we show that the derivatives of determinant is mandatory to prepare Jacobian matrix for Newton's method. We obtain the derivative of matrix A and compute the sum of determinants obtained by replacing each element in the columns with its corresponding derivative. Listing 3.4 provides an example of a function that calculates the derivative of a

determinant. Python gives the simple description by using list slices.

Listing 3.4: An example implementation of determinant differentiation

```
def det_derivative(A, dA):
    ret = 0+0j
    for i in range(n):
        temp = A.copy()
        temp[:, i] = dA[:, i]
        ret += np.linalg.det(temp)
    return ret
```

3.5.3 Bialternate product and its derivatives

As shown in Sec. 2.1.3, the Bialternate product can be used for the conditions of Hopf bifurcations and Neimark-Sacker bifurcations. The Biproduct, used for Hopf bifurcation conditions, can be easily implemented not only in Python but also in C language by using conditional branching. However, the implementation of the Bialternate product is a little bit complex as it require multi-index. Here, we introduce a method to calculate the derivative of the Bialternate product using `itertools`, one of Python's libraries. Listing 3.5 shows an example of implementation of bialternate product. `result` is a matrix of bialternate product and `dresult` is its derivative.

3.5.4 Tangent bifurcation condition in continuous autonomous systems

In Sec. 2.2.3, we discussed bifurcation problems in autonomous systems. The period-doubling and Neimark-Sacker bifurcations in autonomous systems can be solved in the same way as the usual bifurcation conditions in discrete dynamical systems. However, for tangent bifurcations, a unique approach specific to autonomous systems is required due to the duplication of the characteristic constant at $\mu = 1$. In this section, we introduce an efficient implementation method for calculating the tangent bifurcation conditions in autonomous systems.

Listing 3.6 shows an example of computing χ of the tangent bifurcation condition. The slicing operations in Eq. (2.50) to (2.52) can be expressed simply. The provided listing

Listing 3.5: An exapmle of implementation of bialternate product

```
from itertools import product
def bialt_square(A, dA):
  n = A.shape[0]
  bialt_dim = sum(range(n))
  result = np.zeros((bialt_dim, bialt_dim))
  dresult = np.zeros((bialt_dim, bialt_dim))
  temp = np.zeros((2, 2))
  result_idx = ((i, j) for i in range(bialt_dim)
                       for j in range(bialt_dim))
  mul_idx = [(i, j) for i in range(1, n)
                    for j in range(i)]
  for row, col in product(mul_idx, mul_idx):
    for i, j in product([0, 1], [0, 1]):
      temp[i, j] = A[row[i], col[j]]
      dtemp[i, j] = dA[row[i], col[j]]
    result[result_idx] = np.linalg.det(temp)
    dresult[next(result_idx)] = det_derivative(temp, dtemp)
  return result, dresult
```

demonstrates the calculation of matrix $\partial\boldsymbol{\chi}/\partial\boldsymbol{x}_0$. The calculations for matrix $\partial\boldsymbol{\chi}/\partial\tau$ and $\partial\boldsymbol{\chi}/\partial\lambda$. can also be implemented by modifying the 7th line, e.g. `dtemp = d2phidxdtau.copy()`, `dtemp = d2phidxdl.copy()`.

Listing 3.6: An exapmle of computations of a tangent bifurcation condition.

```
chara_poly = dphidx0 - np.eye(n)
dchidmudx = np.zeros(n)
for i in range(n):
    for j in range(n):
        temp = chara_poly.copy()
        temp[:, j] = -np.eye(n)[j]
        dtemp = d2phidx2[i].copy()
        dtemp[:, j] = np.zeros(n)
        dchidmudx[i] += det_derivative(temp, dtemp)
```

3.5.5 Solving the state on Poincaré sections

As previously mentioned, in the Python language, numerical integration can be performed using the `solve_ivp` function. However, when using C/C++ for performance considerations,

there isn't a rich numerical integration library like `solve_ivp` available. In order to solve bifurcation conditions using the Newton method, it is necessary to determine whether the solution crosses the Poincaré section during numerical integration and to accurately find the coordinates on the Poincaré section. For this purpose, the Newton method can similarly be used. For determining whether the solution has crossed the Poincaré section, it is beneficial to refer to the `find_active_events()` function included in the implementation of `solve_ivp`.

The given mathematical context relates to the determination of the precise time at which a state $\boldsymbol{x}$ near the Poincaré section crosses the section after a time h. The condition for this crossing is given by the equation $q(\boldsymbol{\varphi}(h, \boldsymbol{x})) = 0$, where $\boldsymbol{\varphi}(h, \boldsymbol{x})$ is the state of the system at time h starting from $\boldsymbol{x}$, and $q(\cdot)$ is a function defining the Poincaré section.

To find the accurate time h when the trajectory intersects the Poincaré section, the Newton method can be used, which iteratively refines the estimate of h. The Newton update rule in this context is expressed as:

$$h_{k+1} = h_k - \frac{q(\boldsymbol{\varphi}(h, \boldsymbol{x}))}{\dfrac{\partial q}{\partial \boldsymbol{x}} \boldsymbol{f}(h, \boldsymbol{x})}. \tag{3.5}$$

Here, h_k is the current estimate of the time, and h_{k+1} is the updated estimate. The term $\dfrac{\partial q}{\partial \boldsymbol{x}} \boldsymbol{f}(h, \boldsymbol{x})$ in the denominator is the derivative of q with respect to the state $\boldsymbol{x}$, evaluated along the flow of the system given by $\boldsymbol{f}(h, \boldsymbol{x})$.

It's noted that the denominator becomes zero only at equilibrium points where $\boldsymbol{f}(t, \boldsymbol{x}) = \boldsymbol{0}$. However, since event detection (i.e., detecting when the trajectory crosses the Poincaré section) is typically applied to periodic orbits rather than equilibrium points, division by zero is not a concern in this case.

While the bisection method could also be used to solve for h, it converges linearly and thus is less efficient compared to the quadratic convergence rate of the Newton method. Therefore, the Newton method is preferred for performance reasons in this context.

3.6 An illustrated examples

3.6.1 Discrete system: two-coupled neuron dynamics

In recent years, neuron dynamical systems have been actively studied, and there are not only continuous-time models but also discrete-time models. In neural networks for learning, discrete neuron dynamical systems are useful in terms of computational cost. The periodic points in these systems are involved in the learning performance of the network, and bifurcation may occur for these points when the parameters are dynamically updated in the learning process[10].

The convergence point of the neural network changes topologically at the bifurcation point in the parameter space. When crossing this bifurcation point, the network generates discontinuous output changes, which does not guarantee learning efficiency or makes it unstable. However, in reality, abundant bifurcation phenomena have been observed in many neuronal dynamical systems, especially the Neimark-Sacker bifurcation, which generates quasi-periodic solutions that may lead to poor learning convergence[11].

In the case of learning, performance improvement by changing the activation function has been done, and various models have been proposed instead of the conventional sigmoid function[12]. In neuron dynamical systems, rich bifurcation phenomena have been observed in the previous research. However, when the activation function is made richer or the scale of the network increases, the complexity of the bifurcation structure is expected to increase due to nonlinearity. In this chapter, we will investigate the bifurcation structure of the system using the Swish function, which is one of the well-considered activation function.

Consider a two-coupled neuron dynamical system[13]:

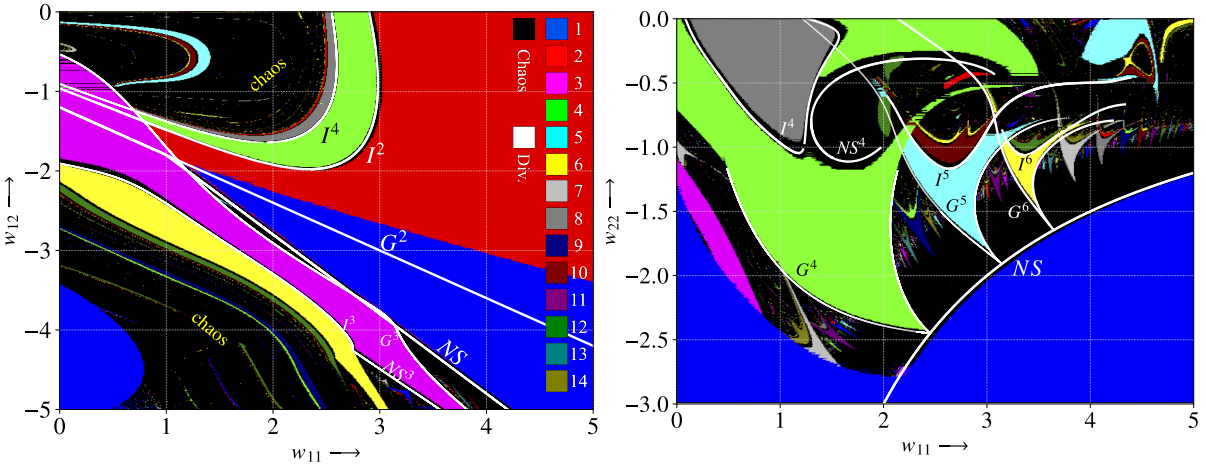
$$\begin{cases} x_{k+1} = f(w_{11}x_k + w_{12}y_k) \\ y_{k+1} = f(w_{21}x_k + w_{22}y_k) \end{cases}, \tag{3.6}$$

where w_{ij} are coupling coefficients. We use Swish function $f(z) = z/(1+e^{-\xi z})$ for the activation function[14]. Since this system has an exponential in the denominator, the derivatives required

for the Newton's method are complicated, but the bifurcation set can be calculated by using symbolic differentiation without hand calculation of derivatives.

By superimposing bifurcation curves computed by proposed algorithm and brute-force computation, we obtain Fig. 3.1a, where, $w_{21} = 5, w_{22} = -5, \xi = 1.5$. I^2 and G^2 show period-doubling and tangent bifurcations of the 2-periodic point, respectively. *NS* shows Neimark-Sacker bifurcation of the fixed point. There are chaotic regions at the top and bottom of the bifurcation diagram, and it is confirmed by the bifurcation set that the chaos is caused by period-doubling cascade. The boundary between the tangent bifurcation G^2 and 1 and 2 periodic points is misaligned because the brute-force algorithm tracks other fixed points, but the proposed method tracks the tangent bifurcation set exactly. In the 3-period region, the Neimark-Sacker bifurcation NS^3 is connected to the period-doubling bifurcation I^3, and chaos due to torus collapse occurs in this region. Figure 3.1b shows the bifurcation diagram at $w_{12} = -2, w_{21} = 5$. A 1-period region appears on the left side of the Neimark-Sacker bifurcation curve, where the Arnold tongue is predicted to exist. In fact, periodic and fixed points due to period locking coexist in the region. The 4-period Arnold tongue is tangent to the Neimark-Sacker bifurcation set. Normally, the Arnold tongue is surrounded by tangent bifurcation and period-doubling bifurcation, but in this system, there are three bifurcations set as in the 3-period region of Fig. 3.1a.

Since the entire program is written in Python, the computation speed of this method is inferior to that of conventional programs written in C/C++ or Fortran. In fact, the calculation of the I^2 bifurcation point at $w_{11} = 2$ and $w_{12} = -1.92351$ in Fig. 3.1a can be done in about 1 ms using a C++ program with the Eigen library, while the proposed Python method requires about 10 ms. By using `.subs()` method without saving algebraic structures as script, it takes about 800 ms. It is clear that the number of iterations required for convergence of the Newton method does not change. The CPU used for the calculation is AMD Ryzen5 1600. However, the shortcomings in the speed of the computation can be solved by using a GPU for matrix computation or by creating a mechanism to call the derivative preprocessor in Python from C++ or other programs.



(a) $w_{21} = 5, w_{22} = -5, \xi = 1.5$.

(b) $w_{12} = -2, w_{21} = 5, \xi = 1.5$.

Figure 3.1: Bifurcation diagram of Eq.(3.6).

3.6.2 Continuous system: extended BVP oscillator

We discuss the extended BVP oscillator[15] as an example

$$\begin{cases} \dfrac{dx}{dt} &= -z + Ax + \tanh Bx \\ \dfrac{dy}{dt} &= z - \dfrac{y}{k} \\ \dfrac{dz}{dt} &= x - y \end{cases} \tag{3.7}$$

where, A, B, k are real parameters. This system has all three types of bifurcations discussed in the previous section.

Figure 3.2 shows the bifurcation diagram of the extended BVP oscillator, where, H is Hopf bifrcation of equilibria, PD is period doubling bifurcation, G is tangent bifurcation, and NS is Neimark-Sacker bifurcation. Arnold tongue is touched to NS. In this figure we show 5-period tongue. The 5-period region is surrounded by G and PD. Bialternate product condition are used for Hopf bifurcation computation. Biproduct in Sec. 2.2.1 satisfies $\lambda_i + \lambda_j = 0$ when Hopf bifurcation occurs, since Hopf bifurcation is based on two pure imaginary eigenvalues. See also Chapter 10 of the reference[16].

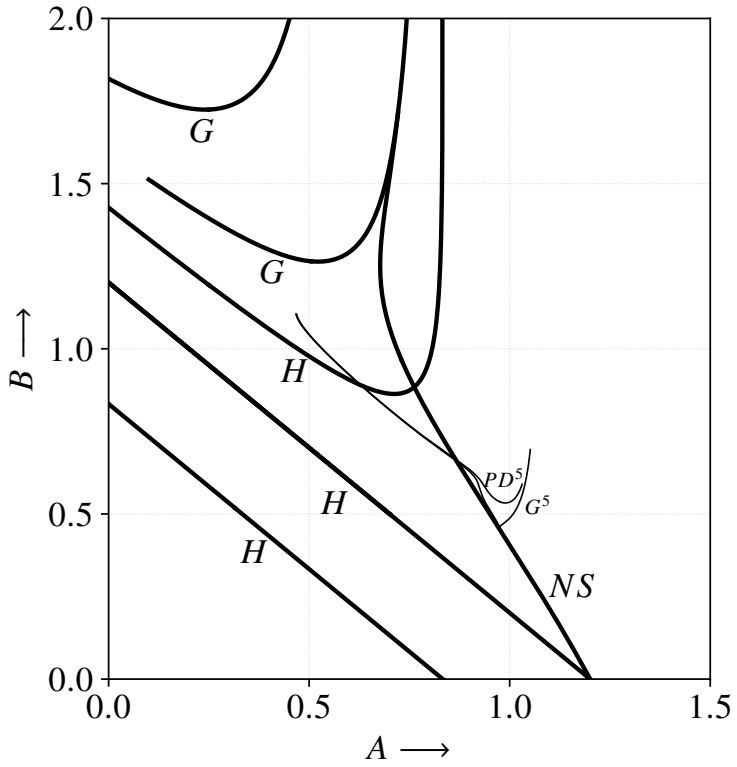


Figure 3.2: Bifurcation diagram of the extended BVP oscillator.

3.7 Conclusion

In this chapter, we discussed the design and development of bifurcation computation programs using Python and C++. We adopted `json` for the program interface, making it easy to repurpose data for other programs such as real-time numerical integration and graph plotting. Additionally, by utilizing `sympy` for symbolic differentiation, program users need only code the differential equations themselves, as their Jacobian matrices and other derivatives are automatically derived by `sympy`. This approach avoids the most significant source of failure in bifurcation computation: human error in manual calculations.

Moreover, we have established better objective functions for the relatively complex bifurcation phenomena in high-dimensional autonomous systems. In the bifurcation computation of periodic solutions of autonomous systems, preparing objective functions for tangent and Neimark-Sacker bifurcations is unique.

Finally, using the proposed method and programs available on the author's `GitHub`, we applied bifurcation computations to both discrete and continuous dynamical systems. While Python-based bifurcation computation programs are slower compared to those in C++, they can compute at speeds within practical ranges for all but the most extreme high-dimensional systems.

Chapter 4

Generalized Hénon map with hidden dynamics

In this chapter, we discuss systems within discrete-time dynamical systems that have hidden dynamics. Hidden dynamics refer to one of the system's attractors, which, compared to normal attractors, have narrow attraction regions or are located far from the attractor. The behavior of hidden attractors often differs from that of major attractors, with their small attraction regions and unique responses, rendering the system's response singular. Here, we utilize only the classical numerical methods for dynamical systems to visualize the hidden attractors and their surrounding bifurcation structures.

4.1 Introduction

Consider the n-dimensional discrete dynamical system Eq. (2.1). The iteration of a discrete dynamical system could be related to chaos theory, which has been a focal topic of intensive research since the discoveries of the Lorenz attractor and Li-Yorke chaos [17] [18] [19].

For a point $\boldsymbol{x}_0 \in \boldsymbol{R}^n$, if there is a positive integer ℓ such that

$$\boldsymbol{f}^{\ell}(\boldsymbol{x}_0, \lambda) = \boldsymbol{x}_0, \tag{4.1}$$

where $\boldsymbol{f}^{\ell} = \overbrace{\boldsymbol{f} \circ \boldsymbol{f} \circ \cdots \circ \boldsymbol{f}}^{\ell}$, then $\boldsymbol{x}_0$ is called a periodic point with period ℓ. In particular, for $\ell = 1$, $\boldsymbol{x}_0$ is a fixed point.

A compact region $V \subset \boldsymbol{R}^n$ is called a trapping region provided that $\boldsymbol{f}(V)$ is contained in the interior of V. A set Λ is called an attracting set if there is a trapping region V such that

$$\Lambda = \bigcap_{k \geq 0} \boldsymbol{f}^k(V).$$

A set Λ is called an attractor provided that it is an attracting set which is nontrivial if the $\boldsymbol{f}$ restricted to Λ has complex dynamics (for example, $\boldsymbol{f}$ has sensitive dependence on initial conditions or positive Lyapunov exponents on Λ).

For dynamical system (2.1), bifurcation analysis means the study of the qualitative change of the dynamics with the variation of some parameters. For a fixed point $\boldsymbol{x}_0$ with a fixed parameter λ_0, i.e., $\boldsymbol{f}(\boldsymbol{x}_0, \lambda_0) = \boldsymbol{x}_0$, there are several types of bifurcations, such as saddle-node bifurcation, period-doubling bifurcation (or flip bifurcation), Andronov-Hopf bifurcation, and so on. For more information, please refer to [20].

For discrete dynamical systems, two well-known systems with chaotic dynamics are the Logistic map and the Hénon map, which are polynomial functions brought forward by May [21] and Hénon [22], respectively. Polynomial maps are important models in discrete dynamical systems because of their simple expressions with complicated dynamical behaviors.

In particular, the Hénon map or a generalized Hénon map is an important model of two-dimensional polynomial diffeomorphic maps defined on $\boldsymbol{R}^2$. The parameter region for the existence of chaotic dynamics for the real quadratic Hénon map was studied by Devaney and Nitecki [23]. An interesting result is that the polynomial diffeomorphic map with a constant Jacobian from the real or complex plane to itself is either conjugate to a composition of general-

ized Hénon maps or dynamically trivial, as shown by Friedland and Milnor [24]. The real cubic Hénon map was considered by Dullin and Meiss [25]. Some comprehensive characterizations between the dynamical behavior and the parameters for the real Hénon map were obtained respectively by Benedicks, Carleson, Viana, and Young et al. [26] [27] [28].

Techniques from complex dynamics were used by Bedford and Smillie [29] [30] to show the existence of chaotic dynamics and a quadratic tangency between stable and unstable manifolds of fixed points for the real Hénon map under certain conditions. The existence of chaotic dynamics and an orbit of tangency for the Hénon-like families of diffeomorphisms on the real plane were obtained by Cao et al. [31] using real analytic methods. The parameter regions for the existence of chaotic dynamics of some generalized Hénon maps were investigated by Zhang [32].

On the other hand, a continuous dynamical system is defined by an ordinary differential equation, as $d\boldsymbol{x}/dt = \Phi(\boldsymbol{x})$ with $\boldsymbol{x} \in \boldsymbol{R}^n$ and $\Phi : \Omega \subset \boldsymbol{R}^n \to \boldsymbol{R}^n$. An equilibrium is a real solution to $\Phi(\boldsymbol{x}) = \boldsymbol{0}$. Generally, there are two types of continuous chaotic systems according to their different types of equilibria, namely systems with self-excited attractors and systems with hidden attractors. For a system with an attractor, if the basin of attraction intersects with arbitrarily small neighborhoods of an equilibrium, then the attractor is classified as self-excited, and the corresponding system is called a self-excited system. Otherwise, the system is said to have a hidden attractor [33]. For example, the chaotic Lorenz system [18] and the chaotic Chen system [34] with the classical parameters are self-excited, while the Chua circuit with a chaotic attractor could be hidden for some particular parameter values [35]. In the studies of continuous dynamical systems, there are many results on various systems with hidden attractors [36] [37] [38] [39].

Intuitively, discrete dynamical systems with hidden attractors can be similarly defined and studied. However, the study of discrete systems with hidden attractors received much less attention. Jafari et al. [40] demonstrated the existence of some hidden attractors in one-dimensional maps by extending the analysis on the Logistic map. Jiang et al. [41] studied a class of two-dimensional quadratic maps with hidden attractors. Zhang and Chen [42] studied a class of generalized Hénon maps and showed the coexistence of an attracting fixed point and

a hidden attractor, and the existence of Smale horseshoe for a subshift of finite type and also Li-Yorke chaos.

It is noted that, in [42], only part of the parameter region was analyzed, leaving many interesting problems for further studies. In this article, we carry out more detailed analysis of the generalized Hénon map. We study its bifurcations in different parameter regions: tangent, period-doubling, and Neimark-Sacker bifurcations, via careful numerical simulations, unveiling some new dynamical phenomena such as the coexistence of two attractors, namely an attracting fixed point and a hidden attractor, where the hidden attractor is either a periodic orbit or a strange attractor depending on the parameter values.

4.2 Bifurcation analysis of generalized Hénon maps

Consider the generalized Hénon map[42]:

$$\begin{cases} x_{k+1} = dy_k \\ y_{k+1} = P(y_k) + cx_k, \end{cases} \tag{4.2}$$

where $P(x) = ax^m(x^2 - b^2)$, $m \in \boldsymbol{N}$. By checking the dynamic behavior roughly in advance, a seems to be an essential parameter for bifurcations. Let us fix $b = 1.0$ and $c = 0.005$.

The bifurcation calculation is performed with a as the variable parameter and d as the incremental parameter. Also, following Theorems 4.1 and 4.4 in Ref.[42], the range of the bifurcation calculation is ensured to be $a > 0$, $0 < d \leq 1$.

The expressions of the stability of a fixed point used in this chapter are summarized in Table.4.1. The ℓ-periodic stable fixed point is denoted by ${}_0D^{\ell}$.

Table 4.1: Classification of the stability of fixed points in 2-dimensional discrete systems.

Stability	Symbol	Multiplier
Completely stable	${}_0D$	$\lvert\mu_1\rvert < 1, \lvert\mu_2\rvert < 1$
Directly unstable	${}_1D$	$0 < \mu_1 < 1 < \mu_2$
Inversely unstable	${}_1I$	$\mu_1 < -1 < \mu_2 < 0$
Completely unstable	${}_2D$	$\lvert\mu_1\rvert > 1, \lvert\mu_2\rvert > 1$

The Jacobian matrices required for the Newton method are

$$\begin{aligned}
\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}} &= \begin{pmatrix} 0 & d \\ c & amy^{m-1}(y^2 - b^2) + 2ay^{m+1} \end{pmatrix}, \quad \frac{\partial^2 \boldsymbol{f}}{\partial \boldsymbol{x} \partial x} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}, \\
\frac{\partial^2 \boldsymbol{f}}{\partial \boldsymbol{x} \partial y} &= \begin{pmatrix} 0 & 0 \\ 0 & a(m^2 - m)y^{m-2}(y^2 - b^2) + 2a(2m+1)y^m \end{pmatrix}, \\
\frac{\partial \boldsymbol{f}}{\partial a} &= \begin{pmatrix} 0 \\ y^m(y^2 - b^2) \end{pmatrix}, \quad \frac{\partial^2 \boldsymbol{f}}{\partial \boldsymbol{x} \partial d} = \begin{pmatrix} 0 & 0 \\ 0 & my^{m-1}(y^2 - b^2) + 2y^{m+1} \end{pmatrix}.
\end{aligned} \tag{4.3}$$

Figure 4.1 shows two stable fixed points of the map with $m = 2, a = 3.16, d = 0.5$. One stable fixed point at the origin (a) and one stable fixed point in the third quadrant (b) are generated. These fixed points are bistable, but bifurcation phenomena occur only for (b).

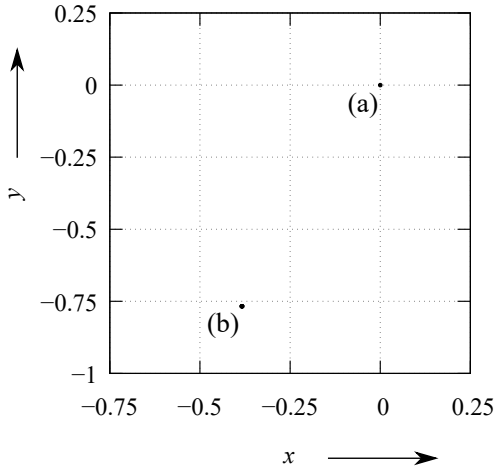


Figure 4.1: Two stable fixed points when $m = 2, a = 3.16, d = 0.5$.

Figure 4.2 shows the bifurcation diagram when $m = 2$, where white lines are bifurcation sets calculated by Newton's method. This bifurcation diagram includes the classification of periods by colors, which is obtained by the exhaustive searching method. Although the this method cannot find multiple attractors simultaneously, bifurcation sets and consequent analyses give supplementary information on topological consistency about saddle periodic points and multistability. The colors in the bifurcation diagrams are assigned according to the number of period i.e., 1:blue, 2:red, 3:magenta, 4:green, 6:yellow, 8:slate blue, 16:purple, chaos or explosion:black.

By increasing a in the bifurcation diagram, the stable fixed point (b) : ${}_0D$ is generated by the tangent bifurcation G, and the process of chaos generation by period-doubling cascade is confirmed.

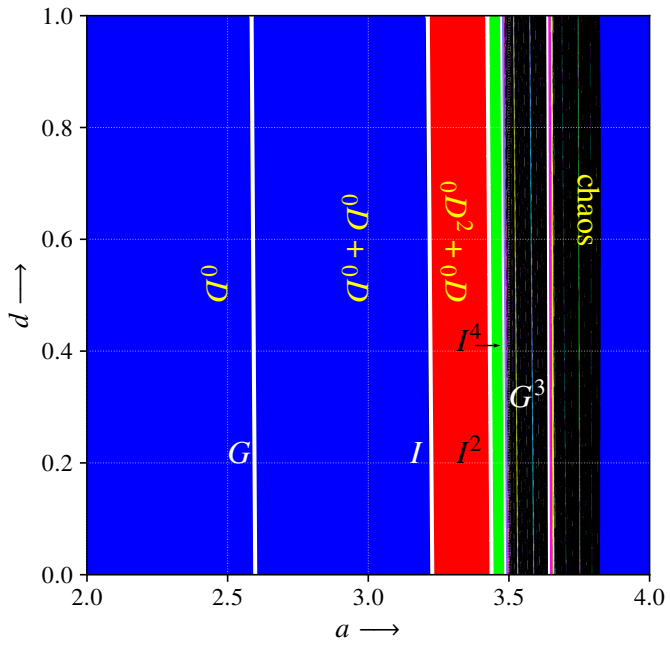


Figure 4.2: The local bifurcation diagram when $m = 2$.

Figure 4.3 shows two stable fixed points when $m = 3, a = 4.5, d = 0.5$. In this case, unlike $m = 2$, a stable origin (a) and a stable 2-period point (b) are generated.

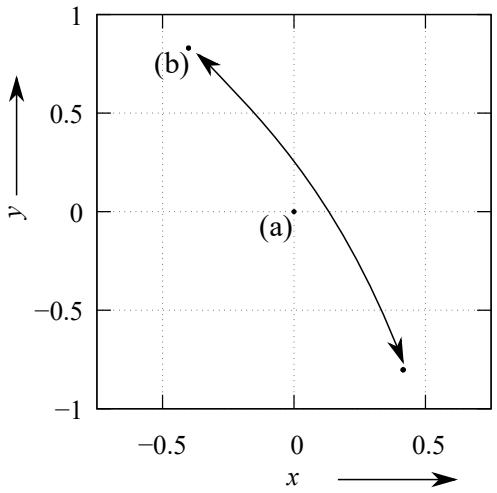


Figure 4.3: Two stable fixed points when $m = 3$.

Figure 4.4 shows the bifurcation diagram when $m = 3$. The bifurcation structure is very similar to that of $m = 2$. The 2-period fixed point (b) : ${}_0D^2$ generated by the tangent bifurcation G^2. When a increases, another 2-period fixed point is generated by the tangential bifurcation G^2 again, and then the two 2-period fixed points $2 \times {}_0D^2$ simultaneously undergo period-doubling bifurcation and change to chaotic attractors. This means that a chaotic attractor is composed of two attractors that are merged together. This is called as "double period-doubling"[42].

Figure 4.5 shows the chaotic hidden attractor when $m = 3, a = 5.0, d = 0.5$.

For the case of $m = 4$, Fig.4.7 shows exactly the same bifurcation structure as $m = 2$. It also has a stable origin and a stable fixed point in the third quadrant, with the process of chaos generation by period-doubling bifurcation.

4.3 Bifurcation analysis with $P(x_k)$

Consider the generalized Hénon map[42] with $P(x_k)$:

$$\begin{cases} x_{k+1} = dy_k \\ y_{k+1} = P(x_k) + cx_k. \end{cases} \tag{4.4}$$

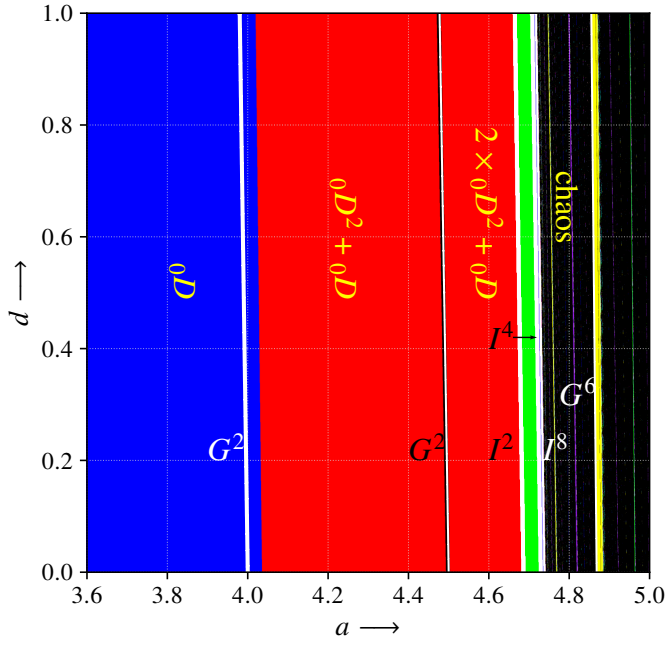


Figure 4.4: The local bifurcation diagram when $m = 3$.

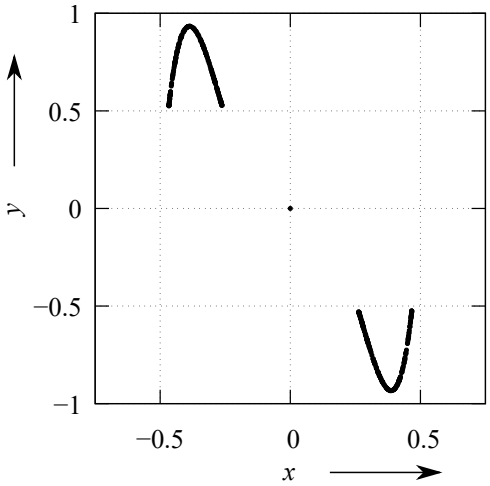


Figure 4.5: The chaotic hidden attractor when $m = 3, a = 5.0, d = 0.5$.

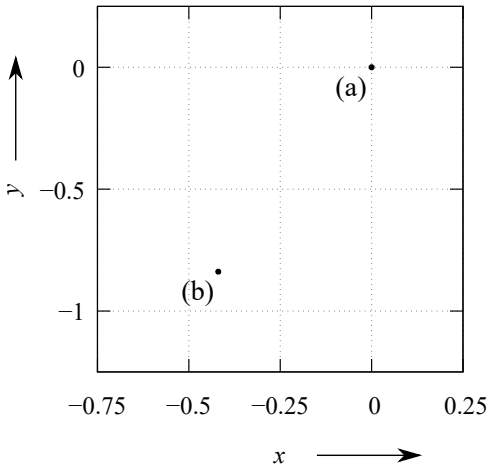


Figure 4.6: Two stable fixed points at $m = 4$.

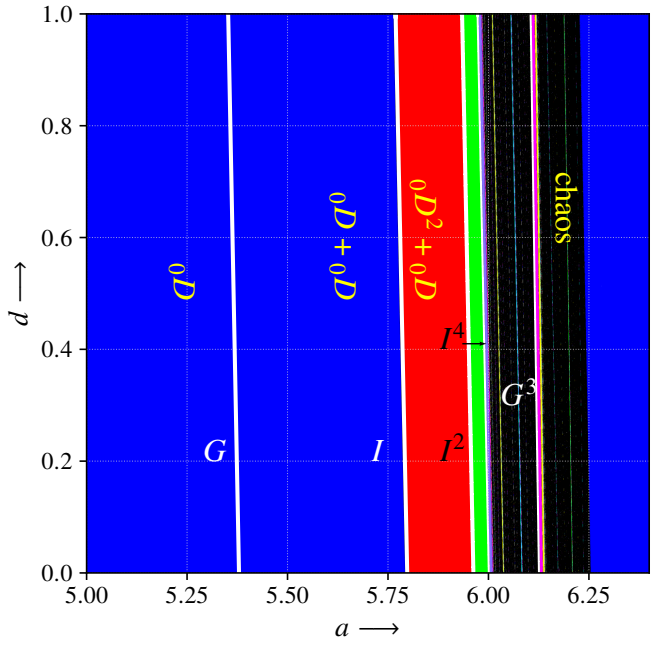


Figure 4.7: The local bifurcation diagram when $m = 4$.

Although this system is effectively a one-dimensional difference equation by substituting the first equation into the second equation, a richer bifurcation phenomenon is observed than those from Eq.(4.2). In addition, when $m = 3$, a hidden attractor can be found in the rectangular chaotic attractor region.

Figure 4.8 shows the three stable fixed points when $m = 2$. The fixed point in (a) is the stable origin, while (c) shows a 1-periodic fixed point. Fixed points in (b) is 2-periodic. These fixed points exist simultaneously, especially (b) and (c) generate chaos through period-doubling cascade.

Now, the parameter d is treated as a variable in the bifurcation computation. Then, Jacobian

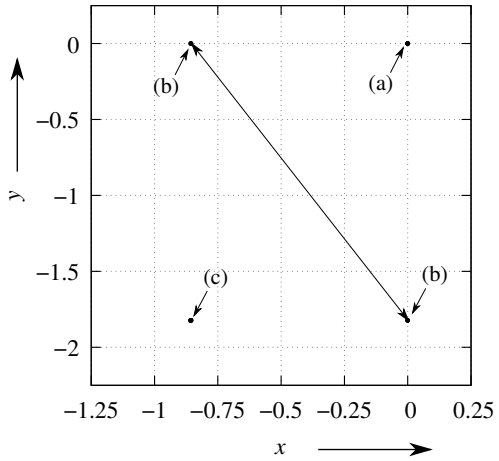


Figure 4.8: Three stable periodic fixed points when $m = 2$, $a = 4.5$, $b = 1.02$, $c = 1.04$, $d = 0.4699$.

matrices required for Newton's method are

$$\begin{aligned}
\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}} &= \begin{pmatrix} 0 & d \\ amx^{m-1}(x^2 - b^2) + 2ax^{m+1} + c & 0 \end{pmatrix}, \\
\frac{\partial^2 \boldsymbol{f}}{\partial \boldsymbol{x} \partial x} &= \begin{pmatrix} 0 & 0 \\ am(m-1)x^{m-2}(x^2 - b^2) + 4amx^m + 2ax^m & 0 \end{pmatrix}, \\
\frac{\partial^2 \boldsymbol{f}}{\partial \boldsymbol{x} \partial y} &= \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}, \frac{\partial \boldsymbol{f}}{\partial d} = \begin{pmatrix} y \\ 0 \end{pmatrix}, \frac{\partial^2 \boldsymbol{f}}{\partial \boldsymbol{x} \partial d} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}.
\end{aligned} \tag{4.5}$$

Figure 4.9 shows the bifurcation diagram of Eq.(4.4) when $m = 2$.

In the lower-left area of the figure, there is only one stable ${}_0D$ at the origin, but changing the parameter will generate fixed points (b) and (c) through tangent bifurcation G. These fixed points lead to chaos through cascade of period-doubling bifurcation I. Let the set be represented as $NS + I$. On this bifurcation set, 2-periodic fixed point ${}_0D^2$ (b) generates period-doubling bifurcation, and 1-periodic fixed point ${}_0D$ (c) generates Neimark-Sacker bifurcation, and changes to 4-periodic fixed points ${}_0D^4$ simultaneously. It should be noted that although there are multiple attractors that cause bifurcation, all these have bifurcation sets at the same positions. This also appears in the case of $m = 3$ to be discussed later.

During G and $NS+I$, (b) and (c) are merged. Like the red area at the top left of the bifurcation map, there is a part where the bifurcation set cannot be confirmed on the color boundary. This is because the exhaustive searching algorithm tracks another attractor, and there is actually no bifurcation set.

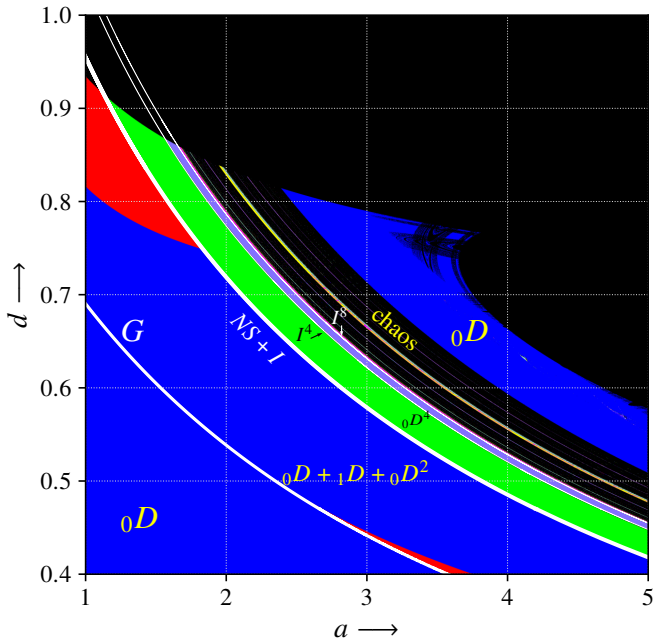


Figure 4.9: The local bifurcation diagram where $m = 2$, $b = 1.02$, $c = 1.04$.

Figures 4.10 and 4.11 show examples of a 4-periodic attractor and a chaotic attractor. From Fig. 4.8, a period-doubling bifurcation occurs in (b), and a Neimark-Sacker bifurcation occurs in (c), with 4-periodic fixed points appearing simultaneously.

Figure 4.12 shows the bifurcation diagrams when $m = 3$. The bifurcation diagrams are similar to those for $m = 2$, where the 1-periodic fixed point goes through the process of 4-periodization via the Neimark-Sacker bifurcation and chaos generation via the period-doubling cascade. In addition, it is found that there is a hidden attractor in the region of chaotic attractor for $m = 3$.

Figure 4.13 shows the basic periodic attractor for $m = 3$. Unlike the $m = 2$ case, it exhibits

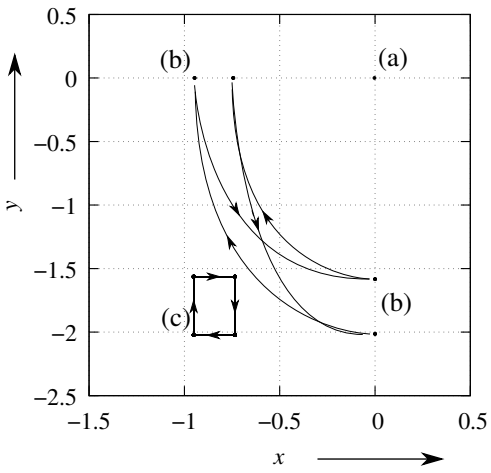


Figure 4.10: 4-periodic attractor when $m = 2$, $a = 4.6$, $d = 0.4699$.

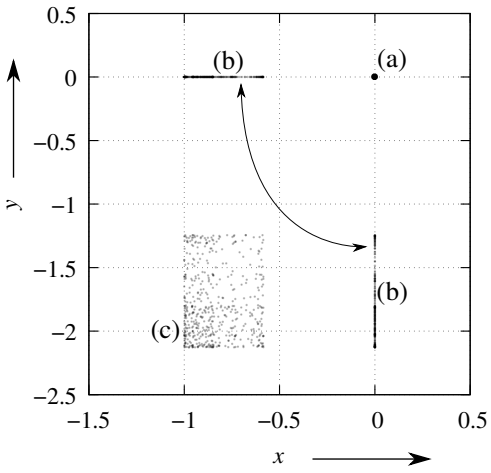


Figure 4.11: Chaotic attractor when $m = 2$, $a = 4.81$, $d = 0.4699$.

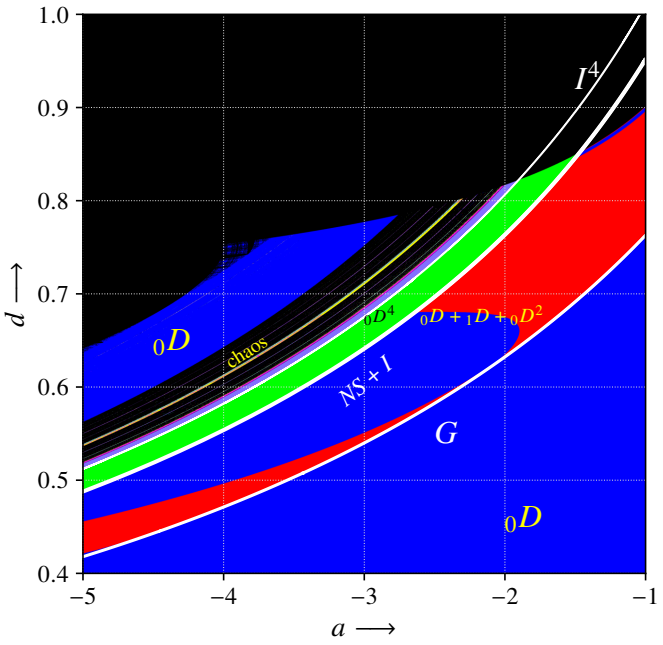


Figure 4.12: The local bifurcation diagram when $m = 3$, $b = 1.02$, $c = 1.04$.

a prominent symmetry structure and a 2-periodic attractor (d):${}_0D^2$, which is not on the x-axis and y-axis. The other attractors are the same as that for $m = 2$: (a) : ${}_0D$ is the stable origin, (b) : ${}_0D^2$ is a 2-periodic fixed point on the two axes, and (c) : ${}_0D$ is a 1-periodic fixed point at which the Neimark-Sacker bifurcation occurs. Each attractor causes bifurcation at the same time as described above, and turns into a chaotic attractor as shown in Fig. 4.14.

Note that the red attractor in the chaotic region is a hidden attractor, which can be observed by giving a large initial value, because there is no attraction region near the hidden attractor. However, the hidden attractor often appears clearly when crossing the window of chaos in the parameter plane. The window of chaos is bounded by the set of tangent and period-doubling bifurcations at the fixed point, and it is especially noticeable when crossing the tangent bifurcation.

More interestingly, although the chaotic attractor originates from only one fixed point (c) and (d), several more periodic attractors appear in the chaotic window. It seems that any one of the several stable fixed points in this chaotic region is closely related to the hidden attractor.

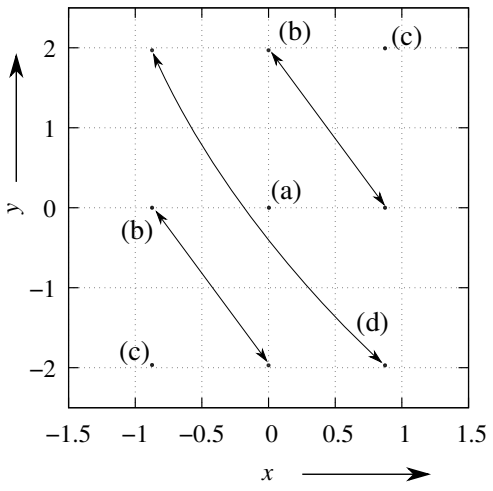


Figure 4.13: Three stable periodic fixed point when $m = 3$, $a = -5.7463$, $d = 0.443346$.

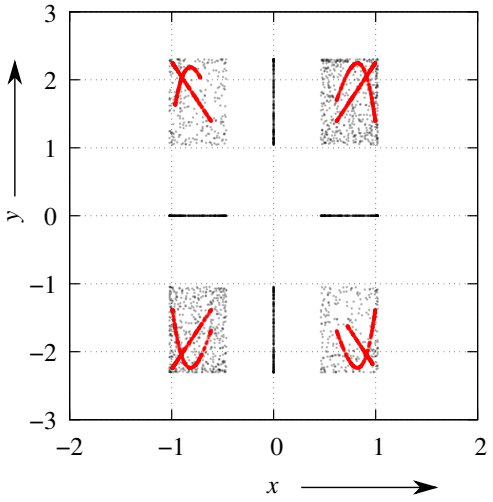


Figure 4.14: Hidden attractors in chaos region when $m = 3$, $c = 1.04$, $d = 0.443336$.

Figures 4.15 and 4.16 show the bifurcation diagram when $m = 4$, and the phase portrait. In

this case, the feature of multiple attractors at the same time with $m = 2, 3$ is not observed, but only one fixed point changes to a 4-period fixed point via the Neimark-Sacker bifurcation, and then the transition to chaos through the period-doubling cascade is confirmed.

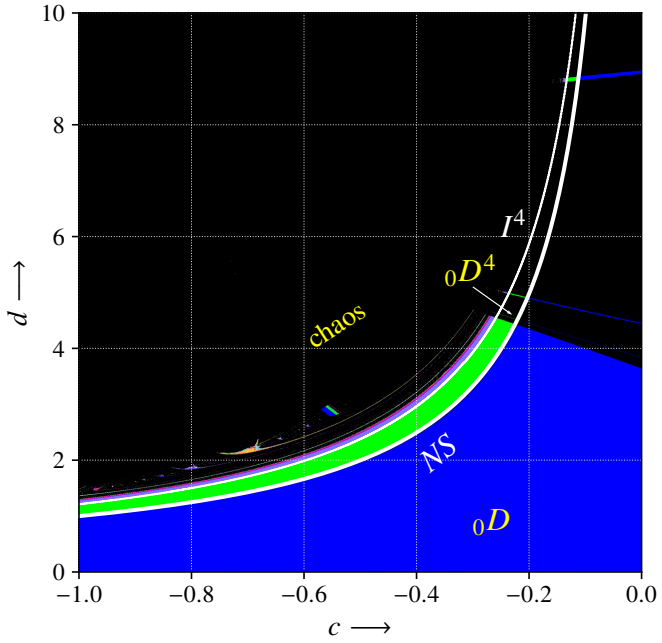
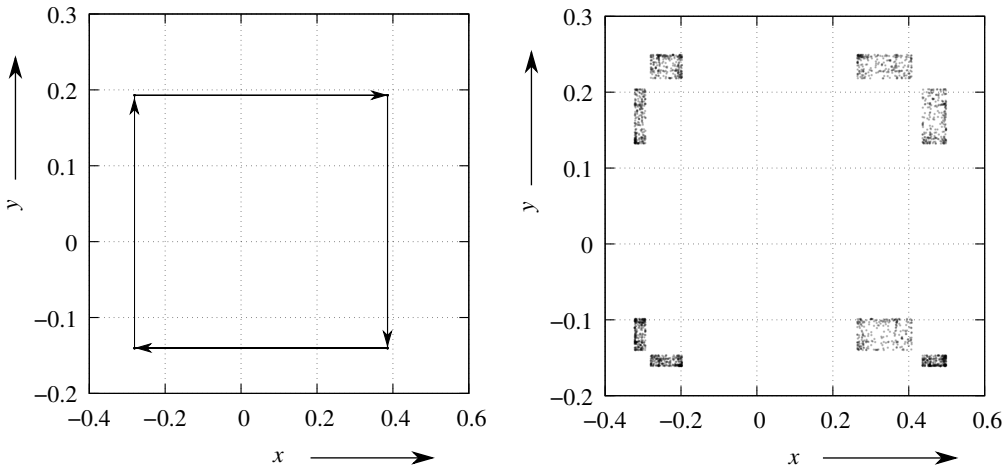


Figure 4.15: The local bifurcation diagram when $m = 4$, $a = -4.4537$, $b = 1.02$.

In particular, the argument θ of the characteristic constant on the Neimark-Sacker bifurcation set is always $\theta = \pm\pi/2$, regardless of the number of m. In other words, in Eq.(4.4), for any m, the fixed point (c) becomes 4-periodic fixed points via the Neimark-Sacker bifurcation, and then chaos is expected via the period-doubling cascade.

4.4 Conclusion

In this chapter, we calculate and analyze local bifurcations of the generalized Hénon map and its hidden attractor. In all the cases of $m = 2, 3, 4$, the mechanism of chaos generation through period-doubling cascade is confirmed. We found that the chaotic hidden attractor has the same



4-period, $c = -0.592$. Chaotic attractor, $c = -0.612$.

Figure 4.16: Phase portrait of the fixed point (c) when $m = 4, a = -4.4537, b = 1.02, d = 2.0$.

process of chaos development as an ordinary strange attractor, although the attraction region is small. We also analyze the generalized Hénon map with $P(x_k)$. As a result, in all cases of $m = 2, 3, 4$, the mechanisms of generating chaos through the cascade of 4-periodization of fixed points and period-doubling bifurcation by $NS + I$ are clarified. Double period-doubling[42] is also confirmed for every m, and we also found that multiple attractors exist separately and bifurcate simultaneously via parameter changes. In particular, in the process of the dynamic change caused by the parameter change, two separate attractors coexist in the chaotic window, one is a chaotic attractor due to self-excited oscillation, and the other is a stable hidden attractor covered by the former one.

Chapter 5

Transient responses to relaxation oscillations in multivibrators

In this chapter, we will discuss the Multivibrator, which is one of the slow-fast dynamical systems. Due to the singular perturbation nature of slow-fast systems, classical numerical methods, including numerical integration, are not applicable. These systems may exhibit a phenomenon known as a canard explosion, where there is a sudden change in the amplitude of periodic solutions. However, determining the parameters that lead to this canard explosion is challenging. In this study, we use numerical methods based on asymptotic expansion to calculate the canard explosion parameter. The slow-fast characteristics caused by the time constants of the operational amplifiers in multivibrators have been confirmed to lead to canard explosion phenomena.

This chapter is based on author's bibliography[7]. All the materials and figures of this chapter are reused from author's bibliography[7], under the permission of the CC-BY 4.0.

5.1 Introduction

A multivibrator[43] is a type of electronic circuit typically implemented using opamps. These circuits are often used as timers or switches. Multivibrators, as the name suggests, can generate multiple types of oscillations. The oscillation states of the system vary depending on the circuit configuration and are classified into three types:

1. Astable: This variety of multivibrator continuously produces oscillations and is commonly used as an oscillator circuit.

2. Monostable: This system produces oscillations once and then stops.

3. Bistable: Bistable multivibrators have two stable states, and the oscillation state that is active is based on the initial conditions or external inputs.

These multivibrators are often modeled as hybrid systems. A hybrid system possesses characteristics of both continuous-time and discrete-time dynamical systems. In multivibrators that are assumed to contain an ideal opamp, the state not only undergoes continuous changes but also experiences discrete changes due to switching. When such circuits are modeled as a hybrid system, one can neglect the transient responses of the oscillation states, making it relatively easy to derive return maps.

The Astable mode of the multivibrator is often used as a square wave oscillator. In related research on dynamical systems, electronic fireflies[44] and their synchronization phenomena[45] have been analyzed using square wave oscillators. In these studies, the square wave oscillators are examined as hybrid systems using ideal operational amplifiers, and precise return maps have been obtained. However, the opamps and operational amplifiers (opamps) used in the realization of these circuits do not possess ideal characteristics. Thus, the system must be modeled as a continuous system when we consider actual circuit systems. Even sophisticated opamps and opamps produce outputs with slight delays[43]. The transient responses of the opamps within multivibrators should not be overlooked. Monostable and bistable multivibrators are designed such that their equilibrium state is also an equilibrium point of the system (rather than an

oscillation state being an equilibrium point), making the analysis of state changes relatively straightforward. However, since astable systems continuously produce oscillations, the delay in the opamp induces different oscillating states. It is particularly apparent in astable multivibrators that transitions from non-oscillating to oscillating states occur when the circuit parameters change. At these transitions, despite the actual device having continuous characteristics, the amplitude of the circuit output is observed to change "discontinuously" from zero. One might intuitively assume that this amplitude explosion is continuous[46]. However, the transient response of the stable state to such parameter changes has not been investigated previously.

The square waves that are produced by multivibrators are often referred to as relaxation oscillations[47]. Relaxation oscillations involve a rapid state change within a certain cycle, followed by a period in which that state is maintained. Thus, "relaxation" implies that both slow and fast changes are involved in a transition between states. Such oscillations can be explained by considering a slow–fast dynamical system[46]. These are systems composed of two continuous-time dynamical systems that operate on different timescales; these systems permit both slow- and fast-state characteristics to be investigated.

A notable phenomenon observed in slow–fast dynamics is the canard[48]. Canards occur only in a very limited range of parameters immediately prior to an amplitude explosion[46], [49]. A canard is a solution that changes its amplitude drastically in response to a small parameter change; this kind of amplitude explosion is referred to as a canard explosion. We also note that the term "Canard" is derived from the French word for "duck", and it refers to the characteristic "duck-like headed shape" of the trajectory with large amplitudes. When it can be shown that the dynamics of a multivibrator contains a canard, it is possible to demonstrate the continuity of the amplitude change that occurs during the relaxation oscillation.

This chapter models a multivibrator as a slow–fast dynamical system and numerically investigates the canard explosion that occurs due to parameters changing during the relaxation oscillation. It is found that the transition to the relaxation oscillation of the multivibrator is continuous. We provide an explanation for this observation based on the bifurcation theory of dynamical systems[4], [50]. Additionally, based on the results obtained via the numerical

experiments presented here, we conduct circuit experiments. Economical opamps are used to observe canards easily, demonstrating the possibility of observing this complex phenomenon without the use of expensive equipment. In the experiments, both canards and canard explosions are observed.

5.2 Slow–Fast dynamical systems

A slow–fast dynamical system[46] can be represented by a system of ordinary differential equations of the form:

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{y}, \lambda, \epsilon), \quad \epsilon\frac{d\boldsymbol{y}}{dt} = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}, \lambda, \epsilon), \tag{5.1}$$

where $\boldsymbol{x} \in \mathbb{R}^m$, $\boldsymbol{y} \in \mathbb{R}^n$, $\boldsymbol{f} : \mathbb{R}^m \times \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^m$, $\boldsymbol{g} : \mathbb{R}^m \times \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$, $\lambda \in \mathbb{R}$, and $0 < \epsilon \ll 1$. Dividing both sides of the second equation in (5.1) by ϵ, $d\boldsymbol{y}/dt$ becomes larger than $d\boldsymbol{x}/dt$. Therefore, in this work, $\boldsymbol{x}$ is referred to as the slow variable and $\boldsymbol{y}$ is called the fast variable. By setting $\tau = t/\epsilon$, we can also obtain the equivalent form of (5.1):

$$\frac{d\boldsymbol{x}}{d\tau} = \epsilon\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{y}, \lambda, \epsilon), \quad \frac{d\boldsymbol{y}}{d\tau} = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}, \lambda, \epsilon). \tag{5.2}$$

In this work, we refer to the dynamics produced by Eqs.(5.1) and (5.2) as the slow-timescale and the fast-timescale dynamics, respectively

Let us consider the case of the singular limit $\epsilon \to 0$. In this limit, (5.1) becomes a differential-algebraic equation, which is given by,

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{y}, \lambda, \epsilon), \quad \boldsymbol{0} = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}, \lambda, \epsilon), \tag{5.3}$$

and (5.2) becomes a layer equation,

$$\frac{d\boldsymbol{x}}{d\tau} = \boldsymbol{0}, \quad \frac{d\boldsymbol{y}}{d\tau} = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}, \lambda, \epsilon). \tag{5.4}$$

We refer to (5.3) and (5.4) as the reduced problem. The set defined by the second equation in (5.3),

$$C_0 = \{(\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{R}^{m\times n} : \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}, 0) = \boldsymbol{0}\}, \tag{5.5}$$

is referred to as the critical manifold. The flow described in (5.3) can be considered to be determined by $d\boldsymbol{x}/dt$ subject to the condition that it is bounded on C_0.

Fig. 5.1 shows an schematic illustration example of a typical flow within slow–fast dynamical systems for $m = n = 1$. The figure is the case of van der Pol equation which is a most classical[48] slow–fast dynamical system. We consider f to be cubic function which is a type of y^3 and g to be a linear function. The points p_- and p_+ in $C_{0,s} = \{p \in C_0 : \partial g/\partial y(p, 0)$ is not invertible$\}$ represent points where the uniqueness of the flow is lost; these are called fold points and satisfy the following expressions:

$$\frac{\partial g}{\partial y} = 0, \quad \frac{\partial^2 g}{\partial y} \neq 0, \quad \frac{\partial g}{\partial x} \neq 0.$$

In the singular limit $\epsilon \longrightarrow 0$, C_0 can be divided into subsets according to $C_0 = C^+_{0,a} \cup C_{0,r} \cup C^-_{0,a} \cup C_{0,s}$. Of these subsets, $C_{0,a} = C^+_{0,a} \cup C^-_{0,a}$ represents the attractive part of the critical manifold and $C_{0,r}$ represents the repelling part of C_0. If an equilibrium point lies on $C_{0,a}$, the equilibrium point is completely stable. When the equilibrium point is situated on $C_{0,r}$, the equilibrium point is completely unstable and there exists a stable periodic solution. It can be seen that the periodic solution evolves along $C^{\pm}_{0,a}$ and jumps to $C^{\mp}_{0,a}$ when the orbit reaches a folding point. This behavior is called a relaxation oscillation. The dynamics of this transition can be considered to be a hybrid system based on (5.3) and (5.4), with the reaching of the folding point being considered as the event.

One of the notable phenomena in slow–fast dynamical systems is the canard solution[48]. The canard solution occurs when the equilibrium point on $C_{0,r}$ transitions to a relaxation oscillation via a Hopf bifurcation. In such a system, the canard is observed only within a very

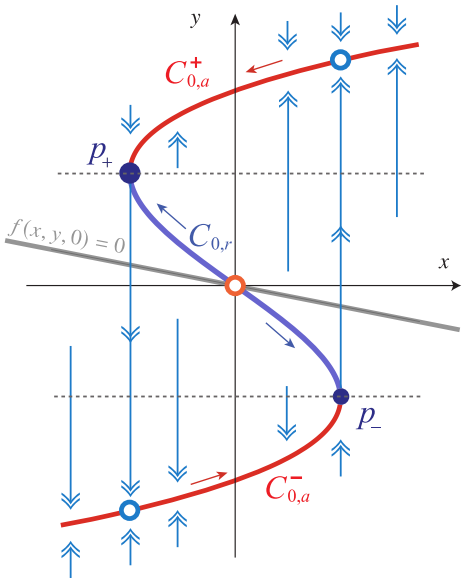


Figure 5.1: Schematic illustration which represents a critical manifold and folding points. This illustration is based on the type of van der Pol equation.

small region in the parameter space, and a amplitude of a stable periodic solution increases explosively as a result of a small change in the parameters. This behavior is referred to as a canard explosion. Fig. 5.2 shows a schematic one-parameter bifurcation diagram around a Hopf bifurcation. This figure is drawn based on the illustration used in Fig. 8.3 of Ref. [46]. In this figure, the parameter $\mathcal{A}$ denotes the amplitude of the attractor, $\mathcal{A} = \max y - \min y$, where $\max y, \min y$ are the maximal and minimal value of the limit cycle. As depicted in Fig. 5.2, canard explosions can be classified into two types: (a) those associated with a supercritical Hopf bifurcation and (b) those associated with a subcritical Hopf bifurcation. Solid lines and dotted lines represent stable and unstable limit cycles/equilibria, respectively. In the case of Fig. 5.2(b), the tangent bifurcation is observed at the parameter where stable and unstable periodic solutions adhere together. This indicates that the system is bistable when the parameter is between the tangent bifurcation and the Hopf bifurcation.

The van der Pol oscillator represents a typical slow–fast dynamical system[48]. In the original van der Pol equation[51], a vacuum tube amplifier is used, resulting in a critical manifold of

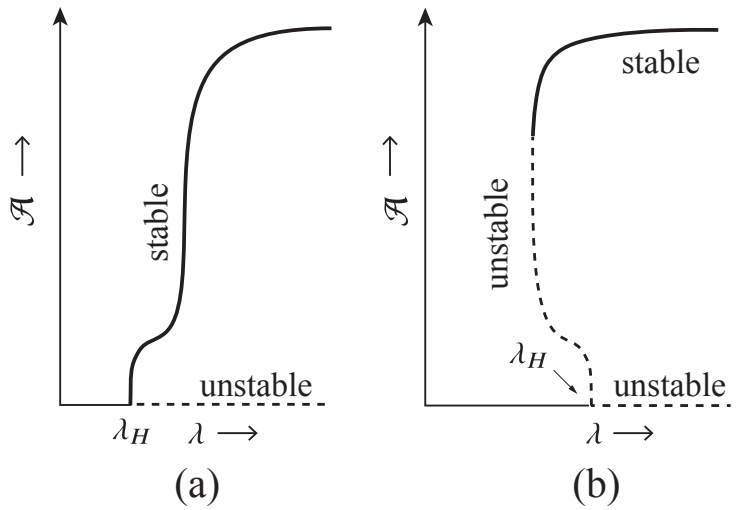


Figure 5.2: Schematics of one-parameter bifurcation diagrams. Canard explosions, i.e., the explosive amplitude changes, are shown around the Hopf bifurcation parameter λ_H. (a) Represents the case of a supercritical Hopf bifurcation and (b) shows the case of a subcritical Hopf bifurcation. This figure is drawn based on a Fig. 8.3 from Ref. [46].

sigmoid shape. However, we consider a model using a cubic function for simplicity:

$$\begin{aligned} \frac{dx}{dt} &= q - y \\ \epsilon\frac{dy}{dt} &= x - \frac{y^3}{3} + y. \end{aligned} \tag{5.6}$$

Fig. 5.3 shows an example of limit cycles and canard solutions in a van der Pol equation for $\epsilon \neq 0$ (see (5.6)). Fig. 5.3(a1–a4) show the phase portraits of the system for $\epsilon = 1$ and $q = 1.05, 0.987, 0.9863$, and 0.5, respectively. This system does not exhibit slow–fast characteristics. The amplitude of the trajectory changes continuously as the parameter changes. Fig. 5.3(b1–b4) are the phase portraits of the system for $\epsilon = 0.1$ and $q = 1.05, 0.987, 0.9863$, and 0.5, respectively. It is noted that slow–fast characteristics can be observed in this system for $\epsilon = 0.1$. In all the figures, regardless of the initial state, the trajectories converge to the orbits shown in the figures for those parameters. We note that the change in the value of q used to obtain Figs.5.3(b2) and (b3) is very small, and this small change in q leads to large changes in the behavior of the system. This rapid amplitude change represents a canard explosion. We classify

the small-amplitude cycle immediately before the observed canard explosion (Figs.5.3(b2)) as a "canard without head" and the large-amplitude cycle immediately after the explosion (Figs.5.3(b3)) as a "canard with head." It can be observed that even a small change in the parameters describing the system can lead to a rapid increase in the amplitude. In the case of van der Pol oscillator, Analytical methods for calculating the Canard explosion parameter are provided[52].

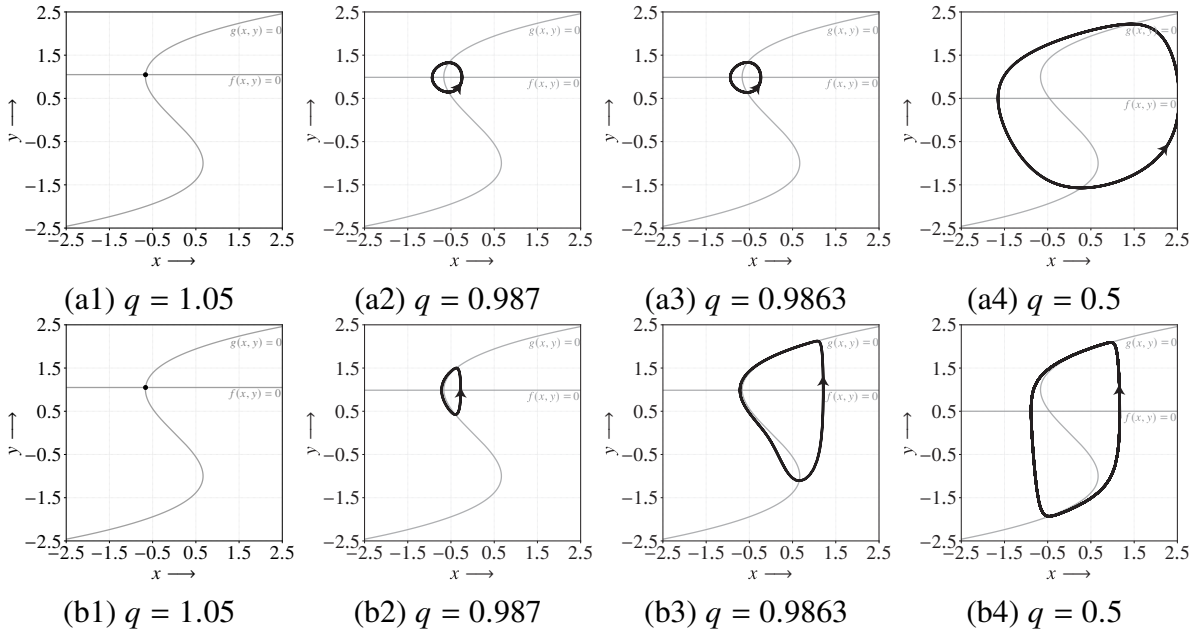
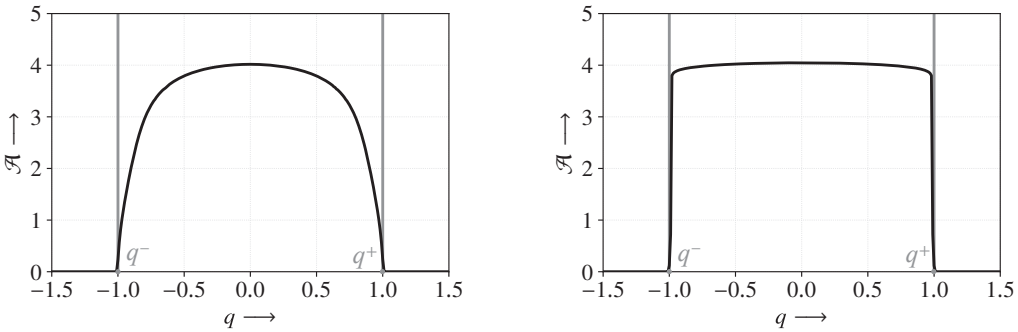


Figure 5.3: Stable equilibrium points and limit cycles in a van der Pol oscillator. (a1–a4) depict the case of $\epsilon = 1$, and (b1–b4) show the case of $\epsilon = 0.1$. The trajectories obtained for the various parameters and an example of a canard are shown. We refer to (b2) as a canard without head and (b3) as a canard with head.

Fig. 5.4 shows the amplitude changes that are induced by parameter variations in a van der Pol oscillator. In Fig. 5.4, the grey lines $q^{\pm}$ indicate the value of q at which the equilibrium points coincide with the fold points $p_{\pm}$. From the figure, it can be seen that when the equilibrium point is on $C_{0,r}$, a Hopf bifurcation occurs and periodic solutions emerge. Conversely, when the equilibrium point is on $C_{0,a}$, the amplitude is zero, indicating that no periodic solutions occur. In systems that correspond to the behavior shown in Fig. 5.4(a), the amplitude of the stable limit cycle changes smoothly. However, in the slow–fast dynamical system characterized

by Fig. 5.4(b), the amplitude increases abruptly. Typically, as the value of ϵ decreases, the amplitude explosion becomes increasingly steep; this makes the rise in amplitude appear to be discontinuous.



(1) Non slow–fast behavior, $\epsilon = 1$. (2) Slow–fast behavior, $\epsilon = 0.1$.

Figure 5.4: Amplitude changes in a van der Pol oscillator. The system without slow–fast dynamics, shown in subfigure (a), exhibits a smooth amplitude change, whereas in the case of the system that exhibits slow–fast dynamics (subfigure (b)), the amplitude increases abruptly. The grey line indicates the parameter values at which the equilibrium points coincide with the fold points $p_{\pm}$.

Another typical system in which canards can be observed is the FitzHugh–Nagumo model[53]. The FitzHugh–Nagumo model describes the electrical activity of neurons, and the rapid changes resembling spike responses can be attributed to the slow–fast dynamics of the system. In the coupled FitzHugh–Nagumo model, canards are observed, and the canard explosion is analytically determined[54]. Canards can also be observed in discrete-time spiking neuron dynamics[55] and self-replicating systems[56]. We also note the existence of an interesting canard phenomenon in aircraft trajectories reported in Ref.[57].

5.3 Multivibrator

In this section, we obtain a circuit of a multivibrator as a slow–fast dynamical system. First, we construct a multivibrator as a hybrid system using an ideal operational amplifier and explain how square wave oscillations are generated. Next, we demonstrate that by using the dynamic characteristics of the operational amplifier, the system is constituted as a slow-fast dynamical system. Thereafter, we describe the relationship between the modes of the multivibrator and its

equilibrium points, providing the prerequisite knowledge necessary for subsequent numerical calculations. We consider the circuit shown in Fig. 5.5.

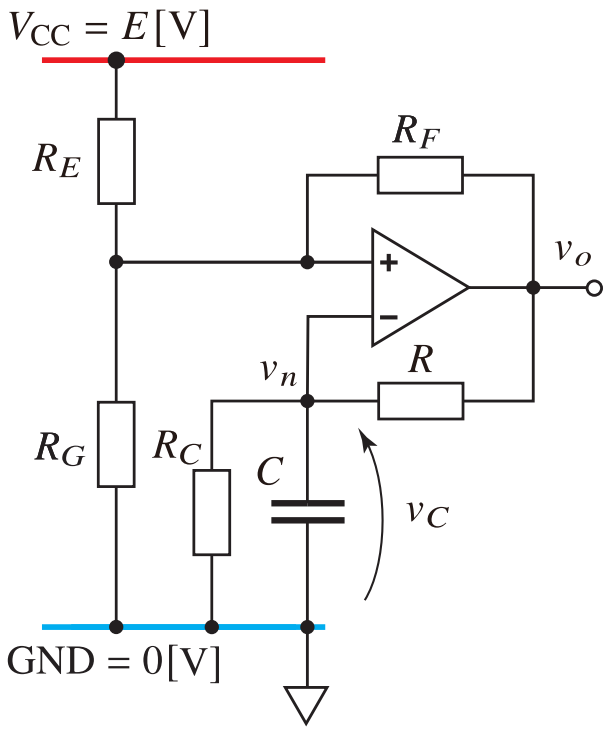


Figure 5.5: Circuit diagram of the multivibrator considered here.

Fig. 5.6 shows the output characteristics of a single-supply opamp. The output of an ideal opamp takes the form of a step function with an increase at $v_d = 0$, where v_d is the input voltage difference, given by

$$v_d = v_p - v_n = v_p - v_C. \tag{5.7}$$

By using v_d, the output of the opamp is expressed as $v_o = a(v_d)$. In the case of an ideal opamp, $a(v_d)$ becomes a step function. The actual output characteristics of an opamp take the shape of a sigmoid curve, but first, we consider the dynamics when using an ideal opamp.

According to Kirchhoff's law, the relationship between the input voltage, v_p and v_o, can be

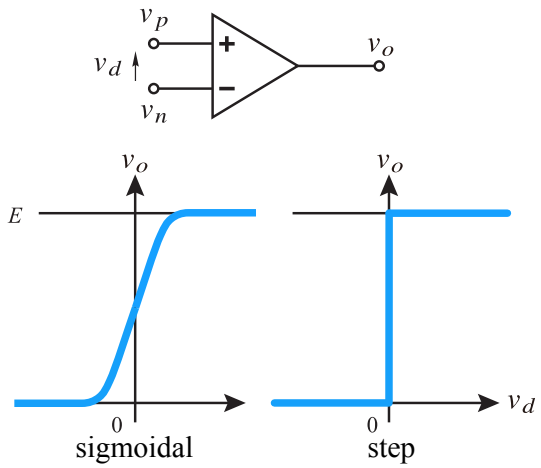


Figure 5.6: Output characteristics of an opamp.

described as follows:

$$v_p = \beta v_o + \gamma E, \tag{5.8}$$

where, we replace coefficients of (5.8) as:

$$\beta = \frac{\frac{1}{R_F}}{\frac{1}{R_E} + \frac{1}{R_F} + \frac{1}{R_G}}, \quad \gamma = \frac{\frac{1}{R_E}}{\frac{1}{R_E} + \frac{1}{R_F} + \frac{1}{R_G}}.$$

Then, we have the ideal opamp output as:

$$v_o = a(v_d) = a(\beta v_o + \gamma E - v_C). \tag{5.9}$$

For the RC circuit at the bottom of the opamp, we have:

$$RC\frac{dv_C}{dt} = v_o - \left(1 + \frac{R}{R_C}\right) v_C. \tag{5.10}$$

From (5.7) to (5.10) and the output characteristic with step function $v_o = a(v_d)$, we can obtain

the system as the following differential algebraic equation:

$$\begin{aligned} RC\frac{dv_C}{dt} &= v_o - \left(1 + \frac{R}{R_C}\right) v_C \\ v_o &= a(\beta v_o + \gamma E - v_C). \end{aligned}, \tag{5.11}$$

Fig. 5.7 shows the dynamics of the multivibrator system with opamp. For sake of simplicity, we consider the case of $R_C \to \infty$. In this limit, the slope of $f(x, y)$ is equal to 1. Here, we consider the case where $v_d = 0$, that is, the point at which $a(v_d)$ is discontinuous. In the case of $v_o = v_p - v_C = E$, we have $v_C = (\beta + \gamma)E$. Similarly, in the case of $v_o = 0$, we have $v_C = \gamma E$. When the system is considered as a hybrid system, these values represent the points where the system triggers an event. The dynamics will be explained using points a, b, c, and d in the figure. Suppose the initial value is given at point a. In this case, the state v_C changes according to the differential equation (5.11). Next, when v_C reaches b, $a(v_d)$ becomes 0 and makes a discontinuous jump to c. This corresponds to an event trigger. Subsequently, following the differential equation, v_C decreases and reaches d. When the system reaches d, $a(v_d) = E$ and it returns to point a. This results in the relaxation oscillation of the multivibrator. Fig. 5.8 shows the time-domain response in Fig. 5.7. In the figure, points a, b, c, and d corresponding to Fig. 5.7 are marked. It can be observed that the mode transitions in a way that switches for both v_C and v_o.

In practice, opamps have an output characterized by a steep increase (but not infinitely steep) around $v_d = 0$; this output has a form similar to that of a sigmoid function. The static output characteristic of the opamp can be described as $v_o = a(v_d)$. Considering the dynamic output characteristic as a first-order lag system[58], we obtain:

$$\tau_0 \frac{dv_o}{dt} + v_o = a(v_d), \tag{5.12}$$

where τ_0 is a time constant and a parameter that causes the circuit to behave as a slow–fast dynamical system. In this work, we approximate the output characteristic of the opamp using

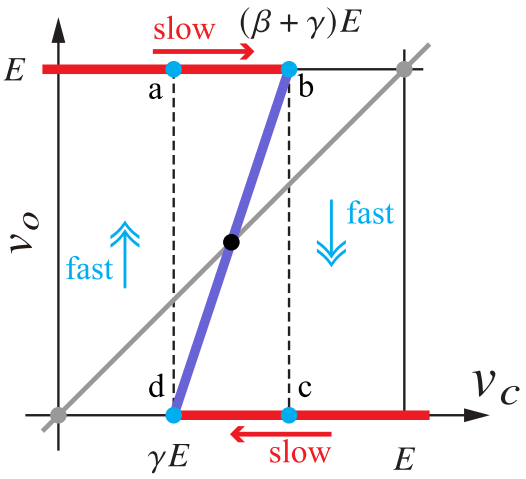


Figure 5.7: Dynamics as $\epsilon \to 0$. The critical manifold takes the form of a sharp "Z-shape".

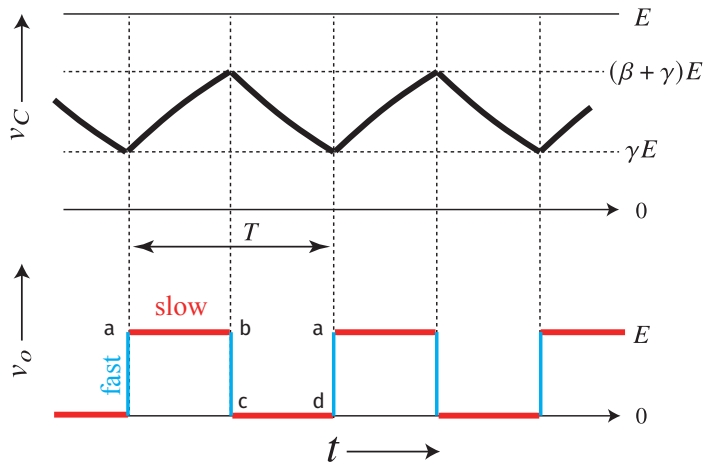


Figure 5.8: An example of the time-domain response as $\epsilon \to 0$.

the hyperbolic tangent function according to:

$$a(v_d) = \frac{E}{2}(\tanh \alpha v_d + 1), \tag{5.13}$$

where α represents the gain. Note that an ideal opamp is characterized by $\tau_0 = 0$ in (5.12) and $\alpha = \infty$ in (5.13).

From (5.8), (5.10), (5.12) and (5.13), we obtain the following system of second-order differential equations:

$$\begin{aligned} \frac{dx}{dt} &= y - \left(1 + \frac{R}{R_C}\right) x \\ \epsilon \frac{dy}{dt} &= \frac{E}{2}\big(\tanh \alpha(\beta y + \gamma E - x) + 1\big) - y \end{aligned}, \tag{5.14}$$

where v_c and v_o are replaced with x and y, respectively; these substitutions are performed to obtain a notation consistent with (5.1). Additionally, we re-scaled the time constant by $RCt =: t$ and we set $\epsilon = \tau_0/RC$ to match the form to (5.1). The parameter ϵ includes the parameters R and C, thus for example, changing R in equation (5.14) will also change ϵ. In this study, we will fix these values of R and C. Note that the changing RC can adjust the slow–fast dynamics easily. Moreover, by adjusting the value of R_C, the slope of the linear equation on the right-hand side of the first equation of (5.14) can be varied, allowing the position of the equilibrium points to be easily manipulated. Unless otherwise noted, in this work, we set $R = 100[\mathrm{k}\Omega]$, $C = 10[\mu\mathrm{F}]$, $R_E = 1[\mathrm{k}\Omega]$, $R_F = 1.5[\mathrm{k}\Omega]$, $R_G = 67[\Omega]$, and $E = 5[\mathrm{V}]$.

Fig. 5.9 shows a classification of the locations in which the equilibrium points can be generated. In the case of an equilibrium point being generated at $C_{0,a}$, as shown in ① and ② in Fig. 5.9, the equilibrium point is completely stable. On the other hand, when an equilibrium point is generated on $C_{0,r}$, as shown in ③ in Fig. 5.9, it may become an unstable equilibrium point via a Hopf bifurcation; the classification of this equilibrium point depends on the value of ϵ. This corresponds to the cause in which relaxation oscillations are observed. In other words, the characteristics of the multivibrator can be adjusted by changing the position of the

equilibrium point.

In our proposed model, by varying the value of R_C, the slope of $f(x, y)$ is changed. This results in the multivibrator depicted in Fig. 5.5 exhibiting two modes: one mode remains at a stable equilibrium point, whereas the other mode exhibits rectangular oscillations.

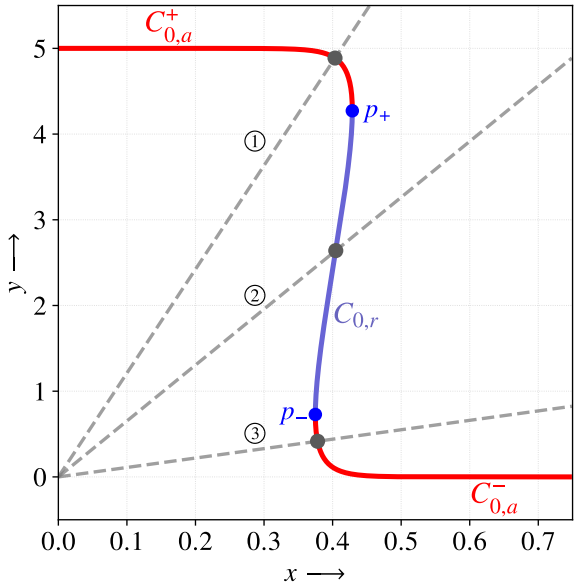


Figure 5.9: Classification of the positions of the equilibrium and singular points, $p_\pm$.

It is interesting to consider when the equilibrium point is near one of the two singular points, $p_\pm$; we show an example illustrating this situation in Fig. 5.10. As indicated by one of the dashed lines in the figure, is it possible for the trajectory to exhibit small amplitudes? The relaxation oscillation of the multivibrator starts drawing suddenly large amplitudes as shown by the solid line. During the process of the equilibrium point moving from $C^-_{0,a}$ to $C_{0,r}$, a square-wave oscillation suddenly emerges. This implies that despite the continuous variation in the parameters defining the system, the amplitude changes discontinuously. This "transient response due to parameter variation" can be explained by considering the canard explosion in slow–fast dynamical systems.

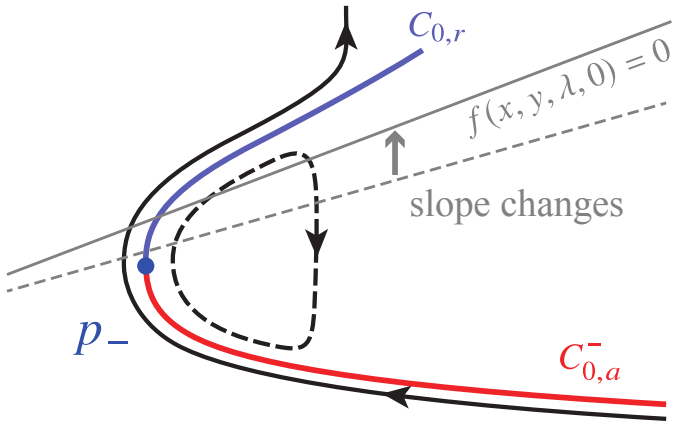


Figure 5.10: An schematic illustration of the situation when the equilibrium point is near one of the two singular points.

5.4 Numerical analysis

In slow–fast dynamical systems, due to the disparity in the timescales, ordinary numerical integration methods often suffer from a loss of accuracy or even generate fake chaotic trajectories (trajectories that do not actually exist)[59]. Analytical methods using singular perturbation theory[60] are available, but in this study, a classical numerical method of the dynamical systems e.g. a shooting method and a numerical continuation method are used, and we discuss the existence of canard explosions from the perspective of bifurcation theory. It is thus necessary to use appropriate numerical integration methods and utilize methods such as multi-precision arithmetic in order to ensure sufficient accuracy. Here, we use the Runge–Kutta–Fehlberg method to perform the necessary numerical integration.

In this section, we compute numerically the canard explosion points[61] and canard solutions. On the faster timescale, we consider the following planar system[49]:

$$\begin{aligned} \frac{dx}{d\tau} &= \epsilon f(x, y, \lambda, \epsilon) \\ \frac{dy}{d\tau} &= g(x, y, \lambda, \epsilon) \end{aligned} \tag{5.15}$$

where f and g are C^∞-class function, $\lambda \in \mathbb{R}$ is a parameter, and $0 < \epsilon \ll 1$. We assume that C_0 is locally parabolic and the minimum is coincident with the origin, $(0, 0)$. We call the origin a

fold point, and this point satisfies:

$$\begin{aligned} &g(0,0,\lambda,0)=0, \qquad &&\frac{\partial g}{\partial y}(0,0,\lambda,0)=0, \\ &\frac{\partial^2 g}{\partial y^2}(0,0,\lambda,0)\neq 0, \quad &&\frac{\partial g}{\partial x}(0,0,\lambda,0)\neq 0, \end{aligned} \tag{5.16}$$

for $\lambda \neq 0$. We can obtain the slow flow on C_0 by differentiating $x = \varphi(y)$ with respect to $t = \tau\epsilon$:

$$\frac{dy}{dt} = \frac{f(\varphi(x), y, \lambda, 0)}{d\varphi/dt(y)}, \tag{5.17}$$

where the function $x = \varphi(y)$ for $\varphi : U \to \mathbb{R}$, U is sufficiently small neighborhood of $y = 0$. The slow flow is singular at the origin for $\lambda \neq 0$ since $d\varphi/dt(0) = 0$ and $f(0,0,\lambda,0) \neq 0$. We assume a non-degenerate canard point, which is an equilibrium point located on the origin, for $\lambda = 0$. A canard point satisfies the following additional conditions:

$$f(0,0,0,0)=0, \quad \frac{\partial f}{\partial x}(0,0,0,0)\neq 0, \quad \frac{\partial f}{\partial \lambda}(0,0,0,0)\neq 0. \tag{5.18}$$

This gives us a well-defined slow flow on C_0 for $\lambda \neq 0$. Near a non-degenerate canard point, we have a normal form[62]:

$$\begin{aligned} \frac{dx}{d\tau} &= \epsilon(yh_4(x,y,\lambda,\epsilon) - \lambda h_5(x,y,\lambda,\epsilon) + xh_6(x,y,\lambda,\epsilon)), \\ \frac{dy}{d\tau} &= -xh_1(x,y,\lambda,\epsilon) + y^2 h_2(x,y,\lambda,\epsilon) + \epsilon h_3(x,y,\lambda,\epsilon), \end{aligned} \tag{5.19}$$

where

$$\begin{aligned} &h_3(x,y,\lambda,\epsilon) = O(x,y,\lambda,\epsilon), \\ &h_j(x,y,\lambda,\epsilon) = 1 + O(x,y,\lambda,\epsilon), \;\; j = 1,2,4,5,6. \end{aligned} \tag{5.20}$$

We obtain the Hopf bifurcation parameter λ_H and the canard explosion parameter λ_c as[49]:

$$\lambda_H = -K_H\epsilon + O(\epsilon^{3/2}), \tag{5.21}$$

$$\lambda_c = -\left(K_H + K_c\right)\epsilon + O(\epsilon^{3/2}), \tag{5.22}$$

where K_H and K_c is are real numbers defined by h_1–h_6. Refer to the reference[49] for a detailed definition of K_H and K_c. It can be seen that the equilibrium is stable for $\lambda < \lambda_H$ and unstable for $\lambda > \lambda_H$. The type of Hopf bifurcations can be identified by considering the sign of K_c; supercritical bifurcations exist for $K_c < 0$, and $K_c > 0$ indicates a subcritical bifurcation. We see that another expression relating λ_H and λ_c can be obtained:

$$\lambda_c = \lambda_H - K_c\epsilon + O(\epsilon^{3/2}). \tag{5.23}$$

We can obtain the Hopf bifurcation parameter, λ_H, via conventional numerical methods. Thus, given K_c, the canard explosion parameter λ_c can be obtained.

When the system is written in a form with strong nonlinearity, it is difficult to apply the method of transformation to the normal form described here. Therefore, to avoid equation transformations, Kuehn developed a method to numerically calculate λ_c using the first Lyapunov coefficient[61]. The first Lyapunov coefficient, l_1, is equal to K_c scaled by a constant, meaning we can obtain λ_c by computing l_1 numerically. However, there are several definitions of the first Lyapunov method[4], [63]. This is due to the background in traditional dynamical systems theory, where the sign of the first Lyapunov coefficient is important, and the actual value of the coefficient does not need to be considered. Therefore, the scaling factor ρ will change depending on the type of first Lyapunov coefficient used. In this work, we use Kuznetsov's convention and notate it l_1^{Ku}[4].

We assume that the equilibrium point $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ of (5.2) is under Hopf bifurcation and is translated to coincide with the origin with the coordinate change $\boldsymbol{z} = (\boldsymbol{x} - \boldsymbol{x}^*, \boldsymbol{y} - \boldsymbol{y}^*)^\top$, and we

thus obtain:

$$\frac{dz}{dt} = Mz + \boldsymbol{F}(z) \tag{5.24}$$

with $\boldsymbol{F}(z) = O(||z||^2)$ and $M \in \mathbb{R}^{(m+n)\times(m+n)}$. This form (5.24) is the linearization around the equilibrium point of (5.2), M is the Jacobian matrix around the equilibrium point. Taking a Taylor series expansion of the nonlinear term F, we have,

$$\frac{dz}{dt} = Mz + \frac{1}{2}B(z,z) + \frac{1}{6}C(z,z,z), \tag{5.25}$$

where the multilinear functions B and C are defined as,

$$B_i(\boldsymbol{u},\boldsymbol{v}) = \sum_{j,k=1}^{n} \left.\frac{\partial^2 F_i(\boldsymbol{\xi})}{\partial\xi_j\partial\xi_k}\right|_{\boldsymbol{\xi}=\boldsymbol{0}} u_j v_k, \tag{5.26}$$

$$C_i(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}) = \sum_{j,k,l=1}^{n} \left.\frac{\partial^3 F_i(\boldsymbol{\xi})}{\partial\xi_j\partial\xi_k\partial\xi_l}\right|_{\boldsymbol{\xi}=\boldsymbol{0}} u_j v_k w_l \tag{5.27}$$

where $B(\boldsymbol{u},\boldsymbol{v})$ and $C(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$ are symmetric multilinear vector functions of $\boldsymbol{u},\boldsymbol{v},\boldsymbol{w} \in \mathbb{R}^{(m+n)}$, F_i denotes the i-th element of the function $\boldsymbol{F}$. In the case of a planar system, we thus obtain a simple form of l_1^{Ku}:

$$l_1^{Ku} = \frac{1}{2\omega_0^2}\Re(ig_{20}g_{11} + \omega_0 g_{21}), \tag{5.28}$$

where ω_0 is given by the eigenvalues of the matrix M, $\lambda_{1,2} = \pm i\omega_0$, $g_{20} = \bar{\boldsymbol{p}}^\top B(\boldsymbol{q},\boldsymbol{q})$, $g_{11} = \bar{\boldsymbol{p}}^\top B(\boldsymbol{q},\bar{\boldsymbol{q}})$, and $g_{21} = \bar{\boldsymbol{p}}^\top C(\boldsymbol{q},\boldsymbol{q},\bar{\boldsymbol{q}})$. $\Re$ takes the real part of complex number. $\boldsymbol{p},\boldsymbol{q} \in \mathbb{C}^{(m+n)}$ are eigenvectors of λ_1 and the transpose $M^\top$, respectively. These are chosen such that they satisfy $\bar{\boldsymbol{p}}^\top\boldsymbol{q} = 1$. Note that M has the pure imaginary eigenvalues $0 + i\omega_0$ since we consider the equilibrium point which undergoes Hopf bifurcation.

The first Lyapunov coefficient, l_1^{Ku}, has the following property[46], [61]:

$$l_1^{Ku} = \bar{\rho} K_c + O(\sqrt{\epsilon}), \tag{5.29}$$

where $\bar{\rho}$ is the positive scaling factor. We then obtain an expression for λ_c,

$$\lambda_c = \lambda_H - \rho l_1^{Ku} \epsilon + O(\epsilon^{3/2}), \tag{5.30}$$

where $\rho = 1/\bar{\rho}$. In (5.30), we can obtain the scaling factor ρ by calculating λ_H and λ_c. λ_H can be obtain by numerical bifurcation analysis which we show later at (5.31). Also, we can obtain approximated values of λ_c with numerical continuation method. We show the specific scaling factor ρ in the numerical analysis later. In this study, we treat R_C as the role of λ.

The Hopf bifurcation parameter, λ_H, is obtained by solving the following conditions to obtain the parameters of an equilibrium point and λ_H, (x^*, y^*, λ_H), numerically:

$$\begin{aligned} f(x^*, y^*, \lambda_H, \epsilon) &= 0 \\ g(x^*, y^*, \lambda_H, \epsilon) &= 0 \\ \det(2J \odot I^{m+n}) &= 0, \end{aligned} \tag{5.31}$$

where J denotes the Jacobian matrix of $(f, g)^\top$ with respect to $(x, y)^\top$, $\odot$ denotes the bialternate product[64], and I^{m+n} is the $(m+n) \times (m+n)$ identity matrix. Here, we use Newton's method to solve the Hopf bifurcation condition.

We show the actual procedure to obtain the canard explosion parameter λ_c below.

1. Obtain the Hopf bifurcation parameter λ_H. We use the shooting method with (5.31).
2. Calculate the first Lyapunov coefficient l_1 at the Hopf bifurcation.
3. Compute the maximal canard parameter λ_c with the (5.30).

Note that the scaling factor ρ is obtained in advance by numerically as we described before. To obtain the scaling factor ρ, we use $\epsilon = 0.001$. In this study, we use the approximated

parameter $\lambda_c = 104130.3954$ at $\epsilon = 0.001$ which is obtained from continuation of periodic orbits using Runge-Kutta-Fehlberg method, the Hopf bifurcation parameter $\lambda_H = 105543.3018$ with $\omega_0 = 0.157901$ which is obtained from the shooting method with (5.31). The first Lyapunov coefficient at the Hopf bifurcation is $l_1^{Ku} = 0.87939173$. Then we have $\rho = 203231491.0351$ and able to calculate the canard explosion set with (5.30). This scale ρ changes with the parameter selected as λ. In this case, we have chosen the parameter R_C, which takes on large values, resulting in a large scale as well.

In slow–fast dynamical systems, it is difficult to compute λ_c; this calculation typically involves numerical integration, and it is difficult due to the precision requirements of the integrator because of the slow–fast characteristics[59]. Kuehn's first Lyapunov coefficient method[61] offers the advantage of involving only algebraic operations and permits the computation of λ_c without utilizing numerical integration.

Fig. 5.11 shows the $(R_C, 1/\epsilon)$ bifurcation diagram. The set represented by the curve labelled H corresponds to the Hopf bifurcation, while $C_\pm$ represent the sets of canard explosion points near the singular points, $p_\pm$. The gray lines in the figure represent the parameter values at which the equilibrium and singular points coincide. The locations of $p_\pm$ are obtained by solving the following condition for (x^*, y^*, λ_0) by Newton's method:

$$\begin{aligned} f(x^*, y^*, \lambda_0, 0) &= 0 \\ g(x^*, y^*, \lambda_0, 0) &= 0 \\ \frac{\partial g}{\partial y}(x^*, y^*, \lambda_0, 0) &= 0, \end{aligned} \tag{5.32}$$

where λ_0 is the parameter which the equilibrium point coincide to fold point. In this case, it is $\lambda_0 = R_C^+, R_C^-$. We obtain the equilibrium point (x^*, y^*) which coincides to a fold point and the parameter λ_0 at the same time by solving the objective function. We show the actual values: $R_C^- = 106.0540876[\mathrm{k}\Omega]$ and its location is $(x^*, y^*) = (x_0, y_0) = (0.7282, 0.3748)$, $R_C^+ = 11.1570095[\mathrm{k}\Omega]$ and its location is $(x_0, y_0) = (4.2717, 0.4287)$, where (x_0, y_0) shows a fold point. In the parameter region within the Hopf bifurcation through $C_\pm$, unstable equilibria

and stable periodic solutions emerge. The stable periodic solutions are characterized by small amplitudes immediately after the Hopf bifurcation, but the canard explosion at $C_\pm$ induces a rapid increase in their amplitude. Further changes in the parameter values lead to relaxation oscillations.

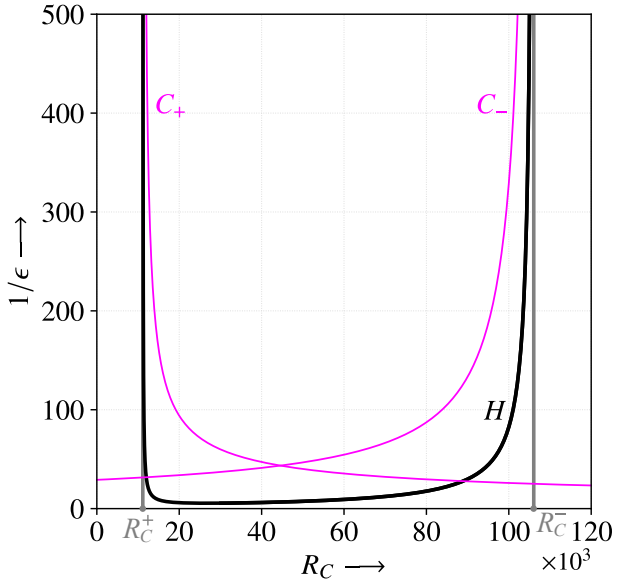


Figure 5.11: $(R_C, 1/\epsilon)$ bifurcation diagram.

Fig. 5.12 shows the one-parameter bifurcation diagram of the amplitudes of periodic solutions; this figure corresponds to $1/\epsilon = 300$ in Fig. 5.11. The amplitude $\mathcal{A}$ is the difference of the maximal and minimal values of limit cycles as same as Fig. 5.2. The gray lines in Fig. 5.12, labelled $R_C^\pm$, represent the parameter that a equilibrium point coincides to a fold point, and the magenta lines $C_\pm$ represent the canard explosion parameter, as is the case in Fig. 5.11. The amplitude can be seen to increase rapidly when the Hopf bifurcation occurs. It appears in Fig. 5.12 that the precision of C_+ is poor, but, as mentioned above, the precision improves as ϵ decreases. Indeed, small values of ϵ lead to sharp increases in the amplitude. The relaxation oscillation observed in the multivibrator can then be attributed to canard explosions that are present for sufficiently small values of ϵ.

The classification of canards as “canards with head” and “canards without head” can be determined by whether the limit cycle contains points with zero curvature or not[65].

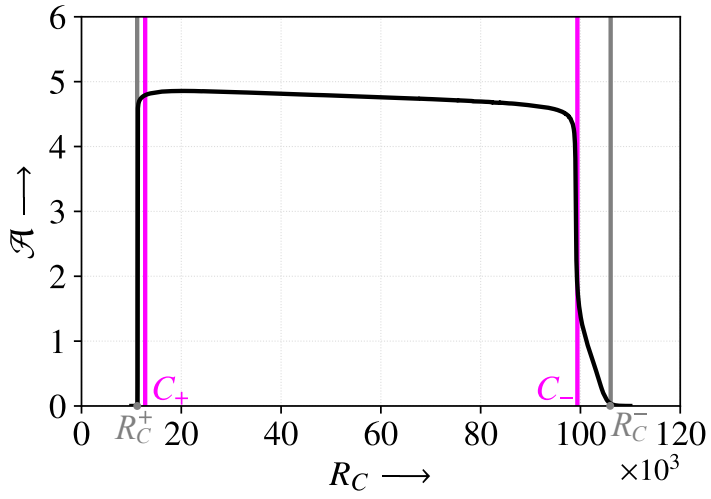


Figure 5.12: $(R_C, \mathcal{A})$ bifurcation diagram. $1/\epsilon = 300$.

Consider the planar system given in (5.1). Trajectories can be obtained by eliminating time, t, from the equation,

$$g(x, y, \epsilon)\frac{dx}{dy} = \epsilon f(x, y, \epsilon). \tag{5.33}$$

Differentiating this expression with respect to y gives,

$$\frac{dx}{dy}\frac{d}{dy}g(x, y, \epsilon) + \frac{d^2x}{dy^2}g(x, y, \epsilon) = \epsilon\frac{d}{dy}f(x, y, \epsilon). \tag{5.34}$$

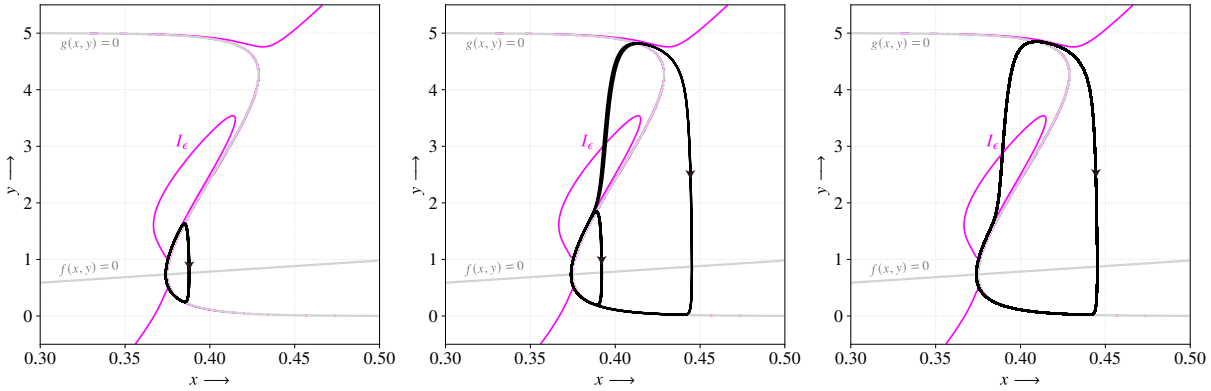
The trajectories with zero curvature thus satisfy,

$$f(x, y, \epsilon)\frac{d}{dy}g(x, y, \epsilon) - g(x, y, \epsilon)\frac{d}{dy}f(x, y, \epsilon) = 0. \tag{5.35}$$

By plotting the set that satisfies (5.35) on the x-y plane, it is possible to determine whether the limit cycle has points with zero curvature.

Fig. 5.13 shows periodic orbits near the canard explosion. Fig. 5.13(a) represents a "canard without head", and Fig. 5.13(c) shows a "canard with head". The parameters for Figs. 5.13(a) and (c) are $R_C = 104.131[\text{k}\Omega]$ and $R_C = 104.130[\text{k}\Omega]$, respectively, with $\epsilon = 0.001$. It can be

seen that the amplitude increases significantly even though the slope of $f(x, y) = 0$ changes only very slightly. The curves labelled I_ϵ in Fig. 5.13 represent the set of solutions where the curvature of the trajectory is 0. It can be seen that the headless canard does not intersect with I_ϵ, whereas the headed canard does. Fig. 5.13(b) represents a case close to the canard explosion point. Here, we observe a fake chaotic trajectory, which occurs due to the slow–fast dynamics. This is because the accuracy of the numerical integration decreases. The multivibrator model proposed in this study is a two-dimensional autonomous system, and such trajectories are not permissible. These fake chaotic trajectories will be confirmed in the subsequent circuit experiments.



(a) $R_C = 104.131[\text{k}\Omega]$ (b) $R_C = 104.1304682881[\text{k}\Omega]$ (c) $R_C = 104.130[\text{k}\Omega]$

Figure 5.13: Examples of canards in a multivibrator: (a) a canard without head, (b) fake chaos near the canard explosion point, (c) a canard with head. In (b), due to the strong slow–fast dynamics, the accuracy of the numerical integration decreases, leading to the observation of fake chaos. The data shown in this figure is obtained for $\epsilon = 0.001$.

5.5 Circuit implementation

In the above numerical simulations, we observed that the multivibrator considered here undergoes a transition from a stable state to a relaxation oscillation, and canards are observed during this process. The region in the parameter space in which canards can be observed is small, but canards have been observed[66] in multiple nonlinear electronic circuits[67]. To validate the findings of the numerical work presented here, we implemented the multivibrator in a circuit

and demonstrated the occurrence of canard explosions experimentally. Here, we aim to demonstrate the occurrence of canards in the real circuit response of the multivibrator and capture the topological changes due to the canard explosion, a characteristic of slow-fast dynamical systems. Since our goal is not to precisely replicate the canards as shown in numerical calculations, we do not verify errors in components such as resistance elements.

Due to the large value of α in (5.13), the actual components of the opamp induce a sharp Z-shape in the curve of C_0, and the amplitude explosion after the Hopf bifurcation is very pronounced. Furthermore, we note that the time-delay characteristic corresponding to ϵ is determined by the slew rate of the opamp; this value is sufficiently small. In the numerical calculations presented in the previous section, R_G was set to be smaller than R_E and R_F to reduce the severity of the Z-shape and to make the change in the slope of C_0 near $p_\pm$ more gradual. This makes it possible to observe canards over a relatively wide range of parameter values.

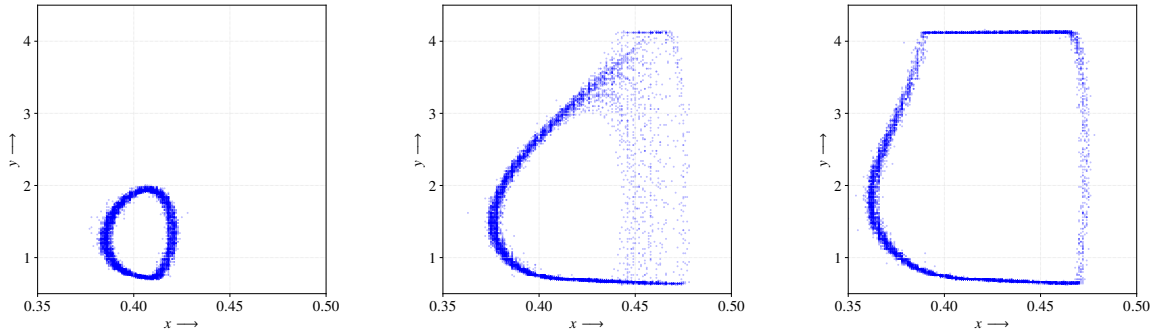
Here, the circuit is implemented according to Fig. 5.5, using an Analog Devices OP177 opamp, which has a relatively low slew-rate; this means the system is more susceptible to canards than it would be using high slew-rate opamps. This low-cost opamp has a gradual output characteristic, which makes it suitable for confirming canards. We note that it would be very challenging to observe canard explosions in systems containing high-performance opamps. However, even in the case of idealized opamps, numerical simulations have predicted the existence of canards within a very limited range of parameters.

Fig. 5.14 shows the circuit response of an experimentally realized multivibrator. It is interesting to observe the variations in the response of the circuit for different values of ϵ. However, since we cannot directly control the time constants of the opamp, we adjust C to modify the slow–fast characteristics of the system. The rows of the figure show the variation in the observed trajectories for a fixed value of C as a result of changes in R_C. As R_C changes, the system transitions from being characterized by a canard without head via a canard explosion to being characterized by a canard with head (relaxation oscillations). The columns in Fig. 5.14 show the variations in the trajectories that occur as a result of changes in the value of C for a

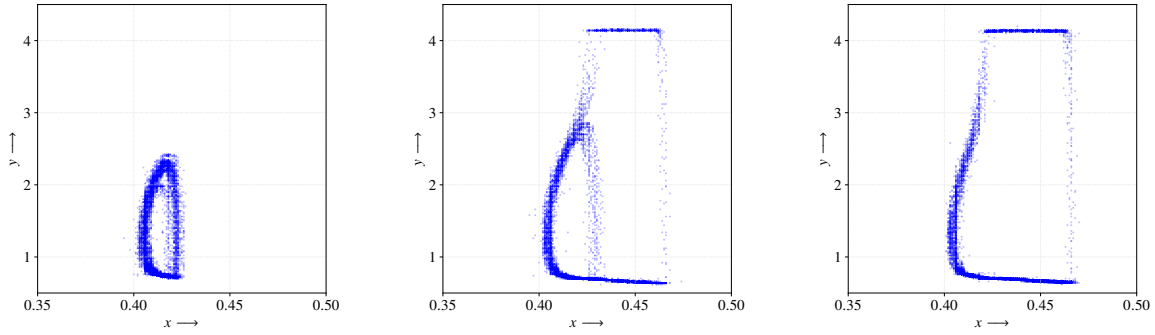
fixed value of R_C. As C increases, the movement along the slow-variable direction becomes slower, resulting in more pronounced slow–fast characteristics, while the fast movement along the fast-variable direction becomes more emphasized. Both the headless and headed canards shown in the previous section are visible in (a2) and (b2) due to the slight parameter variations induced by small external noise. The multivibrator is a planar system, thus, such trajectories are not possible in the absence of external noise.

5.6 Conclusion

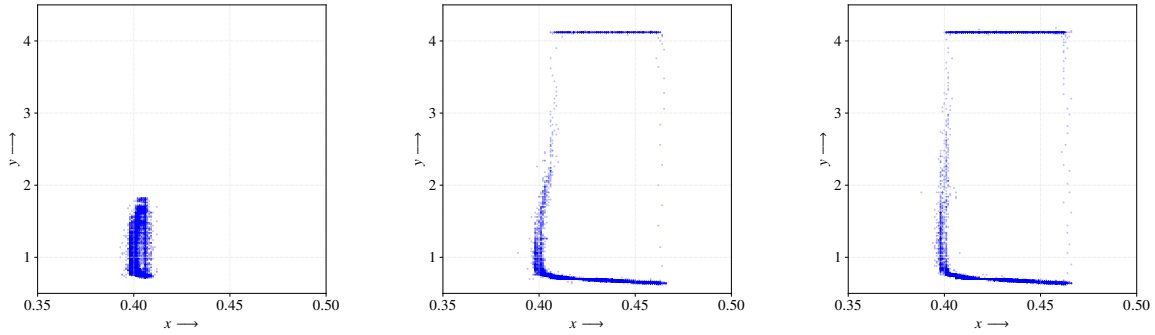
In this work, we constructed a multivibrator as a slow–fast dynamical system, which exhibited responses typical of slow–fast systems. The proposed multivibrator allows for easy adjustment of the slow–fast characteristics by modifying the circuit components. Furthermore, the position of the equilibrium point of the system can be easily changed. The Hopf bifurcation set was obtained via conventional numerical computations. The set of canard explosion points was obtained using a method based on the first Lyapunov coefficient which only requires non-complicated numerical computation. Both the numerical simulations and circuit experiments undertaken here demonstrate the existence of canard explosions in the system. As future work, the series of numerical methods presented in this study will be applied to other slow–fast dynamical systems. This includes applications to higher-dimensional systems where the fast dynamics involve two or more. In these higher-dimensional systems, analytical solutions are challenging, underscoring the importance of numerical computations.



(a1) $C = 1[\mu F]$, $R_C = 47.75[\text{k}\Omega]$ (a2) $C = 1[\mu F]$, $R_C = 47.8[\text{k}\Omega]$ (a3) $C = 1[\mu F]$, $R_C = 47.85[\text{k}\Omega]$



(b1) $C = 10[\mu F]$, $R_C = 47.75[\text{k}\Omega]$ (b2) $C = 10[\mu F]$, $R_C = 47.8[\text{k}\Omega]$ (b3) $C = 10[\mu F]$, $R_C = 47.85[\text{k}\Omega]$



(c1) $C = 30[\mu F]$, $R_C = 47.75[\text{k}\Omega]$ (c2) $C = 30[\mu F]$, $R_C = 47.8[\text{k}\Omega]$ (c3) $C = 30[\mu F]$, $R_C = 47.85[\text{k}\Omega]$

Figure 5.14: Canards observed in an experimental circuit. We capture the data using an Agilent DSO1024A oscilloscope. The sampling rate used to obtain the data presented here was 12.5 kHz. 8192 data points are plotted in each subfigure.

Chapter 6

Conclusions

In this thesis, we focused on numerical methods for singular dynamical systems, specifically addressing Hidden Dynamics and Slow–Fast Dynamics.

In Chapters 2 and 3, we discussed bifurcation computation methods for dynamical systems and explained specific implementation techniques using Python and C/C++. Most of the currently available bifurcation computation programs are implemented in C or MATLAB and are quite outdated. Using modern languages like Python and C++, we proposed methods that are computationally efficient and reduce human error. A future challenge for bifurcation computation programs is the development of programs that can handle hybrid systems, which require dealing with complex composite functions. For such cases, Python remains a good choice, except for extremely high-dimensional systems.

Chapter 4 discussed the generalized Hénon map, which exhibits hidden dynamics. Due to the narrowness of their attractor regions, hidden attractors are challenging to compute and visualize. However, using the methods from Chapters 1 and 2, we clarified the bifurcation structure of the system and visualized bifurcations and hidden attractors at relevant parameters. Future challenges include calculating and visualizing hidden attractors and related bifurcation phenomena in continuous-time systems, which are even more challenging to compute.

In Chapter 5, we discussed the multivibrator, an example of a slow–fast dynamical system.

The singular perturbation characteristics of slow–fast systems makes numerical integration difficult, rendering traditional bifurcation computation methods (presented in Chapters 2 and 3) ineffective. The periodic solutions in multivibrators, although originating from Hopf bifurcations and leading to simple periodic solutions, require asymptotic expansion methods for numerical calculations of canard explosions. This allowed us to identify not only the Hopf bifurcation sets but also the parameters for canard explosions, which were traditionally challenging to compute. Such slow–fast characteristics are inherent in operational amplifiers, suggesting that oscillator circuits involving op-amps inherently have the potential for canard explosions. Future challenges include computing bifurcation phenomena of periodic solutions in higher-dimensional systems or other slow–fast dynamical systems coupled with multivibrators. Although the multivibrator in this study is two-dimensional and does not exhibit bifurcations of periodic solutions, higher-dimensional systems can exhibit complex oscillations known as mixed mode oscillations (MMOs). Calculating these requires numerical integration, which, as mentioned, is challenging. Potential solutions include using implicit methods for numerical integration to solve variational equations.

Bibliography

[1] A. Solo, "Multidimensional matrix mathematics: Multidimensional matrix equality, addition, subtraction, and multiplication, part 2 of 6," *Lecture Notes in Engineering and Computer Science*, vol. 2185, 2010.

[2] T. Kousaka, T. Ueta, and H. Kawakami, "Bifurcation of switched nonlinear dynamical systems," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 46, pp. 878–885, 1999.

[3] T. Ueta, T. Yoshinaga, H. Kawakami, and G. Chen, "A method to calculate Neimark–Sacker bifurcation in autonomous systems," *IEICE Transaction on Fundamentals of Electronics*, vol. 10, pp. 1141–1147, 2000.

[4] Y. A. Kuznetsov, "One-parameter bifurcations of equilibria in continuous-time dynamical systems," *Elements of applied bifurcation theory*, pp. 77–115, 2004.

[5] K. Tsumoto, T. Ueta, T. Yoshinaga, and H. Kawakami, "Bifurcation analyses of nonlinear dynamical systems: From theory to numerical computations," *Nonlinear Theory and Its Applications, IEICE*, vol. 3, no. 4, pp. 458–476, 2012.

[6] H. Kawakami, "Bifurcation of periodic responses in forced dynamic nonlinear circuits: Computation of bifurcation values of the system parameters," *IEEE Trans. Circuits and Systems*, vol. 31, no. 3, pp. 248–260, 1984.

[7] E. J. Doedal, *AUTO: Software for continuation and bifurcation problems in ordinary differential equations*, `http://indy.cs.concordia.ca/auto/`.

[8] A. Dhooge *et al.*, "MATCONT: A MATLAB package for numerical bifurcation analysis of ODEs," *ACM Transactions on Mathematical Software*, vol. 29, no. 2, pp. 141–164, 2003.

[9] S. Amoh, M. Ogura, and T. Ueta, "Computation of bifurcations: Automatic provisioning of variational equations," *NOLTA*, vol. 13, pp. 440–445, 2022.

[10] H. Nakajima, ""nonlinear analyses of the processes of learning dynamical systems in neural network," Ph.D. dissertation, Kyoto University, 1995.

[11] K. Doya, "Bifurcations of recurrent neural networks in gradient descent learning," *IEEE Transactions on neural networks*, vol. 1, no. 75, p. 218, 1993.

[12] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, *Activation functions: Comparison of trends in practice and research for deep learning*, 2018. eprint: arXiv:1811.03378.

[13] X. Wang, "Period-doublings to chaos in a simple neural network," in *IJCNN-91-Seattle International Joint Conference on Neural Networks*, vol. II, 1991, 333–339 vol.2.

[14] P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for activation functions," *CoRR*, vol. abs/1710.05941, 2017. arXiv: 1710.05941.

[15] Y. Nishiuchi, T. Ueta, and H. Kawakami, "Stable torus and its bifurcation phenomena in a simple three-dimensional autonomous circuit," *Chaos, Solitons and Fractals*, vol. 27, pp. 941–951, 2006.

[16] Y. A. Kuznetsov, *Elements of applied bifurcation theory*, 2nd ed., Springer, Berlin, 1998.

[17] T.-Y. Li and J. A. Yorke, "Period three implies chaos," *The theory of chaotic attractors*, pp. 77–84, 2004.

[18] E. N. Lorenz, "Deterministic nonperiodic flow," *Journal of atmospheric sciences*, vol. 20, no. 2, pp. 130–141, 1963.

[19] C. Sparrow, *The Lorenz equations: bifurcations, chaos, and strange attractors*, Springer Science & Business Media, vol. 41, 2012.

[20] C. Robinson, *Dynamical systems: stability, symbolic dynamics, and chaos*, CRC press, 1998.

[21] R. M. May, "Simple mathematical models with very complicated dynamics," *Nature*, vol. 261, no. 5560, pp. 459–467, 1976.

[22] M. Hénon, "A two-dimensional mapping with a strange attractor," *The theory of chaotic attractors*, pp. 94–102, 2004.

[23] R. Devaney and Z. Nitecki, "Shift automorphisms in the Hénon mapping," *Communications in Mathematical Physics*, vol. 67, pp. 137–146, 1979.

[24] S. Friedland and J. Milnor, "Dynamical properties of plane polynomial automorphisms," *Ergodic Theory and Dynamical Systems*, vol. 9, no. 1, pp. 67–99, 1989.

[25] H. R. Dullin and J. Meiss, "Generalized Hénon maps: The cubic diffeomorphisms of the plane," *Physica D: Nonlinear Phenomena*, vol. 143, no. 1-4, pp. 262–289, 2000.

[26] M. Benedicks and L. Carleson, "The dynamics of the Hénon map," *Annals of Mathematics*, pp. 73–169, 1991.

[27] M. Benedicks and M. Viana, "Solution of the basin problem for Hénon-like attractors," *Inventiones mathematicae*, vol. 143, no. 2, pp. 375–434, 2001.

[28] M. Benedicks and L.-S. Young, "Sinai-Bowen-Ruelle measures for certain Hénon maps," *The Theory of Chaotic Attractors*, pp. 364–399, 2004.

[29] E. Bedford and J. Smillie, "Real polynomial diffeomorphisms with maximal entropy: Tangencies," *Annals of mathematics*, pp. 1–26, 2004.

[30] E. Bedford and J. Smillie, "Real polynomial diffeomorphisms with maximal entropy: II. small Jacobian," *Ergodic Theory and Dynamical Systems*, vol. 26, no. 5, pp. 1259–1283, 2006.

[31] Y. Cao, S. Luzzatto, and I. Rios, "The boundary of hyperbolicity for Hénon-like families," *Ergodic Theory and Dynamical Systems*, vol. 28, no. 4, pp. 1049–1080, 2008.

[32] X. Zhang, "Hyperbolic invariant sets of the real generalized Hénon maps," *Chaos, Solitons & Fractals*, vol. 43, no. 1-12, pp. 31–41, 2010.

[33] G. Leonov, N. Kuznetsov, and V. Vagaitsev, "Localization of hidden Chua's attractors," *Physics Letters A*, vol. 375, no. 23, pp. 2230–2233, 2011.

[34] G. Chen and T. Ueta, "Yet another chaotic attractor," *International Journal of Bifurcation and chaos*, vol. 9, no. 07, pp. 1465–1466, 1999.

[35] G. A. Leonov and N. V. Kuznetsov, "Hidden Attractors in Dynamical Systems. From Hidden Oscillations in Hilbert-Kolmogorov Aizerman, and Kalman Problems to Hidden Chaotic Attractor in Chua Circuits," *International Journal of Bifurcation and Chaos*, vol. 23, no. 1, 1330002-219, pp. 1 330 002–219, 2013.

[36] G. Chen, N. V. Kuznetsov, G. A. Leonov, and T. N. Mokaev, "Hidden attractors on one path: Glukhovsky–Dolzhansky, Lorenz, and Rabinovich systems," *International Journal of Bifurcation and Chaos*, vol. 27, no. 08, p. 1 750 115, 2017.

[37] S. Jafari, J. Sprott, and F. Nazarimehr, "Recent new examples of hidden attractors," *The European Physical Journal Special Topics*, vol. 224, no. 8, pp. 1469–1476, 2015.

[38] M. Molaie, S. Jafari, J. C. Sprott, and S. M. R. Hashemi Golpayegani, "Simple chaotic flows with one stable equilibrium," *International Journal of Bifurcation and Chaos*, vol. 23, p. 1 350 188, 2013.

[39] J. C. Sprott, "Some simple chaotic flows," *Physical review E*, vol. 50, no. 2, R647, 1994.

[40] S. Jafari, V.-T. Pham, S. M. R. H. Golpayegani, M. Moghtadaei, and S. T. Kingni, "The relationship between chaotic maps and some chaotic systems with hidden attractors," *International Journal of Bifurcation and Chaos*, vol. 26, no. 13, p. 1 650 211, 2016.

[41] H. Jiang, Y. Liu, Z. Wei, and L. Zhang, "Hidden chaotic attractors in a class of two-dimensional maps," *Nonlinear dynamics*, vol. 85, pp. 2719–2727, 2016.

[42] X. Zhang and G. Chen, "Polynomial maps with hidden complex dynamics," *Discrete & Continuous Dynamical Systems - B*, vol. 24, pp. 2941–2954, 6 2019.

[43] G. B. Clayton and S. Winder, *Operational amplifiers*, Elsevier, 2003.

[44] W. Garver and F. Moss, "Electronic fireflies," *Scientific American*, vol. 269, no. 6, pp. 128–130, 1993.

[45] T. Kousaka, H. Kawakami, and T. Ueta, "Synchronization of electric fireflies by using square wave generators," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 81, no. 4, pp. 656–663, 1998.

[46] C. Kuehn *et al.*, *Multiple time scale dynamics*, Springer, vol. 191, 2015.

[47] W. A. Edson, *Vacuum-tube oscillators*, Wiley, 1953.

[48] E. Benoit, "Chasse au canard," *Collectanea Mathematica*, vol. 32, no. 2, pp. 37–119, 1981.

[49] M. Krupa and P. Szmolyan, "Relaxation oscillation and canard explosion," *Journal of Differential Equations*, vol. 174, no. 2, pp. 312–368, 2001.

[50] Y. A. Kuznetsov, "Numerical analysis of bifurcations," *Elements of applied bifurcation theory*, pp. 505–585, 2004.

[51] B. Van der Pol, "Relaxation-oscillations," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 978–992, 1926.

[52] A. Algaba, K.-W. Chung, B.-W. Qin, and A. J. Rodríguez-Luis, "Analytical approximation of the canard explosion in a van der Pol system with the nonlinear time transformation method," *Physica D: Nonlinear Phenomena*, vol. 406, p. 132 384, 2020.

[53] J.-M. Ginoux and J. Llibre, "Canards existence in FitzHugh-Nagumo and Hodgkin-Huxley neuronal models," *Mathematical Problems in Engineering*, vol. 2015, 2015.

[54] A. Makeeva, A. Dmitrichev, and V. I. Nekorkin, "Cycles-canards and torus-canards in a weakly inhomogeneous ensemble of FitzHugh–Nagumo neurons with excitatory synaptic couplings," *Izvestiya VUZ. Applied Nonlinear Dynamics*, vol. 28, no. 5, pp. 524–546, 2020.

[55] M. Courbage, V. I. Nekorkin, and L. V. Vdovin, "Chaotic oscillations in a map-based model of neural activity," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 17, no. 4, 2007.

[56] M. Brøns, "Canard explosion of limit cycles in templator models of self-replication mechanisms," *The Journal of chemical physics*, vol. 134, no. 14, 2011.

[57] J. Rankin, M. Desroches, B. Krauskopf, and M. Lowenberg, "Canard cycles in aircraft ground dynamics," *Nonlinear Dynamics*, vol. 66, pp. 681–688, 2011.

[58] J. Karki, "Effect of parasitic capacitance in op amp circuits," no. SLOA013A, 2000.

[59] S. Doi *et al.*, "Importance of sufficient precision in stable dynamics for the numerical computation of canards in singularly perturbed systems," *Nonlinear Theory and Its Applications, IEICE*, vol. 6, no. 4, pp. 454–465, 2015.

[60] N. Fenichel, "Geometric singular perturbation theory for ordinary differential equations," *Journal of Differential Equations*, vol. 31, no. 1, pp. 53–98, 1979.

[61] C. Kuehn, "From first Lyapunov coefficients to maximal canards," *International Journal of Bifurcation and Chaos in Applied Sciences and Engineering*, vol. 20, no. 05, pp. 1467–1475, 2010.

[62] M. Krupa and P. Szmolyan, "Extending geometric singular perturbation theory to non-hyperbolic points - fold and canard points in two dimensions," *SIAM Journal on Mathematical Analysis*, vol. 33, no. 2, pp. 286–314, 2001.

[63] S. Chow *et al.*, *Normal forms and bifurcation of planar vector fields*, CUP, 1994.

[64] Y. A. Kuznetsov, "Numerical analysis of bifurcations," in *Multiple Time Scale Dynamics (Third Edition)*. Springer, 2004, pp. 505–585.

[65] M. Desroches *et al.*, "Canard and curvature: The 'smallness of ϵ' in slow–fast dynamics," *Proceedings of the Royal Society A*, no. 467, pp. 2404–22 421, 2020.

[66] M. Itoh and R. Tomiyasu, "Experimental study of the missing solutions "canards"," *Transactions of IEICE*, vol. E73, no. 6, pp. 848–854, 1990.

[67] M. Itoh and R. Tomiyasu, "Canards in a nonlinear circuit," *Transactions of IEICE*, vol. E73, no. 11, pp. 1866–1873, 1990.

A List of the Related Papers by the Author

Thesis

[1] S. Amoh, "Optimal solution search by pump rate variance adjustment in ising machine(in Japanese)," Bachelor's Thesis, Tokushima University, 2019.

[2] S. Amoh, "A bifurcation analysis method for high-dimensional hybrid autonomous systems," M.S. thesis, Tokushima University, 2021.

Papers

[3] S. Amoh, D. Ito, and T. Ueta, "A method to suppress local minima for symmetrical DOPO networks," *Nonlinear Theory and Its Applications, IEICE*, vol. 11, no. 4, pp. 580–589, 2020.

[4] S. Amoh, X. Zhang, G. Chen, and T. Ueta, "Bifurcation analysis of a class of generalized Hénon maps with hidden dynamics," *IEEJ Transactions on Electrical and Electronic Engineering*, vol. 16, no. 11, pp. 1456–1462, 2021.

[5] S. Amoh, M. Ogura, and T. Ueta, "Computation of bifurcations: Automatic provisioning of variational equations," *Nonlinear Theory and Its Applications, IEICE*, vol. 13, no. 2, pp. 440–445, 2022.

[6] T. Ueta and S. Amoh, "To tackle bifurcation problems with Python (in Japanese)," *IEICE Fundam. rev.*, vol. 16, no. 3, 2023.

[7] S. Amoh, T. Ueta, and H. Kawakami, "Transient responses to relaxation oscillations in multivibrators," *IEEE Access*, pp. 471–482, 2024.

Proceeding papers

[8] S. Amoh, D. Ito, and T. Ueta, "A method to suppress excited states for high symmetrical DOPO networks," in *Proceedings of the 2019 IEICE International Symposium on Nonlinear Theory and its Applications*, IEICE, Kuala Lumpur, Malaysia, 2019, pp. 173–176.

[9] S. Amoh and T. Ueta, "Bifurcation analysis of 3d hybrid autonomous system," in *Proceedings of the 2020 IEICE International Symposium on Nonlinear Theory and its Applications*, IEICE, Zoom Conference, 2020.

[10] S. Amoh, M. Ogura, and T. Ueta, "A method to obtain variational equations automatically by applying symbolic differentiation," in *Proceedings of the 2021 Nonlinear Science Workshop*, IEICE, Zoom Conference, 2021.

[11] S. Amoh and T. Ueta, "Multivibrator with slow–fast dynamics," in *Proceedings of the 2023 IEICE International Symposium on Nonlinear Theory and its Applications*, IEICE, Catania, Italy, 2023, pp. 13–16.

[12] S. Amoh and T. Ueta, "Design and development of bifurcation analysis software using Python," in *Proceedings of the 2023 IEICE International Symposium on Nonlinear Theory and its Applications*, IEICE, Catania, Italy, 2023, pp. 699–702.

Workshop articles

[13] S. Amoh, D. Ito, Y. Miino, and T. Ueta, "Optimal solution search by pump rate variance adjustment in Ising machine(in Japanese)," in *Proceedings of the 2019 IEICE NOLTA society conference*, IEICE, Nagaoka, Japan, 2019, NLS–31.

[14] S. Amoh, *Suppression of suboptimal solutions of Ising machine (in Japanese)*, Kagawa, Japan: NBA, 2019.

[15] S. Amoh, T. Ueta, and H. Kawakami, "Canard generation of multivibrators (in Japanese)," in *Proceedings of the 2021 Shikoku-Section Joint Convention of the Institutes of Electrical and related Engineers*, Shikoku-Section Joint Convention of the Institutes of Electrical and related Engineers, Zoom Conference, 2021, p. 12.

[16] R. Yoshida, S. Amoh, and T. Ueta, "How does a 3d autonomous system without equilibria generate periodic solutions? (in Japanese)," in *Proceedings of the 2021 IEICE NOLTA society conference*, IEICE, Zoom Conference, 2021, NLS–21.

[17] S. Amoh, T. Ueta, and H. Kawakami, "Multivibrator and canard (in Japanese)," in *Proceedings of the 2022 IEICE NOLTA society conference*, IEICE, Osaka, Japan, 2022, NLS–19.

[18] S. Amoh, T. Ueta, and H. Kawakami, "Multivibrator as slow-fast dynamics — basic bifurcation phenomena and canard solutions —(in Japanese)," in *IEICE Technical Report*, IEICE, vol. 122, Osaka, Japan, 2022, pp. 72–77.

[19] M. Yoshikawa, S. Amoh, and T. Ueta, "Bifurcation analysis tools by Python (in Japanese)," in *Proceedings of the 2022 Shikoku-Section Joint Convention of the Institutes of Electrical and related Engineers*, Shikoku-Section Joint Convention of the Institutes of Electrical and related Engineers, Tokushima, Japan, 2022, p. 1.

[20] M. Katayama, S. Amoh, and T. Ueta, "Bifurcations of periodic orbits in synaptically coupled FHN model (in Japanese)," in *Proceedings of the 2022 Shikoku-Section Joint Convention of the Institutes of Electrical and related Engineers*, Shikoku-Section Joint Convention of the Institutes of Electrical and related Engineers, Tokushima Japan, 2022, p. 2.

[21] S. Amoh and T. Ueta, "Multivibrator as slow-fast dynamics — canard explosion and its circuit response — (in Japanese)," in *IEICE Technical Report*, IEICE, vol. 123, Tokyo, Japan, 2023, pp. 61–64.

[22] S. Amoh and T. Ueta, "Multivibrator with canard explosion and its circuit response (in Japanese)," in *Proceedings of the 2023 IEICE NOLTA society conference*, IEICE, Tokyo, Japan, 2023, p. 31.

[23] M. Saito, S. Amoh, and T. Ueta, "Bifurcation of the Hindmarsh-Rose model under electromagnetic induction (in Japanese)," in *Proceedings of the 2023 Shikoku-Section Joint Convention of the Institutes of Electrical and related Engineers*, Shikoku-Section Joint Convention of the Institutes of Electrical and related Engineers, Tokushima Japan, 2023, p. 43.

[24] S. Amoh and T. Ueta, "Canard cycles in multivibrators with piece-wise constant identical operational amplifier (in Japanese)," in *IEICE Technical Report*, IEICE, vol. 123, Tokushima, Japan, 2024, pp. 61–64.

[25] M. Yoshikawa, S. Amoh, and T. Ueta, "Bifurcation of three Wilson-Cowan neuron models (in Japanese)," in *IEICE Technical Report*, IEICE, vol. 123, Tokushima, Japan, 2024, pp. 61–64.

Mentions

[26] ***Student Paper Award***, NOLTA2019, 2019.

[27] ***NOLTA Society Paper Award***, NOLTA Society 2022, 2022.

[28] ***Specified recommended paper***, NonLinear Problem Study Group; NLP, IEICE, 2022.

[29] ***IEEJ Top Downloaded Article***, Wiley, 2023.

[30] ***Honarable Mention***, NonLinear Problem Study Group; NLP, IEICE, 2023.

Acknowledgement

This research was conducted under the guidance of Professor *Tetsushi Ueta*. I am deeply grateful to get his enthusiastic guidance. I remember I decided to belong to Professor *Ueta*'s laboratory while I take the electronics circuit class during my undergraduate studies. The way Professor *Ueta* strived to convey the essence to students motivated with me, and I sensed a deep dedication to research guidance. Over the six years, Professor *Ueta* has been considerate of my research in ways I am both aware of and unaware of. The wonderful research life I have experienced is unequivocally due to Professor *Ueta*'s cooperation. Going forward, I will continue to be humble and dedicated to self-improvement.

Professor Emeritus *Hiroshi Kawakami* guided my research for three years during my doctoral program. I am especially grateful for the enthusiastic guidance provided on the discussion of slow-fast dynamical systems included in this thesis. Having the opportunity to conduct research with one of the scientist who made the history of dynamical systems is really amazing experience for me. Their emphasis on "understanding the essence of science" is exactly in line with Professor *Ueta*'s dedication to continuing Professor *Kawakami*'s legacy. I am truly grateful for all the support and guidance.

Professor *Kenji Matsuura*, Associate Professor *Masahiko Sano*, and Associate Professor *Hiroki Tanioka*, who are faculty members of A6 laboratory, gave me a lot of advice in many fields. I am honored to be able to carry out my research under the A6 laboratory. I would also like to thank everyone in the laboratory for their cooperation in various situations. Especially, I am deeply thank to my colleague *Shunya Izumi*, *Kaoru Kimura*, *Takahiro Kobayashi*, *Shu Yano*, *Takahiro Oriishi*, *Naoya Kohda*, *Miho Ogura*, *Rina Yano*, *Rikiya Yamaguchi*, *Shintaro*

Fujimoto. We influenced each other and shared a fulfilling research experience. I am also grateful to *Michiru Katayama*, with whom I conducted research during my doctoral program. *Michiru Katayama* is extremely dedicated to research and shared his knowledge of mathematics and physics with me. *Quankai Zhang* and *Marian Cichy* were international students in our laboratory. They shared me a lot of international knowledge. I am also grateful to Dr. *Akinori Tsuji*. When I was a undergraduate student, the book on operating systems that he introduced to me changed my life. He is the person who inspired me to take a doctoral course.

In the summer of 2019, I visited Holland and it was the first experience of going to abroad. I'm very grateful to meet *Milan Van Tuin-de Wolff* in Holland. The first time we met was in a Izakaya restaurant *Shu* in front of Kuramoto station. He was a pilgrim for 88 temples of Shikoku. He really motivated me to learn English and foreign cultures.

In the winter of 2021, I visited Institute of Applied Physics, Russian Academy of Science in Russia, Nizhny Novgorod. I am grateful to Professor *Vladmir Nekorkin* for kindly research guidance. Life in Russia during winter is very very cold, but the kindness of people softens it. I also express my deep gratitude to Dr. *Dmitry Shchapin*, who supported my daily life, and Dr. *Mechislav Pugavko*, who cooperated in my research. I would also like to thank everyone in the laboratory. Although traveling is difficult now, but let us share vodka again.

My research life during the COVID pandemic was lonely, but friends living far away helped my life. While I would like to list all the friends who supported me, due to space constraints, I will mention only a few names here. *Naoki Okada*, I am deeply grateful of your maintained international community. *Sampo Kokko* from Finland, he is one of the biggest reasons why I speak English. ES PARISEE. *Lee Doyeon* a.k.a. `DOBOB/DS1TLZ` from Korea, his engineering theory motivated me a lot. *Dor Windman* from Israel, she cheered me very much. *Vladimir Sobolev* from Russia, now in Tokyo, I am grateful for his cultural supports while I'm in Russia. *Egor Kuzmin* from Russia, now in Cyprus, he has always encourage me.

I would like to thank *Tatsuya Fujita*, the owner of *NEAT BAR*. It was 18 y.o. when I start the part-time job at the authentic bar. This experience has influenced my personal development throughout my growth process. He provide me such a lot of social experiences. I would like to

thank *Takumi Sakurada* from Tokushima as well. We made bunch of story in our city from the age of Tokushima Johoku high school. I remember we share cheep beer and few tobaccos in the cold winter park, since we were so much poor at that day. *Shu* and *Nice*, the places we share a lot of beer, was gone. We spent such a long time. However, these experience made us strong enough right. We strongly take a step forward together.

Finally, I am grateful to my parents *Hisanao* and *Kazumi*, my brother *Yuuki* and my sister *Rina*, my grandparents *Shigeko* and *Hiroshi Hukuoka* and *Sueko Hukuoka*, my uncle *Toshio Hukuoka*. Today, I am 27 years old. While I'm writing this. I was a child like requiring a lot of attention, however, from now, I am committed to contributing to the betterment of human society and the advancement of science and technology. Just like all of you, I never forget the value of sincerity. Thank you very much and I love you.

From my motherland Tokushima,

Seiya Amoh
Tokushima University Graduate School of Advanced Technology and Science,
College of Systems Innovation Engineering,
Information Science and Intelligent Systems, Tokushima University