A numerical framework for designing periodic orbits embedded in chaotic attractors

Hidetaka Ito\textsuperscript{1a)}, Hiroomi Hikawa\textsuperscript{1}, and Yutaka Maeda\textsuperscript{1}

\textsuperscript{1}Faculty of Engineering Science, Kansai University
3-3-35 Yamate-cho, Suita, Osaka 564-8680, Japan

\textsuperscript{a)}h.ito@kansai-u.ac.jp

Received September 19, 2018; Revised December 30, 2018; Published April 1, 2019

Abstract: This paper proposes a framework for the numerical design of continuous-time dynamical systems that bind desirably configured unstable periodic orbits (UPOs) into a chaotic attractor. The proposed numerical framework is comprised of the following four steps: (a) construction of a chaos-generating template structure consisting of a set of trajectories including explicitly embedded UPOs, (b) topology-preserving deformation of the template structure according to the desired configuration of UPOs, (c) assignment of attracting properties to the deformed template structure, and (d) function approximation that yields a vector-field function of a dynamical system that generates a chaotic attractor with desired UPOs. This paper elaborates on each step of the framework and presents a reference implementation, supported with numerical examples. From the viewpoint of numerical vector-field design, our proposal intends to extend the functionality of (stable) limit-cycle generators by introducing transitivity among configured UPOs that is intrinsic to chaotic systems.

Key Words: chaos, unstable periodic orbits, vector field, numerical design

1. Introduction

The progress of the analysis and understanding of the richness of nonlinear dynamics has stimulated investigations on its applications to intelligent and flexible systems in many fields including neurocomputing, communications technology, computer vision, and robotics. Among various nonlinear phenomena studied in this context such as the generation of limit cycles, their bifurcations, and synchronizations, chaotic phenomena have also attracted strong interests particularly from the viewpoints of complex and adaptive behaviors. Thus, the synthesis of chaotic dynamical systems from various approaches [1–3] has for some time been an active direction of research. Primary concerns of these approaches include statistical and topological characteristics (e.g., invariant measure, Lyapunov spectrum, novel scrolling behaviors) that would be important in designing chaos-based information processing and communication applications.

Another important aspect of chaos, which underlies the present study, is that chaotic attractors embed an infinite number of unstable periodic orbits (UPOs) bifurcated from pre-chaotic states [4].
Among them, some distinctive orbits can be used for characterization or control purposes. For example, a variety of chaos control methods [5–7] can stabilize UPOs embedded in chaotic attractors, enlarging the operation range of the system. In the present study, we view such chaotic attractors as a container of UPOs (patterns, waveforms) where they can be stabilized or targeted by external inputs into the dynamical system.

Thus, this paper, while sharing some common motivation with the studies mentioned earlier for the synthesis of chaos, we put primary focus on realizing tailored UPOs in a chaotic attractor. In particular, we propose a framework for the numerical design of continuous-time dynamical systems (in the form of differential equations) that bind desirably configured UPOs into a chaotic attractor governed by a vector field (flow). From the viewpoint of numerical vector-field design, our proposal intends to extend the functionality of (stable) limit-cycle generators [8, 9], by introducing transitivity among configured UPOs that is intrinsic to chaotic systems.

2. Proposed framework

In this paper, we propose a framework for the numerical design of continuous-time dynamical systems $\dot{x} = f(x)(x \in \mathbb{R}^n)$ that bind desired UPOs into a chaotic attractor. This framework is comprised of the following four steps, all implemented numerically:

(a) construction of a template structure consisting of a set of state-space trajectories, including explicitly embedded skeleton-forming UPOs, that obeys some chaos-generating mechanism

(b) topology-preserving deformation of the template structure for placing desired UPOs

(c) assignment of attracting properties to the deformed template structure

(d) function approximation of a vector-field function $f(x)$ that generates a chaotic attractor with desired UPOs.

The first three steps (a)–(c) form the core of the novelty of our proposal that allows the compatible and simultaneous design of periodic orbits and a chaotic attractor in a modular and systematic way. These steps can be seen as a design process of a training set that is to be used in (d). In what follows, we elaborate on these steps, presenting a reference implementation by adopting one of most fundamental mechanisms of chaos generation. Note, however, that the modularity of the proposed framework allows us to choose alternative implementations as long as the key strategy is maintained.

An earlier version of this framework employing less flexible implementations in steps (b) and (d) was presented in [10], and a digest of the current proposal with a partial demonstration was presented in [11].

2.1 Construction of a chaos-generating template structure

The first and most fundamental step is to construct a template structure consisting of a set of state-space trajectories, including explicitly embedded skeleton-forming UPOs, that obeys some chaos-generating mechanism such as simple stretching and folding. A key consideration here is to embed some UPOs explicitly as trajectories within the trajectory set. This allows us to deform the template structure, i.e. the set of trajectories, in reference to the desired configuration of UPOs.

Now we proceed with the details, presenting a reference implementation by adopting the tent-map mechanism, one of most fundamental mechanisms of chaos generation. We here consider the one-dimensional map shown in Fig. 1 for the $x$-coordinate of the $n$-th crossing on the Poincaré section $\Sigma = \{(x, y, z)|y = 0, x \geq 0\}$. This Poincaré section is indicated in gray in Fig. 2 (drawn from two viewpoints), where each trajectory crosses the Poincaré section in anticlockwise direction when seen from above. We consider that one of the simplest situations of continuous-time trajectories that lead to the tent map in Fig. 1 is a combination of the “stretching” at the rate of twice per rotation (mapping), corresponding to the absolute slope of the map, and the “folding” at the rate of once per rotation (mapping), corresponding to the unimodality of the map. Thus, we here propose one such construction (detailed in Eqs. (1)–(6)) of a bundle of trajectories shown in Fig. 2 as a
nominal chaotic flow. According to this construction, the trajectories leaving the Poincaré section at \((x_0, 0, 0)\) with \(0.5 \leq x_0 \leq 1\) will remain on the \(xy\) plane during rotation and return to the Poincaré section at \((0.5 + 2(x_0 - 0.5), 0, 0)\). On the other hand, the trajectories leaving the Poincaré section at \((x_0, 0, 0)\) with \(1 < x_0 \leq 1.5\) will be “lifted” and “folded” in the \(xyz\) space and return to the section at \((1.5 - 2(x_0 - 1), 0, 0)\). It is important to note here that this flow (bundle of trajectories) embeds a period-1 UPO that leaves the Poincaré section at \((7/6, 0, 0)\) and returns to the same point on the section, corresponding to the fixed point of the tent map.

The individual trajectories in Fig. 2 are numbered with index \(i\) from the innermost \((i = 0)\) to the outermost \((i = i_{\text{max}})\) when starting from the Poincaré section. Each trajectory is represented by a set of points (plotted with lines in figures) that are numbered anticlockwise with index \(j\) from the Poincaré section \((j = 0)\) to the final point after (almost) one rotation \((j = j_{\text{max}})\). With this notation, the points constituting the bundle of trajectories are denoted by their positions \((r_{ij}, \theta_{ij}, z_{ij})\), and these positions are obtained by transforming predetermined polar coordinate expressions \((r_{ij}, \theta_{ij}, z_{ij})\) into the Cartesian coordinates.

The stretching and folding properties in Fig. 2 can be conveniently formulated in polar coordinates, and their coordinate values are given by

\[
\begin{align*}
r_{ij} &= \begin{cases} 
  r'_{ij} & \text{if } r'_{ij} < r_{th} \\
  r_{th} + (r'_{ij} - r_{th}) \cos \frac{j \Delta \theta}{2} & \text{otherwise,}
\end{cases} \\
\theta_{ij} &= j \Delta \theta, \\
z_{ij} &= \begin{cases} 
  0 & \text{if } r'_{ij} < r_{th} \\
  (r'_{ij} - r_{th}) \sin \frac{j \Delta \theta}{2} & \text{otherwise,}
\end{cases}
\end{align*}
\]

where

\[\frac{\Delta x}{x_{n+1}} = \frac{\Delta x}{x_{n}} \approx 1\]
Fig. 3. The stretching and folding characteristics of the nominal chaotic flow.

\[
\begin{align*}
    r'_{ij} &= r_{min}^P + i\Delta r \left(1 + \frac{j\Delta\theta}{2\pi}\right), \\
    \Delta r &= (r_{max}^P - r_{min}^P)/i_{max}, \\
    \Delta\theta &= 2\pi/j_{max}.
\end{align*}
\]

Here, \(r_{th}\) is the threshold radius for folding, and \(r_{min}^P\) and \(r_{max}^P\) are the minimum and maximum radius on the Poincaré section, respectively. Figure 3(a) shows the trajectories defined by Eqs. (4) and (2) with \(r_{min}^P = 0.5\), \(r_{max}^P = 1.5\), and \(i_{max} = 30\). At this stage, the trajectories exhibit the stretching between neighboring trajectories at the rate of twice per rotation. The factor \(1 + \frac{j\Delta\theta}{2\pi}\) in Eq. (4), which varies from 1 to 2 during one rotation, determines the amount of stretching along the direction of the radius. Figure 3(b) shows the trajectories defined by Eqs. (1) and (2) with \(r_{th} = 1.5\). In this implementation, the folding is formulated as a continuous rotation of the segment \(r'_{ij} - r_{th}\) within the rz plane, and takes place at the rate of once per rotation. The angle \(\frac{j\Delta\theta}{2}\) in Eqs. (1) and (3), which varies from 0 to \(\pi\) during one rotation, determines the amount of folding around the point with the threshold radius \(r_{th}\).

The above strategy for embedding a single period-1 UPO can be extended to embed, or bind, several period-1 UPOs by considering the \(N_c\) times composite of the nominal one-dimensional (tent) map. For example, we can construct a bundle of trajectories shown in Fig. 4 that leads to the twice composite \((N_c = 2)\) tent map shown in Fig. 5. The polar coordinate values of \((r_{ij}, \theta_{ij}, z_{ij})\) in this case are given by

\[
\begin{align*}
    r_{ij} &= \begin{cases} 
    r'_{ij} & \text{if } r'_{ij} < r_{th} \\
    r_{th} + (r'_{ij} - r_{th}) \cos \left(\frac{j \mod (j_{max}/N_c)(N_c\Delta\theta)}{2}\right) & \text{otherwise}
    \end{cases}, \\
    \theta_{ij} &= j\Delta\theta, \\
    z_{ij} &= \begin{cases} 
    0 & \text{if } r'_{ij} < r_{th} \\
    (r'_{ij} - r_{th}) \sin \left(\frac{j \mod (j_{max}/N_c)(N_c\Delta\theta)}{2}\right) & \text{otherwise}
    \end{cases},
\end{align*}
\]

where

\[
    r'_{ij} = r_{min}^P + i\Delta r \left(1 + \frac{(j \mod (j_{max}/N_c))(N_c\Delta\theta)}{2\pi}\right).
\]

Note that the vertical axis of Fig. 5 has been relabelled from \(x_{n+2}\), meaning this is the twice composite of the tent map, to \(x_{n+1}\), meaning this is one iteration of the Poincaré map of the flow in Fig. 4 because the arguments in the rest of this paper will be based on the flow in Fig. 4.

From the one-dimensional map in Fig. 5, we see that the flow now embeds three period-1 UPOs, corresponding to three fixed points of the map, whose continuous-time trajectories are shown in Fig. 6. Incidentally, according to Fig. 5, the orbit starting from \((x, y, z) = (0.5, 0, 0)\) is also periodic.
However, we will omit this orbit from the present study because this orbit can easily detach from the resulting chaotic attractors. Now we consider the above bundle of trajectories as a (three-dimensional) chaos-generating template structure.

The three period-1 UPOs discussed so far are entangled one another as shown in Fig. 6, which poses restrictions on the configuration of the deformed UPOs. Therefore, we further consider introducing another, fourth dimension for the state space and embedding the template structure into this four-dimensional ($xyzw$) space. Here we propose an implementation in which we set $w_{ij} = \sqrt{x_{ij}^2 + y_{ij}^2}$ to obtain a set of four-dimensional points $(x_{ij}, y_{ij}, z_{ij}, w_{ij} = \sqrt{x_{ij}^2 + y_{ij}^2})$.

In the subsequent steps, this template structure will be represented and handled as a set of discrete
points
\[ \mathbf{x}_{ij} = (x_{ij}, y_{ij}, z_{ij}, w_{ij}), \] (11)
hereafter referred to as the design points. In particular, the template structure described above, i.e. before deformation (in Section 2.2), is denoted with superscript \( \text{tmpl} \) as
\[ \mathbf{x}_{ij}^{\text{tmpl}} = (x_{ij}^{\text{tmpl}}, y_{ij}^{\text{tmpl}}, z_{ij}^{\text{tmpl}}, w_{ij}^{\text{tmpl}}), \] (12)
and with this notation, the three period-1 UPOs embedded in the template structure are represented by the following three subsets:
\[ \mathbf{x}_{ij}^{\text{tmpl}, a} = (x_{ij}^{\text{tmpl}, a,j}, y_{ij}^{\text{tmpl}, a,j}, z_{ij}^{\text{tmpl}, a,j}, w_{ij}^{\text{tmpl}, a,j}), \] (13)
\[ \mathbf{x}_{ij}^{\text{tmpl}, b} = (x_{ij}^{\text{tmpl}, b,j}, y_{ij}^{\text{tmpl}, b,j}, z_{ij}^{\text{tmpl}, b,j}, w_{ij}^{\text{tmpl}, b,j}), \] (14)
\[ \mathbf{x}_{ij}^{\text{tmpl}, c} = (x_{ij}^{\text{tmpl}, c,j}, y_{ij}^{\text{tmpl}, c,j}, z_{ij}^{\text{tmpl}, c,j}, w_{ij}^{\text{tmpl}, c,j}), \] (15)
where \( \text{idx}_a, \text{idx}_b, \) and \( \text{idx}_c \) denote the index numbers for orbits \( a, b, \) and \( c, \) respectively. In the case with \( N_c = 2 \) and \( i_{\text{max}} = 30, \) these index numbers are found to be \( \text{idx}_a = 12, \text{idx}_b = 24, \) and \( \text{idx}_c = 20. \)

2.2 Topology-preserving deformation of the template structure for placing desired periodic orbits

In order to utilize the UPOs for specific, pattern generation applications, we need to deform the UPOs embedded in the template structure according to the desired dynamical patterns and, at the same time, deform the entire template structure in a topology-preserving way for maintaining the chaotic dynamics. We will handle this deformation as if we anchor three UPOs drawn on a rubber-sheet structure to desired positions and let all the other parts of the structure relax to equilibrium. More specifically, the goal of the deformation process is to deform the entire template structure so that the three target period-1 orbits \( \mathbf{x}_{ij}^{\text{tgt}, a}, \mathbf{x}_{ij}^{\text{tgt}, b}, \mathbf{x}_{ij}^{\text{tgt}, c}, \) specified by the user,
\[ \mathbf{x}_{ij}^{\text{tgt}, a} = (x_{ij}^{\text{tgt}, a,j}, y_{ij}^{\text{tgt}, a,j}, z_{ij}^{\text{tgt}, a,j}, w_{ij}^{\text{tgt}, a,j}), \] (16)
\[ \mathbf{x}_{ij}^{\text{tgt}, b} = (x_{ij}^{\text{tgt}, b,j}, y_{ij}^{\text{tgt}, b,j}, z_{ij}^{\text{tgt}, b,j}, w_{ij}^{\text{tgt}, b,j}), \] (17)
\[ \mathbf{x}_{ij}^{\text{tgt}, c} = (x_{ij}^{\text{tgt}, c,j}, y_{ij}^{\text{tgt}, c,j}, z_{ij}^{\text{tgt}, c,j}, w_{ij}^{\text{tgt}, c,j}), \] (18)
\((j = 0, \ldots, j_{\text{max}})\) are embedded in the deformed template \( \mathbf{x}_{ij}^{\text{defd}} \)
\[ \mathbf{x}_{ij}^{\text{defd}} = (x_{ij}^{\text{defd}, i,j}, y_{ij}^{\text{defd}, i,j}, z_{ij}^{\text{defd}, i,j}, w_{ij}^{\text{defd}, i,j}). \] (19)

Note that while this goal is expressed as
\[ \mathbf{x}_{ij}^{\text{defd}} = \mathbf{x}_{ij}^{\text{tgt}}, \] (20)
\[ \mathbf{x}_{ij}^{\text{defd}} = \mathbf{x}_{ij}^{\text{tgt}}. \] (21)
the locations of the remaining design points after deformation will exhibit a small dependence on various settings of the deformation procedure.

In the present implementation, we deform the template structure using a mass-spring model, where we treat the design points \( \mathbf{x}_{ij} \) as mass points that are connected to eight neighboring mass points with springs. Using this setting, the deformation process is formulated as gradual adjustment of the mass point positions \( \mathbf{x}_{ij} \) from their initial positions \( \mathbf{x}_{ij}^{\text{tmpl}} \) toward their final positions \( \mathbf{x}_{ij}^{\text{defd}} \) that will constitute the deformed template structure. The gradual adjustment is performed according to the iterative formula
\[ x_{ij}^{m+1} = x_{ij}^m + \alpha \Sigma_{pq \in N(ij)} g(|x_{pq} - x_{ij}| - \ell_{ijpq}) \frac{x_{pq} - x_{ij}}{|x_{pq} - x_{ij}|} \]  

(23)

where, \( m \) is the iteration count, \( \alpha \) is the step size, \( N(ij) \) denotes the set of indices of eight mass points \( x_{pq} \) neighboring the point \( x_{ij} \), \( g(\cdot) \) represents the restoring force function that obeys Hooke’s law, and \( \ell_{ijpq} \) denotes the natural length of the spring connecting the mass points \( x_{ij} \) and \( x_{pq} \). These natural lengths are set at the beginning to the initial distances between \( x_{ij} \) and \( x_{pq} \).

During the deformation, only the positions of the design points on the three period-1 orbits \( x_{idx,a,j} \), \( x_{idx,b,j} \), and \( x_{idx,c,j} \) are specified (moved) by the user (as explained below), and the positions of all the other points are computed by Eq. (23) that converges toward an equilibrium configuration of the mass-spring structure. The second term of the right-hand side of Eq. (23) gives the direction of the net spring force acting on the mass point \( x_{ij} \).

The whole deformation procedure is divided into sequentially executed \( k_{\text{max}} \) sub-deformations. At the beginning of each sub-deformation denoted with index \( k = 1, \ldots, k_{\text{max}} \), the natural lengths \( \ell_{ijpq} \) of all the springs are reset to their current lengths, and the positions of \( x_{idx,a,j} \), \( x_{idx,b,j} \), and \( x_{idx,c,j} \) are preset to their sub-targets given by

\[ x_{j}^{\text{subtgt,a}} = (1 - k/k_{\text{max}}) x_{j}^{\text{tmpl,a}} + (k/k_{\text{max}}) x_{j}^{\text{tgt,a}} \]  

(24)

\[ x_{j}^{\text{subtgt,b}} = (1 - k/k_{\text{max}}) x_{j}^{\text{tmpl,b}} + (k/k_{\text{max}}) x_{j}^{\text{tgt,b}} \]  

(25)

\[ x_{j}^{\text{subtgt,c}} = (1 - k/k_{\text{max}}) x_{j}^{\text{tmpl,c}} + (k/k_{\text{max}}) x_{j}^{\text{tgt,c}} \]  

(26)

respectively. Then, all the mass points \( x_{ij} \) (except for the above preset points \( x_{idx,a,j} \), \( x_{idx,b,j} \), and \( x_{idx,c,j} \)) are allowed to move gradually according to the iterative formula (23).

### 2.3 Assignment of attracting properties to the deformed template structure

From the positions of the design points (state vectors) on the deformed template structure (bundle of trajectories), velocity vectors are obtained at each design point using a simple finite difference formula such as

\[ (\dot{x}_{ij}, \dot{y}_{ij}, \dot{z}_{ij}, \dot{w}_{ij}) = (x_{i,j+1} - x_{ij}, y_{i,j+1} - y_{ij}, z_{i,j+1} - z_{ij}, w_{i,j+1} - w_{ij}) / \Delta t \]  

(27)

where the superscript \( defd \) is omitted for simplicity, and in the present implementation, the transit time \( \Delta t \) between two successive points is treated as a constant. The resulting pairs of the state vector \( (x_{ij}, y_{ij}, z_{ij}, w_{ij}) \) and the velocity vector \( (\dot{x}_{ij}, \dot{y}_{ij}, \dot{z}_{ij}, \dot{w}_{ij}) \) constitute a major part of the training set used in the subsequent step for function approximation of a vector-field function \( f(x) \) that generates a chaotic flow along the deformed template structure. Here, even though our primary design target is a set of UPOs, it is desirable that the chaotic flow should be of attracting type instead of repelling or other nonattracting types. Thus we assign attracting properties by setting appropriate velocity vectors in the vicinity of the template structure. This can be done in various ways, and we here propose a method that we consider to be relatively easy to handle because of the small number of design parameters. As an implementation of this method, we place additional pairs of attracting target velocity vectors and design points placed at some small distances, \( \Delta z \) and \( \Delta w \), from the deformed template structure

\[ (\dot{x}_{ij}, \dot{y}_{ij}, \dot{z}_{ij} \pm \lambda v, \dot{w}_{ij}) \quad \text{at} \quad (x_{ij}, y_{ij}, z_{ij} \pm \Delta z, w_{ij}) \]  

(28)

\[ (\dot{x}_{ij}, \dot{y}_{ij}, \dot{z}_{ij}, \dot{w}_{ij} \pm \lambda v) \quad \text{at} \quad (x_{ij}, y_{ij}, z_{ij}, w_{ij} \pm \Delta w) \]  

(29)

where \( v = \sqrt{(\dot{x}_{ij})^2 + (\dot{y}_{ij})^2 + (\dot{z}_{ij})^2 + (\dot{w}_{ij})^2} \) and the double-signs correspond. The coefficient \( \lambda \) defines the degree of transverse stability of the flow along the deformed template structure. Note that, while the configuration of the additional design points would be more geometrically consistent if it reflects the curvature and fine structure of the deformed template, we have adopted the above configuration for a great simplicity.
2.4 Function approximation of the vector-field function $f(x)$

The final step, function approximation, is a rather general subject that is not particularly tied to chaos generation with desirably configured UPOs. Nevertheless, this step is essential for constructing the final form of the dynamical system $\dot{x} = f(x)$, and therefore we briefly describe and make some comments on this element here. In the present implementation, we adopt a classical three-layer neural network with sigmoidal nodes for function approximation. This form of neural-network based dynamical system was considered in studies on learning chaotic dynamics using various learning rules [12] and also in a study of generating chaotic dynamics using a genetic-algorithm-type weight search [13, 14]. In the present study, we use an explicit learning of the vector-field function $f(x)$ based on backpropagation with a training set comprising pairs of $x \in \mathbb{R}^n$ and the desired value of $f(x) \in \mathbb{R}^n$. This is feasible because we construct the training set from scratch in the process involving the steps (a)–(c).

3. Numerical example

In this section, we present some implemented examples to show how our proposal works, also adding further comments on implementation considerations. Here, the adopted template structure before deformation is the same as the example shown in Fig. 4, i.e., $r_{\text{min}}^P = 0.5, r_{\text{max}}^P = 1.5, r_{\text{th}} = 1.5, N_c = 2, i_{\text{max}} = 30, j_{\text{max}} = 719$, and $i = 12, 24, 20$ correspond to the three nominal period-1 UPOs $a, b, c$, respectively.

In what follows, we consider the three coexisting periodic orbits shown in Fig. 7 as the design target. These orbits are denoted in Eqs. (16), (17), and (18) as $x_{tgt}^a_j$, $x_{tgt}^b_j$, and $x_{tgt}^c_j$, respectively. Figure 8 shows the progress of the deformation process of the template structure projected onto the $xy$ plane. The red curves represent periodic orbits embedded in the template structure. The deformation process was divided into $k_{\text{max}} = 500$ sub-deformations, and the snapshots are presented for the initial configuration, intermediate configurations after 150 and 350 sub-deformations, and the final, completely deformed configuration of the template structure embedding three target orbits.

While the observed degree of smoothness of the deformed template structure (typically obtained for sufficiently large values of $k_{\text{max}}$) suggests the success of the deformation process, the eventual success will depend on the consistency of the deformed trajectories when viewed as a vector field. It should be noted that, since the template has a sheet-like structure, valid (almost topology-preserving) deformation can lead to a vector field that is not permissible for a differential-equation system or too difficult to realize through the later step of neural-network learning. Figure 9 shows the completely deformed template structure (projection onto three-dimensional, $xyz$ and $xyw$ spaces). These figures show how the target periodic orbits, specified in two dimensions as in Fig. 7, are embedded in the four-dimensional deformed template structure.

The above result of deformation determines the positions of the $(i_{\text{max}} + 1)(j_{\text{max}} + 1) = 31 \times 720 = 22,620$.
22320 design points that constitute the 31 trajectories, which are then converted by Eq. (27) into the 22320 pairs of the state vector and the velocity vector. Assigning attracting properties using Eqs. (28) and (29) with $\Delta z = 0.01, 0.02$ and $\Delta w = 0.01, 0.02$ completes the preparation of the training set for the subsequent step of function approximation. Thus the number of the prepared pairs is $22320(\text{on the template}) + 22320 \times 8(\text{in the neighborhood}) = 200800$ in total. The value of $\lambda$ was set as $\lambda = 0.1$ for $\Delta z, \Delta w = 0.01$ and $\lambda = 0.2$ for $\Delta z, \Delta w = 0.02$.

The prepared pairs of the state vector and the velocity vector is then used as the training set for the backpropagation learning of a three-layer neural network with four input nodes, four output nodes, and 300 intermediate-layer nodes. Upon convergence of learning, we obtain the vector-field...
function $f(x)$, which constitute a neural-network based four dimensional dynamical system $\dot{x} = f(x)$. Figure 10 shows the chaotic attractor of the neural-network based dynamical system, obtained by the fourth-order Runge-Kutta method. The resulting chaotic attractor confirms that the chaos generating mechanism expressed in the form of the deformed template structure is realized as a single trajectory.
that obeys a vector field. This also implies that the training set derived from the deformed template structure was actually consistent when viewed as a vector field, and the neural-network learning was successfully performed. We have also observed good overall transverse stability of the attractor as a result of the simple scheme in Eqs. (28) and (29) for assigning attracting properties. Although this scheme is not optimal, a majority of the additional design points given in this scheme can yield velocity vectors that assist transverse stability, and the generalization ability of the neural-network learning is considered to smooth out inconsistent velocity vectors yielded by a small number of design points.

Finally, Fig. 11 shows the comparison of the target periodic orbits and the corresponding periodic orbits that have been realized in the chaotic attractor. The realized periodic orbits were extracted by an iterative fixed-point procedure by Schmelcher and Diakonos [15]. For all the three targets, the obtained UPO (red) is in good agreement with the desired one (blue).

4. Conclusion

We have proposed a framework for the numerical design of chaotic vector field with desirably configured periodic orbits, and demonstrated reference implementations where we have successfully embedded target periodic orbits while keeping them unstable and transitive but confined in an attractor. The presented examples have shown good feasibility and precision of each step and overall. Owing to the modularity of the proposed framework, it will be interesting to consider various implementation alternatives including e.g. other choices of the underlying structure of chaotic attractors [16], and choices of emerging function approximators such as extreme learning machines that have recently been used for constructing dynamical system models [17, 18].

This work was supported by JSPS KAKENHI Grant Number JP18K11483.

References


