Iterative Construction of Complete Lyapunov Functions

Carlos Argáez[1], Peter Giesl[2] and Sigurdur Hafstein[1]
[1]*Science Institute, University of Iceland, VR-III, 107 Reykjavík, Iceland*
[2]*Department of Mathematics, University of Sussex, U.K.*

Keywords: Dynamical System, Complete Lyapunov Function, Orbital Derivative, Meshless Collocation, Radial Basis Functions.

Abstract: Dynamical systems describe the evolution of quantities governed by differential equations. Hence, they represent a very powerful prediction tool in many disciplines such as physics and engineering, chemistry and biology and even in economics, among others. Their importance relies on their capability of predicting, as a function of time, future states of the corresponding system under consideration by means of the current, known state. Many difficulties arise when trying to solve such systems. Complete Lyapunov functions allow for the systematic study of complicated dynamical systems. In this paper, we present a new iterative algorithm that avoids obtaining trivial solutions when constructing complete Lyapunov functions. This algorithm is based on mesh-free numerical approximation and analyzes the failure of convergence in certain areas to determine the chain-recurrent set.

1 INTRODUCTION

In this paper, we describe a new algorithm to analyze the behaviour of a dynamical system by means of complete Lyapunov functions. Let us start by considering a general autonomous ordinary differential equation (ODE) of the form,

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}), \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^n$.

A classical (strict) Lyapunov function $V(\mathbf{x})$, is a scalar-valued C^1 function defined on a subset of $\mathbb{R}^n$ which can be used to analyze the basin of attraction of one attractor, e.g. an equilibrium or a periodic orbit (Lyapunov, 1992). It attains its minimum at the attractor, and is otherwise strictly decreasing along solutions of the ODE; it is defined on a neighbourhood of the attractor for which it was constructed.

To generalize this idea, we introduce the concept of a complete Lyapunov function (Auslander, 1964; Conley, 1978a; Conley, 1988; Hurley, 1995; Hurley, 1998), which characterizes the complete behaviour of the dynamical system and not only around one attractor as the classical Lyapunov functions. In particular, it is capable of capturing and describing the long-term behaviour of the system by dividing the phase space into the chain-recurrent set and the gradient-like flow. A point is in the chain-recurrent set, if an ε-trajectory through it comes back to it after any given time. An ε-trajectory is arbitrarily close to a true solution of the system. This indicates recurrent motion; for a precise definition see, e.g. (Conley, 1978a). The dynamics outside the chain-recurrent set are similar to a gradient system, i.e. a system (1) where the right-hand side $\mathbf{f}(\mathbf{x}) = \nabla W(\mathbf{x})$ is given by the gradient of a function $W : \mathbb{R}^n \to \mathbb{R}$.

A complete Lyapunov function is a scalar-valued continuous function $V : \mathbb{R}^n \to \mathbb{R}$, defined on the whole phase space of the ODE. It is non-increasing along solutions of the ODE; it is strictly decreasing where the flow is gradient-like and constant along solution trajectories on each transitive component of the chain-recurrent set, such as local attractors and repellers. Furthermore, the level sets of V provide information about the stability and attraction properties: minima of V correspond to attractors, while maxima represent repellers.

Our method computes a C^1 function V and for such a function the condition *non-increasing* translates into the condition $V'(\mathbf{x}) \leq 0$, where $V'(\mathbf{x}) = \nabla V(\mathbf{x}) \cdot \mathbf{f}(\mathbf{x})$ denotes the orbital derivative, the derivative along solutions of (1). For such a function we know that any point $\mathbf{x}$ fulfilling $V'(\mathbf{x}) < 0$ must be in the set where the flow is gradient-like. By abuse of terminology we refer to any C^1 function fulfilling $V'(\mathbf{x}) \leq 0$ as a complete Lyapunov function.

Argáez, C., Giesl, P. and Hafstein, S.
Iterative Construction of Complete Lyapunov Functions.
DOI: 10.5220/0006835402110222
In *Proceedings of 8th International Conference on Simulation and Modeling Methodologies, Technologies and Applications (SIMULTECH 2018)*, pages 211-222
ISBN: 978-989-758-323-0

Lyapunov theory is not the only available tool to analyze the general behaviour of dynamical systems. Other methodologies like the direct simulation of solutions with many different initial conditions are also useful. However, this is a highly costly computational method that provides only limited information about the very general behaviour of the system, unless estimates are available, e.g. when shadowing solutions.

Other, more sophisticated methods like the computation of invariant manifolds, forming the boundaries of basins of attraction of attractors (Krauskopf et al., 2005) are still unpractical because they require additional analysis of the parts with gradient-like flow.

Other methods divide the phase space into cells to compute the dynamics between them, see for example (Osipenko, 2007). Such methods are known as cell mapping approach (Hsu, 1987) or set oriented methods (Dellnitz and Junge, 2002) and they are also capable of obtaining complete Lyapunov functions.

We will consider a general autonomous ODE of the form (1). The first proof of the existence of a complete Lyapunov function for dynamical systems was given in (Auslander, 1964; Conley, 1978b; Conley, 1978a). These proofs holds for a compact metric space and considers the corresponding attractor-repeller pairs while constructing a function which is 1 on the repeller, 0 on the attractor and decreasing in between. Then these functions are summed up over all attractor-repeller pairs. Later, Hurley generalized these ideas to more general spaces (Hurley, 1992; Hurley, 1995; Hurley, 1998).

Several other computational approaches to construct complete Lyapunov functions have been published, for example (Kalies et al., 2005; Ban and Kalies, 2006; Goullet et al., 2015), where the phase space was subdivided into cells and a discrete-time system was given by the time-T map. Then, a multivalued map was introduced using the computer package GAIO (Dellnitz et al., 2001) to compute the dynamics between them. Using graph algorithms (Ban and Kalies, 2006), an approximate complete Lyapunov function was computed. This approach, however, is not efficient because it requires a high number of cells even for low dimensions.

Our methodology, which is faster and works well in higher dimensions, is inspired by the construction of classical Lyapunov functions. In (Björnsson et al., 2014a), the method of (Ban and Kalies, 2006) is compared to mesh-free collocation (see below) for one particular example. While the mesh-free collocation method is very efficient on the gradient-like part, in that particular case, the approach of (Ban and Kalies, 2006) works properly only on the chain-recurrent set but requires a much finer grid.

Another method construct a complete Lyapunov as a continuous piecewise affine (CPA) function, affine on a fixed simplicial complex, see (Björnsson et al., 2015). However, they require a superset of the chain-recurrent set as an input. In this paper, the proposed method does not require any information about the system under consideration except for the right-hand side function $\mathbf{f}$ of system (1).

Let us give an overview of this paper: in Section 2 we discuss mesh-free collocation to approximate solutions of a general linear PDE as well as previous, related algorithms. In Section 3 we present our algorithm to compute a complete Lyapunov function, apply the method to an example, and discuss the results in detail before we conclude.

2 PRELIMINARIES

2.1 Mesh-free Collocation

Numerous numerical construction methods have been proposed to obtain classical Lyapunov functions, e.g. (Johansson, 1999; Johansen, 2000; Marinósson, 2002; Giesl, 2007; Hafstein, 2007; Björnsson et al., 2014b; Kamyar and Peet, 2015; Anderson and Papachristodoulou, 2015; Doban, 2016; Doban and Lazar, 2016), see also the review (Giesl and Hafstein, 2015). In this paper, like previously in (Argáez et al., 2017a; Argáez et al., 2018b), our algorithm will be based on the Radial Basis Function (RBF) method with mesh-free collocation. Under this approach, the solution of a linear PDE is approximated and the orbital derivative is specified to approximate the solution of a linear PDE.

Mesh-free collocation, in particular using Radial Basis Functions, solves generalized interpolation problems efficiently. In our case, we interpolate the data given by a partial differential operator applied to the function at given collocation points. Examples for Radial Basis Functions include the Gaussians, multiquadrics, inverse multiquadrics and Wendland's compactly supported basis functions. An interpolation example, as well as a C++ code to compute Wendland's basis functions of arbitrary order, can be found in (Argáez et al., 2017b).

We assume that the target function belongs to a Hilbert space H of continuous functions (often a Sobolev space). We assume that the Hilbert space H has a reproducing kernel $\varphi : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$, given by a convenient Radial Basis Function Φ through $\varphi(\mathbf{x},\mathbf{y}) := \Phi(\mathbf{x}-\mathbf{y})$, where $\Phi(\mathbf{x}) = \psi_0(\|\mathbf{x}\|)$ is a radial function.

We seek to reconstruct the target function $V \in H$ from the information $r_1,\dots,r_N \in \mathbb{R}$ generated by N linearly independent functionals $\lambda_j \in H^*$, i.e. $\lambda_j(V) = r_j$ holds for $j = 1,\dots,N$. The optimal (norm-minimal) reconstruction of the function V is the solution of the problem

$$\min\{\|v\|_H : \lambda_j(v) = r_j, 1 \le j \le N\}.$$

It is well-known (Wendland, 2005) that the solution can be written as

$$v(\mathbf{x}) = \sum_{j=1}^{N} \beta_j \lambda_j^{\mathbf{y}} \varphi(\mathbf{x},\mathbf{y}),$$

where the coefficients β_j are determined by the interpolation conditions $\lambda_j(v) = r_j$, $1 \le j \le N$.

In our case, we consider the PDE $V'(\mathbf{x}) = r(\mathbf{x})$, where $r(\mathbf{x})$ is a given function and $V'(\mathbf{x})$ is the orbital derivative. We choose N collocation points $\mathbf{x}_1,\dots,\mathbf{x}_N \in \mathbb{R}^n$ of the phase space and define functionals $\lambda_j(v) := (\delta_{\mathbf{x}_j} \circ L)^{\mathbf{x}} v = v'(\mathbf{x}_j) = \nabla v(\mathbf{x}_j) \cdot \mathbf{f}(\mathbf{x}_j)$, where L denotes the linear operator of the orbital derivative $LV(\mathbf{x}) = V'(\mathbf{x})$ and δ is Dirac's delta distribution. The information is given by the right-hand side $r_j = r(\mathbf{x}_j)$ for all $1 \le j \le N$. The approximation is then

$$v(\mathbf{x}) = \sum_{j=1}^{N} \beta_j (\delta_{\mathbf{x}_j} \circ L)^{\mathbf{y}} \Phi(\mathbf{x}-\mathbf{y}),$$

where Φ is a positive definite Radial Basis Function, and the coefficients $\beta_j \in \mathbb{R}$ can be calculated by solving a system of N linear equations. A crucial ingredient is the knowledge on the behaviour of the error function $|V'(\mathbf{x}) - v'(\mathbf{x})|$ in terms of the so-called fill distance which measures how dense the points $\{\mathbf{x}_1,\dots,\mathbf{x}_N\}$ are, since it gives information when the approximate solution indeed becomes a Lyapunov function, i.e. has a negative orbital derivative. Such error estimates were derived, for example in (Giesl, 2007; Giesl and Wendland, 2007), see also (Narcowich et al., 2005; Wendland, 2005).

The advantage of mesh-free collocation over other methods for solving PDEs is that scattered points can be added to improve the approximation, no triangulation of the phase space is necessary, the approximating function is smooth and the method works in any dimension.

In this paper, we use Wendland functions (Wendland, 1998) as Radial Basis Functions through $\psi_0(r) := \psi_{l,k}(cr)$, where $c > 0$, $k \in \mathbb{N}$ is a smoothness parameter and $l = \lfloor \frac{n}{2} \rfloor + k + 1$. Wendland functions are positive definite functions with compact support, which are polynomials on their support; the corresponding reproducing kernel Hilbert space is norm-equivalent to the Sobolev space $W_2^{k+(n+1)/2}(\mathbb{R}^n)$. They are defined by recursion: for $l \in \mathbb{N}$, $k \in \mathbb{N}_0$ we define

$$\begin{aligned} \psi_{l,0}(r) &= (1-r)_+^l \\ \psi_{l,k+1}(r) &= \textstyle\int_r^1 t\psi_{l,k}(t)\mathrm{d}t \end{aligned} \tag{2}$$

for $r \in \mathbb{R}_0^+$, where $x_+ = x$ for $x \ge 0$ and $x_+ = 0$ for $x < 0$.

As collocation points $X = \{\mathbf{x}_1,\dots,\mathbf{x}_N\} \subset \mathbb{R}^n$ we use a subset of the hexagonal grid with $\alpha_{\text{Hexa-basis}} \in \mathbb{R}^+$ constructed according to

$$\left\{ \alpha_{\text{Hexa-basis}} \sum_{k=1}^{n} i_k w_k : i_k \in \mathbb{Z} \right\}, \text{ where}$$

$$\begin{aligned} w_1 &= (2e_1, 0, 0, \dots, 0) \\ w_2 &= (e_1, 3e_2, 0, \dots, 0) \\ &\vdots \quad\quad \vdots \\ w_n &= (e_1, e_2, e_3, \dots, (n+1)e_n) \text{ and} \\ e_k &= \sqrt{\frac{1}{2k(k+1)}}, \quad k \in \mathbb{N}. \end{aligned}$$

Note that the hexagonal grid is optimal to balance the opposing aims of a dense grid in order to achieve a small error and a large separation distance of points so that the collocation matrix has a small condition number (Iske, 1998).

We set $\psi_0(r) := \psi_{l,k}(cr)$ with positive constant c and define recursively $\psi_i(r) = \frac{1}{r}\frac{d\psi_{i-1}}{dr}(r)$ for $i = 1,2$ and $r > 0$. Note that $\lim_{r\to 0} \psi_i(r)$ exists if the smoothness parameter k of the Wendland function is sufficiently large. The explicit formulas for v and its orbital derivative are

$$\begin{aligned} v(\mathbf{x}) &= \sum_{j=1}^{N} \beta_j \langle \mathbf{x}_j - \mathbf{x}, \mathbf{f}(\mathbf{x}_j)\rangle \psi_1(\|\mathbf{x}-\mathbf{x}_j\|), \\ v'(\mathbf{x}) &= \sum_{j=1}^{N} \beta_j \Big[-\psi_1(\|\mathbf{x}-\mathbf{x}_j\|)\langle \mathbf{f}(\mathbf{x}), \mathbf{f}(\mathbf{x}_j)\rangle \\ &\qquad + \psi_2(\|\mathbf{x}-\mathbf{x}_j\|)\langle \mathbf{x}-\mathbf{x}_j, \mathbf{f}(\mathbf{x})\rangle \cdot \langle \mathbf{x}_j - \mathbf{x}, \mathbf{f}(\mathbf{x}_j)\rangle \Big] \end{aligned}$$

where $\langle\cdot,\cdot\rangle$ denotes the standard scalar product and $\|\cdot\|$ the Euclidean norm in $\mathbb{R}^n$, β is the solution of $A\beta = \mathbf{r}$, $r_j = r(\mathbf{x}_j)$ and A is the $N \times N$ matrix with entries

$$\begin{aligned} a_{ij} = {} & \psi_2(\|\mathbf{x}_i - \mathbf{x}_j\|)\langle \mathbf{x}_i - \mathbf{x}_j, \mathbf{f}(\mathbf{x}_i)\rangle \langle \mathbf{x}_j - \mathbf{x}_i, \mathbf{f}(\mathbf{x}_j)\rangle \\ & - \psi_1(\|\mathbf{x}_i - \mathbf{x}_j\|)\langle \mathbf{f}(\mathbf{x}_i), \mathbf{f}(\mathbf{x}_j)\rangle \end{aligned}$$

for $i \ne j$ and

$$a_{ii} = -\psi_1(0)\|\mathbf{f}(\mathbf{x}_i)\|^2.$$

More detailed explanations on this construction are given in (Giesl, 2007, Chapter 3).

If no collocation point $\mathbf{x}_j$ is an equilibrium for the system, i.e. $\mathbf{f}(\mathbf{x}_j) \neq \mathbf{0}$ for all j, then the matrix A is positive definite and the system of equations $A\beta = \mathbf{r}$ has a unique solution. Note that this holds true independent of whether the underlying discretized PDE has a solution or not, while the error estimates are only available if the PDE has a solution.

2.2 Previous Algorithms

To obtain a classical Lyapunov function used to be a very hard task in engineering and other disciplines. However, mathematical research has given very efficient algorithms to compute Lyapunov functions (Giesl and Hafstein, 2015). One of these algorithms to compute classical Lyapunov functions for an equilibrium approximates the solution of the PDE $V'(\mathbf{x}) = \nabla V(\mathbf{x}) \cdot \mathbf{f}(\mathbf{x}) = -1$ (Giesl, 2007) using mesh-free collocation with Radial Basis Function. It constructs an approximate solution to this linear partial differential equation (PDE), which satisfies the equation in a given, finite set of collocation points X.

This method has been extended to the construction of complete Lyapunov function. However, a complete Lyapunov function cannot have a negative derivative on the chain-recurrent set, hence, the equation does not have a solution. However, turning the argument around, the area where the approximation is poor, i.e. where the approximation v to the Lyapunov function V does not satisfy $v'(\mathbf{x}) \approx -1$, gives an indication of where the chain-recurrent set is located. In previous work, the authors of this paper have developed and continuously improved such algorithms. Firstly, the algorithm was implemented to identify both the chain-recurrent sets and the gradient-like flow region, see (Argáez et al., 2017a). We determine the chain-recurrent set as the area, in which the condition $v'(\mathbf{x}) \approx -1$ is not satisfied or the approximation fails. In the next step, we then split the collocation points X into two different sets: X^0 where the approximation fails, and X^-, where it works well. Given that a complete Lyapunov function should be constant in X^0, we reconstructed the Lyapunov function with two conditions for the orbital derivative: $V'(\mathbf{x}) = 0$ for all $\mathbf{x} \in X^0$ and $V'(\mathbf{x}) = -1$ for all $\mathbf{x} \in X^-$, which is an approximation of a complete Lyapunov function. These two sets provide important information about the ODE under consideration: the set X^0, where $v'(\mathbf{x}) \approx 0$ approximates the chain-recurrent set, including equilibria, periodic and homoclinic orbits, while the set X^-, where $v'(\mathbf{x}) \approx -1$, approximates the area where the flow is gradient-like, i.e. where solutions pass through. This approach was further iterated until the sets X^0 and X^- do not change.

However, such methodology was not capable to isolate chain-recurrent sets completely in dynamical systems with considerable differences in speed. Hence, the method was enhanced (Argáez et al., 2018b) by considering an "almost" normalized speed of the dynamical system. This means that the original dynamical system (1) was substituted by,

$$\begin{aligned} \dot{\mathbf{x}} &= \hat{\mathbf{f}}(\mathbf{x}), \\ \text{where } \hat{\mathbf{f}}(\mathbf{x}) &= \frac{\mathbf{f}(\mathbf{x})}{\sqrt{\delta^2 + \|\mathbf{f}(\mathbf{x})\|^2}} \end{aligned} \tag{3}$$

with parameter $\delta > 0$.

Furthermore, the right-hand side of $V'(\mathbf{x}) = \begin{cases} 0 & \text{if } \mathbf{x} \in X^0, \\ -1 & \text{if } \mathbf{x} \notin X^0 \end{cases}$ has a jump. To obtain a smooth right-hand side, we replaced this by

$$V'(\mathbf{x}) = r(\mathbf{x}) := \begin{cases} 0 & \text{if } \mathbf{x} \in X^0, \\ -\exp\left(-\frac{1}{\xi \cdot \mathfrak{d}^2(\mathbf{x})}\right) & \text{if } \mathbf{x} \notin X^0, \end{cases} \tag{4}$$

where $\mathfrak{d}$ denotes the distance to the set X^0 and $\xi > 0$ is a parameter. This improved the method to construct complete Lyapunov functions and to describe the chain-recurrent sets.

However, when iterating using this method, the area where $v'(\mathbf{x}) \approx 0$ holds (or X^0) could become larger and larger, and the functions v could converge towards a trivial, constant solution. Indeed, we will later show that this behaviour is observed.

In this paper we will propose a modified method that prevents the iterations from converging to the trivial solution.

Firstly, instead of using just 0 or -1 for the right-hand side, or 0 and the exponential function in (4), we use the average of $v'(\mathbf{y})$ for points $\mathbf{y}$ around each point $\mathbf{x}_j \in X$, where v denotes the result of the last iteration and regardless of $\mathbf{x}_j$ lying in X^- or X^0. Secondly, we then scale the right-hand side, such that the sum $\sum_{i=1}^{N} r(\mathbf{x}_i)$ of the right-hand side over all collocation points is constant in each iteration. This prevents the iterations from converging to the trivial solution. We will show the improvement of the performance in an example.

For the evaluation points $\mathbf{y}$ around each collocation point $\mathbf{x}_j$ we use a directional evaluation grid, which was first proposed for the higher-dimensional case, i.e., $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$ with $\mathbf{x} \in \mathbb{R}^n$ and $n \geq 3$ (Argáez et al., 2018a), since the set of evaluation points remains 1-dimensional. For the directional evaluation grid, we use points in the direction of the flow $\mathbf{f}(\mathbf{x}_j)$ at each collocation point $\mathbf{x}_j$. Previously, in (Argáez et al., 2017a; Argáez et al., 2018b) and dimension 2,

we used a circular evaluation grid around each collocation point consisting of two concentric circumferences whose centre was the collocation point. In the n-dimensional case, this circumference would become an $(n-1)$-dimensional set, requiring a higher amount of points to evaluate.

3 ALGORITHM

Our algorithm is based on the algorithms described in (Argáez et al., 2017a) and where the speed of the system has been normalized as in (3) and explained in detail in (Argáez et al., 2018b). The Wendland functions are constructed with the software from (Argáez et al., 2017b). In this paper, we add the next improvement: we scale the right-hand side of the linear system $A\beta = \mathbf{r}$ in each iteration to avoid convergence to the trivial solution $\beta = 0$ corresponding to $V(\mathbf{x}) = 0$. Let us explain the method and the improvements in more detail.

We fix a set of collocation points X, none of which is an equilibrium for the system. We use an evaluation grid to determine for each collocation point whether the approximation was poor or good, and thus whether the collocation point is part of the chain-recurrent set (X^0) or the gradient-like flow (X^-). The evaluation grid at the collocation point $\mathbf{x}_j$ is given by

$$Y_{\mathbf{x}_j} = \left\{ \mathbf{x}_j \pm \frac{r \cdot k \cdot \alpha_{\text{Hexa-basis}} \cdot \mathbf{f}(\mathbf{x}_j)}{m \cdot \|\mathbf{f}(\mathbf{x}_j)\|} : k \in \{1, \ldots, m\} \right\} \tag{5}$$

where $\alpha_{\text{Hexa-basis}}$ is the parameter used to build the hexagonal grid defined above, $r \in (0,1)$ is the ratio up to which the evaluation points will be placed and $m \in \mathbb{N}$ denotes the number of points in the evaluation grid that will be placed on both sides of the collocation points aligned to the flow. Altogether, will have $2m$ points for each collocation point, so $2mN$ points in the evaluation grid overall. We notice that we have chosen $r < 1$ to avoid overlap of evaluation grids.

We start by computing the approximate solution v_0 of $V'(\mathbf{x}) = -1$ with collocation points X. As we have previously done in (Argáez et al., 2017a; Argáez et al., 2018b), we define a tolerance parameter $-1 < \gamma \leq 0$. In each step i of the iteration we mark a collocation point $\mathbf{x}_j$ as being in the chain-recurrent set ($\mathbf{x}_j \in X^0$) if there is at least one point $\mathbf{y} \in Y_{\mathbf{x}_j}$ such that $v_i'(\mathbf{y}) > \gamma$. The points for which the condition $v_i'(\mathbf{y}) \leq \gamma$ holds for all $\mathbf{y} \in Y_{\mathbf{x}_j}$ are considered to belong to the gradient-like flow ($\mathbf{x}_j \in X^-$).

In this paper, we now improve the algorithm by replacing the right-hand side -1 by the average of the values $v_i'(\mathbf{y})$ over the evaluation grid $\mathbf{y} \in Y_{\mathbf{x}_j}$ at each collocation point $\mathbf{x}_j$ or, if this average is positive, by 0. In formulas, we calculate the approximate solution v_{i+1} of $V'(\mathbf{x}_j) = \tilde{r}_j$ with

$$\tilde{r}_j = \left(\frac{1}{2m} \sum_{\mathbf{y} \in Y_{\mathbf{x}_j}} v_i'(\mathbf{y}) \right)_-,$$

where $x_- = x$ if $x \leq 0$ and $x_- = 0$ otherwise. We will refer to this as the "non-scaled" version.

While in (Argáez et al., 2018b) we have used the distance to the set X^0 to ensure that the right-hand side is a continuous function, this new approach also allows us to obtain a complete Lyapunov function with continuous orbital derivative. However, this approach can lead to a decrease of energy from the original Lyapunov function. Let the reader remember that the original value of the orbital derivative condition at the first iteration is -1, but the new value is obtained by averaging and bounding by 0, so it could tend to zero and thus force the total energy of the Lyapunov function to decrease. To avoid this, we scale the condition of the orbital derivative after the first iteration onwards so that the sum of all r_j over all collocation points is constant for all iterations; we will refer to this as the "scaled" method.

Our new algorithm to compute complete Lyapunov functions and classify the chain-recurrent set can be summarized as follows.

1. Create the set of collocation points X and compute the approximate solution v_0 of $V'(\mathbf{x}) = -1$; set $i = 0$
2. For each collocation point $\mathbf{x}_j$, compute $v_i'(\mathbf{y})$ for all $\mathbf{y} \in Y_{\mathbf{x}_j}$: if $v_i'(\mathbf{y}) > \gamma$ for a point $\mathbf{y} \in Y_{\mathbf{x}_j}$, then $\mathbf{x}_j \in X^0$, otherwise $\mathbf{x}_j \in X^-$, where $\gamma \leq 0$ is a chosen critical value
3. Define $\tilde{r}_j = \left(\frac{1}{2m} \sum_{\mathbf{y} \in Y_{\mathbf{x}_j}} v_i'(\mathbf{y}) \right)_-$
4. Define $r_j = \frac{N}{\sum_{l=1}^{N} |\tilde{r}_l|} \tilde{r}_j$,
5. Compute the approximate solution v_{i+1} of $V'(\mathbf{x}_j) = r_j$ for $j = 1, \ldots, N$; this is the scaled version, while approximating the solution of $V'(\mathbf{x}_j) = \tilde{r}_j$ would be the non-scaled version
6. Set $i \to i+1$ and repeat steps 2. to 5. until no more points are added to X^0

Note that the sets X^0 and X^- change in each step of the algorithm.

3.1 Results

While the method is applicable in any dimension n, we focus here on an example in two dimensions, because the results can be easily visualized.

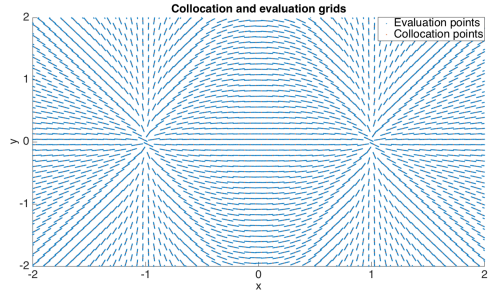

Figure 1: Evaluation grid for (6). All points of the evaluation grid are aligned with the direction of the flow of the dynamical system.

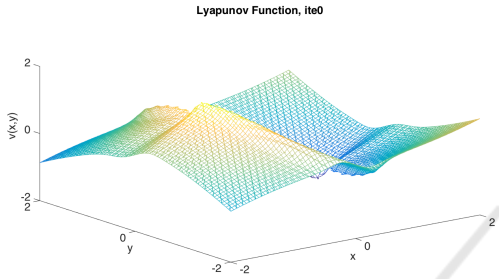

Figure 2: Complete Lyapunov function v_0 for system (6), obtained by solving $V'(\mathbf{x}) = -1$, iteration 0.

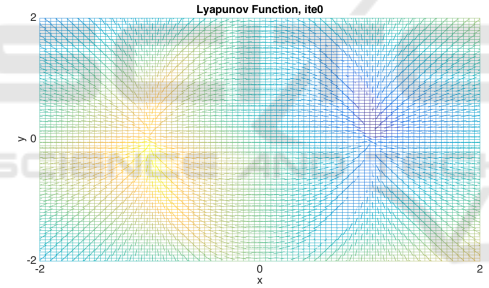

Figure 3: Complete Lyapunov function v_0 for system (6) as seen from above. It shows the general behaviour of a system "falling" down from the maximum value into the minimal one.

The results we present here are computed using the normalization of the speed of the system (3), a method we first introduced in (Argáez et al., 2018b). The evaluation grid used is built according to (5) in the algorithm in Sec. 3. The system we use to benchmark our methodology is the following:

$$\mathbf{f}(x,y) = \begin{pmatrix} 1-x^2 \\ -xy \end{pmatrix}, \tag{6}$$

This system has two equilibria $(\pm 1, 0)$ where $(-1,0)$ is unstable and $(1,0)$ is asymptotically stable.

We now apply the method proposed in this paper. The parameters we defined to compute the complete Lyapunov function for the system (6) are: we replace $\mathbf{f}$ by $\hat{\mathbf{f}}$ according to (3) with $\delta^2 = 10^{-8}$, and we choose

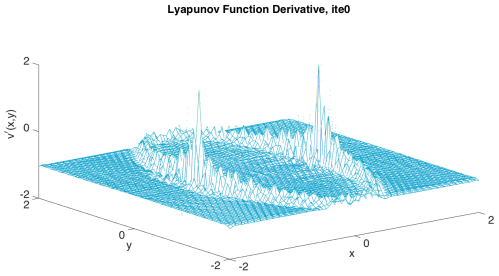

Figure 4: Complete Lyapunov function derivative v_0' for system (6), iteration 0.

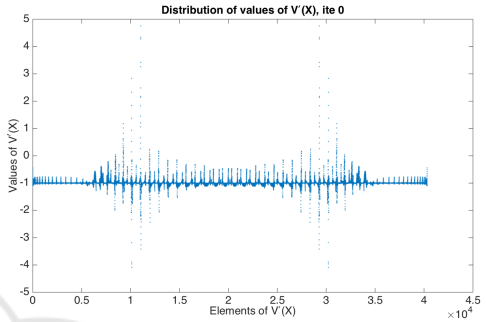

Figure 5: Distribution of v_0' for system (6). The x-axis represents all points of the evaluation grid.

$\alpha_{\text{Hexa-basis}} = 0.1$ and used all points in the grid that are in the area $[-2,2]^2 \in \mathbb{R}^2$. This gave us a total amount of 2016 collocation points. The critical value to define the failing points is $\gamma = -0.5$ and the Wendland function parameters are $l = 4$, $k = 2$ and $c = 1$. The number of evaluation points around each collocation point is 20, i.e., $m = 10$, and we choose $r = 0.5$; the total amount of points in our evaluation grid is $40,320$.

Figure 1 shows the evaluation grid for system (6). The evaluation grid is placed along the direction of flow of the dynamical system, see (5). Therefore, it already gives an idea of how the system is behaving.

We start by approximating the solution of $V'(\mathbf{x}) = -1$ by v_0. The function v_0 and its orbital derivative v_0' are shown in Figures 2 and 4, respectively. Figure 5 shows the orbital derivative v_0' as well, where the x-axis represents all points of the evaluation grid. Figure 2 already shows that the unstable equilibrium in $(-1,0)$ is a maximum of v_0 and the asymptotically stable equilibrium in $(1,0)$ is a minimum; the orbital derivative v_0' approximates -1 very well apart from the equilibria and an ellipse covering the heteroclinic orbits, connecting the two equilibria, which touch the boundary of the considered area. Figure 17 shows the values over the evaluation grid that failed in the sense that $v_0'(\mathbf{y}) > \gamma$. These points are an approximation of the chain-recurrent set under the scaled method.

However, a different perspective of Figure 2, see

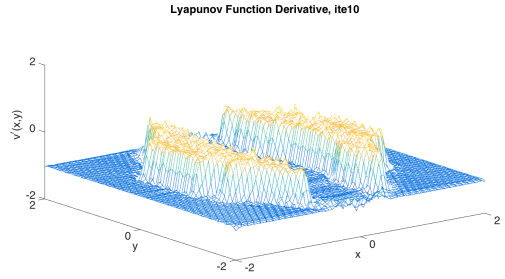

Figure 6: Complete Lyapunov function derivative for system (6) with the previous method after 10 iterations. These iterations were obtained with the method in (Argáez et al., 2017a).

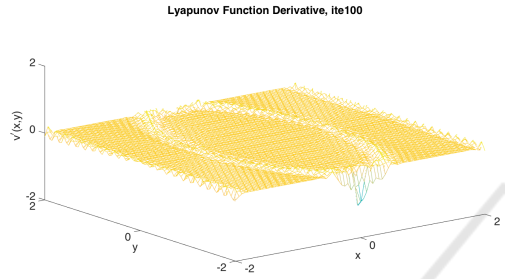

Figure 7: Complete Lyapunov function derivative for system (6) with the previous method after 100 iterations. These iterations were obtained with the previous method from (Argáez et al., 2017a). The derivative converges to 0 as the collocation points are nearly all in the set X^0.

Figure 3, shows that there are many heteroclinic connections between the two equilibria. These connections have different lengths. In particular, even if v is constant in a neighbourhood of each of the two equilibria, there is no solution such that $v'(\mathbf{x}) = -1$ holds in the rest of the gradient-like flow part since the orbital derivative reflects the length of the route. Hence, this is an example where the previous method (Argáez et al., 2017a) must fail.

Indeed, if we now follow the previous method and iteratively split the collocation points into the sets X^0 where the approximation is poor and X^- where the approximation is good and solve $V'(\mathbf{x}_j) = -1$ for $\mathbf{x}_j \in X^-$ and $V'(\mathbf{x}_j) = 0$ for $\mathbf{x}_j$, then we approximate the trivial, constant solution, see Figures 6 and 7 after 10 and 100 iterations.

3.2 Non-scaled Method

Now we apply the new method as described in Section 3 by iteratively solving $V'(\mathbf{x}_j) = \tilde{r}_j$, i.e. the non-scaled method, where $\tilde{r}_j$ is the average of the values of v' around each collocation point. If the average becomes positive then it is substituted by zero. Figures 8 (iteration 0) and Figure 9 (iterations 100 and 10000) show the values of $\tilde{r}_j$. After 100 iterations clearly fewer points are failing than with the previous method. However, after 10000 iterations, the values $\tilde{r}_j$ have slowly converged to zero. Correspondingly, also the approximation v of $V'(\mathbf{x}_j) = \tilde{r}_j$ satisfies $v'(\mathbf{x}) \approx 0$ for all $\mathbf{x}$, see Figure 11, and the function $v(\mathbf{x})$ is nearly constant, see Figure 13.

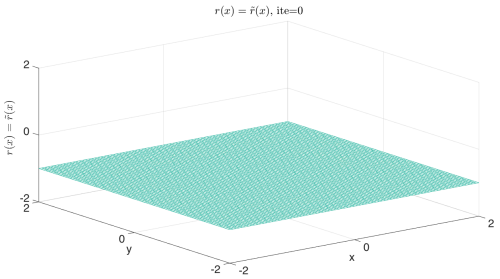

Figure 8: Plot of the right-hand side $r_j = \tilde{r}_j = -1$ at each collocation point for iteration 0. For iteration 0, there is no difference between the non-scaled and scaled methods.

This behaviour is also shown in Figure 20: in the top figure $\sum_{j=1}^{N} \tilde{r}_j$ (blue) is shown for each iteration, which quickly converges to 0, implying that for each j the quantity $\tilde{r}_j$ converges to zero as i increases (note that $\tilde{r}_j \leq 0$ by definition). In the bottom figure, the sum over all evaluation points $\sum_{j=1}^{N} \sum_{\mathbf{y} \in Y_{\mathbf{x}_j}} v_i'(\mathbf{y})$ (blue) is shown, which also quickly converges to 0, showing that $v_i'(\mathbf{x}) \approx 0$ for all $\mathbf{x}$. Finally, Figure 21 shows the total amount of failing points (blue) in both the evaluation grid and the collocation grid which quickly converges to a number very close to the total amount of evaluation and collocation points, respectively.

3.3 Scaled Method: Avoiding Convergence to the Trivial Solution

To overcome the convergence to a trivial solution, we now consider the scaled version, where we approximate $V'(\mathbf{x}_j) = r_j$ and r_j denotes the scaled $\tilde{r}_j$ such that the sum $\sum_{j=1}^{N} r_j = -N$ remains constant in each iterative step.

Figures 15 and 16 show v' after 10 iterations. Even after many iterations, the values r_j do not converge to zero, see Figures 8 (iteration 0) and Figure 10 (iterations 100 and 10000). Correspondingly, also the approximation v of $V'(\mathbf{x}_j) = r_j$ behaves in a similar way, see Figure 12, and the function $v(\mathbf{x})$ is not constant, but gives a clear indication of the dynamics, see Figure 14.

This behaviour is also shown in Figure 20: in the top figure $\sum_{j=1}^{N} r_j$ (red) is shown, which is constantly $-N$ by construction. In the bottom figure, the

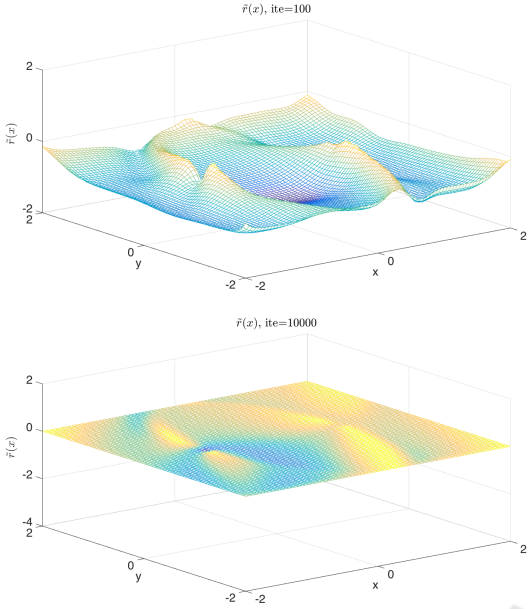

Figure 9: The right-hand side $\tilde{r}_j$ at each collocation point, iterations 100 and 10000 (non-scaled method). After 100 iterations, $\tilde{r}_j$ is not zero everywhere, but at iteration 10000 $\tilde{r}_j$ is nearly zero everywhere, providing a trivial result.

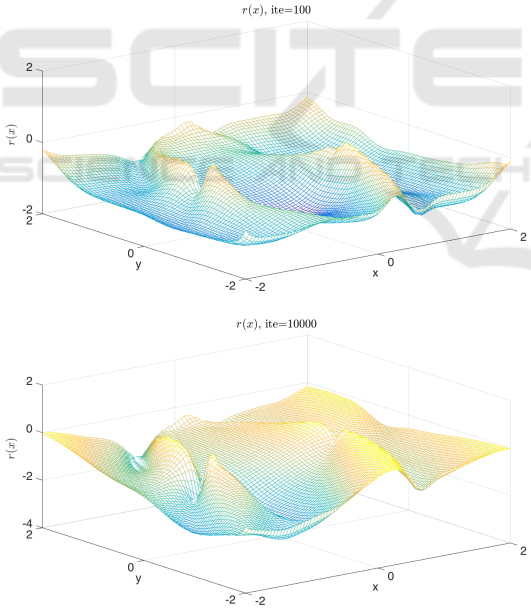

Figure 10: The right-hand side r_j at each collocation point for iterations 100 and 10000 (scaled method). Even after 10000 iterations r_j is not zero everywhere and thus provides a good complete Lyapunov function.

sum over all evaluation points $\sum_{j=1}^{N} \sum_{\mathbf{y} \in Y_{\mathbf{x}_j}} v_i'(\mathbf{y})$ (red) is shown, which after a slight adjustment also stays constant. Finally, Figure 21 shows the total amount of failing evaluation and collocation points (both red)

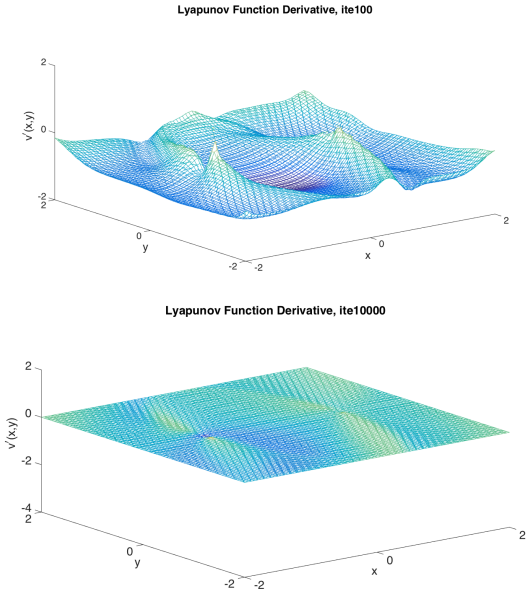

Figure 11: $v'(\mathbf{x})$ for iterations 100 and 10000 (non-scaled version). After 100 iterations, $v'(\mathbf{x})$ is not zero everywhere, but at iteration 10000 $v'(\mathbf{x}) \approx 0$, providing a trivial result.

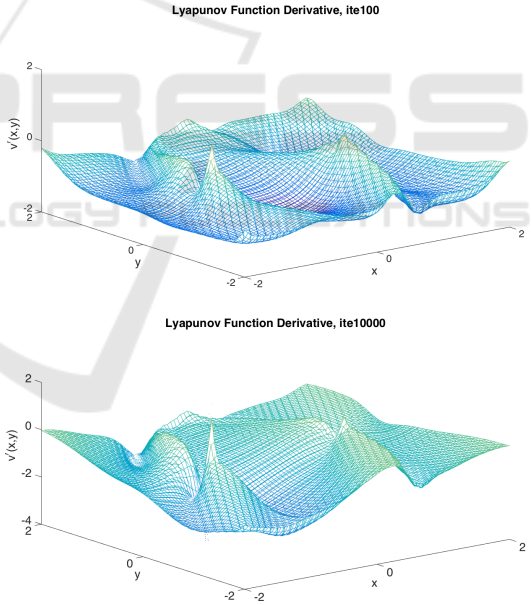

Figure 12: $v'(\mathbf{x})$ for iterations 100 and 10000 (scaled version). Even after 10000 iterations, $v'(\mathbf{x})$ does not converges to zero everywhere and thus produces a non-trivial complete Lyapunov function.

which converges to a number far away from the total amount of evaluation or collocation points, respectively, and thus giving a much more accurate estimate of the chain-recurrent set. In fact, Figure 18 shows the approximation of the chain-recurrent set through the failing points with the scaled method after itera-

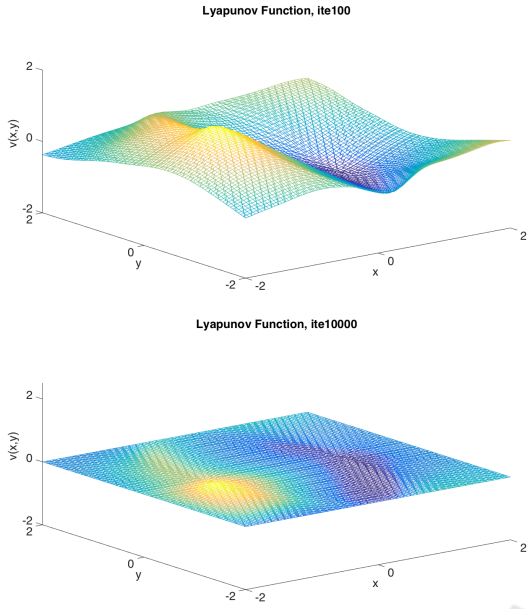

Figure 13: $v(\mathbf{x})$ for iterations 100 and 10000 (non-scaled version). After 100 iterations, $v(\mathbf{x})$ is not constant, but at iteration 10000 $v(\mathbf{x})$ is nearly constant, providing a trivial result.

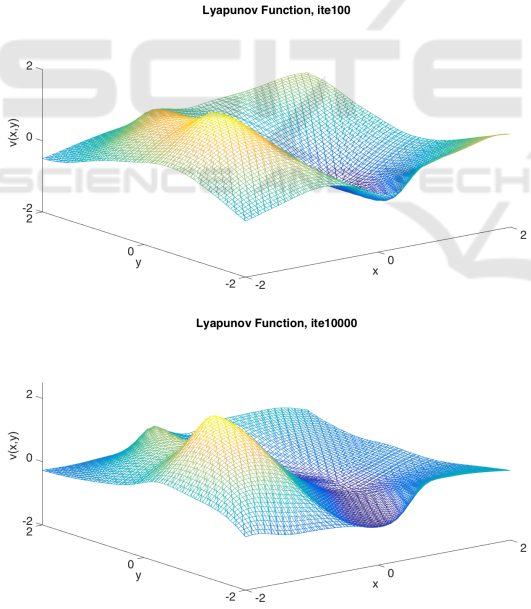

Figure 14: $v(\mathbf{x})$ for iterations 100 and 10000 (scaled version). Even after 10000 iterations, $v(\mathbf{x})$ is not a constant function and thus provides a non-trivial complete Lyapunov function.

tion 50, and Figure 19 shows the same after iteration 10000. Both overestimate the true chain-recurrent set, which consists just of the two equilibria, but the figures show that the iterations do not converge towards the trivial, constant solution.

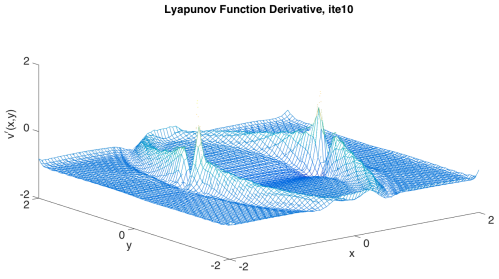

Figure 15: Complete Lyapunov function derivative v'_{10} for system (6), iteration 10 with the scaled method.

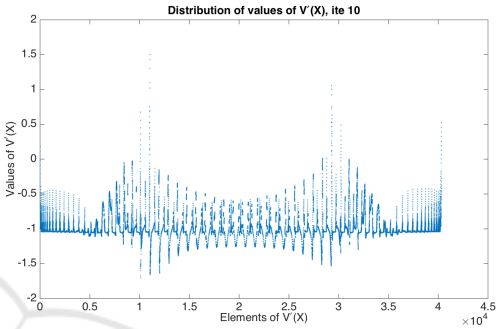

Figure 16: Distribution of v'_{10} for system (6) with the scaled method after 10 iterations. The x-axis represents all points of the evaluation grid.

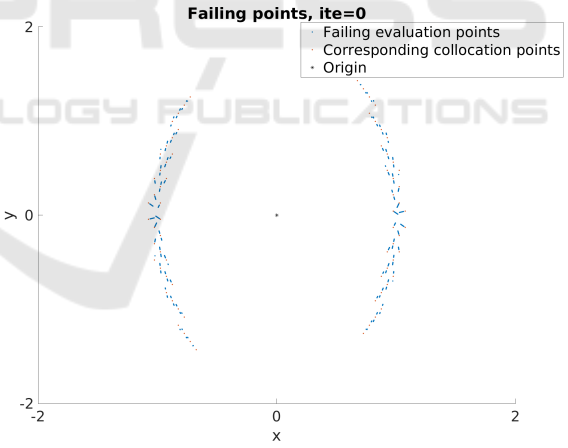

Figure 17: Scaled method. Approximation of chain-recurrent set for (6), $\gamma = -0.5$, iteration 0.

Summarizing, Figure 20 shows the behaviour of the sum of the $\tilde{r}_j$, r_j, respectively for the non-scaled and the scaled cases. In the non-scaled case (blue), the value converges to zero, i.e. we would reach a trivial complete Lyapunov function, which does not provide any information about the system. Scaling, however, avoids this converges and thus we obtain a non-trivial complete Lyapunov function.

Comparing Figures 9 and 10 again clearly shows that the non-scaled version produces a trivial com-

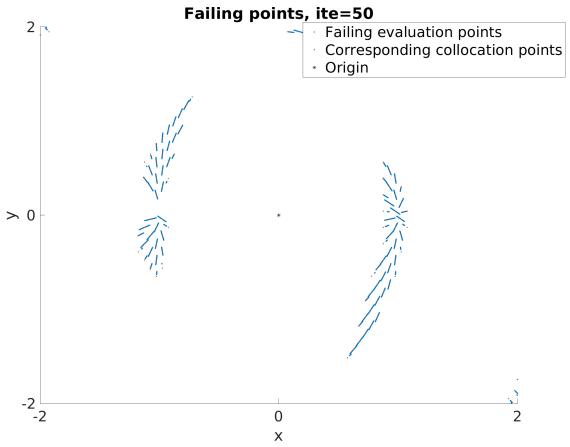

Figure 18: Approximation of chain-recurrent set for (6), $\gamma = -0.5$, iteration 50, scaled version.

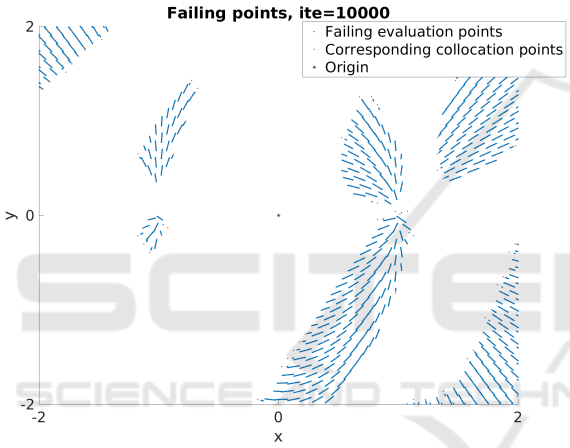

Figure 19: Approximation of chain-recurrent set for (6), $\gamma = -0.5$, iteration 10000, scaled version.

plete Lyapunov function, and the same can be concluded from comparing Figures 11 and 12 for v' as well as Figures 13 and 14 for v.

4 CONCLUSIONS

In this paper we have improved a method to compute complete Lyapunov functions by introducing a new feature to avoid convergence to the trivial solution when iterating. We have shown in an example that our method converges to a non-trivial complete Lyapunov function and gives us a reasonable approximation, whereas the old algorithm converges to the trivial solution. Furthermore, the developed algorithm was shown to be able to avoid convergence to zero. The idea behind it is to scale the orbital derivative condition to keep the addition of its values constant. Finally, this method keeps the level of efficiency as previous methods for its computational effort to solve the Lyapunov equations is $O(N^3)$ while the evaluation of one point is $O(N)$, where N is the number of collocation points. The method works in any dimension. However, when increasing the dimension of the problem, the amount of collocation points will increase. We do, however, control the total amount of points to be evaluated by considering a one-dimensional domain that is aligned to the flow. This avoids an exponential grow of evaluation points.

Future work will include the application to the method to higher-dimensional and dynamically more challenging examples, such as systems with a chaotic attractor or examples from applications.

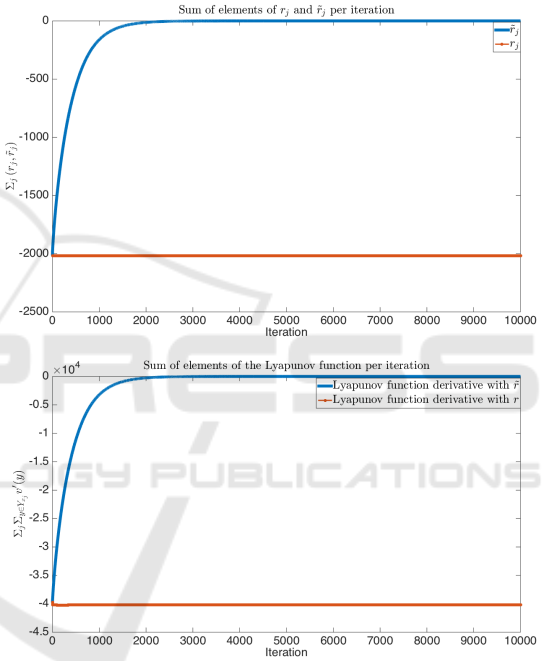

Figure 20: Top: $\sum_{j=1}^{N} \tilde{r}_j$ (blue, non-scaled version) and $\sum_{j=1}^{N} r_j$ (red, scaled version) for each iteration up to 10000. The sum of $\tilde{r}_j$ quickly converges to zero, while the sum of r_j is constant by construction. Bottom: $\sum_{j=1}^{N} \sum_{\mathbf{y} \in Y_{\mathbf{x}_j}} v'(\mathbf{y})$ in blue for the non-scaled version and in red for the scaled version for each iteration up to 10000. The sum of the calculated function v' over all points of the evaluation grid behaves in a similar way as the right-hand side $V'(\mathbf{x}) = \tilde{r}_j, r_j$, respectively, of the equation we approximate. In the non-scaled case, $v'(\mathbf{x})$ quickly converges to zero, while the sum over all evaluation points in the scaled case stays constant.

ACKNOWLEDGEMENTS

The first author in this paper is supported by the Icelandic Research Fund (Rannís) grant number 163074-

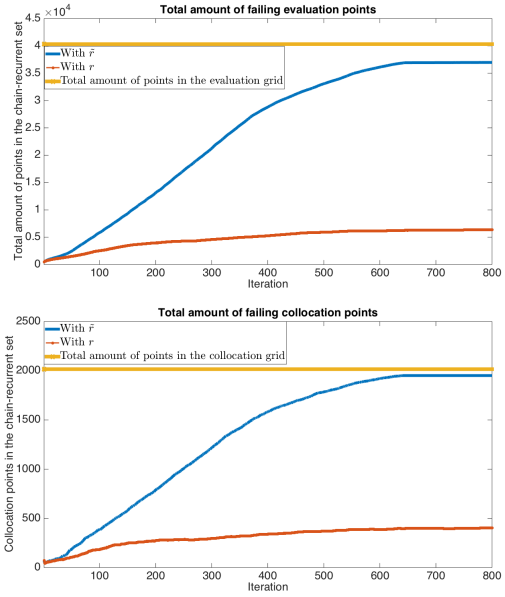

Figure 21: Top: Total amount of failing evaluation points in the non-scaled (blue) and the scaled (red) case compared to the total amount of evaluation points (yellow). Bottom: Total amount of failing collocation points in the non-scaled (blue) and the scaled (red) case compared to the total amount of collocation points (yellow). In both the scaled and non-scaled case the amount of failing evaluation points converges quickly (after 800 iterations), but while close to all points fail in the non-scaled case, the number of failing points in the scaled case is relatively small. Hence, the approximated chain-recurrent set is relatively small and the complete Lyapunov function provides more information.

052, Complete Lyapunov functions: Efficient numerical computation.

REFERENCES

Anderson, J. and Papachristodoulou, A. (2015). Advances in computational Lyapunov analysis using sum-of-squares programming. *Discrete Contin. Dyn. Syst. Ser. B*, 20(8):2361–2381.

Argáez, C., Giesl, P., and Hafstein, S. (2017a). Analysing dynamical systems towards computing complete Lyapunov functions. *Proceedings of the 7th International Conference on Simulation and Modeling Methodologies, Technologies and Applications*. SIMULTECH 2017, Madrid, Spain.

Argáez, C., Giesl, P., and Hafstein, S. (2018a). Computation of complete Lyapunov functions for three-dimensional systems. *Submitted*.

Argáez, C., Giesl, P., and Hafstein, S. (2018b). Computational approach for complete Lyapunov functions. *ACCEPTED IN Springer Proceedings in Mathematics and Statistics*.

Argáez, C., Hafstein, S., and Giesl, P. (2017b). Wendland functions a C++ code to compute them. *Proceedings of the 7th International Conference on Simulation and Modeling Methodologies, Technologies and Applications*, pages 323–330. SIMULTECH 2017, Madrid, Spain.

Auslander, J. (1964). Generalized recurrence in dynamical systems. *Contr. to Diff. Equ.*, 3:65–74.

Ban, H. and Kalies, W. (2006). A computational approach to Conley's decomposition theorem. *J. Comput. Nonlinear Dynam*, 1(4):312–319.

Björnsson, J., Giesl, P., and Hafstein, S. (2014a). Algorithmic verification of approximations to complete Lyapunov functions. In *Proceedings of the 21st International Symposium on Mathematical Theory of Networks and Systems*, pages 1181–1188 (no. 0180), Groningen, The Netherlands.

Björnsson, J., Giesl, P., Hafstein, S., Kellett, C., and Li, H. (2014b). Computation of continuous and piecewise affine Lyapunov functions by numerical approximations of the Massera construction. In *Proceedings of the CDC, 53rd IEEE Conference on Decision and Control*, Los Angeles (CA), USA.

Björnsson, J., Giesl, P., Hafstein, S., Kellett, C., and Li, H. (2015). Computation of Lyapunov functions for systems with multiple attractors. *Discrete Contin. Dyn. Syst. Ser. A*, 35(9):4019–4039.

Conley, C. (1978a). *Isolated Invariant Sets and the Morse Index*. CBMS Regional Conference Series no. 38. American Mathematical Society.

Conley, C. (1978b). *Isolated Invariant Sets and the Morse Index*. CBMS Regional Conference Series no. 38. American Mathematical Society.

Conley, C. (1988). The gradient structure of a flow I. *Ergodic Theory Dynam. Systems*, 8:11–26.

Dellnitz, M., Froyland, G., and Junge, O. (2001). The algorithms behind GAIO – set oriented numerical methods for dynamical systems. In *Ergodic theory, analysis, and efficient simulation of dynamical systems*, pages 145–174, 805–807. Springer, Berlin.

Dellnitz, M. and Junge, O. (2002). Set oriented numerical methods for dynamical systems. In *Handbook of dynamical systems, Vol. 2*, pages 221–264. North-Holland, Amsterdam.

Doban, A. (2016). *Stability domains computation and stabilization of nonlinear systems: implications for biological systems*. PhD thesis: Eindhoven University of Technology.

Doban, A. and Lazar, M. (2016). Computation of Lyapunov functions for nonlinear differential equations via a Yoshizawa-type construction. *IFAC-PapersOnLine*, 49(18):29 – 34. 10th IFAC Symposium on Nonlinear Control Systems NOLCOS 2016, Monterey, California, USA, 23-25 August 2016.

Giesl, P. (2007). *Construction of Global Lyapunov Functions Using Radial Basis Functions*. Lecture Notes in Math. 1904, Springer.

Giesl, P. and Hafstein, S. (2015). Review of computational methods for Lyapunov functions. *Discrete Contin. Dyn. Syst. Ser. B*, 20(8):2291–2331.

Giesl, P. and Wendland, H. (2007). Meshless collocation: error estimates with application to Dynamical Systems. *SIAM J. Numer. Anal.*, 45(4):1723–1741.

Goullet, A., Harker, S., Mischaikow, K., Kalies, W., and Kasti, D. (2015). Efficient computation of Lyapunov functions for Morse decompositions. *Discrete Contin. Dyn. Syst. Ser. B*, 20(8):2419–2451.

Hafstein, S. (2007). *An algorithm for constructing Lyapunov functions*. Monograph. Electron. J. Diff. Eqns.

Hsu, C. S. (1987). *Cell-to-cell mapping*, volume 64 of *Applied Mathematical Sciences*. Springer-Verlag, New York.

Hurley, M. (1992). Noncompact chain recurrence and attraction. *Proc. Amer. Math. Soc.*, 115:1139–1148.

Hurley, M. (1995). Chain recurrence, semiflows, and gradients. *J Dyn Diff Equat*, 7(3):437–456.

Hurley, M. (1998). Lyapunov functions and attractors in arbitrary metric spaces. *Proc. Amer. Math. Soc.*, 126:245–256.

Iske, A. (1998). Perfect centre placement for radial basis function methods. Technical Report TUM-M9809, TU Munich, Germany.

Johansen, T. (2000). Computation of Lyapunov functions for smooth, nonlinear systems using convex optimization. *Automatica*, 36:1617–1626.

Johansson, M. (1999). *Piecewise Linear Control Systems*. PhD thesis: Lund University, Sweden.

Kalies, W., Mischaikow, K., and VanderVorst, R. (2005). An algorithmic approach to chain recurrence. *Found. Comput. Math*, 5(4):409–449.

Kamyar, R. and Peet, M. (2015). Polynomial optimization with applications to stability analysis and control – an alternative to sum of squares. *Discrete Contin. Dyn. Syst. Ser. B*, 20(8):2383–2417.

Krauskopf, B., Osinga, H., Doedel, E. J., Henderson, M., Guckenheimer, J., Vladimirsky, A., Dellnitz, M., and Junge, O. (2005). A survey of methods for computing (un)stable manifolds of vector fields. *Internat. J. Bifur. Chaos Appl. Sci. Engrg.*, 15(3):763–791.

Lyapunov, A. M. (1992). The general problem of the stability of motion. *Internat. J. Control*, 55(3):521–790. Translated by A. T. Fuller from Édouard Davaux's French translation (1907) of the 1892 Russian original.

Marinósson, S. (2002). Lyapunov function construction for ordinary differential equations with linear programming. *Dynamical Systems: An International Journal*, 17:137–150.

Narcowich, F. J., Ward, J. D., and Wendland, H. (2005). Sobolev bounds on functions with scattered zeros, with applications to radial basis function surface fitting. *Mathematics of Computation*, 74:743–763.

Osipenko, G. (2007). *Dynamical systems, graphs, and algorithms*. Springer, Berlin. Lecture Notes in Mathematics 1889.

Wendland, H. (1998). Error estimates for interpolation by compactly supported Radial Basis Functions of minimal degree. *J. Approx. Theory*, 93:258–272.

Wendland, H. (2005). *Scattered data approximation*, volume 17 of *Cambridge Monographs on Applied and Computational Mathematics*. Cambridge University Press, Cambridge.