Iterative construction of complete Lyapunov functions: Analysis of algorithm efficiency [⋆]

Carlos Argáez$^{1[0000-0002-0455-8015]}$, Peter Giesl$^{2[0000-0003-1421-6980]}$, and Sigurdur Hafstein$^{1[0000-0003-0073-2765]}$

[1] Science Institute, University of Iceland, Dunhagi 3, 107 Reykjavík, Iceland
[2] Department of Mathematics, University of Sussex, Falmer, BN1 9QH, United Kingdom
{carlos, shafstein}@hi.is, P.A.Giesl@sussex.ac.uk

Abstract. Differential equations describe many interesting phenomena arising from various disciplines. This includes many important models, e.g. predator-prey in population biology or the Van der Pol oscillator in electrical engineering. Complete Lyapunov functions allow for the systematic study of the qualitative behaviour of complicated systems. In this paper, we extend the analysis of the algorithm presented in [1]. We study the efficiency of our algorithm and discuss important sections of the code.

Keywords: Dynamical System, Complete Lyapunov function, Orbital derivative, Meshless collocation, Radial Basis Functions, Algorithms, Scalability

1 INTRODUCTION

Studying systems that change in time often requires the semantics of differential equations. Autonomous differential equations define dynamical systems which describe the evolution of a given system.

The study of dynamical systems is a mathematical discipline originated by Newtonian mechanics. It is in fact a consequence of Henri Poincaré's work on celestial mechanics; hence, he is often regarded as the founder of dynamical systems. Poincaré studied the problem of three bodies motion describing in detail several concepts commonly studied in dynamical systems such as stability.

Let us consider a general autonomous ordinary differential equation (ODE) of the form,

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

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

To solve the differential equation (1) we consider an initial condition. Then, by time-integration, we would obtain a specific solution, however, in general this

[⋆] The first author in this paper is supported by the Icelandic Research Fund (Rannís) grant number 163074-052, Complete Lyapunov functions: Efficient numerical computation.

is not possible analytically, but only numerically. Moreover, this procedure needs to be repeated for each new initial condition and is thus inefficient.

In recent years, the increase of computational power and more efficient algorithms, provided by numerical methods, allowed to speed up such procedures and to increase the collection of problems that can be analysed and studied. The initial condition dependence, however, restricts the understanding of complicated systems that cannot be described by a small sample of orbits and known trajectories.

The correct description of such systems may find application in physics, engineering, biology and other disciplines. For that reason, several techniques to expand the understanding and the analysis of such systems have been developed during the years. Many famous techniques differ in approach, difficulty and efficiency. To summarize some, we can point out the direct simulations of solutions with many different initial conditions, as discussed previously and computing invariant manifolds, which form the boundaries of attractors' basin of attraction [33]. Other techniques to analyse dynamical systems are the set oriented methods [16] and the cell mapping approach [24]. These methods are supposed to work provided enough computational resources are available.

While there have been many advances introduced by different brilliant minds, the approach provided by Aleksandr Lyapunov is of particular relevance. His work is based on defining stability of sets in the phase space ordinary differential equations and provides another approach to study dynamical systems by their qualitative behaviour. In particular, it turns out to be useful to characterize the flow by studying attractors and repellers of the dynamics.

Lyapunov proposed to construct an auxiliary scalar-valued function whose domain is a subset of the state-space. This function is strictly decreasing along all solution trajectories in a neighbourhood of an attractor, e.g. an equilibrium point or a periodic orbit. The function's minimum is attained on the attractor, hence, all solutions starting on a decreasing trajectory are bound to converge to the attractor. Today this function is known as a strict *Lyapunov function* in his honor. Here its classical definition [34].

Definition 1 (Classical (strict) Lyapunov function). *A classical (strict) Lyapunov function $V(\mathbf{x})$ for the system* (1) *is a scalar-valued C^1 function defined on a neighbourhood of an attractor of the system. It attains its minimum at the attractor, and is otherwise strictly decreasing along solutions of the ODE.*

Note that the function in the last definition is limited to the neighbourhood of one single attractor. A redefinition of this function was later given to the whole phase space and it is called a *complete Lyapunov function* [13,14,26,27], see Definition 2.

Definition 2 (Complete Lyapunov function). *A complete Lyapunov function for the system* (1) *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.*

This general case provides tools to describe the complete qualitative behaviour of the dynamical system on the whole phase space and divides the state-space into two disjoint areas, in which the system behaves fundamentally differently. The first one, referred to as the gradient-like flow, is an area where the systems flows through and has the property of having negative orbital derivative, if smooth enough (the orbital derivative is the derivative along solutions). The second, referred to as the chain-recurrent set, is an area where infinitesimal perturbations can make the system recurrent, see Definition 3; the points in this set have vanishing orbital derivative. In this work we will refer to these two conditions (negative orbital derivative for the gradient-like flow and zero orbital derivative to the chain-recurrent set) as the *Lyapunov condition.*

Definition 3 (Chain-recurrent set). *A point is said to be in the chain-recurrent set, if for any $\epsilon > 0$ and any given time, an ϵ-trajectory exists that starts at the point and returns to it after the given time.*

An ϵ-trajectory is arbitrarily close to a true system's solution and a point in the chain-recurrent set is recurrent or almost recurrent; for a precise definition see, e.g. [13]. The dynamics outside of the chain-recurrent set are similar to a gradient system, i.e. a system (1) where the right-hand side $\mathbf{f}(\mathbf{x})$ is given by the gradient $\nabla U(\mathbf{x})$ of a function $U\colon \mathbb{R}^n \to \mathbb{R}$.

Conley [13] gave the first mathematical proof of existence of complete Lyapunov functions for flows in compact metric spaces. Hurley [27] extended these results to flows in separable metric spaces.

In general, this work is a continuation of the algorithms described in [3,4,1,6,7]. It is a modification of a general method to compute classical Lyapunov functions for one stable equilibrium using Radial Basis Functions [19]. Nonetheless, along this work we describe an algorithm to construct complete Lyapunov functions considering that the $l^{\#}$-norm, with $\# = 1, 2$, for the Lyapunov condition at a given set of points must be kept constant. Previously, we described the benefits of keeping the Lyapunov condition constant in the l^1-norm [1]. However, selecting the right norm turns out to be a non-trivial task.

In Sec. 3.2, we discuss this question in more detail.

Furthermore, when developing algorithms to study and to describe dynamical systems, an important question on the efficiency arises and we address this issue in Sec. 4.

The general idea to construct a complete Lyapunov function under this approach, is to approximate a "solution" to the ill-posed problem $V'(\mathbf{x}) = -1$, where $V'(\mathbf{x}) = \nabla V(\mathbf{x}) \cdot \mathbf{f}(\mathbf{x})$ is the derivative along solutions of the ODE, i.e. the orbital derivative.

A function v is computed using Radial Basis Functions, a mesh-free collocation technique, such that $v'(\mathbf{x}) = -1$ is fulfilled at all points $\mathbf{x}$ in a finite set of collocation points X.

The discretized problem of computing v is well-posed and has a unique solution. However, the computed function v will fail to fulfil the PDE at some points of the chain-recurrent set, such as an equilibrium or a periodic orbit. For some $\mathbf{x}$ in the chain-recurrent set we must have $v'(\mathbf{x}) \geq 0$ and this is the key component of our general algorithm to localize the chain-recurrent set; we determine the chain-recurrent through the area where $v'(\mathbf{x}) \not\approx -1$.

2 PRELIMINARIES

2.1 MESH-FREE COLLOCATION

Radial Basis Functions (RBFs) are a powerful methodology [19] for general interpolation problems based on mesh-free collocation methods. The construction of complete Lyapunov functions can be posed as a generalized interpolation problem.

RBFs are real-valued functions whose value depends only on the distance of its argument from the origin. Many examples of RBFs can be given, however, the most common ones are Gaussians, multiquadrics and Wendland functions.

Our algorithm uses Wendland functions, which are compactly supported and positive definite functions introduced in [38]. Their form is that of a polynomial on their compact support defined by a recurrent formula. The corresponding Reproducing Kernel Hilbert Space is norm-equivalent to a Sobolev space.

Note that in the context of RBF, positive definite function ψ refers to the matrix $(\psi(\|\mathbf{x}_i - \mathbf{x}_j\|))_{i,j}$ being positive definite for $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$, where $\mathbf{x}_i \neq \mathbf{x}_j$ if $i \neq j$.

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 positive definite 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.

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

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

It is well-known [39] 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 $\lambda_j(v) = r_j$, $1 \leq j \leq N$ and the superscript $\mathbf{y}$ denotes the application of the operator λ_j with respect to the variable $\mathbf{y}$.

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 pairwise distinct collocation points $\mathbf{x}_1, \ldots, \mathbf{x}_N \in \mathbb{R}^n$ of the phase space and define functionals

$$\lambda_j(v) := (\delta_{\mathbf{x}_j} \circ L)v = v'(\mathbf{x}_j) = \nabla v(\mathbf{x}_j) \cdot \mathbf{f}(\mathbf{x}_j),$$

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 $1 \leq j \leq N$. 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. The superscript $\mathbf{y}$ denotes that the operator is applied to the function $\mathbf{y} \mapsto \Phi(\mathbf{x} - \mathbf{y})$ for a fixed $\mathbf{x}$. A crucial ingredient is the knowledge of 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 $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ 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 [19,21], see also [36,39].

The advantage of mesh-free collocation 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.

Wendland functions The Wendland functions are expressed by the general form: $\psi(\mathbf{x}) := \psi_{l,k}(c\|\mathbf{x}\|)$, where $c > 0$ and $k \in \mathbb{N}$ is a smoothness parameter. Along this work, the parameter l is fixed as $l = \lfloor \frac{n}{2} \rfloor + k + 1$. The Reproducing Kernel Hilbert Space corresponding to $\psi_{l,k}$ contains the same functions as the Sobolev space $W_2^{k+(n+1)/2}(\mathbb{R}^n)$ and the spaces are norm equivalent.

The Wendland functions $\psi_{l,k}$ are defined by the recursion: For $l \in \mathbb{N}$ and $k \in \mathbb{N}_0$, 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 $x_+ = \max(0, x)$ and has higher precedence than taking power, i.e. $(1-r)_+^l = [(1-r)_+]^l$.

Collocation points Our algorithm is based on the idea of solving the ill-posed problem $V'(\mathbf{x}) = \nabla V(\mathbf{x}) \cdot \mathbf{f}(\mathbf{x}) = -1$ on a collocation of points $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \subset \mathbb{R}^n$.

Our set X of collocation points, is a subset of a hexagonal grid with fineness-parameter $\alpha_{\text{Hexa-basis}} \in \mathbb{R}^+$ constructed according to the next equation:

$$\begin{aligned}
&\left\{ \alpha_{\text{Hexa-basis}} \sum_{k=1}^{n} i_k \omega_k : i_k \in \mathbb{Z} \right\}, \text{ where} \qquad (3)\\
\omega_1 &= (2\epsilon_1, 0, 0, \ldots, 0)\\
\omega_2 &= (\epsilon_1, 3\epsilon_2, 0, \ldots, 0)\\
\vdots &\quad \vdots\\
\omega_n &= (\epsilon_1, \epsilon_2, \epsilon_3, \ldots, (n+1)\epsilon_n) \text{ and}\\
\epsilon_k &= \sqrt{\frac{1}{2k(k+1)}}, \quad k \in \mathbb{N}.
\end{aligned}$$

The hexagonal grid is optimal to balance the opposing aims of a dense grid and a low condition number of the collocation matrix. It delivers the matrix with the minimal condition number for a fixed fill distance [28].

Since $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ for an equilibrium $\mathbf{x}$, an equilibrium cannot be used as a collocation point, because otherwise the collocation matrix is singular.

The approximation v is then given by the function that satisfies the PDE $v'(\mathbf{x}) = -1$ at all collocation points and it is norm minimal in the corresponding Reproducing Kernel Hilbert space. Practically, we compute v by solving a system of N linear equations, where N is the number of collocation points.

We set $\psi_0(r) := \psi_{l,k}(cr)$ with a 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

$$v(\mathbf{x}) = \sum_{j=1}^{N} \beta_j \langle \mathbf{x}_j - \mathbf{x}, \mathbf{f}(\mathbf{x}_j)\rangle \psi_1(\|\mathbf{x} - \mathbf{x}_j\|), \qquad (4)$$

$$\begin{aligned}
v'(\mathbf{x}) = \sum_{j=1}^{N} \beta_j \Big[&- \psi_1(\|\mathbf{x} - \mathbf{x}_j\|)\langle \mathbf{f}(\mathbf{x}), \mathbf{f}(\mathbf{x}_j)\rangle \qquad (5)\\
&+ \psi_2(\|\mathbf{x} - \mathbf{x}_j\|)\langle \mathbf{x} - \mathbf{x}_j, \mathbf{f}(\mathbf{x})\rangle \cdot \langle \mathbf{x}_j - \mathbf{x}, \mathbf{f}(\mathbf{x}_j)\rangle \Big]
\end{aligned}$$

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

$$\begin{aligned}
a_{ij} = {}& \psi_2(\|\mathbf{x}_i - \mathbf{x}_j\|)\langle \mathbf{x}_i - \mathbf{x}_j, \mathbf{f}(\mathbf{x}_i)\rangle\langle \mathbf{x}_j - \mathbf{x}_i, \mathbf{f}(\mathbf{x}_j)\rangle \qquad (6)\\
&- \psi_1(\|\mathbf{x}_i - \mathbf{x}_j\|)\langle \mathbf{f}(\mathbf{x}_i), \mathbf{f}(\mathbf{x}_j)\rangle
\end{aligned}$$

for $i \neq j$ and

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

More detailed explanations on this construction are given in [19, 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.

Evaluation grid Once we have solved the PDE on the collocation points, we use a different evaluation grid $Y_{\mathbf{x}_j}$, around each collocation point $\mathbf{x}_j$. This grid can be constructed in many different ways. Important is, however, to always associate each evaluation point to a unique collocation point.

2.2 PREVIOUS ALGORITHMS

To obtain a classical Lyapunov function for a nonlinear system already is a hard task. However, thanks to mathematical research, efficient algorithms to compute Lyapunov functions have been proposed, cf. [20] for a recent review of such methods. One of these algorithms to compute classical Lyapunov functions for an equilibrium approximates the solution of the PDE $V'(\mathbf{x}) = \nabla V(\mathbf{x})\cdot\mathbf{f}(\mathbf{x}) = -1$ [19] using mesh-free collocation with Radial Basis Functions. It constructs an approximate solution to this linear partial differential 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 functions. However, as discussed before, a complete Lyapunov function cannot have a negative derivative on the chain-recurrent set, hence, the equation is ill-posed. 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 set and the gradient-like flow region, see [3]. We determine the chain-recurrent set as the area, in which the condition $v'(\mathbf{x}) \approx -1$ is not satisfied and 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.

This classification allows us to reconstruct the complete Lyapunov function considering that it should be constant on X^0. However, such methodology requires certain considerations. The speed of the flow of the dynamical system can vary considerably. Therefore, the orbital derivative of the complete Lyapunov function can vary as well. These speed variations render it difficult to classify the chain-recurrent set under a fixed criterion.

Therefore, we introduced in [4] an “almost” normalized approach, i.e., the original dynamical system (1) was substituted by

$$\dot{\mathbf{x}} = \hat{\mathbf{f}}(\mathbf{x}), \quad \text{where} \quad \hat{\mathbf{f}}(\mathbf{x}) = \frac{\mathbf{f}(\mathbf{x})}{\sqrt{\delta^2 + \|\mathbf{f}(\mathbf{x})\|^2}} \tag{7}$$

with a small parameter $\delta > 0$. This allowed us to reduce the over-estimation, i.e. the "noise", of the chain-recurrent set.

2.3 RECONSTRUCTION OF A COMPLETE LYAPUNOV FUNCTION

Once the chain-recurrent set is classified, enough information is obtained to reconstruct the complete Lyapunov using different Lyapunov conditions for X^0 and X^-. That is, we solve the PDE for V again, but now using different values on the right-hand side. We have considered three different ways of reconstructing the complete Lyapunov function. Next, we give a review of these three cases.

Reconstruction with binary parameters The two sets X^0 and X^- provide important information about the ODE under consideration: the first one X^0, where $v'(\mathbf{x}) \approx 0$ approximates the chain-recurrent set, including equilibria, periodic and homoclinic orbits, while the second set X^-, where $v'(\mathbf{x}) \approx -1$, approximates the area where the flow is gradient-like, i.e. where solutions pass through.

Our first approach was to reconstruct the complete Lyapunov function with the Lyapunov conditions 0 and -1 respectively for the two different sets, i.e.,

$$V'(\mathbf{x}) = \begin{cases} 0 & \text{if } \mathbf{x} \in X^0, \\ -1 & \text{if } \mathbf{x} \notin X^0 \end{cases}$$

This approximation, however, can lead to convergence to a trivial (constant) solution, i.e. $V'(\mathbf{x}) = 0$. The right-hand side of $V'(\mathbf{x})$ has a jump, so that the constructed function does not have a "smooth" orbital derivative. Illustrative examples can be found in Figures 6 and 7 in [1].

Reconstruction with exponential decay This approach was based on the idea that smoothing the right-hand side would replace the jump with a smoother transition. To do that, we replaced the original binary right-hand side by a smooth exponential decay,

$$V'(\mathbf{x}) = r(\mathbf{x}) := \begin{cases} 0 & \text{if } \mathbf{x} \in X^0, \\ -\exp\left(-\frac{1}{\xi \cdot \mathfrak{d}^2(\mathbf{x})}\right) & \text{if } \mathbf{x} \notin X^0, \end{cases} \tag{8}$$

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 determine the chain-recurrent set.

However, there is a problem with this methodology. If iterated in an attempt to improve the reconstruction of the Lyapunov function, the area where $v'(\mathbf{x}) \approx 0$ holds (or X^0) could grow in size and the functions v could converge towards a trivial, constant solution, although slower that when using binary parameters. More details are found in paper [4].

Reconstruction by averaging the orbital derivative This methodology is based on using the average of $v'(\mathbf{y})$ for points $\mathbf{y} \in Y_{\mathbf{x}_j}$ (or 0 if the average is positive), i.e. the points in the evaluation grid for 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. Additionally, we scale the right-hand side $\tilde{r}_j$ by $1/\|\tilde{r}(\mathbf{x}_i)\|_{l^1} = 1/(\sum_{i=1}^N |\tilde{r}(\mathbf{x}_i)|)$, such that the new sum $\sum_{i=1}^N r(\mathbf{x}_i)$ of the values in right-hand side over all collocation points is constant in each iteration, see the algorithm in the next section. This prevents the iterations from converging to the trivial solution. We will show the improvement of the performance of the algorithm in an example. This method was introduced in [1].

For the evaluation points $\mathbf{y} \in Y_{\mathbf{x}_j}$ 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 $n \geq 3$ [6], 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 [3,4] 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 large amount of points to evaluate. More details on the evaluation grid are given in Sec. 3.1.

3 ALGORITHM

Our algorithm is based on the algorithms described in [3] and where the speed of the system has been normalized as in (7); explained in detail in [4]. The Wendland functions for the numerical computations are constructed with the software from [5]. In this paper, as in [1], 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 pairwise distinct 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

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

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 note that we have chosen $r < 1$ to avoid overlap of evaluation grids [1,7].

This means that there will not be any evaluated points to provide information about the dynamical system apart from the ones aligned with the flow. On the other hand, this evaluation grid avoids exponential growth in size as the system's dimension increases.

We start by computing the approximate solution v_0 of $V'(\mathbf{x}) = -1$ with collocation points X. As we have previously done in [3,4], 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 follow the same idea as [1] and replace 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 $x_- = \min(0, x)$. We will refer to this as the “non-scaled” version.

While in [4] 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 a smoothed out orbital derivative. However, this approach can lead to a decrease of “energy” from the original Lyapunov function. Recall that the original value of the orbital derivative condition in 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

The system we use to benchmark our methodology is the following:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \mathbf{f}(x, y) = \begin{pmatrix} 1 - x^2 \\ -xy \end{pmatrix}, \tag{9}$$

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 (9) are: we replace $\mathbf{f}$ by $\hat{\mathbf{f}}$ according to (7) with $\delta^2 = 10^{-8}$, and we choose $\alpha_{\text{Hexa-basis}} = 0.1$ and used all points in the hexagonal grid that are in the area $[-2, 2]^2 \in \mathbb{R}^2$. This gave us a total amount of $2,016$ 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 thus $40,320$.

Evaluation grid for our system In Figure 1 we plot the evaluation grid (9) for the system (9) Notice that we use colour to indicate the value of the complete Lyapunov function computed.

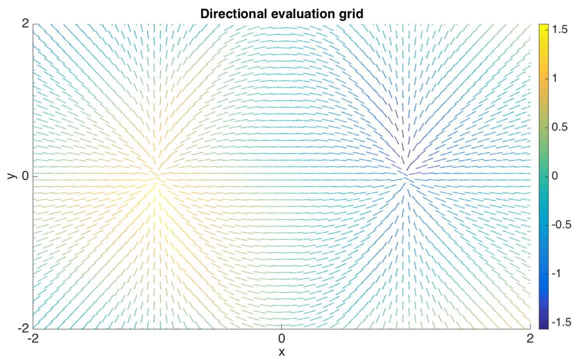

Fig. 1. Evaluation grid for (9). All points of the evaluation grid are aligned with the direction of the flow of the dynamical system. The flow is marked with colours to indicate the value of the computed complete Lyapunov functions.

Complete Lyapunov function, orbital derivative and iterations Now, we present the results obtained with our algorithm using the l^1-norm for the scaling and using 10,000 iterations.

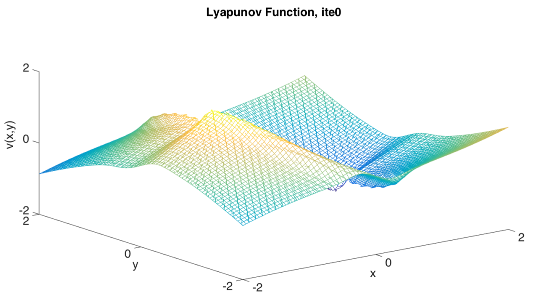

Fig. 2. Complete Lyapunov function v_0 for system (9), obtained by solving $V'(\mathbf{x}) = -1$, iteration 0, [1].

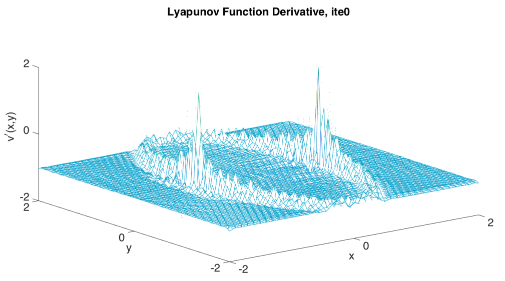

Fig. 3. Complete Lyapunov function derivative v_0' for system (9), iteration 0, [1].

The complete Lyapunov function approximation for system (9) is given in Figure 2 and its orbital derivative in Figure 3. We start by approximating the solution of $V'(\mathbf{x}) = -1$ by v_0.

The complete Lyapunov function at this iteration (ite=0) poorly approximates the orbital derivative fixe to -1 in some areas. Indeed, Figure 3 shows that the orbital derivative can be positive with values between 0 and 2. From this, we obtain information on the values over the evaluation grid that failed as $v_0'(\mathbf{y}) > \gamma$. These points are an approximation of the chain-recurrent set under the scaled method.

The function v_0 and its orbital derivative v_0' are shown in Figures 2 and 3, respectively. Figure 2 already shows that the unstable equilibrium at $(-1, 0)$ is a maximum of v_0 and the asymptotically stable equilibrium at $(1, 0)$ is a minimum; the orbital derivative v_0' approximates -1 quite well apart from the equilibria and an ellipse covering the heteroclinic orbits, connecting the two equilibria, which touch the boundary of the considered area.

However, a different perspective of Figure 2, see Figure 1 with the directional grid, 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 [3] must fail.

We used the previous non-scaled method and solved $V'(\mathbf{x}_j) = \tilde{r}_j$ iteratively and the scaled-method $V'(\mathbf{x}_j) = r_j = \frac{N}{\sum_{l=1}^{N} |\tilde{r}_l|} \tilde{r}_j$ (l^1-scaled method), see algorithm Sec. 3.1. In the first case (non-scaled method) the solution converges to the constant solution, while in the second case (scaled-method) we obtain a smooth complete Lyapunov function and a smooth orbital derivative, see Figure 4.

3.2 l^1- and l^2-NORM SCALING

In this section we analyse the difference between using the l^1- and l^2-norms for scaling in our method. In [1], we originally considered scaling using the l^1 norm. However, when we replace step 4 in the algorithm with

$$r_j = \left(\frac{\tilde{r}_j}{\|\tilde{r}_j\|_{l^2}} \right) \|\tilde{r}_0\|_{l^2},$$

then we obtain a method that keeps the l^2 norm of the vector $(r_j)_{j=1,\ldots,N}$ constant in every step. Similarly, we can define

$$r_j = \left(\frac{\tilde{r}_j}{\|\tilde{r}_j\|_{l^p}} \right) \|\tilde{r}_0\|_{l^p}$$

for any $p \geq 1$, which scales by the l^p-norm.

When comparing the results using the l^2-norm instead of the l^1-norm, one notices that there are considerable differences, see Figures 5 and 6. Indeed, it look like scaling using the l^2-norm gives a smoother and better behaved complete Lyapunov function.

Figure 9 gives the evolution of the l^1- and l^2-norms of the Lyapunov condition vector at each iteration for the non-scaled method. It shows how the norms tend to zero and after 3000 iterations both deliver nearly constant approximations with orbital derivative close to zero. Clearly scaling is necessary for this problem and additionally improves previous results be smoothing the orbital derivative. Understanding the different results for the l^1- and l^2-norm scalings requires further analysis, which is a subject of future work.

4 COMPLEXITY ANALYSIS OF THE ALGORITHM

In this section we analyze the computational complexity of our algorithm. As before n stands for the dimension of the system and N for the number of collocation points. Each collocation points has $2m$ evaluation points associated to it. We assume that the evaluation of $\mathbf{f}(\mathbf{x})$ is $O(n)$ because it has n coordinates. Our algorithm has the following structure:

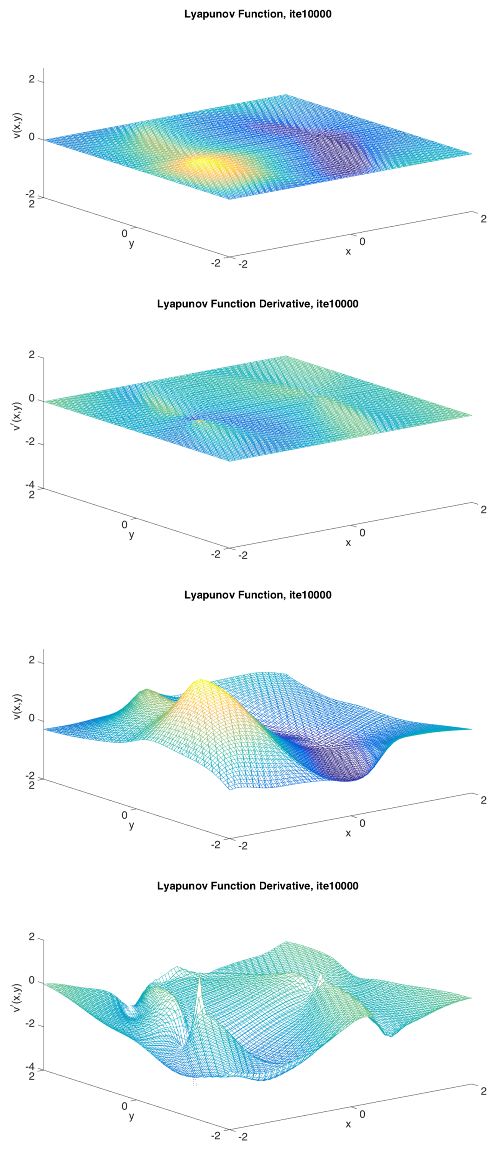

Fig. 4. Evolution of the complete Lyapunov function for system (9) and its orbital derivative. The two-upper figures are obtained with the non-scaled method while the lower-two figures are obtained with the scaled method. ite=10000, [1].

Algorithm 1 *The different parts of the algorithm are the following:*

(i) **Generate the collocation points** We construct N collocation points, each of dimension n. For this we need $O(Nn)$ operations.

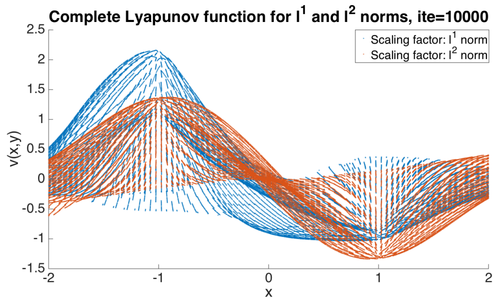

Fig. 5. Complete Lyapunov function v for system (9), iteration 10000 with the scaled method (l^1 blue, l^2 red). Plane $Z - X$.

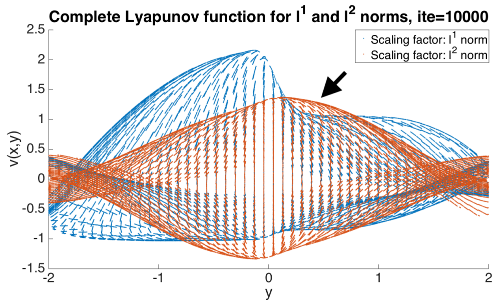

Fig. 6. Complete Lyapunov function v for system (9), iteration 10000 with the scaled method (l^1 blue, l^2 red). Plane $Z - Y$.

(ii) **Interpolation matrix and Cholesky decomposition** The interpolation matrix A is built using the formula (6). Each of the N^2-elements a_{ij} can be computed in $O(n)$ so the complexity is $O(N^2n)$. Since A is positive definite we Cholesky decompose it, which is $O(N^3)$. Note that the matrix A and its Cholesky decomposition must only be computed once, regardless of the number of iterations!

(iii) **Construction of the evaluation grid** For each of the N collocation points we construct $2m$ evaluation points, each of dimension n. This is $O(Nnm)$ operations.

(iv) **Compute the Lyapunov function**

To compute the coefficients β_j in formulas (4) and (5) we need to solve the equation $A\beta = \mathbf{r}$, where A is Cholesky decomposed. This is well known to be $O(N^2)$ (back substitution).

(v) **Classification of the failing points in the chain-recurrent set** We have to evaluate v' at $2m$ evaluation points for each collocation point, of which there are N. From formula (5) we see that each evaluation is $O(Nn)$. Together we need $O(N^2nm)$ operations for the evaluation.

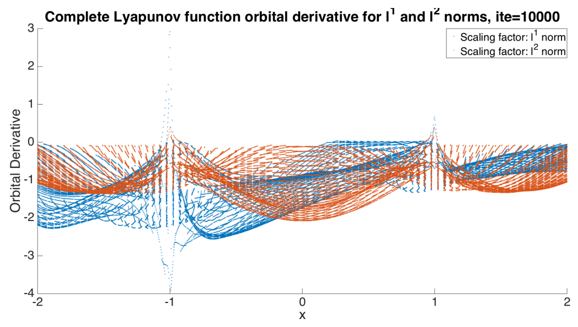

Fig. 7. Complete Lyapunov function derivative v' for system (9), iteration 10000 with the scaled method (l^1 blue, l^2 red). Plane $Z-X$.

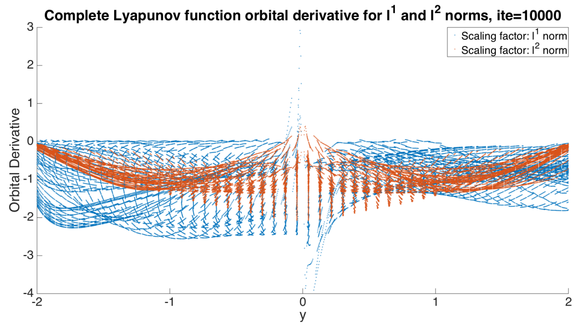

Fig. 8. Complete Lyapunov function derivative v' for system (9), iteration 10000 with the scaled method (l^1 blue, l^2 red). Plane $Z-Y$.

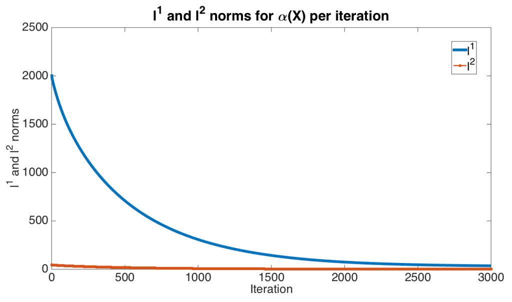

Fig. 9. Evolution of the l^1-and l^2-norms along iterations.

(vi) **Lyapunov condition** First we need to compute the average value of v' at the evaluation points for each collocation point. Since we already have evaluated v' at all evaluation points in the last step, this is $O(Nm)$. Then we have to compute the average value (l^1-norm) or the root-means-square

value (l^2-norm) over all the collocation points, which is $O(N)$. Together we have complexity of $O(Nm)$.

Altogether this proves the following result, noting that we only once have to generate and Cholesky decompose the collocation matrix A and that the most expensive operation in each iteration is computing the average of v' at the evaluation points for each collocation point.

Lemma 1. *To compute an approximation to a complete Lyapunov function with our method using I iterations, we need $O(N^3 + IN^2nm)$ operations.*

5 CONCLUSIONS

We have introduced a methodology to construct approximations to smooth complete Lyapunov function. In particular, we have addressed three main questions in this paper: First, by introducing a scaling factor on the Lyapunov condition, we introduced a fundamental change in the iterative construction. Second, we compared scaling using the l^1-norm and the l^2-norm. The l^2-norm scaling seems to be superior, but we do not have a convincing reason for this. This should definitely be addressed in future work. Third, we analysed the numerical complexity of our algorithm.

6 ACKNOWLEDGEMENT

First author wants to thank Dr. A. Argáez for nice discussions on normed spaces as well as the Icelandic Research Fund (Rannís) for funding this work under the grant: number 163074-052, Complete Lyapunov functions: Efficient numerical computation.

References

1. Argez, C., Giesl, P. and Hafstein, S. (2018) 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
2. 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.
3. 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.
4. Argáez, C., Giesl, P., and Hafstein, S. (2018b). Computational approach for complete Lyapunov functions. *ACCEPTED IN Springer Proceedings in Mathematics and Statistics.*

5. 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.
6. Argáez, C., Giesl, P., and Hafstein, S. (2018a). Computation of complete Lyapunov functions for three-dimensional systems. *Submitted*.
7. C. Argáez, P. Giesl, and S. Hafstein, *Computation of complete Lyapunov functions for three-dimensional systems.* 57th IEEE Conference on Decision and Control (CDC), 2018 (To be published)
8. Auslander, J. (1964). Generalized recurrence in dynamical systems. *Contr. to Diff. Equ.*, 3:65–74.
9. Ban, H. and Kalies, W. (2006). A computational approach to Conley's decomposition theorem. *J. Comput. Nonlinear Dynam*, 1(4):312–319.
10. 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.
11. 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.
12. 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.
13. Conley, C. (1978a). *Isolated Invariant Sets and the Morse Index.* CBMS Regional Conference Series no. 38. American Mathematical Society.
14. Conley, C. (1988). The gradient structure of a flow I. *Ergodic Theory Dynam. Systems*, 8:11–26.
15. 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.
16. 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.
17. Doban, A. (2016). *Stability domains computation and stabilization of nonlinear systems: implications for biological systems.* PhD thesis: Eindhoven University of Technology.
18. 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.
19. Giesl, P. (2007). *Construction of Global Lyapunov Functions Using Radial Basis Functions.* Lecture Notes in Math. 1904, Springer.
20. Giesl, P. and Hafstein, S. (2015). Review of computational methods for Lyapunov functions. *Discrete Contin. Dyn. Syst. Ser. B*, 20(8):2291–2331.
21. Giesl, P. and Wendland, H. (2007). Meshless collocation: error estimates with application to Dynamical Systems. *SIAM J. Numer. Anal.*, 45(4):1723–1741.
22. 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.

23. Hafstein, S. (2007). *An algorithm for constructing Lyapunov functions.* Monograph. Electron. J. Diff. Eqns.
24. Hsu, C. S. (1987). *Cell-to-cell mapping*, volume 64 of *Applied Mathematical Sciences.* Springer-Verlag, New York.
25. Hurley, M. (1992). Noncompact chain recurrence and attraction. *Proc. Amer. Math. Soc.*, 115:1139–1148.
26. Hurley, M. (1995). Chain recurrence, semiflows, and gradients. *J Dyn Diff Equat*, 7(3):437–456.
27. Hurley, M. (1998). Lyapunov functions and attractors in arbitrary metric spaces. *Proc. Amer. Math. Soc.*, 126:245–256.
28. Iske, A. (1998). Perfect centre placement for radial basis function methods. Technical Report TUM-M9809, TU Munich, Germany.
29. Johansen, T. (2000). Computation of Lyapunov functions for smooth, nonlinear systems using convex optimization. *Automatica*, 36:1617–1626.
30. Johansson, M. (1999). *Piecewise Linear Control Systems.* PhD thesis: Lund University, Sweden.
31. Kalies, W., Mischaikow, K., and VanderVorst, R. (2005). An algorithmic approach to chain recurrence. *Found. Comput. Math*, 5(4):409–449.
32. 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.
33. 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.
34. 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.
35. Marinósson, S. (2002). Lyapunov function construction for ordinary differential equations with linear programming. *Dynamical Systems: An International Journal*, 17:137–150.
36. 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.
37. Osipenko, G. (2007). *Dynamical systems, graphs, and algorithms.* Springer, Berlin. Lecture Notes in Mathematics 1889.
38. Wendland, H. (1998). Error estimates for interpolation by compactly supported Radial Basis Functions of minimal degree. *J. Approx. Theory*, 93:258–272.
39. Wendland, H. (2005). *Scattered data approximation*, volume 17 of *Cambridge Monographs on Applied and Computational Mathematics.* Cambridge University Press, Cambridge.