Adaptive Sine-Cosine Optimization Technique for Stability and Domain of Attraction Analysis

Messaoud Aloui[1], Faiçal Hamidi[2], Mohammed Aoun[3], Houssem Jerbi[4]
Research Laboratory MACS LR16ES22, University of Gabes, Gabes, Tunisia[1]
Laboratory of Information-Communication and Knowledge Sciences and Techniques,
University of Western Brittany, Lorient, France[2]
Research Laboratory MACS LR16ES22, University of Gabes, Gabes, Tunisia[2,3]
Department of Industrial Engineering-College of Engineering, University of Hail, Hail 1234, Saudi Arabia[4]

***Abstract*—In the last few years, researchers have concentrated on estimating and maximizing the Domain of Attraction of autonomous nonlinear systems. Based on the Lyapunov theory, the proposed approach in this paper aims to give an accurate estimation of the Domain of Attraction with high performance against the existing conventional methods. The Adaptive Sine-Cosine Algorithm has been considered one of the most advanced algorithms. It combines a large exploration with a strong local search and provides high-quality convergence conditions. This paper uses the benefits of the Adaptive Sine-Cosine Algorithm to develop a flexible method to estimate the Domain of Attraction by an oriented sampling to guarantee the largest sublevel related to the given Lyapunov function. The approach is applied to some benchmark examples and validates its efficiency and its ability to provide performant results.**

***Keywords*—Domain of Attraction; nonlinear autonomous systems; Lyapunov function; Lyapunov's theory; stability; optimization; Adaptive Sine-Cosine Algorithm**

I. Introduction

In the pursuit of excellence, individuals often strive for perfection in order to effectively navigate a wide array of situations. However, as absolute perfection is unattainable, the human focus shifts towards identifying the most favorable conditions that respect reliability constraints, thereby giving rise to the notion of "constrained optimization problems" [1].

Optimization problems exist in most scientific research fields. For example, they are frequently encountered in engineering applications. As a result, sophisticated algorithms are necessary for addressing optimization issues [2].

The selection of the convenient optimization algorithm is related to the type and the complexity of the addressed problem. For convex optimization problems with a low level of complexity, well-efficient algorithms relying on gradient computation are generally recommended, thanks to their simplicity and accuracy [3]. However, dealing with non-convex or nonlinear problems that involve a high number of decision variables requires employing a different class of optimization methods called metaheuristics [4].

Metaheuristics are highly recommended optimization tools, owing to their ability to handle high-dimensional optimization problems even without a high amount of information about the objective function itself. Many metaheuristic algorithms have been developed based on inspiration from some behaviors observed in nature, particularly in swarm intelligence. The simplicity of their structure, the minimal number of parameters required, the no need for derivative and gradient mechanisms, and the ability to avoid local solutions are among the advantages that have given metaheuristics importance in different areas of research [4], [5].

The study of system performances, especially stability analysis, has greatly benefited from the use of metaheuristics. Researchers become able to improve the performance of the system in search based on an adequate selection of parameters using metaheuristics. Some works use metaheuristics in the observability study [6]. Some others take the benefits of metaheuristics to determine an optimized tuning to the PID controller [7], [8].

The field of control engineering presents two primary types of system behavior: linear and nonlinear.

For linear systems, a comprehensive theory of stability analysis already exists, which involves techniques such as checking the eigenvalues of the state matrix, applying the Routh criterion, and examining the poles of the transfer function. However, in practice, most systems exhibit nonlinear behavior, making these conventional methods inapplicable. The wide variety of nonlinearities creates the challenge to develop well-structured and detailed theories of stability analysis for nonlinear systems.

To address this issue, two main approaches are typically employed. The first one involves approximating the system under study to linear modeling and applying the classic theories of stability. The second approach, known as "Lyapunov theory" [9], looks to draw conclusions about the stability based on the energy of the system. While Lyapunov's theory provides global judgments about stability, it has some weaknesses when it comes to analyzing instability.

The central idea of Lyapunov's theory is to identify a region in which the energy of the system decreases over time, which means that the system heads to an equilibrium state. This region is known as the Domain of Attraction (DA).

The DA is defined as the set of initial states in which the energy of system, mathematically modeled with the Lyapunov Function (LF), decreases over the time so the state heads to an equilibrium state [10]. The size and shape of the DA are strongly influenced by the form and the parameters of the LF. The quadratic form of LF is the most widely used due to its ease of implementation. However, the rational form of LF can

provide a larger DA, which is why the estimation of the DA via rational LFs has a particular interest.

In this context, the main question is: How to estimate accurataly the largest DA of a nonlinear system related to a given LF despite its form.

The existing methods of estimating the DA have several limitations. These include a lack of flexibility in handling various nonlinearities and different forms of LF, as well as inaccuracies where the estimated DA contains failure zones. Additionally, these methods involve a high level of complexity.

The principal target of this work is to develop a method that can estimate the DA from a given LF ensuring the following highlights:

- The estimated DA rising from the given LF is maximized.
- There are no failure sets in the estimated DA.
- The developed method is flexible towards diverse types of nonlinearity and LF's forms.
- The implemented algorithm presents performant convergence conditions nad a low level of complexity.

This paper is organized as follows: after the introduction, there comes Section II, the related works that discuss the estimation of the DA, and the historical steps of the Sine-Cosine Algorithm. Section III presents some generalities on estimating the DA using the Lyapunov theory. Section IV presents the Sampling method to estimate the DA. The main theoretical results are presented in Section V: proposed Sine-Cosine Algorithm for state assessment. Section VI is booked to the simulations and comparative studies. Sections VII and VIII present respectively the discussions of the method and the main conclusions besides the suggested future works.

II. Related Work

This section mentions some works that aim at the DA estimation problem and the use of optimization problems in this context. It presents as well a brief literature review on the Adaptive Sine-Cosine Algorithm ASCA, which has an important role in this contribution. Over the last three decades, researchers have tried to develop an efficient method for estimating the DA. Some of these works are based on the Linear Matrix Inequality (LMI) computation [11], [12]. The works presented in [13], [14] show approaches to estimating the DA of polynomial systems via LMI solving and quadratic LFs. The works in [15], [16] take the benefits of [14] to select the best parameters of the LF that give the largest DA using metaheuristics. Rational Lyapunov functions and LMIs are used to estimate the DA of polynomial systems [17], [18]. The work proposed in [19] presents a method to estimate the DA of non-polynomial systems through LMIs. Other approaches estimate the DA by a mechanism of sampling, setting the system in random initial states, and evaluating the Lyapunov stability conditions. One of the most famous methods has been proposed in [20]. The power of [20] manifests in its ability to deal with polynomials and non-polynomial systems as well as its availability towards the different forms of LF. This method has been the fundamental method ameliorated in [21]. However, it exhibits a weakness in precision: there are some failure sets in the estimated DA. The work proposed in [21] takes the flexibility from the sampling method presented in [20], and replaces the random mechanism with oriented research using the Chaoti-Krill Herd (CKH) optimization heuristic method [22], aiming to compensate for the weakness of [20]. Similarly to [21], the current work is based on an optimization heuristic and sampling mechanism to concept an accurate estimation of the DA. The selection of ASCA is due to the thought that it provides better conditions of convergence thanks to its interior mechanisms. The ASCA is a modified version of the Sine-Cosine Algorithm [23]. It has been born in 2016. It has demonstrated superior performance compared to other metaheuristic optimization algorithms like Particle Swarm Optimization (PSO) [24], Genetic Algorithm (GA) [25], and Dragonfly Algorithm (DfA) [26]. However, it suffers from convergence accuracy issues and a high risk of falling into local optimum. According to the no free lunch theorem (NFL) [27], there is no one-size-fits-all algorithm that can be applied to all optimization problems, which has motivated researchers in the field of metaheuristic algorithms to develop new versions of existing algorithms to improve their performance. This paper emphasizes the benefits of the Adaptive Sine-Cosine Algorithm (ASCA) [28], which features an interesting transition between the exploration of the research universe and the exploitation of results through Chaotic Local Search. Table I shows a general qualitative comparison between methods of estimating the DA.

III. Estimation of the DA using Lyapunov Theory

The Lyapunov theory is a powerful method for ensuring the stability of nonlinear systems within the DA. In light of the fundamental principles of nonlinear system stability, the objective of this section is to approximate the DA using a predetermined LF.

Let us observe the following dynamical autonomous system:

$$\frac{dx}{dt} = f(x),\ x \in \partial \subseteq R^n; \quad x_0 = x(t_0) \tag{1}$$

In the context of the described system, x represents the state vector, ∂ denotes the state space, and $f : \partial \to R^n$ is the system's dynamic. The initial conditions of the state are given by $x_0 = x(t_0)$.

If x_eq is a stable equilibrium state of the closed-loop system and $x(t, x_0)$ denotes the solution of (1) at time t with respect to the initial condition, the region of stability of the system described by (1) is:

$$\theta = \left\{ x_0 \in \partial : \lim_{t \to \infty} x(t, x_0) = x_{eq} \right\} \tag{2}$$

In literature, a sophisticated analytical technique is employed for the estimation of the DA. This methodology is grounded in the principles of Lyapunov stability theory and is subsequently executed as follows [29], [30].

Theorem III.1. *[29].*

TABLE I. GENERAL QUALITATIVE COMPARISON

	Accuracy	Complexity performance	Flexibility	Elapsed time performance	Convergence condition
[14]	High	Average	Low	Average	High
[17]	High	Average	Low	Average	High
[18]	High	Low	Low	Average	High
[19]	High	Low	Low	Average	High
[20]	Low	High	High	High	Average
[21]	High	High	High	Average	Average
Current work	High	High	High	High	High

A closed set $S \subset R^n$, where the origin of system (1) *is its equilibrium, can conclude an approximation of the DA for this origin if:*

- *S is an invariant set for the system* (1)*;*
- *A candidate LF $V(x)$, positive definite, such that its derivative $V(x)$ is negative definite within the set S can be found.*

If the equilibrium state x_{eq} is shifted from the origin of the system (1), a substitution can be made by introducing $w = x - x^*_{eq}$, where x^*_{eq} is the nonzero equilibrium. This transformation can be carried out without any loss of generality and allows for the analysis of the system to be centered on the equilibrium state [31]. The conditions cited in Theorem III.1 guarantee that the set S is certainly included in the absolute DA. The selection of an appropriate candidate LF is not an easy task. As well, the approximation of the DA is sensitive to the shape of the level sets related to the chosen LF. A proposed procedure is detailed in [32] to find a performant LF, where algorithms based on the gradient search are implemented in order to compute a performant candidate LF. Furthermore, the use of composite polynomial and rational forms of LF instead of quadratic forms could lead to better approximations thanks to their rich representation power [33]. Quadratic LFs are quite conservative since they restrict the estimates to ellipsoids [34].

The sublevel set $\Omega(r)$ of $V(x)$ could be defined as follows:

$$\Omega(r) = \{x \in \partial : V(x) \leq r\} \tag{3}$$

If $V(x)$ is quadratic, it can be represented as:

$$V(x) = x^T P x \tag{4}$$

where P is a symmetric matrix in $R^{n\times n}$.

Based on Theorem III.1, every sublevel set $\Omega(r)$ of a candidate LF satisfying the locally asymptotic stability of x_{eq}, could be an estimating of the DA with respect to the time derivative of $V(r)$ is negative for every state included in $\Omega(r)$. Since the largest sublevel set provides an estimation with better accuracy of the DA, the DA approximation could be converted to estimate the largest sublevel set of a chosen LF [35]. In order to find the largest estimated DA, one has to find the maximum value $r \in R$ for $\Omega(r)$ satisfying the conditions of Theorem III.1.

Theorem III.2. *[35]. The invariant set $\Omega(r^*)$, sublevel set of $V(x)$, is the largest estimate of the DA for the origin of system* (1) *if:*

$$\begin{cases} r^* = \max r \\ st \quad \Omega(r) \subseteq \Psi(x) \\ \Psi(x) = \{0\} \cup \left\{x \in R^n : \dot{V}(x) < 0\right\} \end{cases} \tag{5}$$

This problem can be presented as an optimization problem that can be solved by calling the Sum Of Square programming, methods applying both simulation and Sum Of Square programming, and methods based on the theory of moments. However, these approaches are restricted to polynomial systems and LFs.

The next section presents an alternative method based on taking random samples and testing the conditions of Lyapunov stability, in order to attend an estimation of the DA.

IV. SAMPLING METHOD TO ESTIMATE THE DA

This sampling method has the same aim as the Lyapunov-based optimization methods: approximating the DA by finding the largest set from a candidate LF. The principle of this procedure is to check the conditions of Theorem III.1 on a given LF such that the state x_i is chosen randomly, then eliminate the level sets relative to x_i with positive derivative of LF. The LF impacts directly the shape and the volume of the DA: for example, a quadratic form of LF provides an ellipsoid shape of the DA. Thus invites researchers to concept more sophisticated types of LF to estimate a DA that covers the majority of the stability region. This paper also puts a light on the LFs with a rational form.

The rational LF $V(x)$ has the following form:

$$V(x) = \frac{N(x)}{D(x)} = \frac{\sum_{s=2}^{k} R_s(x)}{1 + \sum_{s=1}^{k-2} Q_s(x)} \tag{6}$$

where $R_s(x)$ and $Q_s(x)$ are homogeneous polynomials of degree s. The sampling method to estimate the DA is based on a random sampling of states, checking the conditions of Theorem III.1, and determining the attractiveness radius r.

A. DA Estimation with Sampling Method [20]

This method aims to maximize the value of r in (5). As a first step, x_i is chosen randomly within ∂. The conditions of Theorem III.1 are checked for $V(x_i)$ and $\dot{V}(x_i)$. Let $\bar{r}^*$ and $\underline{r}^*$ be respectively the upper and the lower bound of r^*. The combination of $\bar{r}^*$ and $\underline{r}^*$ offers an accurate prediction for the DA related to $V(x)$. At the start of the mechanism, $\bar{r}^*$ and $\underline{r}^*$ are initialized respectively to ∞ and 0. If $\dot{V}(x_i) < 0$ and $V(x_i)$ is between $\bar{r}^*$ and $\underline{r}^*$, then the value of $\underline{r}^*$ is updated to $\underline{r}^* = V(x_i)$.

Otherwise, in the case when $\dot{V}(x_i) \geq 0$ and $V(x_i) < \bar{r}^*$, then $\bar{r}^*$ takes the value of $V(x_i)$. With proceeding with the algorithm, after a sufficient number of samples, r^* increases, but not obligatorily monotonically. It converges, eventually, to an estimate r^*. As a result, the largest sublevel set $\omega(r^*)$ is determined. Likewise, the lower bound $\underline{r}^*$ increases to converge finally to r^*. When all conditions of Theorem III.1 are "checked true" for a state x_i, considering the value of $V(x_i)$ as a possible estimate for r^*, it is stored then in an array. The usefulness of this array is to guarantee the obedience of the approximated DA found by $\underline{r}^*$ to the conditions of Theorem III.1. Storing the results in an array provides tighter estimates. This array, denoted ϵ, has to be initialized null, its length of in the worst is the number of samples $n_{samples}$. When $\dot{V}(x_i)$ and $V(x_i) < \bar{r}^*$, $V(x_i)$ is stored in ϵ as $\tau(V(x_i))$ is a potential estimation of the DA. When $\dot{V}(x_i) \geq 0$ and $V(x_i) < \bar{r}^*$, if $\underline{r}^* \geq \bar{r}^*$ then the algorithm has to update the lower bound $\underline{r}^*$ among the values stored in the array ϵ. To ensure the no-failure of convergence, the algorithm chooses the maximum value of $\underline{r}^*$ from ϵ respecting that $\underline{r}^* \geq \bar{r}^*$. The selection of a previously stored lower bound has to satisfy the condition $\dot{V} < 0$for the sublevel set $\omega(r^*)$. In the worst case, $\underline{r}^* = 0$.

Algorithm 1 Sampling Method for Estimating the DA

```
Define: V(x),its derivative and n_samples
Initializing r̂* = ∞
for i going from 1 to n_samples do
    Generate a random state x_i within the state space ∂
    if V̇(x_i) < 0 et V(x_i) < r̂* then
        Store V(x_i) in ε
        if V(x_i) > r*_ then
            update r*_ with r*_ = V(x_i)
        end if
    else if V̇(x_i) ≥ 0 then
        if V(x_i) < r̄* then
            r̄* = V(x_i)
            if r*_ ≥ r̄* then
                r*_ = arg max{r ∈ ε | r < r̄*}
            end if
        end if
    end if
end for
Return r*_
```

To have a more accurate estimation the random mechanism is replaced with an optimization technique that looks for maximizing the attractiveness radius r^*, based on assessment of the state x.

V. Proposed Sine-Cosine Algorithm for State Assessment

The objective of this section is to find the most distant initial state from the origin with respect to the conditions of Theorem III.1. Which means maximizing the DA's radius r^*. This task needs the Sine-Cosine Algorithm to be achieved [23].

A. Sine-Cosine Algorithm (SCA) [23]

The mechanism of the Sine-Cosine Algorithm starts from a set of random generation of solutions. The updating's formula makes the algorithm converge to an "accepted global" optimal solution continuously after a large exploration all over the research universe, then an exploitation stage in a tighter region in which the optimum is placed. Initially, the algorithm generates a population of decision variables (initial state's vector x_0) with random positions, it calculates then the fitness of each position (radius r), and stores the position of the optimum, by proceeding iterations, the position is updated as follows:

The updating's function of the position X related to the agent i is determined through the value of the random term b_4 distributed on $[0, 1]$.

$$X_i^{k+1} = \begin{cases} X_i^k + b_1 \sin b_2 \left| b_3 P_i^k - X_i^k \right|, b_4 < 5 \\ X_i^k + b_1 \cos b_2 \left| b_3 P_i^k - X_i^k \right|, b_4 \geq 5 \end{cases} \quad (7)$$

where k is the actual iteration number, X_i^k represents the i^th agent position at iteration k, P_i^k represents the i^th agent of the best population after the k^th iteration, and the usefulness of b_1 is the generation of a linear decreasing phenomena, it can be modeled as follow:

$$b_1 = a - k\frac{a}{T} \quad (8)$$

where a is a constant (chosen equal to 2 in most cases), T presents the maximum iterations bound, b_2 and b_3 are random scalars respectively in the ranges $[0, 2\pi]$ and $[-2, 2]$.

The new computed solution is evaluated by its fitness function and compared with the actual optimum, if a better solution is obtained, the optimal solution will be updated. These tasks will be repeated for all the iterations and for every agent of the population. Algorithm 2 presents the pseudo-code of SCA.

B. Adaptive Sine-Cosine Algorithm (ASCA) [28]

The main parameters of the original SCA are b_1, b_2, b_3, and b_4, mentioned in the previous paragraph. When ($[b_1 \sin b_2$) or ($b_1 \cos b_2$ is in $[-1, 1]$, the algorithm has already attained the local exploitation phase. If it is outside, then it is a global search stage. The parameters b_1 and b_2 influence the value of the updated population X. The parameter b_1 has a more significant impact on the convergence to the local stage. In the original SCA, b_1 is calculated using Eq. (8) which is linearly decreasing with iterations. However, a linear decreasing convergence may affect the ultimate search performance of the

Algorithm 2 Sine Cosine Algorithm (SCA)

Initialize: N (population size), dim (problem dimension), a (control parameter), and T (maximum iteration number).
Initialize the actual iteration number k at 0.
Initialize randomly the population X.
while $k \leq T$ **do**
  for $i = 1$ to N **do**
    for $j = 1$ to dim **do**
      Evaluate the solution by calculating the fitness of X.
      Record the optimal individual X_{best}.
      Recalculate b_1 by equation (8).
      Update b_2, b_3, b_4.
      if $b_4 < 0.5$ **then**
        Update the population X by equation (7) (sine part).
      else
        Update the population X by equation (7) (cosine part).
      end if
      Evaluate the solution by calculating the fitness of the updated population X.
      Update X_{best}.
    end for
  end for
  $k = k + 1$.
end while
Return the best solution.

SCA: the attacked objective function is always complicated, nonlinear, and non-convex and it may not be continuous. Therefore, the parameter b_1 has to be represented differently. In this aim, and to ameliorate the SCA computing power, the parameter b_1 has a form able to balance the phase of exploration and local intensification stage of the SCA. The new parameter b_1, called adaptive, has to be reduced quickly in earlier iterations of the algorithm to move quickly to the exploitation stage. Therefore, the value of b_1 has to be larger in the early iterations to guarantee a better exploration in the search universe and then to move to the local intensification phase with a high decreasing rate. Therefore, the proposed adaptive b_1 has the form shown in the following formula:

$$b_1 = 4\left(1 - \frac{k}{T}\right)\left(1 - 2^{\left(\left(k/T\right)-1\right)}\right) \tag{9}$$

such that T represents the maximum number of iterations and k is the actual iteration.

The new formula of b_1 represented by Eq. (9), by the negative exponential term, is decreasing at a high rate at the beginning of the algorithm progress and this rate becomes lower in the end. Fig. 1 shows the difference between b_1 in Eq. (8) and (9), knowing that the solid line represents b_1 of SCA, and the dashed line represents b_1 of ASCA on 100 iterations.

Fig. 2 shows a comparison between the decreasing pattern for the range of sine and cosine in SCA and ASCA on 100 iterations.

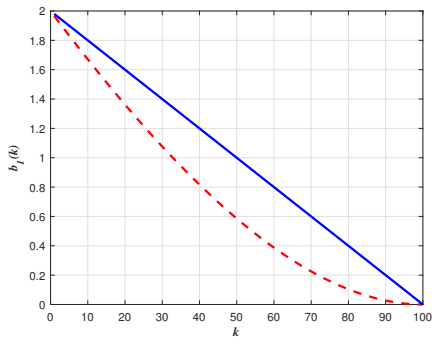

Fig. 1. Comparison between b_1 with Eq. (8) and (9).

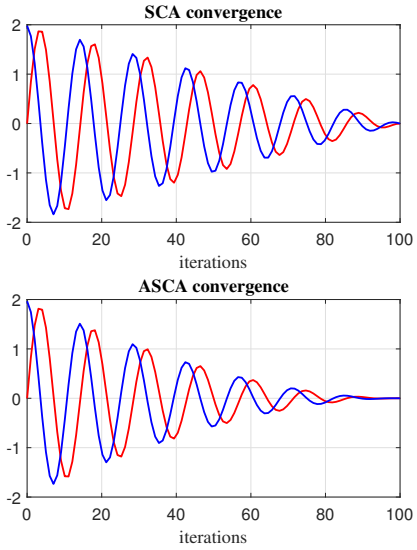

Fig. 2. Decreasing pattern for a range of sine and cosine in SCA and ASCA.

Fig. 2 presents a faster convergence of ASCA than that of SCA, which provides more iterations for local exploitation.

C. Chaotic Local Search (CLS) [36], [37]

Chaotic phenomenon is one of the most interesting figured phenomena. It has an arbitrary, disorganized behavior with a complicated structure. Despite that it looks disorganized, the chaotic phenomena have two principal characteristics: "randomness" and "regularity". The chaos system can conserve the characteristic of randomness thanks to the random update process of the SCA, it explores the totality of search space as much as possible. The Chaotic Local Search (CLS) searches in the neighborhood of the optimum and generates new random solutions without repetition. Since population diversity decreases in the second half proceeding of SCA, CLS can be used to improve search space exploration and local exploitation capacity at the same time [27], [28]. In literature, several kinds of chaotic systems are figured. The chosen chaotic system of this paper is a common logistic map shown as follows:

$$y_{k+1} = \rho y_k (1 - y_k) \tag{10}$$

where k represents the iteration number and ρ represents the control parameter. When ρ and y_0 are selected as $\rho = 4$ and $y_0 \notin \{0.25, 0.5, 0.75, 1\}$, the Eq. (10) is a chaotic system.

The local search (LS) is useful for searching within a tight region. The search made with LS in the neighborhood of the actual optimal solution may lead to a new better optimum. The CLS adds the chaotic aspect to the LS to avoid local optimization. It can help the algorithm avoid premature convergence due to the "randomness" of a chaotic system. The local search for chaos is shown in the following equation:

$$Loc = (1 - \lambda)\, X_{best} + \lambda\,(min + y_k\,(max - min)) \quad (11)$$

where Loc is the location generated through the CLS, X_{best} is the actual optimum, min and max are respectively the lower and upper bounds of the search universe, y_k is the chaotic sequence shown in (10), and λ is found from the following statement:

$$\lambda = \frac{(T - k + 1)}{T} \quad (12)$$

where T is the upper iteration limit, and k is the current iteration.

Eq. (10) produces a chaotic sequence following the CLS in the $[0, 1]$. For every independent execution of (10), y_k is initialized randomly. The chaotic value y_k produced with the logistical map with 100 runs and $y_0 = 0.001$ is shown in Fig. 3.

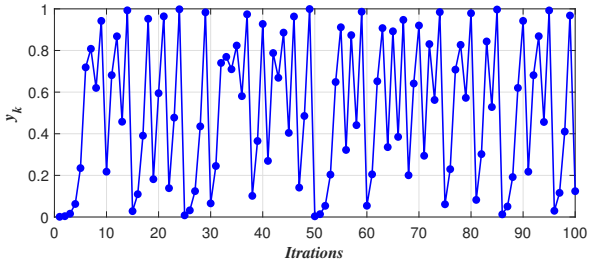

Fig. 3. Chaotic sequence of y_k on 100 iterations.

Algorithm 3 recapitulates the principle and the different steps of the ASCA.

D. Applying ASCA to Optimize the State

This section has opted to combine the sampling method with the Adaptive Sine-Cosine Algorithm, it is an optimization problem where r is the cost to maximize and the state x is the decision's variable. Algorithms 4, 5 and the flowchart in Fig. 4 explain how to apply ASCA to find the best state corresponding to the maximum estimated r rising from the given LF.

Algorithm 3 Adaptive Sine-Cosine Algorithm

Initialize: N (population size), dim (problem dimension), and T (maximum iteration number).
Initialize the actual iteration number k at 0.
Initialize randomly the population X.
while $k \leq T$ **do**
  for $i = 1$ to N **do**
    for $j = 1$ to dim **do**
      Evaluate the solution by calculating the fitness of X.
      Record the optimal individual X_{best}.
      Recalculate b_1 by equation (9).
      Update b_2, b_3, b_4.
      if $b_4 < 0.5$ **then**
        Update the population X by equation (7) (sine part).
      else
        Update the population X by equation (7) (cosine part).
      end if
      Evaluate the solution by calculating the fitness of X.
      Update X_{best}.
      Calculate λ by equation (12).
      Generate the chaotic sequence by equation (10).
      Substitute X_{best} into equation (11) to generate the new individuals Loc.
      Evaluate Loc by calculating its fitness and comparing it with X_{best}.
      if Loc is better than X_{best} **then**
        X_{best} takes the value of Loc.
      end if
    end for
  end for
  $k = k + 1$.
end while
Return X_{best}.

Algorithm 4 Objective ***r***

if $\dot{V}(X) < 0$ **and** $V(X) < \hat{r}^*$ **then**
  Store $V(X)$ in ε
  if $V(X) > \underline{r}^*$ **then**
    Update $\underline{r}^*$ with $\underline{r}^* = V(X)$
  end if
else if $\dot{V}(X) \geq 0$ **then**
  if $V(X) < \bar{r}^*$ **then**
    $\bar{r}^* = V(X)$
    if $\underline{r}^* \geq \bar{r}^*$ **then**
      $\underline{r}^* = \arg\max\{r \in \varepsilon \mid r < \bar{r}^*\}$
    end if
  end if
end if

Algorithm 5 Applying Adaptive Sine-Cosine Algorithm on Sampling with Memory Method

Initialize: N (population size), dim (problem dimension), a (control parameter), and T (maximum iteration number).
Initialize the actual iteration number k at 0.
Initialize randomly the population X.
while $k \leq T$ **do**
  for $i = 1$ to N **do**
    for $j = 1$ to dim **do**
      Evaluate the solution by calculating the fitness (**Objective *r***) of X.
      Record the optimal individual X_{best}.
      Recalculate b_1 by equation (9).
      Update b_2, b_3, b_4.
      if $b_4 < 0.5$ **then**
        Update the population X by equation (7) (sine part).
      else
        Update the population X by equation (7) (cosine part).
      end if
      Evaluate the solution by calculating the fitness (**Objective *r***) of X.
      Update X_{best}.
      Calculate λ by equation (12).
      Generate the chaotic sequence by equation (10).
      Substitute X_{best} into equation (11) to generate the new individuals Loc.
      Evaluate Loc by calculating its fitness (**Objective *r***) and comparing it with X_{best}.
      if Loc is better than X_{best} **then**
        X_{best} takes the value of Loc.
      end if
    end for
  end for
  $k = k + 1$.
end while
Return X_{best}.

VI. Simulations

The objective of this work is to find the farthest initial conditions x_0 from which the system converges to the equilibrium point. This objective is achieved by maximizing the radius r. In this section, there are some two-order and three-order examples illustrating our method on which we applied the ASCA to find the optimal state x maximizing the radius r. The parameter values used in ASCA are: 100 search agents for 100 iterations for all examples.

Example 1 The following expression represents the state space dynamical medialization of the Van Der Pol oscillator:

$$\begin{cases} \frac{dx_1}{dt} = -x_2 \\ \frac{dx_2}{dt} = x_1 - x_2 + x_1^2 x_2 \end{cases} \tag{13}$$

This modal fits with a simple pendulum with non-linear damping where x_1 represents the angular position θ and x_2 is representing the angular velocity $\dot{\theta}$.

The Van Der Pol oscillator modeling becomes:

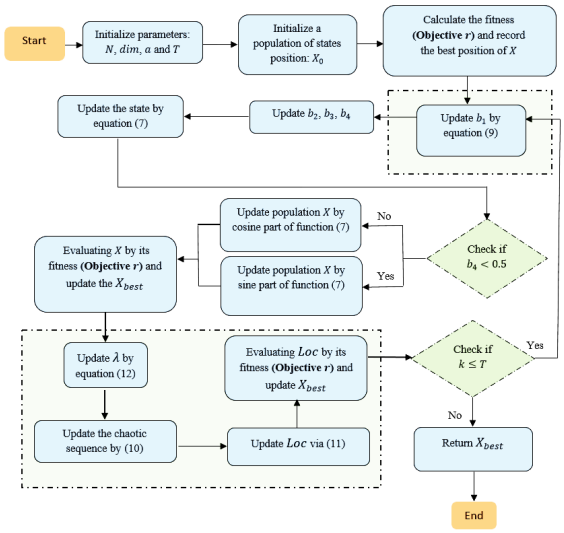

Fig. 4. Flowchart of sampling method with ASCA.

$$\begin{cases} \frac{dx_1}{dt} = -\dot{\theta} \\ \frac{d\dot{\theta}}{dt} = \theta - \dot{\theta} + \theta^2 \dot{\theta} \end{cases} \tag{14}$$

The select of the LF is one of the most interesting issues in the realms of control engineering. Based on a theoretical analysis, some approaches are developed to synthetize the LF. In this context, we find the method of LaSalle [38], method of Zubov [39], etc. Some other methods based on an iterative test are adopted [40]. One of the most popular approaches admitted to synthetize a candidate LF is the linearization around the equilibrium point.

The Jacobean linearization of the system (13) around the origin $\begin{bmatrix} 0 \\ 0 \end{bmatrix}$ is computed with the following formula:

$$J = \frac{\partial f}{\partial x}\bigg|_{x=[0,0]^T} = A_L = \begin{bmatrix} 0 & -1 \\ 1 & -1 \end{bmatrix} \tag{15}$$

To identify the parameters of $V(x)$ it is sufficient to find the matrix P positive definite by solving the following equation:

$$A_L^T P + P A_L = Q \tag{16}$$

where Q is a symmetric matrix that has to be negative definite.

To proceed, it is supposed for Q to be as follows:

$$Q = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} \tag{17}$$

The computing of P using (16) gives:

$$P = \begin{bmatrix} 1.5 & -0.5 \\ -0.5 & 1 \end{bmatrix} \tag{18}$$

As a result:

$$V(x) = 1.5x_1^2 - x_1x_2 + x_2^2 \tag{19}$$

In order to validate the robustness of the convergence, the Monti-Carlo statistic study is established. The algorithm ASCA is applied 100 times, the standard deviation σ, the variance σ^2 and the mean value μ are calculated. Table II shows the values of each term.

TABLE II. MONTI-CARLO STATISTICAL STUDY

σ	σ^2	μ
0.0027	$7.4095e-06$	2.3047

The values mentioned above present a high robustness of convergence of the algorithm with a law standard deviation ($\simeq 0.3\%$).

Fig. 5 presents the distribution of the optimized values obtained in 100 reprises of ASCA on the Van Der Pol system.

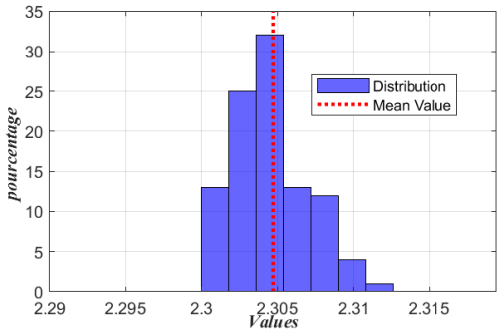

Fig. 5. Distribution of the optimized values obtained in 100 reprises of ASCA.

The distribution of the optimized values is centered on the value of $r = 2.3047$ (about 32% of the trials).

The result of applying ASCA on this example is the following:

$$X = \begin{bmatrix} -0.8569 \\ 0.7492 \end{bmatrix}$$
$$r = 2.3047$$

with an elapsed time of $0.365ms$.

In Fig. 6, there is a representation of the DA optimized with ASCA, where the solid blue line is representing the LF $V(x)$, the dashed line represents its derivative $\dot{V}(x)$, and the solid red oriented line shows the state trajectory beginning from the initial state X found with the ASCA.

As it is shown in the zoomed part of the Fig. 6, there is no states in the domain with a positive derivative of the LF (curves are not secant), so the result is admitted correct.

The Fig. 7 shows the evolution of state in the time, where the red and the blue lines present respectively the evolution of $x_1(t)$ and $x_2(t)$.

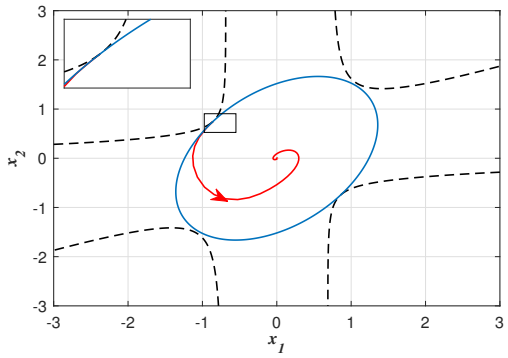

Fig. 6. Representation of LF $V(x)$ of example 1 and its derivative.

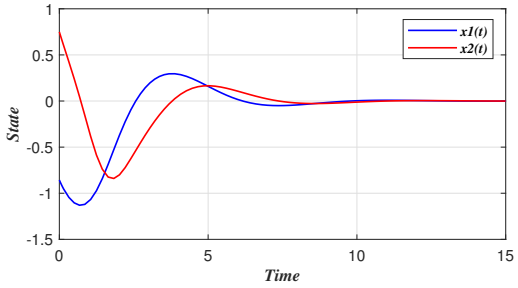

Fig. 7. Representation of state's evolution $x_1(t)$ and $x_2(t)$.

As it is shown in Fig. 7, $x(t)$ clearly attain the equilibrium state $\begin{bmatrix} 0 \\ 0 \end{bmatrix}$. As a result, the convergence is guaranteed.

Fig. 8 presents a comparison of the results of applying ASCA to maximize the radius r with the apply of SCA and CKH

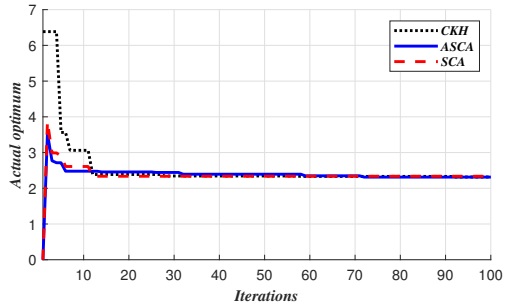

Fig. 8. Comparison of convergences dynamics.

The red dashed line represents the convergence of SCA, the blue solid line corresponds to the convergence of ASCA and the black dotted represents the convergence dynamic of CKH. As it is clearly shown, ASCA is the first algorithm that moves from the exploration to the local search.

Example 2 Let us see the following system:

$$\begin{cases} \frac{dx_1}{dt} = -2x_1 + x_1x_2 \\ \frac{dx_2}{dt} = -x_2 + x_1x_2 \end{cases}$$

The LF corresponding to this system is the following:

$$V(x) = \|x\|^2$$

Applying the optimization metaheuristic ASCA with the conditions declared above gives the following results:

$$X = \begin{bmatrix} 1.2194 \\ 1.6156 \end{bmatrix}$$
$$r = 4.0971$$

with an elapsed time of $0.205ms$

The Fig. 9 represents the DA optimized with ASCA where the solid line is representing $V(x)$ and the dashed line represents its derivative:

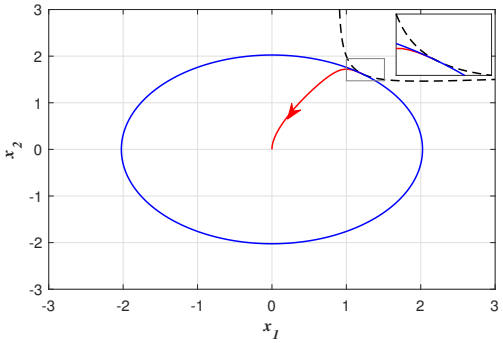

Fig. 9. Representation of LF $V(x)$ of example 2 and its derivative.

As it is shown, there is no states in the domain with a positive LF's derivative (curves are not secant) so the result is admitted correct.

Example 3 Let us see the following system:

$$\begin{cases} \frac{dx_1}{dt} = -\frac{1}{4}x_1 + \ln(1 + x_2) \\ \frac{dx_2}{dt} = -\frac{3}{8}x_1 - \frac{1}{5}x_1x_2 + \left(\frac{1}{8}x_1 - x_2\right)\cos(x_1) \end{cases}$$

The LF corresponding to this system is the following:

$$V(x) = \|x\|^2$$

The results of applying ASCA on this example are the following:

$$X = \begin{bmatrix} -0.4446 \\ -0.2726 \end{bmatrix}$$
$$r = 0.2740$$

with an elapsed time of $0.372ms$.

In Fig. 10, there is a representation of the DA optimized with ASCA, where the solid line is representing $V(x)$ and the dashed line represents its derivative:

Example 4 Let us observe the following system:

$$\begin{cases} \frac{dx_1}{dt} = x_2 \\ \frac{dx_2}{dt} = -0.2x_2 + 0.81\sin(x_1)\cos(x_1) - \sin(x_1) \end{cases}$$

We take the LF as follows:

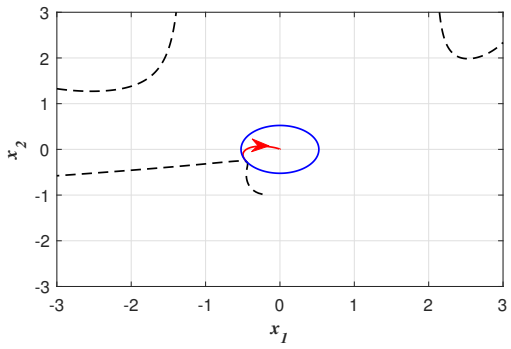

Fig. 10. Representation of LF $V(x)$ of example 3 and its derivative.

$$V(x) = x_1^2 + x_1x_2 + 4x_2^2$$

The results of applying ASCA on this example are the following:

$$X = \begin{bmatrix} -0.7409 \\ 0.3077 \end{bmatrix}$$
$$r = 0.6997$$

with an elapsed time of $0.340ms$.

In Fig. 11, there is a representation of the DA optimized with ASCA, where the solid line is representing $V(x)$ and the dashed line represents its derivative:

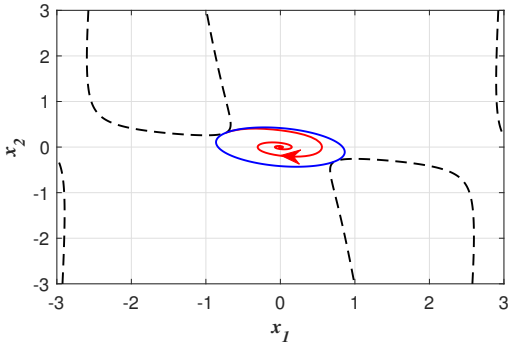

Fig. 11. Representation of LF $V(x)$ of example 4 and its derivative.

Example 5 Let us observe the following third order system system:

$$\begin{cases} \frac{dx_1}{dt} = -x_1 + x_2x_3^2 \\ \frac{dx_2}{dt} = -x_2 + x_1x_2 \\ \frac{dx_3}{dt} = -x_3 \end{cases}$$

We take the LF as follows:

$$V(x) = x_1^2 + x_2^2 + x_3^2$$

When we applied the ASCA, we found these results:

$$X = \begin{bmatrix} 1.1806 \\ 1.5407 \\ -1.0917 \end{bmatrix}$$
$$r = 4.9594$$

With an elapsed time of $0.155ms$.

In Fig. 12, there is a representation of the DA optimized with ASCA, where the yellow spherical form represents $V(x)$ and the blue surface represents its derivative:

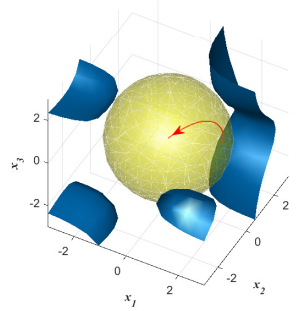

Fig. 12. Representation of LF $V(x)$ of example 5 and its derivative.

Example 6 Let us observe the following third order system system:

$$\begin{cases} \frac{dx_1}{dt} = 1 + x_3 + \frac{1}{8}x_3^2 - \exp(x_1) \\ \frac{dx_2}{dt} = -x_2 - x_3 \\ \frac{dx_2}{dt} = -x_2 - 2x_3 - \frac{1}{2}x_1^2 \end{cases}$$

We take the LF as follows:

$$V(x) = x_1^2 + x_2^2 + x_3^2$$

When we applied the ASCA, we found these results:

$$X = \begin{bmatrix} -1.339 \\ 0.5708 \\ 0.7523 \end{bmatrix}$$
$$r = 2.6865$$

With an elapsed time of $0.166ms$.

In Fig. 13, there is a representation of the DA optimized with ASCA, where the yellow spherical form represents $V(x)$ and the blue surface represents its derivative:

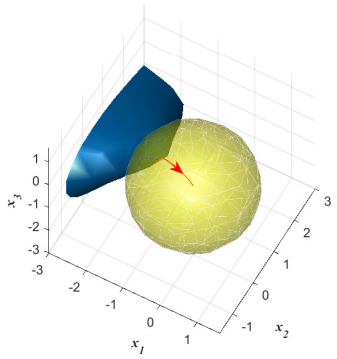

Fig. 13. Representation of LF $V(x)$ of example 6 and its derivative.

Comparison with other heuristic methods

In Table III, we made a comparison between the Adaptive Sine Cosine Algorithm And the Chaotic Krill Herd.

This comparison shows that in most cases, the results of ASCA are better than those of CKH with a shorter elapsed time without loss of precision. This statement means that the method applied ameliorates two factors at the same time: the computing time reserved for the estimation of the DA, and the stability region guaranteed from a given LF.

In the following, some examples with rational LF are shown.

Example 7 [20] Consider the following system:

$$\begin{cases} \frac{dx_1}{dt} = -x_1 + x_2 + 0.5(\exp(x_1) - 1) \\ \frac{dx_2}{dt} = -x_1 - x_2 + x_1x_2 + x_1\cos(x_1) \end{cases}$$

Table IV presents the rational LF.

After applying the approach to this example, we find the following result:

$$X = \begin{bmatrix} 1.3010 \\ -0.6179 \end{bmatrix}$$
$$r = 1.2252$$

The result obtained in [20] is $r = 1.2251$. We can see that the DA obtained in this paper is larger than the DA obtained in [20].

Fig. 14 shows the DA obtained by using a rational LF on example 7, where the LF and its time derivative are represented respectively with the solid blue line and the dashed black line. The red solid-oriented line presents the trajectory of the system initialized in the optimal state found with the ASCA.

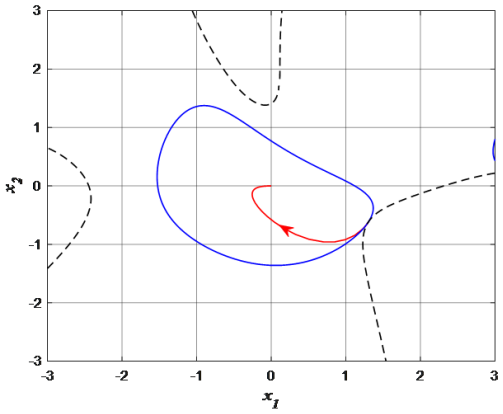

Fig. 14. Representation of LF $V(x)$ of example 7 and its derivative.

The following example cites a comparison between results obtained with this approach and those in the literature. As well, a comparison between DAs related to polynomial and rational LFs is presented.

Example 8 [18] Consider the following system:

$$\begin{cases} \frac{dx_1}{dt} = x_2 \\ \frac{dx_2}{dt} = -3x_1 - 2x_2 + x_1^2 \end{cases}$$

TABLE III. COMPARISON BETWEEN RESULTS OF ASCA AND CKH

Example	r Optimized with ASCA	r Optimized with CKH	Elapsed time ASCA (ms)	Elapsed time CKH (ms)
1	2.3047	2.3045	0 .365	0.542
2	4.0971	4.0955	0.205	0.350
3	0.2740	0.2737	0.372	0.258
4	0.6997	0.3611	0.340	0.236
5	4.9594	4.969	0.155	0.770
6	2.6865	2.6617	0.166	0.813

TABLE IV. LF OF EXAMPLE 7

$R_s(x)$	$Q_s(x)$
$R_2(x) = x_1^2 + 1.3333x_1x_2$ $+1.1667x_2^2$ $R_3(x) = -0.2272x_1^3$ $-0.1396x_1^2x_2 + 0.3785x_1x_2^2$ $+0.1798x_2^3$ $R_4(x) = 0.0136x_1^4$ $-0.2864x_1^3x_2$ $+0.1918x_1^2x_2^2 - 0.053x_1x_2^3$ $+0.0172x_2^4$	$Q_1(x) = -0.5605x_1 - 0.7255x_2$ $Q_2(x) = 0.3254x_1^2 + 0.0910x_1x_2$ $+0.1015x_2^2$

TABLE V. LFS OF EXAMPLE 8

Polynomial LF	Rational LF
$V(x) = 2x_1^2 + x_1x_2 + x_2^2$	$R_2(x) = 2x_1^2 + x_1x_2 + x_2^2$ $R_4(x) = 6x_1^4 + 7x_1^3x_2 + 7x_1^2x_2^2$ $+3x_1x_2^3 + x_2^4$ $Q_2(x) = x_1^2 + x_2^2$

Table V presents the quadratic and rational LFs.

The results of the application of the algorithm are shown in Table VI and Fig. 15:

TABLE VI. RESULTS OF EXAMPLE 8

Polynomial LF	Rational LF
$X = \begin{bmatrix} -1.8188 \\ 2.1008 \end{bmatrix}$ $r = 7.2085$	$X = \begin{bmatrix} 1.2592 \\ 2.2879 \end{bmatrix}$ $r = 24.1795$

When we observe these results we realize that: The DA related to a rational LF is larger than the DA of a polynomial LF. The approach provides better results than those in [18] ($r = 24.1795$) which approve the high performance of the algorithm.

Fig. 15 shows a comparison between the domains of attraction obtained with polynomial and rational LFs. The red and the blue solid lines represent respectively the rational and the polynomial LF. The red and the blue dashed lines represent respectively the derivatives of rational and polynomial LFs.

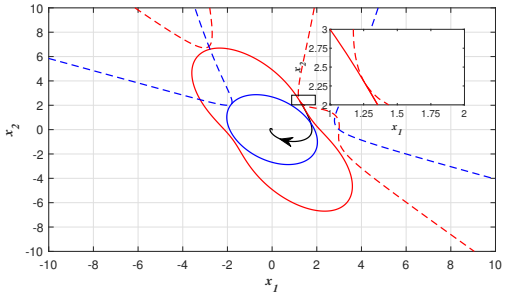

Fig. 15. Comparison of DAs related to polynomial and rational LF of example 8.

VII. DISCUSSION

This section presents a detailed discussion on the comparison between methods. The proposed method shows a flexibility towards diverse forms of nonlinearity and LFs. Unlike the LMI based methods [14], [19], this method does not require any approximation to any conventional form of nonlinearity. Another benefit of the proposed method is mentioned in the Table II. This table shows a more performant convergence dynamic than the Chaoti-Krill Herd method [21].

The Table III gives a recapitulative comparison between the current work and the method proposed in [21]. The estimation with ASCA takes a less amount of time for computing than the CKH method. It gives also a larger estimation of DA without containing failure sets in which the time derivative of the LF is positive.

As this work has huge benefits, it has also some weaknesses. The general aim of estimating the DA is to determine the largest region of stability, which is influenced by the LF selection. This work does not provide a way to select the optimal parameters of LF.

Another weakness of this work is related to the use of heuristic methods. The heuristics in general do not provide a proof that the optimum found is absolutely global, even with the integration of the CLS. It also presents a low performance in the case of real time cascading architectures. As a result, it appears a need of other optimization algorithms providing better qualities of results and respecting the real time constraints.

VIII. CONCLUSION

This paper uses a hybrid technique that combines a sampling and testing method with the ASCA, in order to find the farthest initial state of the DA related to the LF. Besides to the larger DA that the followed approach provided, it proved a high accuracy against the classic sampling method that may include some failure sets. This method achieved two principal goals. It gives an accurate estimation of the DA related to a given LF, and it maximizes this DA by applying the ASCA at the same time.

The hybridization with ASCA proved a high performance in the elapsed time and the results qualities in relation to some other metaheuristic methods (SCA, CKH).

The weaknesses mentioned in the discussion section lead to some ideas of future works. As a perspective, by a non-Lyapunov and inverse modeling method we will try to integrate "deep learning" in order to build a candidate LF providing an optimized stability region with a respect to the real time constraints.

REFERENCES

[1] M. W. Krentel, The complexity of optimization problems, *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, (1986, November) 69-76. http://dx.doi.org/10.1016/0022-0000(88)90039-6

[2] A. K. Hartmann, H. Rieger, *Optimization algorithms in physics*, 2002.

[3] Y. Bengio, Gradient-based optimization of hyperparameters, *Neural computation, 12*(8) (2000), 1889-1900. https://doi.org/10.1162/089976600300015187

[4] E. G. Talbi, Metaheuristics: From Design to Implementation, *John Wiley & Sons google schola*, (2009), 268-308.

[5] X. S. Yang, *Engineering optimization: an introduction with metaheuristic applications*, John Wiley & Sons, 2010.

[6] R. Luo, Z. Wang, Y. Sun, Optimized Luenberger Observer-Based PMSM Sensorless Control by PSO, *Modelling and Simulation in Engineering*, 1(2022), 3328719. https://doi.org/10.1155/2022/3328719

[7] K. Khuwaja, I. C. Tarca, R. C. Tarca, PID controller tuning optimization with genetic algorithms for a quadcopter, *Recent Innovations in Mechatronics*, 5(1) (2018), 1-7. https://doi.org/10.17667/riim.2018.1/11

[8] S. Gupta, V. P. Singh, S. P. Singh, T. Prakash, N. S. Rathore, Elephant herding optimization based PID controller tuning, *International Journal of Advanced Technology and Engineering Exploration*, 3(24) (2016), 194. http://dx.doi.org/10.19101/IJATEE.2016.324005

[9] A. Lamperski, A. D. Ames, Lyapunov theory for Zeno stability, *IEEE Transactions on Automatic Control*, 58(1) (2012), 100-112.

[10] M. Escobar-Bach, R. Maller, I. Van Keilegom, M. Zhao, Estimation of the cure rate for distributions in the Gumbel maximum domain of attraction under insufficient follow-up, *Biometrika*, 109(1) (2022), 243-256. https://doi.org/10.1093/biomet/asac001

[11] S. Boyd, L. El Ghaoui, E. Feron, V. Balakrishnan, *Linear matrix inequalities in system and control theory*, Society for industrial and applied mathematics, 1994.

[12] C. Scherer, S. Weiland, Linear matrix inequalities in control, *Lecture Notes, Dutch Institute for Systems and Control, Delft, The Netherlands*, 3(2) (2000).

[13] B. Tibken, Estimation of the domain of attraction for polynomial systems via LMIs, *Proceedings of the 39th IEEE Conference on Decision and Control (Cat. No. 00CH37187)*, 4 (2000), 3860-3864. https://doi.org/10.1109/CDC.2000.912314

[14] G. Chesi, Computing output feedback controllers to enlarge the domain of attraction in polynomial systems, *IEEE Transactions on Automatic Control*, 49(10) (2004), 1846-1853. https://doi.org/10.1109/TAC.2004.835589

[15] F. Hamidi, H. Jerbi, W. Aggoune, M. Djemai, M. N. Abdelkrim, Enlarging the domain of attraction in nonlinear polynomial systems, *International Journal of Computers Communications & Control*, 8(4) (2013), 538-547. http://dx.doi.org/10.15837/ijccc.2013.4.152

[16] F. Hamidi, M. Aloui, H. Jerbi, M. Kchaou, R. Abbassi, D. Popescu, Chaotic particle swarm optimisation for enlarging the domain of attraction of polynomial nonlinear systems, *Electronics*, 9(10) (2020), 1704. https://doi.org/10.3390/electronics9101704

[17] O. Hachicho, A novel LMI-based optimization algorithm for the guaranteed estimation of the domain of attraction using rational Lyapunov functions, *Journal of the Franklin Institute*, 344(5) (2007), 535-552. https://doi.org/10.1016/j.jfranklin.2006.02.032

[18] G. Chesi, On the estimation and control of the domain of attraction through rational Lyapunov functions, *American Control Conference (ACC)*, (2012), 3322-3327. https://doi.org/10.1109/ACC.2012.6314658

[19] G. Chesi, Estimating the domain of attraction for non-polynomial systems via LMI optimizations, *Automatica*, 45(6) (2009), 1536-1541. https://doi.org/10.1016/j.automatica.2009.02.011

[20] E. Najafi, R. Babuška, G. A. Lopes, A fast sampling method for estimating the domain of attraction, *Nonlinear dynamics*, 86 (2016), 823-834. https://doi.org/10.1007/s11071-016-2926-7

[21] M. Aloui, F. Hamidi, H. Jerbi, M. Omri, D. Popescu, R. Abbassi, A chaotic krill herd optimization algorithm for global numerical estimation of the attraction domain for nonlinear systems, *Mathematics*, 9(15) (2021), 1743. https://doi.org/10.3390/math9151743

[22] G. G. Wang, A. H. Gandomi, A. H. Alavi, D. Gong, A comprehensive review of krill herd algorithm: variants, hybrids and applications, *Artificial Intelligence Review*, 51 (2019), 119-148. https://doi.org/10.1007/s10462-017-9559-1

[23] S. Mirjalili, SCA: a sine cosine algorithm for solving optimization problems, *Knowledge-based systems*, 96 (2016), 120-133. https://doi.org/10.1016/j.knosys.2015.12.022

[24] F. Wang, H. Zhang, A. Zhou, A particle swarm optimization algorithm for mixed-variable optimization problems, *Swarm and Evolutionary Computation*, 60 (2021), 100808. https://doi.org/10.1016/j.swevo.2020.100808

[25] S. Katoch, S. S. Chauhan, V. Kumar, A review on genetic algorithm: past, present, and future, *Multimedia tools and applications*, 80 (2021), 8091-8126. https://doi.org/10.1007/s11042-020-10139-6

[26] M. Alshinwan, L. Abualigah, M. Shehab, M. A. Elaziz, A. M. Khasawneh, H. Alabool, H. A. Hamad, Dragonfly algorithm: a comprehensive survey of its results, variants, and applications, *Multimedia Tools and Applications*, 80 (2021), 14979-15016. https://doi.org/10.1007/s11042-020-10255-3

[27] S. P. Adam, S. A. N. Alexandropoulos, P. M. Pardalos, M. N. Vrahatis, No free lunch theorem: A review, *Approximation and optimization: Algorithms, complexity and applications*, (2019), 57-82. https://doi.org/10.1007/978-3-030-12767-1_5

[28] Y. Ji, J. Tu, H. Zhou, W. Gui, G. Liang, H. Chen, M. Wang, An adaptive chaotic sine cosine algorithm for constrained and unconstrained optimization, *Complexity*, 1 (2020), 6084917. https://doi.org/10.1155/2020/6084917

[29] G. Chesi, Estimating the domain of attraction via union of continuous families of Lyapunov estimates, *Systems & control letters*, 56(4) (2007), 326-333. https://doi.org/10.1016/j.sysconle.2006.10.012

[30] H. K. Khalil, *Nonlinear systems*, 2002.

[31] F. Amato, C. Cosentino, A. Merola, On the region of attraction of nonlinear quadratic systems, *Automatica*, 43(12) (2007), 2119-2123. https://doi.org/10.1016/j.automatica.2007.03.022

[32] G. Chesi, A. Garulli, A. Tesi, A. Vicino, LMI-based computation of optimal quadratic Lyapunov functions for odd polynomial systems, *International Journal of Robust and Nonlinear Control: IFAC-Affiliated Journal*, 15(1) (2005), 35-49. https://doi.org/10.1002/rnc.967

[33] W. Tan, A. Packard, Stability region analysis using polynomial and composite polynomial Lyapunov functions and sum-of-squares programming, *IEEE Transactions on Automatic Control*, 53(2) (2008), 565-571. https://doi.org/10.1109/TAC.2007.914221

[34] A. Tesi, F. Villoresi, R. Genesio, On the stability domain estimation via a quadratic Lyapunov function: convexity and optimality properties for polynomial systems, *IEEE Transactions on Automatic Control*, 41(11) (1996), 1650-1657. https://doi.org/10.1109/9.544002

[35] G. Chesi, Domain of attraction: analysis and control via SOS programming, *Springer Science & Business Media*, 415 (2011).

[36] C. Choi, J. J. Lee, Chaotic local search algorithm, *Artificial Life and Robotics*, 2 (1998), 41-47. https://doi.org/10.1007/BF02471151

[37] S. Gao, Y. Yu, Y. Wang, J. Wang, J. Cheng, M. Zhou, Chaotic local search-based differential evolution algorithms for optimization, *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 51(6) (2019), 3954-3967. https://doi.org/10.1109/TSMC.2019.2956121

[38] J. P. LaSalle, Stability theory for ordinary differential equations, *Journal of Differential equations*, 4(1) (1968), 57-65.

[39] V. I. Zubov, Methods of AM Lyapunov and their application, *US Atomic Energy Commission*, 4439 (1961).

[40] F. Hamidi, M. N. Abdelkrim, J. Houssem, Searching Candidate Lyapunov Function with Threshold Accepting Algorithm, *2011 Third International Conference on Computational Intelligence, Communication Systems and Networks*, (2011), 26-31. https://doi.org/10.1109/CICSyN.2011.19