Article

Evolutionary Search for Polynomial Lyapunov Functions: A Genetic Programming Method for Exponential Stability Certification

Roman Pykhnivskyi *, Anton Ryzhov, Andrii Sobchuk, Yurii Kravchenko

Posted Date: 1 April 2025

doi: 10.20944/preprints202504.0027.v1

Keywords: nonlinear dynamics; stability; Lyapunov functions; numerical methods; genetic programming

Article

Evolutionary Search for Polynomial Lyapunov Functions: A Genetic Programming Method for Exponential Stability Certification

Roman Pykhnivskyi [1,*], Anton Ryzhov [1], Andrii Sobchuk [2] and Yurii Kravchenko [3]

[1] Faculty of Mechanics and Mathematics, Taras Shevchenko National University of Kyiv, 4E Academician Glushkov Avenue, 03127 Kyiv, Ukraine; roman.pykhnivskyi@gmail.com (R.P.); ryzhov@knu.ua (A.R.)

[2] Educational and Scientific Institute of Cybersecurity and Information Protection, State University of Information and Communication Technologies, 7, Solomyanska Str., 03110 Kyiv, Ukraine; anri.sobchuk@gmail.com (A.S.)

[3] Faculty of Information Technology, Taras Shevchenko National University of Kyiv, 24, Bohdana Havrylyshyna Str., 02000 Kyiv, Ukraine; yurii.kravchenko@knu.ua (Y.K.)

* Correspondence: roman.pykhnivskyi@gmail.com;

Abstract: This paper presents a method for constructing polynomial Lyapunov functions to analyze the stability of nonlinear dynamical systems. The approach is based on Genetic Programming, a variant of Genetic Algorithms where the search space is consists hierarchical tree structures. In our formulation, these polynomial functions are represented as binary trees. The Lyapunov conditions for exponential stability are interpreted as a minimax optimization problem, using a carefully designed fitness metric to ensure positivity and dissipation within a chosen domain. The Genetic Algorithm then evolves candidate polynomial trees, minimizing constraint violations and continuously refining stability guarantees. Numerical examples illustrate that this methodology can effectively identifies and optimizes Lyapunov functions for a wide range of systems, indicating a promising direction for automated stability proofs in engineering applications.

Keywords: nonlinear dynamics; stability; Lyapunov functions; numerical methods; genetic programming

MSC: 34D20; 68W50

1. Introduction

Lyapunov functions are fundamental tools for analyzing the stability of nonlinear systems [1–3], based on energy-based reasoning. If a system consistently dissipates energy everywhere except at the origin, it must eventually converge to the equilibrium point. Consequently, a Lyapunov function—serving as an energy-like measure—must be strictly positive definite for all states excluding the origin. This principle extends naturally to general nonlinear systems. Lyapunov's direct method involves constructing a scalar function representing the system's dynamics and analyzing its evolution over time. However, identifying such a function remains a significant challenge. Importantly, failing to find a Lyapunov function does not imply that the system is unstable, adding complexity to the task.

Numerous classical methods for identifying or approximating Lyapunov functions have been proposed, including the variable gradient method, Krasovskii's method, and basis-function linearization; see [1–6] for a comprehensive overview. However, no universal technique exists, and constructing a Lyapunov function is generally as difficult as solving the system itself. In recent decades, advances in computational power have spurred research into numerical approaches, such as sum-of-squares polynomials [7,8], piecewise affine functions [9], formal synthesis via satisfiability modulo theories (SMT) [10], and deep neural networks [11]. An in-depth survey of these methods can be found in [12].

This work introduces a numerical method for synthesizing polynomial Lyapunov functions tailored to exponentially stable dynamical systems, utilizing the Genetic Programming (GP) paradigm [13]. GP is an evolutionary computation technique, an extension of Genetic Algorithms [14], which operates directly on symbolic expressions, enabling the automatic discovery and optimization of mathematical formulas. A key advantage of this approach is its ability to generate interpretable, symbolic representations of Lyapunov functions, facilitating further analysis of system dynamics.

Several prior works have explored related methodologies. Grosman and Lewin [15] applied GP techniques to the analysis of asymptotic stability. McGough et al. [16] utilized Grammatical Evolution, a specialized variant of GP. Feng et al. [17] combined deep neural networks with symbolic regression to synthesize Lyapunov functions; additional insights are provided in [18].

While these methods have demonstrated significant progress, many lack rigorous mathematical guarantees or general-purpose construction procedures. Our contributions address this gap by establishing existence criteria for polynomial Lyapunov functions in exponentially stable systems and developing a robust algorithm for their automated synthesis. The proposed method was validated across multiple benchmark systems, demonstrating its strong performance and high accuracy.

2. Exponential Stability of Dynamical Systems

In this section, we recall several fundamental concepts of exponential stability in dynamical systems. For additional details, we refer the reader to [1–3].

We consider an autonomous system of ordinary differential equations of the form

$$\dot{x}(t) = f\big(x(t)\big) \tag{1}$$

where $f: D \to \mathbb{R}^n$ is a locally Lipschitz continuous map defined on a domain $D \subseteq \mathbb{R}^n$.

We analyzed an exponentially stable equilibrium according to the following definition.

Definition 1. *The equilibrium point $x = 0$ of* (1) *is exponentially stable if there exist positive constants c, k, and λ such that*

$$\|x(t)\| \le k\|x(t_0)\|^{-\lambda(t-t_0)},\ \forall\|x(t_0)\| < c$$

and globally exponentially stable if satisfied for any initial state $x(t_0)$.

The classical Lyapunov theorem states that if one can find a continuously differentiable function $V(x): D \to \mathbb{R}$ that satisfies specified bounding conditions, then the equilibrium at the origin is exponentially stable.

Theorem 2 (Khalil [2])**.** *Let $x = 0$ be an equilibrium point for the nonlinear system*

$$\dot{x} = f(t, x)$$

where $f: [0,\infty) \times D \to \mathbb{R}^n$ is piecewise continuous in t and locally Lipschitz in x on $[0,\infty) \times D$ and $D \subset \mathbb{R}^n$ is a domain that contains the origin $x = 0$. Let $V(x): [0,\infty) \times D \to \mathbb{R}$ be a continuously differentiable function such that

$$k_1\|x\|^{\alpha} \le V(t, x) \le k_2\|x\|^{\alpha},$$

doi:10.20944/preprints202504.0027.v1

$$\frac{\partial V}{\partial t} + \frac{\partial V}{\partial x} f(t,x) \leq -k_3 \|x\|^{\alpha},$$

$\forall t \geq 0$ and $\forall x \in D$, where k_1, k_2, k_3 and α are positive constants. Then, $x = 0$ is exponentially stable. If the assumption holds globally, then $x = 0$ is globally exponentially stable.

For nonlinear systems, stability can also be analyzed via linearization. The following theorem relates exponential stability of the nonlinear system to the stability of its linearization using the Jacobian matrix.

Theorem 3 (Khalil [2]). *Let $x = 0$ be an equilibrium point for the nonlinear system*

$$\dot{x} = f(t,x)$$

where $f\colon [0,\infty) \times D \to \mathbb{R}^n$ is continuously differentiable, $D = \{x \in \mathbb{R}^n \mid \|x\|_2 < r\}$, and the Jacobian matrix $A(t)$ is bounded and Lipschitz on D, uniformly in t.

Let

$$A(t) = \frac{\partial f}{\partial x}(t,x)\bigg|_{x=0}.$$

Then, the origin is an exponentially stable equilibrium point for the nonlinear system if and only if it is an exponentially stable equilibrium point for the linear system

$$\dot{x} = A(t)x.$$

For the autonomous system (1) the stability of $x = 0$ can be determined by the eigenvalues of the constant linearization matrix.

Definition 4. *A squared matrix $A \in \mathbb{R}^{n \times n}$ is called Hurwitz matrix if all its eigenvalues λ_i satisfy*

$$Re(\lambda_i) < 0.$$

Corollary 5. *Let $x = 0$ be an equilibrium point for the autonomous nonlinear system (1), where $f(x)$ is continuously differentiable in a neighborhood of $x = 0$. Let A be a linearization matrix of (1). Then, $x = 0$ is an exponentially stable equilibrium point for the nonlinear system if and only if A is a Hurwitz matrix.*

The following converse theorem presented by Peet [19] guarantees the existence of a polynomial Lyapunov function for an exponentially stable equilibrium under sufficient smoothness and boundedness conditions.

Theorem 6 (Peet [19]**).** *Consider the system defined by equation (1), where $f \in C^{n+2}(\mathbb{R}^n)$. Suppose there exist constants $\mu, \delta, r > 0$ such that*

$$\|x(t)\|_2 \leq \mu \|x(t_0)\|_2 e^{-\delta t}$$

for all $t \geq 0$ and $\|x(t_0)\|_2 \leq r$. Then, there exists a polynomial $v\colon \mathbb{R}^n \to \mathbb{R}$ and constants $k_1, k_2, k_3 > 0$ such that

$$k_1 \|x\|_2^2 \leq v(x) \leq k_2 \|x\|_2^2,$$

$$\nabla v(x)^T f(x) \leq -k_3 \|x\|_2^2,$$

doi:10.20944/preprints202504.0027.v1

for all $\|x(t)\|_2 \leq r$.

3. Genetic Programming Method for Exponential Stability Certification

In this section, we outline our proposed method, which leverages genetic programming to compute a polynomial Lyapunov function for a given dynamical system. We provide a comprehensive explanation of the underlying theoretical principles and describe the construction process of our approach.

3.1. Existence Criterion for a Polynomial Lyapunov Function

We now present a converse Lyapunov theorem, which provides the main criterion for the applicability of the proposed method.

Theorem 7. *Let* $x = 0$ *be an equilibrium for the nonlinear system* (1), *where* $f \in C^{n+2}(D)$ *and* $D = \{x \in \mathbb{R}^n \mid \|x\|_2 < r\}$. *Let linearization matrix A of the system* (1) *be Hurwitz matrix. Then, there exists a polynomial* $v: \mathbb{R}^n \to \mathbb{R}$ *and constants* $k_1, k_2, k_3 > 0$ *such that*

$$k_1\|x\|_2^2 \leq v(x) \leq k_2\|x\|_2^2,$$

$$\nabla v(x)^T f(x) \leq -k_3\|x\|_2^2,$$

for all $x \in D$.

Proof of Theorem 7. By Corollary 5, the Hurwitz condition on the linearization implies that $x = 0$ is an exponentially stable equilibrium for the nonlinear system. Then, by Theorem 6, under the assumed smoothness of f and the exponential stability of the origin, there exists a polynomial Lyapunov function v satisfying the required quadratic bounds and dissipation condition on D. □

3.2. The Lyapunov Fitness

For the remainder of our work, we focus on dynamical systems that satisfy the conditions of Theorem 7, and our objective is to develop methods for constructing a polynomial Lyapunov function V that meets the following criteria for exponential stability on domain $D \subset \mathbb{R}^n$:

- Quadratic Bounds: There exist constants $k_1, k_2 > 0$ such that:

$$k_1\|x\|^2 \leq V(x) \leq k_2\|x\|^2,\ \forall x \in D.$$

- Dissipation Condition: There exist $k_3 > 0$ such that the orbital derivative of V satisfies:

$$\nabla V(x)^T f(x) \leq -k_3\|x\|^2,\ \forall x \in D.$$

Rather than strictly enforcing these conditions, we introduce a fitness measure that quantifies the degree of violation by formulating the search for a Lyapunov function as a minimax optimization problem:

$$inf_{V \in \mathcal{V}} sup_{x \in D}\big(F(V,x)\big), \tag{2}$$

where $\mathcal{V}$ is a space of candidate function (in our case, polynomials on D) and $F(V,x)$ is a fitness functional that penalizes deviations from the desired Lyapunov conditions.

Definition 8. *Let* $\mathcal{V}$ *be a chosen space of continuously differentiable functions* $V(x): D \to \mathbb{R},\ D \subset \mathbb{R}^n$. *Define the Lyapunov fitness functional* $F: \mathcal{V} \times D \to \mathbb{R}$ *by*

$$F(V,x) = F_1(V,x) + F_2(V,x) + F_3(V,x),$$

with the following penalty terms:

1. *Lower Bound Penalty:*

$$F_1(V,x) = tanh(max\{0, \quad k_1\|x\|^2 - V(x)\}).$$

2. *Upper Bound Penalty:*

$$F_2(V,x) = tanh(\ max\{0, \quad V(x) - \ k_2\|x\|^2\}).$$

3. *Dissipation Penalty:*

$$F_3(V,x) = tanh(max\{0,\ \nabla V(x)^T f(x) + \ k_3\|x\|^2\}).$$

Here, k_1, k_2, k_3 *are positive constants,* $f(x)$ *is the vector field of a dynamic system, and the hyperbolic tangent function*

$$tanh(x) = \frac{e^x - \ e^{-x}}{e^x + e^{-x}}$$

serves to smooth and bound the penalty values.

Definition 9. *For a candidate Lyapunov function* V *and the vector field* f*, the Lyapunov fitness* $L_f(V)$ *over domain* D *is defined as*

$$L_{f,D}(V) = \int_D^{\square} F(V,x)dx. \tag{3}$$

This integral measure the total violation of the Lyapunov conditions over the domain D*.*

Definition 10. *Given a probability distribution* ρ *on* D *and a set of sample points* $\{\hat{x}_1, \dots, \hat{x}_N\} \subset D$ *drawn according to* ρ*, the empirical Lyapunov fitness is defined as*

$$L_{f,\rho(D)}(V) = \frac{\mathbf{1}}{\boldsymbol{N}}\sum_{\boldsymbol{i=1}}^{\boldsymbol{N}} F(V,x_i). \tag{4}$$

This discrete approximation converges to $L_f(V)$ *as* $N \to \infty$*.*

It is straightforward to verify that if $\hat{V}$ is a Lyapunov function satisfying the exponential stability conditions on D, then $\hat{V}$ is a global minimizer of both (3) and (4), as well as a solution to the minimax problem (2).

3.3. Lyapunov Function Representation

In our formulation, a candidate Lyapunov function V is represented as a binary tree, with the following structure:

- The leaves of a tree (also known as terminal nodes) are drawn from a predefined set:

$$T = \{x_1, \dots, x_n, a_1, \dots, a_n\}, \tag{5}$$

where $\{x_1, \dots, x_n\}$ denote the state variables and $\{a_1, \dots, a_n\} \subset \mathbb{R}$ are constant coefficients.

- The internal nodes (branches or nonterminal nodes) represent binary operations. In our implementation, the set of functions is

$$H = \{h, g\}, \tag{6}$$

where $h(a, b) = a + b$, and $g(a, b) = a \times b$.

Although the sets T and H can be expanded, they suffice for our purposes as our focus is on constructing polynomial functions.

3.4. Evolution Strategy

Our genetic algorithm employs the $(\mu + \lambda)$ – evolution strategy ($(\mu + \lambda)$ – ES). Here, μ denotes the number of individuals selected for the next generation, and λ is the number of offspring generated at each iteration. In a $(\mu + \lambda) - ES$, both parents and offspring are retained in a combined selection pool of size $\gamma = \mu + \lambda$. This strategy, commonly known as elitism, unlike the alternative $(\mu, \lambda) - ES$, guarantees that the best individual found so far is preserved across generations. Under suitable conditions, such elitist selection ensures the global convergence of the evolution strategy. For further theoretical details and an in-depth explanation of evolution strategies, we refer the reader to [20].

4. Step-by-Step Schema of the Algorithm

Our method implements the classic flow of a genetic algorithm. It begins with an Initialization phase to create the initial population of individuals, followed by an Initial Fitness Evaluation to prepare this population for further evolution. Next, the Evolutionary Cycle, the main body of the algorithm, allows candidates to evolve towards a solution. Then, a New Generation is formed by evaluating the fitness of newly generated individuals for the next evolution cycle. Finally, a Termination Check is performed to determine whether a solution has been found or if the algorithm is trapped in a local minimum.

Step 1. Initialization.

Generate an initial population of individuals $P = \{V_i\}_{i=1}^{G}$, where G is the size of the initial population and each individual V_i is a candidate solution represented as a binary tree. Each tree is constructed using two sets: the set of terminal nodes T (5) and the set of non-terminal nodes H (6).

The tree is constructed recursively. At a given depth d, the node type (terminal or non-terminal) is selected according to a probability that decreases with depth. Specifically, the probability of choosing a non-terminal node is defined as

$$P_{non-terminal}(d) = e^{-\alpha d},$$

where $\alpha > 0$ is a predetermined decay parameter. Consequently, the probability of selecting a terminal node at depth d is

$$P_{terminal}(d) = 1 - e^{-\alpha d}.$$

When a non-terminal node is selected, a node from the set F is placed, and its children are recursively generated (for binary trees, two children), with the depth incremented by one at each recursive call. Conversely, if a terminal node is selected, a node from T is placed, ending that branch of the recursion.

This strategy ensures that as the depth d increases, the likelihood of selecting a terminal node increase, thereby regulating the overall tree size and preventing uncontrolled growth. Each individual tree in the initial population is constructed independently according to this scheme, resulting in a diverse set of candidate solutions for the algorithm.

doi:10.20944/preprints202504.0027.v1

Step 2. Initial Fitness Evaluation.

The numerical evaluation of $L_{f,\rho}(V_i)$ proceeds as follows:

1. Domain Sampling:

   Let $D \subset \mathbb{R}^n$ be a predefined domain of interest. To calculate the fitness function, we generate a set of N test points $\{\hat{x}_j\}_{j=1}^{N}$ by sampling independently from the uniform distribution $U(D)$ over D. These points form a well-distributed mesh covering the domain D and serve as the basis for numerical approximation of the fitness values.
2. Fitness Computation:

   For each individual V_i in the population $P = \{V_i\}_{i=1}^{G}$, we evaluate its fitness value l_i on the set of test points $\{\hat{x}_j\}_{j=1}^{N}$ using the empirical Lyapunov fitness $L_{f,\rho}$ (4), which quantifies the degree to which V_i violates the exponential stability conditions on domain D.

The computed fitness values $\{l_i\}_{i=1}^{G}$ determine the likelihood of each individual being selected for reproduction or mutation in subsequent generations. Individuals with lower fitness values (i.e., those that better satisfy the exponential stability conditions) typically have a greater chance of producing offspring, guiding the population towards better solutions over successive generations.

Step 3. Evolutionary Cycle.

The evolutionary cycle simulates Darwinian natural selection by iteratively applying selection, crossover, and mutation to evolve the population toward improved solutions. These genetic operations balance exploration (discovering new regions of the solution space) and exploitation (refining high-quality candidates).

The selection operator is defined as:

$$select(\{V_i\}_{i=1}^{N}, \{l_i\}_{i=1}^{N}),$$

where $\{V_i\}_{i=1}^{N}$ is the set of individuals in the current generation, and $\{l_i\}_{i=1}^{N}$ are their corresponding fitness values. We employ tournament selection [21] in our implementation. In each tournament, a fixed number n_t of individuals are randomly sampled from the population, and the individual with the lowest fitness is selected for reproduction. This approach ensures that high-quality candidates have a higher chance of producing offspring while maintaining sufficient diversity to prevent premature convergence.

The crossover operator is defined as:

$$cross(V_p, V_q),$$

where V_p and V_q are two parent individuals chosen via selection, and p_c is the predefined crossover probability. With probability p_c, a random node is selected in the tree representations of both parents V_p and V_q. The corresponding subtrees are swapped, generating two offspring that inherit genetic material from both parents, introducing new variations. If crossover is not performed (with probability $1 - p_c$), the parent individuals are passed forward unchanged. Figure 1 illustrates the crossover operation, where shaded nodes are selected to be swapped between parents.

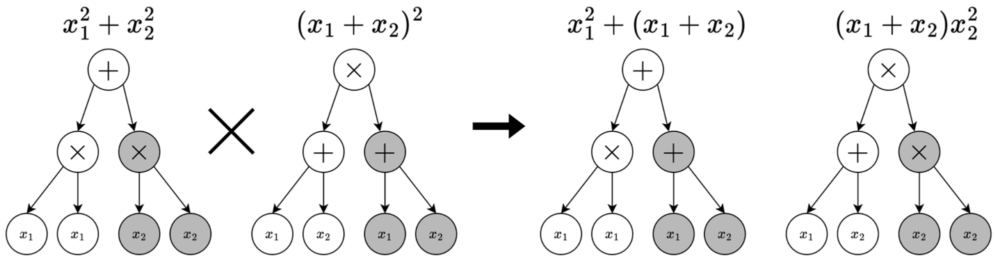


Figure 1. Crossover diagram.

The mutation operator is defined as:

$$mutate(V, p_m),$$

where V is an individual (either offspring of crossover or a direct parent copy), and p_m is the mutation probability. With probability p_m, a random node in the tree representation of V is replaced with a newly generated subtree, constructed using the recursive initialization procedure. If no mutation occurs (with probability $1 - p_m$), then V is retained in the offspring pool, if it results from crossover; however, if V is a direct copy of a parent, it is removed to avoid stagnation and promote population diversity. Figure 2 illustrates the mutation operation, where shaded nodes from the individual were replaced by randomly generated subtree.

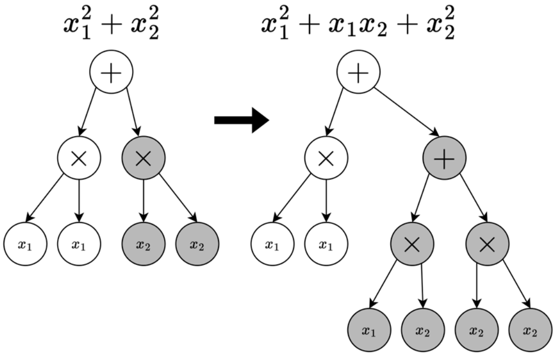


Figure 2. Mutation diagram.

These operations are repeated until the offspring pool reaches the desired size of λ. After the offspring pool is fully populated, each individual's fitness from the offspring pool is evaluated using the empirical Lyapunov fitness (4).

Step 4. Forming a New Generation.

Following the $(\mu + \lambda) - ES$, we merge the current generation with the offspring pool, forming a combined pool of size $\gamma = \mu + \lambda$.

All individuals in this pool are ranked by fitness, and the top μ (i.e., those with the lowest fitness values) are selected to form the next generation. This elitist strategy ensures the survival of the best-performing individuals from the previous generation, while allowing new variations to improve diversity. This balance enhances convergence and maintains stability in the evolutionary process.

Step 5. Termination Check

The algorithm continues to evolve until one of the following stopping criteria is met:

- Maximum Generations Reached: The algorithm runs for a predefined number of generations n_g, after which the algorithm terminates.
- Early Stopping: If an individual achieves a predefined target fitness value of 0.
- No Improvement: If the best fitness value remains unchanged for n_b, consecutive generations, the algorithm is considered converged, and execution stops.

These termination conditions ensure computational efficiency, preventing unnecessary iterations while striving for optimal solutions.

5. Experiments

In this section, we validate the proposed algorithm by constructing polynomial Lyapunov functions for several nonlinear dynamical systems.

doi:10.20944/preprints202504.0027.v1

We implemented an application for numerical experiments. The environment parameters of workstation and environment are as follow:

- Programming Language: Python 3.13.
- Libraries:
    - Symbolic Computations: PySR [22].
    - Evolutionary algorithms framework: DEAP [23].
    - Visualization: Matplotlib [24].
- Algorithm settings:
    - Initial Population Size: $G = 1000$.
    - Maximum Number of Generations: $n_g = 100$.
    - Number of Individuals Selected for the Next Generation: $\mu = 500$.
    - Number of Offspring: $\lambda = 250$.
    - Crossover Probability: $p_c = 0.2$.
    - Mutation Probability: $p_m = 0.1$.
    - Number of Random Test Points: $N = 5000$.
- Constraints:
    - Domain of Interest: $D = \{x \in \mathbb{R}^n \mid \|x\|_2 \leq 1\}$.
    - Constans $\{a_1, \dots, a_n\} \subset \mathbb{Z}$.

For visualization purposes, we define three error functions:

$$E_1(x) = k_1\|x\|^2 - V(x), \tag{7}$$

$$E_2(x) = V(x) - k_2\|x\|^2, \tag{8}$$

$$E_1(x) = k_3\|x\|^2 + \nabla V(x)^T f(x), \tag{9}$$

which correspond to the lower-bound, upper-bound, and dissipation penalties, respectively. They are each expected to be non-positive on a chosen domain to satisfy (2) and to ensure the exponential stability conditions are met.

Example 1: Simple Two-Dimensional Linear System

$$\begin{cases} \dot{x_1} = -x_1, \\ \dot{x_2} = -x_2. \end{cases} \tag{10}$$

This system describes two independent, exponentially stable modes, where each state variable decays exponentially to the origin.

After two iterations, the algorithm identified the following quadratic function, which is commonly used as an initial guess for a Lyapunov function:

$$V(x) = x_1^2 + x_2^2 = \|x\|^2. \tag{11}$$

It is straightforward to verify that V is a valid Lyapunov function for this system. Its orbital derivative is

$$V(x)^T f(x) = -2(x_1^2 + x_2^2) = -2\|x\|^2. \tag{12}$$

Thus, the exponential stability conditions are satisfied for any neighborhood of the origin with the constants $k_1, k_2 = 1$ and $k_3 = 2$.

The surface and contour plots of the Lyapunov function (11) in Figure 3 illustrate its quadratic form, where the level sets form concentric circles centered at the origin, while the orbital derivative (12) in Figure 4 confirms strict negativity across all directions. Although the error functions (7–9) are zero when $k_1, k_2 = 1$ and $k_3 = 2$, for illustration we used $k_1 = 0.5$, $k_2 = 2$ and $k_3 = 1$. Figures 5–7 present surface and contour plots of the errors E_1, E_2, E_3 using illustrative parameter.

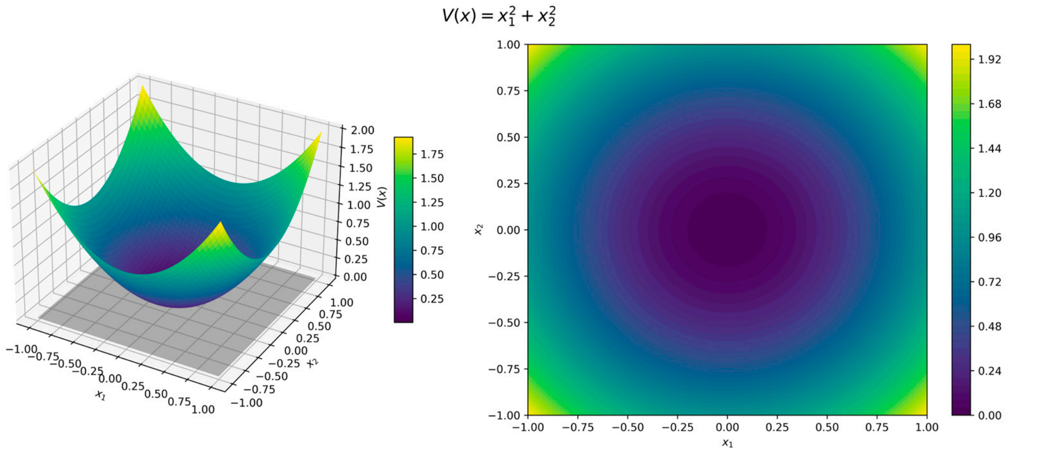


Figure 3. The Lyapunov function (11) for the linear system (10).

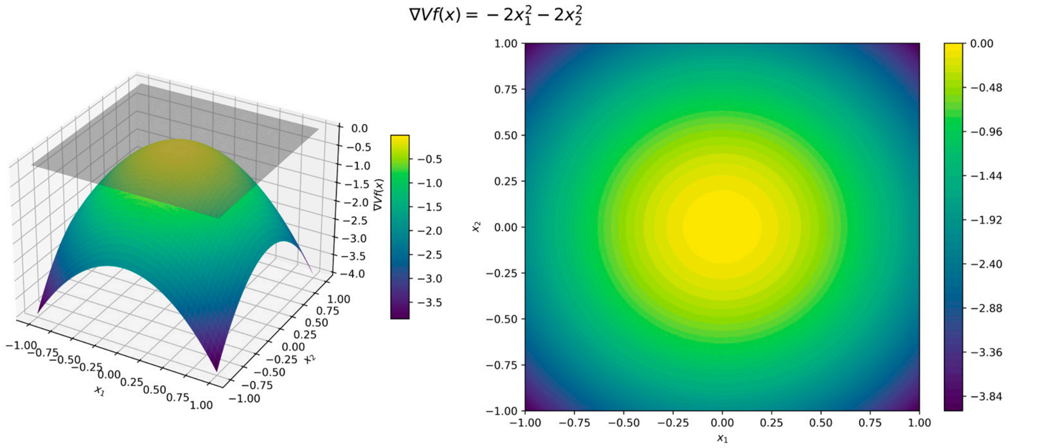


Figure 4. The orbital derivative (12) of the Lyapunov function (11) for the linear system (10).

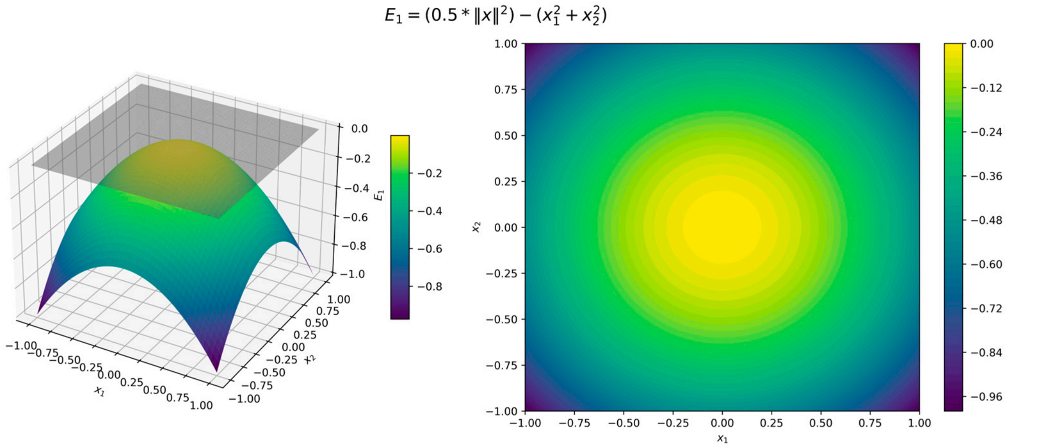


Figure 5. Lower-bound penalty (7) of the Lyapunov function (11) for the linear system (10).

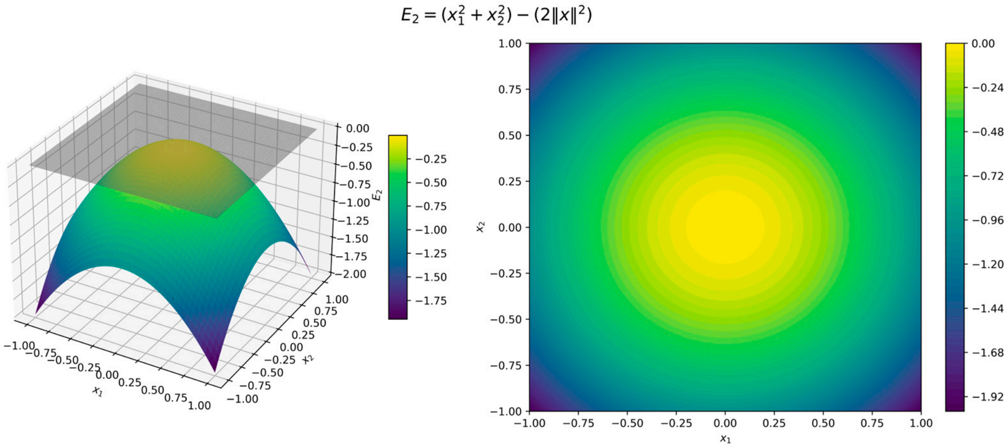


Figure 6. Upper-bound penalty (8) of the Lyapunov function (11) for the linear system (10).

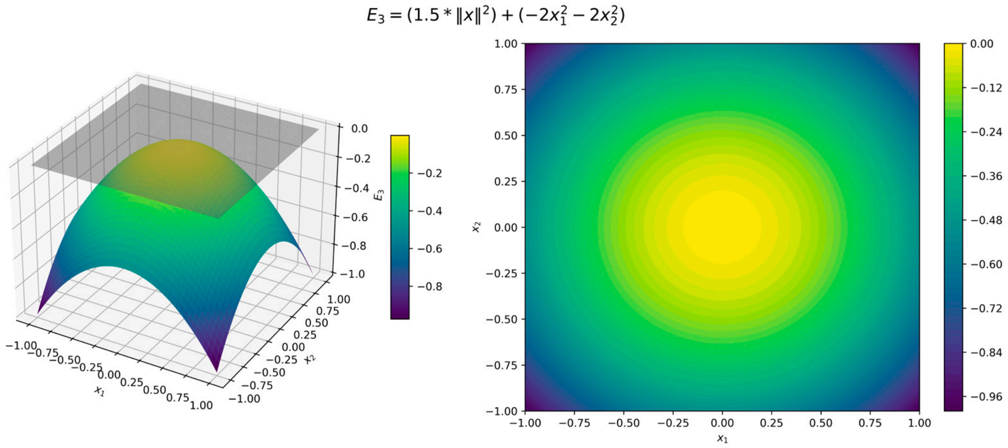


Figure 7. Dissipation penalty (9) of the orbital derivative (12) for the linear system (10).

Example 2: The Damped Pendulum

$$\begin{cases} \dot{x_1} = x_2, \\ \dot{x_2} = -sin(x_1) - x_2. \end{cases} \tag{13}$$

This is a classic example of a nonlinear dynamical system. This system satisfies the conditions of Theorem 7, where the Hurwitz linearization matrix at the equilibrium point $x^* = 0$ confirms the exponential stability of the origin.

The algorithm successfully identified the following quadratic Lyapunov function in the ninth generation:

$$V(x) = 2x_1^2 + 2x_1x_2 + 2x_2^2. \tag{14}$$

The orbital derivative is

$$\nabla V(x)^T f(x) = 2x_2(2x_1 + x_2) - 2(x_1 + 2x_2)\big(x_2 + sin(x_1)\big). \tag{15}$$

The Lyapunov function guarantees the exponential stability of the damped pendulum system (13) within the domain $\|x\| \le 1$, and constants $k_1 = 1, k_2 = 3$ and $k_3 = 1$.

Figure 8 presents the surface and contour plots of Lyapunov function (14), highlighting its quadratic form, where the level sets form concentric ellipses centered at the origin. The corresponding orbital derivative (15), shown in Figure 9, confirms strict negativity across all direction. Additionally, Figures 10–12 presents the surface and contour plots of the errors E_1, E_2, E_3 which visually confirming that the exponential stability conditions of the origin are met.

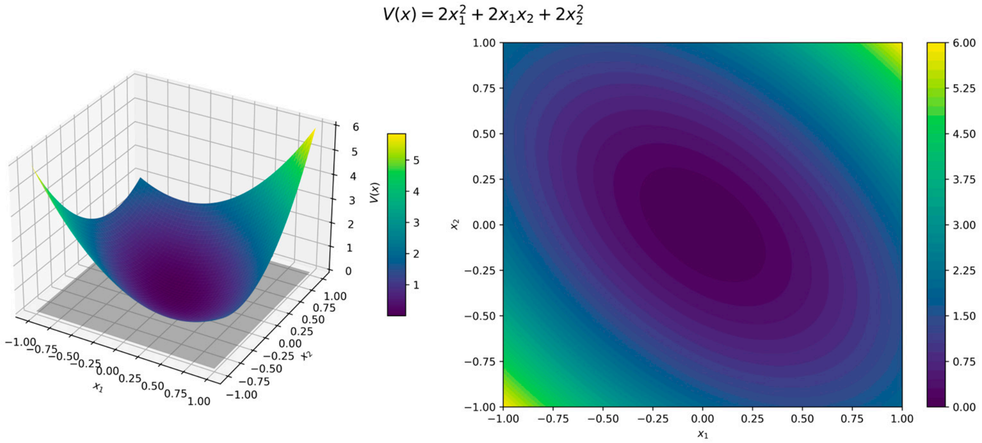


Figure 8. The Lyapunov function (14) for the dumped pendulum (13).

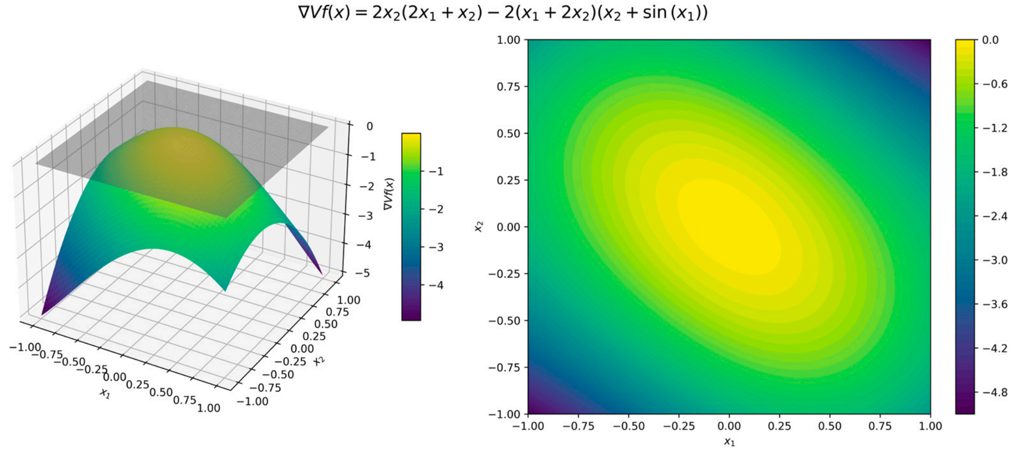


Figure 9. The orbital derivative (15) of the Lyapunov function (14) for the dumped pendulum (13).

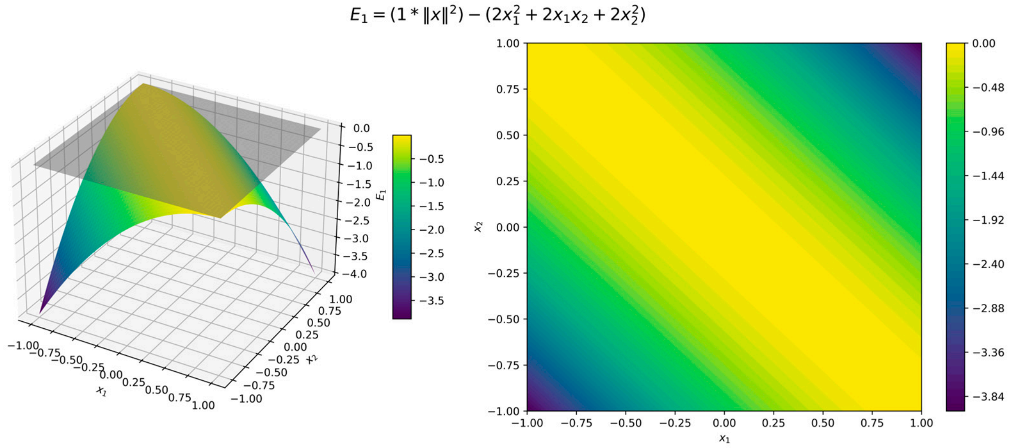


Figure 10. Lower-bound penalty (7) of the Lyapunov function (14) for the dumped pendulum (13).

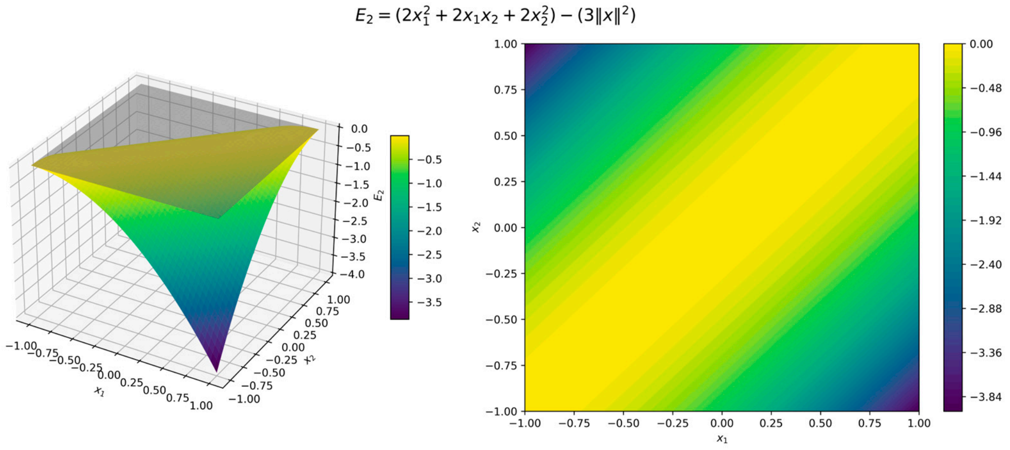

$E_2 = (2x_1^2 + 2x_1x_2 + 2x_2^2) - (3\|x\|^2)$

Figure 11. Upper-bound penalty (8) of the Lyapunov function (14) for the dumped pendulum (13).

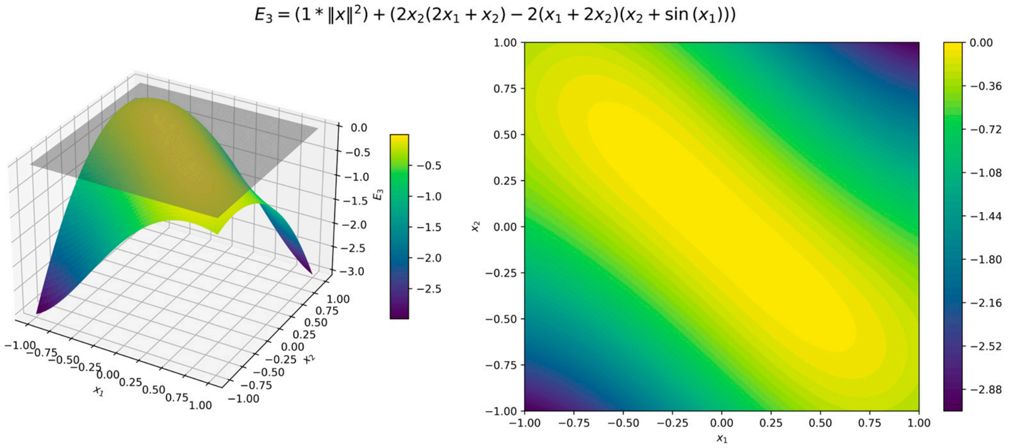

$E_3 = (1 * \|x\|^2) + (2x_2(2x_1 + x_2) - 2(x_1 + 2x_2)(x_2 + \sin(x_1)))$

Figure 12. Dissipation penalty (9) of the orbital derivative (15) for the dumped pendulum (13).

Example 3: A two-dimensional polynomial system

$$\begin{cases} \dot{x_1} = -x_1 - 10x_2^2, \\ \dot{x_2} = -2x_2. \end{cases} \tag{16}$$

This example, presented in [9], examines a two-dimensional system that employs a compositional Lyapunov function. This system meets the criteria of Theorem 7, where the Hurwitz linearization matrix at the equilibrium point $x^* = 0$ confirms the exponential stability of the origin.

The algorithm successfully identified the Lyapunov function after 16 iterations:

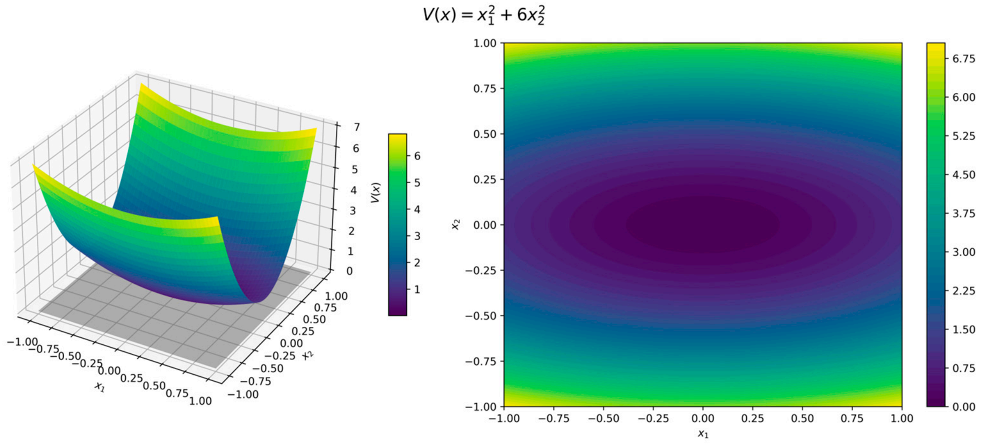
$$V(x) = x_1^2 + 6x_2^2. \tag{17}$$

The orbital derivative is

$$\nabla V(x)^T f(x) = -2x_1(x_1 + 10x_2^2) - 24x_2^2. \tag{18}$$

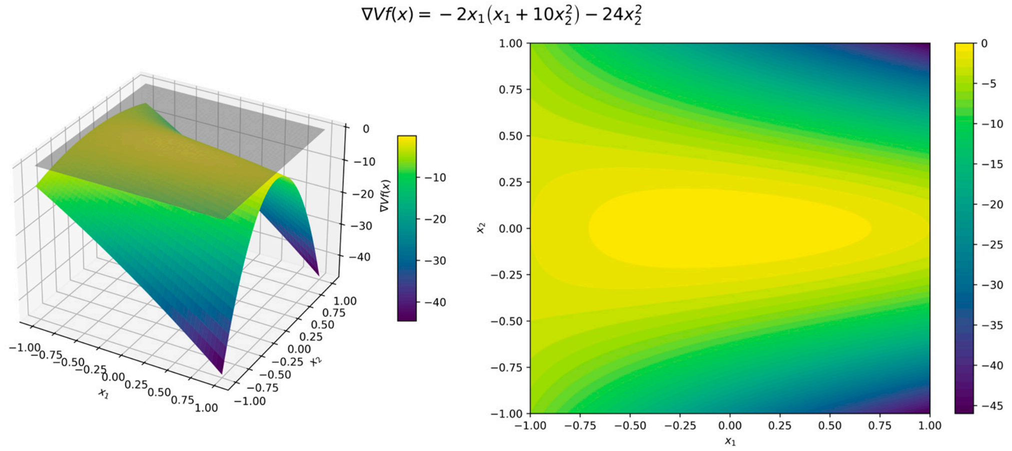
The Lyapunov function (17) ensures that the polynomial system (16) is exponentially stable within the domain $\|x\| \leq 1$, and constants $k_1 = 1, k_2 = 6$ and $k_3 = 2$.

Figure 13 shows the surface and contour plots of function (17), visually confirming its properties. In Figure 14 shows that, the orbital derivative is negative over the considered domain. Figures 15–17 display the surface and contour plots of the error functions E_1, E_2, and E_3 providing additional insight into the effectiveness of (17) in capturing the stability characteristics of the polynomial system (16).



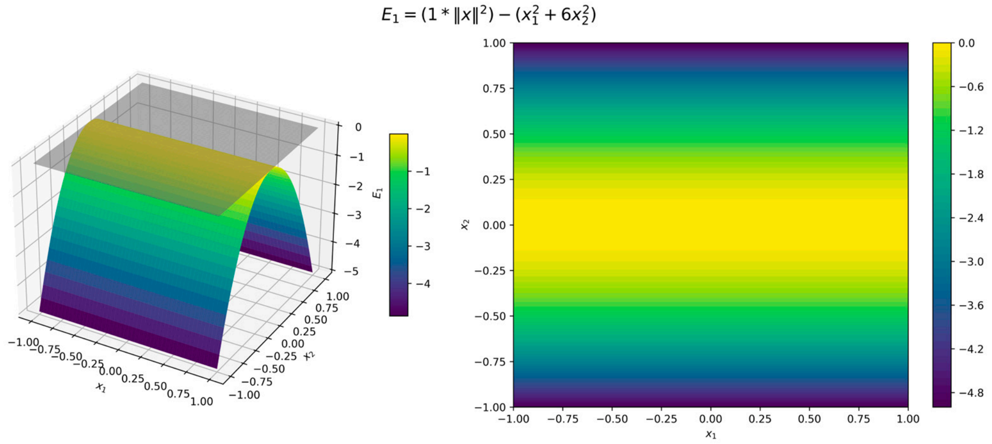
$V(x) = x_1^2 + 6x_2^2$

Figure 13. The Lyapunov function (17) for a polynomial system (16).



$\nabla V f(x) = -2x_1(x_1 + 10x_2^2) - 24x_2^2$

Figure 14. The orbital derivative (18) of the Lyapunov function (17) for a polynomial system (16).



$E_1 = (1 * \|x\|^2) - (x_1^2 + 6x_2^2)$

Figure 15. Lower-bound penalty (7) of the Lyapunov function (17) a polynomial system (16).

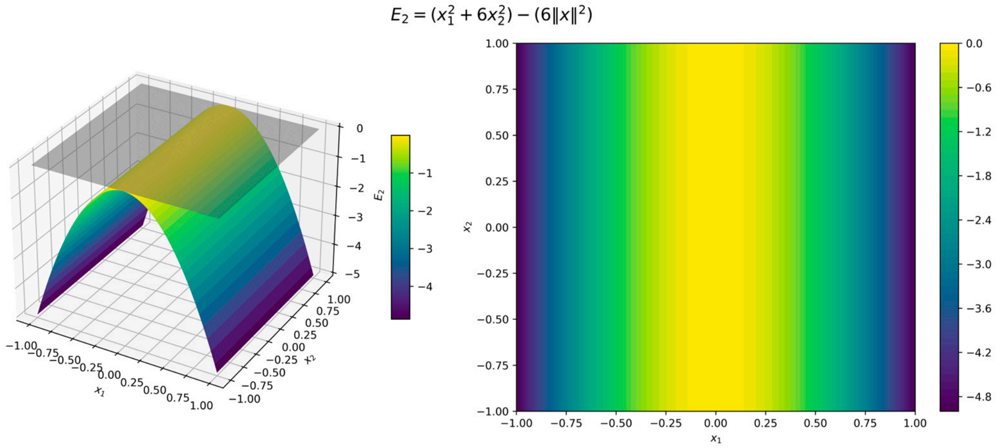


Figure 16. Upper-bound penalty (8) of the Lyapunov function (17) a polynomial system (16).

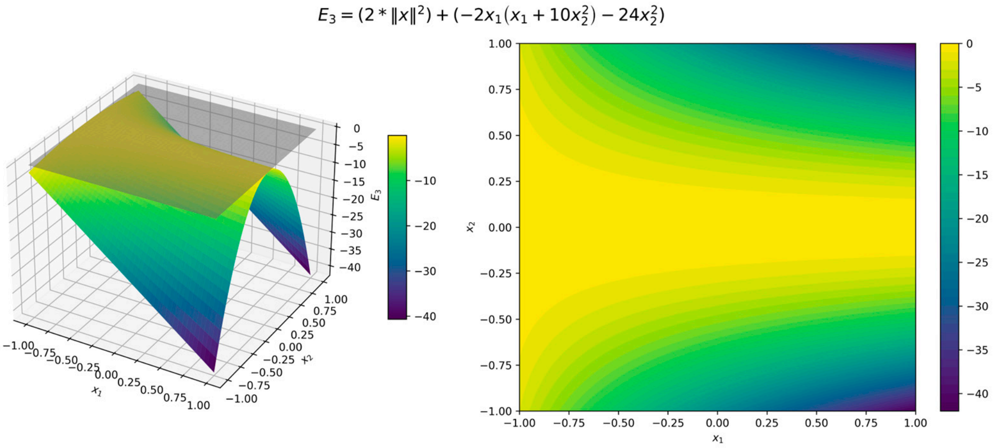


Figure 17. Dissipation penalty (9) of the orbital derivative (18) a polynomial system (16).

Example 4: The Van Der Pol oscillator

$$\begin{cases} \dot{x_1} = -x_2, \\ \dot{x_2} = x_1 + x_2(x_1^2 - 1). \end{cases} \tag{19}$$

This system provides another classic example of a nonlinear dynamical system - a Van der Pol equation in reverse time, that is, with t replaced by $-t$. It fulfills the conditions outlined in Theorem 7, with the Hurwitz linearization matrix at the equilibrium point $x^* = 0$, thus verifying the exponential stability of the origin.

The algorithm successfully identified the Lyapunov function in the twelfth generation:

$$V(x) = x_1^2 - x_1x_2 + x_2^2. \tag{20}$$

The orbital derivative is

$$\nabla V(x)^T f(x) = -x_2(2x_1 - x_2) - (x_1 - 2x_2)\big(x_1 + x_2(x_1^2 - 1)\big). \tag{21}$$

The Lyapunov function guarantees the exponential stability of the Van Der Pol system (19) within the domain $\|x\| \leq 1$, and constants $k_1 = 0.1, k_2 = 3$ and $k_3 = 0.1$.

Figure 18 shows the surface and contour plots of function (20), visually confirming its properties. Figure 19 depicts, the orbital derivative, revealing the intricate nonlinear dynamics of the system.

doi:10.20944/preprints202504.0027.v1

Figures 20–22 display the surface and contour plots of the error functions E_1, E_2, E_3, offering further insight into the accuracy with which function (20) represents the stability properties of the Van der Pol system (19). The red region in Figure 22 indicates areas where $E_3 > 0$, meaning the dissipation conditions are not met. However, this region falls outside the domain D.

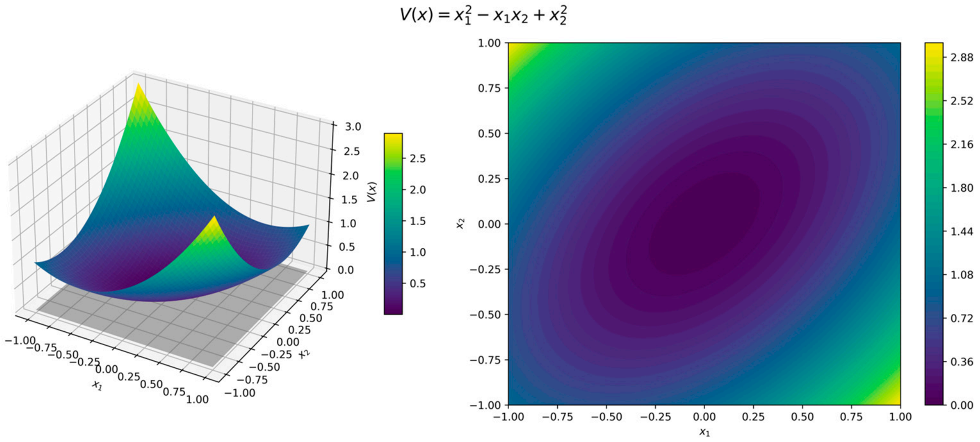


Figure 18. The Lyapunov function (20) for the Van Der Pol oscillator (19).

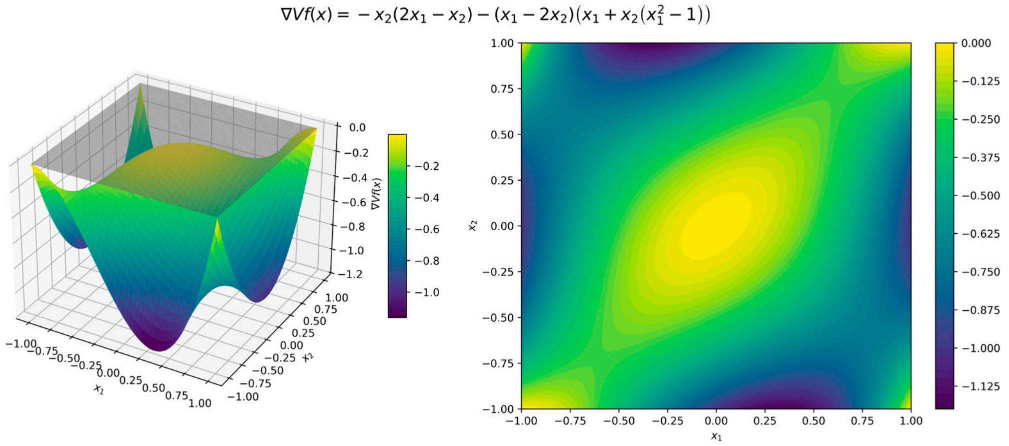


Figure 19. The orbital derivative (21) of the Lyapunov function (20) for the Van Der Pol oscillator (19).

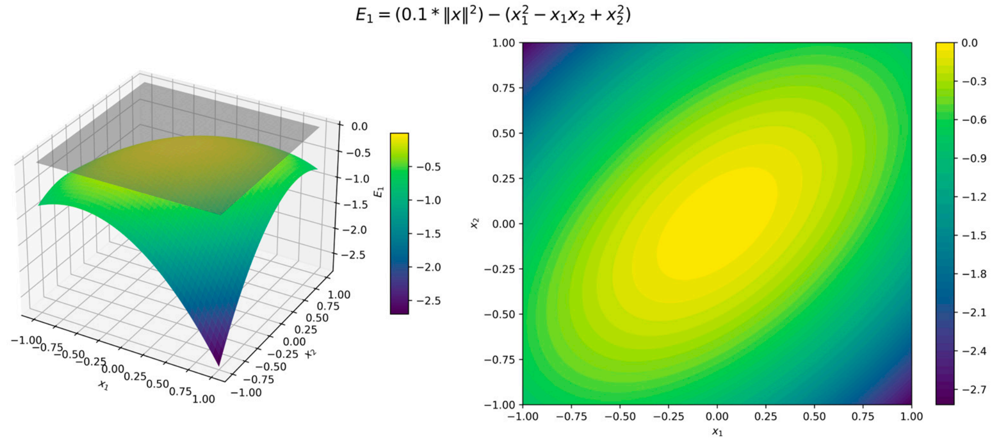


Figure 20. Lower-bound penalty (7) of the Lyapunov function (20) for the Van Der Pol oscillator (19).

doi:10.20944/preprints202504.0027.v1

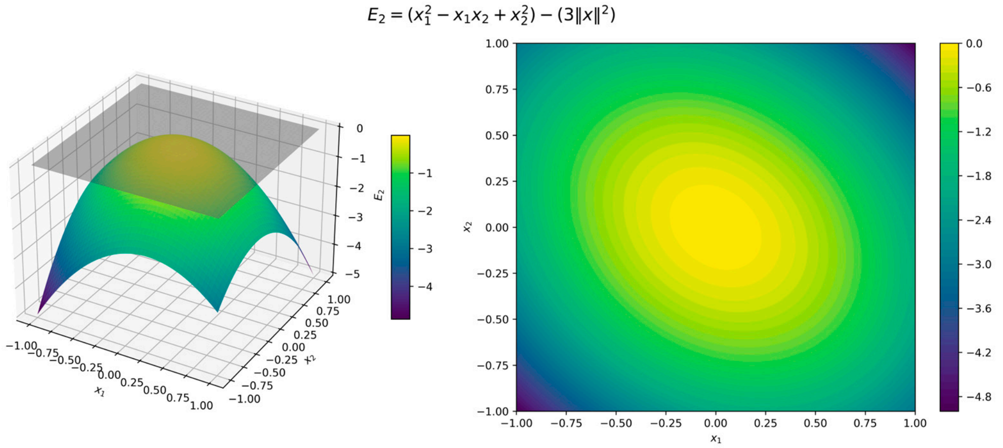


Figure 21. Upper-bound penalty (8) of the Lyapunov function (20) for the Van Der Pol oscillator (19).

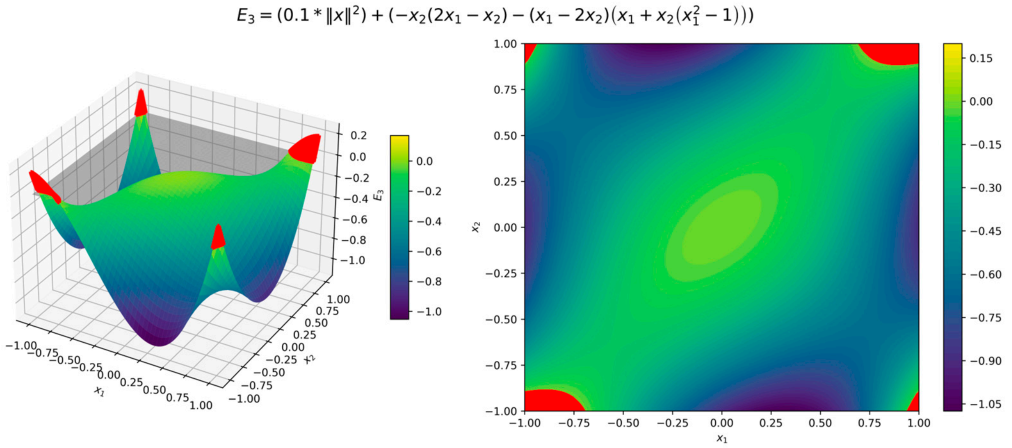


Figure 22. Dissipation penalty (9) of the orbital derivative (21) for the Van Der Pol oscillator (19). The red regions indicate areas where $E_3 > 0$; these areas lie outside the domain D.

6. Discussion

Our numerical experiments demonstrate that a GP-based search for polynomial Lyapunov functions reliably verifies stability of nonlinear systems. The fitness metric (4) employed in our GP-based method was specifically designed to enforce both positivity and dissipation conditions, ensuring that every Lyapunov criterion is satisfied. This metric effectively steered the evolutionary process toward valid polynomial solutions in our numerical tests. Additionally, the method's inherent flexibility allows it to adapt to a variety of problem configurations and complexities. However, further work is needed to optimize the algorithm for higher-dimensional systems.

Future research can build on this work in several promising directions:

- Enlarging the Domain of Attraction: Refinements are required to systematically expand the region over which the Lyapunov function remains valid. By prioritizing domain extensions in GP, the resulting Lyapunov function could cover a larger portion of the state space, yielding stronger stability guarantees.
- Enhanced Constant Selection and Optimization: This approach can be strengthened further by developing advanced heuristics and adaptive strategies for tuning the numerical constants, including the terminal node constants $\{a_1, \ldots, a_n\}$ and the Lyapunov fitness coefficients k_1, k_2 and k_3. These enhancements could mitigate convergence to suboptimal solutions, optimize computational load, and improve the method's effectiveness in estimating viable solutions.
- Compositional Analytical Functions for High-Dimensional Systems: For larger-scale systems, adapting the algorithm to construct compositional (separable or modular) Lyapunov functions

is promising. Such an adaptation could mitigate the curse of dimensionality while preserving solution interpretability in complex, real-world applications.

Additionally, our GP-based method can be extended to the control domain by generating control Lyapunov functions to facilitate the design of robust feedback controllers. Future work could also explore theoretical convergence guarantees, performance bounds, and the integration of these evolutionary methods with classical control techniques to develop a unified framework that combines the strengths of both data-driven and analytical approaches in stability analysis.

7. Conclusions

In this paper, we proposed a Genetic Programming-based method to construct polynomial Lyapunov functions for exponentially stable nonlinear systems. In Theorem 7, we established the existence criterion for these functions, which serve as the primary indicator of the method's applicability. We also introduced the Lyapunov fitness (3) to quantify deviations from exponential stability conditions.

We validated the method through numerical experiments on several dynamical systems, including a simple linear system (10), a damped pendulum (13), a polynomial system (16), and the Van der Pol oscillator (19). These tests demonstrate that the proposed method effectively constructs Lyapunov functions and can be used as an automated tool for stability analysis.

The future directions outlined highlight the potential to further enhance and extend the proposed approach. This work offers a flexible, automated method for Lyapunov-based stability certification, bridging evolutionary computation with classical control theory.

Author Contributions: Conceptualization, Roman Pykhnivskyi, Anton Ryzhov, Andrii Sobchuk and Yurii Kravchenko; Formal analysis, Roman Pykhnivskyi and Yurii Kravchenko; Investigation, Roman Pykhnivskyi; Methodology, Roman Pykhnivskyi and Anton Ryzhov; Software, Roman Pykhnivskyi; Validation, Roman Pykhnivskyi; Visualization, Roman Pykhnivskyi and Andrii Sobchuk; Writing – original draft, Roman Pykhnivskyi; Writing – review & editing, Roman Pykhnivskyi and Anton Ryzhov.

Funding: This research received no external funding.

Data Availability Statement: Data is contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Pichkur, V.; Kapustyan, O.; Sobchuk, V. *Stability and Attractors of Evolutionary Equations*; Taras Shevchenko National University of Kyiv: Kyiv, Ukraine, 2023. (in Ukrainian)
2. Khalil, H.K. *Nonlinear Systems*, 3rd ed.; Prentice Hall: Upper Saddle River, NJ, 2002.
3. Yoshizawa, T. *Stability Theory by Lyapunov's Second Method*; Mathematical Society of Japan: Tokyo, Japan, 1966.
4. Nikravesh, S.K.Y. *Nonlinear Systems Stability Analysis: Lyapunov-Based Approach*; 1st ed.; CRC Press, 2018.
5. Sobchuk, V.; Barabash, O.; Musienko, A.; Tsyganivska, I.; Kurylko, O. Mathematical Model of Cyber Risks Management Based on the Expansion of Piecewise Continuous Analytical Approximation Functions of Cyber Attacks in the Fourier Series. *Axioms* 2023, *12*, 924, doi:10.3390/axioms12100924.
6. Kapustian, O.A.; Kapustyan, O.V.; Ryzhov, A.; Sobchuk, V. Approximate Optimal Control for a Parabolic System with Perturbations in the Coefficients on the Half-Axis. *Axioms* 2022, 11, 175. doi:10.3390/axioms11040175.
7. Anderson, J.; Papachristodoulou, A. Advances in Computational Lyapunov Analysis Using Sum-of-Squares Programming. *Discrete and Continuous Dynamical Systems - B* 2015, 20(8): 2361-2381, doi:10.3934/dcdsb.2015.20.2361.
8. Ahmadi, A.A.; Parrilo, P.A. Sum of Squares Certificates for Stability of Planar, Homogeneous, and Switched Systems. *IEEE Trans. Automat. Contr.* 2017, *62*, 5269–5274, doi:10.1109/TAC.2016.2647253.

9. Hafstein, S.F.H. Algorithm for Constructing Lyapunov Functions. *Electronic Journal of Differential Equations* 2007, 1 (Mon. 01-09), Mon. 08, 1-101, doi:10.58997/ejde.mon.08.
10. Munser, L.; Devadze, G.; Streif, S. Synthesis of Lyapunov Functions Using Formal Verification 2021, doi:10.48550/arXiv.2112.01835
11. Grüne, L. Computing Lyapunov Functions Using Deep Neural Networks. *JCD* 2021, *8*, 131. https://doi.org/10.3934/jcd.2021006.
12. Hafstein, S.; Giesl, P. Review on Computational Methods for Lyapunov Functions. *DCDS-B* 2015, *20*, 2291–2331, doi:10.3934/dcdsb.2015.20.2291.
13. Koza, JohnR. Genetic Programming as a Means for Programming Computers by Natural Selection. *Stat Comput* 1994, *4*, doi:10.1007/BF00175355.
14. Holland, J.H. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*; The MIT Press, 1992.
15. Grosman, B.; Lewin, D.R. Automatic generation of Lyapunov functions using genetic programming. *IFAC Proceedings Volumes* 2005, *38*, 75–80, doi:10.3182/20050703-6-CZ-1902.00873.
16. McGough, J.S.; Christianson, A.W.; Hoover, R.C. Symbolic Computation of Lyapunov Functions Using Evolutionary Algorithms. In Proceedings of the IASTED Technology Conferences / 696:MS / 697:CA / 698: WC / 699: EME / 700: SOE; ACTAPRESS: Banff, Alberta, Canada, 2010, doi:10.2316/P.2010.697-093
17. Feng, J.; Zou, H.; Shi, Y. Combining Neural Networks and Symbolic Regression for Analytical Lyapunov Function Discovery 2024. doi:10.48550/arXiv.2406.15675.
18. Angelis, D.; Sofos, F.; Karakasidis, T.E. Artificial Intelligence in Physical Sciences: Symbolic Regression Trends and Perspectives. *Arch Computat Methods Eng* 2023, *30*, 3845–3865, doi:10.1007/s11831-023-09922-z.
19. Peet, M.M. Exponentially Stable Nonlinear Systems Have Polynomial Lyapunov Functions on Bounded Regions. *IEEE Trans. Automat. Contr.* 2009, *54*, 979–987, doi:10.1109/TAC.2009.2017116.
20. Beyer, H.-G.; Schwefel, H.-P. Evolution Strategies – A Comprehensive Introduction. *Natural Computing* 2002, *1*, 3–52, doi:10.1023/A:1015059928466.
21. Hancock, P.J.B. An Empirical Comparison of Selection Methods in Evolutionary Algorithms. In *Evolutionary Computing*; Fogarty, T.C., Ed.; Lecture Notes in Computer Science; Springer Berlin Heidelberg: Berlin, Heidelberg, 1994; Vol. 865, pp. 80–94 ISBN 978-3-540-58483-4.
22. Cranmer, M. Interpretable Machine Learning for Science with PySR and SymbolicRegression.jl. doi:10.48550/arXiv.2305.01582
23. Fortin, F.-A.; Rainville, F.-M.D.; Gardner, M.-A.; Parizeau, M.; Gagné, C. DEAP: Evolutionary Algorithms Made Easy. *Journal of Machine Learning Research* 2012, *13*, 2171–2175.
24. Hunter, J.D. Matplotlib: A 2D Graphics Environment. *Comput. Sci. Eng.* 2007, 9, 90–95, doi:10.1109/MCSE.2007.55.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.