Scalable Sampling-based Sum-of-Squares Programs for Systems Verification

Shen Shen and Russ Tedrake

Abstract—This paper presents a novel method, combining new problem formulations and sampling, to improve the scalability of sum-of-squares (SOS) programs for stability verification. Region-of-attraction approximation problems are considered for polynomial, polynomial with generalized Lur’e sector uncertainty, and rational trigonometrical rigid-body systems. Our method starts by identifying that Lagrange multipliers, traditionally heavily used for S-procedure or clearing the rational’s denominator, are a major culprit of creating bloated SOS programs. In light of this, we exploit the inherent problem properties such as continuity, convexity, and implicit algebraic structure, and reformulate the problems as quotient ring SOS programs, thereby eliminating all multipliers. These new programs are smaller, sparser, less constrained, yet less conservative. Their computation is further improved by leveraging a recent result on sampling algebraic varieties. Remarkably, solution correctness is guaranteed with just a finite (in practice, very small) number of sampled instances. Altogether, the proposed method can verify systems well beyond the reach of existing SOS-based approaches (29 states). On smaller problems where a baseline is available, it computes tighter solution 2-3 orders faster. Source code is included.

I. INTRODUCTION

We consider the fundamental stability verification problem of approximating the region-of-attraction (ROA) for nonlinear systems, where the system dynamics are either polynomial, polynomial with generalized Lur’e sector uncertainty, or rational trigonometrical. Sum-of-squares (SOS) programs are widely accepted as a standard way to approach these problems [7], [11], [12], [9]. Powered by semidefinite programs (SDPs), the SOS framework provides a systematic way to search and optimize sub-levelset of a polynomial Lyapunov function for the inner-approximation task.

Despite the popularity and rich theories, SOS-based approaches have been successful at solving problems of only modest dimension (10-15 states) [4]. This is limiting, as many interesting real-world applications are well beyond that scale, for example, networked systems or mechanical systems consisting of many rigid bodies.

Improving the scalability for SOS is therefore of great importance, and various techniques have been proposed. Most of them, rightfully so, identify the lower-level SDPs as the computational bottleneck. Specifically, some methods assume the problem data has certain structures, such as compositional [10] or sparse [5]; these assumptions in turn limit the generality of the methods. Another closely-related direction is to impose structures on the decision variables at the cost of conservatism; for example, the dsos and sdos hierarchy [1], which relax the optimization class to linear or second-order cone programs. There has also been considerable effort in developing SDP solvers, often to complement or break away from the interior point method, for instance, augmented Lagrangian methods, or Burer-Monteiro method.

Unlike these SDP-oriented approaches, the proposed method improves the scalability more directly at the problem formulation and polynomial optimization level. Further, it does not assume any special system structure and the solutions are in fact less conservative. If so desired though, it is compatible and can be combined with the aforementioned techniques for even greater computational gain.

At a high-level, the proposed method can be summarized as: reformulate the three ROA verification problems as simpler yet stronger quotient-ring SOS programs, and leverage a more efficient sampling-based method to solve them. We begin by identifying that traditional formulations rely heavily on auxiliary Lagrange multipliers. Take polynomial systems for example, the traditional ROA formulations there rely on the pipeline of (in)equality implication, S-procedure, and high-degree Lagrange multipliers. This not only introduces a large number of auxiliary decision variables and possibly an

This work was supported by (1) ONR, N00014-18-1-2210, and (2) N00014-17-1-2699. The authors are with Computer Science and Artificial Intelligence Lab, MIT, {shenshen, russt}@mit.edu

The source code can be found at https://github.com/shensquared/S4VC. It includes all of the algorithms and examples presented in this paper.

Fig. 1: The proposed methods combine several scaling-improving techniques, and allow us to handle problems well beyond the reach of current SOS program-based approaches (29 states). One such technique is illustrated above on the ROA approximation problem for the simple van-der-pol benchmark. Traditional approaches typically rely on complicated conditions, e.g., on the set of all states enclosed within the yellow line. Our method, without loss of correctness or tightness, only involves examining few randomly sampled states, shown as blue dots, on the yellow line.
extra expensive SOS constraint on the multiplier, it inflates the problem dimension as well, all of which responsible for increasing the program complexity.

The same multiplier-based recipe can be found in the other two system classes. In particular, the standard Lur’e type uncertainties are usually modeled with a sector condition encoded as a quadratic inequality, which then goes through the same S-procedure and multipliers pipeline. Multipliers are more heavily used for analyzing rigid-body mechanical systems, where the dynamics have explicitly rational trigonometric form, and additional multipliers are therefore introduced to clear the denominator. It is worth pointing out that in these two cases, auxiliary indeterminates are also necessary, aggregating the complexity even further in a similar fashion as the multipliers.

Motivated to eliminate all these multipliers and possibly auxiliary indeterminates, we re-examine the problem description for reformulations. Taking advantage of continuity, convexity and f or implicit algebraic structure inherent in these problems, and reformulate them all as quotient ring SOS programs. These are SOS programs that directly reason on algebraic varieties (objects defined by fixed polynomial equality constraints, for example, the yellow line defined by $V(x) = 0$ in Figure 1). Basic algebraic geometry properties imply that the reformulated programs are less conservative, while also smaller, sparse, and less constrained, than their multiplier-based counterpart.

The computation of these quotient ring SOS programs is further improved, significantly, by leveraging a sampling algebraic variety approach, recently introduced in [2]. The method reduces a quotient ring SOS program to sampled numerical instances on the defining algebraic varieties. This is a paradigm shift, whereas traditionally polynomial coefficients are matched, the sampled version matches polynomial evaluations. Remarkably, solution correctness can be guaranteed with a finite (in practice, very small) number of samples.

We point out that, while this paper focuses on the ROA problem, the techniques presented can be straightforwardly extended to closely-related problems such as reachability analysis via barrier functions.

Contributions: Combining the new formulations and sampling variety approach, the proposed method can verify systems well beyond the reach of existing SOS-based approaches (29 states). On smaller problems where a baseline is available, it computes tighter solution 2-3 orders faster. More specifically,

(i) For each of the three types of systems considered, we identify and leverage inherent problem structures, and present a new formulation of the ROA approximation problem as a quotient ring SOS program.

(ii) We apply the efficient sampling variety approach from polynomial optimization to applications arising in system and control.

Both of them, to the best of our knowledge, are proposed for the first time.

II. Problem Statement and Approach Outline

Given a continuous-time closed-loop nonlinear system with dynamics $\dot{x} = f(x)$ and a fixed positive definite polynomial Lyapunov candidate $V(x)$, we consider the task of quantitatively verifying if the system is locally asymptotically stable around the fixed point (assumed to be the origin). Concretely, we are interested in finding a sub-levelset $\mathcal{E}(x; V, \rho) := \{x \mid V(x) < \rho\}$ whose volume grows with $\rho$. The connected component of $\mathcal{E}$ that includes the origin is an inner approximation of the ROA if the following is true:

$$\max \rho \quad \text{s.t.} \quad \dot{V}(x) = \frac{\partial V}{\partial x} f(x) < 0, \forall x \in \mathcal{E}(x; V, \rho) \setminus \{0\} \quad (1)$$

The cost on $\rho$ encourages the enlarging such sub-levelset, thus providing tighter approximation to the true ROA.

We consider solving the ROA approximation on three sub-problems; they differ in the dynamics characterization:

(i) Polynomial problem: The “vanilla” case where $f(x)$ is polynomial in $x$.

(ii) Lur’e problem: The dynamics is polynomial with additive uncertainty:

$$f(x) = f_0(x) + \delta(x)$$

where $f_0(x)$ is the nominal dynamics and $\delta(x)$ is a generalized Lur’e type uncertainty with polynomial $\alpha(x)$ and $\beta(x)$ polynomial in $x$.

(iii) Rigid-body problem: The dynamics comes from the equation of motion (EOM) of rigid-body systems:

$$f(x) = M^{-1}(x) F(x)$$

where both $M(x)$ and $F(x)$ include terms like $\sin(x)$, thus $f(x)$ is rational trigonometric.

The overall approach in this paper is to formulate the generic optimization problem Eq. (1) as quotient ring SOS programs for these three problems. The sampling variety subroutine is then applied to efficiently, without loss of correctness, solve the quotient ring SOS program on only a small number of sampled instances on the defining variety.

We begin by describing the complete solution (formulation and sampling) to the polynomial problem. For Lur’e and rigid-body problems, we will focus on illustrating the tailor-made new formulations only, since the sampling subroutine is almost identically applied.

III. Formulation - Polynomial Problem

A. Existing formulations

For polynomial systems, there are two known SOS formulations, as sufficient conditions for problem Eq. (1); both of them rely on multipliers. The more popular one is formulated through a straightforward inequality implication $V \leq \rho \Rightarrow \dot{V} \leq 0$, S-procedure and Lagrange multipliers, we call this the formulation (IE):

$$...$$
\[
\begin{align*}
\max_{\rho,\lambda(x)} & \quad \rho \\
\text{s.t.} & \quad \lambda(V(x) - \rho) - \dot{V}(x) \text{ is SOS} \\
& \quad \lambda(x) \text{ is SOS}
\end{align*}
\]  

(IE)

An alternative equality constrained formulation is due to [7]. In particular, under the assumption that the Hessian of \( \dot{V} \) is negative definite at the origin, the following is also a sufficient condition for problem Eq. (1):

\[
\begin{align*}
\max_{\rho,Q,\lambda(x)} & \quad \rho \\
\text{s.t.} & \quad (x'x)^d (V(x) - \rho) - \lambda(x)\dot{V}(x) = \ldots \\
& \quad m'(x)Q m(x) \forall x
\end{align*}
\]  

where here we explicitly write out the SOS factorization constraint on the right hand side (for easy reference later), with \( m(x) \) denoting the standard monomial basis of appropriate degree.

When the auxiliary multipliers \( \lambda(x) \) are of the same degree choices, the two existing formulations translate into low-level SDPs of similar dimension and lead to similar solutions, but the equality constrained one (E) is much simpler to solve due to the elimination of the SOS condition on the multipliers.

Surprisingly, however, formulation (E) has less presence in the literature, having appeared only in one other reference [4] (to the best of our knowledge). We speculate this lack of popularity is because neither reference has emphasized on the relative advantage, or answered questions such as what is the formulation based on? and what is the purpose of the \( (x'x)^d \) term?

The answers to these questions are not very obvious. Yet, they are important for us to address because our proposed formulation is closely related to them.

B. Proposed formulation

We begin by reverse-engineering formulation (E) to discover its underlying implication condition, described below.

**Theorem 1:** Under the assumption that the Hessian of \( \dot{V} \) is negative definite at the origin, the implication condition

\[ \dot{V}(x) = 0 \Rightarrow V(x) \geq \rho \text{ or } x = 0 \]  

is a sufficient condition for Eq. (1).

**Proof:** The Hessian condition ensures that \( \dot{V}(0) = 0 \) is a local maximum. Therefore, locally around the origin, we must have \( \dot{V} < 0 \); but for states outside the true ROA, \( \dot{V} \) becomes positive. Since both \( V \) and \( f \) are continuous in \( x \), so is \( \dot{V}(x) \). Given the continuity of \( \dot{V}(x) \), and that it changes sign, zero-crossing event(s) must have occurred at some states.

If at all states where zero-crossing occurs, the value of \( V \geq 0 \) or it is precisely the origin, encoded by Eq. (2), then by contraposition, it is equivalent to

\[ x \in \{ x \mid V(x) < \rho, x \neq 0 \} \Rightarrow \dot{V}(x) \neq 0 \]

Given the local behavior of \( \dot{V} \) around the origin, the connected component of the \( \rho \) sub-levelset that includes the origin, must have \( \dot{V} < 0 \) (except for the origin itself).  

An interactive animation of the idea, visualized on the benchmark van-der-pol problem is available online\(^1\). Figure 1 shows a snapshot of it, where the yellow line precisely defines those important non-origin zero-crossings \( \dot{V}(x) = 0 \).

With Theorem1 in place, it should be clear that Formulation (E) is a multiplier-based sufficient condition for Eq. (2), therefore sufficient for Eq. (1) as well. Note the importance of the negative definite Hessian condition\(^2\), it sufficiently implies the local maximum condition needed in the proof. Note also the importance of the \( (x'x)^d \) term, where \( d \), an auxiliary problem data, is a strictly positive integer. Without this term, the optimization is either meaningless, because zero has to be the solution; or infeasible since \( \lambda\dot{V} \) is usually of much higher degree than \( V - \rho \) alone.

Our formulation is a direct application of algebraic geometry ideas on Eq. (2) using basic objects such as affine variety, quotient ring, and Gröbner basis (due to space limitation, we prioritize making the high-level idea clear, and refer to [3] (Chapter 1) for the background and definitions).

In particular, simply by defining an algebraic variety \( V := \{ x \mid \dot{V}(x) = 0 \} \), a sufficient condition to Eq. (2) can be given by:

\[
\begin{align*}
\max_{\rho,Q} & \quad \rho \\
\text{s.t.} & \quad (x'x)^d (V - \rho) = n'(x)Q n(x), \forall x \in V
\end{align*}
\]  

where \( n(x) \) is the standard monomial of Gröbner basis.

(Q) and (E) may seem trivially equivalent and only differ in terminology; after all, they stem from the same high-level polynomial equality constraint Eq. (2). However, there are three facts that make the proposed reformulation (Q) more appealing:

(i) Due to Gröbner, the basis \( n(x) \) in (Q) is lower dimensional than its counterpart, \( m(x) \) in (E);

(ii) Due to the elimination of the \( \lambda\dot{V} \) term, the fixed degree \( d \) is lower in (Q); and

(iii) Due to the reliance on degree-bounded multipliers, any optimal solution \( \rho \) to (E) is only suboptimal to (Q). That is, formulation (Q) is intrinsically less conservative than (E) (unless the multiplier can be of infinite-degree, which is impossible practically). This is an important but subtle fact, we illustrate it via a simple example below.

**Degree-bounded multipliers are “bounded”:** Suppose we want to check whether or not this implication \( x + 1 = 0 \Rightarrow x^2 - 1 \leq 0 \) is true. Multiplier-based formulation would search for a \( \lambda(x) \) such that \( (x^2 + 1) + \lambda(x)(x + 1) \leq 0, \forall x \). This optimization problem can not be feasible if \( \lambda \) is limited to be a constant, even though the implication is true. It takes at least an affine multiplier, for example \( \lambda(x) = -(x - 1) \) to make the problem feasible. In contrast, quotient ring formulation interprets the left hand side condition of the implication as \( x \equiv -1 \), so the right hand side becomes \( 1^2 - 1 = 0 \leq 0 \) which is trivially true.

\(^1\)http://web.mit.edu/~shshen/www/VDP-animation.html

\(^2\)While one reference does not make this assumption explicit and the other has a sign flip, both should be clear in the context
Fact (i) and (ii) mean the proposed quotient ring SOS program (Q) leads to a much smaller SDP; yet it is also stronger due to fact (iii). This makes the quotient ring formulation (Q) strictly better, theoretically. The only potential downside is that Gröbner basis methods are challenging to find, especially for when the defining equations for the variety get more complicated, as will be the case for Lur’e and rigid-body problems. To overcome this potential difficulty, we apply a recent sampling-based method to solve these quotient ring problems.

IV. SAMPLING ON ALGEBRAIC VARIETIES

We apply the sampling algebraic varieties techniques introduced in [2] to solve the quotient ring problem (Q). The high-level idea is rather simple and straightforward: instead of solving the optimization for all real-valued $x$, solve a set of sampled numerical instances of it:

$$\begin{align*}
\max_{\rho \in \mathcal{Q}} & \quad \rho \\
\text{s.t.} & \quad \hat{V}(x_i) = 0, \ \forall x_i \\
& \quad (x_i^T x_i)^d (V(x_i) - \rho) = n'(x_i)Qn(x_i), \ \forall x_i
\end{align*}$$

where $n(x)$ is an implicit Gröbner basis which we will describe later, and all quantities involving a subscript are sampled values.

As is hinted before, there are certain numerical motivations to solve this sampled program (S). But given that the ultimate goal is to produce stability certificate, we should be immediately asking: a solution to the sampled problem (S) is necessarily a solution to problem (Q), what guarantee do we have regarding the sufficiency (as is required to claim correctness)? We first summarize a very brief high-level answer; the detailed technical treatment and proofs can be found in [2].

A. Correctness and sample bound

The short answer is, the sampled program (S) is equivalent to the original program (Q), with probability one, if the samples used $x_i$ are generic.

The genericity condition can be interpreted as checking if there enough samples that are drawn randomly. In theory, there exists a finite random sample upper-bound to guarantee genericity. This bound depends on many factors including the problem size, variety structure, etc. but the important fact is that it is guaranteed to be finite.

In practice, the genericity condition can be rigorously checked by a simple matrix rank test, where the elements in this matrix are simple monomial evaluations of the samples. It is through this practical case-by-case numerical rank check, that we accumulated enough empirical evidence that the samples needed are in fact, very small. Usually, the number of samples we need is less than the number of elements in the Gram matrix. In Section VII, we document the number of samples used for each program, which could serve as an empirical reference.

The “with probability one guarantee” is due to the floating point arithmetic, which is inevitable for any numerical optimization, including any interior-point method based SDP solvers. While not perfect, an additional sanity-check can be easily carried out and improve numerical confidence. In particular, one more random test sample (different from all the samples the solution are based on) can be drawn on the variety, and the sampling-based solution can be tested as to whether it holds on this additional test sample.

Finally, while rigorous proofs of these statements can be found in the reference paper, it might also be helpful to include an intuitive explanation. After all, this finite sample-bound with guarantee may seem very surprising because-with-probability-one is usually stated in the asymptotic regime. Intuitively, the combination of “being exactly on the variety” and “degree-bounded polynomial parameterization” imposes a very strong constraint. To some extend, it is similar to the setup of polynomial interpolation, where a finite number of samples can faithfully determine the coefficients of a degree-bounded polynomial function.

B. Computational benefits.

The major computational gain comes from the paradigm shift of how SOS programs are solved. Whereas traditional methods match polynomial coefficients, sampled approach matches polynomial evaluations for each sample $x_i$.

One direct consequence of this is a low-rank data structure in the resulting SDP. In particular, note that in Problem (S), the right hand side is a scalar evaluation. Via cyclic property, $n'(x_i)Qn(x_i) = \langle Q, n(x_i)n'(x_i) \rangle$, where $n(x_i)n'(x_i)$ is a numerical matrix of rank at most one by construction (because, recall that $n(x_i)$ is a vector). It is important to note that, it is the problem data, rather than the decision variable $Q$, that is low-rank, and solvers usually have mechanisms to take advantage of it. Such low-rank data structure does not appear in traditional SOS programs, since $n(x)$ are monomials.

Another computational advantage comes from the computation of the basis. As mentioned before, standard method of solving quotient-ring SOS programs rely on the construction of explicit Gröbner basis. Here, the basis $n(x)$ are computed numerically via simple SVD procedure; they can be thought of as the orthogonal basis with respect to a natural inner product supported on the samples. As a byproduct of this more efficient basis computation, orthogonalization has been shown to improve SDP program numerical condition as well.

We finally note that, the sampling procedure itself involves finding roots to polynomial equation(s). In the simple case where dynamics itself is polynomial, sampling means finding roots of a single multi-variate polynomial $V$, which can be easily done via open-source packages (in our case, we use shooting and numpy). As the variety gets more complicated (usually as having more defining equations), so will the sampling process. Fortunately, sampling is a trivially parallelizable process, where each single thread only comes with very low processing and memory requirement.

Comparison of the four SOS programs.: We have presented four different formulations for the polynomial ROA problem: inequality constrained multiplier based (IE), equality constrained multiplier based (E), equality constrained
quotient ring based (Q) and equality constrained sampling based (S). Figure 2 summarizes a qualitative comparison of the solution quality and underlying SDP complexity. Note that the relative scale of the gap between different formulation varies case-by-case, for example, the proposed method achieves more significant computational gain for more complex systems; Section VII includes these more quantitative comparison.

V. FORMULATION - LUR’E PROBLEM

Consider system with dynamics subject to Lur’e type sector uncertainty:

\[ f(x) = f_0(x) + \delta(x) \]  

where \( f_0(x) \) is the nominal dynamics and \( \delta(x) \in \{(\alpha(x) - \delta(x))(\beta(x) - \delta(x)) \leq 0\} \) is the uncertainty.

In standard Lur’e settings, \( \alpha(x) \) and \( \beta(x) \) are both linear functions of the state \( x \); they can be easily generalized as polynomial functions here. Figure 3 visualizes a one dimensional example of such uncertainty.

![Fig. 3: Generalized Lur’e type bounded nonlinearity, where the uncertainty \( \delta(x) \in \{(\alpha(x) - \delta(x))(\beta(x) - \delta(x)) \leq 0\} \), where the red curve is \( \alpha(x) = \), and the blue line is \( \beta(x) = \), and the uncertainty \( \delta(x) \) can take any value in the shaded grey area.](image)

A. Existing formulation

We use a linear case ... The standard approach is to first treat \( \delta \) as an indeterminate, independent from \( x \); then using using S-procedure to incorporate the set condition, encoded as inequality constraint. formulate an SDP in the linear case, similar multiplier-based formulation can be written as:

B. Proposed formulation

The proposed formulation is based on two observations. First, the uncertainty \( \delta(x) \) can be written as a convex combination of \( \alpha(x) \) and \( \beta(x) \) for any fixed \( x \). Second, the uncertainty is additive in the dynamics, implying \( V \) is linear with respect to \( \delta(x) \). This allows us to analyze all admissible dynamics by analyzing only the two extreme cases, via the following theorem.

**Theorem 2:** Given \( V(x) \), if the ROA approximation optimization (S) is feasible for both dynamics \( f_\alpha := f_0(x) + \alpha(x) \) and \( f_\beta := f_0(x) + \beta(x) \) where the optimal solutions are \( \rho_\alpha \) and \( \rho_\beta \), then the sub-levelset \( \{x : V(x) \leq \min(\rho_\alpha, \rho_\beta)\} \) satisfies condition Eq. (1) for all admissible \( f \) defined by Eq. (3).

**Proof:** The optimal solutions imply that for all \( x \) in \( \{x : V(x) \leq \min(\rho_\alpha, \rho_\beta)\} \), \( \hat{V}_\alpha(x) := \frac{\partial V(x)}{\partial x} \hat{x} \), and \( \hat{V}_\beta(x) \) similarly defined is negative too. Since \( \hat{V} \) is a convex combination of \( \hat{V}_\beta \) and \( \hat{V}_\alpha \) (with possibly state-dependent convex combination coefficients), we have \( \hat{V} < 0 \).

In addition to preserving all the computational benefit gained in the polynomial problem, the new Lur’e problem formulation, compared with ... provides four additional benefits: (1) elimination of sign-constrained multiplier, (2) elimination of more higher degree constraints (3) elimination of auxiliary indeterminate \( \delta \), and finally, (4) the formulation is less conservative.

As we will show in Subsection , the new formulation may offer new insights into the application too. We delay the discussion there.

VI. FORMULATION - RIGID-BODY PROBLEM

Rigid-body systems are intrinsically challenging to analyze; one major reason is that their dynamics involve the inversion of the mass matrix. The mass matrix include terms like \( \sin(\cdot) \), thus the dynamics is rational trigonometric.

We use the example of N-link pendulum on a cart, shown in Figure 4 to illustrate the main idea. The system has \( 2 + 2N \) states, which are the position and velocity of the cart and the \( N \) links. There are \( N \) inputs, one is a force applied on the cart, and the rest are torques applied on the \( N - 1 \) links, starting from the attached on the cart. The task is to balance all the links upright.

![Fig. 4: N-link pendulum on a cart](image)
from the sampling-variety framework. Let the states be $y = [q_0, q_1, \dot{q}_0, \dot{q}_1]$, where $q_0$ is the position of the cart, $q_1$ the angle of the pole, and $\dot{q}_0, \dot{q}_1$ the corresponding velocities, and let the force on the cart be $u$.

For simplicity, suppose the cart and pole both have mass $m = 1$ and the pole has length $l = 1$, and $g = 9.81$, the EoM is then:

$$
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & -\sin q_1 & 1 \\
0 & 0 & -\sin q_1 & 1
\end{bmatrix}
\begin{bmatrix}
\dot{q}_0 \\
\dot{q}_1 \\
\ddot{q}_0 \\
\ddot{q}_1
\end{bmatrix}
= 
\begin{bmatrix}
\dot{q}_0 \\
\dot{q}_1 \\
\ddot{q}_0 \\
\ddot{q}_1
\end{bmatrix}
= 
\begin{bmatrix}
\dot{q}_0 \\
\dot{q}_1 \\
\dot{q}_0 \cos q_1 + u \\
-9.81 \cos q_1
\end{bmatrix}
$$

(4)

where $M(y)$ is the inertia matrix, $F(y, u)$ the force matrix, and $\dot{y}$ the dynamics in the original coordinate.

A. Existing formulations

Taylor expansion is the commonly used to handle this issue. However, Taylor has two major limitations. One comes from its local nature; intrinsically by expansion error are inevitable. Error-bounding introduces further complexity into the analysis pipeline. The second issue is its own scalability, Taylor expansion itself can be challenging to apply, if higher-orders are needed to reduce the error, the scalability would be even worse.

A recasting technique [6], combined with multipliers for clearing the denominators, have been successfully applied to transform the dynamics into power form.

B. Proposed formulation

Using the recasting techniques, let $x = [q_0, s_1, c_1, \dot{q}_0, \dot{q}_1]$ where $s_1 = \sin q_1$ and $c_1 = \cos q_1$, and suppose a feedback controller $u = u(x)$ is given. This recasting leads to three equality constraints. First, the recast states must satisfy the unit circle condition $s_1^2 + c_1^2 = 1$, which is equivalent to

$$
x' S x = 1
$$

(5)

where $S = \text{diag}(0, 1, 1, 0, 0)$. 

Secondly, simple variable substitution of the recast states $x$ into the EoM Eq. (4) gives:

$$
M(x) \dot{y} = F(x)
$$

(6)

Thirdly, due to the recasting, we also have the dynamics in the original coordinate $\dot{y}$ transformed into the recast coordinate as:

$$
\begin{bmatrix}
\dot{q}_0 \\
\dot{q}_1 \\
\ddot{q}_0 \\
\ddot{q}_1
\end{bmatrix} = 
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
\dot{q}_0 \\
\dot{q}_1 \\
\ddot{q}_0 \\
\ddot{q}_1
\end{bmatrix} = T(x) \dot{y}
$$

where $T(x)$ is the recasting transformation matrix (purely dependent on $x$). Consequentially, the derivative of the Lyapunov function is then

$$
\dot{V}(x) = \frac{\partial V}{\partial x} \dot{x} = \frac{\partial V}{\partial x} T(x) \dot{y}
$$

(7)

In the ‘vanilla’ case where the dynamics is polynomial, the variety has only one component that is $\dot{V} = 0$. Here, due to the recasting, the variety $V$ is much more involved:

$$
V = V_1 \cup V_2 \cup V_3
$$

where the three defining equations are Eq. (5)-(7) (relisted here for convenience):

$$
x' S x = 1
$$

(8a)

$$
M(x) \dot{y} = F(x)
$$

(8b)

$$
\frac{\partial V(x)}{\partial x} \bigg|_x T(x) \dot{y} = 0
$$

(8c)

Note that $\dot{y} = [\dot{q}_0, \dot{q}_1, \ddot{q}_0, \ddot{q}_1]$ may seem necessary to be included in the right hand side basis $n(x)$. However, because the first half of the elements (first-order derivatives of $q$) is included in $x$, whereas the second half (second-order derivatives of $q$) is otherwise independent of $x$, the variety captures all dependencies between $\dot{y}$ and $x$. In other words, it is not necessary for $\dot{y}$ to appear explicitly in the basis, hence the basis notation $n(x)$.

VII. EXAMPLES

A. Polynomial problems

We first consider three polynomial systems: Van der Pol oscillator, Ninja star, and Pendubot. While they all demonstrate speed improvement (up to 2-3 orders faster), Pendubot and Ninja star example also demonstrate superior solution quality. The numerical comparisons are documented in Table I.

a) Van der Pol: Van der Pol oscillator is a two dimensional system of dynamics degree 3. It has a known true ROA, and therefore has been a staple in comparing algorithms for ROA inner-approximation. We set the Lyapunov degree bound $d_L = 6$, and run the bilinear alternation, equality constrained formulation, and the proposed method. Shown in Figure 5, the resulting approximations using the three methods are almost identical, and all very close to the ground truth.

![Fig. 5: True versus verified ROA for Van-der-pol system](image-url)
b) **Ninja star:** To showcase the efficacy of our method in numerically challenging situations, we purposely create a system with known true ROA that resembles a Ninja star, see Figure 6.

The dynamics are very ‘badly conditioned’ despite the low dimensionality. It is of high degree (7-degree polynomial); the coefficients are very unbalanced (relative scale difference is $10^3$); and the dynamics linearization $A$ matrix at the origin are precisely zero.

![Fig. 6: True versus verified ROA for Ninja-star system](image)

### Table 1: Numerical comparison of three method for ROA verification

<table>
<thead>
<tr>
<th>Method</th>
<th>Van der Pol</th>
<th>Ninja star</th>
<th>Pendubot</th>
</tr>
</thead>
<tbody>
<tr>
<td>(IE)</td>
<td>(E)</td>
<td>(S)</td>
<td>(IE)</td>
</tr>
<tr>
<td>PSD variable dim</td>
<td>45</td>
<td>45</td>
<td>15</td>
</tr>
<tr>
<td>num. scalar var.</td>
<td>46</td>
<td>46</td>
<td>1</td>
</tr>
<tr>
<td>num. constraints</td>
<td>214</td>
<td>152</td>
<td>29</td>
</tr>
<tr>
<td>time (sec)</td>
<td>4.98</td>
<td>0.18</td>
<td>0.01</td>
</tr>
</tbody>
</table>

*a equals the number of samples on the variety for method (S)*

*b err " indicates the solver encounters numerical error*

![Fig. 7: $(x_1, x_3)$ slice of the simulated versus verified ROA for Pendubot system](image)

B. **Lur’e problem - Dubins car tracking varying curvature**

Consider a Dubins car defined in the error frame relative to the virtual vehicle along a path to be tracked, illustrated in Figure 8.

![Fig. 8: Virtual error frame for Dubins path tracking](image)

The model is:

$$
\dot{\psi}_E = u_1 - k elementary \psi \\
\dot{X}_E = u_1 Y_E + u_2 - \ell \cos \psi_E \\
\dot{Y}_E = -u_1 X_E + \ell \sin \psi_E
$$

where $\psi_E, X_E, Y_E$ are the angle error and linear displacements, $l$ and $k_\psi$ are the target speed and path curvature, and $u_1$ and $u_2$ are the angular and linear torques. Stabilization at zero error means the car achieves perfect tracking.

The controller gain is designed for a constant nominal tracking curvature $k_\psi = 1$. The true curvature can be between $[0.8, 1.2]$, and potentially time-varying, and the task is to
find an ROA approximation robust to this run-time parameter variation.

Based on Subsection V, since the closed-loop dynamics is affine in \( k_v \), and since the run-time variation on \( k_v \) forms a convex set, we only need to verify two ROAs for the two extreme cases with a common Lyapunov function, and the robust ROA will then be their intersection.

The outer neon yellow is the ROA for when the curvature is 0.8 (straighter path), whereas the inner-tube, thus the intersection, is the ROA for when the curvature is 1.2.

![Fig. 9](image)

Two additional interesting findings: - While it is not necessary for the robust ROA, we analyze under the nominal constant curvature \( k_v = 1 \). The resulting ROA is smaller than when the the curvature is \( k_v = 0.8 \) (though bigger than the other extreme when \( k_v = 1.2 \)); it is sandwiched in between the neon and dark yellow part (not shown in the plot). While our ROAs are approximations (and other possible gaps), it is still interesting that we can verify a larger region in an arguably “easier” environment, than the nominal case.

- Dense sampling (line space of 1e-3) and forward simulation failed to find any counter-example in the regions shown. In other words, all trajectories initialized from the plotted area are stabilized. This seem to indicate our ROA approximation is too loose. However, analytical reasoning on the dynamics lead to three fixed points (other than the origin), one for each of the three \( k_v \) values, represented as red dots in the plot. Eigen analysis shows that these fixed points are non-stable. Sampling fails to find any counter-example because only a measure zero initial states on the stable manifold would ‘trap’ into these bad fixed points; but it demonstrates the necessity of verification and the quality of our ROA.

C. Rigid-body problem - Cart with N-link Pole

Consider the problem of N-link pendulum on a cart, illustrated in Figure 4. The system has \( 2 + 2N \) states, which are the position and velocity of the cart and the \( N \) links. There are \( N \) inputs, one is a force applied on the cart, and the rest are torques applied on the \( N - 1 \) links, starting from the attached on the cart. The task is to balance all the links upright.

The numerical comparisons for different \( N \) are documented in Table II.

VIII. CONCLUSION

Future work include, extend the techniques to more applications. A direct example is the contact example described in [8]. Where the mass matrix can be handled via DAE technique as described in Section VI, contact friction condition can be simplified as described in Section V.

An important issue we did not address is how to find a high quality Lyapunov candidate in the first place. We have a parallel line of work that precisely addresses this issue.

REFERENCES

<table>
<thead>
<tr>
<th>Link</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>states</td>
<td>4</td>
<td>6</td>
<td>8</td>
<td>10</td>
<td>12</td>
<td>14</td>
<td>16</td>
<td>18</td>
</tr>
<tr>
<td>dim(x)</td>
<td>5</td>
<td>8</td>
<td>11</td>
<td>14</td>
<td>17</td>
<td>20</td>
<td>23</td>
<td>26</td>
</tr>
<tr>
<td>dim $Q^a$</td>
<td>21</td>
<td>45</td>
<td>78</td>
<td>120</td>
<td>171</td>
<td>231</td>
<td>300</td>
<td>378</td>
</tr>
<tr>
<td>Constraints$^b$</td>
<td>45</td>
<td>89</td>
<td>179</td>
<td>298</td>
<td>390</td>
<td>531</td>
<td>690</td>
<td>1189</td>
</tr>
<tr>
<td>time (sec)</td>
<td>0.03</td>
<td>0.48</td>
<td>23.67</td>
<td>91.87</td>
<td>178.78</td>
<td>338.56</td>
<td>510.98</td>
<td>738.98</td>
</tr>
</tbody>
</table>

$^a$The matrix variable in the SDP is dim $Q$-by-dim $Q$.
$^b$The same as the number of samples.

**TABLE II:** Numerical results of the ROA problem different number of links on the cart. We first produce an LQR controller and a quadratic Lyapunov function in the original coordinate; then with small-angle approximations, transform them into the $x$ coordinate. Optimization parameters are set as $d_V = 2$, $d_q = 1$, $dn = 4$.

![PSD matrix size n-by-n](image1.png)

![Number of scalar decision variables](image2.png)

![Number of scalar equality constraints](image3.png)

**Fig. 10:** Complexity comparison