Decentralized Polya's Algorithm for Stability Analysis of Large-scale Nonlinear Systems

Reza Kamyar and Matthew M. Peet

Abstract— **In this paper, we introduce an algorithm to decentralize the computation associated with the stability analysis of systems of nonlinear differential equations with a large number of states. The algorithm applies to dynamical systems with polynomial vector fields and checks the local asymptotic stability on hypercubes. We perform the analysis in three steps. First, by applying a multi-simplex version of Polya's theorem to some Lyapunov inequalities, we derive a sequence of stability conditions of increasing accuracy in the form of structured linear matrix inequalities. Then, we design a set-up algorithm to decentralize the computation of the coefficients of the LMIs, among the processing units of a parallel environment. Finally, we use a parallel primal-dual central path algorithm, specifically designed to solve the structured LMIs given by the set-up algorithm. For a sufficiently large number of available processors, the per-core computational complexity of the resulting algorithm is fixed with the accuracy. The algorithm demonstrates a near-linear speed-up in numerical experiments.**

I. INTRODUCTION

Since most systems in the real world are inherently nonlinear, using linear models to describe them may not provide the accuracy required for certain applications. This is the case when the deviation of the system variables from their operating points is large. In addition, common discontinuous nonlinearities in the physical systems such as relay, backlash and hysteresis cannot be locally approximated by linear models. Thus, accurate study and control of real world systems necessitates improvements in the subject of the stability and control of nonlinear models. In this paper, we address the problem of stability of large-scale nonlinear models described by differential equations and defined on hypercubic domains.

The stability of nonlinear systems has been studied thoroughly both in the frequency domain and the time domain. Perhaps the most significant method for the stability analysis of nonlinear systems in time domain is the Lyapunov approach [1]. The Lyapunov-based methods [2] and [3] constitute a comprehensive framework for performing stability analysis and providing stability certificates for closed-loop nonlinear control systems.

In this paper, we seek a systematic method to generate Lyapunov functions. There exist several classical methods for generating Lyapunov functions, e.g., Zubov's method [4], [5] Popov's method [6] and generalized integral methods [7]; all of which address certain mathematical forms for the nonlinear system and the Lyapunov function. In this context, modern methods such as Sum-of-Squares (SOS) [8], [9] and moments [10] use polynomial computing techniques to generate increasingly accurate Lyapunov functions. By augmenting the SOS method with the Positivstellensatz results [11], local stability analysis can be performed on complex geometries such as semi-algebraic sets. In [12], it is shown that exponential stability on bounded sets implies the existence of SOS Lyapunov polynomials with a degree bound. The application of the SOS method in robust stability analysis of nonlinear uncertain systems can be found in [13] and [14].

The SOS method represents the Lyapunov stability conditions in the form of Semi-Definite programming (SDP) relaxations. The conservatism of the stability analysis can be reduced by increasing the degree of the Lyapunov candidate polynomials. However, this results in an exponential increase in the size of the associated SDPs. Also, the size of the SDPs depend polynomially on the state-space dimension. Thus, increasing the accuracy of the analysis of large-scale systems requires high computational complexity and a large amount of memory (typically in the order of terabytes). This makes it impractical to solve large analysis problems with high accuracy using the SOS algorithm on desktop computers.

Parallel computing techniques can be used to perform highly intensive computations on cluster-computers and super-computers. In this technique, chunks of data and computation are distributed among a large number of processors that can communicate through a certain communication architecture. To take the full advantage of the excellent computational capabilities provided by the cluster computers, decentralized algorithms must be developed to distribute the tasks and data among the processors such that synchronization and load balance in the network of processors are maintained. Unfortunately, the SDPs associated with the SOS algorithm do not have a readily distributable structure. In fact, it is proved that Semi-definite programming is P-complete, also known as the class of inherently sequential problems. This means that Semi-definite programs cannot be fully distributed among the processors. For this reason, none of the available parallel SDP solvers [15], [16] can attain perfect speed-up (Amdahl's law).

To avoid Amdahl's law, in this paper we develop a decentralized polynomial computing algorithm based on Polya's theorem [17]. Polya's theorem proves the positivity of homogeneous polynomials on the positive orthant (equivalently on a simplex) by giving a sum of even-powered

Reza Kamyar is a Ph.D student with the Cybernetic Systems and Controls Lab (CSCL), Department of Mechanical Engineering, Arizona State University, Tempe, AZ, 85281 USA, rkamyar@asu.edu

Matthew M. Peet is an assistant professor with the School for Engineering of Matter, Transport and Energy, Arizona State University, Tempe, AZ, 85281 USA, mpeet@asu.edu

monomials representation with a uniform denominator. A variant of Polya's theorem for positivity on a multi-simplex is provided by [18] and its decentralized version applied to robust stability problems with uncertain parameters on the hypercube is proposed in [19]. Several upper-bounds on the number of iterations of Polya's algorithm can be found in [20], [21]. There exist different extensions to Polya's algorithm for positivity on the entire real domain [21] and non-negativity of polynomials with zeros on the vertices [22] and edges [23] of the simplex.

Similar to the SOS algorithm, the accuracy of Polya's algorithm improves with the number and degree of the monomial bases, with cost of an increase in the computational complexity and memory requirements. Unlike the SDPs associated with the SOS algorithm, in this paper we show that the SDPs associated with Polya's algorithm are highly structured and can be mapped to a parallel computing environment [24] with no centralized computation. First, we derive a set of stability conditions in the form of Linear Matrix Inequalities (LMIs) by applying the multi-simplex version of Polya's theorem to the Lyapunov inequalities. Next we develop a fully decentralized asynchronous set-up algorithm that maps the SDP formulation of the LMIs to a set of processors. Finally, we use our primal-dual parallel SDP solver in [25], [24], specifically designed to solve SDPs with block-diagonal structure, to provide the Lyapunov function in the case of stability. Numerical examples show that the proposed algorithm achieves a near-linear speed-up, in particular for the case of systems with large state-space dimension.

II. Notation

We represent a monomial by $x^\gamma = \prod_{i=1}^{l} x_i^{\gamma_i}$, where x is the vector of variables in $\mathbb{R}^l$ and γ is the vector of exponents in $\mathbb{N}^l$. We define $W_d := \left\{\gamma \in \mathbb{N}^l : \sum_{i=1}^{l} \gamma_i = d\right\}$ as the set of exponents of all of the $l-$variate monomials of degree d. We represent homogeneous polynomials of degree d_p as $P(x) = \sum_{h \in W_{d_p}} P_h x^h$, where $P_h \in \mathbb{R}^{n\times n}$ are the matrix coefficients of the monomials of $P(x)$. We denote the n-dimensional unit simplex as

$$\Delta_l := \left\{ x \in \mathbb{R}^l, \sum_{i=1}^{l} x_i = 1, x_i \geq 0 \right\}$$

and the multi-simplex as $\tilde{\Delta}_{\{l_1,\cdots,l_N\}} := \Delta_{l_1} \times \cdots \times \Delta_{l_N}$. We will frequently use the following representation for the class of multi-homogeneous polynomials defined on the multi-simplex:

$$P(x,\tilde{x}) = \sum_{h_1 \in W_{d_{p_1}}} \cdots \sum_{h_N \in W_{d_{p_N}}} P_{\{h_1,\cdots,h_N\}} x_1^{h_{1,1}} \tilde{x}_1^{h_{1,2}} \cdots x_N^{h_{N,1}} \tilde{x}_N^{h_{N,2}},$$

where $d_p := [d_{p_1}, \cdots, d_{p_N}] \in \mathbb{N}^N$ is the degree vector, $(x_i, \tilde{x}_i) \in \Delta_2$ for $i = 1, \cdots, N$ and $h_i \in W_{d_{p_i}}$, where $h_{i,j}$ denotes the j^{th} component f the i^{th} element in $W_{d_{p_i}}$ according to lexicographical ordering. According to the lexicographical ordering, the monomial $x^{h_1} \cdots x^{h_N}$ precedes the monomial $x^{h'_1} \cdots x^{h'_N}$, if the left-most non-zero entry of $h - h' = (h_1 - h'_1, \cdots, h_N - h'_N)$ is positive. For brevity, we denote the coefficients $P_{\{h_1,\cdots,h_N\}}$ by $P_{\mathbf{h_N}}$. We define $n-$dimensional hypercubes as

$$\Phi_{n,r} := \{x \in \mathbb{R}^n : |x_i| \leq r_i,\ i = 1, \cdots, n\}.$$

In [19], for any polynomial $R(x)$, $x \in \Phi_{n,r}$, we constructed the homogenized polynomial denoted by $\mathscr{H}(R)(x,\tilde{x})$, $(x,\tilde{x}) \in \tilde{\Delta}_{\{l_1,\cdots,l_N\}}$, where

$$\{R(x), x \in \Phi_{n,r}\} = \{\mathscr{H}(R)(x,\tilde{x}), (x,\tilde{x}) \in \tilde{\Delta}_{\{l_1,\cdots,l_N\}}\}.$$

The subspace of symmetric matrices in $\mathbb{R}^{n\times n}$ is denoted by $\mathbb{S}_n$ and the cone of positive definite symmetric matrices is denoted by $\mathbb{S}_n^+$. We define the standard basis for $\mathbb{S}_n$ as follows.

$$[E_k]_{i,j} = \begin{cases} 1 & i = j = k \\ 0 & \text{otherwise} \end{cases}, \quad \text{for } k \leq n$$

and $[E_k]_{i,j} = [A_k]_{i,j} + [A_k]_{i,j}^T$, for $k > n$, where

$$[A_k]_{i,j} = \begin{cases} 1 & i = j - 1 = k - n \\ 0 & \text{otherwise.} \end{cases}$$

The canonical basis for $\mathbb{R}^n$ is shown as e_i for $i = 1, \cdots, n$, where $e_i = [0 \ldots 0 \underbrace{1}_{i^{th}} 0 \ldots 0]$. A block-diagonal matrix in $\mathbb{R}^{mn\times mn}$ with diagonal blocks $X_1, \cdots, X_m \in \mathbb{R}^{n\times n}$ is denoted by $\text{diag}(X_1, \cdots, X_m)$. Finally, $\vec{1}_n \in \mathbb{N}^n$ is the vector with all elements equal to 1 and $\mathbf{0}_n$ is the zero matrix in $\mathbb{R}^{n\times n}$.

III. Background and Problem Set-up

In this paper, we address the local stability of nonlinear systems of the form

$$\dot{x}(t) = A(x)x(t), \tag{1}$$

where $A(x) \in \mathbb{R}^{n\times n}$ is a matrix-valued polynomial and $A(0) \neq \mathbf{0}$ (see *Remark* 3). Consider the following well-known Lyapunov result.

Theorem 1: If there exists a continuously differentiable function $V : \mathbb{R}^n \to \mathbb{R}$, such that for some $r \in \mathbb{R}^n$ and continuous positive definite functions W_1, W_2, W_3,

$$W_1(x) \leq V(x) \leq W_2(x) \text{ for all } x \in \Phi_{n,r} \text{ and}$$
$$\nabla V^T f(x) \leq -W_3(x) \text{ for all } x \in \Phi_{n,r},$$

then system (1) is asymptotically stable on $\{x : \{y : V(y) \leq V(x)\} \subset \Phi_{n,r}\}$.

We would like to find the largest $r_i, i = 1, \cdots, n$ for the hypercube $\Phi_{n,r}$ and a Lyapunov function $V(x)$, such that system (1) is asymptotically stable on the largest level-set of $V(x)$ contained in $\Phi_{n,r}$. To avoid intractability, we restrict $V(x)$ to be a polynomial and apply Polya's Theorem [17] to the homogenized form of the Lyapunov inequalities in Theorem (1). This constructs a set of tractable, yet relaxed stability conditions in the form of linear matrix inequalities whose feasibility implies the feasibility of the Lyapunov inequalities. We discuss Polya's Theorem and its associated stability conditions in the following subsection.

A. Polya's Theorem and its associated stability conditions

For every strictly positive homogeneous polynomial defined on the positive orthant, Polya's theorem [17] constructs a rational sum of squared monomials representation with the uniform denominator $(\sum_{i=1}^{n} x_i^2)^d$ for some $d \in \mathbb{N}^+$, hence a certificate for positivity. We use a multi-simplex version of Polya's Theorem as in [18], [19].

Theorem 2: (Polya's Theorem - muti-simplex version) The multi-homogeneous polynomial $F(x) > 0$ on $\tilde{\Delta}_{\{l_1,\cdots,l_N\}}$ if and only if for some sufficiently large d (Polya's exponent),

$$\prod_{i=1}^{N}\left(\sum_{j=1}^{l_i} x_{i,j}\right)^d F(x) \tag{2}$$

has all positive definite coefficients.

The following is the main result that provides the sufficient conditions for stability of system (1).

Theorem 3: Suppose for some $d \geq 0$, there exists a matrix-valued polynomial $P(x) \succ 0$ for all $x \in \Phi_{n,r}$, such that

$$-\prod_{i=1}^{n}\left(X_i+\tilde{X}_i\right)^d \Bigg(\Bigg(\mathscr{H}(\hat{P})(X,\tilde{X}) + \frac{1}{2}\begin{bmatrix}\mathscr{H}(f^T)(X,\tilde{X})\,\mathscr{H}(\frac{\partial \hat{P}}{\partial X_1})(X,\tilde{X}) \\ \vdots \\ \mathscr{H}(f^T)(X,\tilde{X})\,\mathscr{H}(\frac{\partial \hat{P}}{\partial X_n})(X,\tilde{X})\end{bmatrix}^T\Bigg)\mathscr{H}(\hat{A})(X,\tilde{X}) + \mathscr{H}(\hat{A})^T(X,\tilde{X})\Bigg(\mathscr{H}(\hat{P})(X,\tilde{X}) + \frac{1}{2}\begin{bmatrix}\mathscr{H}(f^T)(X,\tilde{X})\,\mathscr{H}(\frac{\partial \hat{P}}{\partial X_1})(X,\tilde{X}) \\ \vdots \\ \mathscr{H}(f^T)(X,\tilde{X})\mathscr{H}(\frac{\partial \hat{P}}{\partial X_n})(X,\tilde{X})\end{bmatrix}\Bigg)\Bigg) \tag{3}$$

has all positive definite coefficients, where

$$\hat{P}(X) := P(f(X)) \quad \hat{A}(X) := A(f(X)), \tag{4}$$

where $X = f^{-1}(x)$ where,

$$f : \{[0,1]\times\cdots\times[0,1]\} \subset \mathbb{R}^n \to \Phi_{n,r},$$

$$f(X) := [f_1(X_1),\cdots,f_n(X_n)]^T, \quad f_i(X_i) := 2r_iX_i - r_i, \tag{5}$$

and where $(X_1,\tilde{X}_1)\times\cdots\times(X_n,\tilde{X}_n) \in \tilde{\Delta}_{\underbrace{\{2,\cdots,2\}}_{n \text{ times}}}$. Then, there exists a polynomial $V(x) : \Phi_{n,r} \to \mathbb{R}^+$ such that $\nabla V(x)^T A(x)x < 0$ for all $x \in \Phi_{n,r}$.

Proof: Choose $V(x) = x^T P(x)x > 0$ on $\Phi_{n,r}$. Then,

$$\nabla V(x)^T A(x)x = x^T\left(2P(x)A(x) + \begin{bmatrix}x^T\frac{\partial P(x)}{\partial x_1}\\ \vdots \\ x^T\frac{\partial P(x)}{\partial x_n}\end{bmatrix}^T A(x)\right)x.$$

$\nabla V(x)^T A(x)x < 0$ for all $x \in \Phi_{n,r}$ if and only if

$$J(x) = A^T(x)P(x) + P(x)A(x) + \frac{1}{2}\left(A^T(x)\begin{bmatrix}x^T\frac{\partial P(x)}{\partial x_1}\\ \vdots \\ x^T\frac{\partial P(x)}{\partial x_n}\end{bmatrix} + \begin{bmatrix}x^T\frac{\partial P(x)}{\partial x_1}\\ \vdots \\ x^T\frac{\partial P(x)}{\partial x_n}\end{bmatrix}^T A(x)\right) \prec 0 \tag{6}$$

for all $x \in \Phi_{n,r}$. Thus, it is enough to show that (6) holds. By utilizing the map (5) and definitions (4) and (5), we have

$$\hat{J}(X) = \hat{A}^T(X)\hat{P}(X) + \hat{P}(X)\hat{A}(X) + \frac{1}{2}\left(\hat{A}^T(x)\begin{bmatrix}f^T(X)\frac{\partial \hat{P}(X)}{\partial X_1}\\ \vdots \\ f^T(X)\frac{\partial \hat{P}(X)}{\partial X_n}\end{bmatrix} + \begin{bmatrix}f^T(X)\frac{\partial \hat{P}(X)}{\partial X_1}\\ \vdots \\ f^T(X)\frac{\partial \hat{P}(X)}{\partial X_n}\end{bmatrix}^T \hat{A}(X)\right),$$

where $X \in \{[0,1]\times\cdots\times[0,1]\} \subset \mathbb{R}^n$. Then, the homogenized $J(X)$ is

$$\mathscr{H}(\hat{J})(X,\tilde{X}) = \Bigg(\Bigg(\mathscr{H}(\hat{P})(X,\tilde{X}) + \frac{1}{2}\begin{bmatrix}\mathscr{H}(f^T)(X,\tilde{X})\,\mathscr{H}(\frac{\partial \hat{P}}{\partial X_1})(X,\tilde{X}) \\ \vdots \\ \mathscr{H}(f^T)(X,\tilde{X})\,\mathscr{H}(\frac{\partial \hat{P}}{\partial X_n})(X,\tilde{X})\end{bmatrix}^T\Bigg)\mathscr{H}(\hat{A})(X,\tilde{X}) + \mathscr{H}(\hat{A})^T(X,\tilde{X})\Bigg(\mathscr{H}(\hat{P})(X,\tilde{X}) + \frac{1}{2}\begin{bmatrix}\mathscr{H}(f^T)(X,\tilde{X})\,\mathscr{H}(\frac{\partial \hat{P}}{\partial X_1})(X,\tilde{X}) \\ \vdots \\ \mathscr{H}(f^T)(X,\tilde{X})\mathscr{H}(\frac{\partial \hat{P}}{\partial X_n})(X,\tilde{X})\end{bmatrix}\Bigg)\Bigg) \tag{7}$$

where $(X_1,\tilde{X}_1)\times\cdots\times(X_n,\tilde{X}_n) \in \tilde{\Delta}_{\{2,\cdots,2\}}$. From the Theorem assumption, there exists an exponent $d \geq 0$ such that $-\prod_{i=1}^{n}\left(X_i+\tilde{X}_i\right)^d \mathscr{H}(\hat{J})(X,\tilde{X})$ has all positive-definite coefficients. Thus, by using Theorem 2 (necessity)

$$\mathscr{H}(\hat{J})(X,\tilde{X}) \prec 0.$$

Since

$$\{J(x),\, x \in \Phi_{n,r}\} = \{\mathscr{H}(\hat{J})(X,\tilde{X}),\, (X_1,\tilde{X}_1)\times\cdots\times(X_n,\tilde{X}_n) \in \tilde{\Delta}_{\{2,\cdots,2\}}\},$$

we conclude that $J(x) \prec 0$ for all $x \in \Phi_{n,r}$. Thus $\nabla V(x)^T A(x)x < 0$ for all $x \in \Phi_{n,r}$. ■

Remark 3: Suppose $A(0) = \mathbf{0}$. Then from (4), $\hat{A}(X)$ will have a zero at $X = \frac{1}{2}\overrightarrow{1_n}$. Then, $\mathscr{H}(\hat{A})(X,\tilde{X})$ will have a zero in $int(\tilde{\Delta}_{\{2,\cdots,2\}})$. Then, according to (7), $\mathscr{H}(\hat{J})(X,\tilde{X})$ will have a zero in $int(\tilde{\Delta}_{\{2,\cdots,2\}})$. Because for any homogeneous polynomial with zeros in the interior of the simplex, there exists no Polya's exponent [22], it concludes that there exists no $d \geq 0$ such that $-\prod_{i=1}^{n}\left(X_i+\tilde{X}_i\right)^d \mathscr{H}(\hat{J})(X,\tilde{X})$ has all positive-definite coefficients.

B. LMI formulation of the stability conditions

In order to verify the stability of system (1), we search for $P(x) \in \mathbb{S}_+^n$ that satisfy the assumptions of Theorem (3), restated as follows.

$$\prod_{i=1}^{n}\left(X_i+\tilde{X}_i\right)^{d_1}\left(\mathscr{H}(\hat{P})(X,\tilde{X})\right) \qquad \text{and} \tag{8}$$

$$-\prod_{i=1}^{n}\left(X_i+\tilde{X}_i\right)^{d_2}\left(\left(\mathscr{H}(\hat{P})(X,\tilde{X})+\frac{1}{2}M^T(X,\hat{X})\right)\mathscr{H}(\hat{A})(X,\tilde{X})\right.$$
$$\left.+\mathscr{H}(\hat{A})^T(X,\tilde{X})\left(\mathscr{H}(\hat{P})(X,\tilde{X})+\frac{1}{2}M(X,\hat{X})\right)\right) \quad (9)$$

have positive-definite coefficients, where for brevity we used

$$M(X,\tilde{X}) := \begin{bmatrix} \mathscr{H}(f^T)(X,\tilde{X})\,\mathscr{H}(\frac{\partial \hat{P}}{\partial X_1})(X,\tilde{X}) \\ \vdots \\ \mathscr{H}(f^T)(X,\tilde{X})\,\mathscr{H}(\frac{\partial \hat{P}}{\partial X_n})(X,\tilde{X}) \end{bmatrix} \in \mathbb{R}^{n\times n}.$$

Let $P(x), x \in \Phi_{n,r}$ be of degree vector $d_p \in \mathbb{N}^N$, with N_p monomials with unknown coefficients $P_1,\cdots,P_{N_p} \in \mathbb{S}_n^+$ in lexicographical ordering. Using the homogenization procedure in [19], $M(X,\tilde{X})$ can be represented in the standard multi-homogeneous format defined Section II as

$$M(X,\tilde{X}) = \sum_{h_1\in W_{d_{M_1}}}\cdots\sum_{h_N\in W_{d_{M_N}}} M_{\mathbf{h}_N} X_1^{h_{1,1}}\tilde{X}_1^{h_{1,2}}\cdots X_N^{h_{N,1}}\tilde{X}_N^{h_{N,2}}, \quad (10)$$

where for brevity we denote $M_{\mathbf{h}_N}(P_1,\cdots,P_{N_p})$ by $M_{\mathbf{h}_N}$ and where $(X_1,\tilde{X}_1)\times\cdots\times(X_n,\tilde{X}_n)\in\tilde{\Delta}_{\{2,\cdots,2\}}$.
Likewise, Given $A(x), x \in \Phi_{n,r}$ of degree vector $d_a \in \mathbb{N}^N$, N_a monomials with coefficients $A_1,\cdots,A_{N_a}$ in lexicographical ordering, we can represent $A(x)$ as

$$\mathscr{H}(\hat{A})(X,\tilde{X}) = \sum_{h_1\in W_{d_{a_1}}}\cdots\sum_{h_N\in W_{d_{an}}} A_{\mathbf{h}_N} X_1^{h_{1,1}}\tilde{X}_1^{h_{1,2}}\cdots X_N^{h_{N,1}}\tilde{X}_N^{h_{N,2}}, \quad (11)$$

where for brevity we denote $A_{\mathbf{h}_N}(A_1,\cdots,A_{N_a})\in\mathbb{R}^{n\times n}$ by $A_{\mathbf{h}_N}$ and where $(X_1,\tilde{X}_1)\times\cdots\times(X_n,\tilde{X}_n)\in\tilde{\Delta}_{\{2,\cdots,2\}}$. Likewise, we represent $P(x), x\in\Phi_{n,r}$ as

$$\mathscr{H}(\hat{P})(X,\tilde{X}) = \sum_{h_1\in W_{d_{p_1}}}\cdots\sum_{h_N\in W_{d_{pN}}} P_{\mathbf{h}_N} X_1^{h_{1,1}}\tilde{X}_1^{h_{1,2}}\cdots X_N^{h_{N,1}}\tilde{X}_N^{h_{N,2}}, \quad (12)$$

where we denote the unknown coefficients $P_{\mathbf{h}_N}(P_1,\cdots,P_{N_p})\in\mathbb{S}_n^+$ by $P_{\mathbf{h}_N}$ and $(X_1,\tilde{X}_1)\times\cdots\times(X_n,\tilde{X}_n)\in\tilde{\Delta}_{\{2,\cdots,2\}}$.

By substituting for $\mathscr{H}(\hat{P})(X,\tilde{X})$ in (8) from (12) and calculating the coefficients of the resulting monomials, the first stability condition can be expressed as the following LMI with unknown variables $P_{\mathbf{n}_N}$.

$$\sum_{h_1\in W_{d_{p_1}}}\cdots\sum_{h_N\in W_{d_{pN}}}\left(\beta_{\{\mathbf{h}_N,\Gamma_N\}}P_{\mathbf{h}_N}\right)\succ 0, \quad (13)$$

where Γ_N denotes $\{\gamma_1,\cdots,\gamma_N\}$, where $\gamma_1\in W_{d_{p_1}+d_1},\cdots,\gamma_N\in W_{d_{pN}+d_1}$ and where $\mathbf{h}_N$ denotes $\{h_1,\cdots,h_N\}$. For some $\mathbf{h}_N$ and Γ_N, we define $\beta_{\{\mathbf{h}_N,\Gamma_N\}}\in\mathbb{N}$ as the coefficient of $P_{\mathbf{h}_N}$ in the monomial with the exponent vector $[\gamma_1,\cdots,\gamma_N]$. Similarly, by substituting for $M(X,\tilde{X})$, $\mathscr{H}(\hat{A})(X,\tilde{X})$ and $\mathscr{H}(\hat{P})(X,\tilde{X})$ in (9) from (10), (11) and (12), and calculating the coefficients of the resulting monomials, the second stability condition can be expressed as

$$\sum_{h_1\in W_{d_{p_1}}}\cdots\sum_{h_N\in W_{d_{pN}}}\left(H^T_{\{\mathbf{h}_N,\Gamma_N\}}(P_{\mathbf{h}_N}+\frac{1}{2}M^T_{\mathbf{h}_N})\right.$$
$$\left.+(P_{\mathbf{h}_N}+\frac{1}{2}M_{\mathbf{h}_N})H_{\{\mathbf{h}_N,\Gamma_N\}}\right)\prec 0, \quad (14)$$

where $h_1\in W_{d_{P_1}},\cdots,h_N\in W_{d_{P_N}}$ and $\gamma_1\in W_{d_{pa_1}+d_2},\cdots,\gamma_N\in W_{d_{pa_N}+d_2}$, where $d_{pa_i}:=d_{p_i}+d_{a_i}$. For some $\mathbf{h}_N$ and Γ_N, we define $H_{\{\mathbf{h}_N,\Gamma_N\}}$ as the coefficient of $P_{\mathbf{h}_N}$ in the monomial with the exponent vector $[\gamma_1,\cdots,\gamma_N]$. Recursive formulas for calculating β and H coefficients are given in [19].

In the following section, we define a sequence of structured SDPs associated with the LMIs in (13) and (14). Then we propose a decentralized set-up algorithm to construct this sequence.

C. SDP formulation of the stability conditions

We solve the LMIs associated with Theorem (3) by means of semi-definite programming. In this section, we introduce a sequence of dual SDP formulations of the LMIs in (13) and (14) with constraints of block-diagonal structure. The sequence is indexed by Polya's exponent d, where for simplicity we consider the case $d_1=d_2=d$.

We define the elements of the sequence $\{\text{SDP}_d\}_{d=0}^{\infty}$ as follows.

$$\text{SDP}_d: \quad \min_{y^{(d)},Z^{(d)}} \; a^T y^{(d)}$$
$$\text{subject to} \quad \sum_{i=1}^{K} y_i^{(d)}B_i^{(d)} - C^{(d)} = Z^{(d)}$$
$$Z^{(d)}\succcurlyeq 0,\; y^{(d)}\in\mathbb{R}^K$$

where $Z^{(d)}\in\mathbb{S}_m^+\cup\{\mathbf{0}\}$ and $y^{(d)}\in\mathbb{R}^K$ are the dual variables. We define the elements of SDP_d, i.e., $C^{(d)}, B_i^{(d)}, a$ as follows. The element $C^{(d)}$ is

$$C^{(d)} := \text{diag}(C_1^{(d)},\cdots C_L^{(d)}, C_{L+1}^{(d)},\cdots C_{L+M}^{(d)}), \quad (15)$$

where

$$L=\prod_{i=1}^{n}\frac{(d_{p_i}+d_1+1)!}{(d_{p_i}+d_1)!} \quad (16)$$

is the number of monomials in (8) and

$$L'=\prod_{i=1}^{n}\frac{(d_{p_i}+d_{a_i}+d_2+1)!}{(d_{p_i}+d_{a_i}+d_2)!} \quad (17)$$

is the number of monomials in (9) and $C_j^{(d)}$ can be calculated as in [19].

For $i=1,\cdots,K$, define the elements $B_i^{(d)}$ as

$$B_i^{(d)} = \text{diag}(B_{i,1}^{(d)},\cdots,B_{i,L}^{(d)},B_{i,L+1}^{(d)},\cdots,B_{i,L+L'}^{(d)}), \quad (18)$$

where

$$K=\frac{n(n+1)}{2}\prod_{i=1}^{n}\frac{(d_{p_i}+1)!}{d_{p_i}!}, \quad (19)$$

is the dimension of the dual variable y and where for $1\le j\le L$,

$$B_{i,j}^{(d)} = \sum_{h_1\in W_{d_{p_1}}}\cdots\sum_{h_N\in W_{d_{pN}}}\beta_{\{\mathbf{h}_N,\Gamma_{N,j}\}}U_{\mathbf{h}_N}(e_i) \quad (20)$$

and for $L+1\le j\le L+L'$,

$$B_{i,j}^{(d)} = \sum_{h_1\in W_{d_{p_1}}}\cdots\sum_{h_N\in W_{d_{pN}}}\left(H^T_{\{\mathbf{h}_N,\Gamma_{N,j-L}\}}\left(U_{\mathbf{h}_N}(e_i)+\frac{1}{2}V^T_{\mathbf{h}_N}(e_i)\right)\right.$$
$$\left.+\left(U_{\mathbf{h}_N}(e_i)+\frac{1}{2}V_{\mathbf{h}_N}(e_i)\right)H_{\{\mathbf{h}_N,\Gamma_{N,j-L}\}}\right), \quad (21)$$

where recall that $\Gamma_{N,j}:=\{\gamma_{1_j},\cdots,\gamma_{N_j}\}$, where γ_{i_j} is the j^{th} element of $W_{d_{M_i}+d}$ using lexicographical ordering, and

$$U_{\mathbf{h}_N}(z) := P_{\mathbf{h}_N}(D_1(z),\cdots,D_{N_p}(z)) \quad (22)$$

and

$$V_{\mathbf{h}_N}(z) := M_{\mathbf{h}_N}(D_1(z), \cdots, D_{N_p}(z)), \tag{23}$$

where

$$D_k(z) := \sum_{l=1}^{\tilde{N}} E_l \, z_{l+\tilde{N}(k-1)}, \tag{24}$$

where recall form Section II that E_k is the basis of $\mathbb{S}_n$ and $\tilde{N} := \frac{n(n+1)}{2}$. By setting $a = \vec{1} \in \mathbb{R}^K$, The definition of SDP_d is complete.

Finally, we provide a pseudo-code for Polya's algorithm to find an approximation of the region of attraction (in case of stability) of system (1).

Algorithm 1: Polya's algorithm

Inputs: Number of monomials in $A(x) : N_a$, Coefficients of $A(x) : A_i,\ i = 1, \cdots, N_a$, degree vector of $A(x) : d_a$, exponent vectors of $P(x) : e_i$, Number of monomials in $P(x) : N_p$, upper-bound on Polya's exponent: d_{max}, lower- and upper-bounds on the size of $\Phi_{n,r} : r_0, r_1$.

Bisection search on r:
while $d < d_{max}$ **do**
  Homogenize $A(x)$ and calculate $A_{\mathbf{h}_N}$ as in [19]
  Calculate $U_{\mathbf{h}_N}$ and $V_{\mathbf{h}_N}$ using (22), (23)
  Calculate $\beta^{(d)}$ and $H^{(d)}$ coefficients as in [19]
  Construct SDP_d using (15)-(21)
  if *SDP_d is feasible* **then**
    Break while loop
  Set $d = d + 1$
Calculate $P_i = \sum_{l=1}^{\tilde{N}} E_l \, x_{l+\tilde{N}(k-1)}$ for $i = 1, \cdots, N_p$
Calculate $P(x) = \sum_{i=1}^{N_p} P_i x^{e_i}$

Outputs: Lyapunov function: $V = x^T(t)P(x)x(t)$ and the Maximum size of $\Phi_{n,r} : r_{max}$ such that (1) is stable on $\{x : \{y : V(y) \le V(x)\} \subset \Phi_{n,r_{max}}\}$

In the following section, we provide a parallel implementation for the set-up algorithm (steps 1 to 4 of the while loop in Algorithm 1).

IV. Parallel Implementation

To construct the sequence $\{\text{SDP}_d\}_{d=0}^{\infty}$, we first develop an asynchronous parallel set-up algorithm with no centralized data and computation. Then the parallel SDP solver in [24] which is specifically designed to solve SDPs with a block-diagonal structure, will be used to perform the last step of Algorithm 1. A simplified pseudo-code for the proposed decentralized set-up algorithm is as follows. The speed-up results of the algorithm are provided in Example 2 of Section V and a parallel C++ implementation of Algorithm 2 is available at `www.sites.google.com/a/asu.edu/kamyar/software`.

Algorithm 2: Decentralized set-up algorithm

Inputs: No. of processors: N_c, Coefficients of $A : A_i$, degrees and No. of monomials in A and P: d_a, d_p, N_a, N_p
Initialization:
for $i = 1, \cdots, N_c$, *processor i* **do**
  Set $d = 0$, $d_{pa} = d_p + d_a$
  Calculate L and L' using (16), (17).
  Calculate per-core number of monomials:
    $\overline{N}_p = \texttt{floor}(N_p/N_c)$, $\overline{N}_a = \texttt{floor}(N_a/N_c)$,
    $\overline{L} = \texttt{floor}(L/N_c)$, $\overline{L'} = \texttt{floor}(L'/N_c)$
  for $\gamma_{(i-1)\overline{L}+1}, \cdots, \gamma_{i\overline{L}}$ *and* $h_1, \cdots, h_L$ **do**
    Calculate $\beta^{(0)}_{\{\mathbf{h}_N, \Gamma_N\}}$ as in [19]
  for $\gamma_{(i-1)\overline{L'}+1}, \cdots, \gamma_{i\overline{L'}}$ *and* $h_1, \cdots, h_L$ **do**
    Calculate $H^{(0)}_{\{\mathbf{h}_N, \Gamma_N\}}$ as in [19]
Polya's iterations:
for $i = 1, \cdots, N_c$, *processor i* **do**
  for $d = 1, \cdots, d_{max}$ **do**
    Set $d_p = d_p + \overrightarrow{1_n}$. Update L and $\overline{L}$
    Set $d_{pa} = d_{pa} + \overrightarrow{1_n}$. Update L' and $\overline{L'}$
    for $\gamma_{(i-1)\overline{L}+1}, \cdots, \gamma_{i\overline{L}}$ *and* $h_1, \cdots, h_L$ **do**
      Update $\beta^{(d)}_{\{\mathbf{h}_N, \Gamma_N\}}$ as in [19]
    for $\gamma_{(i-1)M'+1}, \cdots, \gamma_{iM'}$ *and* $h_1, \cdots, h_L$ **do**
      Update $H^{(d)}_{\{\mathbf{h}_N, \Gamma_N\}}$ as in [19]
  Calculating the SDP elements:
  for $j = (i-1)\overline{L} + 1, \cdots, i\overline{L}$ **do**
    Calculate $C^{(d)}_j$ as in [19]
  for $j = 1, \cdots, K$ **do**
    for $k = (i-1)\overline{L}, \cdots, i\overline{L}$ **do**
      Calculate $B^{(d)}_{j,k}$ using (20)
    for $k = L + (i-1)\overline{L'} + 1, \cdots, L + i\overline{L'}$ **do**
      for $l = (i-1)\overline{N}_p, \cdots, i\overline{N}_p$ **do**
        Calculate $D_l(e_j)$ using (24)
      Calculate $B^{(d)}_{j,k}$ using (21).
Outputs: processor i returns the elements:
$C^{(d)}_j$ for $j = (i-1)\overline{L} + 1, \cdots, i\overline{L}$, $B^{(d)}_{j,k}$ for $j = 1, \cdots, K$ and $k = (i-1)\overline{L}, \cdots, i\overline{L}, L + (i-1)\overline{L'} + 1, \cdots, L + i\overline{L'}$

V. Numerical Results

In this section, we assess the conservatism and speed-up of the developed algorithms in two numerical examples.

Example 1: Accuracy
In this example, we show how the accuracy of the algorithm in approximating the region of attraction of a typical nonlinear system improves by increasing the degree of $P(x)$ in the Lyapunov function $V(x) = x^T P(x) x$. Consider the Van der Pol oscillator in reverse time, modeled as

$$\dot{x}_1(t) = -x_2(t), \ \ \dot{x}_2(t) = x_1(t) + x_2(t)\left(x_1^2(t) - 1\right)$$

and the hypercubes $\Phi_{2,r_1}, \cdots, \Phi_{2,r_4}$, where $r_1 = [1,1], r_2 = [1.5,1.5], r_3 = [1.7,1.8], r_4 = [1.9,2.4]$. For the hypercube of radius $r_i, i = 1, \cdots, 4$, we solved the problem

$$\begin{aligned} &\min \lambda \quad \text{subject to} \\ &V(x) = x^T P(x) x > 0, \text{ for all } x \in \Phi_{2,r_i} \text{ and with } d_p = [\lambda, \lambda] \\ &\dot{V} < 0 \quad \text{for all } x \in \Phi_{2,r_i} \end{aligned}$$

using a bisection search on λ in an outer-loop, and Algorithm 1 in the inner-loop. Recall that $d_p \in \mathbb{N}^2$ is the degree vector of $P(x)$. The thick curve in Fig. 1 is the region of attraction of the Van der Pol equation. For each hypercube of size r_i, we have shown the largest inscribed level-set of the Lyapunov function and its corresponding d_p which are found by solving the above optimization problem. The figure suggests that increasing d_p results in less conservative Lyapunov functions and better approximations for the region of attraction.

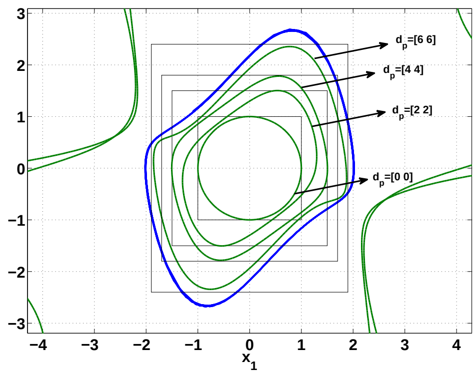

Fig. 1. Largest invariant sets of the Lyapunov functions $V = x^T P(x) x$ associated with hypercubes of different sizes

Example 2: Speed-up

In this example, we ran the decentralized set-up algorithm for three random nonlinear systems of state-space dimensions $n = 10, 20$ and 50. Polya's exponent is $d = 1$ and $A(x)$ has degree 2 in all the cases. We used a linux-based Karlin cluster computer at Illinois Institute of Technology. Fig. 2 shows the speed-up of the algorithm versus the number of processors. An algorithm is scalable if the speed-up versus the number of the processors shows a linear trend. It is observed that the speed-up of the algorithm is closer to the ideal linear speed-up for larger state-space dimensions. So the algorithm achieves a better scalability for systems of higher dimensions.

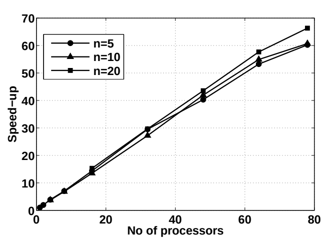

Fig. 2. Speed-up of the decentralized set-up algorithm

VI. Conclusion and Future Works

A decentralized algorithm based on Polya's theorem is proposed to solve the stability problem of systems with large number of nonlinear differential equations. The states are assumed to lie on a hypercube. The numerical experiments show the excellent scalability of the algorithm. The accuracy of the algorithm is assessed in a numerical example for different degrees of the Lyapunov functions and Polya's exponents. The method can be readily extended to solve nonlinear robust stability problems, by applying Polya's algorithm on the parameter-dependent Lyapunov conditions with parameters on the hypercube. Also, the algorithm can be used to perform a decentralized computation for robust controller synthesis, by setting-up and solving the LMIs associated with H_2 and H_∞ synthesis. Decentralized compuytation for stability analysis on more complicated geometries such as convex polytopes will be studied in future.

Acknowledgments

The authors would like to gratefully acknowledge funding from NSF Grant # CMMI-1100376.

References

[1] A. M. Lyapunov, "The General Problem of Stability of Motion," *A.T. Fuller, trans. and ed, Taylor and Francis*, 1992.
[2] W. Haddad and V. Chellaboina, *Nonlinear Dynamical Systems and Control: A Lyapunov-based Approach*. Princeton Univ. Press, 2008.
[3] H. K. Khalil, *Nonlinear Systems*. Prentice Hall, 2001.
[4] V. I. Zubov, "Methods of A. M. Lyapunov and Their Applications," *Gronigen, The netherlands: P. Noordhoff*, 1964.
[5] E. Davison and E. Kurak, "A computational method for determining quadratic lyapunov functions for non-linear systems," *Automatica*, vol. 7, no. 5, pp. 627–636, 1971.
[6] E. P. Popov, *Applied theory of control of nonlinear systems*. Verlag Technik, Berlin, 1964.
[7] P. S. CHIN, "Generalized integral method to derive lyapunov functions for nonlinear systems," *International Journal of Control*, vol. 46, no. 3, pp. 933–943, 1987.
[8] P. Parrilo, *Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimization*. PhD thesis, California Institute of Technology, 2009.
[9] P. Rajna, A. Papachristodoulou, and P. A. Parrilo, "Introducing SOSTOOLS: a general purpose sum of squares programming solver," in *Proceedings of IEEE Conference on Decision and Control*, 2002.
[10] J. B. Lasserre, "Global optimization with polynomials and the problem of moments," *SIAM Journal on Optim.*, vol. 11, pp. 796–817, 2001.
[11] G. Stengle, "A nullstellensatz and a positivstellensatz in semialgebraic geometry," *Mathematische Annalen*, vol. 207, no. 2, pp. 87–97, 1973.
[12] M. M. Peet and A. Papachristodoulou, "A converse sum of squares lyapunov result with a degree bound," *Automatic Control, IEEE Transactions on*, vol. 57, no. 9, pp. 2281–2293, 2012.
[13] E. M. Aylward, P. A. Parrilo, and J.-J. E. Slotine, "Stability and robustness analysis of nonlinear systems via contraction metrics and sos programming," *Automatica*, vol. 44, no. 8, pp. 2163–2170, 2008.
[14] U. Topcu and A. Packard, "Stability region analysis for uncertain nonlinear systems," in *Decision and Control, 2007 46th IEEE Conference on*, pp. 1693–1698, IEEE, 2007.
[15] B. Borchers and J. G. Young, "Implementation of a primal dual method for SDP on a shared memory parallel architecture," *Computational Optimization and Applications*, vol. 37, no. 3, pp. 355–369, 2007.
[16] M. Yamashita, K. Fujisawa, and M. Kojima, "SDPARA: Semidefinite programming algorithm parallel version," *Parallel Computing*, vol. 29, pp. 1053–1067, 2003.
[17] G. Hardy, J. E. Littlewood, and G. Pólya, *Inequalities*. Cambridge University Press, 1934.
[18] R. C. L. F. Oliveira, P.-A. Bliman, and P. L. D. Peres, "Robust LMIs with parameters in multi-simplex: Existence of solutions and applications," pp. 2226–2231, Proceedings of IEEE Conference on Decision and Control, 2008.
[19] R. Kamyar and M. M. Peet, "Decentralized computation for robust stability of large-scale systems with parameters on the hypercube," *IEEE conference on decision and control*, 2012.
[20] V. Powers and B. Reznick, "A new bound for polya's theorem with applications to polynomials positive on polyhedra," *Journal of Pure and Applied Algebra*, vol. 164, no. 1, pp. 221–229, 2001.
[21] J. A. De Loera and F. Santos, "An effective version of pólya's theorem on positive definite forms," *Journal of Pure and Applied Algebra*, vol. 108, no. 3, pp. 231–240, 1996.
[22] V. Powers and B. Reznick, "A quantitative pólya's theorem with corner zeros," in *Proceedings of the 2006 international symposium on Symbolic and algebraic computation*, pp. 285–289, ACM, 2006.
[23] M. Castle, V. Powers, and B. Reznick, "A quantitative pólyas theorem with zeros," *Journal of Symbolic Computation*, vol. 44, no. 9, pp. 1285–1290, 2009.
[24] R. Kamyar, M. Peet, and Y. Peet, "Solving large-scale robust stability problems by exploiting the parallel structure of polyas theorem," *IEEE Transactions on Automatic Control*, vol. 58, pp. 1931–1947, 2013.
[25] R. Kamyar and M. M. Peet, "Decentralized computation for robust stability analysis of large state-space systems using polya's theorem," in *Proceedings of the American Control Conference*, Jun-Jul 2012.