Electrical Engineering (2018) 100:2205–2216
https://doi.org/10.1007/s00202-018-0690-z

ORIGINAL PAPER

Estimation of regions of attraction of power systems by using sum of squares programming

Shinsaku Izumi[1] · Hiroki Somekawa[2] · Xin Xin[1] · Taiga Yamasaki[1]

Received: 2 October 2017 / Accepted: 4 April 2018 / Published online: 3 May 2018

Abstract
This paper presents an algorithm estimating the regions of attraction of power systems based on the Lyapunov function approach where a sublevel set of a Lyapunov function for a target system is used as the estimate. In particular, we focus here on the algorithm based on sum of squares (SOS) programming, which has been recently proposed, and aim to develop a simpler algorithm for the practical use. For this aim, we present an algorithm overcoming the difficulty of the SOS programming problem addressed in the existing study, i.e., the bilinear constraints, in a simpler way. In the proposed algorithm, two SOS programming problems are iteratively solved, and the number of the problems solved at each iteration is reduced to half of that in the existing algorithm. In addition, we theoretically analyze the proposed algorithm, and show the convergence under certain conditions. The performance of our algorithm is demonstrated by numerical examples.

Keywords Power systems · Transient stability · Regions of attraction · Lyapunov functions · Sum of squares

1 Introduction

Stability analysis of power systems has been a major topic in the field of power engineering. This is because analyzing the stability of power systems leads to the safe and efficient operation of them. In fact, if the stability is not analyzed, the impact of accidents on the stability cannot be estimated and systems may unexpectedly become unstable. Moreover, in such a case, we have to make the operation of systems conservative, which decreases the efficiency.

A typical type of the stability of power systems is transient stability [13]. The transient stability is the ability of power systems to maintain the synchronization of generators for transient disturbances such as faults and the loss of a portion of transmission networks. An index to evaluate the transient stability is the size of the *region of attraction* (ROA). The ROA is the set of all system states from which the system converges to an equilibrium state. This is illustrated in Fig. 1. If the state after the disturbances is in the ROA, then it again converges to the equilibrium state. That is, the larger ROA means the higher transient stability.

A straightforward method to investigate the ROA is time-domain simulation [11] with differential equations describing system dynamics. This provides the exact ROA, but is impractical for large-scale systems due to high computational costs. A promising method to solve this problem is the *Lyapunov function* approach, i.e., to find a Lyapunov function for a target system and use the sublevel set as an estimate of the ROA. This method does not require computing the trajectory of the system for each initial state, and thus its computational cost is low. Motivated by this, many studies have been conducted so far. For example, much effort has been devoted to studying methods utilizing energy functions [4,5,9,15]. Also, there are studies on methods utilizing nonlinear control theory [10], extended Lyapunov functions [3], and numerical optimization [1].

Among these studies, [1] has proposed an algorithm calculating a Lyapunov function for a given power system by using

This work was partly supported by JSPS KAKENHI Grant Numbers 26420425 and 16K18124.

✉ Shinsaku Izumi
izumi@cse.oka-pu.ac.jp

Xin Xin
xxin@cse.oka-pu.ac.jp

Taiga Yamasaki
taiga@cse.oka-pu.ac.jp

[1] Faculty of Computer Science and Systems Engineering, Okayama Prefectural University, 111 Kuboki, Soja, Okayama 719-1197, Japan

[2] Mitsubishi Motors Corporation, 1-1 Mizushimakaigandori, Kurashiki, Okayama 712-8501, Japan

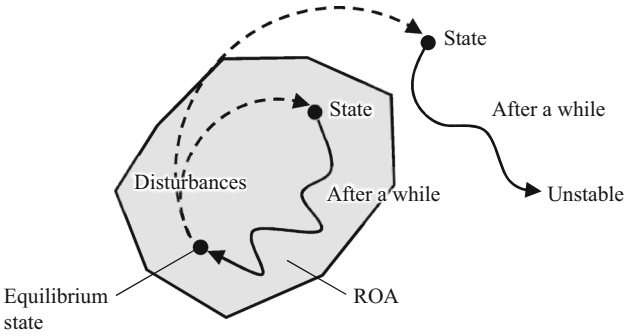

Fig. 1 Region of attraction (ROA)

sum of squares (SOS) programming [6,7]. This algorithm is promising for the following two reasons. First, the algorithm is available for power systems with transfer conductances. Transfer conductances correspond to losses in power systems, and cannot be neglected in practice. However, existing studies have assumed that transfer conductances are zero [5,9,15] or small [3,4]. Meanwhile, the algorithm in [1] is available without assumptions on transfer conductances. Second, the algorithm in [1] enables us to systematically obtain Lyapunov functions. As a result, the time and effort spent to find Lyapunov functions are reduced. On the other hand, the algorithm in [1] is complicated. In fact, the algorithm is composed of two loops, and we have to iteratively solve four different types of SOS programming problems in a loop. Such complexity is undesirable for the users because the time and effort spent to understand and implement the algorithm increase. Moreover, the complexity makes the analysis of the algorithm difficult. This leads to the lack of the theoretical guarantees of the algorithm. In fact, [1] has not provided a theoretical result on the behavior (e.g., the convergence) of the algorithm.

The purpose of this paper is to develop a simpler algorithm estimating the ROA than the algorithm in [1]. For this purpose, we make two contributions. First, we present an improved algorithm. In our algorithm, two SOS programming problems are iteratively solved, and the number of the problems solved at each iteration is reduced to half of that in the algorithm in [1]. The key idea is to consider a simpler method to overcome the difficulty of the SOS programming problem addressed in [1]. The problem addressed in [1] includes bilinear constraints, and thus cannot be efficiently solved by numerical optimization techniques. To overcome this difficulty, [1] has introduced four subproblems, while we overcome it by only two subproblems. Second, by utilizing the simple structure of the proposed algorithm, we analyze it and clarify its behavior. As a result, we guarantee that there exist solutions to the two SOS programming problems solved at each iteration, and show that the algorithm converges under certain conditions. This provides a theoretical guarantee for the proposed algorithm.

Notation: Let $\mathbb{R}$, $\mathbb{R}_+$, and $\mathbb{R}_{0+}$ be the real number field, the set of positive real numbers, and the set of nonnegative real numbers, respectively. Both the zero scalar and the zero vector are expressed by 0. For the number $c \in \mathbb{R}$ and the function $f : \mathbb{R}^n \to \mathbb{R}$, $\mathbb{L}_c(f(x))$ denotes the sublevel set of f, i.e., $\mathbb{L}_c(f(x)) := \{x \in \mathbb{R}^n \mid f(x) \leq c\}$. We denote by $\mathbb{P}$ the set of polynomials. Moreover, for $x \in \mathbb{R}^n$, let $\mathbb{P}_0 := \{p(x) \in \mathbb{P} \mid p(0) = 0\}$, and let $\mathbb{P}_+$ be the set of positive definite polynomials, i.e., $\mathbb{P}_+ := \{p(x) \in \mathbb{P}_0 \mid p(x) > 0 \ \forall x \in \mathbb{R}^n \setminus \{0\}\}$. Finally, we use $\deg(p)$ to represent the degree of the polynomial p.

2 Problem formulation

Consider the power system Σ in Fig. 2, composed of n generators.

From the swing equation, the dynamics of generator i ($i \in \{1, 2, \ldots, n\}$) is described by

$$G_i : \ M_i \ddot{\delta}_i(t) = P_{mi} - P_{ei}(\delta(t)) - D_i \dot{\delta}_i(t) \tag{1}$$

where $\delta_i(t) \in \mathbb{R}$ is the phase angle of the generator voltage, $\delta(t) \in \mathbb{R}^n$ denotes the phase angles of all the generator voltages, i.e., $\delta(t) := [\delta_1(t) \ \delta_2(t) \ \cdots \ \delta_n(t)]^\top$, $M_i \in \mathbb{R}_+$ is the moment of inertia, $P_{mi} \in \mathbb{R}$ is the mechanical input, and $D_i \in \mathbb{R}_+$ is the damping coefficient. The variable $P_{ei}(\delta(t)) \in \mathbb{R}$ is the electrical output given by

$$P_{ei}(\delta(t)) := \sum_{j \in \{1,2,\ldots,n\}} E_i E_j (B_{ij} \sin(\delta_i(t) - \delta_j(t)) + C_{ij} \cos(\delta_i(t) - \delta_j(t))) \tag{2}$$

where $E_i \in \mathbb{R}_+$ is the generator voltage and $B_{ij}, C_{ij} \in \mathbb{R}$ are the susceptance and conductance between generators i and j, respectively.

By considering $\delta_i(t) - \delta_j(t)$ in (2), we introduce the state variable $x(t) := [\delta_1(t) - \delta_n(t) \ \ \delta_2(t) - \delta_n(t) \ \cdots \ \delta_{n-1}(t) - \delta_n(t) \ \ \dot{\delta}_1(t) \ \ \dot{\delta}_2(t) \ \cdots \ \dot{\delta}_n(t)]^\top \in \mathbb{R}^{2n-1}$. Then, from (1) and (2), the state equation of the system Σ is of the form

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \vdots \\ \dot{x}_{n-1}(t) \\ \dot{x}_n(t) \\ \dot{x}_{n+1}(t) \\ \vdots \\ \dot{x}_{2n-1}(t) \end{bmatrix} = \begin{bmatrix} x_n(t) - x_{2n-1}(t) \\ x_{n+1}(t) - x_{2n-1}(t) \\ \vdots \\ x_{2n-2}(t) - x_{2n-1}(t) \\ (1/M_1)(P_{m1} - P_{e1,x}(x(t)) - D_1 x_n(t)) \\ (1/M_2)(P_{m2} - P_{e2,x}(x(t)) - D_2 x_{n+1}(t)) \\ \vdots \\ (1/M_n)(P_{mn} - P_{en,x}(x(t)) - D_n x_{2n-1}(t)) \end{bmatrix} \tag{3}$$

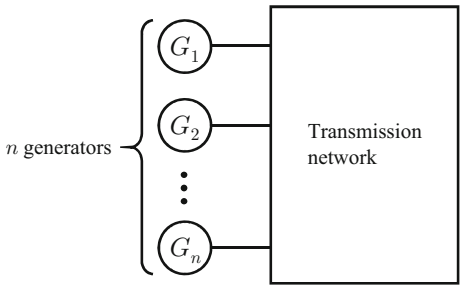

Fig. 2 Power system Σ

where $x_i(t)$ $(i \in \{1, 2, \ldots, 2n-1\})$ is the i-th element of $x(t)$ and $P_{ei,x}(x(t))$ $(i \in \{1, 2, \ldots, n\})$ is given by

$$P_{ei,x}(x(t)) := \begin{cases} E_i E_n (B_{in} \sin x_i(t) + C_{in} \cos x_i(t)) + \sum_{j \in \{1,2,\ldots,n-1\}} E_i E_j \\ \times (B_{ij} \sin(x_i(t) - x_j(t)) + C_{ij} \cos(x_i(t) - x_j(t))) \\ \qquad \text{if } i \in \{1, 2, \ldots, n-1\}, \\ E_n^2 C_{nn} + \sum_{j \in \{1,2,\ldots,n-1\}} E_n E_j (-B_{nj} \sin x_j(t) + C_{nj} \cos x_j(t)) \\ \qquad \text{if } i = n. \end{cases} \tag{4}$$

Furthermore, for simplicity, we assume that an equilibrium point of Σ is $x = 0$; otherwise, we perform a coordinate transformation so as to shift the equilibrium point to $x = 0$.

Then, we address the following problem.

Problem 1 For the power system Σ, assume that the equilibrium point $x = 0$ is asymptotically stable. Then, estimate the region of attraction (ROA), i.e., the set of all $x \in \mathbb{R}^{2n-1}$ such that the solution of Σ starting from x converges to the equilibrium point.

3 Estimation of region of attraction by using sum of squares programming [1]

As a method to solve Problem 1, [1] has proposed an estimation method of the ROA based on sum of squares (SOS) programming. In this section, we briefly introduce it.

3.1 Preliminary

For the power system Σ, the Lyapunov stability theory (see, e.g., [8]) yields the following result.

Lemma 1 *Consider the power system Σ with the asymptotically stable equilibrium point $x = 0$. Assume that there exist a set $\mathbb{D} \subset \mathbb{R}^{2n-1}$ containing $x = 0$ and a continuously differentiable function $V : \mathbb{D} \to \mathbb{R}_{0+}$ such that*

$$V(0) = 0 \text{ and } V(x) > 0 \quad \forall x \in \mathbb{D} \setminus \{0\}, \tag{5}$$

$$\dot{V}(0) = 0 \text{ and } \dot{V}(x) < 0 \quad \forall x \in \mathbb{D} \setminus \{0\}. \tag{6}$$

Then, the set $\mathbb{L}_c(V(x))$ satisfying $\mathbb{L}_c(V(x)) \subseteq \mathbb{D}$ is included in the ROA, where $c \in \mathbb{R}_+$ is a positive number.

The function $V(x)$ is called the Lyapunov function.

Lemma 1 means that $\mathbb{L}_c(V(x)) \subseteq \mathbb{D}$ is an estimate of the ROA. That is, this result reduces Problem 1 to the problem of finding a pair $(V(x), c)$ satisfying (5), (6), and $\mathbb{L}_c(V(x)) \subseteq \mathbb{D}$ for a set $\mathbb{D}$.

3.2 Estimation algorithm of region of attraction based on sum of squares programming

The authors in [1] have proposed the use of SOS programming to find appropriate $V(x)$ and c.

3.2.1 Sum of squares programming problems

We first define the SOS.

Definition 1 For $z \in \mathbb{R}^m$, the polynomial $p(z) \in \mathbb{P}$ is said to be the SOS if there exist polynomials $q_1(z), q_2(z), \ldots, q_\mu(z)$ satisfying

$$p(z) = \sum_{i=1}^{\mu} q_i^2(z). \tag{7}$$

As shown in Definition 1, SOS polynomials are polynomials which can be expressed as the sum of the squares of polynomials. For example, $p(z) := 4z_1^2 + 2z_2^2 + 4z_1 z_2 + 2z_2 + 1$ for $z := [z_1 \ z_2]^\top$ is an SOS polynomial. In fact, $q_1(z) := 2z_1 + z_2$ and $q_2(z) := z_2 + 1$ satisfy (7).

SOS programming problems are optimization problems with constraints that polynomials must be SOS ones. This type of problems can be transformed into semidefinite programming problems [2] under certain conditions, and can be efficiently solved by numerical optimization techniques.

3.2.2 Coordinate transformation

However, the idea of the SOS cannot be directly applied to the system Σ because (3) is not described by polynomials due to the trigonometric functions in (4). Hence, [1] has performed the coordinate transformation $z(t) := h(x(t))$ where $h : \mathbb{R}^{2n-1} \to \mathbb{R}^{3n-2}$ is the function such that

$$z_i(t) := \begin{cases} \sin x_i(t) & \text{if } i \in \{1, 2, \ldots, n-1\}, \\ x_i(t) & \text{if } i \in \{n, n+1, \ldots, 2n-1\}, \\ 1 - \cos x_{i-2n+1}(t) & \text{if } i \in \{2n, 2n+1, \ldots, 3n-2\} \end{cases} \tag{8}$$

for the i-th element $z_i(t)$ $(i \in \{1, 2, \ldots, 3n-2\})$ of $z(t)$. Here, for the convenience of explanation, the numbering of $z_i(t)$ $(i = 1, 2, \ldots, 3n-2)$ is different form that in [1]. From

(3), (4), and (8), the transformed system is written as

$$\begin{cases} \dot{z}(t) = f(z(t)), \\ g(z(t)) = 0 \end{cases} \tag{9}$$

where $f : \mathbb{R}^{3n-2} \to \mathbb{R}^{3n-2}$ and $g : \mathbb{R}^{3n-2} \to \mathbb{R}^{n-1}$ are the functions whose i-th elements $f_i(z(t))(i \in \{1, 2, \ldots, 3n-2\})$ and $g_i(z(t))$ $(i \in \{1, 2, \ldots, n-1\})$ are defined as

$$f_i(z(t)) := \begin{cases} (1 - z_{i+2n-1}(t))(z_{i+n-1}(t) - z_{2n-1}(t)) \\ \qquad \text{if } i \in \{1, 2, \ldots, n-1\}, \\ (1/M_{i-n+1})(P_{m(i-n+1)} - P_{e(i-n+1),z}(z(t)) - D_{i-n+1} z_i(t)) \\ \qquad \text{if } i \in \{n, n+1, \ldots, 2n-1\}, \\ z_{i-2n+1}(t)(z_{i-n}(t) - z_{2n-1}(t)) \\ \qquad \text{if } i \in \{2n, 2n+1, \ldots, 3n-2\}, \end{cases} \tag{10}$$

$$g_i(z(t)) := z_i^2(t) + z_{i+2n-1}^2(t) - 2z_{i+2n-1}(t) \tag{11}$$

for

$$P_{ej,z}(z(t)) := \begin{cases} E_j E_n (B_{jn} z_j(t) + C_{jn}(1 - z_{j+2n-1}(t))) + \sum_{k \in \{1,2,\ldots,n-1\}} E_j E_k \\ \times (B_{jk}(z_j(t)(1 - z_{k+2n-1}(t)) - z_k(t)(1 - z_{j+2n-1}(t))) \\ + C_{jk}((1 - z_{j+2n-1}(t))(1 - z_{k+2n-1}(t)) + z_j(t) z_k(t))) \\ \qquad \text{if } j \in \{1, 2, \ldots, n-1\}, \\ E_n^2 C_{nn} + \sum_{k \in \{1,2,\ldots,n-1\}} E_n E_k \\ \times (-B_{nk} z_k(t) + C_{nk}(1 - z_{k+2n-1}(t))) \qquad \text{if } j = n. \end{cases} \tag{12}$$

In (9), $g(z(t)) = 0$ is the constraint imposed as the property of the trigonometric functions, i.e., $\sin^2 x_i + \cos^2 x_i = 1$ for every $x_i \in \mathbb{R}$. We see from (9)–(12) that the transformed system is described by polynomials.

For this system, if we find a pair $(V(z), c)$ satisfying (5), (6), and $\mathbb{L}_c(V(z)) \subseteq \mathbb{D}$ where x corresponds to z, by SOS programming, then from Lemma 1 we can use $\mathbb{L}_c(V(h(x)))$ as an estimate of the ROA of the original system (3).

3.2.3 Estimation algorithm

The estimation algorithm in [1] is called the *expanding interior algorithm*. This has been originally introduced in [6], and the idea is explained as follows. Consider the set $\mathbb{L}_\gamma(p_+(z)) \subset \mathbb{R}^{3n-2}$ where $p_+(z) \in \mathbb{P}_+$ is a positive definite polynomial and $\gamma \in \mathbb{R}_+$ is a positive number. If $\mathbb{L}_\gamma(p_+(z)) \subseteq \mathbb{L}_c(V(z))$, then we can expand $\mathbb{L}_c(V(z))$ by expanding $\mathbb{L}_\gamma(p_+(z))$ as illustrated in Fig. 3, which will present a better estimate of the ROA. Hence, for a given

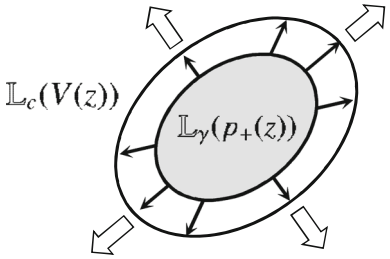

Fig. 3 Idea of expanding interior algorithm

$p_+(z)$, we find the pair $(V(z), c)$ maximizing γ subject to $\mathbb{L}_\gamma(p_+(z)) \subseteq \mathbb{L}_c(V(z))$.

Based on this idea, we consider the following optimization problem:

$$\text{(OP)} \quad \max_{V \in \mathbb{P}_0} \gamma$$

s.t.

$$\{z \in \mathbb{R}^{3n-2} \mid V(z) \le 0, g(z) = 0, z \ne 0\} = \emptyset,$$
$$\{z \in \mathbb{R}^{3n-2} \mid p_+(z) \le \gamma, g(z) = 0, V(z) \ge c, V(z) \ne c\} = \emptyset,$$
$$\{z \in \mathbb{R}^{3n-2} \mid V(z) \le c, g(z) = 0, \dot{V}(z) \ge 0, z \ne 0\} = \emptyset$$

where $p_+(z)$ and c are assumed to be given. The first, second, and third constraints correspond to (5), $\mathbb{L}_\gamma(p_+(z)) \subseteq \mathbb{L}_c(V(z))$, and (6), respectively, where $\mathbb{L}_c(V(z))$ corresponds to $\mathbb{D}$. By replacing $z \ne 0$ with $q_+(z) \ne 0$ where $q_+(z) \in \mathbb{P}_+$ and applying the Positivstellensatz theorem (see, e.g., [7]) to the resulting constraints, we obtain the following SOS programming problem:

$$\text{(SOSP)} \quad \max_{V \in \mathbb{P}_0, v_1, v_2, v_3 \in \mathbb{P}^{n-1}, s_1, s_2, s_3 \in \mathbb{S}} \gamma$$

s.t.

$$V(z) - v_1^\top(z) g(z) - q_+(z) \in \mathbb{S}, \tag{13}$$
$$-s_1(z)(\gamma - p_+(z)) - v_2^\top(z) g(z) - (V(z) - c) \in \mathbb{S}, \tag{14}$$
$$-s_2(z)(c - V(z)) - s_3(z)\dot{V}(z) - v_3^\top(z) g(z) - q_+(z) \in \mathbb{S} \tag{15}$$

where $\mathbb{S}$ is the set of SOS polynomials, $v_1(z), v_2(z), v_3(z) \in \mathbb{P}^{n-1}$ are polynomial vectors, $s_1(z), s_2(z), s_3(z) \in \mathbb{S}$ are SOS polynomials, and $q_+(z)$ is assumed to be given. The constraints (13)–(15) correspond to sufficient conditions for satisfying the three constraints of the OP, respectively.

Since the SOSP contains the products of the variables such as $s_2(z)V(z)$, it cannot be transformed into a semidefinite programming problem and cannot be efficiently solved. Therefore, [1] has proposed the following algorithm. Here, $k \in \mathbb{R}_+$ and $\ell \in \mathbb{R}_+$ are the iteration indices, and superscripts are used to represent the variables at each iteration. For instance, $V^k(z)$ denotes $V(z)$ at iteration k. Note that the

following algorithm has been given as a modified expanding interior algorithm, and the original algorithm can be found in [6]. Also, there are some variations of the expanding interior algorithm in [7].

Algorithm 1 [1]

Step 0 Choose the initial Lyapunov function $V^0(z)$ and polynomial $p_+^0(z)$, the polynomial $q_+(z)$, and the thresholds $\epsilon_\gamma, \epsilon_p \in \mathbb{R}_+$ in Steps 6 and 7. Set $\gamma^0 := 0$, $k := 1$, and $\ell := 1$.

Step 1 Set $p_+(z) := p_+^{\ell-1}(z)$.

Step 2 Set $V(z) := V^{k-1}(z)$ and $\gamma := \gamma^{k-1}$, and solve the SOS programming problem

$$\text{(SOSP1)} \quad \max_{v_2, v_3 \in \mathbb{P}^{n-1}, s_1, s_2, s_3 \in \mathbb{S}} c \quad \text{s.t. (14) and (15).}$$

Save the resulting c as c^k.

Step 3 Set $V(z) := V^{k-1}(z)$ and $c := c^k$, and solve the SOS programming problem

$$\text{(SOSP2)} \quad \max_{v_2, v_3 \in \mathbb{P}^{n-1}, s_1, s_2, s_3 \in \mathbb{S}} \gamma \quad \text{s.t. (14) and (15).}$$

Save the resulting γ, $s_2(z)$, and $s_3(z)$ as γ^k, $s_2^k(z)$, and $s_3^k(z)$, respectively.

Step 4 Set $\gamma := \gamma^k$, $s_2(z) := s_2^k(z)$, and $s_3(z) := s_3^k(z)$, and solve the SOS programming problem

$$\text{(SOSP3)} \quad \min_{V \in \mathbb{P}_0, v_1, v_2, v_3 \in \mathbb{P}^{n-1}, s_1 \in \mathbb{S}} c \quad \text{s.t. (13)–(15).}$$

Save the resulting c as c^k.

Step 5 Set $c := c^k$, $s_2(z) := s_2^k(z)$, and $s_3(z) := s_3^k(z)$, and solve the SOS programming problem

$$\text{(SOSP4)} \quad \max_{V \in \mathbb{P}_0, v_1, v_2, v_3 \in \mathbb{P}^{n-1}, s_1 \in \mathbb{S}} \gamma \quad \text{s.t. (13)–(15).}$$

Save the resulting γ and $V(z)$ as γ^k and $V^k(z)$, respectively.

Step 6 If $\gamma^k - \gamma^{k-1} < \epsilon_\gamma$, then go to Step 7. Otherwise, set $k \leftarrow k+1$, and go to Step 2.

Step 7 If $\ell > 1$ and the largest absolute value of the coefficients of $p_+^{\ell-1}(z) - p_+^{\ell-2}(z)$ is smaller than ϵ_p, then go to Step 8. Otherwise, set $V^0(z) := V^k(z)$, $p_+^\ell(z) := V^k(z)$, $\gamma^0 := c^k$, $k := 1$, and $\ell \leftarrow \ell+1$, and go to Step 1.

Step 8 Output $V^k(h(x))$ and c^k.

The flowchart of Algorithm 1 is illustrated in Fig. 4. The algorithm is composed of the two loops. In Loop 1, we solve four SOS programming problems, i.e., SOSP1–SOSP4 at each iteration k for obtaining a solution to the SOSP. The SOSP1 is for determining c to be given in the SOSP2. This problem can be solved by the linear search for c. In fact, the constraints (14) and (15) include no products of the variables when $V(z)$, γ, and c are fixed, and thus the problem of finding $v_2(z)$, $v_3(z)$, $s_1(z)$, $s_2(z)$, and $s_3(z)$ satisfying the constraints can be solved as a semidefinite feasibility problem. The SOSP2 is for finding a better γ. In a similar way to the above, we can show that this problem also can be solved by the linear search for γ. The SOSP3 and the SOSP4 play similar roles to the SOSP1 and the SOSP2, respectively, where $s_2(z)$ and $s_3(z)$ are fixed instead of $V(z)$. In summary, we fix some variables as an approach to the problem of the products of the variables. Meanwhile, in Loop 2, we update $p_+(z)$ and γ^0 based on $V(z)$ and c obtained in Loop 1. This is because setting $\mathbb{L}_\gamma(p_+(z)) := \mathbb{L}_c(V(z))$ and solving again the SOSP will give a better estimate of the ROA as seen from Fig. 3.

Remark 1 From $V^0(z) := V^k(z)$, $p_+^\ell(z) := V^k(z)$, and $\gamma^0 := c^k$ in Step 7, the constraint (14) of the SOSP1 at $\ell > 1$ and $k = 1$ means $\mathbb{L}_{c_*^{\ell-1}}(V_*^{\ell-1}(z)) \subseteq \mathbb{L}_c(V_*^{\ell-1}(z))$ where $c_*^{\ell-1}$ and $V_*^{\ell-1}(z)$ are the final c and $V(z)$ at iteration $\ell-1$, respectively. Thus, the maximum value of c in the SOSP1 at $\ell > 1$ and $k = 1$ would be $c_*^{\ell-1}$; that is, c would not change. However, the algorithm is not stuck because $V(z)$ is updated so as to increase γ in the SOSP4.

3.3 Problem to be considered

As mentioned in Sect. 1, the above algorithm is complicated. In fact, the algorithm consists of the nine steps (Steps 0–8), and we have to iteratively solve the four SOS programming problems (SOSP1–SOSP4), while changing the fixed variables. Such complexity causes the increase of the time and effort spent to understand and implement the algorithm, which is undesirable in practice. In addition, due to the complexity, it is difficult to theoretically analyze the behavior of the algorithm.

4 Improvement of algorithm

Now, we present a simpler algorithm to solve the SOSP.

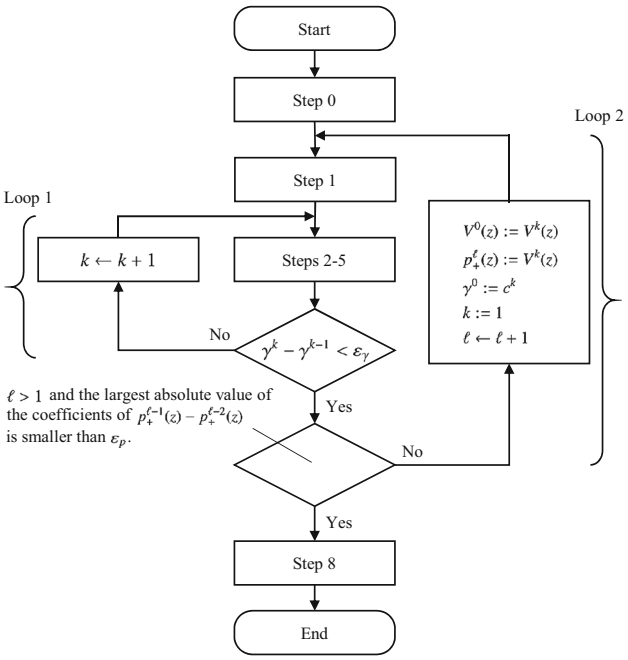

Fig. 4 Flowchart of Algorithm 1

4.1 Proposed algorithm

As explained in Sect. 3.2.3, the SOSP includes the products of the variables, and as a result, it cannot be efficiently solved as a semidefinite programming problem. From the above discussion and the fact that the products are $s_1(z)\gamma$, $s_2(z)V(z)$, and $s_3(z)\dot{V}(z)$, we can solve the SOSP by the linear search for γ if $s_2(z)$ and $s_3(z)$ are fixed. Therefore, we consider the two SOS programming problems:

- a problem to determine $s_2(z)$ and $s_3(z)$ in addition to c,
- a problem to find $V(z)$ maximizing γ,

and alternately solve them.

Based on this idea, we modify Algorithm 1 as follows:

- Step 2 is modified as Step 2':

  Step 2' Set $V(z) := V^{k-1}(z)$ and $\gamma := \gamma^{k-1}$, and solve the SOSP1. Then, save the resulting c, $s_2(z)$, and $s_3(z)$ as c^k, $s_2^k(z)$, and $s_3^k(z)$, respectively.

- Steps 3 and 4 are removed.

In the modified algorithm, we only have to solve the SOSP1 and the SOSP4. The SOSP1 is for determining c, $s_2(z)$, and $s_3(z)$, and the SOSP4 is for finding $V(z)$ maximizing γ. This algorithm does not need the steps to solve the SOSP2 and the SOSP3, and thus is simpler than the original one.

Furthermore, we choose the initial Lyapunov function $V^0(z)$ in Step 0 by solving the following problem:

(SOSP0) find (V^0, v_1, v_3, s_2)

s.t.

$$V^0(z) - v_1^\top(z)g(z) - q_+(z) \in \mathbb{S}, \tag{16}$$

$$-s_2(z)(\beta - r_+(z)) - \dot{V}^0(z) - v_3^\top(z)g(z) - q_+(z) \in \mathbb{S} \tag{17}$$

where $\beta \in \mathbb{R}_+$ is a positive number, $r_+(z) \in \mathbb{P}_+$ is a positive definite polynomial, and $q_+(z)$, β, and $r_+(z)$ are assumed to be given. The first constraint corresponds to (13), that is, the first constraint of the OP. Meanwhile, the second constraint corresponds to the third constraint of the OP. In fact, this is derived by substituting $s_3(z) = 1$ for the constraint (15) of the SOSP and by replacing c and $V(z)$ with β and $r_+(z)$, respectively. By imposing these constraints, we can obtain a Lyapunov function satisfying (5) and (6). Also, the SOSP0 is an SOS feasibility problem, and does not include the products of the variables. Thus, this problem, can be efficiently solved as a semidefinite feasibility problem.

4.2 Analysis

For the proposed algorithm, we obtain the following result.

Theorem 1 *For the proposed algorithm, let the initial Lyapunov function $V^0(z)$ in Step 0 be given as a solution to the SOSP0. Then, for each $\ell \in \{1, 2, \ldots\}$, if there exists a solution to the SOSP1 at $k = 1$, the following statements hold.*

(i) There exist solutions to the SOSP1 and the SOSP4 and

$$\gamma^k \geq \gamma^{k-1} \tag{18}$$

holds at every $k \in \{1, 2, \ldots\}$.

(ii) The relation $\mathbb{L}_{c_*^{\ell-1}}(V_*^{\ell-1}(z)) \subseteq \mathbb{L}_{c^1}(V^1(z))$ holds.

Proof (i) We prove the statement by showing the following three facts for the four cases of $\ell = 1$ and $k = 1$, $\ell = 1$ and $k > 1$, $\ell > 1$ and $k = 1$, and $\ell > 1$ and $k > 1$.

(a) There exists a solution to the SOSP1.
(b) There exists a solution to the SOSP4.
(c) The relation (18) holds.

The proofs of (a)–(c) for those cases are given in Appendix A.

(ii) From (i), for each $\ell \in \{1, 2, \ldots\}$, there exists a solution to the SOSP4 at $k = 1$. The constraint (14) of the SOSP4 at $k = 1$ implies that $\mathbb{L}_{\gamma^1}(p_+^\ell(z)) \subseteq \mathbb{L}_{c^1}(V^1(z))$ holds at each iteration $\ell \in \{1, 2, \ldots\}$. This and $p_+^\ell(z) := V_*^{\ell-1}(z)$ in Step 7 provide $\mathbb{L}_{\gamma^1}(V_*^{\ell-1}(z)) \subseteq \mathbb{L}_{c^1}(V^1(z))$ at each iteration $\ell \in \{1, 2, \ldots\}$. In addition, (18) implies $\mathbb{L}_{\gamma^0}(V_*^{\ell-1}(z)) \subseteq$

$\mathbb{L}_{\gamma^1}(V_*^{\ell-1}(z))$. Therefore, from $\gamma^0 := c_*^{\ell-1}$ in Step 7, we have $\mathbb{L}_{c_*^{\ell-1}}(V_*^{\ell-1}(z)) \subseteq \mathbb{L}_{c^1}(V^1(z))$, which proves (ii).

In Theorem 1, (i) guarantees the existence of solutions to the SOSP1 and the SOSP4 at each iteration $k \in \{2, 3, \ldots\}$, and further clarifies the behavior of γ^k. More precisely, γ^k is monotone nondecreasing with respect to $k \in \{0, 1, \ldots\}$ for each $\ell \in \{1, 2, \ldots\}$. In this sense, the estimate of the ROA does not become worse in Loop 1. Meanwhile, (ii) means that when ℓ is updated, the resulting $\mathbb{L}_c(V(z))$ does not become smaller than the previous one; that is, Loop 2 also does not make the estimate of the ROA worse.

From Theorem 1, the proposed algorithm works as long as there exists a solution to the SOSP1 at $k = 1$, and it converges if the ROA is bounded. The convergence is proven as follows. From (18), γ^k converges as $k \to \infty$ for each $\ell \in \{1, 2, \ldots\}$ if the ROA is bounded. Moreover, for a bounded ROA, $p_+^\ell(z)$ converges as $\ell \to \infty$. In fact, $\mathbb{L}_c(V(z))$ does not become smaller in Loop 2 from (ii) in Theorem 1, and thus $\mathbb{L}_\gamma(p_+(z))$ is asymptotically close to $\mathbb{L}_c(V(z))$ if the ROA is bounded (see Fig. 3). These two facts prove the convergence of the proposed algorithm.

Theorem 1 does not guarantee the optimality of the obtained solution, but is reasonable for the SOSP. In fact, as explained in Sect. 3.2.3, the SOSP cannot be transformed into a semidefinite programming problem, and thus it is difficult to directly obtain the optimal solution.

4.3 Examples

In order to demonstrate the performance of the proposed algorithm, we provide examples for the two power systems discussed in [1].

4.3.1 Model A: System without transfer conductances

Consider the power system

$$\begin{cases} \dot{x}_1(t) = x_3(t), \\ \dot{x}_2(t) = x_4(t), \\ \dot{x}_3(t) = -\sin(x_1(t)) - 0.5\sin(x_1(t) - x_2(t)) - 0.4x_3(t), \\ \dot{x}_4(t) = -0.5\sin(x_2(t)) - 0.5\sin(x_2(t) - x_1(t)) \\ \qquad\quad -0.5x_4(t) + 0.05. \end{cases} \tag{19}$$

This is a power system without transfer conductances, composed of three generators, which has been called Model A in [1]. In (19), for simplifying the description, the relative phase angles and angular velocities to generator 3 are chosen as the state variable vector, i.e., $x(t) := [\delta_1(t) - \delta_3(t)\ \ \delta_2(t) - \delta_3(t)\ \ \dot{\delta}_1 - \dot{\delta}_3(t)\ \ \dot{\delta}_2(t) - \dot{\delta}_3(t)]^\top$. Since (19) has the asymptotically stable equilibrium point $x = [0.020\ \ 0.060\ \ 0\ \ 0]^\top$, we perform a coordinate transformation for shifting the equilibrium point to $x = 0$.

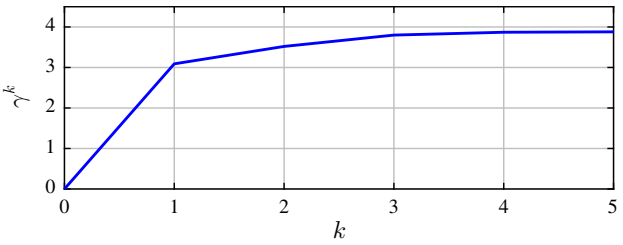

Fig. 5 Time evolution of γ^k at $\ell = 1$ for Model A in [1]

For the shifted system, we use the proposed algorithm, where the SOS problems are handled by the free MATLAB toolboxes: SOSTOOLS [12] version 3.00 and SeDuMi [14] version 1.3. As the inputs to SOSTOOLS, let the degrees of the polynomials be $\deg(V) := 2$, $\deg(v_1) := 2$, $\deg(v_2) := 0$, $\deg(v_3) := 2$, $\deg(s_1) := 0$, $\deg(s_2) := 2$, and $\deg(s_3) := 0$. Solving the SOSP0 for $q_+(z) := 10^{-3}\sum_{i=1}^6 z_i^2$, $\beta := 3$, and $r_+(z) := \sum_{i=1}^6 z_i^2$, we obtain the initial Lyapunov function

$$\begin{aligned} V^0(z) = {} & 3.01z_1^2 - 1.34z_1z_2 + 0.958z_1z_3 + 0.0354z_1z_4 \\ & - 0.00252z_1z_5 + 0.0643z_1z_6 + 2.37z_2^2 + 0.259z_2z_3 \\ & + 1.36z_2z_4 - 0.0827z_2z_5 - 0.0726z_2z_6 + 2.24z_3^2 \\ & + 0.741z_3z_4 - 0.00507z_3z_5 - 0.0216z_3z_6 \\ & + 2.62z_4^2 - 0.0166z_4z_5 - 0.00272z_4z_6 \\ & + 3.06z_5^2 - 0.955z_5z_6 + 2.51z_6^2. \end{aligned}$$

We further set $p_+^0(z) := \sum_{i=1}^6 z_i^2 - 1.5z_1z_2 - 0.5z_5z_6$, $\epsilon_\gamma := 0.05$, and $\epsilon_p := 0.2$.

Figure 5 illustrates the time evolution of γ^k at $\ell = 1$. This shows that γ^k increases as k increases, which demonstrates (18) in Theorem 1. Figure 6 also illustrates the time evolution of $\mathbb{L}_{c^k}(V^k(h(x)))$ and $\mathbb{L}_{\gamma^k}(p_+(h(x)))$, where $\ell = 1$, $(x_3, x_4) = (0, 0)$, and the thick and thin lines express the boundaries of $\mathbb{L}_{c^k}(V^k(h(x)))$ and $\mathbb{L}_{\gamma^k}(p_+(h(x)))$, respectively. We observe that $\mathbb{L}_{\gamma^k}(p_+(h(x))) \subseteq \mathbb{L}_{c^k}(V^k(h(x)))$ holds at each k and that $\mathbb{L}_{c^k}(V^k(h(x)))$ and $\mathbb{L}_{\gamma^k}(p_+(h(x)))$ become larger as k increases.

The proposed algorithm then outputs

$$\begin{aligned} V(h(x)) = {} & 2.32\sin^2 x_1 - 1.96\sin x_1 \sin x_2 + 2.01\sin^2 x_2 \\ & + 2.58\cos^2 x_1 - 2.09\cos x_1 \cos x_2 + 2.58\cos^2 x_2 \\ & - 0.0174\sin x_1 \cos x_1 - 0.0101\sin x_1 \cos x_2 \\ & + 0.0400\sin x_2 \cos x_1 + 0.0405\sin x_2 \cos x_2 \\ & + 0.445x_3 \sin x_1 - 0.0162x_3 \cos x_1 \\ & + 0.158x_3 \sin x_2 + 0.0221x_3 \cos x_2 \\ & + 0.0536x_4 \sin x_1 + 0.0256x_4 \cos x_1 \\ & + 1.15x_4 \sin x_2 - 0.0939x_4 \cos x_2 \end{aligned}$$

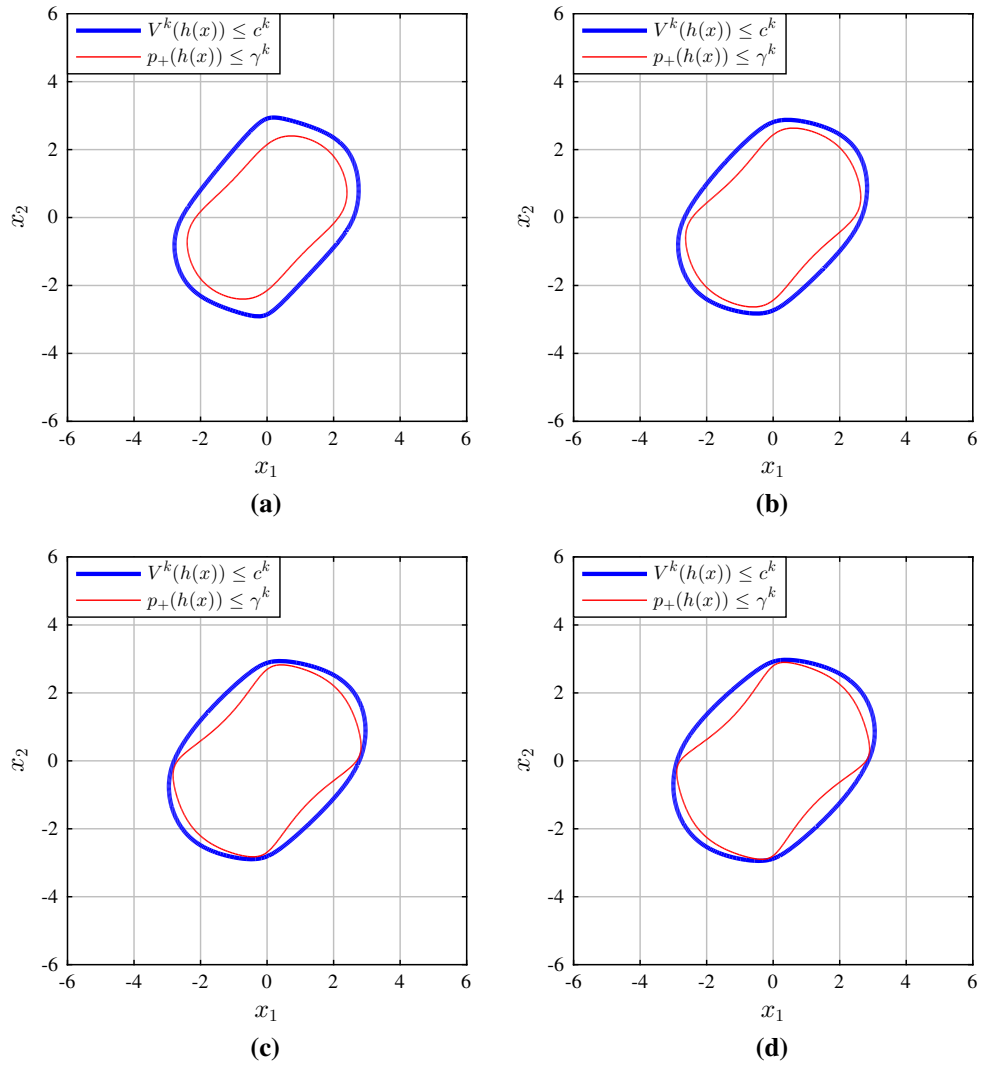

Fig. 6 Time evolution of $\mathbb{L}_{c^k}(V^k(h(x)))$ and $\mathbb{L}_{\gamma^k}(p_+(h(x)))$ at $\ell = 1$ for Model A. **a** $k = 1$, **b** $k = 2$, **c** $k = 3$ **d** $k = 5$

$$
\begin{aligned}
&+ 1.55x_3^2 - 0.550x_3x_4 + 1.53x_4^2 \\
&+ 0.0275 \sin x_1 - 3.06 \cos x_1 - 0.0805 \sin x_2 \\
&- 3.07 \cos x_2 - 0.00596x_3 + 0.0683x_4 + 3.07
\end{aligned}
$$

and $c = 10.18$. The resulting estimate of the ROA (that is, $\mathbb{L}_c(V(h(x)))$) is depicted in Fig. 7 where (a) is for $(x_3, x_4) = (0, 0)$, (b) is for $(x_3, x_4) = (1, 1)$, the thin line expresses the estimate given in [1], i.e., the estimate obtained by Algorithm 1, and the gray area represents the true ROA. We see that the resulting estimate is included in the true ROA. In addition, by numerical integration, the volume of the estimated sets is calculated as 2.28×10^2 for the proposed algorithm and 2.02×10^2 for Algorithm 1. Thus, we conclude that the estimate of the ROA by the proposed algorithm is better than that by the existing algorithm.

4.3.2 Model B: System with transfer conductances

We next consider the power system

$$
\begin{cases}
\dot{x}_1(t) = x_3(t), \\
\dot{x}_2(t) = x_4(t), \\
\dot{x}_3(t) = 33.5849 - 16.9811 \sin(x_1(t) - x_2(t)) \\
\qquad -1.8868 \cos(x_1(t) - x_2(t)) - 59.6226 \sin(x_1(t)) \\
\qquad -5.2830 \cos(x_1(t)) - 1.8868x_3(t), \\
\dot{x}_4(t) = 48.4810 + 11.3924 \sin(x_1(t) - x_2(t)) \\
\qquad -1.2658 \cos(x_1(t) - x_2(t)) - 99.3671 \sin(x_2(t)) \\
\qquad -3.2278 \cos(x_2(t)) - 1.2658x_4(t).
\end{cases}
\tag{20}
$$

This is a power system with transfer conductances, composed of two generators and an infinite bus, which has been called

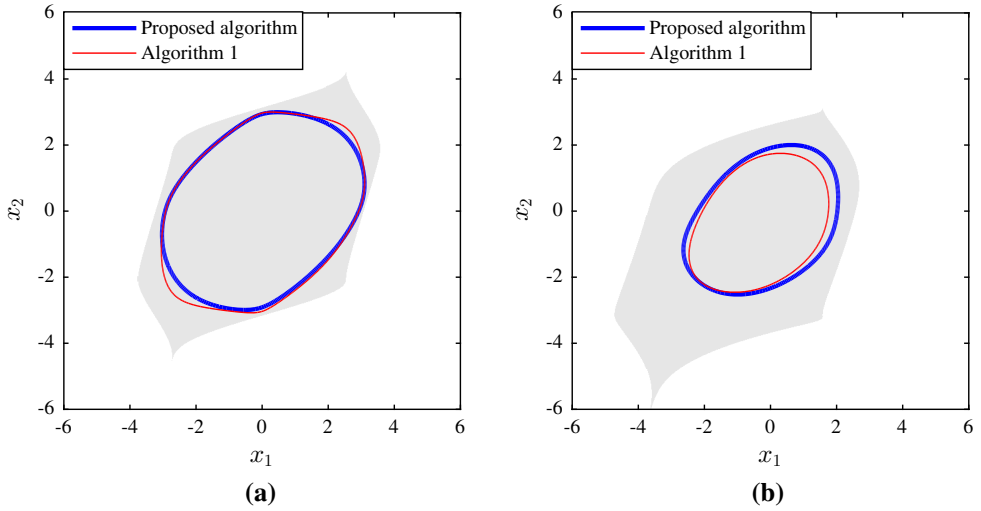

Fig. 7 Estimates of ROA of Model A for two (x_3, x_4). $\mathbf{a}(x_3, x_4) = (0, 0)$, $\mathbf{b}(x_3, x_4) = (1, 1)$

Model B in [1]. Note here that the infinite bus is regarded as generator 3 and $x(t) := [\delta_1(t)\ \ \delta_2(t)\ \ \dot{\delta}_1(t)\ \ \dot{\delta}_2(t)]^\top$ due to $\delta_3(t) \equiv 0$ and $\dot{\delta}_3(t) \equiv 0$. Since (20) has the asymptotically stable equilibrium point $x = [0.4680\ 0.4630\ 0\ 0]^\top$, we perform a coordinate transformation for shifting the equilibrium point to $x = 0$.

For the shifted system, we again use the proposed algorithm, where the degrees of the polynomials and the parameters of the algorithm are the same as those in Sect. 4.3.1 unless otherwise stated. Solving the SOSP0, we obtain the initial Lyapunov function

$$\begin{aligned} V^0(z) = {} & 4.17z_1^2 - 0.568z_1z_2 + 0.0991z_1z_3 + 0.0371z_1z_4 \\ & - 0.676z_1z_5 - 0.904z_1z_6 + 5.10z_2^2 - 0.0394z_2z_3 \\ & + 0.0598z_2z_4 - 0.385z_2z_5 - 0.774z_2z_6 + 0.0643z_3^2 \\ & + 0.0133z_3z_4 - 0.0133z_3z_5 - 0.0140z_3z_6 \\ & + 0.0529z_4^2 - 0.0299z_4z_5 - 0.0178z_4z_6 \\ & + 2.56z_5^2 - 0.116z_5z_6 + 3.16z_6^2. \end{aligned}$$

We also set $p_+^0(z) := \sum_{i=1}^6 z_i^2$.

Figure 8 illustrates the time evolution of γ^k at $\ell = 1$. Figure 9 also illustrates the time evolution of $\mathbb{L}_{c^k}(V^k(h(x)))$ and $\mathbb{L}_{\gamma^k}(p_+(h(x)))$ at $\ell = 1$ in the same way as that in Fig. 6. We see that similar results to those in Sect. 4.3.1 are obtained.

The proposed algorithm then outputs

$$\begin{aligned} V(h(x)) = {} & 1.09\sin^2 x_1 - 0.204\sin x_1 \sin x_2 + 1.21\sin^2 x_2 \\ & + 1.04\cos^2 x_1 - 0.287\cos x_1 \cos x_2 \\ & + 1.03\cos^2 x_2 \\ & + 0.336\sin x_1 \cos x_1 + 0.228\sin x_1 \cos x_2 \\ & + 0.139\sin x_2 \cos x_1 + 0.376\sin x_2 \cos x_2 \\ & + 0.0517x_3 \sin x_1 - 0.0259x_3 \cos x_1 \\ & - 0.0254x_3 \sin x_2 + 0.0133x_3 \cos x_2 \end{aligned}$$

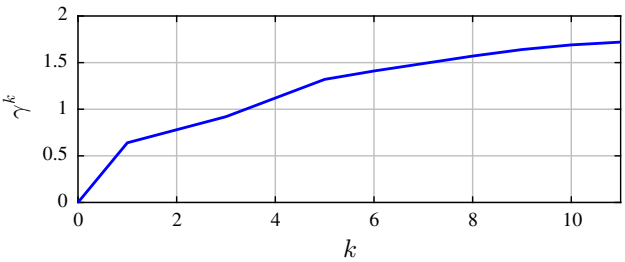

Fig. 8 Time evolution of γ^k at $\ell = 1$ for Model B in [1]

$$\begin{aligned} & + 0.0242x_4 \sin x_1 - 0.00378x_4 \cos x_1 \\ & + 0.0262x_4 \sin x_2 - 0.00508x_4 \cos x_2 \\ & + 0.0176x_3^2 + 0.00289x_3x_4 + 0.0120x_4^2 \\ & - 0.564\sin x_1 - 1.79\cos x_1 - 0.515\sin x_2 \\ & - 1.77\cos x_2 + 0.0126x_3 + 0.00885x_4 + 1.78 \end{aligned}$$

and $c = 2.30$. Figure 10 depicts the resulting estimate of the ROA (that is, $\mathbb{L}_c(V(h(x)))$) in the same way as that in Fig. 7, where (a) is for $(x_3, x_4) = (0, 0)$ and (b) is for $(x_3, x_4) = (7.5, 7.5)$. It turns out that the resulting estimate is included in the true ROA. Moreover, by numerical integration, the volume of the estimated sets is calculated as 1.97×10^3 for the proposed algorithm and 1.67×10^3 for Algorithm 1. This shows that the estimate of the ROA by our algorithm is better than that by the existing algorithm also in this case.

Finally, we show examples of the time response of the (shifted) system (20) in Fig. 11 where (a) is for $x(0) := [0.5\ \ -0.5\ \ 0\ \ 0]^\top$ in the estimated ROA $\mathbb{L}_c(V(h(x)))$, (b) is for $x(0) := [1\ \ -2\ \ 0\ \ 0]^\top$ outside it, and each line corresponds to each element of $x(t)$. It turns out that $x(t)$ in (a) converges to the equilibrium point $x = 0$ but that in (b) does not.

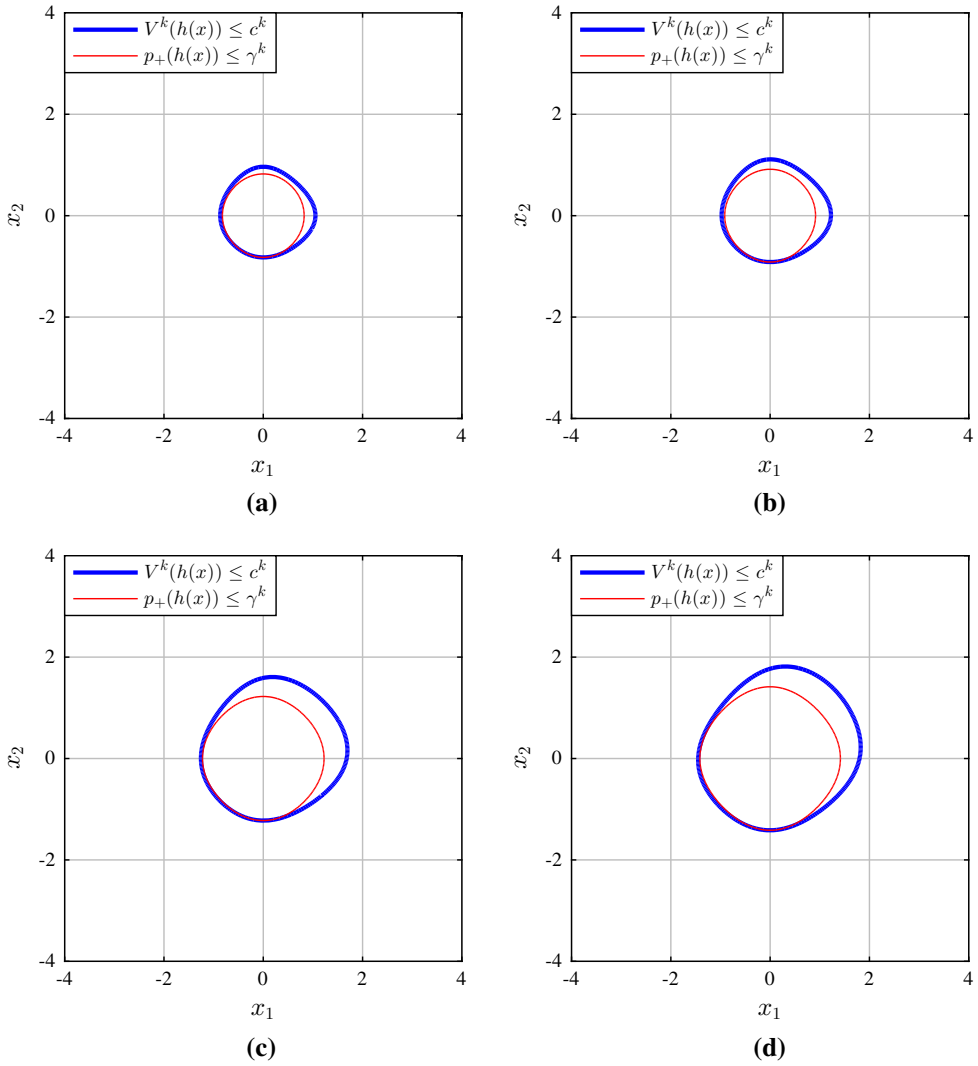

Fig. 9 Time evolution of $\mathbb{L}_{c^k}(V^k(h(x)))$ and $\mathbb{L}_{\gamma^k}(p_+(h(x)))$ at $\ell = 1$ for Model B. **a** $k = 1$, **b** $k = 2$, **c** $k = 5$, **d** $k = 10$

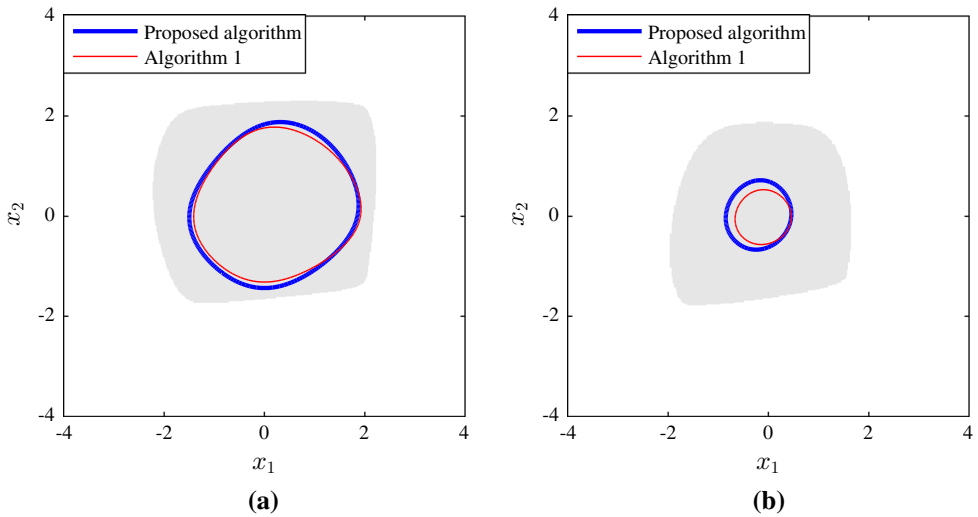

Fig. 10 Estimates of ROA of Model B for two (x_3, x_4). **a** $(x_3, x_4) = (0, 0)$, **b** $(x_3, x_4) = (7.5, 7.5)$

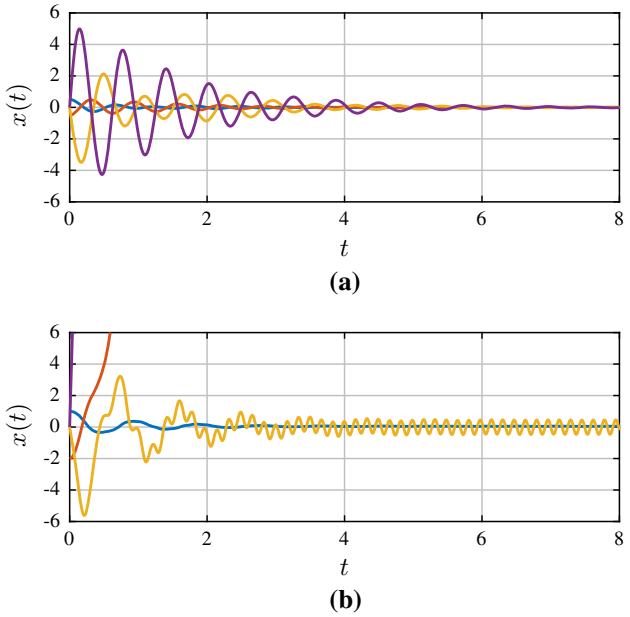

Fig. 11 Time responses of (shifted) Model B for two initial states. **a** $x(0) := [0.5\ \ -0.5\ \ 0\ \ 0]^\top$, **b** $x(0) := [1\ \ -2\ \ 0\ \ 0]^\top$

5 Conclusion

This paper has considered an algorithm estimating the ROA of a given power system by using SOS programming. By simplifying the existing method to handle the bilinear constraints of an SOS programming problem, we have presented a simpler algorithm than the existing one. In addition, we have analyzed the algorithm, and shown the convergence under some conditions. These results provide an estimation algorithm of the ROA which is simpler and with a theoretical guarantee.

A future work is to extend our algorithm to power systems with uncertain parameters. Also, we should consider the extension to controller design for improving the transient stability of power systems.

Acknowledgements The authors would like to thank Mr. Kota Matsuoka for his support.

A Proofs of facts (a)–(c) in proof of (i) in Theorem 1

A.1 Case of $\ell = 1$ and $k = 1$

(a) This follows from the condition in Theorem 1.

(b) The constraints (14) and (15) of the SOSP1 yield

$$-s_1^1(z)(\gamma^0 - p_+(z)) - {v_2^1}^\top(z)g(z) - (V^0(z) - c^1) \in \mathbb{S}, \tag{21}$$

$$-s_2^1(z)(c^1 - V^0(z)) - s_3^1(z)\dot{V}^0(z) - {v_3^1}^\top(z)g(z) - q_+(z) \in \mathbb{S} \tag{22}$$

where $s_1^1(z) \in \mathbb{S}$ and $v_2^1(z), v_3^1(z) \in \mathbb{P}^{n-1}$ are $s_1(z)$, $v_2(z)$, and $v_3(z)$ given as a solution to the considered SOS programming problem (e.g., the SOSP1 in this case) at $k = 1$. By letting $v_1^0(z) \in \mathbb{P}^{n-1}$ be $v_1(z)$ given as a solution to the SOSP0, it follows from (16), (21), and (22) that the tuple $(V^0(z), v_1^0(z), v_2^1(z), v_3^1(z), s_1^1(z))$ satisfies the constraints (13)–(15) of the SOSP4 for γ^0. That is, there exists at least one feasible solution to the SOSP4, which completes the proof.

(c) As shown in the proof of (b), there exists a tuple $(V(z), v_1(z), v_2(z), v_3(z), s_1(z))$ satisfying the constraints (13)–(15) of the SOSP4 for γ^0. This implies that γ^1 is not worse than γ^0. Hence, (18) holds.

A.2 Case of $\ell = 1$ and $k > 1$

(a) By noting the constraints (14) and (15) of the SOSP4 at iteration $k-1$, we obtain (21) and (22) where the superscripts 0 and 1 are replaced with $k-1$. This means that the tuple $(v_2^{k-1}(z), v_3^{k-1}(z), s_1^{k-1}(z), s_2^{k-1}(z), s_3^{k-1}(z))$ satisfies the constraints (14) and (15) of the SOSP1 at iteration k for c^{k-1}. That is, there exists at least one feasible solution to the SOSP1 at each iteration $k \in \{2, 3, \ldots\}$, which completes the proof.

(b) The constraint (13) of the SOSP4 at iteration $k-1$ yields

$$V^{k-1}(z) - {v_1^{k-1}}^\top(z)g(z) - q_+(z) \in \mathbb{S} \tag{23}$$

where $v_1^{k-1}(z) \in \mathbb{P}^{n-1}$ is defined similar to $v_2^{k-1}(z)$ and $v_3^{k-1}(z)$. Next, by noting the constraints (14) and (15) of the SOSP1 at iteration k, we obtain (21) and (22) where the superscripts 0 and 1 are replaced with $k-1$ and k, respectively. This, together with (23), shows that the tuple $(V^{k-1}(z), v_1^{k-1}(z), v_2^k(z), v_3^k(z), s_1^k(z))$ satisfies the constraints (13)–(15) of the SOSP4 at iteration k for γ^{k-1}. That is, there exists at least one feasible solution to the SOSP4 at each iteration $k \in \{2, 3, \ldots\}$, which proves (b).

(c) The proof of (b) implies that a similar discussion to the proof of (c) in Appendix A.1 holds for γ^k and γ^{k-1} at every $k \in \{2, 3, \ldots\}$. Hence, (18) holds.

A.3 Case of $\ell > 1$ and $k = 1$

(a) This follows from the condition in Theorem 1.

(b) From the constraint (13) of the SOSP4 and $V^0(z) := V^k(z)$ in Step 7, there exists a $v_1(z) \in \mathbb{P}^{n-1}$ satisfying (16) at each iteration $\ell \in \{2, 3, \ldots\}$. This, together with the proof of (b) in Appendix A.1, shows (b).
(c) Similar to the proof of (c) in Appendix A.1, we can prove (c).

A.4 Case of $\ell > 1$ and $k > 1$

The difference from the case of $\ell = 1$ and $k > 1$ is only $V^0(z)$, and this does not relate to the discussion in Appendix A.2. Hence, similar to Appendix A.2, (a)–(c) are proven.

References

1. Anghel M, Milano F, Papachristodoulou A (2013) Algorithmic construction of Lyapunov functions for power system stability analysis. IEEE Trans Circuits Syst I Regul Pap 60(9):2533–2546
2. Boyd S, Vandenberghe L (2004) Convex Optimization. Cambridge University Press, Cambridge
3. Bretas NG, Alberto LFC (2003) Lyapunov function for power systems with transfer conductances: extension of the invariance principle. IEEE Trans Power Syst 18(2):769–777
4. Chiang HD, Chu CC (1995) Theoretical foundation of the BCU method for direct stability analysis of network-reduction power system. Models with small transfer conductances. IEEE Trans Circuits Syst I Fundam Theory Appl 42(5):252–265
5. Chiang HD, Wu FF, Varaiya PP (1988) Foundations of the potential energy boundary surface method for power system transient stability analysis. IEEE Trans Circuits Syst 35(6):712–728
6. Jarvis-Wloszek Z (2003) Lyapunov based analysis and controller synthesis for polynomial systems using sum-of-squares optimization. PhD thesis, University of California, Berkeley
7. Jarvis-Wloszek Z, Feeley R, Tan W, Sun K, Packard A (2005) Control applications of sum of squares programming. In: Henrion D, Garulli A (eds) Positive Polynomials in Control. Springer, Berlin, pp 3–22
8. Khalil HK (2002) Nonlinear Systems. Prentice Hall, Upper Saddle River
9. Kojima C, Susuki Y, Tsumura K, Hara S (2015) Decomposition of energy function and hierarchical transient stability diagnosis for power networks. In: Proceedings of the 54th IEEE Conference on Decision and Control, pp 3266–3271
10. Machias A (1987) Improved Lyapunov function for synchronous machine transient stability study. Electr Eng 70(3):163–170
11. Nagel I, Fabre L, Pastre M, Krummenacher F, Cherkaoui R, Kayal M (2013) High-speed power system transient stability simulation using highly dedicated hardware. IEEE Trans Power Syst 28(4):4218–4227
12. Papachristodoulou A, Anderson J, Valmorbida G, Prajna S, Seiler P, Parrilo PA (2013) SOSTOOLS: Sum of squares optimization toolbox for MATLAB. http://www.cds.caltech.edu/sostools/
13. Pavella M, Murthy PG (1994) Transient Stability of Power Systems. Wiley, Hoboken
14. Sturm JF (1998) SeDuMi. http://sedumi.ie.lehigh.edu/
15. Varaiya PP, Wu FF, Chen RL (1985) Direct methods for transient stability analysis of power systems: recent results. Proc IEEE 73(12):1703–1715