MASTERARBEIT | MASTER'S THESIS

Titel | Title

Termination results for hybrid approach of Joint Spectral Radius computation

verfasst von | submitted by

Aaron Pumm, B.Sc.

angestrebter akademischer Grad | in partial fulfilment of the requirements for the degree of

Master Zusatz M.Sc.

Wien, | Vienna, 2025

Studienkennzahl lt. Studienblatt | degree programme code as it appears on the student record sheet:	UA 066 821
Studienrichtung lt. Studienblatt | degree programme as it appears on the student record sheet:	Masterstudium Mathematik
Betreut von | Supervisor:	Assoz. Prof. Dr. Martin Ehler
Ko-betreut von | Co-Supervisor:	Dr. Thomas Mejstrik MMag.

Contents

Zusammenfassung

Diese Arbeit präsentiert einen hybriden Ansatz zur Berechnung des gemeinsamen Spektralradius (JSR) einer endlichen Menge von Matrizen. Die vorgeschlagene Methode integriert die Invariante-Polytop und Endlicher-Baum Algorithmen, um deren jeweilige Stärken zu nutzen. Die allgemeine Anwendbarkeit des baumbasierten Ansatzes wird mit der Effizienz der Polytop-basierten Techniken kombiniert. Der entwickelte Algorithmus konstruiert ein Baumdarstellung und überprüft maximale Produkte mithilfe angepasster Normen, wodurch eine Terminierung in Fällen ermöglicht wird, in denen klassische Methoden an ihre Grenzen stoßen (Bedingungen an Terminierung oder Laufzeit). Theoretische Garantien werden bereitgestellt, um Korrektheit und Effizienz zu belegen. Es werden auch JSR-Berechnungen für Matrizenmengen, bei denen bestehende Methoden scheitern oder ineffizient sind, demonstriert.

Abstract

This thesis presents a hybrid approach for computing the joint spectral radius (JSR) of a finite set of matrices. The proposed method integrates the invariant polytope algorithm and the finite tree algorithm to leverage their respective strengths — integrating the vast applicability of the finite-tree-based approach with the efficiency of polytope-based techniques. The developed algorithm constructs a structured tree representation and verifies spectrum-maximizing products using adapted norms, enabling termination in cases where classical methods take too long or fail. Theoretical guarantees are provided to demonstrate correctness and efficiency.

Keywords: Joint spectral radius, hybrid algorithm, invariant polytope, finite tree method, matrix norms.

1 Introduction

The *Joint Spectral Radius* (JSR) was first introduced by G.-C. Rota and G. Strang in 1960 (Rota and Gilbert Strang, 1960). They described the JSR as the maximal exponential growth rate of a product of matrices taken from a finite set. Since its inception, the JSR has become a cornerstone in various mathematical and engineering disciplines due to its ability to encapsulate the asymptotic behavior of matrix long products.

The concept gained significant traction in the 1990s when researchers began exploring its theoretical properties and practical implications. Notable advancements include its application in wavelet theory, where it assists in the construction of refinable functions (Daubechies and Lagarias, 1992) as well as in control theory, where it is used to analyze the stability of switched linear systems (Blondel and Tsitsiklis, 2000b), for which we will give a small example in the following. The computational challenges associated with determining the JSR have inspired the development of several algorithms, such as the invariant-polytope method (Guglielmi and Protasov, 2013) and the finite-tree method (Möller and Reif, 2014).

Despite the progress, the JSR computation remains a challenging problem, particularly due to the exponential complexity of exploring all possible matrix products. This thesis seeks to contribute to this ongoing effort by leveraging the invariant-polytope algorithm and the finite-tree algorithm to create a hybrid methodology that mitigates their respective limitations.

Structure of the Thesis

The remainder of this thesis is structured as follows: Chapter 1 provides a sufficient background on the JSR and its basic properties. Chapters 2 and 3 present the ideas and concepts of the algorithms that will be exploited to create the proposed hybrid approach, outlining their theoretical foundation and algorithmic implementation. Chapter 4 discusses possible combinations of former approaches, proposes the so-called Tree-flavored-invariant-polytope algorithm, and brings proofs of termination which is the main result of this thesis. Chapter 5 presents numerical results to analyze the efficiency and applicability of the hybrid algorithm. Chapter 6 concludes with insights and future directions.

1.1 Preliminaries

In this thesis, the spectral radius of a matrix A is denoted by $\rho(A)$. Unless stated otherwise, all matrices considered are assumed to have real entries, and all matrix norms are presumed to be submultiplicative. The analogous results for complex matrices follow similarly and may be reduced to the real case when necessary.

Let $\mathcal{A} = \{A_1, \ldots, A_n\}$ denote a finite collection of matrices, each of size $d \times d$.

For any positive integer $k \in \mathbb{N}$, we define the set of positive multi-indices of length k as

$$\mathcal{I}_k := \{1, \ldots, n\}^k.$$

Given a positive multi-index $I = [i_1, \ldots, i_k] \in \mathcal{I}_k$, we define the associated matrix product by

$$A_I = A_{[i_1,\ldots,i_k]} := A_{i_k} \cdots A_{i_1}.$$

Note that the indices are in reversed order compared to the multiplications.

The set of all possible products of length k formed from elements of $\mathcal{A}$ is then given by

$$\mathcal{A}^k := \{A_I \mid I \in \mathcal{I}_k\}.$$

For any scalar $\alpha \in \mathbb{R}$, we define the scaled matrix set by

$$\alpha\mathcal{A} := \{\alpha A_1, \ldots, \alpha A_n\}.$$

1.2 Motivation of the JSR

Let $\mathcal{A} = \{A_1, \cdots, A_n\}$ be a finite set of $d \times d$ matrices. Consider a discrete-time linear switched system of the form:

$$x_{k+1} = A_{\sigma(k)} x_k, \quad x_0 \in \mathbb{R}^d, \tag{1.1}$$

where $\sigma : \mathbb{N} \to \{1, \ldots, n\}$ is a switching signal.

A key question is whether the system is *uniformly asymptotically stable*, i.e., whether every trajectory (x_k) converges to zero for all initial states and switching sequences:

$$\lim_{k\to\infty} x_k = 0.$$

This behavior is governed by a generalization of the spectral radius for a single matrix the so-called *joint spectral radius (JSR)* of a matrix set $\mathcal{A}$. reference

Definition 1.2.1. reference For a finite set of matrices $\mathcal{A} = \{A_1, \ldots, A_n\}$, the JSR is defined as:

$$\mathrm{JSR}(\mathcal{A}) := \lim_{k\to\infty} \max_{P\in\mathcal{A}^k} \|P\|^{1/k}. \tag{1.2}$$

where $\|\cdot\|$ denotes any matrix norm.

This formulation captures the maximal asymptotic growth rate of matrix products.

Example

Consider the matrix set:

$$\mathcal{A} = \left\{ A_1 = \begin{bmatrix} 0.5 & 1 \\ 0 & 0.5 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 0.3 & 0 \\ 1 & 0.3 \end{bmatrix} \right\}.$$

Each matrix individually has spectral radius less than 1. However, their product

$$A_1 A_2 = \begin{bmatrix} 0.5 & 1 \\ 0 & 0.5 \end{bmatrix} \begin{bmatrix} 0.3 & 0 \\ 1 & 0.3 \end{bmatrix} = \begin{bmatrix} 1.3 & 0.3 \\ 0.15 & 0.15 \end{bmatrix}$$

has spectral radius of approximately

$$\rho(A_1 A_2) \approx 1.2929 > 1.$$

This example shows that switching can amplify the effects of individual matrices, leading to divergent behavior even if all single matrices are contractive. Thus, the JSR can exceed the spectral radius of any individual matrix in the set.

The following result motivates the importance of bounding or approximating $\mathrm{JSR}(\mathcal{A})$. It is known that:

- The system (1.1) is *uniformly bounded* (i.e., $\sup_k \|x_k\| < \infty$ for all $x_0 \in \mathbb{R}^d$) if and only if $\mathrm{JSR}(\mathcal{A}) \leq 1$.
- It is *uniformly asymptotically stable* if and only if $\mathrm{JSR}(\mathcal{A}) < 1$.

(Blondel and Tsitsiklis, 2000b),

Understanding the JSR is therefore critical for analyzing the asymptotic behavior of switched systems and motivates the development of reliable algorithms for finding or approximating the JSR, such as the hybrid algorithm proposed in (Mejstrik and Reif, 2024). also other important results depending on JSR

1.3 Theoretical background

Theorem 1.3.1. *The JSR is well-defined and independent of the choice of the matrix norm.*

Proof. Let $\|\cdot\|_1$ and $\|\cdot\|_2$ be two matrix norms on $\mathbb{R}^{n\times n}$. By equivalence of norms in finite-dimensional vector spaces, there exist constants $c, C > 0$ such that:

$$c\|P\|_1 \leq \|P\|_2 \leq C\|P\|_1 \quad \forall P \in \mathbb{R}^{d\times d}.$$

Therfore we have that

$$\begin{aligned}
&\lim_{k\to\infty} \max_{P\in\mathcal{A}^k} \|P\|_1^{1/k} \leq \lim_{k\to\infty} \left(\frac{1}{c}\right)^{\frac{1}{k}} \max_{P\in\mathcal{A}^k} \|P\|_2^{1/k} \\
&= \lim_{k\to\infty} \max_{P\in\mathcal{A}^k} \|P\|_2^{1/k} \leq \lim_{k\to\infty} C^{\frac{1}{k}} \max_{P\in\mathcal{A}^k} \|P\|_1^{1/k} \\
&= \lim_{k\to\infty} \max_{P\in\mathcal{A}^k} \|P\|_1^{1/k},
\end{aligned}$$

which makes the limit independent of the choice of the used matrix norm. The existence of this limit is a special result from Fekete's Lemma which can bee seen in the appendix. appendix ref □

To facilitate the analysis of the algorithms and presentation of the main results, we first establish fundamental properties of the JSR.

Homogeneity

Proposition 1.3.2. *The JSR is homogeneous, meaning for any set of matrices $\mathcal{A}$ and scalar $\alpha \in \mathbb{R}$ we have*

$$\mathrm{JSR}(\alpha\mathcal{A}) = |\alpha|\,\mathrm{JSR}(\mathcal{A}). \tag{1.3}$$

Proof. Let $\mathcal{A} = \{A_1, \ldots, A_n\}$ and $\alpha \in \mathbb{R}$. Then:

$$\begin{aligned}
\mathrm{JSR}(\alpha\mathcal{A}) &= \lim_{k\to\infty} \max_{A_i\in\mathcal{A}^k} \|\alpha A_{i_k} \cdots \alpha A_{i_1}\|^{1/k} \\
&= |\alpha| \lim_{k\to\infty} \max_{A_i\in\mathcal{A}^k} \|A_{i_k} \cdots A_{i_1}\|^{1/k} \\
&= |\alpha| JSR(\mathcal{A})
\end{aligned}$$

□

qedhere

Three-member inequality

reference

Proposition 1.3.3. *The* JSR *can be equally defined as*

$$\mathrm{JSR}(\mathcal{A}) = \limsup_{k\to\infty} \max_{P\in\mathcal{A}^k} \rho(P)^{1/k},$$

if $\mathcal{A}$ *is a finite (or compact) set.*

proof here

Proposition 1.3.4. *Let* $\|\cdot\|$ *be a matrix norm. The inequality*

$$\max_{P\in\mathcal{A}^k} \rho(P)^{\frac{1}{k}} \leq \mathrm{JSR}(\mathcal{A}) \leq \max_{P\in\mathcal{A}^k} \|P\|^{\frac{1}{k}} \tag{1.4}$$

holds for every $k \in \mathbb{N}$.

Proof. For a fixed $k \in \mathbb{N}$ let

$$P_{\max} = \arg\max_{P\in\mathcal{A}^k} \rho(P)^{\frac{1}{k}}.$$

Then for the left-hand side we have

$$\begin{aligned}
\max_{P\in\mathcal{A}^k} \rho(P)^{\frac{1}{k}} &= \rho(P_{\max})^{\frac{1}{k}} \\
&= \rho(P_{\max})^{\frac{r}{kr}} \quad \forall r \in \mathbb{N} \\
&= \rho(P_{\max}^r)^{\frac{1}{kr}} \quad \forall r \in \mathbb{N} \\
&\leq \max_{P\in\mathcal{A}^{kr}} \rho(P)^{\frac{1}{kr}} \quad \forall r \in \mathbb{N} \\
&\leq \limsup_{r\to\infty} \max_{P\in\mathcal{A}^{kr}} \rho(P)^{1/kr} \\
&\leq \limsup_{r\to\infty} \max_{P\in\mathcal{A}^{r}} \rho(P)^{1/r} \\
&= \mathrm{JSR}(\mathcal{A}).
\end{aligned}$$

The right-hand side of the equation follows from a special case of Fekete's lemma which can be seen in the appendix. appendix ref No. [DL1992] □

This result forms a starting point for many computational approaches as the bounds are sharp in the sense that both sides converge to the JSR as $k \to \infty$ (left side in lim sup). left side is not generally converging but but its lim sup

Similarity and reducibility

Proposition 1.3.5. *The* JSR *is invariant under similarity transformations i.e. for any invertable matrix T we have*

$$\mathrm{JSR}(\mathcal{A}) = \mathrm{JSR}(T^{-1}\mathcal{A}T). \tag{1.5}$$

Proof. This follows from the definition of the JSR and the properties of the spectral radius.

$$\begin{aligned}
\mathrm{JSR}(T^{-1}\mathcal{A}T) &= \limsup_{k\to\infty} \max_{A_i\in\mathcal{A}^k} \rho(T^{-1}A_{i_k}T\cdot\ldots\cdot T^{-1}A_{i_1}T)^{1/k} \\
&= \limsup_{k\to\infty} \max_{A_i\in\mathcal{A}^k} \rho(T^{-1}A_{i_k}\cdot\ldots\cdot A_{i_1}T)^{1/k} \\
&= \limsup_{k\to\infty} \max_{A_i\in\mathcal{A}^k} \rho(A_{i_k}\cdot\ldots\cdot A_{i_1})^{1/k} \\
&= \mathrm{JSR}(\mathcal{A})
\end{aligned}$$

□

Definition 1.3.6. A set of matrices is called *(commonly) reducible* if there exists a nontrivial subspace of $\mathbb{R}^n$ that is invariant under all matrices in the set. This means there exists a change of basis that block-triangularizes all matrices in $\mathcal{A}$ at the same time. If $\mathcal{A}$ is not reducible it is called *irreducible*.

Proposition 1.3.7. *If $\mathcal{A}$ is reducible, then*

$$\mathrm{JSR}(\mathcal{A}) = \max\{\mathrm{JSR}(\mathcal{B}), \mathrm{JSR}(\mathcal{D})\}, \tag{1.6}$$

where $\mathcal{B} = \{B_1, \ldots, B_n\}$ and $\mathcal{D} = \{D_1, \ldots, D_n\}$ are the blocks of the block-triangularized matrices

$$T^{-1}A_jT = \begin{bmatrix} B_j & C_j \\ 0 & D_j \end{bmatrix}$$

under some invertable matrix T.

proof here

Example 1.3.8.

$$A_1 = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 3 & 2 \\ 2 & 3 \end{bmatrix}$$

These matrices are reducible because there exists a nontrivial change of basis that transforms them into block-triangular form. The eigenvectors common to both matrices are

$$v_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad v_2 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}.$$

Using the change of basis matrix

$$T = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix},$$

we compute the similarity transformations

$$T^{-1}A_1T = \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix}, \quad T^{-1}A_2T = \begin{bmatrix} 5 & 0 \\ 0 & 1 \end{bmatrix}.$$

Since both matrices are upper triangular in this basis, they share a common invariant subspace, proving that they are reducible. Now from proposition 1.3.7 we know that the JSR can be calculated with

$$\mathrm{JSR}(\mathcal{A}) = \max\{\mathrm{JSR}(\{3,5\}), \mathrm{JSR}(\{1,1\})\} = 5$$

This can be applied iteratively until the sets of blocks are all irreducible. The problem was split into similar problems of smaller dimension. For the following considerations we can now assume $\mathcal{A}$ to be irreducible.

Infimum over norms

Proposition 1.3.9. *reference Rota und Strang 1960 The JSR can be equivalently defined as the infimum over all submultiplicative norms:*

$$\mathrm{JSR}(\mathcal{A}) = \inf_{\|\cdot\|} \max_{A\in\mathcal{A}} \|A\|. \tag{1.7}$$

Proof. This is done by defining the family of norms $\|x\|_\epsilon := \max\{\|\frac{A}{\lambda+\epsilon}x\|_2 : A \in \mathcal{A}\}$, where $\epsilon > 0$ and λ is the JSR of the set $\mathcal{A}$. Now, using the induced matrix norm we arrive at:

$$\sup_{A\in\mathcal{A}} \|Ax\|_\epsilon \leq \lambda + \epsilon$$

qedhere and if $\mathcal{A} is irreducible than the infimum is attained [Berger Wang]$

□

might not be needed so may be disregarded in the future

Finiteness property

Definition 1.3.10. A matrix set $\mathcal{A} = \{A_1, \ldots, A_n\}$ is said to possess the *finiteness property* if there exists a finite sequence of matrices $A_{i_1}, \ldots, A_{i_k}$ such that

$$\mathrm{JSR}(\mathcal{A}) = \|A_{i_k} \cdots A_{i_1}\|^{1/k}. \tag{1.8}$$

Any such product is referred to as a *spectrum-maximizing product* (s.m.p.). [Begarias, Wang 1995]

Although the finiteness property does not hold for all matrix sets, it is crucial for the effectiveness and termination of most algorithmic approaches. [Bousch, Mairesse 2002]

Complexity

The computation of the joint spectral radius (JSR) is theoretically challenging due to several complexity results.

Determining whether the JSR of a given set of matrices is below a threshold is **NP-hard** (Tsitsiklis and Blondel, 1997), meaning no polynomial-time algorithm is expected unless P = NP. Moreover, for general sets of matrices, even deciding whether the JSR is strictly less than one is **undecidable** (Blondel and Tsitsiklis, 2000a). Despite these theoretical limitations, many practical cases allow for efficient numerical estimation.

Unlike the spectral radius of a single matrix, the JSR is often **non-algebraic** (**?**), meaning it cannot always be expressed as a root of a polynomial with rational coefficients.(needs refactoring from result of [Kozyakin])

These complexity results highlight fundamental challenges. There are still applied settings where efficient approximation algorithms provide useful results.

Candidates and generators

Approximating the JSR, using the invariant-polytope or the finite-tree algorithms, requires identifying candidate products or *generators* of the matrix set that contribute most significantly to the asymptotic growth rate. These generators are often derived through optimization techniques and their identification is a key step in computational algorithms.

Spectral gap

Definition 1.3.11. include

Remark 1.3.12. It could be hybrid doesnt need spectral gap if JSR was guessed correctly upfront

Example 1.3.13. include

1.4 Preprocessing

This thesis aims to address the challenge of computing the JSR by combining two existing algorithms that have demonstrated practical effectiveness in calculationg the JSR for nontrivial sets of matrices. Both algorithms are based on the following simple concept:

We want to find the JSR of the finite set of matrices $\mathcal{A} = \{A_1, \cdots, A_n\}$

1. **Assumptions**: $\mathcal{A}$ is irreducible.
2. **Candidates**: Efficiently find products $P = A_{i_k} \cdots A_{i_1}$ of matrices from $\mathcal{A}$ that maximize the averaged-spectral radius $\hat{\rho} := \rho(P)^{\frac{1}{k}}$.
3. **Rescaling**: Transform $\mathcal{A} \to \tilde{\mathcal{A}}$ with $\tilde{A}_i := \frac{1}{\hat{\rho}} A_i$.
4. **Proofing**: Now establish the fact that $\mathrm{JSR}(\tilde{\mathcal{A}}) = 1$ using the three-member-inequality. By homogenity this is equivalent to $\mathrm{JSR}(\mathcal{A}) = \hat{\rho}$.

The considered algorithms only differ in step 4, while the invariant-polytope algorithm tries to find a vector norm whos induced matrix norm bounds the matrices from $\mathcal{A}$ already enough. The finite-tree algorithm, on the other hand, bounds the products using some partitioning-space that separates every product into products that are bounded by 1 and some rest-term that does not grow fast enough to overcome the k-th root of the JSR definition (polynomial growth). By integrating these algorithms into a hybrid approach, this work advances the computational tools available for JSR analysis combining efficiency and a vast space of matrix sets for which the algorithm terminates.

2 Invariant-polytope algorithm

In this chapter we focus on the invariant-polytope algorithm. At its core this algorithm tries to find a polytope $\mathcal{P} \subset \mathbb{R}^d$ which is invariant under the action of the already scaled matrices in $\tilde{\mathcal{A}}$. The according Minkowski functional of the polytope is then a so-called extremal norm for the family $\mathcal{A}$. As we will see, this is enough to proof that the chosen candidate is indeed a s.m.p., which is our goal.

2.1 Extremal norms

Definition 2.1.1. (Barabanov, 1988) For a family of matrices $\mathcal{A}$ a norm $\|\cdot\|$ is called *invariant* or *Barabanov* if there exists a number $\lambda \geq 0$ such that

$$\max_{A\in\mathcal{A}}\|Ax\| = \lambda\|x\|, \quad \forall x \in \mathbb{R}^d.$$

It can be shown that such a λ will always equal the JSR.

Theorem 2.1.2. *(Barabanov, 1988) Every irreducible family $\mathcal{A}$ possesses an invariant norm.*

We are actually interested in norms with a weaker condition.

Definition 2.1.3. (Protasov, 1996) A norm $\|\cdot\|$ is called *extremal* or *protasov* for a family of matrices $\mathcal{A}$ if

$$\max_{A\in\mathcal{A}}\|Ax\| \leq \mathrm{JSR}(\mathcal{A})\|x\| \quad \forall x \in \mathbb{R}^d$$

Every invariant norm is of course extremal which guarantees the existance of such a norm.

Theorem 2.1.4. *For every extremal norm $\|\cdot\|$ of a family of matrices $\mathcal{A}$*

$$\max_{P\in\mathcal{A}^k}\|P\| = \mathrm{JSR}(\mathcal{A})^k$$

holds for every $k \in \mathbb{N}$

Proof. By the submultiplicativity and extremality of the norm we have:

$$\begin{aligned}\max_{P\in\mathcal{A}^k}\|P\| &= \max_{d_k,\dots,d_1}\|A_{d_k}\cdots A_{d_1}\|\\ &\leq \max_{d_k,\dots,d_1}\|A_{d_k}\|\cdots\|A_{d_1}\|\\ &\leq \mathrm{JSR}(\mathcal{A})^k\end{aligned}$$

By the three-member-inequality (1.4) we know

$$\begin{aligned}\mathrm{JSR}(\mathcal{A}) &\leq \max_{P\in\mathcal{A}^k}\|P\|^{\frac{1}{k}}\\ \mathrm{JSR}(\mathcal{A})^k &\leq \max_{P\in\mathcal{A}^k}\|P\|\end{aligned}$$

for every $k\in\mathbb{N}$.

□

Remark 2.1.5. In particular, when $k=1$, we have $\max_{A\in\mathcal{A}}\|A\| = \mathrm{JSR}(\mathcal{A})$. The existence of an extremal norm for $\mathcal{A}$ therefore enables the exact computation of the joint spectral radius.

In the following we are trying to build up an extremal norm whose unit ball is a balanced polytope introduced in (Protasov, 1996) and (Guglielmi and Zennaro, 2008).

Definition 2.1.6. A polytope $\mathcal{P}$ is called *balanced* if it spans the whole space and is equal to the symmetriced convex hull of a finite set of vertices.

$$\mathcal{P} = \mathrm{co_s}(V) := \mathrm{co}(V,-V) = \{x\in\mathbb{R}^d : x=\sum_{i=1}^{M}\alpha_i v_i \text{ and } \sum_{i=1}^{M}|\alpha_i| = 1\}$$

If a polytope fulfills $\mathcal{A}\mathcal{P}\subseteq\mathcal{P}$ for some family $\mathcal{A}$ it is called *invariant under* $\mathcal{A}$.

Definition 2.1.7. Let $C\subseteq\mathbb{R}^d$ be a nonempty, convex set such that $0\in\mathrm{int}(C)$. The *Minkowski functional* (or *gauge function*) associated to C is the mapping $p_C:\mathbb{R}^d\to[0,\infty)$ defined by:

$$p_C(x) = \inf\{\lambda>0 \mid x\in\lambda C\}\quad \forall x\in\mathbb{R}^d$$

This is in general a semi-norm on $\mathbb{R}^d$.

Remark 2.1.8. If $\mathcal{P}$ is a balanced polytope, then the associated Minkowski functional defines a norm on $\mathbb{R}^d$. Moreover, the corresponding induced matrix norm on $\mathbb{R}^{d\times d}$ is submultiplicative. In both cases, we denote the norm by $\|\cdot\|_{\mathcal{P}}$.

Theorem 2.1.9. *If a balanced polytope $\mathcal{P}$ is invariant under $\tilde{\mathcal{A}}$ from* (1.4) *the according Minkowski functional is an extremal norm for the family $\mathcal{A}$ and* $\mathrm{JSR}(\mathcal{A}) = \hat{\rho}$.

Proof.

$$\begin{aligned}\tilde{\mathcal{A}}\mathcal{P} \subseteq \mathcal{P} &\implies \mathcal{A}\mathcal{P} \subseteq \hat{\rho}\mathcal{P} \\ &\implies \|A\|_{\mathcal{P}} \leq \hat{\rho} \quad \forall A \in \mathcal{A} \\ &\implies \max_{A\in\mathcal{A}} \|A\|_{\mathcal{P}} \leq \hat{\rho}\end{aligned}$$

In 1.4 already established $\max_{P\in\mathcal{A}^k} \rho(P)^{\frac{1}{k}} = \hat{\rho}$ for a particular k depending on our search space for preprocessing and by the three-member-inequality (1.4) we have $\hat{\rho} \leq \mathrm{JSR}(\mathcal{A}) \leq \hat{\rho} \implies \mathrm{JSR}(\mathcal{A}) = \hat{\rho}$. □

So by that last theorem we just need to find a balanced polytope that is invariant under the action of $\tilde{\mathcal{A}}$ and we have proven that the chosen candidate from the preprocessing is indeed a s.m.p..

2.2 Structure of the invariant-polytope algorithm

After the preprocessing has been carried out we end up with a candidate $\tilde{\Pi} = \tilde{A_{d_k}} \cdots \tilde{A_{d_1}}$ for our finiteness property hypothesis. We want to establish $\mathrm{JSR}(\tilde{\mathcal{A}}) = 1$ by finding an invariant balanced polytope $\mathcal{P} = \mathrm{co_s}(V)$. Now we define $V := \{v_1, \cdots, v_k\}$ where v_1 is the leading eigenvector of Π which is assumed to be real and $v_i := \tilde{A_{d_{i-1}}} \cdot \ldots \cdot \tilde{A_{d_1}} v_1$ an leading eigenvectors of the cyclic-permutations of Π. Now we add new vertices to V iteratively by multiplying with the matrices from $\tilde{\mathcal{A}}$ i.e.

$$V \overset{\cup}{\leftarrow} \mathcal{A}V : \iff V \leftarrow V \cup \mathcal{A}V.$$

All vertices that fall into the symmetriced convex hull of V can technically be disregarded as they dont contribute to a change of the polytope $\mathcal{P}$, which will lessen the computational effort. In practice it is very important to drop as many vertices as possible since V grows exponentially. This is done by solving a standard linear programming problem to find whether $\|Av\|_{\mathrm{co_s}(V)} \geq 1$. If the algorithm does not produce new vertices that lie outside of the current polytope for every $\tilde{A} \in \tilde{\mathcal{A}}$ it is by definition invariant and the finiteness property was proven for the used candidate.

2.3 Stopping criterion

The runtime of algorithm 1 is not finite in general. In (Guglielmi and Protasov, 2013) it is proposed, in the case of the candidate $\tilde{\Pi}$ having an unique simple leading eigenvalue and there also exists only one s.m.p., to define v_1^* as the leading eigenvector of $\tilde{\Pi}^*$ the conjugate operator of the candidate $\tilde{\Pi}$ normalized by $(v_1, v_1^*) = 1$ as well as

Algorithm 1 invariant-polytope algorithm

V := $\{v_1, \cdots, v_M\}$
$V_{\text{new}} \leftarrow V$
while $V_{\text{new}} \neq \emptyset$ **do**
  $V_{\text{rem}} \leftarrow V_{\text{new}}$
  $V_{\text{new}} \leftarrow \emptyset$
  for $v \in V_{\text{rem}}$ **do**
    for $A \in \mathcal{A}$ **do**
      if $\|Av\|_{\text{co}_\text{s}(V)} \geq 1$ **then**
        $V \overset{\cup}{\leftarrow} Av$
        $V_{\text{new}} \overset{\cup}{\leftarrow} Av$
return $\text{co}_\text{s}(V)$

$v_i^* := \tilde{A}_{d_i}^* \cdots \tilde{A}_{d_k}^* v_1^*$ which are the leading eigenvectors of the conjugates of the cyclic-permutations of $\Pi \in \mathcal{A}^k$. If now $\|Av\|_{\text{co}_\text{s}(V)} > 1$ if there exists a j such that $|(v_j^*, Av)| > 1$ then the chosen candidate is not a s.m.p. and the algorithm either stops ore restarts with a new candidate.

2.4 Termination conditions

There exist some rare cases where the finiteness property either does not hold or there are no invariant polytopes. In those cases the algorithm will not find the value of the JSR.

2.5 Rebalancing and added starting vertices

Three years after publishing the fundamental algorithm with the provided stopping criterion, the writers released a new paper on rebalancing multiple s.m.p.s as well as starting with some extra vertices so the polytope is conditioned better (Guglielmi and Protasov, 2016) in that case its possible to even terminate with multiple s.m.p.s..

2.6 Eigenvector cases

If the eigenvector is complex, then a different norm must be used, a so called complex-balanced-polytope norm. Also in the case of nonnegative matrices a different norm can be used to vastly increase the efficiency. In case of the implementation just the linear programming problem changes.

3 Finite-tree algorithm

In this chapter we introduce basic definitions and concepts to another approach of JSR computation. It is called *finite-tree algrithm* as its trying to decompose arbitrary products $P \in \mathcal{A}^k$ into sub-products that can be 1-bounded by an a priori chosen norm.

3.1 Notation and definitions

Definition 3.1.1. A *tree* is a connected, acyclic graph T with a distinguished node t_0 called the *root*. The tree structure induces a natural hierarchy: for any node t_i, there exists a unique path from t_0 to t_i. The tree can then be described in terms of the following concepts:

- **Parent**: For any node $t \in T$, the node preceding t on the path from the root t_0 is called its *parent*.
- **Children**: The nodes adjacent to a given node t that are further from the root t_0 are called its *children*.
- **Leaf**: A node with no children is called a *leaf*.
- **Internal vertex**: A node with at least one child is called an *internal* or *non-leaf* node.

We define $T_{\mathcal{A}}$ as the tree consisting of all products in the set $\bigcup_{k\in\mathbb{N}_0} \mathcal{A}^k$ where every multi-index represents a node. For the empty index ($k = 0$) we set $A_{[]} = I_d$ the identity matrix in $\mathbb{R}^{d\times d}$. An example can be seen in figure 3.1.

The objective is now to find a sub-tree $T \subset T_{\mathcal{A}}$ for a chosen norm, with the condition of every leaf encoding a product that has a norm less then 1 and every node thats not a leaf having all possible children $A_1, \cdots, A_J$. If such a tree has been found it easily allows for a decomposition of every possible product $P = A_{[i_1,\cdots,i_k]} \in \mathcal{A}^k$ into sub-products encoded by leafs and one possible rest term encoded by an internal-node. Leading to a proof of $\mathrm{JSR}(\tilde{\mathcal{A}}) = 1$.
Such sub-trees are in general not finite and therefore unusable for a computational tree-search algorithm. In the paper (Möller and Reif, 2014), the writers propose to use a different kind of encoding making use of so-called generators and set-valued nodes.

We fixate a set $\mathbf{G} = \{g_1, \cdots, g_m\}$ with $g_i \in \bigcup_{k\in\mathbb{N}} \mathcal{A}^k$ and $\rho(g_i)^{\frac{1}{k}} \leq 1$. All products from this set will be called *generators*. We also define $A_l := g_{-l}$ for $l \in \{-m, \ldots, -1\}$.

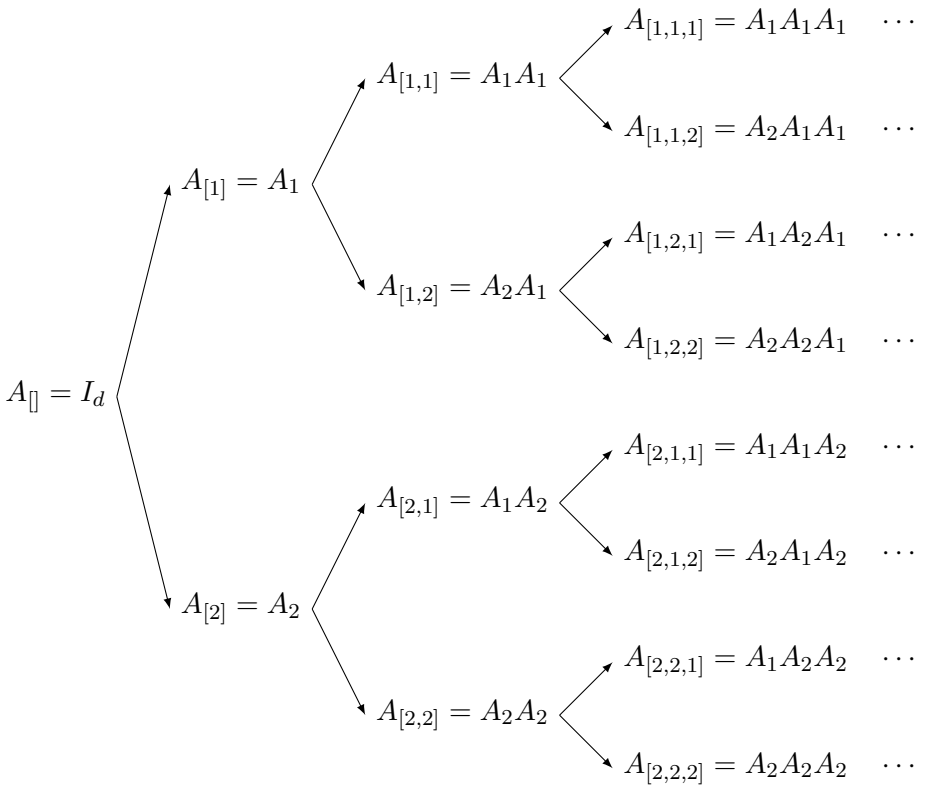

Figure 3.1: An example of the tree $T_{\mathcal{A}}$ on $\bigcup_{k\in\mathbb{N}_0} \mathcal{A}^k$ with $\mathcal{A} = \{A_1, A_2\}$.

Definition 3.1.2. The set $\mathcal{J}^k := \{-m, \ldots, -1, 1, \ldots, n\}^k$ denotes the collection of all multi-indeces of length k with entries in $\{-m, \ldots, -1, 1, \ldots, n\}$. For an multi-index $J = [j_1, \ldots, j_k] \in \mathcal{J}^k$, the corresponding matrix product set is defined by:

$$A_J := \{A_{j_k}^{e_k} \cdots A_{j_1}^{e_1} \mid e_i = 1 \text{ if } j_i > 0 \text{ or } \forall e_i \in \mathbb{N}_0 \text{ else}\}$$

Again this creates a tree structure in a similar way to positive multi-indeces, except now the nodes represent (possibly infinite) sets of products.

Definition 3.1.3. Let $J = [j_1, \cdots, j_k]$ be a multi-index. We define the multi-index extension with $J + j_{k+1} := [j_1, \cdots, j_k, j_{k+1}]$.

Definition 3.1.4. Let $t \in T$ be a node with the multi-index J. A child node $\tilde{t}$ with the multi-index $J + j_{k+1}$ is called:

- *positive* if j_{k+1} is positive.
- *negative* or *generator* if j_{k+1} is negative.

Now we want to find a subtree of a special form.

Definition 3.1.5. A so-called $(\mathcal{A}, \mathbf{G})$-tree T has the following structure:

- The root node contains the identity matrix: $t_0 = \{I_d\}$.
- Each node $t \in T$ is either:
  - A leaf,
  - A parent of exactly n positive children.
  - A parent of only generators.

Definition 3.1.6. A node in the tree is called *covered* if it is a subset of one of its ancestors in the tree. Otherwise, it is called *uncovered*. The set of uncovered leaves is denoted as

$$\mathcal{L}(T) := \{L \in t : t \in T \text{ is an uncovered leaf}\}.$$

and called *leafage* of the tree T.

Definition 3.1.7. An $(\mathcal{A}, \mathbf{G})$-tree T is called *1-bounded* with respect to a matrix norm $\|\cdot\|$ if

$$\sup_{L \in \mathcal{L}(T)} \|L\| \leq 1.$$

If $\sup_{L \in \mathcal{L}(T)} \|L\| < 1$, then the tree is called *strictly 1-bounded*.

3.2 Structure of the finite-tree algorithm

The finite-tree algorithm tries to find a 1-bounded $(\mathcal{A}, \mathbf{G})$-tree. We start with only the root node $t_0 = \{I\}$ and iteratively check in every step $\sup\{\|P\| : P \in L\} \leq 1$ for every leaf L in the current tree. If for one particular leaf the bound doe not hold we add either all positive children or an arbitrary amount of generator children.

Algorithm 2 Finite-tree-algorithm

Input: $\mathcal{A} = \{A_1, \ldots, A_n\}$ and $\mathbf{G} = \{g_1, \cdots, g_m\}$
Output: A boolean value indicating whether the tree condition holds (True/False)
$Q \leftarrow \{[1], \cdots, [n]\}$
while $Q \neq \emptyset$ **do**
  $Q_{\text{new}} \leftarrow \emptyset$
  for $J \in Q$ **do**
    if $\sup\{\|P\| : P \in A_J\} > 1$ **then**
      $\mathbf{J} \leftarrow$ selectChildren$(J, \mathcal{A}, \mathbf{G})$
      $Q_{\text{new}} \overset{\cup}{\leftarrow} \{J + j : j \in \mathbf{J}\}$
  $Q \leftarrow Q_{\text{new}}$
return True

Remark 3.2.1. The procedure selectChildren determines whether all positive children (from 1 to n) or a subset of negative children (from -1 to $-m$) is chosen, according

to the theory established in (Möller and Reif, 2014).

Lemma 3.2.2. *For every $(\mathcal{A}, \mathbf{G})$-tree T there exists a strictly-positive, monotone polynomial p such that for every node $t \in T$ and every product $P \in \mathcal{A}^k$ encoded by t, $\|P\| \leq p(k)$ holds.*

Proof. Take an arbitrary node $t \in T$ with multi-index $J = [j_1, \cdots, j_l]$. Now every j_i encodes either a matrix from $\mathcal{A}$ or arbitrary powers of a generator from $\mathcal{G}$. Since all those matrices have spectral radius less then or equal to 1, its Jordan normal forms grow utmost polynomially und thus, due to equality of norms in finite-dimensional vector spaces, there exists a strictly-positive, monotone polynomial p_i with

$$\|A_{j_i}^e\| \leq p_i(e) \quad \forall e \in \mathbb{N}_0.$$

We define:

$$p_t := p_l \cdots p_1$$
$$p := \sum_{t \in T} p_t$$

The sum over all of T is finite since T has only finitely many nodes. Take now an arbitrary product $P = A_{j_l}^{e_l} \cdots A_{j_1}^{e_1} \in \mathcal{A}^k$ that is encoded by t.

Now we have:

$$\begin{aligned}\|P\| &= \|A_{j_l}^{e_l} \cdots A_{j_1}^{e_1}\| \\ &\leq \|A_{j_l}^{e_l}\| \cdots \|A_{j_1}^{e_1}\| \\ &\leq p_l(e_l) \cdots p_1(e_1) \\ &\leq p_l(k) \cdots p_1(k) \\ &= p_t(k) \leq p(k)\end{aligned}$$

Since the node t and the product P were chosen arbitrary, this concludes the proof. □

Theorem 3.2.3. *If a 1-bounded $(\mathcal{A}, \mathbf{G})$-tree was found for a given matrix set $\mathcal{A}$ and generator set $\mathbf{G}$ then* $\mathrm{JSR}(\mathcal{A}) = 1$

Proof. We take an arbitrary positive multi-index $I = [i_1, \cdots, i_k]$. Our found $(\mathcal{A}, \mathbf{G})$-tree T allows for a unique decomposition of this multi-index into $I_k, \cdots, I_1$ where

$$A_I = A_{I_k} A_{I_{k-1}} \cdots A_{I_1}.$$

Each sub-multi-index $I_j \quad j = 1, \cdots, k-1$ encodes either a leaf of T or is the empty set and I_k encodes any node of T. Now due to Lemma 3.2.2 there exists a polynomial p such that

$$\|A_I\| = \|A_{I_k} A_{I_{k-1}} \cdots A_{I_1}\| \leq \|A_{I_k}\| \cdot \|A_{I_{k-1}}\| \cdots \|A_{I_1}\| \leq \|A_{I_k}\| \leq p(k).$$

The last equality holds since all $I_j \quad j = 1, \cdots, k-1$ are either the empty multi-index or leafs such that their products A_{I_j} are bounded by 1. Furthermore A_{I_k} can be bounded by the polynomial p and the length of the product, which is less then k. Now we have

$$\mathrm{JSR}(\mathcal{A}) = \lim_k \max_{P \in \mathcal{A}^k} \|P\|^{\frac{1}{k}} \leq \lim_k p(k)^{\frac{1}{k}} = 1$$

since the chosen multi-index was arbitrary. □

Remark 3.2.4. The decomposition used in the proof of Theorem 3.2.3 exists and is unique due to the structure of an $(\mathcal{A}, \mathbf{G})$-tree. glueing tree on covered nodes creating an infinite tree that allows huffman encoding directly. Empty sets for every I_i if product already lies on internal node before I_k For more info see (Möller and Reif, 2014).

3.3 Conditions of termination

Remark 3.3.1. In (Möller and Reif, 2014), the authors show that the use of generators is essential for ensuring termination in cases where the original invariant-polytope algorithm fails to terminate. But deciding when to use generators instead of all positive children is still part of active research and is handled via highly heuristical approaches.

4 Hybrid approach

In this chapter we will explore some possible combinations of the abeforementioned algorithms and then present the main result of this work. The tree-flavored-invariant-polytope-algorithm and its termination results.

4.1 Goal and options

In its heart the invariant-polytope algorithm tries to find a polytope, whose corrisponding Minkowski norm is specifically optimized on the given problem, whilst the finite-tree algorithm connects growth-rate to decompositions of products. A clear combination scheme arises naturally, where we use the optimized polytope norm to estimate the products of the finite-tree. From there we can choose a specific order or level of concurrency.
A modular approach would be to first run the invariant-polytope algorithm for a couple of runs and then use the calculated norm thats already optimized for the finite-tree algorithm. But that seems to be wasteful since valuable matrix calculations from the finite-tree algorithm could have been used for an even more optimized norm and some polytopes might have already cleared insight for the decompositions that the finite-tree algorithm tries to find. In (Mejstrik and Reif, 2024) the authors came up with a more concurrent algorithm that builds up norms and decompositions in every step.

4.2 Structure of the hybrid algorithm

4.2.1 Objective

After the preprocessing stage from 1.4 is done, we end up with a scaled set of matrices $\tilde{\mathcal{A}}$ and from the chosen candidate and the three-member-inequality (1.4) we already know $\mathrm{JSR}(\tilde{\mathcal{A}}) \geq 1$. Such that our goal of establishing $\mathrm{JSR}(\tilde{\mathcal{A}}) = 1$ becomes equivalent to $\mathrm{JSR}(\tilde{\mathcal{A}}) \leq 1$. For this we want to show that there exists a polynomial p and a norm $\|\cdot\|$, such that, similarly to chapter 3, every norm of a product of matrices from $\tilde{\mathcal{A}}$ can be bounded by this polynomial in its length i.e.

$$\|P\| \leq p(k) \quad \forall P \in \tilde{\mathcal{A}}^k, k \in \mathbb{N}.$$

We find the polynomial by decomposing every multi-index into leaf

4.2.2 Process

The algorithm starts just as the invariant-polytope algorithm. After the initial set of vertices was calculated we now choose a $(\mathcal{A}, \mathbf{G})$-tree T_v like in (Mejstrik and Reif, 2024) for every vertex $v \in V$. If there exists a product with $\|\|\|$ - after initial, start with A-G-trees in every branch of invariant-poly - refine norm for not 1-bounded trees - old trees stay bounded for new norms

We try to decompose arbitrary products $P \in \mathcal{A}^k$, such that their polytope-norms are less then $p(k)$ where p is a strictly-positive, monotone polynomial. This removes the invariance property of the polytope to be build up, since the norms dont have to be less than 1 but it still proofes the JSR identity because we take the averaged norms in the length of the products in the definition of the JSR.
Starting the loop of the invariant-polytope algorithm with a cycle on top that is connected via the generators factors and also the first branches represented by images from the missing $J - 1$ factors from $\mathcal{A}$. Instead of only adding images under vertices from V and matrices from $\mathcal{A}$ directly, from now on we try to find an $(\mathcal{A}, \mathbf{G})$-tree which is one-bounded i.e its leafage-polytope-norm is less than 1, for every $v \in V$. For that we generate $(\mathcal{A}, \mathbf{G})$-tree patterns in the beginning and just go through every remaining vertex and calculate the leafage-norms. From the structure of those trees we can assume that every matrix in $\mathcal{A}$ represents a node for the first branches. For the branches that lead to a leafage-norm less than or equal 1 we are done, for the other branches we have the choice to go deeper or just add some points to V that changes the leafage-norm of those branches to less than 1. Here we decided to add the points since going deeper just would mean to consider possibly the same products but the tree generation would be more complex with options for depth-first- or breadth-first-search and even using some s.m.p and generator trickery. [might change it in the future]
First points that come to mind are the leafage-points itself since this is what we have tested but generators could be involved meaning there are possibly infinitely many leafage-points. So the next best thing would be the roots of the branches which are guaranteed to be a single matrix from $\mathcal{A}$. This makes tree generation easy and adds points with likely more distance to the faces of the polytope and makes the norm stronger more quickly.
So in principle for every $v \in V_{\text{rem}}$ take a tree from the generating pool, check the leafage-norm for every root branch, if it is larger than 1 add the point from the root branch to V_{new} and V. Repeat this as long as new vertices have been added. We use V for the polytope-norms and since new points are only being added the norms decrease over time so all 1-bounded trees stay bounded.
After termination the set of trees generated promise a valid decomposition for every product from $\mathcal{A}$ into chunks of norm lesser 1 and one suffix thats of norm less than $p(k)$ for some monotone polynomial, which proofes the question if the chosen radius is maximal.

Algorithm 3 Tree-flavored-invariant-polytope-algorithm

Input: $\mathcal{A} = \{A_1, \ldots, A_n\}$, $\mathbf{G} = \{g_1, \cdots, g_m\}$ and $V = \{v_1, \cdots, v_M\}$
Output: A boolean value indicating whether the tree condition holds (True/False)
$V_{\text{new}} \leftarrow V$
while $V_{\text{new}} \neq \emptyset$ **do**
  $V_{\text{rem}} \leftarrow V_{\text{new}}$
  $V_{\text{new}} \leftarrow \emptyset$
  for $v \in V_{\text{rem}}$ **do**
    Construct$(\mathcal{A}, \mathbf{G})$-tree $\mathbf{T}$ with sepcifications from (Mejstrik and Reif, 2024)
    if $\exists L \in \mathcal{L}(T)$ with $\|Lv\|_{\text{cos}(V)} \geq 1$ **then**
      $V \overset{\cup}{\leftarrow} \mathcal{A}v$
      $V_{\text{new}} \overset{\cup}{\leftarrow} \mathcal{A}v$
return True

4.2.3 Outline of proof

The following figure 4.1 shows an example of the generated tree structure as well as the tested $(\mathcal{A}, \mathbf{G})$-trees that have been used to bound the products. Here $\mathcal{A}$ consists of two matrices A and B and the chosen candidate is $\Pi = ABA$. At the top you can see the cycle generated by the candidate and the starting vector v_1 as well as branches coming of, that are the products that stay in contrast to the cycle. This is what i call the crown and it is generated for every matrix set at the beginning, so no testing has been made so far. All the nodes that are connected through a solid arrow have been added to the vertices V.

Now the finite-tree theory comes in and every vertex gets tested by a $(\mathcal{A}, \mathbf{G})$-tree (dashed arrows). Take the vector v_6 for example it has been tested by an tree and the branch starting with the matrix B was sufficient i.e. every leaf-node has a polytope-norm of less than 1 (at the time the polytope consists of the vertices $V = \{v_1, \cdots, v_6\}$). The branch starting with the matrix A on the other hand did not, so the vertex $v_7 = Av_6$ has been added to V and the next tree was tested, which was fully sufficient (on every branch). Since the other branches also terminated earlier the algorithm stops, no more vertices are beeing added and the candidate Π was proofen to be a s.m.p product.

4.3 Termination results

Lemma 4.3.1. *If algorithm 3 terminates there exists an monotone polynomial p such that for every product $P \in \mathcal{A}^k$ and $v \in V$, $\|Pv\|_{cos(V)} \leq p(k)$ holds.*

Proof. If the algorithm terminates, then the set of vertices $V = \{v_1, \cdots, v_m\}$ is finite and for each $v \in V$ there exists an $(\mathcal{A}, \mathbf{G})$-tree T_v such that $\|Lv\|_{\text{cos}(V)} \leq 1$ for every $L \in \mathcal{L}(T_v)$, which means there exists $\lambda_1, \cdots, \lambda_m$ such that $Lv = \sum \lambda_j v_j$ with $\sum |\lambda_j| \leq 1$. According to Lemma **??**, there exists a polynomial p_v such that for every product $P \in \mathcal{A}^k$

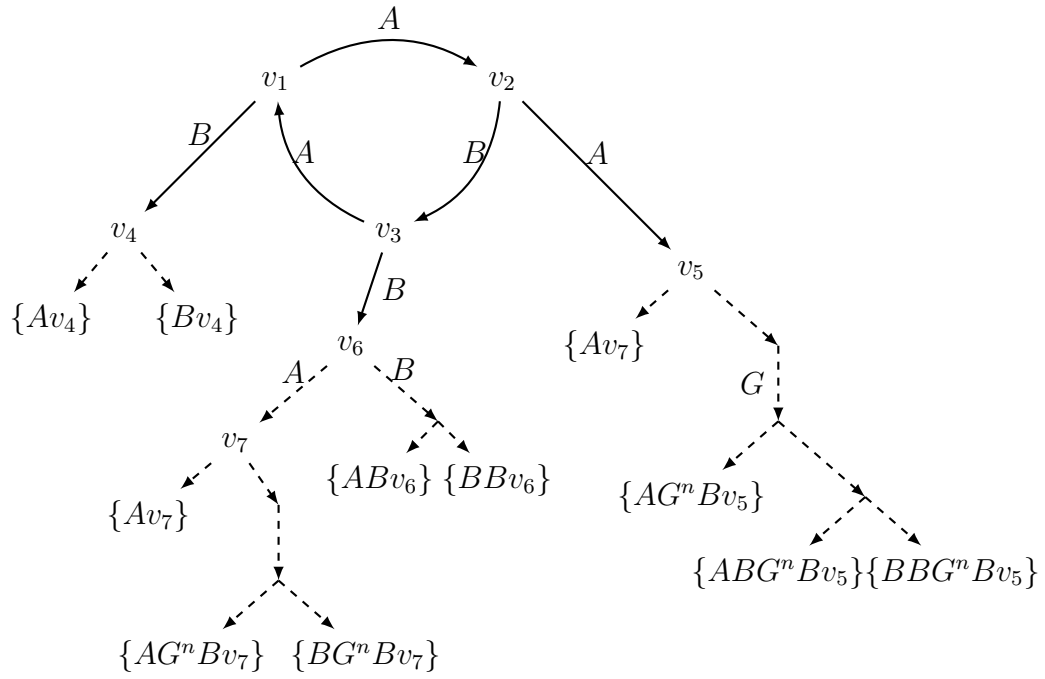

Figure 4.1: Cyclic tree structure generated by the algorithm. Vertices added to V (solid arrows) and finite-trees for bounding products (dashed arrows).

encoded by T_v, $\|Pv\|_{\cos(V)} \leq p_v(k)$. Take an arbitrary product $P \in \mathcal{A}^k$ and $v \in V$. Define:

$$
\begin{aligned}
p &:= \sum_{v \in V} p_v \\
P'L &:= P \text{ with } L \in \mathcal{L}(T_v) \text{ if } P \notin T_v \text{ and } L := I \text{ else} \\
P'_{i_1 \dots i_r} L_{i_1 \dots i_r} &:= P'_{i_1 \dots i_{r-1}} \text{ with } L_{i_1 \dots i_r} \in \mathcal{L}(T_{v_{i_r}}) \text{ if } P'_{i_1 \dots i_{r-1}} \notin T_{v_{i_r}} \\
\text{and } L_{i_1 \dots i_r} &:= I \text{ else.} \\
\text{Where } i_j \in \mathcal{I} &:= \{1, \cdots, m\}
\end{aligned}
$$

These decompositions exist and are unique because of the special structure of $(\mathcal{A}, \mathbf{G})$-trees. A simple prrof can be found in the appendix [appendix proof of decomp] and an in-depth explanaition of finding the decomposition can be found in (Möller and Reif, 2014).

Now we have:

$$\begin{aligned}
\|Pv\|_{co_s(V)} &= \|P'Lv\|_{co_s(V)} \\
&= \|P' \sum_{i_1 \in \mathcal{I}} \lambda_{i_1} v_{i_1}\|_{co_s(V)} \\
&= \sum_{i_1 \in \mathcal{I}} |\lambda_{i_1}| \|P' v_{i_1}\|_{co_s(V)} \\
&= \sum_{i_1 \in \mathcal{I}} |\lambda_{i_1}| \|P'_{i_1} L_{i_1} v_{i_1}\|_{co_s(V)} \\
&= \sum_{i \in \mathcal{I}^2} |\lambda_{i_1}||\lambda_{i_2}| \|P'_i v_{i_2}\|_{co_s(V)} \\
&\dots \\
&= \sum_{i \in \mathcal{I}^k} |\lambda_{i_1}| \cdot \ldots \cdot |\lambda_{i_k}| \|P'_i v_{i_k}\|_{co_s(V)} \\
&\leq \sum_{i \in \mathcal{I}^k} |\lambda_{i_1}| \cdot \ldots \cdot |\lambda_{i_k}| \; p_{v_{i_k}}(k) \\
&\leq p(k) \sum_{i \in \mathcal{I}^k} |\lambda_{i_1}| \cdot \ldots \cdot |\lambda_{i_k}| \\
&= p(k) \sum_{i_1 \in \mathcal{I}} |\lambda_{i_1}| \sum_{i_2 \in \mathcal{I}} |\lambda_{i_2}| \; \ldots \sum_{i_k \in \mathcal{I}} |\lambda_{i_k}| \\
&\leq p(k) \cdot 1 \cdot \ldots \cdot 1 \\
&\leq p(k)
\end{aligned}$$

Here the first inequality works since after k steps the prefix P'_i is guaranteed to be in the tree $T_{v_{i_k}}$ and thus the norm is bounded by $p_{v_{i_k}}(k)$, which in turn is of course less then $p(k)$ by definition. The second to last inequality works since those sums of the absolute values of the coefficients of a linear combination of vectors are less than or equal to 1 from the 1-boundedness of the $(\mathcal{A}, \mathbf{G})$-trees. □

Theorem 4.3.2. *If algorithm 3 terminates then* $\mathrm{JSR}(\mathcal{A}) \leq 1$

Proof. Due to the prework in the lemmas this is now easy to show. Lemma 4.3.1 implies that for every $P \in \mathcal{A}^k$ and $v \in V$, $\|Pv\|_{\mathrm{co_s}(V)} \leq p(k)$ trivially this implies:

$$\forall P \in \mathcal{A}^k : \|P\|_{\mathrm{co_s}(V)} \leq p(k)$$

Now we take the definition of the JSR and get:

$$\mathrm{JSR}(\mathcal{A}) = \lim_{k \geq 1} \max_{P \in \mathcal{A}^k} \|P\|^{\frac{1}{k}} \leq \lim_{k \geq 1} p(k)^{\frac{1}{k}} = 1$$

□

Of course, if the candidate had a spectral radius of 1 $\implies$ $\mathrm{JSR}(\mathcal{A}) = 1$.

Corollary 4.3.3. *If the chosen trees are always* $T_v = \mathcal{A}$*, then we would have exactly the*

invariant-polytope algorithm 1. Which makes the new approach a real generalization of the original.

4.4 Remarks

Used trees

It was proven in (Möller and Reif, 2014) that in order to have a bigger solution space then algorithm 1 the use of generators is mandatory. The choice of the testing-trees is not trivial and subject to active research. We propose to use so-called *minimal trees* that fulfill every necessary and beneficial condition of the $(\mathcal{A}, \mathbf{G})$-trees but keep the amount of nodes minimal.

Polytope invariance

Since the polytope-norms of matrices in $\mathcal{A}$ are allowed to be bigger than 1 just less than $p(1)$ it is possible and very likely the construccted polytope is not invariant under $\mathcal{A}$ in general anymore but eventually invariant.

4.5 Implementation

Definition 4.5.1. An $(\mathcal{A}, \mathbf{G})$-tree with just one generator at the root, one path from root to the covered node and all neccessarily added branches, to make it comply with definition 3.1.5, is called a *minimal tree*. This tree only depends on the set $\mathcal{A}$ and the chosen generator G. Its denoted by $T_{(\mathcal{A},\mathbf{G})}$ and if the context of $\mathcal{A}$ is clear, it is shortened to $T_{\mathbf{G}}$.

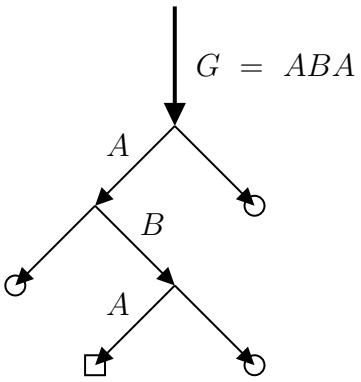

Figure 4.2: Minimal tree structure $T_{(G,\mathcal{A})}$ for the candidate $G = ABA$ and $\mathcal{A} = \{A, B\}$. The arrows marked with circles are uncovered-leafs, the arrow marked with a square is a covered-leaf

5 Numerics

5.1 Runtime of algorithm 3 on generated matrices

5.2 Comparison algorithm 3 and 1

5.3 Comparison algorithm 3 and 2

5.4 Solving open problems

6 Conclusion

6.1 New approach

6.2 Solved Problems

6.3 Efficiency

6.4 Unsolved problems

6.5 Further research

6.5.1 Complex case

6.5.2 Nonnegative case

6.5.3 Optimizing tree generation and vertex choice

Bibliography

Barabanov, N. E. (1988). Lyapunov indicator of discrete inclusions. 1. *Automation and Remote Control*, 49(2):152–157.

Blondel, V. D. and Tsitsiklis, J. N. (2000a). The boundedness of all products of a pair of matrices is undecidable. *Systems & Control Letters*, 41(2):135–140.

Blondel, V. D. and Tsitsiklis, J. N. (2000b). A survey of computational complexity results in systems and control. *Automatica*, 36(9):1249–1274.

Daubechies, I. and Lagarias, J. C. (1992). Sets of matrices all infinite products of which converge. *Linear algebra and its applications*, 161:227–263.

Gripenberg, G. (1996). Computing the joint spectral radius. *Linear Algebra and its Applications*, 234:43–60.

Guglielmi, A. (2012). A personal perspective on deep inference and computer science.

Guglielmi, N. and Protasov, V. (2013). Exact Computation of Joint Spectral Characteristics of Linear Operators. *Foundations of Computational Mathematics*, 13(1):37–97.

Guglielmi, N. and Protasov, V. Y. (2016). Invariant polytopes of sets of matrices with application to regularity of wavelets and subdivisions. *SIAM Journal on Matrix Analysis and Applications*, 37(1):18–52.

Guglielmi, N. and Zennaro, M. (2008). An algorithm for finding extremal polytope norms of matrix families. *Linear Algebra and its Applications*, 428(10):2265–2282.

Heil, C. (1993). Ten Lectures on Wavelets (Ingrid Daubechies). *SIAM Review*, 35(4):666–669.

Jungers, R. M. (2009). The Joint Spectral Radius. *none*.

Mejstrik, T. (2020). Improved invariant polytope algorithm and applications.

Mejstrik, T. and Reif, U. (2024). Hybrid approach to the joint spectral radius computation.

Möller, C. and Reif, U. (2014). A tree-based approach to joint spectral radius determination. *Linear Algebra and its Applications*, 463:154–170.

Protasov, V. Y. (1996). The joint spectral radius and invariant sets of the several linear operators. *Fundamentalnaya i prikladnaya matematika*, 2(1):205–231.

Rota, G.-C. and Gilbert Strang, W. (1960). A note on the joint spectral radius. *Indagationes Mathematicae (Proceedings)*, 63:379–381.

Tsitsiklis, J. N. and Blondel, V. D. (1997). The lyapunov exponent and joint spectral radius of pairs of matrices are hard—when not impossible—to compute and to approximate. *Mathematics of Control, Signals and Systems*, 10:31–40.