universität wien

MASTERARBEIT | MASTER'S THESIS

Titel | Title

Termination results for hybrid approach of Joint Spectral Radius computation

verfasst von | submitted by

Aaron Pumm, BA

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

Master Zusatz (Abkürzung)

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

Contents

Zusammenfassung

Hier gehört die Zusammenfassung in Deutsch hin.

Abstract

English abstract here.

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 sequences.

The concept gained significant traction in the 1990s when researchers began exploring its theoretical properties and practical implications. Notable advancements include its application in control theory, where it is used to analyze the stability of switched linear systems (Blondel and Tsitsiklis, 2000), and in wavelet theory, where it assists in the construction of refinable functions (citepdaubechies1992ten). 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 (Jungers, 2009).

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.

To fully grasp the subsequent mathematical framework, the reader should be familiar with linear algebra, specifically matrix norms, eigenvalues, and spectral radius. A basic understanding of combinatorial optimization and algorithm design will also be beneficial.

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 form 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 on ... problems to analyze the efficiency and applicability of the hybrid algorithm. Chapter 6 concludes with insights and future directions.

1.1 Theoretical background

The *joint spectral radius* (JSR) of a set of matrices is a generalization of the spectral radius for a single matrix. For a finite set of matrices $\mathcal{A} = \{A_1, A_2, \ldots, A_m\}$, the JSR is formally defined

as:

$$JSR(\mathcal{A}) = \lim_{k\to\infty} \max_{A_i\in\mathcal{A}} \|A_{i_k} \dots A_{i_1}\|^{1/k}, \tag{1.1}$$

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

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

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

Now if we consider this and take asymptotic equality into account we get:

$$\begin{aligned}
&\lim_{k\to\infty} \max_{A_i\in\mathcal{A}} \|A_{i_k} \dots A_{i_1}\|_1^{1/k} \\
&\leq \lim_{k\to\infty} (\frac{1}{c})^{\frac{1}{k}} \max_{A_i\in\mathcal{A}} \|A_{i_k} \dots A_{i_1}\|_2^{1/k} \\
&= \lim_{k\to\infty} \max_{A_i\in\mathcal{A}} \|A_{i_k} \dots A_{i_1}\|_2^{1/k} \\
&\leq \lim_{k\to\infty} C^{\frac{1}{k}} \max_{A_i\in\mathcal{A}} \|A_{i_k} \dots A_{i_1}\|_1^{1/k} \\
&= \lim_{k\to\infty} \max_{A_i\in\mathcal{A}} \|A_{i_k} \dots A_{i_1}\|_1^{1/k}
\end{aligned}$$

□

A simple application would be the categorization of the stability of linear switched dynamical systems in discrete time. lets define $v_{n+1} := A_{i_n} v_n$ where $i_n \in \{1, \cdots, m\}$. Now the system is stable (i.e. $v_n \to 0$) exactly if $JSR(\mathcal{A}) < 1$.

This hints at the more general problem of finding whether $JSR(\mathcal{A}) < 1$ which is proofen to be mathematically undecidable [Jungers] as seen in a future section. This and some other complexity results are the reason of vast research efforts in JSR analysis and numerics.

To understand the state-of-the-art algorithms considered as well as the main results that will follow, it is necessary to introduce some basic properties of the JSR.

Homogeneity

For any scalar α and set of matrices $\mathcal{A}$, the scaling property

$$JSR(\alpha\mathcal{A}) = |\alpha| JSR(\mathcal{A}) \tag{1.2}$$

holds, which follows directly from norm homogenity.

Irreducibility

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 reducable it is called irreducable.

Three-member inequality

The *three-member inequality* provides essential bounds for the JSR. For any submultiplicative matrix norm $\|\cdot\|$, the inequality

$$\max_{P\in\mathcal{A}^k} \rho(P)^{\frac{1}{k}} \leq JSR(\mathcal{A}) \leq \max_{P\in\mathcal{A}^k} \|P\|^{\frac{1}{k}}, \tag{1.3}$$

holds for every $k \in \mathbb{N}$ (Jungers, 2009). 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 $\limsup$).

Minimum over norms

The JSR can be equivalently defined as the minimum over all submultiplicative norms:

$$JSR(\mathcal{A}) = \min_{\|\cdot\|} \max_{A\in\mathcal{A}} \|A\|. \tag{1.4}$$

Proof. equivalence

□

Complexity and the Finiteness Property

The computation of the JSR is known to be computationally challenging, with determining its exact value classified as NP-hard. However, certain matrix sets exhibit the *finiteness property*, which states that there exists a finite sequence of matrices $A_{i_1}, \ldots, A_{i_k}$ such that:

$$JSR(\mathcal{A}) = \|A_{i_k} \cdots A_{i_1}\|^{1/k}. \tag{1.5}$$

While this property does not hold universally, it is essential for algorithmic approaches.

Candidates and Generators

Approximating the JSR 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.

Similarity and reducibility

In the following section it is assumed that the set of interest is irreducable for some later results. If the set happens to be reducable there is an easy work-around but for that we need an intermediate result: For any invertable matrix T and reducable set $\mathcal{A}$

$$JSR(\mathcal{A}) = JSR(T^{-1}\mathcal{A}T) \tag{1.6}$$

holds. Now per definition there exists a change of basis such that all $A \in \mathcal{A}$ are block-triangularized:

$$\exists T : T^{-1}A_iT = \begin{bmatrix} B_i & C_i \\ 0 & D_i \end{bmatrix}$$

Now

$$JSR(\mathcal{A}) = \max\{JSR(\{B_i\}), JSR(\{D_i\})\} \tag{1.7}$$

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

1.2 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 and posesses the finiteness property.
2. **Candidates**: Efficiently find products $P = A_{i_k} \cdots A_{i_1}$ of matrices from $\mathcal{A}$ that maximize the averaged-spectral radius $\hat{\rho} := \rho(P)^{\frac{1}{k}}$ for a given maximal length $k_{\max}$.
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 JSR$(\tilde{\mathcal{A}}) = 1$ using the three-member-inequality. By homogenity this is equivalent to JSR$(\mathcal{A}) = \hat{\rho}$.

The considered algorithms only differ in step 4, while the invariant-polytope algorithm tries to find a norm that bounds the products of length 1 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 1-bounded and some rest-term that doesnt 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 aims to advance the computational tools available for JSR analysis combining efficiency and a wide solution-space of the former.

2 Invariant-polytope algorithm

In this chapter we bring our interest to the underlying invariant-polytope algorithm. One result about JSR computation, that every irreducible family possesses an invariant norm is helpful. We observe that there always exists an norm that is in some sense extremal.

2.1 Extremal norms

A norm is called invariant if it is preserved under the action of the family of matrices. That is, for all $A_j \in \mathcal{A}$ and $x \in \mathbb{R}^d$ we have $\|A_j x\| = \|x\|$.

Theorem 2.1.1. *Every irreducible family $\mathcal{A}$ possesses an invariant norm.*

Definition 2.1.2. *A norm $\|\cdot\|$ is called extremal for a family of matrices $\mathcal{A}$ if $\|A_j x\| \leq \rho(\mathcal{A})\|x\|$ for all $x \in \mathbb{R}^d$ and $A_j \in \mathcal{A}$.*

Every invariant norm is extremal

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**
                $V \leftarrow V \cup Av$
                $V_{\text{new}} \leftarrow V_{\text{new}} \cup Av$
return $\text{co}_\text{s}(V)$

3 Finite-tree algorithm

Definition 3.0.1. $(\mathcal{A}, \mathbf{G})$*-tree A Tree with the following structure:*

1. *The root consists of the identity matrix* I
2. *The root has* $\{A : A \in \mathcal{A}\}$ *as children*
3. *every other node is either a leaf, has exactly* $\{A : A \in \mathcal{A}\}$ *or any amount of generators* $A_g \in \mathbf{G}$ *as children*

Is called $(\mathcal{A}, \mathbf{G})$*-tree*

4 Hybrid approach

In this chapter we want to explore some possible combinations of the before mentioned algorithm schemes and then present the main result of this work, the Tree-flavored-invariant-polytope-algorithm and its termination results.
In its heart the invariant-polytope algorithm tries to find a norm thats specifically optimized on the given problem, whilst the finite-tree algorithm connects growth 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.
The most modular approach would be to first run the invariant-polytope algorithm for a couple of runs and then use the calculated norm thats specially 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. So after a bit of rethinking we managed to come up with a more concurrent algorithm that builds up norms and decompositions in every step.

Algorithm 2 Tree-flavored-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**
        Construct some $(\mathcal{A}, \mathbf{G})$-tree $\mathbf{T}$
        for $L = L'A \in \mathcal{L}(T)$ with $A \in \mathcal{A}$ **do**
            if $\|Lv\|_{\text{co}_s(V)} \geq 1$ **then**
                $V \leftarrow V \cup Av$
                $V_{\text{new}} \leftarrow V_{\text{new}} \cup Av$
return

4.1 Structure of the hybrid algorithm

We try to decompose arbitrary products P from factors from $\mathcal{A}$, such that their polytope-norms are less then $p(k)$ where k is the number of factrors from P and p is a 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 weighted norms in the length of the products in the three-member-inequality 1.3.
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. After one step change V_{rem} to V_{new} and V_{new} to the empty set and 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 like the one in the introduction. Which proofes the question if the chosen radius is maximal.

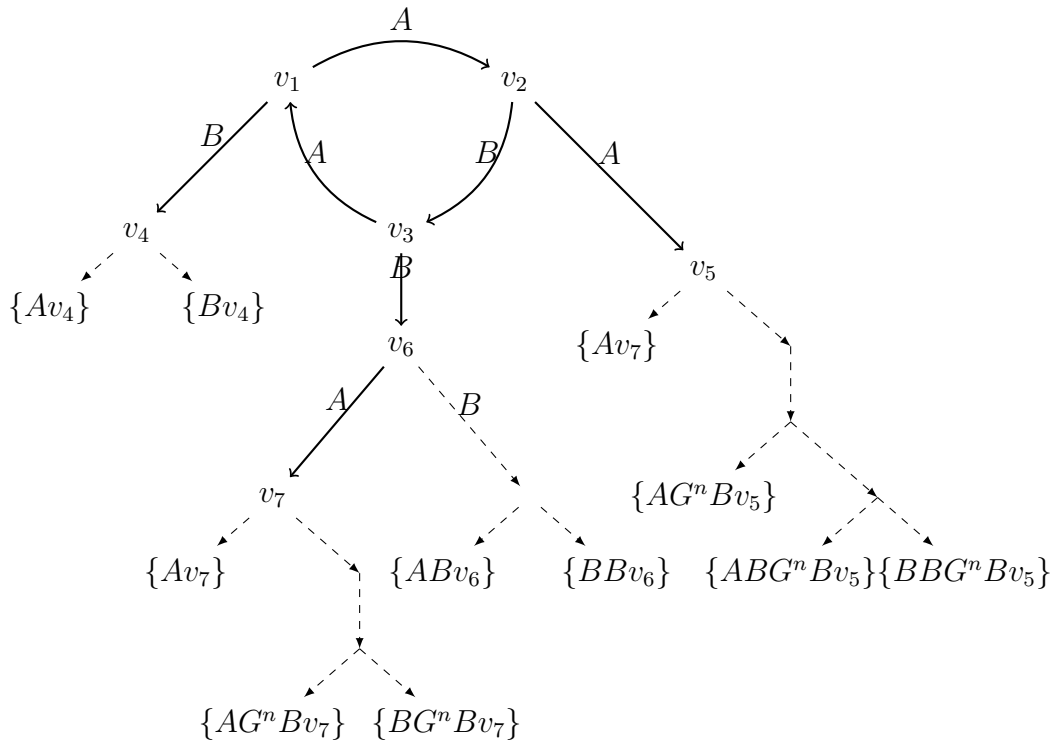

Figure 4.1: Cyclic tree structure generated by the algorithm. Vertices added to V (solid arrows) and finite-trees for bounding products (dashed arrows)

The algorithm creates a so-called cyclic-tree structure, where the starting set V is the root-cycle connected via the s.m.p theory $\Pi = ABA \implies v_2 = Av_1$ etc. After termination we have $(\mathcal{A}, \mathbf{G})$-Trees to our disposal which lets us create a decomposition for arbitrary products from $\mathcal{A}$ into smaller products profen to be less than 1 in the $\|\cdot\|_{co(V)}$ norm. For this we use the linear combination for each vector from vertices from V. Since we take the symmetriced convex hull and the vectors lie within the polytope spanned by V, all the factors sum to at most 1 in absolute.

4.2 Termination results

Theorem 4.2.1. *If Algorithm 2 terminates then* $JSR(\mathcal{A}) \leq 1$

Proof. Suppose the algorithm terminates and creates a set V of vertices then for each $v \in V$ there exists an $(\mathcal{A}, \mathbf{G})$-Tree $\mathbf{T}_v$ such that $\|Lv\|_{\mathbf{co}_s(V)} \leq 1 \quad \forall L \in \mathcal{L}(\mathbf{T}_v)$.
Taking a random product $P \in \mathcal{A}^k$ and a random $v \in \mathbf{co}_s(V)$ we get:

$$\|Pv\| = \|P \sum \lambda_i v_i\| \leq \sum \lambda_i \|Pv_i\| \quad \text{with} \quad \sum |\lambda_i| \leq 1$$

Now for every $i = 1 \cdots |V|$ there either exists a Tree-partition where $P = P_i' L_i$ with $L_i \in \mathcal{L}(T_{v_i})$ and $\|L_i v_i\| \leq 1$ or P is element of a node of T_{v_i} thats not a leave. If the partition exists

the vector $L_i v_i$ lands within the symmetrized convex hull of V and thus has a linear combination like before.

$$\|Pv_i\| = \|P'_i(L_i v_i)\| \leq \sum \mu_j \|P'_i v_j\| \quad \text{with} \quad \sum |\mu_j| \leq 1$$

If there exists no such partition then the product is in some sense already small enough. Now a similar argument, like in [finite tree paper] can be made, that theres only finitely many nodes in every tree and the spectral radius of every factor of such a matrix-product is less then 1 so norms of such products can be bounded by a monotone increasing polynomial $p(k)$ in the amount of factors k.
Since every product can be reduced to a product that has less factors until it lies within a according tree we eventually get:

$$\|Pv\| \leq \sum_i \lambda_i \|Pv_i\| \leq \sum_{i_1} \sum_{i_2} \lambda_{i_1,i_2} \|P'_{i_1} v_{i_2}\| \leq \cdots$$

$$\leq \sum_{i_1,\cdots,i_k \in \sigma} \lambda_{i_1,\cdots,i_k} \|P'_{i_1,\cdots,i_k} v_{i_k}\| \leq \sum_{i_1,\cdots,i_k \in \sigma} \lambda_{i_1,\cdots,i_k} p(k) \leq p(k)$$

wich implies $\|P\| \leq p(k)$ thus

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

□

Now the question is can the hybrid algorithm bring together efficiency and wide solution space from the algorithms it originated from or does it lack generalization. Lets say for a given problem $JSR(\mathcal{A})$ both invariant-polytope algorithm and hybrid algorithm estimated the same s.m.p and calculates the same cyclic root from there on the invariant polytope algorithm behaves like the hybrid approach with $\{A \in \mathcal{A}\}$ as $(\mathcal{A}, \mathbf{G})$-trees so in fact the hybrid algorithm could produce the same result under the same efficiency. In general the chosen trees will be different and also will make use of the generators so the calculated points might differ but the norms are only getting stronger and more of the proofing is being done by finding those trees so naturally it will find the solution. On the other side there are plenty of problems the efficient invariant-polytope algorithm cannot solve for instance a set with multiple s.m.ps or special eigenvalue cases for this the generators should give a good new tool to overcome problems. For this we make few numerical tests with suggested problems and compare the results in the *Chapter 5*.

5 Numerical testing

6 Conclusion

Bibliography

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

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

Guglielmi, N. and Protasov, V. (2011). Exact computation of joint spectral characteristics of linear operators.

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. (2015a). Invariant polytopes of linear operators with applications to regularity of wavelets and of subdivisions. *none*.

Guglielmi, N. and Protasov, V. Y. (2015b). Invariant polytopes of linear operators with applications to regularity of wavelets and of subdivisions.

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. (2020a). Algorithm 1011: Improved Invariant Polytope Algorithm and Applications. *ACM Transactions on Mathematical Software*, 46(3):1–26.

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

Mejstrik, T. and Reif, U. (2024a). A Hybrid Approach to Joint Spectral Radius Computation. *none*.

Mejstrik, T. and Reif, U. (2024b). 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.

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