Approximating the joint spectral radius using a genetic algorithm framework ⋆

Chia-Tche Chang *, **Vincent D. Blondel** *

* *ICTEAM Institute, Université catholique de Louvain*
Avenue Georges Lemaître, 4, B-1348 Louvain-la-Neuve, Belgium
(e-mail: {chia-tche.chang, vincent.blondel}@uclouvain.be)

Abstract: The joint spectral radius of a set of matrices is a measure of the maximal asymptotic growth rate of products of matrices in the set. This quantity appears in many applications but is known to be difficult to approximate. Several approaches to approximate the joint spectral radius involve the construction of long products of matrices, or the construction of an appropriate extremal matrix norm. In this article we present a brief overview of several recent approximation algorithms and introduce a genetic algorithm approximation method. This new method does not give any accuracy guarantees but is quite fast in comparison to other techniques. The performances of the different methods are compared and are illustrated on some benchmark examples. Our results show that, for large sets of matrices or matrices of large dimension, our genetic algorithm may provide better estimates or estimates for situations where these are simply too expensive to compute with other methods. As an illustration of this we compute in less than a minute a bound on the capacity of a code avoiding a given forbidden pattern that improves the bound currently reported in the literature.

Keywords: Joint spectral radius, generalized spectral radius, genetic algorithms, product of matrices, dynamic systems, discrete-time systems.

1. INTRODUCTION

The *joint spectral radius (jsr)* $\rho(\Sigma)$ of a set of matrices $\Sigma \subset \mathbb{R}^{n\times n}$ is a quantity characterizing the maximal asymptotic growth rate of products of matrices in the set. More precisely, it is defined by:

$$\rho(\Sigma) = \lim_{t\to\infty} \rho_t(\Sigma), \tag{1}$$

with:

$$\rho_t(\Sigma) = \max\left\{ \|M\|^{1/t} \mid M \in \Sigma^t \right\},$$

independently of the chosen submultiplicative matrix norm. Here, Σ^t denotes the set of products of length t of matrices in Σ. In the particular case where $\Sigma = \{M\}$, the jsr is equal to the usual spectral radius, i.e., the largest magnitude of the eigenvalues.

The jsr was introduced by Rota and Strang in Rota and Strang (1960) and has since then appeared in many applications such as stability of switched systems (Gurvits (1995)), continuity of wavelets (Daubechies and Lagarias (1992)), combinatorics and language theory (Jungers et al. (2009)), capacity of codes (Moision et al. (2001)), etc.

The issue of approximating the jsr has been widely studied. The first algorithms proposed consisted in constructing products of increasing length and using (1) as upper bounds. Lower bounds can be obtained using the *generalized spectral radius* $\bar{\rho}(\Sigma)$, defined by:

$$\bar{\rho}(\Sigma) = \limsup_{t\to\infty} \bar{\rho}_t(\Sigma),$$

with:

$$\bar{\rho}_t(\Sigma) = \max\left\{ \rho(M)^{1/t} \mid M \in \Sigma^t \right\}.$$

Indeed, we have the following inequalities for all t (see Jungers (2009)):

$$\bar{\rho}_t(\Sigma) \le \bar{\rho}(\Sigma) \le \rho(\Sigma) \le \rho_t(\Sigma). \tag{2}$$

The generalized spectral radius is equal to the jsr if Σ is a bounded (in particular, a finite) set (see Berger and Wang (1992)). This generalizes thus the well-known Gelfand formula for the spectral radius of a single matrix.

Unfortunately, the sequence of bounds in (2) converges slowly to $\rho(\Sigma)$ except in some particular cases, and so any approximation algorithm directly based on these inequalities is bound to be inefficient. The problem of approximating the jsr is indeed NP-Hard (see Tsitsiklis and Blondel (1997)). Several branch-and-bound methods have been designed, some of which even allowing arbitrarily accurate approximations (see Gripenberg (1996)), but this is thus at the expense of a high computation time. In order to speed up this procedure, one could try to find an appropriate norm that gives a fast convergence rate. In some cases one can even find a norm that is *extremal* for some set of matrices, that is, a norm such that the jsr is reached with $t = 1$. More precisely, we have the following definition:

Definition 1. (Extremal norm). A norm $\|\cdot\|$ is said to be extremal for a set of matrices Σ if

⋆ This paper presents research results of the Belgian Network DYSCO (Dynamical Systems, Control, and Optimization), funded by the Interuniversity Attraction Poles Programme, initiated by the Belgian State, Science Policy Office. The scientific responsibility rests with the authors. Chia-Tche Chang is a F.R.S.-FNRS Research Fellow (Belgian Fund for Scientific Research).

$$\|M\| \leq \rho(\Sigma) \qquad \text{for all } M \in \Sigma.$$

We also have the property (see Berger and Wang (1992)):

$$\rho(\Sigma) = \inf_{\|\cdot\|} \sup_{M\in\Sigma} \|M\|, \tag{3}$$

where the infimum is taken on the set of matrix norms. So, an extremal norm attains the infimum in (3).

In order to obtain a good approximation of the jsr using matrix norms, we can thus use two different approaches. Either we try to approximate an extremal norm, i.e., build a sequence of norms that converges to an extremal norm, or we solve the optimization problem (3), taking the infimum on a well-chosen set of norms which hopefully contains an extremal norm. Most recent approximation algorithms focus on building an adequate norm to obtain good bounds on the jsr. In our approach, we focus on building an adequate product of matrices at a low computational cost in order to obtain good lower bounds thanks to (2). This is achieved by the use a genetic algorithm framework: the method starts with a set of random products of limited length and iteratively generates new products by heuristically combining existing ones depending on their performance. The maximal allowed product length is slowly increased during the computation and each new product may provide a better bound. Experimental results tend to show that the bounds obtained by our method are tighter than those obtained by other algorithms and are often optimal on examples of small size. The required computation time is also much smaller, however, there is no a priori guarantee on the quality of the bounds returned by our method.

The remainder of the paper is organized as follows. In the following section, we will present a brief survey of existing methods to approximate the jsr, such as branch-and-bound or semidefinite optimization. In the subsequent section, we present a new approach to this problem using genetic algorithms. The different methods are then compared on several examples in order to illustrate their performances and finally, we end with some conclusions in Section 5.

2. EXISTING METHODS APPROXIMATING THE JOINT SPECTRAL RADIUS

2.1 A branch-and-bound approach to generate matrix products

A first method for the approximation of the jsr is Gripenberg's branch-and-bound algorithm (see Gripenberg (1996)). This algorithm is based on (2) but tries to reduce the number of products to consider by removing candidates known to be suboptimal.

More precisely, given a tolerance ε, the algorithm starts with the set $\Pi_1 = \Sigma$ of candidates, the lower bound $\alpha_1 = \max_{M\in\Sigma} \rho(M)$ and the upper bound $\beta_1 = \max_{M\in\Sigma} \|M\|$ for a fixed norm $\|\cdot\|$. At the k^{th} iteration, we define:

$$\Pi_k = \{MP \mid M \in \Sigma, P \in \Pi_{k-1}, \mu(MP) > \alpha_{k-1} + \varepsilon\},$$

where $\mu(M_1 \dots M_k) = \min_{1\leq i\leq k} \|M_1 \dots M_i\|^{1/i}$. The bounds are then updated as follows:

$$\alpha_k = \max\left\{\alpha_{k-1}, \max_{P\in\Pi_k} \rho(P)^{1/k}\right\},$$

$$\beta_k = \min\left\{\beta_{k-1}, \max\left\{\alpha_{k-1} + \varepsilon, \max_{P\in\Pi_k} \mu(P)\right\}\right\}.$$

At each iteration, the bounds satisfy $\alpha_k \leq \rho(\Sigma) \leq \beta_k$ and we have $\lim_{k\to\infty}(\beta_k - \alpha_k) \leq \varepsilon$. However, the number of iterations required to achieve this absolute accuracy of ε is not known a priori and may depend much on the choice of the norm $\|\cdot\|$. It may also be necessary to analyze very long products in order to reach the tolerance threshold.

2.2 The ellipsoidal norm approximation

As mentioned in the introduction, an alternative approach to the construction of long products is to try to find an extremal norm, for instance by considering (3). As optimizing on the set of all matrix norms is not easy, one can instead consider a smaller class of norms so that the modified problem becomes easily solvable. One such example is the class of *ellipsoidal norms*.

Given a positive definite matrix $P \in \mathbb{R}^{n\times n}$, which is denoted by $P \succ 0$, we can define a vector norm with $\|x\|_P = \sqrt{x^T P x}$ for $x \in \mathbb{R}^n$. The associated *ellipsoidal* norm is the corresponding induced matrix norm:

$$\|M\|_P = \max_{x\neq 0} \frac{\|Mx\|_P}{\|x\|_P} = \max_{x\neq 0} \frac{\sqrt{x^T M^T P M x}}{\sqrt{x^T P x}}. \tag{4}$$

The ellipsoidal norm approximation $\widehat{\rho}_{Ell}(\Sigma)$ is defined in Blondel et al. (2005) by:

$$\widehat{\rho}_{Ell}(\Sigma) = \inf_{P\succ 0} \max_{i=1,\dots,q} \|M_i\|,$$

which is obviously an upper bound on the jsr.

This approximation can be easily computed using semidefinite programming. Indeed, (4) implies that for all i:

$$\|M_i\|_P^2 x^T P x \geq x^T M_i^T P M_i x, \qquad \forall x \in \mathbb{R}^n,$$

which can be rewritten as:

$$x^T \left(\|M_i\|_P^2 P - M_i^T P M_i\right) x \geq 0, \qquad \forall x \in \mathbb{R}^n.$$

This means that for every i, $\|M_i\|_P^2 P - M_i^T P M_i$ is a positive semidefinite matrix. The ellipsoidal norm approximation can thus be viewed as the minimum value of γ such that there exists a positive definite matrix $P \succ 0$ with $\gamma^2 P - M_i^T P M_i \succeq 0$ for all i. For a given value of γ, the problem of finding a matrix P corresponds to an SDP feasibility problem, and the optimal value of γ can be found by bisection.

It is possible to prove the following bounds for the ellipsoidal norm approximation:

Proposition 2. For an arbitrary set of q $n \times n$ matrices Σ, the ellipsoidal norm approximation $\widehat{\rho}_{Ell}(\Sigma)$ of the jsr $\rho(\Sigma)$ satisfies the following inequality:

$$\frac{1}{\sqrt{\min\{n, q\}}} \widehat{\rho}_{Ell}(\Sigma) \leq \rho(\Sigma) \leq \widehat{\rho}_{Ell}(\Sigma).$$

Notice that in order to obtain bounds of arbitrary accuracy, one can compute the ellipsoidal norm approximation on a lifted set of matrices. Indeed, we have the following identity (see Blondel and Nesterov (2005)):

$$\rho(\Sigma)^k = \rho(\Sigma^k) = \rho(\Sigma^{\otimes k}),$$

where $\Sigma^{\otimes k}$ denotes the kth Kronecker power of Σ.

2.3 A generalization using sum-of-squares

In Parrilo and Jadbabaie (2008), the authors propose a generalization of the ellipsoidal norm approximation, by replacing the norms by polynomials. Upper bounds on the jsr can be obtained using the following theorem:

Proposition 3. Let $\Sigma = \{M_1, \dots, M_q\}$ be a set of $n \times n$ matrices and $p(x)$ be a strictly positive homogeneous polynomial of degree $2d$ with n variables that satisfies $p(M_i x) \leq \gamma^{2d} p(x) \forall x \in \mathbb{R}^n$ for all i. Then, $\rho(\Sigma) \leq \gamma$.

Even though positive polynomials are hard to characterize, a positivity constraint can be relaxed into a sum-of-squares constraint: instead of $p(x) \geq 0$, we require the existence of a decomposition $p(x) = \sum_i p_i(x)^2$. A sum of squares is obviously nonnegative, whereas most (but not all) nonnegative polynomials can be rewritten as sums of squares. The sum-of-squares decomposition can also be written as $p(x) = \left(x^{[d]}\right)^T P x^{[d]}$, where $x^{[d]}$ is a vector containing all monomials of degree d with n variables, and $P \succeq 0$ is positive semidefinite. The problem of checking if a polynomial is a sum-of-squares is thus equivalent to an SDP feasibility problem. Again, the optimal value of γ can be found by bisection.

The sum-of-squares approximation can thus be defined by:

$$\begin{aligned} \widehat{\rho}_{SOS,2d}(\Sigma) \;=\; & \min\ \gamma \\ & \text{s.t. } \exists\, p(x) \in \mathbb{R}[x]_{2d} \text{ homogeneous} \\ & \qquad p(x), \gamma^{2d} p(x) - p(Mx) \text{ are SOS } \forall M \in \Sigma. \end{aligned}$$

In the particular case $d = 1$, we have the same problem as with the ellipsoidal norm case: all quadratic nonnegative polynomials are SOS. In the general case, we have the following bounds on the approximation accuracy:

Proposition 4. For an arbitrary set of q $n \times n$ matrices Σ, the sum-of-squares approximation $\widehat{\rho}_{SOS,2d}(\Sigma)$ of the jsr $\rho(\Sigma)$ satisfies the following inequality:

$$\left(\frac{1}{\sqrt{\eta}}\right)^d \widehat{\rho}_{SOS,2d}(\Sigma) \leq \rho(\Sigma) \leq \widehat{\rho}_{SOS,2d}(\Sigma),$$

where $\eta = \min\left\{\binom{n+d-1}{d}, q\right\}$.

In theory, one can thus obtain arbitrary sharp approximations by taking polynomials of sufficiently large degree, but the computational cost increases accordingly.

Another generalization of the ellipsoidal norm approximation has also been studied in Protasov et al. (2010), where the authors extend the SDP problem to conic programming in the general case. Indeed, this extension allows to derive upper and lower bounds on the jsr, provided that the matrices in Σ leave a common cone K *invariant*, i.e., $MK \subset K$ for all $M \in \Sigma$. In particular, for an arbitrary set Σ, it is possible to build a set $\widetilde{\Sigma}$ leaving the cone $\mathbb{S}^n_+$ invariant and satisfying $\rho(\widetilde{\Sigma}) = \rho(\Sigma)^2$. The result is then equivalent to the ellipsoidal norm approximation.

2.4 Iterative schemes to construct a Barabanov norm

A particular class of extremal norms, viz. *Barabanov norms*, can also be considered for the approximation of the jsr. These norms are defined as follows:

Definition 5. (Barabanov norm). Given a set of $n \times n$ matrices $\Sigma = \{M_1, \dots, M_q\}$, a vector norm $\|\cdot\|$ on $\mathbb{R}^n$ is said to be a Barabanov norm for Σ if we have the condition $\max_i \|M_i x\| = \rho(\Sigma)\|x\|$ for all $x \in \mathbb{R}^n$.

All Barabanov norms are thus extremal.

Two different iterative schemes are proposed in Kozyakin (2010a,b) for the construction of a Barabanov norm. Suppose we are given an arbitrary initial norm on $\mathbb{R}^n$. In the first algorithm (called the *Linear relaxation iteration* or *LR-procedure*), we build a sequence of norms by using linear combinations of already computed norms. Bounds on the jsr are available at each iteration.

More precisely, the LR-procedure defines a sequence of norms $(\|\cdot\|_k)_{k\in\mathbb{N}}$ according to the following rules:

- Start with a norm $\|\cdot\|_0$ on $\mathbb{R}^n$ and let $e \neq 0$ be a vector such that $\|e\|_0 = 1$. Let us also choose λ^-, λ^+ such that $0 < \lambda^- \leq \lambda^+ < 1$.
- At each iteration, bounds on the jsr are given by $\rho(\Sigma) \in [\rho_k^-, \rho_k^+]$ with:
$$\rho_k^+ = \max_{x\neq 0} \frac{\max_i \|M_i x\|_k}{\|x\|_k},\ \rho_k^- = \min_{x\neq 0} \frac{\max_i \|M_i x\|_k}{\|x\|_k}.$$
- Let λ_k be in the interval $[\lambda^-, \lambda^+]$ and define the new norm $\|\cdot\|_{k+1}$ as follows:
$$\|x\|_{k+1} = \lambda_k \|x\|_k + (1-\lambda_k)\frac{\max_i \|M_i x\|_k}{\max_i \|M_i e\|_k}.$$

This procedure converges to a Barabanov norm, and the two sequences $\left(\rho_k^{\pm}\right)_{k\in\mathbb{N}}$ converge to $\rho(\Sigma)$. Alternatively, one can also apply the *Max-relaxation iteration* that replaces the linear combination with a maximum operation and an averaging function. The MR-procedure has similar convergence properties as the LR-procedure, and more details can be found in Kozyakin (2010a,b).

2.5 An approximation of the joint spectral radius using polytope norms

A geometric approach involving polytopes has been studied in Cicone et al. (2010). The procedure has been used in Guglielmi and Zennaro (2009) where complex polytopes are used to compute the jsr of real matrices:

Definition 6. (Balanced complex polytope). Let $\mathcal{P} \subset \mathbb{C}^n$ be bounded. $\mathcal{P}$ is called a balanced complex polytope if there exists a finite set of vectors $\mathcal{V} = \{v_1, \dots, v_k\}$ such that $\text{span}(\mathcal{V}) = \mathbb{C}^n$ and $\mathcal{P} = \text{absconv}(\mathcal{V})$, that is:

$$\mathcal{P} = \left\{ x \in \mathbb{C}^n \;\middle|\; x = \sum_{i=1}^k \lambda_i v_i \text{ with } \sum_{i=1}^k |\lambda_i| \leq 1, \lambda_i \in \mathbb{C} \right\}.$$

A *complex polytope norm* is any norm whose unit ball is a balanced complex polytope. It is known (see Guglielmi and Zennaro (2007)) that the set of induced matrix polytope norms is dense in the set of induced matrix norms, so (3) holds even if we take the infimum on all polytope norms.

In order to find a polytope norm, the authors present an algorithm which essentially considers the trajectory of a vector $\tilde{x}$ under all possible products of matrices in Σ^*. If $\tilde{x}$ is well-chosen and some hypotheses hold, then the convex hull of the trajectory will describe a balanced polytope,

which will give us the value of the jsr. More precisely, it is supposed that the set Σ has the finiteness property, i.e., its jsr is reached by some periodic product: $\rho(\Sigma) = \rho(M_{i_1} \dots M_{i_t})^{1/t}$. The vector $\tilde{x}$ is then taken as a leading eigenvector of a candidate product. This method requires thus a good initial guess, but if this candidate product is optimal, then it can be certified by the algorithm. Note that the finiteness property does *not* hold for all sets of real matrices (see Bousch and Mairesse (2002); Blondel et al. (2003); Hare et al. (2011)) even though it seems that it is the case for nearly all sets arising in practice.

3. FINDING A LOWER BOUND WITH A GENETIC ALGORITHM

In practice, the performance of all these techniques varies widely. A branch-and-bound method such as Gripenberg's algorithm provides an interval containing the value of the jsr, but the computation time becomes prohibitive when a small interval is required. Optimization methods such as the ellipsoidal norm approximation only give an upper bound but may be able to find the exact value as upper bound in some cases. However, the computation time increases rapidly when the size of the problem grows and numerical problems may become a significant issue. The balanced polytope technique requires a good initial guess, and other geometric algorithms such as the LR-procedure require the manipulation of geometric objects. The accuracy of the result is thus significantly influenced by the discretization level and numerical errors.

Hence, the objective of our algorithm is to provide approximations of the jsr at a low computational cost. Thanks to (2), as any product of finite or infinite length can provide a lower bound, the idea of our method is to initially generate a given number of products, and then try to combine them in order to obtain candidates that can provide "good" lower bounds. These results can also be combined with another algorithm, e.g., a method that checks if a given candidate is optimal, or that returns an upper bound on the jsr. We have chosen the genetic algorithm framework for these purposes. There will be no guarantee on the quality of the bounds returned by the algorithm, but this heuristic approach does not require expensive computation steps, which is the main objective here.

The different steps of the algorithm are presented below:

- *Initialization.* Evaluate the spectral radius all products of matrices of length at most κ, where κ is chosen depending on $|\Sigma|$ in order to limit this step to a small number of products. Let $K = 2\kappa$. Create a population P_0 of size S. Each element of this set is a product of matrices $M_{i_1} \dots M_{i_k}$, satisfying $k \leq K$.
- *Evaluation.* At each generation g, the spectral radius of all products of the corresponding population P_g is evaluated in order to obtain lower bounds on the jsr $\rho(\Sigma)$. If this improves the current lower bound on $\rho(\Sigma)$, the algorithm explores the neighborhood of the product and tries to obtain a better lower bound. More precisely, the neighborhood corresponds to all products that can be obtained with a single deletion, substitution or insertion of a matrix in the candidate product, while keeping a length of at most K. If such an improvement is possible, then the corresponding product is inserted in the population, replacing the worst product in the set.
- *Selection.* In order to obtain the next population P_{g+1}, the current population P_g is partitioned into several subsets. The n_A best products are kept unchanged, while the n_B worst population elements are discarded and replaced by randomly generated products, for some values of n_A and n_B. The remaining products are obtained using crossover operations so that the new population has size S.
- *Crossover I.* A first crossover operation generates products of the form $M_{a_1} \dots M_{a_c} M_{b_{c+1}} M_{b_k}$ for some value of c and where both $M_{a_1} \dots M_{a_k}$ and $M_{b_1} \dots M_{b_k}$ are among the 50% best products in P_g. This corresponds thus to swapping operations between two products. At most $\frac{S}{2}$ elements of the next population P_{g+1} are obtained using the crossover I.
- *Crossover II.* A second crossover gives products of the form $M_{i_1} \dots M_{i_k}$. For each product, the matrix M_{i_j} at position j corresponds to either M_{a_j} or M_{b_j}, where $M_{a_1} \dots M_{a_k}$ and $M_{b_1} \dots M_{b_k}$ are products present in P_g. This corresponds thus to mixing operations between two elements.
- *Mutation.* A mutation operation is applied on the new population P_{g+1}: each element has a given probability to be mutated. If this is the case, then some matrices in the product are replaced by random matrices chosen in Σ. The number of modified positions can be fixed. It can also depend on the value of K, e.g., if we choose to modify x% of the positions.
- *Stopping criteria.* If there is no improvement on the lower bound during T_1 generations, then the maximum allowed length K is increased. If the best lower bound continues to stall during $T_2 > T_1$ iterations, or if the total number of generations reaches a specified threshold, then the algorithm terminates and returns the best lower bound found.

To summarize, the algorithm considers a set of S products of length at most K, with K slowly increasing during the computation. New candidates are generated by heuristically combining existing products according to their performance. The current lower bound is updated with each new product and the algorithm terminates if there is no improvement during too many iterations. The generation of new candidates is mainly done by the crossover steps, and some random perturbation may be applied at each generation to ensure an exploration of the search space.

4. NUMERICAL RESULTS

The different algorithms presented in this paper have been implemented in Matlab®. Some results [1] are presented below in order to compare several techniques for the approximation of the jsr. Main parameters used for the genetic algorithm are $15 \leq S \leq 100$, $T_1 \in \{10, 15\}$, and $T_2 \in \{50, 100\}$. In particular, results shown in the tables correspond to $S = 100$, $T_1 = 15$, $T_2 = 100$. Tests have been repeated 25 times in order to take the randomness of the genetic algorithm into account.

[1] Experiments have been done with Matlab® 7.6.0 (R2008a) and SeDuMi 1.21 on an Intel® Core™2 Duo 2.80 GHz CPU with 3 GiB RAM and Ubuntu Linux 10.04. The implementation of Gripenberg's algorithm has been done by G.Gripenberg.

Table 1. Bounds on $\rho(\Sigma(D))$ obtained with Gripenberg's algorithm, the ellipsoidal norm and SOS approximations and our genetic algorithm. The number of matrices in $\Sigma(D)$ and their size is given for each example.

$D_1 = \{++\}$, $(\lvert\Sigma\rvert, n) = (2, 2)$, Exact value $\rho = \phi$		
Gripenberg	$1.6180 \leq \rho \leq 1.6180$	< 1 s
Ellipsoidal	$\rho \leq 1.6180$	< 1 s
SOS($d = 2$)	$\rho \leq 1.6180$	< 1 s
Genetic	$1.6180 \leq \rho$	3 s
$D_2 = \{0++-\}$, $(\lvert\Sigma\rvert, n) = (4, 8)$, Exact value $\rho \approx 1.7549$		
Gripenberg	$1.7549 \leq \rho \leq 1.7649$	100 s
Ellipsoidal	$\rho \leq 1.7556$	3 s
SOS($d = 2$)	$\rho \leq 1.7549$	226 s
Genetic	$1.7549 \leq \rho$	4 s
$D_3 = \{0+0+\}$, $(\lvert\Sigma\rvert, n) = (16, 8)$, Exact value ρ unknown		
Gripenberg	$1.6585 \leq \rho \leq 1.6685$	120 s
Ellipsoidal	$\rho \leq 1.6822$	20 s
SOS($d = 2$)	$\rho \leq 1.6663$	about 32,000 s
Genetic	$1.6585 \leq \rho$	9 s
$D_4 = \{00+0-\}$, $(\lvert\Sigma\rvert, n) = (256, 16)$, Exact value ρ unknown		
Gripenberg	$1.6663 \leq \rho \leq 1.9663$	150 s
Ellipsoidal		Out of memory
SOS($d = 2$)		Out of memory
Genetic	$1.6738 \leq \rho$	17 s

4.1 Capacity of codes

In Moision et al. (2001), the authors use the jsr to compute the capacity of binary codes avoiding a set of forbidden difference patterns. The jsr gives a measure of the maximal theoretical performance we can obtain with a code subject to a specific type of constraint, the forbidden difference constraint. The number of matrices in the set Σ and their size n may both grow exponentially with the length of the forbidden patterns. Four algorithms have been tested on the sets $\Sigma(D)$ arising from sets D consisting of a single forbidden pattern, for all patterns of length 2 to 5. This is illustrated in Table 1. Methods such as the LR-procedure are not showed in this part due to numerical issues, the large memory consumption and high computation time required to achieve a good accuracy for such high-dimensional problems. In fact, these memory requirements are already too expensive for the set D_3 with 8×8 matrices.

The set D_1 shows a trivial case with $\rho(\Sigma(D_1)) = \phi = \frac{1+\sqrt{5}}{2}$. The four presented methods reach this value, but the genetic algorithm is slower due to the nature of its stopping criterion as the methods needs to stall during a given number of iterations. In fact, the running time will not be significantly influenced by the size of the problem, even if obtaining good results will be more difficult as the search space increases in size. In the second example, Σ contains four 8×8 matrices and there is already a significant increase in time for Gripenberg's algorithm and the sum-of-squares approach. However, they both reach the exact value of ρ, which is not the case for the ellipsoidal norm approximation. This is even more apparent with D_3, where the SOS method returns a (suboptimal) bound after nearly 9 hours. The exact value of the jsr is unknown and is thus contained in the interval $[1.6585; 1.6663]$. This lower bound has been found by both Gripenberg's algorithm and the genetic algorithm, the corresponding product following the pattern $M_1^2 M_2 M_3^2 M_4$, which is nontrivial. For a larger size such as with D_4, solving the SDP problem requires too much memory for this configuration of Matlab®/SeDuMi. Gripenberg's algorithm was able to return an interval in about 2.5 minutes, but this is at the expense of the interval length, which is much larger than in the previous cases. The genetic approach has been able to find a better lower bound with a product of the form $M_1 M_2 M_3 M_4 M_3 M_2$, but the exact value is still unknown. As $|\Sigma| = 256$, examining all products is computationally prohibitive, even for length 4 or 5. The balanced polytope technique with the product found by the genetic algorithm as candidate is subject to the same problem when the problem size is large, even if it does work on small instances.

Table 2. Quality of the bounds obtained with different algorithms on 100 sets of random matrices. The three columns represent the number of cases where the bound found by the method was equal to the exact value of the jsr, distant of at most 10^{-4} or 10^{-1} respectively. The number of points for the LR and MR algorithms corresponds to the number of discretization points on the unit ball of the norms.

Lower bounds	Exact	$< 10^{-4}$	$< 10^{-1}$
LR or MR (10^4 points)	0	46	100
LR or MR (5×10^5 points)	0	100	100
Gripenberg (500 evals)	96	96	100
Brute-force (length ≤ 10)	97	97	100
Genetic algorithm	98	98	100
Upper bounds	**Exact**	**$< 10^{-4}$**	**$< 10^{-1}$**
Brute-force (length ≤ 10, $\lVert\cdot\rVert_2$)	0	6	76
Ellipsoidal norm	0	56	77
Sum-of-squares ($d = 2$)	0	78	93
Gripenberg (500 evals, $\lVert\cdot\rVert_2$)	0	55	100
LR or MR (10^4 points)	0	46	100
LR or MR (5×10^5 points)	0	100	100

4.2 Pairs of random matrices

The different algorithms have also been tested on 100 pairs of randomly generated 2×2 matrices. The entries were chosen between -5 and 5 with uniform distribution. Lower and upper bounds were obtained using different methods. This may also provide the exact value, e.g., when the two bounds coincide, or when a product of matrices can be certified as optimal using techniques such as the balanced polytope algorithm. The results are shown in Table 2.

As the size of the problem is quite small, it was possible to apply a brute-force algorithm enumerating all products of given length. This also shows that there were three cases where the optimal product has a length greater than 10. The LR- and MR-procedures are converging to the exact value, but their accuracy depends on the discretization level. Furthermore, the performance is much weaker for higher problem sizes. Indeed, obtaining a uniform discretization of a unit ball in $\mathbb{R}^2$ is much easier than in $\mathbb{R}^n$ for $n \geq 3$. Optimization methods may give fairly good results at a low cost, but more accurate upper bounds are much more expensive and numerical errors become non-negligible. Concerning Gripenberg's branch-and-bound method, stopping the algorithm after 500 evaluations of norms and spectral radii gives a success rate of 96% for lower bounds. However, measuring this perfor-

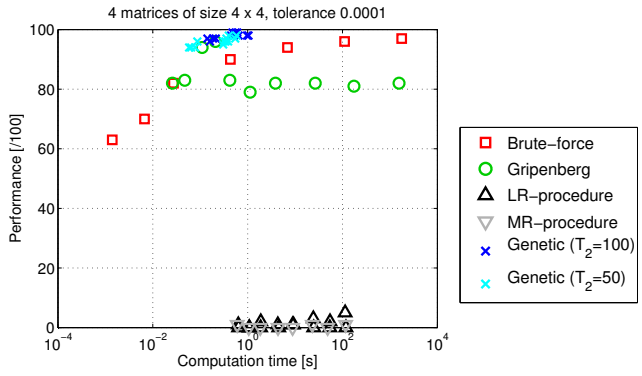

Fig. 1. Lower bounds on the jsr for 100 sets of four random 4×4 matrices. The performance is the number of cases with results of relative accuracy less than 10^{-4}.

mance after 1000 evaluations gives a lower success rate. Indeed, the algorithm is then subject to numerical problems for several instances due to the large product lengths.

If we increase the problem size, for example by considering lower bounds for sets of four 4×4 matrices (see Fig. 1), it appears that LR- and MR-procedures fail in nearly all cases, due to numerical issues. Gripenberg's algorithm is still able to provide some good lower bounds, but the success rate is significantly lower. In fact, if we increase $|\Sigma|$ and n to 16, the performance for the same tolerance drops below 50% whereas the genetic algorithm is still able to find better bounds in a couple of seconds. Of course, the main drawback is that there is no information about the accuracy of these results without relying on another method. As far as upper bounds are concerned, only Gripenberg's method manages to achieve an acceptable success rate, but already at a high computational cost.

5. CONCLUSION

The approximation of the joint spectral radius is a difficult computational problem. Methods constructing long products were introduced first and, can only provide approximations of limited accuracy in practice, due to excessive computation time. Lyapunov methods such as the ellipsoidal norm method, or the SOS approach may provide good upper bounds or the exact value in several cases but are not bound to work in general due to numerical problems when the optimization problem increases in size. Methods explicitly building an extremal norm are reasonably fast and may perform well but are subject to requirements such as the need of a good discretization, which can be difficult to obtain and has a high computational cost.

We propose a heuristic approach based on the genetic algorithm in order to obtain lower bounds at a low cost. Although there is no a priori guarantee on the quality of the bounds, numerical experiments tend to show that the method may give significantly good results. However, another method has to be used in order to assess their quality. More investigation on the parameters of the method may also lead to an improved version of the algorithm.

REFERENCES

Berger, M.A. and Wang, Y. (1992). Bounded semi-groups of matrices. *Linear Algebra Appl.*, 166, 21–27.

Blondel, V.D. and Nesterov, Y. (2005). Computationally Efficient Approximations of the Joint Spectral Radius. *SIAM J. Matr. Anal. Appl.*, 27(1), 256–272.

Blondel, V.D., Nesterov, Y., and Theys, J. (2005). On the accuracy of the ellipsoid norm approximation of the joint spectral radius. *Linear Algebra Appl.*, 394(1), 91–107.

Blondel, V.D., Theys, J., and Vladimirov, A.A. (2003). An Elementary Counterexample to the Finiteness Conjecture. *SIAM J. Matr. Anal. Appl.*, 24(4), 963–970.

Bousch, T. and Mairesse, J. (2002). Asymptotic height optimization for topical IFS, Tetris heaps, and the finiteness conjecture. *Journal of AMS*, 15(1), 77–111.

Cicone, A., Guglielmi, N., Serra-Capizzano, S., and Zennaro, M. (2010). Finiteness property of pairs of 2x2 sign-matrices via real extremal polytope norms. *Linear Algebra Appl.*, 432(2-3), 796–816.

Daubechies, I. and Lagarias, J.C. (1992). Two-Scale Difference Equations II. Local Regularity, Infinite Products of Matrices and Fractals. *SIAM J. Math. Anal.*, 23(4), 1031–1079.

Gripenberg, G. (1996). Computing the joint spectral radius. *Linear Algebra Appl.*, 234, 43–60.

Guglielmi, N. and Zennaro, M. (2007). Balanced complex polytopes and related vector and matrix norms. *Journal of Convex Analysis*, 14, 729–766.

Guglielmi, N. and Zennaro, M. (2009). Finding extremal complex polytope norms for families of real matrices. *SIAM J. Matr. Anal. Appl.*, 31(2), 602–620.

Gurvits, L. (1995). Stability of discrete linear inclusion. *Linear Algebra Appl.*, 231, 47–85.

Hare, K.G., Morris, I.D., Sidorov, N., and Theys, J. (2011). An explicit counterexample to the Lagarias-Wang finiteness conjecture. *Advances in Mathematics*, 226(6), 4667–4701.

Jungers, R.M. (2009). *The Joint Spectral Radius: Theory and Applications*. Springer-Verlag, Berlin, Germany.

Jungers, R.M., Protasov, V.Y., and Blondel, V.D. (2009). Overlap-free words and spectra of matrices. *Theor. Comp. Sci.*, 410(38-40), 3670–3684.

Kozyakin, V.S. (2010a). A relaxation scheme for computation of the joint spectral radius of matrix sets. *Journal of Difference Equations and Applications*, 1–16.

Kozyakin, V.S. (2010b). Iterative building of Barabanov norms and computation of the joint spectral radius for matrix sets. *Discrete and Continuous Dynamical Systems - Series B*, 14(1), 143–158.

Moision, B.E., Orlitsky, A., and Siegel, P.H. (2001). On codes that avoid specified differences. *IEEE Trans. Inf. Theory*, 47(1), 433–442.

Parrilo, P.A. and Jadbabaie, A. (2008). Approximation of the joint spectral radius using sum of squares. *Linear Algebra Appl.*, 428(10), 2385–2402.

Protasov, V.Y., Jungers, R.M., and Blondel, V.D. (2010). Joint spectral characteristics of matrices: a conic programming approach. *SIAM J. Matr. Anal.*, 31(4), 2146–2162.

Rota, G.C. and Strang, G. (1960). A note on the joint spectral radius. *Indag. Math.*, 22, 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. *Math. Control Sign. Syst.*, 10(1), 31–40.