INTERNATIONAL JOURNAL OF ROBUST AND NONLINEAR CONTROL
Int. J. Robust. Nonlinear Control 2016; **00**:1–17
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/rnc

Worst-case stability and performance with mixed parametric and dynamic uncertainties

P. Apkarian[1*] D. Noll[2]

[1] *Control System Department, ONERA, 2, av. Ed. Belin, 31055, Toulouse, France*
[2] *Institut de Mathématiques de Toulouse, 118 route de Narbonne F-31062, Toulouse, France*

SUMMARY

This work deals with computing the worst-case stability and the worst-case H_∞ performance of Linear Time-Invariant systems subject to mixed real parametric and complex dynamic uncertainties in a compact parameter set. Our novel algorithmic approach is tailored to the properties of the non-smooth worst-case functions associated with stability and performance, and this leads to a fast and reliable optimization method, which finds good lower bounds of μ. We justify our approach theoretically by proving a local convergence certificate. Since computing μ is known to be NP-hard, our technique should be used in tandem with a classical μ upper bound to assess global optimality. Extensive testing indicates that the technique is practically attractive. Copyright © 2016 John Wiley & Sons, Ltd.

Received ...

KEY WORDS: Robustness analysis, parametric uncertainty, dynamic uncertainties, non-smooth optimization

1. PROBLEM SPECIFICATION

Consider an LFT and LTI plant with real parametric or dynamic complex uncertainties $\mathcal{F}_u(P,\Delta)$ as in Figure 1, where

$$P(s):\left\{\begin{array}{rcccccc}\dot{x} & = & Ax & + & B_\delta w_\delta & + & B_2 w \\ z_\delta & = & C_\delta x & + & D_{\delta\delta} w_\delta & + & D_{\delta w} w \\ z & = & C_z x & + & D_{z\delta} w_\delta & + & D_{zw} w\end{array}\right. \tag{1}$$

and $x \in \mathbb{R}^n$ is the state, $w \in \mathbb{R}^{m_1}$ a vector of exogenous inputs, and $z \in \mathbb{R}^{p_1}$ a vector of regulated outputs. The uncertainty channel is defined as

$$w_\delta = \Delta z_\delta, \tag{2}$$

where the uncertain matrix Δ has the block-diagonal form

$$\Delta = \mathrm{diag}\left[\Delta_1, \ldots, \Delta_m\right] \in \mathbb{C}^{r\times c} \tag{3}$$

with blocks Δ_i in one of the following categories:

- $\Delta_i := \delta_i I_{r_i}$, $\delta_i \in \mathbb{R}$ for real parametric uncertainties,
- $\Delta_i \in \mathbb{C}^{p_i \times q_i}$ for complex dynamic uncertainties.

*Correspondence to: P. Apkarian, ONERA, Toulouse, France. E-mail: Pierre.Apkarian@onera.fr

Without loss we assume that Δ evolves in the 2-norm unit ball $\mathbf{\Delta} = \{\Delta : \overline{\sigma}(\Delta) \le 1\}$. This means $\delta_i = [-1, 1]$ for real parameters, and $\overline{\sigma}(\Delta_i) \le 1$ for complex blocks.

Assessing robust stability of the uncertain system (1)-(2) over $\mathbf{\Delta}$ can be based on maximizing the spectral abscissa of the system A-matrix over $\mathbf{\Delta}$, that is,

$$\alpha^* = \max\{\alpha(A(\Delta)) : \Delta \in \mathbf{\Delta}\}, \tag{4}$$

where

$$A(\Delta) = A + B_\delta \Delta (I - D_{\delta\delta}\Delta)^{-1} C_\delta, \tag{5}$$

and where the spectral abscissa of a square matrix A is defined as $\alpha(A) = \max\{\mathrm{Re}\,\lambda : \lambda \text{ eigenvalue of } A\}$. Since A is stable if and only if $\alpha(A) < 0$, robust stability of (1)-(2) over $\mathbf{\Delta}$ is certified as soon as $\alpha^* < 0$, while a destabilizing $\Delta^* \in \mathbf{\Delta}$ is found as soon as $\alpha^* \ge 0$.

Our second problem is similar in nature, as it allows to verify whether the uncertain system (1)-(2) satisfies a robust H_∞ performance over $\mathbf{\Delta}$. This can be tested by computing the worst-case scenario:

$$h^* = \max\{\|T_{wz}(\Delta)\|_\infty : \Delta \in \mathbf{\Delta}\}, \tag{6}$$

where $\|\cdot\|_\infty$ is the H_∞-norm, and where $T_{wz}(\Delta, s)$ is the transfer function $z(s) = \mathcal{F}_u(P(s), \Delta)w(s)$, obtained by closing the loop between (1) and (2) in Figure 1.

Note, however, that a decision in favor of robust stability over $\mathbf{\Delta}$ based on $\alpha^* < 0$ in (4), or a decision in favor of robust performance $\|T_{wz}(\Delta)\|_\infty \le h^*$ in (6), is only valid when global maxima over $\mathbf{\Delta}$ are computed. Unfortunately, global optimization of (4) and (6) is known to be NP-hard [1, 2], and it is therefore of interest to develop fast and reliable local solvers to compute lower bounds of α^* and h^*. Computing upper and lower bounds is useful in its own right. Upper bounds give conservative estimates of the size of allowable uncertainties, while lower bounds indicate critical uncertain scenarios, where the system looses stability or performance. Note that when these bounds are close then little information has been lost in analyzing the system robustness.

Systems featuring only real uncertain parameters have been frequently studied in the literature. In contrast, the case of mixed real and complex uncertainties (3) is less explored. This is in large parts due to the non-smooth character of the underlying optimization problems (4) and (6). In particular, as soon as one of the complex constraints $\overline{\sigma}(\Delta_i) \le 1$ is active at the optimum, the maximum singular value $\overline{\sigma}(\Delta_i)$ generally has multiplicity greater than one, which creates an annoying non-smoothness. Standard NLP solvers designed for smooth optimization problems will then encounter numerical difficulties, which lead to deadlock or convergence to non-optimal points. For instance, Halton *et al.* [3] report this type of phenomenon and observe that convergence fails in the vast majority of cases.

Pioneering approaches to computation of mixed-μ lower bounds are the power iteration algorithm (PIA) of [4, 5], and the gain-based algorithm (GBA) of [6]. As reported in [7], PIA is highly efficient for purely complex problems, but experiences typical difficulties for mixed uncertainties. In [8] the authors demonstrate that GBA can be considered a valid workaround in these cases, provided it is used in tandem with a suitable regularization technique. Yet another attractive alternative for parametric robust stability (3) is the pole migration technique (PMT) of [9]. PMT is based on a continuation method, which traces pole trajectories as functions of the uncertainty. The difficulty in PMT is pole coalescence, which may be rather intricate to handle. Among a plethora of papers dedicated to computing the worst-case H_∞-norm we also mention [10], which proposes a coordinate ascent technique with line search driven by Hamiltonian bisection. The authors observe that the approach is computationally demanding in the case of mixed uncertainties.

Using non-smooth optimization to compute lower bounds is not entirely new. A first attempt was made in [11], where the authors compute worst-case uncertainties via non-smooth optimization to achieve singularity in the loop transfer. Their approach is often exceedingly slow and prone to complications or failure, as the determinant is not a reliable indicator of singularity. Finally, a combined Hamiltonian and gradient-based approach is proposed in [12].

In this work we present a novel approach to worst-case stability and H_∞ performance, which has the following two key elements:

DOI: 10.1002/rnc

i. A non-smooth ascent algorithm tailored to objective functions like those in (4) and (6). We prove that iterates converge to a locally optimal solution from an arbitrary starting point.
ii. Experimental testing. We demonstrate the efficiency of our algorithm for mixed uncertainties, and indicate that sole complex or real uncertainties no longer require special handling.

We also emphasize that programs (4) and (6) are particularly useful in an inner relaxation technique to solve robust control design problems [13]. Both programs can be understood as oracles for generating *bad* scenarios, which are successively taken into account in the controller design task to improve robustness.

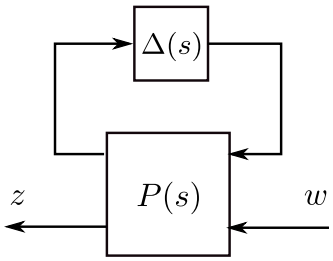

Figure 1. Robust system interconnection

2. APPROACH

Programs (4) and (6) are NP-hard when solved to global optimality, and it is therefore of avail to develop fast and reliable local optimal solvers to compute good lower bounds. Those may be used within a global optimization strategy to obtain global certificates. Even though the use of local optimization techniques generally alleviates the difficulty, a new complication arises in the present context due to the non-smooth character of both optimization programs. This concerns not only the objectives of (4) and (6), but also the semi-definite constraints $\overline{\sigma}(\Delta_j) \le 1$, a fact which further complicates matters.

Algorithmic ways to address the local minimization of the spectral abscissa α and the H_∞-norm $\|T_{wz}\|_\infty$ have been studied in the literature at least since [14, 15] and [16]. However, here we are facing the relatively new problem of *maximizing* these criteria, or in the more standard terminology of local optimization, of minimizing $-\alpha$, and the negative H_∞-norm $-\|T_{wz}\|_\infty$. Not unexpectedly, this leads to completely different challenges. A first analysis of this type of problem was presented in [13], and in that reference a tailored bundle method was proposed. Here we shall investigate a trust-region algorithm, which has the advantage that step-sizes can be controlled more tightly, and that a suitable polyhedral norm, better adapted to the structure of the problem, can be used. In contrast, the traditional bundle approach is somewhat fused on the Euclidean norm.

As in [13], bundling is required in response to the non-smoothness of the criteria, but for this to take effect, we first have to deal with the non-smooth constraints $\overline{\sigma}(\Delta_j) \le 1$. We propose to reduce them to simpler box-constraints by a change of variables. Even though this turns out arduous technically, it is ultimately beneficial as it avoids the use of penalization techniques such as augmenting constraints [17, 18], exact penalization [19], or the progress function techniques of [20], [21, Chapter 2], as those approaches often exhibit slow convergence.

In the sequel, we first deal with the semi-definite constraints $\Delta \in \mathbf{\Delta}$. Our new trust-region algorithm is presented in section 4. Convergence aspects of the algorithm are covered in Section 5. Suitable stopping criteria and subgradient computation for worst-case functions $-\alpha$ and $-\|T_{wz}\|_\infty$ are addressed in section 6. The numerical assessment with comparisons is developed in section 7.

DOI: 10.1002/rnc

3. SEMIDEFINITE CONSTRAINTS

In this section we discuss the semidefinite constraints $\overline{\sigma}(\Delta_i) \le 1$ arising from the complex blocks in (4) and (6). We show how those can be reduced to more convenient box constraints by way of a change of variables. This is made possible by the following key result [22, 5], which we slightly extend to our context.

Lemma 1. *Consider an uncertainty structure* $\Delta = \mathrm{diag}\,[\Delta_1, \ldots, \Delta_m] \in \mathbb{C}^{r\times c}$ *with real and complex blocks as in* (3)*. Suppose* Δ^* *solves program* (4) *globally. Then there exist a matrix* $\Delta^\# \in \mathbb{C}^{r\times c}$ *such that* $\overline{\sigma}(\Delta^\#) \le \overline{\sigma}(\Delta^*) \le 1$*, with the same structure* (3)*, but with rank one complex blocks, and which also solves* (4) *globally.*

Proof
If Δ^* solves program (4) globally, then we have $\alpha(A(\Delta^*) - \alpha^* I) = 0$, where α^* is the value of (4). Therefore the A-matrix $A(\Delta^*) - \alpha^* I$ is unstable at $\Delta^* \in \mathbf{\Delta}$. From the definition of μ in [5], this is equivalent to the existence of a frequency ω_0 such that $\det(I - M(j\omega_0)\Delta^*) = 0$, where we define $M(s) := C_q(sI - (A - \alpha^* I))^{-1}B_p + D_{pq}$. Alternatively, the singularity condition can be expressed as the existence of a vector $x \neq 0$ with $M(j\omega_0)\Delta^* x = x$. Now partition x conformable to the structure of Δ, that is,

$$x = \left[\begin{array}{cccc} x_1^T, & x_2^T, & \ldots, & x_m^T \end{array}\right]^T .$$

Then construct $\Delta^\#$ as follows. Let the real blocks of $\Delta^\#$ be those of Δ^*, and replace the complex blocks of Δ^* by the dyads:

$$\Delta_i^\# := \left\{\begin{array}{ll} y_i \frac{x_i^H}{\|x_i\|} \text{ with } y_i := \frac{\Delta_i^* x_i}{\|x_i\|}, & \text{if } x_i \neq 0 \\ 0 & \text{if } x_i = 0\,. \end{array}\right. \tag{7}$$

It is readily verified that $\Delta^* x = \Delta^\# x$, and that $\overline{\sigma}(\Delta^\#) \le \overline{\sigma}(\Delta^*) \le 1$. We thus have found a $\Delta^\#$ of appropriate structure, with rank one complex blocks, and such that $\det(I - M(j\omega_0)\Delta^\#) = 0$. We infer that $A(\Delta^\#) - \alpha^* I$ is unstable, so $\alpha(A(\Delta^\#)) \ge \alpha^*$. Since α^* is the global maximum of (4), we have $\alpha(A(\Delta^\#)) = \alpha^*$, and so $\Delta^\#$ also solves program (4) globally. Note that when Δ^* corresponds to ill-posedness of (5), then ω_0 becomes infinite. □

Remark 1. A similar result holds for worst-case performance, since (6) can be regarded as an *augmented* stability problem. This follows essentially from *The Main Loop Theorem* in [23]. Note that in the local versions of (4) and (6) optimization may of course also be restricted to rank one blocks, even though this might eliminate some local maxima.

For ease of notation we define the set S_Δ of Δ's where complex blocks have dyadic structure, that is,

- $\Delta_j := \delta_j I_{r_j}$, $\delta_i \in [-1,\ 1]$ for real uncertain parameters,
- $\Delta_j := y_j x_j^H \in \mathbb{C}^{p_j \times q_j}$ with $\|y_j\| \le 1,\ \|x_j\| \le 1$ for complex dynamic uncertainties.

Programs (4) and (6) can then, without changing values, be recast as

$$\alpha^* = \max_{\Delta\in S_\Delta} \alpha(A(\Delta)), \qquad h^* = \max_{\Delta\in S_\Delta} \|T_{wz}(\Delta)\|_\infty\,. \tag{8}$$

So far we have replaced the non-smooth constraints $\overline{\sigma}(\Delta_j) \le 1$ by vector complex ball constraints $\|y_j\| \le 1$ and $\|x_j\| \le 1$. In a second step we shall now use polar coordinates to represent y_j and x_j. We first re-parameterize as follows:

$$\Delta_j = \rho_j\,(v_j \circ e^{i\theta_j^v})(u_j \circ e^{i\theta_j^u})^H, \quad i := \sqrt{-1}\,, \tag{9}$$

where $\circ$ denotes the Hadamard element-by-element matrix product, and

$$e^{i\theta_j^v} := \left[\begin{array}{c} e^{i\theta_{j,1}^v} \\ \vdots \\ e^{i\theta_{j,p_j}^v} \end{array}\right], \qquad e^{i\theta_j^u} := \left[\begin{array}{c} e^{i\theta_{j,1}^u} \\ \vdots \\ e^{i\theta_{j,q_j}^u} \end{array}\right],$$

DOI: 10.1002/rnc

with

$$\rho_j \in [0,\,1]\,, \qquad \theta_j^v \in [0,\,2\pi]^{p_j}\,, \qquad \theta_j^u \in [0,\,2\pi]^{q_j} \tag{10}$$

and $\|v_j\| = 1$ and $\|u_j\| = 1$. The constraints on v_j and u_j are now simplified to box constraints using spherical coordinates:

$$v_j := v_j(\phi^v) := \begin{bmatrix} \cos(\phi_{j,1}^v) \\ \sin(\phi_{j,1}^v)\cos(\phi_{j,2}^v) \\ \vdots \\ \sin(\phi_{j,1}^v)\ldots\sin(\phi_{j,p_j-2}^v)\cos(\phi_{j,p_j-1}^v) \\ \sin(\phi_{j,1}^v)\ldots\sin(\phi_{j,p_j-2}^v)\sin(\phi_{j,p_j-1}^v) \end{bmatrix},$$

and similarly for $u_j := u_j(\phi_j^u)$. The constraints are now

$$\phi_j^v \in [0,\pi]^{p_j-2} \times [0,\,2\pi]\,, \qquad \phi_j^u \in [0,\pi]^{q_j-2} \times [0,\,2\pi]\,. \tag{11}$$

To summarize, we have represented the semidefinite constraint $\overline{\sigma}(\Delta_j) \le 1$ by box-constraints (10) and (11) using the non-linear change of variables

$$\Delta_j = \rho_j\,(v_j(\phi_j^v) \circ e^{i\theta_j^v})(u_j(\phi_j^u) \circ e^{i\theta_j^u})^H.$$

During the following, we shall denote the independent variables as $\mathbf{x}$. That means $\mathbf{x}_j = \delta_j$ for a real uncertain block Δ_j, and $\mathbf{x}_j = (\rho_j, \phi_j^v, \theta_j^v, \phi_j^u, \theta_j^u)$ for a complex block Δ_j. We write the box-constraints as $\mathbf{x} \in B$, where $\mathbf{x}_j = \delta_j \in [-1, 1]$ for real blocks, and where $\mathbf{x}_j$ satisfies (10) and (11) for a complex block. Altogether we have turned programs (4) and (6) into the *non-smooth* box-constrained optimization programs

$$\alpha^* = \max_{\mathbf{x}\in B} \alpha(A(\Delta(\mathbf{x}))),\; h^* = \max_{\mathbf{x}\in B} \|T_{wz}(\Delta(\mathbf{x}))\|_\infty\,, \tag{12}$$

where $\Delta(\mathbf{x})$ represents the change of variables above, and where the non-smoothness is now solely due to the non-smoothness of the functions $-\alpha$ and $-\|\cdot\|_\infty$. This latter aspect will be systematically addressed in the following sections.

4. TRUST REGION ALGORITHM

In order to solve the robust analysis problem we employ a novel trust-region algorithm suited among others for the nonsmooth criteria arising in the applications (6) and (4). For the sake of generality we consider an abstract optimization problem of the form

$$\begin{array}{ll} \text{minimize} & f(\mathbf{x}) \\ \text{subject to} & \mathbf{x} \in B \end{array} \tag{13}$$

where f is potentially non-smooth and non-convex, and where $B \subset \mathbb{R}^n$ is a simply structured closed convex set. The applications we have in mind include $f(\mathbf{x}) = -\|T_{wz}(\Delta(\mathbf{x}))\|_\infty$ and $f(\mathbf{x}) = -\alpha(A(\Delta(\mathbf{x})))$ from programs (6) and (4), where $\mathbf{x} \in B$ represents the box constraints derived in section 3. As we shall see, in these applications it is justified to further assume that the objective f is locally Lipschitz and strictly differentiable at the points $\mathbf{z}$ of a dense full measure subset of B. This property allows us to state the following trust-region algorithm (see algorithm 1), which resembles its classical alter ego in smooth optimization. Convergence, as we shall see, requires stronger hypotheses, which we shall discuss in section 5.

Motivated by the classical trust-region approach, we define the trust-region tangent program $\mathcal{T}_g(\mathbf{x}^j, R_k)$ for (13) at the current iterate $\mathbf{x}^j$ and with the current trust-region radius R_k as follows:

$$\begin{array}{ll} \text{minimize} & f(\mathbf{x}^j) + \nabla f(\mathbf{x}^j)^T(\mathbf{y} - \mathbf{x}^j) \\ \text{subject to} & \mathbf{y} \in B, \|\mathbf{y} - \mathbf{x}^j\| \le R_k \end{array} \tag{14}$$

Prepared using rncauth.cls
DOI: 10.1002/rnc

Algorithm 1. First-order trust-region method for (13)

Parameters: $0 < \gamma < \Gamma < 1$, $0 < \theta < 1$, $M > 1$.

▷ **Step 1** (`Initialize`)**.** Put outer loop counter $j = 1$, choose initial guess $\mathbf{x}^1 \in B$ such that f is strictly differentiable at $\mathbf{x}^1$, and initialize memory trust-region radius as $R_1^\sharp > 0$.

◇ **Step 2** (`Stopping`)**.** If $\mathbf{x}^j$ is a Karush-Kuhn-Tucker point of (13) then exit, otherwise go to inner loop.

▷ **Step 3** (`Initialize inner loop`)**.** Put inner loop counter $k = 1$ and initialize trust-region radius as $R_1 = R_j^\sharp$.

▷ **Step 4** (`Cauchy point`)**.** At inner loop counter k and trust region radius $R_k > 0$ compute the solution $\mathbf{y}^k$ of the tangent program $\mathcal{T}_g(\mathbf{x}^j, R_k)$ in (14).

▷ **Step 5** (`Trial step`)**.** Find trial point $\mathbf{z}^k \in B$ of strict differentiability of f such that $\|\mathbf{x}^j - \mathbf{z}^k\| \leq M\|\mathbf{x}^j - \mathbf{y}^k\|$ and

$$\nabla f(\mathbf{x}^j)^T(\mathbf{x}^j - \mathbf{z}^k) \geq \theta \nabla f(\mathbf{x}^j)^T(\mathbf{x}^j - \mathbf{y}^k).$$

▷ **Step 6** (`Acceptance`)**.** If

$$\rho_k = \frac{f(\mathbf{x}^j) - f(\mathbf{z}^j)}{\nabla f(\mathbf{x}^j)^T(\mathbf{x}^j - \mathbf{z}^k)} \geq \gamma,$$

then accept $\mathbf{z}^k$ as the next serious iterate $\mathbf{x}^{j+1} = \mathbf{z}^k$, quit inner loop and goto step 7. Otherwise reduce trust-region radius $R_{k+1} = \frac{1}{2}R_k$, increment inner loop counter k, and continue inner loop with step 4.

◇ **Step 7** (`Update trust-region`)**.** If $\rho_k \geq \Gamma$ upon acceptance of $\mathbf{z}^k$ then define memory trust-region as $R_{j+1}^\sharp = 2R_k$. Otherwise $R_{j+1}^\sharp = R_k$. Increment outer loop counter j and loop on with step 2.

Any optimal solution of (14) in step 4 of the algorithm will be denoted $\mathbf{y}^k$ and will serve as a reference point to generate trial-step. In classical trust-region methods $\mathbf{y}^k$ is sometimes called the Cauchy step.

We observe that in contrast with the classical situation our objective f is nonsmooth, so that its gradient ∇f exists only on a dense set. However, we have to assure that $\nabla f(\mathbf{x}^j)$ exists at the serious iterates $\mathbf{x}^j$ of our method. Fortunately, under the assumption that f is almost everywhere strictly differentiable on B, it is possible to arrange that the serious iterates $\mathbf{x}^j \in B$ generated by the algorithm are points of strict differentiability of f, so that the tangent program (14) is well-defined.

Since $\mathbf{y}^k \in B$ is typically *not* a point of differentiability of f, let alone of strict differentiability, we have to enlarge the set of possible trial points as follows. Fixing $M > 1$ and $0 < \theta < 1$, we accept a point of strict differentiablility $\mathbf{z}^k \in B$ of f as a trial step if

$$\|\mathbf{z}^k - \mathbf{x}^j\| \leq M\|\mathbf{y}^k - \mathbf{x}^j\|, \tag{15}$$

and if in addition the estimate

$$\nabla f(\mathbf{x}^j)^T(\mathbf{x}^j - \mathbf{z}^k) \geq \theta \nabla f(\mathbf{x}^j)^T(\mathbf{x}^j - \mathbf{y}^k) \tag{16}$$

is satisfied. Note that there exists a full neighborhood of $\mathbf{y}^k$ on which (15) and (16) are satisfied. Hence under the hypothesis that f is almost everywhere strictly differentiable on B, it is always possible to find a point of strict differentiability $\mathbf{z}^k \in B$ arbitrarily close to $\mathbf{y}^k$, where in consequence the properties (15) and (16) are satisfied.

The meaning of estimate (16) is that the model predicted progress at $\mathbf{z}^k$ is at least the θ-fraction of the model predicted progress at $\mathbf{y}^k$. Here $\mathbf{y} \mapsto f(\mathbf{x}^j) + \nabla f(\mathbf{x}^j)^T(\mathbf{y} - \mathbf{x}^j)$ serves as local first-order model of f in the neighborhood of $\mathbf{x}^j$, the latter being a point of strict differentiability of f.

DOI: 10.1002/rnc

Remark 2. Typical parameter values in algorithm 1 are $\gamma = 0.05$, $\Gamma = 0.9$, $\theta = 0.01$, $M = 2$. Instead of $R+ = \frac{1}{2}R$ one may use $R^+ = \frac{1}{4}R$ and instead of $R^+ = 2R$ other rules like $R^+ = 2.5R$.

Let us continue to explain the elements of Algorithm 1. Observe that acceptance in step 6 is based on the usual Armijo test [21]. The tangent program in step 4 reduces to an LP if a polyhedral norm is used. In the applications (4) and (6), B is a box aligned with the coordinate axes, so that the natural choice of vector norm is $\|\cdot\|_\infty$, and in that case the solution $\mathbf{y}^k$ of the tangent program can even be computed explicitly, which makes our algorithm extremely fast.

Remark 3. By Rademacher's theorem every locally Lipschitz function is almost everywhere differentiable, but algorithm 1 requires dense *strict* differentiability on the set B. While there exist pathological examples of locally Lipschitz functions which are nowhere strictly differentiable, see e.g. the lightning function of [24], all functions of practical interest have this property. Sufficient conditions for almost everywhere or dense strict differentiability are for instance semi-smoothness in the sense of [25], or essential smoothness as discussed in [26]. In particular, the functions $-\alpha$ and $-\|T_{wz}\|_\infty$ in which we are interested here have this property.

5. CONVERGENCE

In this section we discuss the convergence aspects of algorithm 1. As was already stressed, obtaining local optimality or criticality certificates is the best we can hope to achieve with reasonable effort for problems which are known to be NP-hard if solved to global optimality. From a practical point of view, our approach is satisfactory since we find the global optima in the vast majority of cases.

During the following our motivation is to present an algorithmic approach, which is as close as possible to the classical trust-region method used in smooth optimization. In doing this we a wary of the fact that in general it is *not possible* to apply smooth algorithms to nonsmooth criteria without putting convergence at stake. For instance, [32] shows that the steepest descent method may fail for a convex nonsmooth function when combined with linesearch as globalization technique, [27] shows that the same happens when trust-regions are used. Being able to use trust-regions therefore hinges on the specific structure of programs (4) and (6), where the objectives have nonsmoothness which resembles that of a concave function. As we shall see, this is type of nonsmoothness nicely captured by the class of so-called upper-C^1 functions, which we discuss in subsection 5.1.

5.1. *Convergence for upper-C^1 functions*

Algorithm 1 can be regarded as a special case of a more general bundle trust-region algorithm analyzed in [27, 28], where the locally Lipschitz function f is approximated by its so-called *standard model*

$$\phi^\sharp(\mathbf{y},\mathbf{x}) = f(\mathbf{x}) + f^\circ(\mathbf{x},\mathbf{y}-\mathbf{x}) = f(\mathbf{x}) + \max_{g\in\partial f(\mathbf{x})} g^T(\mathbf{y}-\mathbf{x}),$$

a natural extension of the first-order Taylor expansion of nonsmooth functions f. If f is strictly differentiable at $\mathbf{x}$, then the standard model $\phi^\sharp$ coincides indeed with the first-order Taylor polynomial, because in that case the Clarke subdifferential reduces to $\partial f(\mathbf{x}) = \{\nabla f(\mathbf{x})\}$. We may now see algorithm 1 as a simpler instance of the main algorithm of [27], where the standard model is used, where the working models $\phi^\sharp_k$ coincide with $\phi^\sharp$, and where in addition trial points $\mathbf{z}^k$ are chosen as points of strict differentiability of f. Convergence of the algorithm now hinges on the following

Definition 1. The standard model $\phi^\sharp$ of f is said to be *strict* at $\mathbf{x}_0$ if for every $\epsilon > 0$ there exists $\delta > 0$ such that $f(\mathbf{y}) \le \phi^\sharp(\mathbf{y},\mathbf{x}) + \epsilon\|\mathbf{y}-\mathbf{x}\|$ for all $\mathbf{x},\mathbf{y} \in B(\mathbf{x}_0,\delta)$, the ball with center x_0 and radius δ. ■

Example 1. The function $f(\mathbf{x}) = \mathbf{x}^2 \sin \mathbf{x}^{-1}$ with $f(0) = 0$ on the real line is a pathological example, which is differentiable, but not strictly differentiable at $\mathbf{x} = 0$. In consequence, its standard model $\phi^\sharp$ is not strict at 0.

DOI: 10.1002/rnc

We can further explain strictness of the standard model in the case where f is strictly differentiable on a full measure subset. In that case no reference to the standard model $\phi^\sharp$ is needed.

Lemma 2. *Suppose f is almost everywhere strictly differentiable. Then strictness of its standard model at $\mathbf{x}_0$ is equivalent to the following condition: For every $\epsilon > 0$ there exists $\delta > 0$ such that for all $\mathbf{x}, \mathbf{y} \in B(\mathbf{x}_0, \delta)$, with $\mathbf{x}$ a point of strict differentiability of f, we have*

$$f(\mathbf{y}) - f(\mathbf{x}) - \nabla f(\mathbf{x})^T(\mathbf{y} - \mathbf{x}) \leq \epsilon\|\mathbf{y} - \mathbf{x}\|. \tag{17}$$

Proof
We observe that $\phi^\sharp(\mathbf{y}, \mathbf{x}) = f(\mathbf{x}) + \nabla f(\mathbf{x})^T(\mathbf{y} - \mathbf{x})$ as soon as $\mathbf{x}$ is a point of strict differentiability of f. Therefore the condition in Definition 1 implies $f(\mathbf{y}) - f(\mathbf{x}) \leq \nabla f(\mathbf{x})^T(\mathbf{y} - \mathbf{x}) + \epsilon\|\mathbf{y} - \mathbf{x}\|$ for all $\mathbf{x} \in B(\mathbf{x}_0, \delta) \cap S_f$ and $\mathbf{y} \in B(\mathbf{x}_0, \delta)$, where S_f is the set of points of strict differentiablity of f.

Conversely, we know from [29, Thm. 2.5.1] that the Clarke directional derivative may be written as $f^\circ(\mathbf{x}, \mathbf{y}) = \limsup\{\nabla f(\mathbf{x}')^T\mathbf{y}, \mathbf{x}' \to \mathbf{x}, \mathbf{x}' \in S_f\}$, because S_f has by assumption a complement of measure zero. Now for any $\mathbf{x}' \in B(\mathbf{x}_0, \delta) \cap S_f$ and $\mathbf{y} \in B(\mathbf{x}_0, \delta)$ we have $f(\mathbf{y}) - f(\mathbf{x}') \leq \nabla f(\mathbf{x}')^T(\mathbf{y} - \mathbf{x}') + \epsilon\|y - \mathbf{x}'\|$, hence by taking the limit superior $\mathbf{x}' \to \mathbf{x}$, we obtain $f(\mathbf{y}) - f(\mathbf{x}) \leq f^\circ(\mathbf{x}, \mathbf{y}) + \epsilon\|\mathbf{y} - \mathbf{x}\|$. Here we use local boundedness [29] of the Clarke subdifferential, which implies boundedness of $\nabla f(\mathbf{x}')$ as $\mathbf{x}' \to \mathbf{x}$. □

Definition 2 (Spingarn [30])**.** A locally Lipschitz function $f : \mathbb{R}^n \to \mathbb{R}$ is lower-C^1 at $\mathbf{x}_0 \in \mathbb{R}^n$ if there exist a compact space $\mathbb{K}$, a neighborhood U of $\mathbf{x}_0$, and a function $F : \mathbb{R}^n \times \mathbb{K} \to \mathbb{R}$ such that

$$f(\mathbf{x}) = \max_{\mathbf{y} \in \mathbb{K}} F(\mathbf{x}, \mathbf{y}) \tag{18}$$

for all $\mathbf{x} \in U$, and F and $\partial F/\partial \mathbf{x}$ are jointly continuous. The function f is said to be upper-C^1 if $-f$ is lower-C^1. ■

Lemma 3. *Suppose the locally Lipschitz function f is upper-C^1, then it is almost everywhere strictly differentiable.*

Proof
This follows with Borwein and Moors [26] if we observe that an upper-C^1 function is semi-smooth. □

This shows that our algorithm is applicable to upper-C^1 functions. In fact, since $B \setminus S_f$ is of measure zero, an arbitrarily small random perturbation of the solution $\mathbf{y}^k$ of the tangent program will with probability 1 provide a trial point $\mathbf{z}^k \in B \cap S_f$ satisfying (15) and (16). The crucial observation is now that upper-C^1 functions behave favorably in a minimization method, as for this class iterates $\mathbf{x}^j$ are moving away from the non-smoothness. It is therefore of interest to note the following

Lemma 4. *Consider the stability set $\mathbb{D} = \{\mathbf{x} : T_{zw}(\Delta(\mathbf{x}))$ is internally stable$\}$. Then $\mathbf{x} \mapsto \|T_{zw}(\Delta(\mathbf{x}))\|_\infty$ is locally Lipschitz and lower-C^1 on $\mathbb{D}$, so that $f : \mathbf{x} \mapsto -\|T_{zw}(\Delta(\mathbf{x}))\|_\infty$ is upper-C^1 on $\mathbb{D}$.* ■

For the proof we refer to [13]. For the spectral abscissa the situation is more complicated, as α may in general even fail to be locally Lipschitz [15]. The following was proved in [13].

Lemma 5. *Suppose all active eigenvalues at $\mathbf{x}_0$ are semi-simple, then $f(\mathbf{x}) = -\alpha(A(\Delta(\mathbf{x})))$ is locally Lipschitz in a neighborhood of $\mathbf{x}_0$. If in addition all active eigenvalues are simple, then f is upper-C^1 at $\mathbf{x}_0$.* ■

The interest in the upper-C^1 property is due to the following fact, a proof of which can be found e.g. in [31, 32, 33].

Proposition 1. *Suppose f is locally Lipschitz and upper-C^1 in a neighborhood of $\mathbf{x}$. Then its standard model $\phi^\sharp$ is strict at $\mathbf{x}$.* ■

DOI: 10.1002/rnc

The consequence is the following

Theorem 1. *The sequence $\mathbf{x}^j \in B \cap S_f$ of iterates generated by algorithm 1 for program (6) converges to a unique KKT-point.*

Proof
As a consequence of Lemma 4 and the main theorem in [27] every accumulation point $\mathbf{x}^*$ of the sequence $\mathbf{x}^j$ is a KKT-point. Convergence to a single critical point $\mathbf{x}^*$ then follows from the Łojasiewicz property of $f(\mathbf{x}) = -\|T_{wz}(\Delta(\mathbf{x}))\|_\infty$, which was established in [27, Theorem 1], and which hinges on the analytical dependence of T_{wz} on $\mathbf{x}$. □

While this represents an ironclad convergence certificate for program (6), the situation for the spectral abscissa (4) is more complicated. We have the following weaker result.

Theorem 2. *Let $\mathbf{x}^j \in B \cap S_f$ be the sequence of iterates generated by algorithm 1 for the minimization of $f(\mathbf{x}) = -\alpha(A(\Delta(\mathbf{x})))$ over B. Suppose that for at least one accumulation point $\mathbf{x}^*$ of the $\mathbf{x}^j$ all active eigenvalues of $A(\Delta(\mathbf{x}^*))$ are simple. Then the sequence converges in fact to this accumulation point, which in addition is then a KKT-point.*

Proof
By hypothesis there exists at least one accumulation point $\mathbf{x}^*$ of the $\mathbf{x}^j$ in which every active eigenvalue is simple. By Lemma 5, f is locally Lipschitz and upper-C^1 at $\mathbf{x}^*$, and now application of the result of [27] shows that $\mathbf{x}^*$ is a critical point of (4). *A priori* the method of proof of [27] does *not* imply criticality of the remaining accumulation points $\tilde{\mathbf{x}}$ of the sequence $\mathbf{x}^j$, but in a second stage we now argue that f has the Łojasiewicz property [27], and that gives convergence of the $\mathbf{x}^j$ to a single point, which must be $\mathbf{x}^*$, and which is therefore a KKT-point. □

5.2. *Approximate standard model*

The convergence result for (4) is not as satisfactory as the result obtained for (6) for two reasons. Firstly, as observed in [15] already, α is not even guaranteed to be locally Lipschitz everywhere. Typical examples where this fails are when a derogatory eigenvalue is active. Secondly, local Lipschitz behavior of $f(\mathbf{x}) = -\alpha(A(\Delta(\mathbf{x})))$ is guaranteed when all active eigenvalues at $\mathbf{x}$ are semi-simple, but the upper-C^1 property needs the stronger hypotheses of Proposition 4, and one would at least hope for a convergence result in the case where all active eigenvalues are semi-simple. All this is in strong contrast with what is observed in practice, where $f = -\alpha$ behaves consistently like an upper-C^1 function. In order to explain this somewhat better by theoretical results, we shall in the following outline an approximate convergence result, which works under a weaker assumption than in Theorem 2

We say that f is ϵ-concave at $\mathbf{x}_0$ if there exists a neighborhood $B(\mathbf{x}_0, \delta)$ of $\mathbf{x}_0$ such that

$$f(\mathbf{y}) \le f(\mathbf{x}) + f^\circ(\mathbf{x}, \mathbf{y} - \mathbf{x}) + \epsilon\|\mathbf{y} - \mathbf{x}\|$$

for all $\mathbf{x}, \mathbf{y} \in B(\mathbf{x}_0, \delta)$. Note that ϵ-concavity for every $\epsilon > 0$ is the same as the upper-C^1 property, but here we fix ϵ, so that a much weaker condition is obtained. Now we have the following

Theorem 3. *Suppose at least one of the accumulation points $\mathbf{x}^*$ of the sequence $\mathbf{x}^j$ of iterates generated by algorithm 1 is an ϵ-concave point for f. Then the entire sequence $\mathbf{x}^j$ converges in fact to $\mathbf{x}^*$, and $\mathbf{x}^*$ is approximately optimal in the sense that $\min\{\|g\| : g \in \partial f(\mathbf{x}^*) + N_B(\mathbf{x}^*)\} \le \epsilon'$, where $N_B(\mathbf{x}^*)$ is the normal cone to B at $\mathbf{x}^* \in B$. Here ϵ' depends in the following way on ϵ: There exists a constant $\sigma > 0$, which depends only on the trust-region norm $\|\cdot\|$, such that*

$$\epsilon' = \frac{M\epsilon}{\sigma\theta(1-\gamma)}, \tag{19}$$

where θ, M, γ are the parameters used in algorithm 1. In particular, if $\|\cdot\| = |\cdot|$ is the Euclidean norm, then $\sigma = 1$.

DOI: 10.1002/rnc

Proof
We follow the proof of [27, Theorem 1], but specialize the model ϕ to the standard model $\phi^\natural(\cdot,\mathbf{x}) = f(\mathbf{x}) + f^\circ(\mathbf{x},\cdot-\mathbf{x})$. Using the fact that f is almost everywhere strictly differentiable, we iterate on trial points $\mathbf{z}^k$ of strict differentiablity of f, which means that all serious iterates $\mathbf{x}^j$ are also points of strict differentiability of f. This reduces the algorithm in [27] to our present algorithm 1, where the test quotient $\widetilde{\rho}_k$ required in [27] becomes redundant because it automatically equals 1. The difference with [27] is that ϵ_k in Lemma 3 and ϵ_j in part 5) of the proof of Theorem 1 of [27] are now held fixed as $\epsilon_k = \epsilon$ and $\epsilon_j = \epsilon$. Since $\widetilde{\rho}_k = 1$ and $\widetilde{\rho}_{k_j-\nu_j} = 1$, while $\rho_k < \gamma$, one concludes for the η arising in that proof that $\eta = \|\nabla f(\mathbf{x}^j)\| \le \epsilon/(\sigma\theta M^{-1}(1-\gamma))$ in Lemma 3, and similarly in the proof of Theorem 1 of [27]. Here θ, M, γ are the constants used in algorithm 1, while σ is found in [27, Lemma 1] and depends only on the trust-region norm. Note that convergence to a single limit point $\mathbf{x}^*$ follows again from the Łojasiewicz property of $f = -\alpha$, for which we refer to [13]. □

Remark 4. The result is even quantitative and should be understood in the following sense. Suppose the user applies algorithm 1 to (4) under the weaker hypothesis of ϵ-concavity of $f = -\alpha$, where the $\epsilon > 0$ remains unknown to the user. Since ϵ cannot be made arbitrarily small, a systematic error remains, but by way of formula (19), users will know in the end that they converged to an ϵ'-optimal solution $\mathbf{x}^*$, where ϵ' is of the same order as the inevitable error ϵ. Indeed, for the euclidean norm $\sigma = 1$, so we can arrange $\epsilon' \approx \Theta\epsilon$, where $\Theta > 1$, but $\Theta \approx 1$. It suffices to choose $M = 1$, $0 \ll \theta < 1$, $0 < \gamma \ll 1$. This further corroborates our approach chosen in algorithm 1 for (4).

6. STOPPING CRITERIA AND COMPUTING SUBGRADIENTS

6.1. *Stopping criteria*

Stopping the algorithm based on a rigorous convergence result can be organized as follows. If the trust-region management finds a new serious iterate $\mathbf{x}^{j+1}$ such that

$$\frac{\|\mathbf{x}^{j+1}-\mathbf{x}^j\|}{1+\|\mathbf{x}^j\|} < \mathrm{tol}_1, \quad \frac{\|P_B(-\nabla f(\mathbf{x}^{j+1}))\|}{1+|f(\mathbf{x}^{j+1})|} < \mathrm{tol}_2,$$

where P_B is the orthogonal projection on B, then we decide that $\mathbf{x}^{j+1}$ is optimal. On the other hand, if the inner loop has difficulties finding a new serious iterate and if 5 consecutive trial steps $\mathbf{z}^k$ with

$$\frac{\|\mathbf{z}^{k}-\mathbf{x}^j\|}{1+\|\mathbf{x}^j\|} < \mathrm{tol}_1, \quad \frac{\|P_B(-\nabla f(\mathbf{x}^{j}))\|}{1+|f(\mathbf{x}^{j})|} < \mathrm{tol}_2$$

occur, then we decide that $\mathbf{x}^j$ was already optimal.

6.2. *Computing subgradients*

Computing subgradients is a key element in our approach, since the use of automatic differentiation is not suited due nonsmoothness, and since the criteria are computed iteratively. Subgradients also provide important information on the problem structure, and they underscore potential difficulties such as semi-infiniteness, non-smoothness, etc. Subgradient information is also a central ingredient of the solver, as our technique is of trust-region type and will solve a tangent problem built from subdifferential information at every iteration. Since subgradients are used repeatedly in the solver, it is essential to establish efficient formulas. Finally, the subgradient set or Clarke's subdifferential is the only means to certify the computed value is a locally optimal solution, based on the criteria of section 6.1.

Subgradients of $\alpha(A(\Delta(\mathbf{x})))$ and $\|T_{wz}(\Delta(\mathbf{x}))\|_\infty$ with respect to $\mathbf{x}$ in (12) are now derived using chain rules [29, 13, 16], while subgradients of $-\alpha$ and $-\|T_{wz}\|_\infty$ are readily derived using the general rule $\partial(-f)(\mathbf{x}) = -\partial f(\mathbf{x})$, see [29].

DOI: 10.1002/rnc

By virtue of the block-diagonal structure of Δ, it is enough to consider each block separately. The whole subgradient is then obtained by piecing the blockwise subgradients together. As subgradients with respect to real parametric blocks $\mathbf{x}_j = \delta_j$ have already been described in [13], we focus on complex blocks. For simplicity we suppress the block index j during the following, so that $\Delta(\mathbf{x})$ becomes a mapping $\Delta(\rho, \phi^v, \theta^v, \phi^u, \theta^u)$ from $[0,\,1] \times [0, \pi]^{p-2} \times [0, 2\pi] \times [0,\,2\pi]^p \times [0, \pi]^{q-2} \times [0, 2\pi] \times [0,\,2\pi]^q$ to $\mathbb{C}^{p\times q}$. From the expression (9), derivatives of Δ with respect to ρ, v, θ^v, u and θ^u are obtained as

$$\begin{array}{rcl} \Delta'_\rho d\rho & = & (v \circ e^{i\theta^v})(u \circ e^{i\theta^u})^H d\rho \\ \Delta'_v dv & = & \rho\,(dv \circ e^{i\theta^v})(u \circ e^{i\theta^u})^H \\ \Delta'_{\theta^v} d\theta^v & = & i\rho\,(v \circ e^{i\theta^v} \circ d\theta^v)(u \circ e^{i\theta^u})^H \\ \Delta'_u du & = & \rho\,(v \circ e^{i\theta^v})(du \circ e^{i\theta^u})^H \\ \Delta'_{\theta^u} d\theta^u & = & -i\rho\,(v \circ e^{i\theta^v})(u \circ e^{i\theta^u} \circ d\theta^u)^H\,. \end{array} \tag{20}$$

Introducing the Jacobians J^v and J^u of $v(\phi^v)$ and $u(\phi^u)$ as mappings $[0, \pi]^{p-2} \times [0,\,2\pi] \to \mathbb{R}^p$ and $[0, \pi]^{q-2} \times [0,\,2\pi] \to \mathbb{R}^q$, respectively, we have

$$\begin{array}{rcl} \Delta'_{\phi^v} d\phi^v & = & \rho\,((J^v d\phi^v) \circ e^{i\theta^v})(u \circ e^{i\theta^u})^H \\ \Delta'_{\phi^u} d\phi^u & = & \rho\,(v \circ e^{i\theta^v})((J^u d\phi^u) \circ e^{i\theta^u})^H\,, \end{array} \tag{21}$$

which gives all the partial derivatives $\partial\Delta/\partial\mathbf{x}$. Using (5), this gives the derivatives of $A(\Delta(\mathbf{x}))$ with respect to $\mathbf{x}$, and similarly, of the transfer function $T_{wz}(\Delta(\mathbf{x}))$ with respect to $\mathbf{x}$.

In order to proceed, we now have to discuss subgradients of α with respect to A, and of the H_∞-norm with respect to the transfer function as an argument, and this is where non-smoothness re-enters the scene. We have the following useful

Definition 3. An eigenvalue $\lambda_l(A(\Delta))$ is active at $\Delta \in S_\Delta$ if $\alpha(A(\Delta)) = \mathrm{Re}\lambda_l(A(\Delta))$. A frequency $\omega_0 \in [0, \infty]$ is active at $\Delta \in S_\Delta$ if $\|T_{wz}(\Delta)\|_\infty = \overline{\sigma}(T_{wz}(\Delta, i\omega_0))$. ■

During the following we exploit the fact that our algorithm does not require computing the entire subdifferential at an iterate $\mathbf{x}$. It is sufficient to compute just one subgradient, and that can be achieved by picking active elements. Moreover, the computation simplifies for points $\mathbf{x}^j$ os strict differentiability of f.

Suppose the eigenvalue $\lambda_l(\Delta)$ is active, that is, $\alpha(A(\Delta)) = \mathrm{Re}\lambda_l$. Following [27], we introduce column matrices V_l and U_l of right and left eigenvectors of $A(\Delta)$ associated with the eigenvalue λ_l, such that $U_l^H V_l = I$. A subgradient G_Δ of $\alpha(\cdot)$ at Δ with respect to Δ as a free matrix variable and with regard to the scalar product $\langle G, H\rangle = \mathrm{Tr}(GH^T)$ on matrix space, is then obtained as

$$G_\Delta = \mathrm{Re}\,\Psi(Y_l)$$

with

$$\Psi(Y_l) := (I - D_{\delta\delta}\Delta)^{-1} C_\delta V_l Y_l U_l^H B_\delta (I - \Delta D_{\delta\delta})^{-1}\,, \tag{22}$$

where Y_l is an arbitrary Hermitian matrix such that $Y_l \succeq 0$, $\mathrm{Tr}\,Y_l = 1$, and with size the multiplicity of λ_l. The corresponding first-order term $\langle G_\Delta, d\Delta\rangle$ is

$$\langle G_\Delta, d\Delta\rangle = \mathrm{ReTr}\left(\Psi(Y_l)\; d\Delta^T\right). \tag{23}$$

Similarly, let ω_0 be a peak frequency of $\|T_{wz}(\Delta)\|_\infty$, so that $\|T_{wz}(\Delta)\|_\infty = \overline{\sigma}(T_{wz}(\Delta, i\omega_0))$. Introduce column matrices U_{ω_0} and V_{ω_0} of left and right singular vectors associated with $\overline{\sigma}(T_{wz}(\Delta, i\omega_0))$ from the SVD and define the transfer functions

$$\begin{bmatrix} * & T_{z_\delta w}(\Delta) \\ T_{zw_\delta}(\Delta) & T_{zw}(\Delta) \end{bmatrix} := \begin{bmatrix} 0 & I \\ I & \Delta \end{bmatrix} \star P\,,$$

where $\star$ stands for the Redheffer product [34]. Then a subgradient G_Δ of $\|T_{wz}(\Delta)\|_\infty$ at Δ with respect to Δ as a free matrix variable is given as

$$G_\Delta = \mathrm{Re}\Psi(Y_{\omega_0}),$$

Prepared using rncauth.cls
DOI: 10.1002/rnc

with

$$\Psi(Y_{\omega_0}) := T_{z_\delta w}(\Delta, j\omega_0) V_{\omega_0} Y_{\omega_0} U_{\omega_0}^H T_{z w_\delta}(\Delta, j\omega_0)\,, \tag{24}$$

where Y_{ω_0} is an arbitrary Hermitian matrix such that $Y_{\omega_0} \succeq 0$, $\mathrm{Tr}\, Y_{\omega_0} = 1$, and with dimension the multiplicity of $\overline{\sigma}(T_{wz}(\Delta, i\omega_0))$. Here the first-order term is

$$\langle G_\Delta, d\Delta\rangle = \mathrm{Re}\mathrm{Tr}\left(\Psi(Y_{\omega_0})\, d\Delta^T\right). \tag{25}$$

Subgradients of $\alpha(A(\Delta(\mathbf{x})))$ and $\|T_{wz}(\Delta(\mathbf{x}))\|_\infty$ with respect to $\mathbf{x} = (\rho, \phi^v, \theta^v, \phi^u, \theta^u)$ are then obtained by explicitly calculating (23) and (25), where partial derivatives of Δ with respect to $\mathbf{x}$ are given in (20) and (21). For the functions $\alpha(\cdot)$ or $\|T_{wz}(\cdot)\|_\infty$, subgradients $g_{\mathbf{x}}^T$ at Δ are obtained as

$$\begin{array}{ll}
g_\rho^T d\rho & = \mathrm{Re}\{(u \circ e^{i\theta^u})^H \Psi(Y)\,(v \circ e^{i\theta^v})\}\, d\rho \\
g_{\phi^v}^T d\phi^v & = \rho\, \mathrm{Re}\{(u \circ e^{i\theta^u})^H \Psi(Y)\, \mathrm{diag}(e^{i\theta^v})\} J^v\, d\phi^v \\
g_{\theta^v}^T d\theta^v & = -\rho\, \mathrm{Im}\{(u \circ e^{i\theta^u})^H \Psi(Y)\, \mathrm{diag}(v \circ e^{i\theta^v})\}\, d\theta^v \\
g_{\phi^u}^T d\phi^u & = \rho\, \mathrm{Re}\{(v \circ e^{i\theta^v})^T \Psi(Y)^T\, \mathrm{diag}(e^{-i\theta^u})\} J^u\, d\phi^u \\
g_{\theta^u}^T d\theta^u & = \rho\, \mathrm{Im}\{(v \circ e^{i\theta^v})^T \Psi(Y)^T \mathrm{diag}(u \circ e^{-i\theta^u})\}\, d\theta^u\,.
\end{array} \tag{26}$$

Here the operator $\mathrm{diag}(\cdot)$ applied to a vector a builds a diagonal matrix with a on the main diagonal, and $\Psi(Y)$ stands short for either $\Psi(Y_l)$ or $\Psi(Y_{\omega_0})$. The whole subdifferential is generated by varying Y_l or Y_{ω_0} over the spectraplex

$$\{Y = Y^H : Y \succeq 0,\ \mathrm{Tr}\, Y = 1\}\,. \tag{27}$$

Remark 5. Note that subgradients (26) substantially simplify for scalar, single-input and/or single-output complex blocks:

$$\rho e^{i\theta}, \quad \rho v(\phi^v) e^{i\theta^v}, \quad (u(\phi^u) e^{i\theta^u})^H\,.$$

Simplifications also occur when the active eigenvalue or the maximum singular value at the peak frequency $\overline{\sigma}(T_{wz}(\Delta, i\omega_0))$ is simple. In that case we may choose $Y = 1$ to obtain the usual gradient for differentiable functions.

Remark 6. The readers will also easily convince themselves that for real active eigenvalues or for active frequencies $\omega = 0, \infty$ in the H_∞-norm the expression $P(i\omega)$ is real, and in that case it suffices to search over real blocks Δ. Then the complex blocks may be reduced to

$$\Delta_j = \rho_j\, v_j(\phi_j^v) u_j(\phi_j^u)^T\,.$$

In contrast to what is sometimes tacitely assumed in the literature, the occurrence of multiple active frequencies at the solution $\mathbf{y}^k$ of the tangent program is not unusual. This typically happens when $\|T_{wz}(\Delta)\|_\infty$ results from a robust control design scheme. We state the general case as

Proposition 2. *Suppose $\alpha(A(\Delta(\mathbf{x})))$, or $\|T_{wz}(\Delta(\mathbf{x}))\|_\infty$, is attained at N active semi–simple eigenvalues $l = 1, \ldots, N$, or at N active frequencies $\omega_1, \ldots, \omega_N$, which means $\alpha(A(\Delta(\mathbf{x}))) = \mathrm{Re}\lambda_l$, or $\|T_{wz}(\Delta(\mathbf{x}))\|_\infty = \overline{\sigma}(T_{wz}(\Delta, i\omega_l))$ for $l = 1, \ldots, N$. For $\alpha(A(\Delta(\mathbf{x})))$ define column matrices U_l and V_l of left and right eigenvectors such that $U_l^H V_l = I$. Alternatively, for $\|T_{wz}(\Delta(\mathbf{x}))\|_\infty$ define column matrices of left and right singular vectors U_l and V_l associated with $\overline{\sigma}(T_{wz}(\Delta(\mathbf{x}), i\omega_l))$ from the SVD. Then Clarke subgradients $g_{\mathbf{x}}^T$ of $\alpha(.)$ or $\|T_{wz}(.)\|_\infty$ with respect to $\mathbf{x} = (\rho, \phi^v, \theta^v, \phi^u, \theta^u)$ at $\Delta = \Delta(\mathbf{x})$ are obtained as*

$$\begin{array}{ll}
g_\rho^T d\rho & = \mathrm{Re}\{(u \circ e^{i\theta^u})^H \Omega(Y)\,(v \circ e^{i\theta^v})\} d\rho \\
g_{\phi^v}^T d\phi^v & = \rho\, \mathrm{Re}\{(u \circ e^{i\theta^u})^H \Omega(Y)\, \mathrm{diag}(e^{i\theta^v})\} J^v d\phi^v \\
g_{\theta^v}^T d\theta^v & = -\rho\, \mathrm{Im}\{(u \circ e^{i\theta^u})^H \Omega(Y)\, \mathrm{diag}(v \circ e^{i\theta^v})\} d\theta^v \\
g_{\phi^u}^T d\phi^u & = \rho\, \mathrm{Re}\{(v \circ e^{i\theta^v})^T \Omega(Y)^T\, \mathrm{diag}(e^{-i\theta^u})\} J^u d\phi^u \\
g_{\theta^u}^T d\theta^u & = \rho\, \mathrm{Im}\{(v \circ e^{i\theta^v})^T \Omega(Y)^T \mathrm{diag}(u \circ e^{-i\theta^u})\} d\theta^u
\end{array} \tag{28}$$

DOI: 10.1002/rnc

with the definition

$$\Omega(Y) := \sum_{l=1}^{N} \Psi(Y_l)\,,$$

where $\Psi(Y_l)$ *is defined in* (22) *or* (24)*, and with* $Y := (Y_1, \ldots, Y_N)$ *an* N*-tuple of Hermitian matrices of appropriate sizes ranging over the set*

$$\mathcal{Y} = \{(Y_1, \ldots, Y_N) : Y_l^H = Y_l,\ Y_l \succeq 0,\ \sum_{l=1}^{N} \operatorname{Tr} Y_l = 1\}.$$

Proof
We use the fact [29] that the entire Clarke subdifferential is obtained as the convex hull of the subdifferentials of all active branches considered separately. Hence the set $\mathcal{Y}$ in the proposition is obtained as the convex hull of spectraplexes in (27). For the H_∞-norm, we use the fact that either there is a finite set of active frequencies, or the system is all-pass. In the first case (28) gives a full characterization of the Clarke subdifferential, in the second case it provides a finitely generated subset of the subdifferential. □

Computation of the Jacobians J^v and J^u in (28) can be based on a column-oriented algorithm. This is readily inferred from the identity

$$v(\phi^v) = \begin{bmatrix} \cos(\phi_1^v) \\ \sin(\phi_1^v) \\ \sin(\phi_1^v) \\ \vdots \end{bmatrix} \circ \begin{bmatrix} 1 \\ \cos(\phi_2^v) \\ \sin(\phi_2^v) \\ \vdots \end{bmatrix} \circ \begin{bmatrix} 1 \\ 1 \\ \cos(\phi_3^v) \\ \vdots \end{bmatrix} \circ \ldots$$

The product rule for derivatives immediately yields the Jacobians J^v and J^u.

7. NUMERICAL TESTING

Our numerical assessment of the proposed technique is for worst-case H_∞ performance. A variety of problems from different engineering fields are shown in Table I. The characteristics of each test case are the system order n and the *mixed* uncertainty structure of Δ. Real parametric blocks are encoded by negative integers, $-3 = -3^1$ stands for $\delta_j I_3$ and -5^3 refers to 3 real blocks with repetition of order 5, that is $\operatorname{diag}(\delta_1 \mathrm{I}_5, \delta_2 \mathrm{I}_5, \delta_3 \mathrm{I}_5)$. Complex full block encoding follows the same convention when they are square, that is, 7^2 specifies $\operatorname{diag}(\Delta_1, \Delta_2)$, where Δ_1 and Δ_2 are both in $\mathbb{C}^{7\times 7}$. Non-square complex blocks are described by their row and column dimensions, i.e., 6×2 refers to $\Delta_j \in \mathbb{C}^{6\times 2}$. The values achieved by algorithm 1 are shown as h^* in column 3 of table II, computed in t^* seconds CPU given in column 5.

These results are compared to those obtained using the lower-bound of routine `WCGAIN` from [35]. Note that `WCGAIN` uses power iteration in tandem with a line search to compute the lower bound. The achieved values and execution times are given in columns $\underline{h}$ and $\underline{t}$, respectively. The symbol 'Inf' in the table means instability has been detected over $\overline{\sigma}(\Delta) \le 1$.

Our testing indicates that both techniques deliver very consistent results in the mixed case, except in test 41, where `WCGAIN` does not detect the instability. Test cases where instability is detected through $h^* = \infty$ obviously correspond to global solutions of program (12).

Computing upper bounds $\overline{h}$ using `WCGAIN` reveals that the results are generally tight. In other words, the computed worst-case uncertainties are global solutions to program (6), except in test cases 9 and 19, where the gap between lower and upper bounds is too large to conclude. Note that this does not necessarily indicate a failure of our method, as the gap may be attributed *either* to conservatism of the upper bound, *or* to failure of any of the lower bounds to reach a global solution. The good agreement of both solvers is somewhat in contrast with our previous analysis in [28], where `WCGAIN` turned out more fragile for sole real parametric uncertainties. In that case our local technique proved more reliable, and certification based on `WCGAIN` was then no longer possible due

Prepared using rncauth.cls
DOI: 10.1002/rnc

to the discrepancy. In the present study our technique is generally faster that WCGAIN, except in a few test cases like 17, 21, 28, 31 and 34. As expected, WCGAIN turns out an excellent technique for pure complex problems like in test cases 15 and 17, but may suffer when real parametric uncertainty or large Δ's are present, as witnessed by 24, 26 and for the four-disk and missile examples.

Table I. Benchmark problems for worst-case H_∞ performance.

♯	**Benchmark**	n	**Structure**
1	Beam1	11	$-1, 1, 5$
2	Beam2	11	$-4, 3$
3	Beam3	11	$-4, 1, 2$
4	Beam4	11	1^7
5	Beam5	11	-1^7
6	DC motor1	7	5
7	DC motor2	7	$-4, 1$
8	DC motor3	7	$-2, 1, 2$
9	DC motor4	7	1^5
10	DC motor5	7	-1^5
11	DVD driver1	10	$5, -3, -6$
12	DVD driver1	10	$-4, 1^3, 2^2, 1 \times 2, 2 \times 1$
13	DVD driver1	10	$-4, 3 \times 2, 2 \times 3, 3 \times 1, 1 \times 3, 1$
14	DVD driver1	10	7^2
15	Dash pot1	17	1^6
16	Dash pot2	17	$-4, 1^2$
17	Dash pot3	17	$3 \times 2, 2 \times 3, 1$
18	Four-disk system1	16	$-1, -3^5, -1^4$
19	Four-disk system2	16	$1, 3^5, -4$
20	Four-disk system3	16	$-10, 10$
21	Four-tank system1	12	1^4
22	Four-tank system2	12	$2, -2$
23	Four-tank system3	12	$-3, 1$
24	Hard disk driver1	22	$-3, 2^4, -1^4$
25	Hard disk driver2	22	$-1^3, -2^4, -1^4$
26	Hard disk driver3	22	$3, 4, -8$
27	Hydraulic servo1	9	-1^8
28	Hydraulic servo2	9	1^8
29	Hydraulic servo2	9	-8
30	Hydraulic servo2	9	$-4, 2^2$
31	Mass-spring1	8	1^2
32	Mass-spring2	8	-1^2
33	Mass-spring2	8	2
34	Filter1	8	1
35	Filter2	8	-1
36	Satellite1	11	$1, -6, 1$
37	Satellite2	11	$-1, -6, -1$
38	Satellite3	11	$2 \times 3, 3 \times 2, -3$
39	Missile1	35	$-1^3, -6^3$
40	Missile2	35	$1, 2, -6^3$
41	Missile3	35	$3, 6, -6^2$
42	Missile4	35	$3, -18$
43	Missile5	35	$-10, 11$

DOI: 10.1002/rnc

Table II. Results for worst-case H_∞-norm on $\mathbf{\Delta}$
running times $\underline{t}$ and t^* in seconds.

♯	$\underline{h}$	h^*	$\overline{h}$	$\underline{t}$	t^*
1	Inf	Inf	Inf	19.869	1.798
2	Inf	Inf	Inf	15.984	1.670
3	Inf	Inf	Inf	16.979	1.101
4	0.062	0.062	0.062	11.395	11.030
5	0.060	0.060	0.060	9.748	0.706
6	Inf	Inf	Inf	5.930	1.016
7	0.836	0.839	Inf	16.453	2.573
8	1.582	1.583	Inf	17.810	8.961
9	0.949	0.951	163515.72	24.981	9.610
10	0.818	0.819	0.818	7.442	0.665
11	Inf	Inf	Inf	59.624	0.521
12	Inf	Inf	Inf	33.545	0.575
13	Inf	Inf	Inf	31.010	1.746
14	Inf	Inf	Inf	12.177	0.626
15	0.299	0.301	0.299	9.464	10.197
16	0.282	0.282	0.282	19.172	4.753
17	0.301	0.301	0.301	7.863	15.592
18	0.071	0.072	0.071	204.361	0.982
19	0.178	0.177	Inf	115.389	63.185
20	Inf	Inf	Inf	440.717	3.219
21	0.532	0.532	0.532	5.212	10.043
22	0.532	0.532	0.532	10.613	3.319
23	0.529	0.530	0.529	12.806	3.143
24	0.003	0.003	0.003	75.132	12.053
25	0.003	0.003	0.003	92.369	0.894
26	Inf	Inf	Inf	175.733	1.218
27	0.053	0.053	0.053	5.530	0.864
28	0.055	0.055	0.055	7.958	12.136
29	0.053	0.053	0.053	31.387	0.863
30	0.055	0.055	0.055	18.113	5.814
31	16.674	16.741	16.676	2.256	4.795
32	0.684	0.684	0.684	1.639	0.498
33	Inf	Inf	Inf	1.395	0.773
34	0.247	0.248	0.247	3.525	6.679
35	0.241	0.242	0.241	2.231	0.753
36	0.015	0.015	0.015	43.517	6.409
37	0.015	0.015	0.015	39.991	1.100
38	Inf	Inf	Inf	10.879	1.512
39	0.173	0.173	0.173	1173.349	1.208
40	Inf	Inf	Inf	1067.826	0.674
41	2738.454	Inf	Inf	682.161	1.820
42	Inf	Inf	Inf	5497.893	0.595
43	Inf	Inf	Inf	961.395	1.344

8. CONCLUSION

We have presented a non-smooth optimization algorithm to compute local solutions for two NP-hard problems in stability and performance analysis of systems with mixed real and complex parametric

DOI: 10.1002/rnc

uncertainty. The local solver exploits subgradient information of the criteria, uses a novel nonsmooth trust-region technique to generate a sequence of iterates, which converges to a critical point from an arbitrary starting point, and performs fast and reliably on the given test set. The test bench features systems with up to 35 states, with up to 11 real or complex uncertainties, and with up to 18 repetitions. The results were certified by comparison with the function `WCGAIN` of [35].

REFERENCES

1. Poljak S, Rohn J. Checking robust nonsingularity is NP-complete. *Mathematics of Control, Signals, and Systems* 1994; **6**(1):1–9.
2. Braatz RD, Young PM, Doyle JC, Morari M. Computational complexity of μ calculation. *IEEE Trans. Aut. Control* 1994; **39**:1000–1002.
3. Halton M, Hayes MJ, Iordanov P. State-space μ analysis for an experimental drive-by-wire vehicle. *International Journal of Robust and Nonlinear Control* 2008; **18**(9):975–992, doi:10.1002/rnc.1322. URL http://dx.doi.org/10.1002/rnc.1322.
4. Young P, Doyle J. A lower bound for the mixed μ problem. *Automatic Control, IEEE Transactions on* Jan 1997; **42**(1):123–128, doi:10.1109/9.553696.
5. Packard A, Doyle J. The complex structured singular value. *Automatica* 1993; **29**(1):71–109.
6. Seiler P, Packard A, Balas GJ. A gain-based lower bound algorithm for real and mixed μ problems. *Automatica* Mar 2010; **46**(3):493–500, doi:10.1016/j.automatica.2009.12.008. URL http://dx.doi.org/10.1016/j.automatica.2009.12.008.
7. Newlin M, Glavaski S. Advances in the computation of the μ lower bound. *American Control Conference, 1995. Proceedings of the*, vol. 1, 1995; 442 –446 vol.1, doi:10.1109/ACC.1995.529286.
8. Packard A, Pandey P. Continuity properties of the real/complex structured singular value. *IEEE Trans. Aut. Control* Mar 1993; **AC-38**(3).
9. Magni J, Doll C, Chiappa C, Frappard B, Girouart B. Mixed μ-analysis for flexible systems. part 1: theory. *Proc. 14th IFAC World Congress on Automatic Control* 1999; :325–360.
10. Packard A, Balas G, Liu R, Shin JY. Results on worst-case performance assessment. *American Control Conference, 2000. Proceedings of the 2000*, vol. 4, 2000; 2425 –2427 vol.4, doi:10.1109/ACC.2000.878616.
11. Rodrigo RL, Simoes A, Apkarian P. A non-smooth lower bound on ν. *International Journal of Robust and Nonlinear Control* 2014; **24**(3):477–494, doi:10.1002/rnc.2898. URL http://dx.doi.org/10.1002/rnc.2898.
12. Roos C. A practical approach to worst-case H_∞ performance computation. *Computer-Aided Control System Design (CACSD), 2010 IEEE International Symposium on*, 2010; 380–385, doi:10.1109/CACSD.2010.5612823.
13. Apkarian P, Dao MN, Noll D. Parametric robust structured control design. *Automatic Control, IEEE Transactions on* 2015; **60**(7):1857–1869.
14. Burke J, Lewis A, Overton M. A robust gradient sampling algorithm for nonsmooth, nonconvex optimization. *SIAM J. Optimization* 2005; **15**:751–779.
15. Burke JV, Overton ML. Differential properties of the spectral abscissa and the spectral radius for analytic matrix-valued mappings. *Nonlinear Anal.* 1994; **23**(4):467–488, doi:http://dx.doi.org/10.1016/0362-546X(94)90090-6.
16. Apkarian P, Noll D. Nonsmooth H_∞ synthesis. *IEEE Trans. Automat. Control* January 2006; **51**(1):71–86.
17. Conn AR, Gould NIM, Toint PL. *Trust-Region Methods*. MPS/SIAM Series on Optimization, SIAM: Philadelphia, 2000.
18. Fletcher R. *Practical Methods of Optimization*. John Wiley & Sons, 1987.
19. Noll D, Apkarian P. Spectral bundle methods for non-convex maximum eigenvalue functions: second-order methods. *Mathematical Programming* 2005; **104**(2-3):729–747.
20. Apkarian P, Noll D, Rondepierre A. Mixed H_2/H_∞ control via nonsmooth optimization. *SIAM J. on Control and Optimization* 2008; **47**(3):1516–1546.
21. Polak E. *Computational Methods in Optimization*. Academic Press: New York, 1971.
22. Zhou K, Doyle JC, Glover K. *Robust and Optimal Control*. Prentice Hall, 1996.
23. Doyle JC, Packard A, Zhou K. Review of LFT's, LMI's and μ. *Proc. IEEE Conf. on Decision and Control*, Brighton, UK, 1991; 1227–1232.
24. Klatte D, Kummer B. *Nonsmooth Equations in Optimization. Regularity, Calculus, Methods and Applications*, *Nonconvex Optim. Appl.*, vol. 60. Kluwer Academic Publishers: Dordrecht, 2002.
25. Mifflin R. Semismooth and semiconvex functions in constrained optimization. *SIAM J. Control Optimization* 1977; **15**(6):959–972.
26. Borwein JM, Moors WB. Essentially smooth lipschitz functions. *journal of functional analysis* 1997; **149**(2):305–351.
27. Apkarian P, Noll D, Ravanbod L. Nonsmooth bundle trust-region algorithm with applications to robust stability. *Set-Valued and variational analysis* 2015; To appear.
28. Apkarian P, Noll D, Ravanbod L. Computing the structured distance to instability. *SIAM Conference on Control and its Applications*, 2015; 423–430, doi:10.1137/1.9781611974072.58.
29. Clarke FH. *Optimization and Nonsmooth Analysis*. Canadian Math. Soc. Series, John Wiley & Sons: New York, 1983.
30. Spingarn JE. Submonotone subdifferentials of Lipschitz functions. *Trans. Amer. Math. Soc.* 1981; **264**(1):77–89.
31. Noll D. Cutting plane oracles to minimize non-smooth non-convex functions. *Set-Valued Var. Anal.* 2010; **18**(3-4):531–568.
32. Noll D. Convergence of non-smooth descent methods using the Kurdyka-Łojasiewicz inequality. *J. Optim. Theory Appl.* 2014; **160**(2):553–572.

33. Dao MN. Bundle method for nonconvex nonsmooth constrained optimization. *J. of Convex Analysis* 2015; To appear.
34. Redheffer RM. On a certain linear fractional transformation. *J. Math. and Phys.* 1960; **39**:269–286.
35. Balas GJ, Doyle JC, Glover K, Packard A, Smith R. *μ-Analysis and synthesis toolbox : User's Guide*. The MathWorks, Inc., 1991.

DOI: 10.1002/rnc