SECOND-ORDER NONSMOOTH OPTIMIZATION FOR H_∞ SYNTHESIS

Vincent Bompart * **Dominikus Noll** **
Pierre Apkarian ***

* *ONERA-CERT, 2 av. Edouard Belin, 31055 Toulouse, France*
and Université Paul Sabatier, Institut de Mathématiques, 118 route de Narbonne, 31062 Toulouse, France,
bompart@cert.fr
** *UPS-Institut de Mathématiques, noll@mip.ups-tlse.fr*
*** *ONERA-CERT and UPS-Institut de Mathématiques, apkarian@cert.fr*

Abstract: We consider optimization programs which arise in automatic control applications for H_∞ controller synthesis. We optimize functions which are finite or infinite maxima of smooth functions, or of semismooth functions like the maximum eigenvalue functions. From a nonsmooth semi-infinite problem formulation, a second-order algorithm is developed. Our method is tested on several examples in controller synthesis.

Keywords: H_∞ synthesis, semi-infinite programming, sequential quadratic programming (SQP), nonsmooth optimization, trust-region.

1. INTRODUCTION

We consider a linear time-invariant plant described in standard form by the state-space equations

$$P(s): \qquad \begin{bmatrix} \dot{x} \\ z \\ y \end{bmatrix} = \begin{bmatrix} A & B_1 & B_2 \\ C_1 & D_{11} & D_{12} \\ C_2 & D_{21} & 0_{p_2 \times m_2} \end{bmatrix} \begin{bmatrix} x \\ w \\ u \end{bmatrix},$$

where $x \in \mathbb{R}^n$, $w \in \mathbb{R}^{m_1}$, $u \in \mathbb{R}^{m_2}$, $z \in \mathbb{R}^{p_1}$ and $y \in \mathbb{R}^{p_2}$.

Let $u = Ky$ be a static, internally stabilizing output feedback, $K \in \mathbb{R}^{m_2 \times p_2}$. We denote by $\mathcal{T}_{w\to z}(K)$ the closed-loop transfer function of the performance channel from w to z for a fixed static controller K. We use the more compact notation

$$\mathcal{T}(K,\omega) = \mathcal{T}_{w\to z}(K)(j\omega).$$

Denoting by λ_1 the maximum eigenvalue function on the space of Hermitian matrices, and by $\bar{\sigma}$ the maximum singular value of $p_1 \times m_1$ matrices, we define

$$f(K,\omega) = \lambda_1(\mathcal{T}(K,\omega)^H \mathcal{T}(x,\omega)) = [\bar{\sigma}(\mathcal{T}(K,\omega))]^2.$$

where X^H stands for the conjugate transpose of a complex matrix X.

The square of the H_∞ norm is then defined as

$$\|\mathcal{T}(K)\|_\infty^2 = \max_{\omega\in[0,\infty]} f(K,\omega).$$

Consequently, we are interested in the function $f(K) = \max_{\omega\in[0,\infty]} f(K,\omega)$, which is nonsmooth with two possible sources of nonsmoothness: (a) the infinite max-operator, and (b) the nonsmoothness of λ_1, which may lead to nonsmoothness of $f(\cdot,\omega)$ for fixed ω.

The H_∞-synthesis problem is now the optimization program

$$\min_{K\in\mathbb{R}^{m_2\times p_2}} \max_{\omega\in[0,\infty]} f(K,\omega). \tag{1}$$

Using standard substitutions (as specified in (Apkarian and Noll, 2006*b*), for example), a similar cast is obtained for the synthesis of dynamic controllers K.

First-order nonsmooth optimization methods for (1) have been developed in (Apkarian and Noll, 2006*b*). Since first-order methods sometimes converge slowly in the neighborhood of a local minimum, a second-order approach should be used to speed up the minimization process at the end. The purpose of this contribution is to present such a second-order method, and to combine it with the first-order algorithm (Apkarian and Noll, 2006*b*).

2. APPROACH VIA SEMI-INFINITE PROGRAMMING

In this section, we focus on the unconstrained program (1). In order to comply with the usual notation in nonlinear optimization, we replace the unknown controller data K by a suitable vector $x \in \mathbb{R}^{m_2 p_2}$, respectively, for a dynamic controller K of order k, by $x \in \mathbb{R}^{(m_2+k)(p_2+k)}$.

We assume that for every given controller x, the set of active frequencies $\Omega(x) := \{\omega \in [0,\infty] : f(x) = f(x,\omega)\}$ attaining the maximum $f(x)$ is finite. This hypothesis is satisfied in all practical cases, see e.g. (Boyd and Balakrishnan, 1990), (Bompart *et al.*, 2006). Moreover, to simplify the outset, we will focus during the following on the case where the multiplicity of $\lambda_1(\mathcal{T}(x,\omega)^H \mathcal{T}(x,\omega))$ is one at all active frequencies ω. This simplification is motivated by our experience, which shows that nonsmoothness (b) rarely occurs in practical cases. Extension to the general nonsmooth case follows the same lines, and is presented in the full version of this paper (Bompart *et al.*, 2006), see also (Noll and Apkarian, 2005).

It is useful to cast program (1) as an equivalent semi-infinite program

$$\begin{array}{l} \text{minimize} \quad t \\ \text{subject to } f(x,\omega) - t \le 0,\ \omega \in [0,\infty] \end{array} \tag{2}$$

with decision variable (x,t). Three classes of numerical methods for solving semi-infinite programs are discussed in (Hettich and Kortanek, 1993): exchange of constraints, discretization, and local reduction. Here we use a local reduction method. The main ideas are presented below, see also (Jongen *et al.*, 2004) for this approach.

Let $(\bar{x},\bar{t})$, with $\bar{t} = f(\bar{x})$, be a local solution of (2). Indexing the active frequencies $\bar{\omega}_1,\ldots,\bar{\omega}_p$ at $\bar{x}$, we suppose that the following conditions are satisfied

(i) $f'_\omega(\bar{x},\bar{\omega}_i) = 0, \quad i = 1,\ldots,p.$
(ii) $f''_{\omega\omega}(\bar{x},\bar{\omega}_i) < 0, \quad i = 1,\ldots,p.$
(iii) $f(\bar{x},\omega) < f(\bar{x}), \quad$ for every $\omega \notin \{\bar{\omega}_1,\ldots,\bar{\omega}_p\}$.

These three conditions express the fact that the frequencies $\bar{\omega}_i \in \Omega(\bar{x})$ are the strict global maximizers of $f(\bar{x},\cdot)$. Notice that condition (iii) is the finiteness hypothesis already mentioned, while condition (ii) is slightly conservative, because the necessary optimality condition only tells us that $f''_{\omega\omega}(\bar{x},\bar{\omega}_i) \le 0$.

Conditions (i) and (ii) allow for the implicit function theorem, according to which we can find a neighborhood U of $\bar{x}$, and neighborhoods V_i of $\bar{\omega}_i$ $(i = 1,\ldots,p)$, together with C^1-functions $\omega_i : U \to V_i$, such that the following conditions are satisfied

(iv) $\omega_i(\bar{x}) = \bar{\omega}_i,\ i = 1,\ldots,p.$
(v) $f'_\omega(x,\omega_i(x)) = 0,\ i = 1,\ldots,p.$
(vi) Whenever $x \in U$ and $\omega \in V_i$ satisfy $f'_\omega(x,\omega) = 0$ then $\omega = \omega_i(x)$.

The first two conditions are consequences of the implicit function theorem. By $f''_{\omega\omega}(\bar{x},\bar{\omega}_i) < 0$, and shrinking U if required, we may arrange that $f''_{\omega\omega}(x,\omega_i(x)) < 0$ for x in U, so that $\omega_i(x)$ are local maxima of $f(x,\cdot)$. Moreover, by (vi), $\omega_i(x)$ is the only critical point of $f(x,\cdot)$ in V_i. By second-order optimality, it is therefore a local maximum. Altogether, we have shown that for $x \in U$, $f(x,\omega) \le t$ for all $\omega \in [0,\infty]$ is equivalent to $f(x,\omega_i(x)) \le t$ for $i = 1,\ldots,p$. In other words, (1) is locally equivalent to the finite constrained program

$$\begin{array}{l} \text{minimize} \quad t \\ \text{subject to } f(x,\omega_i(x)) - t \le 0,\ i = 1,\ldots,p \end{array} \tag{3}$$

This allows us to compute $\bar{x}$ via an SQP method applied to (3). In the next section we will discuss how this should be organized, and how the jet information can be computed efficiently.

Remark. Notice that $f(x,\cdot)$ is twice differentiable at each active frequency $\omega_i \in \Omega(x)$, even without the hypothesis that the maximum singular value of $\mathcal{T}(x,\omega_i)$ has multiplicity 1. This holds because the maximum singular value function $\omega \mapsto \bar{\sigma}(H(j\omega))$ of a stable transfer matrix H is twice continuously differentiable at its local maxima (Boyd and Balakrishnan, 1990).

3. SOLVING WITH SQP

3.1 Quadratic tangent subproblem

In order to derive the tangent quadratic program for (3), let us write $G_i(x,t) = f(x,\omega_i(x)) - t$, $F(x,t) = t$. The Lagrangian of (3) is then

$$L(x,t;\tau) = F(x,t) + \sum_{i=1}^{p} \tau_i G_i(x,t),$$

so that, using condition (vi) above

$$L'_x(x,t;\tau) = \sum_{i=1}^{p} \tau_i f'_x\left(x, \omega_i(x)\right)$$
$$L'_t(x,t;\tau) = 1 - \sum_{i=1}^{p} \tau_i$$
$$L''_{xt}(x,t;\tau) = L''_{tt}(x,t;\tau) = 0$$
$$L''_{xx}(x,t;\tau) =$$
$$\sum_{i=1}^{p} \tau_i \Big[f''_{xx}\left(x, \omega_i(x)\right) + f''_{\omega x}\left(x, \omega_i(x)\right) \omega'_i(x)^\top \Big]$$

Differentiating condition (vi) gives

$$0 = f''_{\omega x}\left(x, \omega_i(x)\right) + f''_{\omega\omega}\left(x, \omega_i(x)\right) \omega'_i(x),$$

which allows us to express $\omega'_i(x)$ through derivatives of f. Altogether,

$$L''(x,t;\tau) = \begin{bmatrix} H(x,t;\tau) & 0 \\ 0 & 0 \end{bmatrix}$$

with $H(x,t;\tau) = \sum_{i=1}^{p} \tau_i \Big[f''_{xx}\left(x, \omega_i(x)\right)$

$- f''_{\omega x}\left(x, \omega_i(x)\right) f''_{\omega\omega}\left(x, \omega_i(x)\right)^{-1} f''_{\omega x}\left(x, \omega_i(x)\right)^\top \Big]$.

The tangent quadratic program is now

$$\begin{aligned} \min_{\delta t, \delta x} \ & \delta t + \frac{1}{2} \delta x^\top H(x,t;\tau) \delta x \\ \text{s.t.} \ & f\left(x, \omega_i(x)\right) + f'_x\left(x, \omega_i(x)\right)^\top \delta x - t - \delta t \le 0, \\ & i = 1, \ldots, p \end{aligned} \tag{4}$$

3.2 First and second derivatives formulae

The exact first and second derivatives formulae of f are derived from matrix perturbation theory, as derivatives of a non-degenerate eigenvalue and an associated eigenvector. General formulae may be found in (Golub and Loan, 1989). We specialize them to the case of a Hermitian matrix depending on real parameters.

Recall that $x = \mathrm{vec}(K)$, $\delta x = \mathrm{vec}(\delta K)$. We note eigenvalues of $\mathcal{T}(K,\omega)^H \mathcal{T}(K,\omega)$ as $\lambda_1 > \lambda_2 \ge \lambda_3 \ge \cdots \ge \lambda_{m_1}$ and an orthonormal set of associated eigenvectors as $(q_i)_{1 \le i \le m_1}$ (we drop the dependency on K and ω for ease of notation).

$$f'_K(K,\omega).\delta K = 2 \ \mathrm{Re} \ \left(q_1^H \mathcal{T}^H (\mathcal{T}'_K.\delta K) q_1 \right)$$
$$f'_\omega(K,\omega) = 2 \ \mathrm{Re} \ \left(q_1^H \mathcal{T}^H \mathcal{T}'_\omega q_1 \right)$$
$$f''_{KK}(K,\omega).(\delta K_1, \delta K_2) =$$
$$2 \ \mathrm{Re} \ \Bigg[q_1^H \big((\mathcal{T}'_K.\delta K_2)^H \mathcal{T}'_K.\delta K_1 + \mathcal{T}^H \mathcal{T}''_{KK}.(\delta K_1, \delta K_2) \big) q_1 + \sum_{i=2}^{m_1} \Big((\lambda_1 - \lambda_i)^{-1} \cdot q_1^H \left(\mathcal{T}^H \mathcal{T}'_K.\delta K_1 + (\mathcal{T}'_K.\delta K_1)^H \mathcal{T} \right) q_i \cdot q_i^H \left(\mathcal{T}^H \mathcal{T}'_K.\delta K_2 + (\mathcal{T}'_K.\delta K_2)^H \mathcal{T} \right) q_1 \Big) \Bigg]$$

$$f''_{\omega\omega}(K,\omega) =$$
$$2 \ \mathrm{Re} \ \left[q_1^H \left(\mathcal{T}'^H_\omega \mathcal{T}'_\omega + \mathcal{T}^H \mathcal{T}''_{\omega\omega} \right) q_1 \right] + 2 \sum_{i=2}^{m_1} \frac{\left| q_1^H \left(\mathcal{T}^H \mathcal{T}'_\omega + \mathcal{T}'^H_\omega \mathcal{T} \right) q_i \right|^2}{\lambda_1 - \lambda_i}$$

$$f''_{\omega K}(K,\omega).\delta K =$$
$$2 \ \mathrm{Re} \ \Bigg[q_1^H \left((\mathcal{T}'_K.\delta K)^H \mathcal{T}'_\omega + \mathcal{T}^H \mathcal{T}''_{\omega K}.\delta K \right) q_1 + \sum_{i=2}^{m_1} \Big((\lambda_1 - \lambda_i)^{-1} q_1^H \left(\mathcal{T}^H \mathcal{T}'_\omega + \mathcal{T}'^H_\omega \mathcal{T} \right) q_i \cdot q_i^H \left(\mathcal{T}^H \mathcal{T}'_K.\delta K + (\mathcal{T}'_K.\delta K)^H \mathcal{T} \right) q_1 \Big) \Bigg]$$

The transfer function and its derivatives can be expanded with the state-space matrices in closed-loop $\mathcal{A}(K)$, $\mathcal{B}(K)$, $\mathcal{C}(K)$, $\mathcal{D}(K)$:

$$\begin{aligned} \mathcal{T} &= \mathcal{D}(K) + \mathcal{C}(K) F(K,\omega) \mathcal{B}(K) \\ \mathcal{T}'_K.\delta K &= G_{12}(K,\omega) \delta K G_{21}(K,\omega) \\ \mathcal{T}''_{KK}.(\delta K_1, \delta K_2) &= G_{12}(K,\omega)[\delta K_2 G_{22}(K,\omega) \delta K_1 \\ &\quad + \delta K_1 G_{22}(K,\omega) \delta K_2] G_{21}(K,\omega) \\ \mathcal{T}'_\omega &= -j \mathcal{C}(K) F(K,\omega)^2 \mathcal{B}(K) \\ \mathcal{T}''_{\omega\omega} &= -2 \mathcal{C}(K) F(K,\omega)^3 \mathcal{B}(K) \\ \mathcal{T}''_{\omega K}.\delta K &= -j (G_{12}(K,\omega) \delta K C_2 F(K,\omega)^2 \mathcal{B}(K) \\ &\quad + \mathcal{C}(K) F(K,\omega)^2 B_2 \delta K G_{21}(K,\omega)). \end{aligned}$$

Here, we use the transfer matrices

$$\begin{aligned} F(K,\omega) &= (j\omega I_n - \mathcal{A}(K))^{-1} \\ G_{12}(K,\omega) &= D_{12} + \mathcal{C}(K) F(K,\omega) B_2 \\ G_{21}(K,\omega) &= D_{21} + C_2 F(K,\omega) \mathcal{B}(K) \\ G_{22}(K,\omega) &= C_2 F(K,\omega) B_2. \end{aligned}$$

Remark. The frequency response $T_{(w_1,w_2)\to(z_1,z_2)}$ of the plant

$$\begin{bmatrix} \dot{x} \\ z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} \mathcal{A}(K) & \mathcal{B}(K) & B_2 \\ \mathcal{C}(K) & \mathcal{D}(K) & D_{12} \\ C_2 & D_{21} & 0_{p_2 \times m_2} \end{bmatrix} \begin{bmatrix} x \\ w_1 \\ w_2 \end{bmatrix},$$

can be used to compute *all* the jet elements of tangent program (4). Indeed, it suffices to partition in order to get $\mathcal{T}$, G_{12}, G_{21} and G_{22}.

Hessenberg reduction (Laub, 1981) is well adapted to the evaluation of $T_{(w_1,w_2)\to(z_1,z_2)}$ on the set $\Omega(K)$. The jet information for (4) may be computed using the Matlab control toolbox.

Similar formulae are obtained for dynamic controllers using the substitutions previously evoked.

3.3 Globalisation via trust-region

Program (3) is dependent on the characteristics of the local solution $(\bar{x}, \bar{t})$ of (2). If the computed Newton step is too large, even the number p of active frequencies may vary. Using Helly type

theorems, an upper bound for p may be derived, see e.g. Thm. 4.2. in (Hettich and Kortanek, 1993). For a k-order controller, we expect $p \leq (m_1 + k)(p_1 + k)$ peaks, but this is pessimistic as a rule. By adding a trust-region constraint $\|\delta x\|_2 \leq \Delta_k$ to the tangent quadratic subproblem (4), we can often assure that $x + \delta x$ remains in the domain of validity of the local model, and we avoid the possibility of an unbounded correction.

In order to control quality of the step computed at a given primal-dual pair $(x_k, t_k; \tau_k)$, we use the following ℓ_1-merit functions ϕ_1 and ψ_1, respectively associated with the local program (3) and the tangent program (4)

$$\phi_1(x, t; \mu) = t + \frac{1}{\mu} \sum_{i=1}^{p} [f(x, \omega_i(x)) - t]^+$$

$$\begin{aligned}\psi_1(\delta x, \delta t; \mu) = \delta t &+ \frac{1}{2} \delta x^\top H(x_k, t_k; \tau_k) \delta x \\ &+ \frac{1}{\mu} \sum_{i=1}^{p} \Big[\delta x^\top \nabla f(x_k, \omega_i(x_k)) - \delta t \\ &+ f(x_k, \omega_i(x_k)) - t_k \Big]^+\end{aligned}$$

The agreement between the actual reduction and the predicted reduction is measured by the ratio

$$\rho_k = \frac{\phi_1(x_k, t_k; \mu) - \phi_1(x_k + \delta x, t_k + \delta t; \mu)}{\psi_1(0; \mu) - \psi_1(\delta x, \delta t; \mu)} \quad (5)$$

Then the trust-region radius is managed according to the algorithm given on the next page, based on a model trust-region algorithm from (Conn *et al.*, 2000).

In the inner loop j, the penalty parameter μ_j is initialized with $\mu_0 = (\|\tau_{k,1}\|_\infty + \alpha)^{-1}$ and updated according to the following rule for $j \geq 1$, with a chosen constant $\alpha > 0$

$$\mu_j = \begin{cases} \mu_{j-1} & \text{if } \mu_{j-1}^{-1} \geq \|\tau_{k,j+1}\|_\infty + \alpha, \\ (\|\tau_{k,j+1}\|_\infty + 2\alpha)^{-1} & \text{otherwise.} \end{cases}$$

In this way, $\mu_j < \|\tau_{k,j+1}\|_\infty^{-1}$ and the ℓ_1 merit function ψ_1 is exact (Nocedal and Wright, 1999).

Notice that the trust region procedure in the inner loop j between steps 4 and 6 follows standard lines, but is based on the guess p and $\{\omega_1(x), \ldots, \omega_p(x)\}$ of model (3), so a few comments are in order here. Namely, since the model may be incorrect, the following phenomenon may be observed. The Newton step may be successful with regard to (3), i.e., with regard to the inner loop, but may nevertheless fail when matched with reality in step 7. This is when the first-order step x^C takes over. In the worst case, our method therefore converges with the speed of the underlying first-order technique (Apkarian and Noll, 2006*b*). Alternative first-order methods could be used instead (Apkarian and Noll, 2005*b*; Apkarian and Noll, 2005*a*). A second phenomenon, which also arises due to the necessity to guess p, is addressed in step 3. It may happen that the new $x_{k+1} \in \{x^N, x^C\}$ is no longer consistent with the old model used in the previous step, because the number p had to undergo a change, or because a first-order step x^C had to be taken. The multiplier estimate $\tau_{k+1} = \tau_{k,j+1}$ from the last instance of step 6 is then of little use. We then restart multipliers afresh, or we recycle the old ones.

Finally, when p and $\omega_i(K)$ have been estimated correctly, the quadratic model will ultimately produce steps with quadratic progress. This means the test in step 8 will ultimately accept the Newton step, showing that our method has a fair chance to give local quadratic convergence.

4. TECHNICAL ASPECTS

4.1 Identifying peak frequencies

Our approach assumes that, for any given closed-loop stabilizing controller $x \in \mathbb{R}^{m_2 p_2}$ respectively $x \in \mathbb{R}^{(m_2+k)(p_2+k)}$, we can compute the finite set of maximizers of the frequency curve $\omega \mapsto f(x, \omega)$

$$\Omega(x) = \{\omega_i(x) \geq 0 \,|\, 1 \leq i \leq p\}.$$

Computing peak frequencies can be based on a classical algorithm for estimating the L_∞ norm of a transfer matrix $G(s) = D + C(sI - A)^{-1}B$ explained in detail in (Boyd and Balakrishnan, 1990). This algorithm detects in the first place the peak frequencies $\Omega(K)$, but may also be used to estimate secondary peaks (i.e. local but not global maxima of $f(x, \cdot)$).

Basically, an increasing sequence of lower bounds γ_k is built. The algorithm terminates as soon as an Hamiltonian matrix $H(\gamma)$ has pure imaginary eigenvalues for $\gamma = \gamma_k$, and none for $\gamma = (1+\varepsilon)\gamma_k$. Then $\|G\|_\infty$ lies in the range $[\gamma_k, (1+\varepsilon)\gamma_k)$. In that case,

$$\Omega_k(x) = \{\omega \geq 0 \,|\, (H(\gamma_k) - j\omega I) \text{ is singular}\}$$

is the set of frequencies ω where γ_k is a singular value of $G(j\omega)$. As the algorithm terminates, the set $\Omega_k(x)$ is an approximation of $\Omega(x)$ with a relative tolerance ε on the maximum singular value. We may have $|\Omega_k(x)| > |\Omega(x)|$, because $\Omega_k(x)$ may contain both lower and upper approximations of some peak frequencies. Furthermore, $\Omega_k(x)$ does not contain secondary peaks.

A direct way to estimate both primary and secondary peaks is to compute the subset of the spectrum of $H((1+\varepsilon)\gamma_k)$ contained in the first orthant of the complex plane. By continuity, the imaginary parts $\hat{\omega}_i$ of the eigenvalues near the imaginary axis give approximations of the peak frequencies in $\Omega(x)$. Our experience shows that this method computes primary peaks with high

Fix $0 < \eta_1 \leq \eta_2 < 1$, $0 < \theta < 1$.

1. **Initialize**. Find an initial closed-loop stabilizing controller x_0. Fix $t_0 = f(x_0)$, p_0 and $\tau_0 \in \mathbb{R}_+^{p_0}$, and set counter for outer loop $k = 0$.
2. **Peak estimation.** Given current x_k and $\tau_k \in \mathbb{R}_+^{p_{k-1}}$, estimate number p_k and positions of primary and secondary peaks $\{\omega_1(x_k), \ldots, \omega_{p_k}(x_k)\}$.
3. **Model inconsistency.** If p_k differs from p_{k-1}, or if last step taken was Cauchy step, then modify old $\tau_k \in \mathbb{R}_+^{p_{k-1}}$ or create a new consistent multiplier vector $\tau_k \in \mathbb{R}_+^{p_k}$.
4. **Initialize Newton method**. Put $x_{k,0} = x_k$, $\tau_{k,0} = \tau_k$, $\Delta_0 = 1$, and set counter for inner loop $j = 0$.
5. **Newton step.** Use current iterate $x_{k,j}$, multiplier estimate $\tau_{k,j}$ and trust region radius Δ_j to solve (4) and generate Newton trial step $x_{k,j+1}$, with associated multiplier estimate $\tau_{k,j+1} \in \mathbb{R}_+^{p_k}$.
6. **Decision.** Compare predicted progress in the local quadratic model ψ_1 to progress in ϕ_1 using progress ratio ρ_j in (5). There are two cases:
   Step accepted: $\rho_j \geq \eta_1$. If even $\rho_j \geq \eta_2$ and $\|x_{k,j} - x_{k,j+1}\| = \Delta_j$, then double radius $\Delta_{j+1} = 2\Delta_j$. Otherwise keep $\Delta_{j+1} = \Delta_j$. Put $x^N = x_{k,j+1}$ and $p^N := f(x_k) - f(x^N)$. Goto step 7.
   Step refused: $\rho_j < \eta_1$. Put $\Delta_{j+1} = \|x_{k,j} - x_{k,j+1}\|/2$, increase counter j, and go back to step 5.
7. **Cauchy step.** Given the current iterate x_k, compute a Cauchy step x^C away from x_k using the first-order method (Apkarian and Noll, 2006*b*). Let $p^C := f(x_k) - f(x^C) \geq 0$ be the first-order progress. If $p^N \geq \theta p^C$ let $x_{k+1} = x^N$, and $\tau_{k+1} = \tau_{k,j+1}$, otherwise put $x_{k+1} = x^C$.
8. **Stopping test.** If accepted step $x_{k+1} \in \{x^N, x^C\}$ offers no progress over x_k, stop. Otherwise increase counter k and go back to step 2.

precision, while still giving a reasonably good approximation of secondary peak frequencies. This is satisfactory, because knowing $\omega_i(x)$ to a high precision if $f(x, \omega_i(x)) < f(x)$ is *not* necessary, as the constraint is currently inactive. The closer a secondary peak comes to being active, the higher the precision to which it is computed. The delicate point is the choice of a tolerance to select the eigenvalues with small enough real part. To this end, the derivatives $f'_\omega(x, \omega)$ and $f''_{\omega\omega}(x, \omega)$ can be helpful. They are computed by the exact formulae given in section 3.2. Namely, it makes sense to keep only those frequencies where $\hat{\omega}_i$ where $f'_\omega(x, \hat{\omega}_i) \approx 0$ and $f''_{\omega\omega}(x, \hat{\omega}_i) < 0$.

Remark. Our approach is robust with respect to the estimation of the cardinality p of the unknown set $\Omega(\bar{x})$ in the sense that if we overestimate p, we create a program (3), where some constraints remain inactive near $\bar{x}$, a situation which is automatically dealt with by the SQP solver.

4.2 Stopping criteria

We implemented two stopping tests in order to check convergence. The first tests criticality of the iterate (x_k, t_k) with multipliers τ_k, through the absolute test

$$\|L'_{(x,t)}(x_k, t_k; \tau_k)\| < \varepsilon_1,$$

where $\|\cdot\|$ is a norm on $\mathbb{R}^{m_2 p_2 + 1}$.

The second stopping condition checks the relative progress of the steps via

$$\|(x_k, t_k) - (x_{k-1}, t_{k-1})\| = \|(\delta x, \delta t)\| < \varepsilon_2(1 + \|(x_{k-1}, t_{k-1})\|)$$

As some SQP iterates may become infeasible for problem (3), we also check if $f(x, \omega_i(x)) - t < \varepsilon$ for all $i = 1, \ldots, p$. The algorithm stops as soon as the current iterate is feasible and one of the stopping tests is satisfied.

5. NUMERICAL RESULTS

The results presented here were obtained with the algorithm described in section 3.3. The examples are taken from (Leibfritz, 2003). We used the first-order nonsmooth algorithm from (Apkarian and Noll, 2006*b*) in order to find a stabilizing controller close enough to a local minimum. This first step is recommended, because our second-order method is defined in a neighborhood of a local solution of (2), and needs a stable number of active frequencies to avoid too many restarts. The initial multipliers were set to $\tau_{0i} = 0$ for $i = 1, \ldots, p$, and the termination tolerances to $\varepsilon_1 = \varepsilon_2 = 10^{-5}$. In our tests we compare the speed of convergence of the first and second-order methods.

The first table shows that the second-order algorithm can be used to improve accuracy in cases where the first-order method already meets its convergence criteria.

problem	γ_0	γ_1	iter_1	γ_2	iter_2	$\lvert\Omega(x)\rvert$
AC8	1.71e2	2.005018	$15^\star$	2.005012	$4^\star$	4
AC10	2.01e2	1.325374e1	$15^\star$	1.323655e1	$8^\star$	5
REA3	4.09e2	7.425130e1	$2^\star$	7.425130e1	$9^\star$	1

Column "γ_0" shows the performance of the initial controller $x_0 = \text{vec}(K_0)$, which is stabilizing and computed by the method in (Apkarian and Noll, 2006*a*). Column "γ_1" shows the result obtained by the first-order method, the number of steps is given in column iter_1. Similarly, column "γ_2" gives the performance obtained by the second-order method, iter_2 gives the number of Newton steps. The last column shows the number p of active frequencies at the optimal solution.
The first two examples (AC8 : 9 states, 1 input, 5 outputs; AC10 : 55 states, 2 inputs, 2 outputs) are aircrafts models. The third example (REA3, 12 states, 1 input, 3 outputs) is a reactor model. In all cases, the second-order method terminated with a positive definite reduced Hessian.

The second table illustrates how the second-order method can speed up convergence when the first-order method gives little progress. In the experiment we fixed a maximum of 200 iterations for each run.

problem	γ_0	γ_1	iter_1	γ_2	iter_2	$\lvert\Omega(x)\rvert$
HE2	9.95	4.317234	200			
HE2	9.95	4.593448	100	3.914219	100	2
HE2	9.95	5.223048	30	3.898566	170	2

HE2 is a helicopter model, with 4 states, 2 inputs and 2 outputs. Here the first-order algorithm does not reach convergence after 200 iterations. Switching to the second-order method after 100 first-order steps gives significantly better results, as seen in the second line of the table. We observed that the reduced Hessian was positive definite from the 18th iteration onwards.
The last line shows that it is possible to switch to the second-order algorithm even earlier, namely, as soon as one gets a first-order iterate whose peak frequencies are close enough to those expected at the optimum. In particular, the correct number of peaks has to be guessed or overestimated. In this example, two peak frequencies $p = 2$ became clearly visible after 30 first-order steps.

6. CONCLUSION

We have developed and tested a second-order method to compute static and reduced-order H_∞-controllers and more general synthesis problems with structural constraints on the controller. Our approach avoids the use of Lyapunov variables and therefore leads to small or medium size optimization programs even for large systems (AC10 with 55 states). We use a methodology from semi-infinite programming to obtain a local nonlinear programming model and apply a trust region SQP method. A first-order nonsmooth spectral bundle method is used to initialize the second-order algorithm in the neighborhood of a local solution. Our numerical testing indicates that speed of convergence and numerical reliability may be improved by using the second-order method.

REFERENCES

Apkarian, P. and D. Noll (2005*a*). Nonsmooth optimization for multiband frequency domain control design. *To appear.*

Apkarian, P. and D. Noll (2005*b*). Nonsmooth optimization for multidisk H_∞ synthesis. *To appear.*

Apkarian, P. and D. Noll (2006*a*). Controller design via nonsmooth multi-directional search. *SIAM Journal on Control and Optimization* **44**(6), 1923–1949.

Apkarian, P. and D. Noll (2006*b*). Nonsmooth H_∞ synthesis. *IEEE TAC* **51**(1), 71–86.

Bompart, V., D. Noll and P. Apkarian (2006). Second-order nonsmooth optimization for H_∞ synthesis. *Full paper submitted.*

Boyd, S. and V. Balakrishnan (1990). A regularity result for the singular values of a transfer matrix and a quadratically convergent algorithm for computing its L_∞-norm. *System and Control Letters* **15**(1), 1–7.

Conn, A., N. Gould and Ph. Toint (2000). *Trust-Region Methods*. MPS-SIAM Series on Optimization. Philadelphia.

Golub, G. and C. Van Loan (1989). *Matrix Computations*. second ed.. Johns Hopkins University Press. Baltimore and London.

Hettich, R. and K. Kortanek (1993). Semi-infinite programming: theory, methods, and applications. *SIAM Review* **35**(3), 380–429.

Jongen, H.Th., K. Meer and E. Triesch (2004). *Optimization theory*. Kluwer Academic Publishers.

Laub, A. (1981). Efficient multivariable frequency response computations. *IEEE TAC* **26**(2), 407–408.

Leibfritz, F. (2003). Compl$_e$ib: COnstrained Matrix-optimization Problem *lib*rary - a collection of test examples for nonlinear semidefinite programs, control system design and related problems. Technical report. University of Trier.

Nocedal, J. and S. Wright (1999). *Numerical optimization*. Springer. New York.

Noll, D. and P. Apkarian (2005). Spectral bundle methods for nonconvex maximum eigenvalue functions: second-order methods. *Mathematical Programming Series B* **104**(2), 729–747.