Simultaneous Plant and Controller Optimization Based on Non-smooth Techniques

Ngoc Minh Dao, and Dominikus Noll

***Abstract*—We present an approach to simultaneous design optimization of a plant and its controller. This is based on a bundling technique for solving non-smooth optimization problems under nonlinear and linear constraints. In the absence of convexity, a substitute for the convex cutting plane mechanism is proposed. The method is illustrated on a problem of steady flow in a graph and in robust feedback control design of a mass-spring-damper system.**

***Index Terms*—Robust control, Hankel norm, system with tunable parameters, nonlinear optimization, steady flow.**

I. Introduction

In modern control system, desirable closed-loop characteristics include stability, speed, accuracy, and robustness and depend on both structural and control specifications. Traditionally, structural design with its drive elements precedes and is disconnected from controller synthesis, which may result in a sub-optimal system. In contrast, optimizing plant structure and controller simultaneously may lead to a truly optimal solution. We therefore propose design methods which allow to optimize various elements such as system structure, actuators, sensors, and the controller simultaneously.

Here we focus on simultaneous optimization of certain plant and controller parameters to achieve the best performance for a closed-loop system with constraints. This leads to a complex nonlinear optimization problem involving non-smooth and non-convex objectives and constraints. Suitable optimization methods are discussed to address such problems.

Consider a stable LTI state-space control system

$$G : \begin{cases} \delta x = Ax + Bu \\ y = Cx + Du \end{cases}$$

where δx represents $\dot{x}(t)$ for continuous-time systems and $x(t+1)$ for discrete-time systems, and where $x \in \mathbb{R}^{n_x}$ is the state vector, $u \in \mathbb{R}^m$ the control input vector, and $y \in \mathbb{R}^p$ the output vector. Our interest is the case in which system G is placed in a control system containing actuators, sensors and a feedback controller K, and matrices A, B, C, D and controller K depend smoothly on a design parameter $\mathbf{x}$ varying in $\mathbb{R}^n$ or in some constrained subset of $\mathbb{R}^n$. Denoting by $T_{w\to z}(\mathbf{x})$ the closed-loop performance channel $w \to z$, this brings to the optimization program

$$\begin{array}{ll} \text{minimize} & \|T_{w\to z}(\mathbf{x})\| \\ \text{subject to} & \mathbf{x} \in \mathbb{R}^n, \\ & K = K(\mathbf{x}) \text{ assures closed-loop stability} \end{array} \tag{1}$$

Here standard choices of $\|\cdot\|$ include the H_∞-norm $\|\cdot\|_\infty$, the H_2-norm $\|\cdot\|_2$, or the Hankel norm $\|\cdot\|_H$ which is discussed in more detail in the sections III and VI. Solving (1) leads to non-smooth optimization problems.

II. A proximity control algorithm

Bundle methods are currently among the most effective approachs to solve non-smooth optimization problems. In these methods, subgradients from past iterations are accumulated in a bundle, and a trial step is obtained by a quadratic tangent program based on information stored in the bundle. In the absence of convexity, tangent planes can no longer be used as cutting planes, and a substitute has to be found. A sophisticated management of the proximity control mechanism is also required to obtain a satisfactory convergence theory. We will show in which way these elements can be assembled into a successful algorithm.

For the purpose of solving the problem (1), we present here a non-smooth algorithm for general constrained optimization programs of the form

$$\begin{array}{ll} \text{minimize} & f(\mathbf{x}) \\ \text{subject to} & c(\mathbf{x}) \leqslant 0 \\ & A\mathbf{x} \leqslant b \end{array} \tag{2}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the decision variable, and f and c are potentially non-smooth and non-convex, and where the linear constraints are gathered in $A\mathbf{x} \leqslant b$ and handled directly.

Expanding on an idea in [15, Section 2.2.2], we use a progress function at the current iterate $\mathbf{x}$,

$$F(\cdot, \mathbf{x}) = \max\{f(\cdot) - f(\mathbf{x}) - \nu c(\mathbf{x})_+, c(\cdot) - c(\mathbf{x})_+\},$$

where $c(\mathbf{x})_+ = \max\{c(\mathbf{x}), 0\}$, and $\nu > 0$ is a fixed parameter. It is easy to see that $F(\mathbf{x}, \mathbf{x}) = 0$, where either the left branch $f(\cdot) - f(\mathbf{x}) - \nu c(\mathbf{x})_+$ or the right branch $c(\cdot) - c(\mathbf{x})_+$ in the expression of $F(\cdot, \mathbf{x})$ is active at $\mathbf{x}$, i.e., attains the maximum, depending on whether $\mathbf{x}$ is feasible for the non-linear constraint or not.

Setting $P = \{\mathbf{x} \in \mathbb{R}^n : A\mathbf{x} \leqslant b\}$, it follows from [16, Theorem 6.46] that the normal cone to P at $\mathbf{x}$ is given by

$$N_P(\mathbf{x}) = \{A^\top \eta : \eta \geqslant 0, \eta^\top (A\mathbf{x} - b) = 0\}.$$

We remark therefore that if $\mathbf{x}^*$ is a local minimum of program (2), it is also a local minimum of $F(\cdot, \mathbf{x}^*)$ on P, and then $0 \in \partial_1 F(\mathbf{x}^*, \mathbf{x}^*) + A^\top \eta^*$ for some multiplier $\eta^* \geqslant 0$ with $\eta^{*\top}(A\mathbf{x}^* - b) = 0$. The symbol ∂_1 here stands for the Clarke subdifferential with respect to the first variable. Indeed, if $\mathbf{x}^*$ is a local minimum of (2) then $c(\mathbf{x}^*) \leqslant 0, A\mathbf{x}^* \leqslant b$, and so for $\mathbf{y}$ in a neighborhood of $\mathbf{x}^*$ we have

$$\begin{aligned} F(\mathbf{y}, \mathbf{x}^*) &= \max\{f(\mathbf{y}) - f(\mathbf{x}^*), c(\mathbf{y})\} \\ &\geqslant f(\mathbf{y}) - f(\mathbf{x}^*) \geqslant 0 = F(\mathbf{x}^*, \mathbf{x}^*). \end{aligned}$$

Manuscript received July 15, 2013.

N. M. Dao is with the Department of Mathematics and Informatics, Hanoi National University of Education, Vietnam. Email: minhdn@hnue.edu.vn.

N. M. Dao and D. Noll are with the Institut de Mathématiques, Université de Toulouse, France. Email: noll@mip.ups-tlse.fr.

This implies that $\mathbf{x}^*$ is a local minimum of $F(\cdot,\mathbf{x}^*)$ on P, and therefore $0 \in \partial_1 F(\mathbf{x}^*,\mathbf{x}^*) + N_P(\mathbf{x}^*)$. We now present the following algorithm for computing solutions of program (2).

Algorithm 1 . Proximity control with downshift

```
Parameters: $0 < \gamma < \widetilde{\gamma} < 1, 0 < \gamma < \Gamma < 1, 0 < q < \infty,$
    $0 < c < \infty$, $q < T < \infty$.
1: Initialize outer loop. Choose initial iterate $\mathbf{x}^1$ with $A\mathbf{x}^1 \leqslant b$ and matrix $Q_1 = Q_1^\top$ with $-qI \preceq Q_1 \preceq qI$.
   Initialize memory element $\tau_1^\sharp$ such that $Q_1 + \tau_1^\sharp I \succ 0$.
   Put $j = 1$.
2: Stopping test. At outer loop counter $j$, stop the algorithm if $0 \in \partial_1 F(\mathbf{x}^j,\mathbf{x}^j) + A^\top \eta^j$, for a multiplier $\eta^j \geqslant 0$
   with $\eta^{j\top}(A\mathbf{x}^j - b) = 0$. Otherwise, goto inner loop.
3: Initialize inner loop. Put inner loop counter $k = 1$ and initialize $\tau_1 = \tau_j^\sharp$. Build initial working model
       $F_1(\cdot,\mathbf{x}^j) = g_{0j}^\top(\cdot - \mathbf{x}^j) + \frac{1}{2}(\cdot - \mathbf{x}^j)^\top Q_j(\cdot - \mathbf{x}^j),$
   where $g_{0j} \in \partial_1 F(\mathbf{x}^j,\mathbf{x}^j)$.
4: Trial step generation. At inner loop counter $k$ find solution $\mathbf{y}^k$ of the tangent program
       minimize    $F_k(\mathbf{y},\mathbf{x}^j) + \frac{\tau_k}{2}\|\mathbf{y} - \mathbf{x}^j\|^2$
       subject to  $A\mathbf{y} \leqslant b, \mathbf{y} \in \mathbb{R}^n$.
5: Acceptance test. If
       $\rho_k = \frac{F(\mathbf{y}^k,\mathbf{x}^j)}{F_k(\mathbf{y}^k,\mathbf{x}^j)} \geqslant \gamma,$
   put $\mathbf{x}^{j+1} = \mathbf{y}^k$ (serious step), quit inner loop and goto step 8. Otherwise (null step), continue inner loop with step 6.
6: Update working model. Generate a cutting plane $m_k(\cdot,\mathbf{x}^j) = a_k + g_k^\top(\cdot - \mathbf{x}^j)$ at null step $\mathbf{y}^k$ and counter
   $k$ using downshifted tangents. Compute aggregate plane $m_k^*(\cdot,\mathbf{x}^j) = a_k^* + g_k^{*\top}(\cdot - \mathbf{x}^j)$ at $\mathbf{y}^k$, and then build new
   working model $F_{k+1}(\cdot,\mathbf{x}^j)$.
7: Update proximity control parameter. Compute secondary control parameter
       $\widetilde{\rho}_k = \frac{F_{k+1}(\mathbf{y}^k,\mathbf{x}^j)}{F_k(\mathbf{y}^k,\mathbf{x}^j)}$
   and put
       $\tau_{k+1} = \begin{cases} \tau_k & \text{if } \widetilde{\rho}_k < \widetilde{\gamma}, \\ 2\tau_k & \text{if } \widetilde{\rho}_k \geqslant \widetilde{\gamma}. \end{cases}$
   Increase inner loop counter $k$ and loop back to step 4.
8: Update $Q_j$ and memory element. Update $Q_j \to Q_{j+1}$ respecting $Q_{j+1} = Q_{j+1}^\top$ and $-qI \preceq Q_{j+1} \preceq qI$. Then
   store new memory element
       $\tau_{j+1}^\sharp = \begin{cases} \tau_k & \text{if } \rho_k < \Gamma, \\ \frac{1}{2}\tau_k & \text{if } \rho_k \geqslant \Gamma. \end{cases}$
   Increase $\tau_{j+1}^\sharp$ if necessary to ensure $Q_{j+1} + \tau_{j+1}^\sharp I \succ 0$.
   If $\tau_{j+1}^\sharp > T$ then re-set $\tau_{j+1}^\sharp = T$. Increase outer loop counter $j$ and loop back to step 2.
```

Convergence theory of Algorithm 1 is discussed in [7], [10] and based on these results, we can prove the following theorem.

Theorem 1: Suppose f and c in program (2) are lower-C^1 functions such that the following conditions hold:

(a) f is weakly coercive on constraint set $\Omega = \{\mathbf{x} \in \mathbb{R}^n : c(\mathbf{x}) \leqslant 0, A\mathbf{x} \leqslant b\}$, i.e., if $\mathbf{x}^j \in \Omega$, $\|\mathbf{x}^j\| \to \infty$, then $f(\mathbf{x}^j)$ is not monotonically decreasing.

(b) c is weakly coercive on P, i.e., if $\mathbf{x}^j \in P$, $\|\mathbf{x}^j\| \to \infty$, then $c(\mathbf{x}^j)$ is not monotonically decreasing.

Then the sequence of serious iterates $\mathbf{x}^j \in P$ generated by Algorithm 1 is bounded, and every accumulation point $\mathbf{x}^*$ of the $\mathbf{x}^j$ satisfies $\mathbf{x}^* \in P$ and $0 \in \partial_1 F(\mathbf{x}^*,\mathbf{x}^*) + A^\top \eta^*$ for some multiplier $\eta^* \geqslant 0$ with $\eta^{*\top}(A\mathbf{x}^* - b) = 0$. □

An immediate consequence of Theorem 1 is the following

Corollary 2: Under the hypotheses of the theorem, every accumulation point of the sequence of serious iterates generated by Algorithm 1 is either a critical point of constraint violation, or a Karush-Kuhn-Tucker point of program (2).

Proof: Suppose $\mathbf{x}^*$ is an accumulation point of the sequence of serious iterates generated by Algorithm 1. According to Theorem 1 we have $0 \in \partial_1 F(\mathbf{x}^*,\mathbf{x}^*) + N_P(\mathbf{x}^*)$. By using [4, Proposition 9] (see also [5, Proposition 2.3.12]), there exist constants λ_0, λ_1 such that

$$0 \in \lambda_0 \partial f(\mathbf{x}^*) + \lambda_1 \partial c(\mathbf{x}^*) + N_P(\mathbf{x}^*),$$
$$\lambda_0 \geqslant 0, \lambda_1 \geqslant 0,\ \lambda_0 + \lambda_1 = 1.$$

If $c(\mathbf{x}^*) > 0$ then $\partial_1 F(\mathbf{x}^*,\mathbf{x}^*) = \partial c(\mathbf{x}^*)$, and therefore $0 \in \partial c(\mathbf{x}^*) + N_P(\mathbf{x}^*)$, which means that $\mathbf{x}^*$ is a critical point of constraint violation. In the case of $c(\mathbf{x}^*) \leqslant 0$, if $\mathbf{x}^*$ fails to be a Karush-Kuhn-Tucker point of (2), then λ_0 must equal 0, and so $0 \in \partial c(\mathbf{x}^*) + N_P(\mathbf{x}^*)$. We obtain that $\mathbf{x}^*$ is either a critical point of constraint violation, or a Karush-Kuhn-Tucker point of program (2). ■

In the absence of convexity, proving convergence to a single Karush-Kuhn-Tucker point is generally out of reach, but the following result gives nonetheless a satisfactory answer for stopping of the algorithm.

Corollary 3: Under the hypotheses of the theorem, for every $\varepsilon > 0$ there exists an index $j_0(\varepsilon) \in \mathbb{N}$ such that for every $j \geqslant j_0(\varepsilon)$, $\mathbf{x}^j$ is within ε-distance of the set L of critical points $\mathbf{x}^*$ in the sense of the theorem.

Proof: By the fact that our algorithm assures always $\mathbf{x}^j - \mathbf{x}^{j+1} \to 0$ and Ostrowski's theorem [14, Theorem 26.1], the set of limit point L of the sequence $\mathbf{x}^j$ is either singleton or a compact continuum. Our construction then assures convergence of $\mathbf{x}^j$ to the limiting set L in the sense of the Hausdorff distance. See [11] for the details. ■

III. Hankel norm

Given a stable LTI system

$$G : \begin{cases} \dot{x} = Ax + Bw \\ z = Cx \end{cases}$$

with state $x \in \mathbb{R}^{n_x}$, input $w \in \mathbb{R}^m$, and output $z \in \mathbb{R}^p$, if we think of $w(t)$ as an excitation at the input which acts over the time period $0 \leqslant t \leqslant T$, then the ring of the system after the excitation has stopped at time T is $z(t)$ for $t > T$. If signals are measured in the energy norm, this leads to that the Hankel norm of the system G is defined as

$$\|G\|_H = \sup_{T>0}\left\{ \left(\int_T^\infty z(t)^2 dt \right)^{1/2} : \int_0^T w(t)^2 dt \leqslant 1, w(t) = 0 \text{ for } t > T \right\}.$$

For the discrete-time case, the Hankel norm of G is given by

$$\|G\|_H = \sup_{T>0}\left\{ \left(\sum_{t=T}^\infty z(t)^2 \right)^{1/2} : \sum_{t=0}^T w(t)^2 \leqslant 1, w(t) = 0 \text{ for } t > T \right\}.$$

The Hankel norm can be understood as measuring the tendency of a system to store energy, which is later retrieved to produce undesired noise effects known as system ring. Minimizing the Hankel norm therefore reduces the ringing in the system. It is worth to note that in both continuous-time and discrete-time cases we have the following

Proposition 4: If X and Y are the controllability and observability Gramians of the stable system G, then

$$\|G\|_H = \sqrt{\lambda_1(XY)},$$

where λ_1 denotes the maximum eigenvalue of a symmetric or Hermitian matrix.

Proof: See [6] and also [8, Section 2.3]. ■

IV. Steady flow in a graph

Here we consider the problem of steady flow in a directed graph $\mathscr{G} = (\mathscr{V}, \mathscr{A})$ with sources, sinks, and interior nodes, $\mathscr{V} = \mathscr{V}_{\text{stay}} \cup \mathscr{V}_{\text{in}} \cup \mathscr{V}_{\text{out}}$, and not excluding self-arcs. For nodes $i, j \in \mathscr{V}$ connected by an arc $(i, j) \in \mathscr{A}$ the transition probability $i \to j$ quantifies the tendency of flow going from node i towards node j. As an example we may for instance consider a large fairground with separated entrances and exits, where itineraries between stands, entrances and exits are represented by the graph. By acting on the transition probabilities between nodes connected by arcs, we expect to guide the crowd in such a way that a steady flow is assured, and a safe evacuation is possible in case of an emergency.

Assume that an individual at interior node $j \in \mathscr{V}_{\text{stay}}$ decides with probability $a_{jj'} \geqslant 0$ to proceed to a neighboring node $j' \in \mathscr{V}_{\text{stay}}$, where neighboring means $(j, j') \in \mathscr{A}$, or with probability $a_{jk} \geqslant 0$ to a neighboring exit node $k \in \mathscr{V}_{\text{out}}$, where $(j, k) \in \mathscr{A}$. The case $(j, j) \in \mathscr{A}$ of deciding to stay at stand $j \in \mathscr{V}_{\text{stay}}$ is not excluded. Similarly, an individual entering at $i \in \mathscr{V}_{\text{in}}$ proceeds to a neighboring interior node $j \in \mathscr{V}_{\text{stay}}$ with probability $b_{ij} \geqslant 0$, where $(i, j) \in \mathscr{A}$. We assume for simplicity that there is no direct transmission from entrances to exits. Then

$$\sum_{j' \in \mathscr{V}_{\text{stay}}:(j,j') \in \mathscr{A}} a_{jj'} + \sum_{k \in \mathscr{V}_{\text{out}}:(j,k) \in \mathscr{A}} a_{jk} = 1, \tag{3}$$

for every $j \in \mathscr{V}_{\text{stay}}$, and

$$\sum_{j \in \mathscr{V}_{\text{stay}}:(i,j) \in \mathscr{A}} b_{ij} = 1 \tag{4}$$

for every $i \in \mathscr{V}_{\text{in}}$. Let $x_j(t)$ denote the number of people present at interior node $j \in \mathscr{V}_{\text{stay}}$ and time t, and $w_i(t)$ the number of people entering the fairground through entry $i \in \mathscr{V}_{\text{in}}$ at time t. Then the number of people present at interior node $j \in \mathscr{V}_{\text{stay}}$ and time $t + 1$ is

$$x_j(t+1) = \sum_{j' \in \mathscr{V}_{\text{stay}}:(j',j) \in \mathscr{A}} a_{j'j} x_{j'}(t) + \sum_{i \in \mathscr{V}_{\text{in}}:(i,j) \in \mathscr{A}} b_{ij} w_i(t).$$

We quantify the total number of individuals still inside the fairground via the weighted sum

$$z(t) = \sum_{j \in \mathscr{V}_{\text{stay}}} c_j x_j(t)$$

at time t, where $c_j > 0$ are fixed weights. We assess the performance of the network by using the L_2-norm to quantify input and output flows w, z. This attributes a high cost to a strong concentration of people at a single spot. Take $\mathbf{x}$ to regroup the parameters $a_{jj'}, a_{jk}, b_{ij}$, the discrete LTI system above has the form $G(\mathbf{x}) = (A(\mathbf{x}), B(\mathbf{x}), C, 0)$, where C is the row vector of c_j's. The Hankel norm $\|G(\mathbf{x})\|_H$ may then be interpreted as computing the worst-case of all scenarios where the inflow w is stopped at some time T, and the outflow is measured via the pattern $z(t)$, $t \geqslant T$, with which the fairground is emptied. Minimizing $\|z\|_{2,[T,\infty)}/\|w\|_{2,(0,T]}$ may then be understood as enhancing overall safety of the network. It leads to the optimization program

$$\begin{array}{ll} \text{minimize} & \|G(\mathbf{x})\|_H \\ \text{subject to} & G(\mathbf{x}) \text{ internally stable} \\ & a_{jj'} \geqslant 0, a_{jk} \geqslant 0, b_{ij} \geqslant 0, (3), (4) \end{array} \tag{5}$$

which is a version of (1).

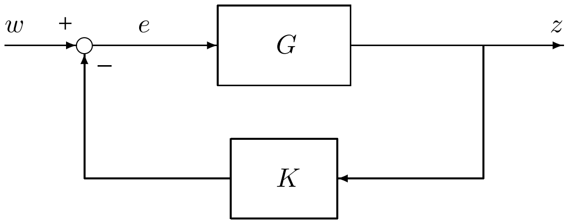

Fig. 1. Control architecture in the fairground.

In an extended model one might consider measuring the number of people at some selected nodes $j \in \mathscr{V}_{\text{stay}} \cup \mathscr{V}_{\text{out}}$, and use this to react via a feedback controller at the entry gates as in Figure 1. With this controller, we can regulate the number of people in the fairground. More accurately, the feedback controller $K = K(\kappa)$ includes admission rates κ_i at entry gate i, and the number of people entering may be restricted based on the total weighted number of people inside the fairground. Letting $T_{w \to z}(\mathbf{x}, \kappa)$ denote the closed-loop transfer function of the performance channel mapping w into z, this leads to the following problem where controller and parts of the plant are optimized simultaneously.

$$\begin{array}{ll} \text{minimize} & \|T_{w \to z}(\mathbf{x}, \kappa)\|_H \\ \text{subject to} & K = K(\kappa) \text{ assures closed-loop stability,} \\ & a_{jj'} \geqslant 0, a_{jk} \geqslant 0, b_{ij} \geqslant 0, \kappa_i \geqslant 0, (3), (4) \end{array} \tag{6}$$

V. Robust control of a mass-spring-damper system

In this section we discuss a 1DOF mass-spring-damper system with mass m, spring stiffness k and damping coefficient c. The values can be in any consistent system of units, for example, in SI units, m in kilograms, k in newtons per meter, and c in newton-seconds per meter or kilograms per second. The system is of second order, since it has a mass which can contain both kinetic and potential energy. The force F is considered as input u, and the displacement p of the mass from the equilibrium position is considered as output y of this system. By Hooke's law, the force exerted by the spring is

$$F_s = -kp.$$

Let v be the velocity of the mass, then the damping force F_d is expressed as

$$F_d = -cv = -c\frac{dp}{dt} = -c\dot{p}$$

due to d'Alembert's principle. Using Newton's second law, we have

$$F + F_s + F_d = m\frac{d^2p}{dt^2} = m\ddot{p},$$

which gives

$$m\ddot{p} + c\dot{p} + kp = u.$$

A possible selection of state variables is the displacement p and the velocity v. The linear model of the mass-spring-damper is then described by

$$G : \begin{cases} \dot{x} = Ax + Bu \\ y = Cx \end{cases}$$

where

$$A = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{c}{m} \end{bmatrix},\ B = \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} \text{ and } C = \begin{bmatrix} 1 & 0 \end{bmatrix}.$$

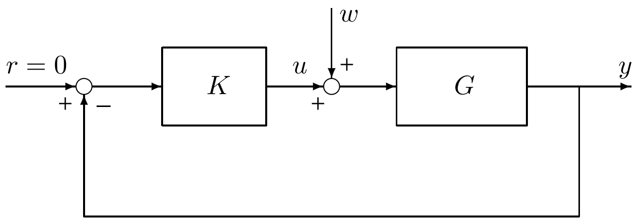

Fig. 2. The structure of mass-damper-spring control system

The design objective for the mass-spring-damper system with a disturbance is to find an output feedback control law $u = Ky$ which stabilizes the closed-loop system while minimizing worst-case energy of output $z = [y\ u]^\top$ in order to avoid the disturbance input w to affect the system. In realistic systems, the physical parameters k and c are not known exactly but can be enclosed in intervals. Assuming the controller is parameterized as $K(\kappa)$, taking $\mathbf{x}$ to regroup the tunable parameters k, c and κ, and denoting by $T_{w\to z}(\mathbf{x})$ the closed-loop performance channel $w \to z$, this leads to the optimization problem

$$\begin{array}{ll} \text{minimize} & \|T_{w\to z}(\mathbf{x})\| \\ \text{subject to} & \mathbf{x} = (k, c, \kappa) \in \mathbb{R}^n, \\ & K = K(\kappa) \text{ assures closed-loop stability,} \\ & k \text{ and } c \text{ are in some intervals} \end{array} \qquad (7)$$

where choices of $\|\cdot\|$ include the H_∞-norm $\|\cdot\|_\infty$ or the Hankel norm $\|\cdot\|_H$.

VI. Clarke subdifferentials of the Hankel norm

In order to apply nonlinear and non-smooth optimization techniques to programs of the form (5), (6) and (7) it is necessary to provide derivative information of the objective function

$$f(\mathbf{x}) = \|G(\mathbf{x})\|_H^2 = \lambda_1(X(\mathbf{x})Y(\mathbf{x})),$$

where $X(\mathbf{x})$ and $Y(\mathbf{x})$ are the controllability and observability Gramians. In the discrete-time case, $X(\mathbf{x})$ and $Y(\mathbf{x})$ can be obtained from the Lyapunov equations

$$A(\mathbf{x})XA^\top(\mathbf{x}) - X + B(\mathbf{x})B^\top(\mathbf{x}) = 0, \qquad (8)$$

$$A^\top(\mathbf{x})YA(\mathbf{x}) - Y + C^\top(\mathbf{x})C(\mathbf{x}) = 0. \qquad (9)$$

Remark that despite the symmetry of X and Y the product XY needs not be symmetric, but stability of $A(\mathbf{x})$ guarantees $X \succ 0$, $Y \succ 0$ in (8), (9), so that we can write

$$\lambda_1(XY) = \lambda_1(X^{\frac{1}{2}}YX^{\frac{1}{2}}) = \lambda_1(Y^{\frac{1}{2}}XY^{\frac{1}{2}}),$$

which brings us back in the realm of eigenvalue theory of symmetric matrices.

Recalling the definition of the spectral radius of a matrix

$$\rho(M) = \max\{|\lambda| : \lambda \text{ eigenvalue of } M\},$$

we can address programs (5) and (6) in the following program

$$\begin{array}{ll} \text{minimize} & f(\mathbf{x}) := \|G(\mathbf{x})\|_H^2 \\ \text{subject to} & c(\mathbf{x}) := \rho(A(\mathbf{x})) - 1 + \varepsilon \leqslant 0 \end{array} \qquad (10)$$

for some fixed small $\varepsilon > 0$. Notice that $f = \|\cdot\|_H^2 \circ G(\cdot)$ is a composite function of a semi-norm and a smooth mapping $\mathbf{x} \mapsto G(\mathbf{x})$, which implies that it is lower-C^2, and therefore also lower-C^1 in the sense of [16, Definition 10.29]. Theoretical properties of the spectral radius $c(\mathbf{x})$, used in the constraint, have been studied in [3]. We also have $X(\mathbf{x}) \succ 0$ and $Y(\mathbf{x}) \succ 0$ on the feasible set $\mathcal{C} = \{\mathbf{x} : c(\mathbf{x}) \leqslant 0\}$, so that f is well-defined and locally Lipschitz on $\mathcal{C}$.

Let $\mathbb{M}_{n,m}$ be the space of $n \times m$ matrices, equipped with the corresponding scalar product $\langle X, Y\rangle = \mathrm{Tr}(X^\top Y)$, where $X^\top$ and $\mathrm{Tr}(X)$ are respectively the transpose and the trace of matrix X. We denote by $\mathbb{B}_m$ the set of $m \times m$ symmetric positive semidefinite matrices with trace 1. Set $Z := X^{\frac{1}{2}}YX^{\frac{1}{2}}$ and pick Q to be a matrix whose columns form an orthonormal basis of the ν-dimensional eigenspace associated with $\lambda_1(Z)$. By [13, Theorem 3], the Clarke subdifferential of f at $\mathbf{x}$ consists of all subgradients g_U of the form

$$g_U = (\mathrm{Tr}(Z_1(\mathbf{x})^\top QUQ^\top), \ldots, \mathrm{Tr}(Z_n(\mathbf{x})^\top QUQ^\top))^\top,$$

where $U \in \mathbb{B}_\nu$, and where $M_i(\mathbf{x}) := \frac{\partial M(\mathbf{x})}{\partial \mathbf{x}_i}, i = 1, \ldots, n$ for any matrix $M(\mathbf{x})$.We next have

$$Z_i(\mathbf{x}) = \chi_i(\mathbf{x})YX^{\frac{1}{2}} + X^{\frac{1}{2}}Y_i(\mathbf{x})X^{\frac{1}{2}} + X^{\frac{1}{2}}Y\chi_i(\mathbf{x}), \quad (11)$$

where $\chi_i(\mathbf{x}) := \frac{\partial X^{\frac{1}{2}}(\mathbf{x})}{\partial \mathbf{x}_i}$. It follows from (8) and (9) that

$$\begin{aligned} &A(\mathbf{x})X_i(\mathbf{x})A^\top(\mathbf{x}) - X_i(\mathbf{x}) = -A_i(\mathbf{x})XA^\top(\mathbf{x}) \\ &\quad - A(\mathbf{x})X[A_i(\mathbf{x})]^\top - B_i(\mathbf{x})B^\top(\mathbf{x}) - B(\mathbf{x})[B_i(\mathbf{x})]^\top, \quad (12) \\ &A^\top(\mathbf{x})Y_i(\mathbf{x})A(\mathbf{x}) - Y_i(\mathbf{x}) = -[A_i(\mathbf{x})]^\top YA(\mathbf{x}) \\ &\quad - A^\top(\mathbf{x})YA_i(\mathbf{x}) - [C_i(\mathbf{x})]^\top C(\mathbf{x}) - C^\top(\mathbf{x})C_i(\mathbf{x}). \quad (13) \end{aligned}$$

ISBN: 978-988-19253-1-2
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Since $X^{\frac{1}{2}}X^{\frac{1}{2}} = X$,

$$X^{\frac{1}{2}}\chi_i(\mathbf{x}) + \chi_i(\mathbf{x})X^{\frac{1}{2}} = X_i(\mathbf{x}). \tag{14}$$

Altogether, we obtain Algorithm 2 to compute elements of the subdifferential of $f(\mathbf{x})$.

Algorithm 2 . Computing subgradients.

```
Input: $\mathbf{x} \in \mathbb{R}^n$. Output: $g \in \partial f(\mathbf{x})$.
Compute $A_i(\mathbf{x}) = \frac{\partial A(\mathbf{x})}{\partial \mathbf{x}_i}, B_i(\mathbf{x}) = \frac{\partial B(\mathbf{x})}{\partial \mathbf{x}_i}$, $C_i(\mathbf{x}) = \frac{\partial C(\mathbf{x})}{\partial \mathbf{x}_i}$, $i = 1, \ldots, n$ and $X, Y$ solutions of (8), (9), respectively.
Compute $X^{\frac{1}{2}}$ and $Z = X^{\frac{1}{2}} Y X^{\frac{1}{2}}$.
For $i = 1, \ldots, n$ compute $X_i(\mathbf{x})$ and $Y_i(\mathbf{x})$ solutions of (12) and (13), respectively.
For $i = 1, \ldots, n$ compute $\chi_i(\mathbf{x})$ solution of (14) and $Z_i(\mathbf{x})$ using (11).
Determine a matrix $Q$ whose columns form an orthonormal basis of the $\nu$-dimensional eigenspace associated with $\lambda_1(Z)$.
Pick $U \in \mathbb{B}_\nu$, and return
    $(\mathrm{Tr}(Z_1(\mathbf{x})^\top QUQ^\top), \ldots, \mathrm{Tr}(Z_n(\mathbf{x})^\top QUQ^\top))^\top$,
    a subgradient of $f$ at $\mathbf{x}$.
```

Remark 1: In the continuous-time case, the Gramians $X(\mathbf{x})$ and $Y(\mathbf{x})$ can be obtained from the continuous Lyapunov equations

$$A(\mathbf{x})X + XA^\top(\mathbf{x}) + B(\mathbf{x})B^\top(\mathbf{x}) = 0, \tag{15}$$
$$A^\top(\mathbf{x})Y + YA(\mathbf{x}) + C^\top(\mathbf{x})C(\mathbf{x}) = 0, \tag{16}$$

Therefore, $X_i(\mathbf{x})$ and $Y_i(\mathbf{x})$ are solutions respectively of the following equations

$$\begin{aligned} A(\mathbf{x})X_i(\mathbf{x}) + X_i(\mathbf{x})A^\top(\mathbf{x}) &= -A_i(\mathbf{x})X - X[A_i(\mathbf{x})]^\top \\ &\quad - B_i(\mathbf{x})B^\top(\mathbf{x}) - B(\mathbf{x})[B_i(\mathbf{x})]^\top, \quad (17) \\ A^\top(\mathbf{x})Y_i(\mathbf{x}) + Y_i(\mathbf{x})A(\mathbf{x}) &= -[A_i(\mathbf{x})]^\top Y - YA_i(\mathbf{x}) \\ &\quad - [C_i(\mathbf{x})]^\top C(x) - C^\top(\mathbf{x})C_i(\mathbf{x}). \quad (18) \end{aligned}$$

In addition, let us note that for this case, the stability constraint in program (10) is $c(\mathbf{x}) = \alpha(A(\mathbf{x})) + \varepsilon \leqslant 0$, where $\alpha(\cdot)$ denotes the spectral abscissa of a square matrix, i.e., the maximum of the real parts of its eigenvalues. □

We now introduce a smooth relaxation of Hankel norm. It is based on a result established by Y. Nesterov in [9], which gives a fine analysis of the convex bundle method in situations where the objective $f(\mathbf{x})$ has the specific structure of a max-function, including the case of a convex maximum eigenvalue function. These findings indicate that for a given precision, such programs may be solved with lower algorithmic complexity using smooth relaxations. While these results are *a priori* limited to the convex case, it may be interesting to apply this idea as a heuristic in the non-convex situation. More precisely, we can try to solve problem (10), (2) by replacing the function $f(\mathbf{x}) = \lambda_1(Z(\mathbf{x}))$ by its smooth approximation

$$f_\mu(\mathbf{x}) := \mu \ln \left(\sum_{i=1}^{n_x} e^{\lambda_i(Z(\mathbf{x}))/\mu} \right), \tag{19}$$

where $\mu > 0$ is a tolerance parameter, n_x the order of matrix Z, and where λ_i denotes the ith eigenvalue of a symmetric or Hermitian matrix. Then

$$\nabla f_\mu(Z) = \left(\sum_{i=1}^{n_x} e^{\lambda_i(Z)/\mu} \right)^{-1} \sum_{i=1}^{n_x} e^{\lambda_i(Z)/\mu} q_i(Z) q_i(Z)^\top,$$

with $q_i(Z)$ the ith column of the orthogonal matrix $Q(Z)$ from the eigendecomposition of symmetric matrix $Z = Q(Z)D(Z)Q(Z)^\top$. This yields

$$\begin{aligned} &\nabla f_\mu(\mathbf{x}) = \\ &\qquad (\mathrm{Tr}(Z_1(\mathbf{x})^\top \nabla f_\mu(Z)), \ldots, \mathrm{Tr}(Z_n(\mathbf{x})^\top \nabla f_\mu(Z)))^\top. \end{aligned}$$

Let us note that

$$f(\mathbf{x}) \leqslant f_\mu(\mathbf{x}) \leqslant f(\mathbf{x}) + \mu \ln n_x.$$

Therefore, to find an ϵ-solution of problem (2), we have to find an $\frac{\epsilon}{2}$-solution of the smooth problem

$$\begin{array}{ll} \text{minimize} & f_\mu(\mathbf{x}) \\ \text{subject to} & c(\mathbf{x}) \leqslant 0 \\ & A\mathbf{x} \leqslant b \end{array} \tag{20}$$

with $\mu = \frac{\epsilon}{2 \ln n_x}$. This smoothed problem can be solved using standard NLP software. We have initialized the non-smooth algorithm 1 with the solution of problem (20).

VII. Numerical experiments

A. Steady Flow in a Graph

We give an illustration of programs (5) and (6). Let $\mathscr{V}_{\text{stay}} = \{1, 2, \ldots, n_x\}$, $\mathscr{V}_{\text{in}} = \{1, 2, \ldots, m\}$ and $\mathscr{V}_{\text{out}} = \{1, 2, \ldots, p\}$. Taking $\mathbf{x}$ to regroup the unknown tunable parameters $a_{jj'}, b_{ij}$ and setting $A(\mathbf{x}) = [a_{jj'}]^\top_{n_x \times n_x}$, $B(\mathbf{x}) = [b_{ij}]^\top_{m \times n_x}$, $C = [c_1, \ldots, c_{n_x}]$, where $a_{jj'} = 0$ if $(j, j') \notin \mathscr{A}, b_{ij} = 0$ if $(i, j) \notin \mathscr{A}$, we have a discrete LTI system

$$G(\mathbf{x}) : \begin{cases} x(t+1) & = A(\mathbf{x})x(t) + B(\mathbf{x})w(t) \\ z(t) & = Cx(t). \end{cases}$$

Note that the linear constraint conditions in (5) as well as (6) can be transferred to the form

$$\begin{cases} A_{\text{eq}}\mathbf{x} = b_{\text{eq}}, \\ \mathbf{x} \geqslant 0. \end{cases}$$

We now take the graph $\mathscr{G} = (\mathscr{V}, \mathscr{A})$ with $n_x = 36$, $m = 2$ and $p = 2$ as in Figure 3. Let $z(t)$ be the total number of individuals inside the fairground with doubled weights at 6 nodes in the center that form a hexagon as compared to the other nodes. We start with the case without controller and initialize at the uniform distribution $\mathbf{x}^1$, where $f(\mathbf{x}^1) = 528.7672$ and $\|G(\mathbf{x}^1)\|_H = 22.9949$. In order to save time, we use the minimizer of the relaxation $f_\mu(\mathbf{x})$ in (19) with initial $\mathbf{x}^1$ to initialize the non-smooth algorithm 1. Our algorithm then returns the optimal $\mathbf{x}^\dagger$ with $f(\mathbf{x}^\dagger) = 16.5817$, meaning $\|G(\mathbf{x}^\dagger)\|_H = 4.0721$.

In the case with controller $K = K(\kappa)$, $\kappa = [\kappa_1 \ \ldots \ \kappa_m]^\top$, as shown in Figure 1, we have

$$T_{w \to z}(\mathbf{x}, \kappa) : \begin{cases} x(t+1) & = A(\mathbf{x})x(t) + B(\mathbf{x})e(t) \\ z(t) & = Cx(t). \end{cases}$$

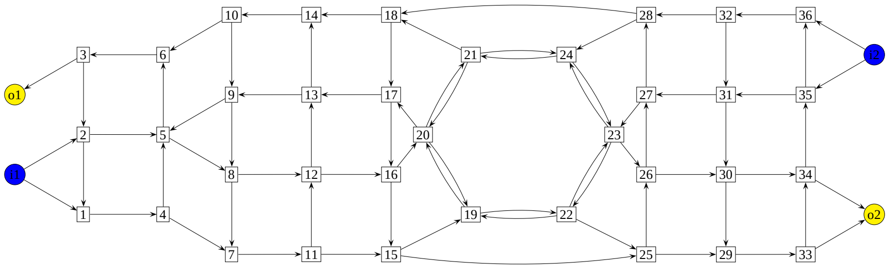

Fig. 3. Model of the fairground.

Here $e(t)=w(t)-Kz(t)=w(t)-KCx(t)$, which gives

$$T_{w\to z}(\mathbf{x},\kappa)=\left[\begin{array}{c|c} A(\mathbf{x})-B(\mathbf{x})K(\kappa)C & B(\mathbf{x})\\ \hline C & 0\end{array}\right].$$

Initializing at $(\mathbf{x},\kappa) = (\mathbf{x}^1,0)$ with remarking that $T_{w\to z}(\mathbf{x},0)=G(\mathbf{x})$ and proceeding as in the previous case, we obtain the optimal $(\mathbf{x}^*,\kappa^*)$ with $f(\mathbf{x}^*,\kappa^*) = 2.0001$, meaning $\|T_{w\to z}(\mathbf{x}^*,\kappa^*)\|_H = 1.4142$. Step responses and ringing effects in unit step and white noise responses truncated at $T=30$ for the three systems $G(\mathbf{x}^1)=T_{w\to z}(\mathbf{x}^1,0)$, $G(\mathbf{x}^\dagger)$ and $T_{w\to z}(\mathbf{x}^*,\kappa^*)$ are compared in Figure 4 and Figure 5.

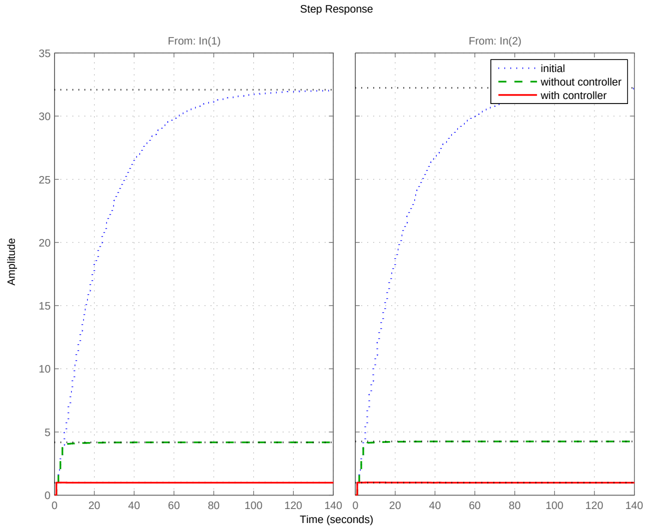

Fig. 4. Experiment 1. Step responses of three systems $G(\mathbf{x}^1)$ (dotted), $G(\mathbf{x}^\dagger)$ (dashed) and $T_{w\to z}(\mathbf{x}^*,\kappa^*)$ (solid).

B. Robust Control of a Mass-Spring-Damper System

Here we apply Algorithm 1 to solve problem (7), where the mass-spring-damper plant with a disturbance is given by

$$P:\begin{bmatrix}\dot{x}\\ z\\ y\end{bmatrix}=\left[\begin{array}{c|cc} A & B_1 & B\\ \hline C_1 & 0 & D_{12}\\ C & 0 & 0\end{array}\right]\begin{bmatrix}x\\ w\\ u\end{bmatrix},$$

with

$$A=\begin{bmatrix}0 & 1\\ -\frac{k}{m} & -\frac{c}{m}\end{bmatrix},\quad B_1=B=\begin{bmatrix}0\\ \frac{1}{m}\end{bmatrix}$$
$$C_1=\begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix},\qquad D_{12}=\begin{bmatrix}0\\ 1\end{bmatrix}\text{ and } C=\begin{bmatrix}1 & 0\end{bmatrix}.$$

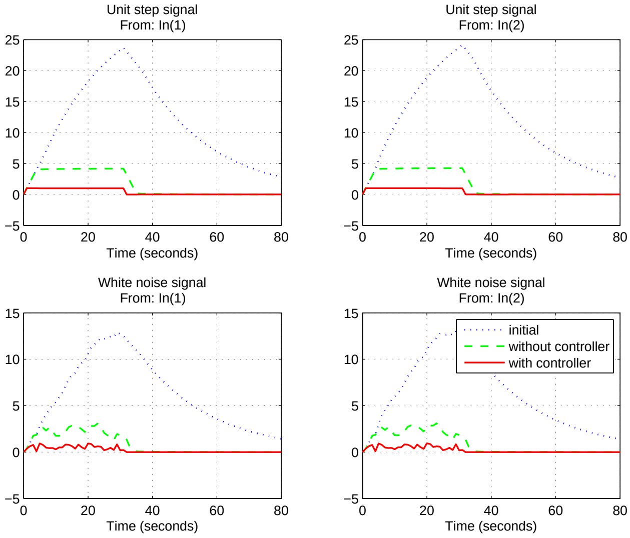

Fig. 5. Experiment 1. Ringing effects of three systems $G(\mathbf{x}^1)$ (dotted), $G(\mathbf{x}^\dagger)$ (dashed) and $T_{w\to z}(\mathbf{x}^*,\kappa^*)$ (solid). Input: Unit step signal (top) and white noise signal (bottom).

The controller K is chosen of order 2, namely

$$K(\kappa)=\frac{\kappa_1 s^2+\kappa_2 s+\kappa_3}{s^2+\kappa_4 s+\kappa_5} =\left[\begin{array}{cc|c} -\kappa_4 & \kappa_5 & 1\\ 1 & 0 & 0\\ \hline \kappa_2-\kappa_1\kappa_4 & \kappa_3-\kappa_1\kappa_5 & \kappa_1\end{array}\right] := \left[\begin{array}{c|c} A_K & B_K\\ \hline C_K & D_K\end{array}\right].$$

Then, the closed-loop transfer function of the performance channel channel $w\to z$ has the state-space representation

$$T_{w\to z}(\mathbf{x}):\begin{bmatrix}\dot{\xi}\\ z\end{bmatrix}=\begin{bmatrix}A(\mathbf{x}) & B(\mathbf{x})\\ C(\mathbf{x}) & 0\end{bmatrix}\begin{bmatrix}\xi\\ w\end{bmatrix},$$

where $\xi=[x\ x_K]^\top$, x_K the state of K, and where

$$A(\mathbf{x})=\begin{bmatrix}A+BD_KC & BC_K\\ B_KC & A_K\end{bmatrix},$$
$$B(\mathbf{x})=\begin{bmatrix}B_1+BD_KD_{21}\\ B_KD_{21}\end{bmatrix},$$
$$C(\mathbf{x})=\begin{bmatrix}C_1+D_{12}D_KC & D_{12}C_K\end{bmatrix}.$$

Assume that mass $m=4$, and spring stiffness k and damping coefficient c belong to the intervals $[4,12]$ and $[0.5,1.5]$, respectively. Using the Matlab function `hinfstruct` based on [1], we optimized H_∞-norm and obtained $k=12, c=1$ and

$$K_\infty=\frac{-6.0927s^2-0.3981s-5.1816}{s^2+19.0834s+1.1708}.$$

In the Hankel norm synthesis case, our Algorithm 1 returned $k = 12, c = 1.5$ and

$$K_H = \frac{-6.1975s^2 - 2.1828s - 4.2523}{s^2 + 19.3261s + 3.9198}.$$

Figure 6 compares step responses and white noise responses in two synthesis cases. Bearing of the algorithm is shown in Figure 7.

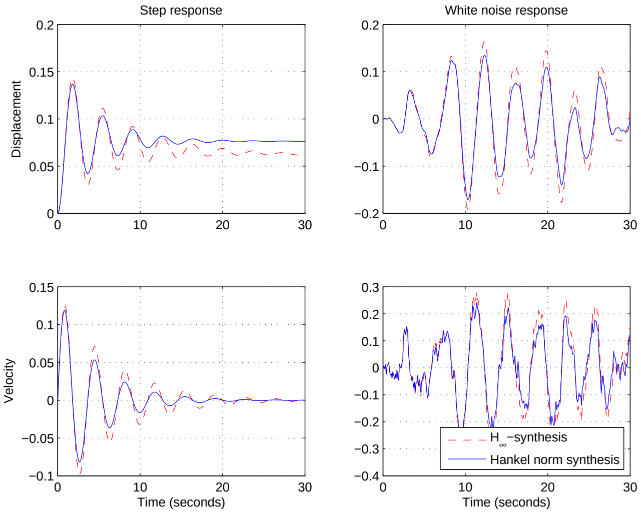

Fig. 6. Experiment 2. Step responses (left) and white noise responses (right) in two synthesis cases.

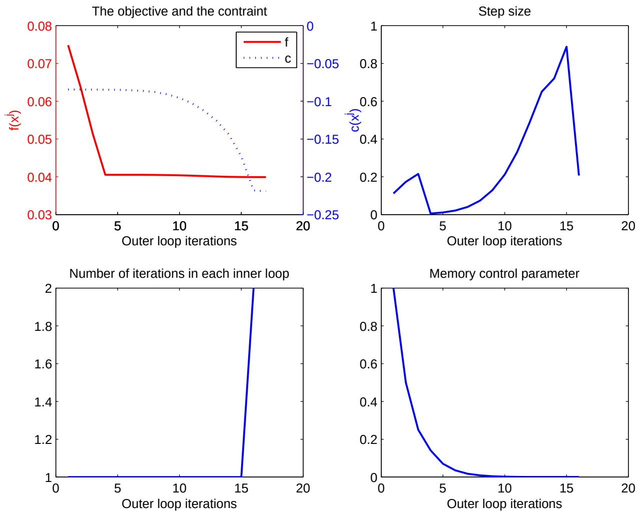

Fig. 7. Experiment 2. Bearing of the algorithm.

VIII. Conclusion

We have shown that it is possible to optimize plant and controller simultaneously if the idea of a structured control law introduced in [1] is applied. Our approach was illustrated for Hankel norm synthesis as well as for H_∞-synthesis, and for a continuous and a discrete system. Due to inherent non-smoothness of the cost functions, non-smooth optimization was applied, and in particular, a non-convex bundle method was presented. For eigenvalue optimization, as required for Hankel norm synthesis, a relaxation developed by Nesterov for the convex case was successfully used as a heuristic in the non-convex case to initialize the bundle method.

References

[1] P. Apkarian and D. Noll, "Nonsmooth H_∞ synthesis," *IEEE Trans. Automat. Control*, vol. 51, no. 1, pp. 71-86, 2006.

[2] P. Apkarian and D. Noll, "Nonsmooth optimization for multidisk H_∞ synthesis," *Eur. J. Control*, vol. 12, no. 3, pp. 229-244, 2006.

[3] J. V. Burke and M. L. Overton, "Differential properties of the spectral abscissa and the spectral radius for analytic matrix-valued mappings," *Nonlinear Anal.*, vol. 23, no. 4, pp. 467-488, 1994.

[4] F. H. Clarke, "Generalized gradients of Lipschitz functionals," *Adv. in Math.*, vol. 40, no. 1, pp. 52-67, 1981.

[5] F. H. Clarke, *Optimization and Nonsmooth Analysis*, John Wiley & Sons, Inc., New York, 1983.

[6] N. M. Dao and D. Noll, "Minimizing the memory of a system," in *Proceedings of the 9th Asian Control Conference*, Istanbul, 2013.

[7] M. Gabarrou, D. Alazard and D. Noll, "Design of a flight control architecture using a non-convex bundle method," *Math. Control Signals Systems*, vol. 25, no. 2, pp. 257-290, 2013.

[8] K. Glover, "All optimal Hankel-norm approximations of linear multivariable systems and their L^∞-error bounds," *Int. J. Control*, vol. 39, no. 6, pp. 1115-1193, 1984.

[9] Y. Nesterov, "Smoothing technique and its applications in semidefinite optimization," *Math. Program., Ser. A*, vol. 110, no. 2, pp. 245-259, 2007.

[10] D. Noll, "Cutting plane oracles to minimize non-smooth non-convex functions," *Set-Valued Var. Anal.*, vol. 18, no. 3-4, pp. 531-568, 2010.

[11] D. Noll, "Convergence of non-smooth descent methods using the Kurdyka-Łojasiewicz inequality," *J. Optim. Theory Appl.*, to appear.

[12] D. Noll, O. Prot and A. Rondepierre, "A proximity control algorithm to minimize nonsmooth and nonconvex functions," *Pac. J. Optim.*, vol. 4, no. 3, pp. 571-604, 2008.

[13] M. L. Overton, "Large-scale optimization of eigenvalues," *SIAM J. Optim.*, vol. 2, no. 1, pp. 88-120, 1992.

[14] A. M. Ostrowski, *Solutions of Equations in Euclidean and Banach Spaces*, Pure and Applied Mathematics, Vol. 9. Academic Press, New York-London, 1973.

[15] E. Polak, *Optimization: Algorithms and Consistent Approximations*, Applied Mathematical Sciences, 124. Springer-Verlag, New York, 1997.

[16] R. T. Rockafellar and R. J.-B. Wets, *Variational Analysis*, Springer-Verlag, Berlin, 1998.