Multi-Model, Multi-Objective Tuning of Fixed-Structure Controllers

Pierre Apkarian, Pascal Gahinet, and Craig Buhr

***Abstract*— We present a new technique for tuning arbitrary linear control structures against multiple plant models, multiple H_2 and H_∞ performance requirements, and additional constraints on open-loop stability and closed-loop pole locations. Our approach relies on non-smooth optimization and strikes a good balance between flexibility and effectiveness. Its capabilities are illustrated on two challenging applications.**

I. Introduction

Control theory values formulations with analytic solutions or nice numerical properties such as convexity. In the area of controller design, some of the most celebrated achievements include H_2 (LQG) synthesis [1], H_∞ synthesis [2], and formulations based on Linear Matrix Inequalities (LMIs) [3], [4]. Control engineers, on the other hand, typically face a multitude of design requirements and constraints that make it difficult to apply such mathematically elegant techniques. They often need to work with low-complexity control architectures (e.g., cascaded PID loops with low-pass filters) to facilitate implementation, validation and possibly on-site re-tuning. Their requirements may include a mix of time- and frequency-domain criteria such as settling time and overshoot, stability margins, noise or gain attenuation in prescribed frequency bands, and damping constraints on the closed-loop poles. Their requirements may pertain to different closed-loop transfers or different feedback configurations (some loops open or closed). Finally, their design ought to be robust to plant variations that may be difficult to model systematically.

There are two main ways to cope with such practical difficulties: engineering know-how and optimization. Optimization methods have the advantage of being extremely flexible. Express each requirement as an objective or a constraint, and just use nonlinear programming to search for optimal values of the design parameters. Without care, however, generic optimizers often struggle to find good designs in reasonable time. Simulation-based approaches tend to be slow, and lack of continuity, smoothness, and convexity all conspire against effectiveness. Yet, by carefully selecting the formulation and using well-adapted optimizers, it is possible to strike a reasonable balance between flexibility and effectiveness. One such example is the non-smooth optimization technique developed in [5], [6] for tuning fixed-structure control systems. By recasting the design requirements in well-posed frequency-domain terms and by using specialized non-smooth optimizers, this approach can solve many practical linear control problems in a matter of seconds.

P. Apkarian is with ONERA, 2 Av. Ed. Belin, 31055, Toulouse, France. Pierre.Apkarian@onera.fr

P. Gahinet and C. Buhr are with MathWorks, 3 Apple Hill, Natick, MA 01760-2098, USA. Pascal.Gahinet@mathworks.com

The paper is organized as follows. Section II discussed the steps involved in turning design requirements for fixed-structure control systems into a multi-model, multi-objective non-smooth program. Section III discusses the non-smooth solvers with emphasis on handling hard constraints. Finally, the last two sections present realistic applications to reliable flight control and active vibration control.

II. From Design Requirements to Non-Smooth Optimization

This section considers the problem of tuning the control elements in a linear, fixed-structure control system and summarizes the key steps involved in formulating this as a non-smooth optimization program.

The first challenge is coping with the array of possible feedback architectures and with structural constraints on the tunable elements (gains, PIDs, etc). If we separate the tunable and fixed blocks in the block-diagram representation of the control system, we can transform any control architecture into the *Standard Form* of Figure 1 where $C_1, \ldots, C_N$ are the tunable elements. This Standard Form is identical to the one used for H_∞ synthesis [2] except for the block diagonal structure of the controller. Note that tunable elements with additional structural constraints (e.g., a PID or a notch filter) can themselves by modeled as in Figure 1 with $C_1, \ldots, C_N$ now comprised of their free parameters [7].

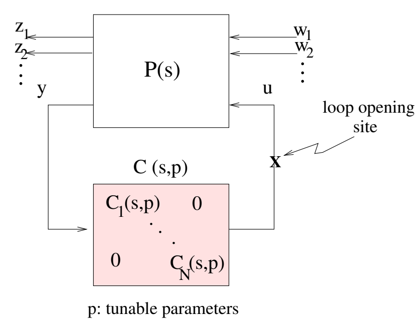

Fig. 1. Synthesis against multiple requirements

The second challenge is coping with the range of design requirements. Here we give up some generality by using a frequency-domain formulation of the control objectives. While control engineers tend to be more comfortable with time-domain specifications, we believe that the frequency-domain perspective has key advantages: it provides a more precise and comprehensive assessment of system performance, it extends nicely to MIMO systems [8], and most design objectives can be expressed in terms of simple metrics: the H_2 norm (average gain), the H_∞ norm (peak gain), and the decay rate, damping, and natural frequency of closed-loop poles. For example, tracking performance can be quantified in terms of open-loop gain and closed-loop peak gain; disturbance rejection can be quantified in terms of minimum loop gain in the rejection band; adequate transient responses can be enforced via model matching; SISO or MIMO stability margins are related to some scaled H_∞ norm [9]; noise attenuation can be quantified with the H_2 norm; etc.

The third challenge is that different requirements may be placed on different closed-loop transfer functions or different feedback configurations. For example, in a cascaded architecture, we may have requirements on the inner loop performance when the outer loop is open. We may also insist on open-loop stability to rule out unstable compensators. Such scenarios are depicted in Figure 1 by multiple $w_j \to z_j$ channels for performance assessment, as well as loop openings to model loss of feedback, e.g., due to fault. Finally, we often need to distinguish between *hard* (must-have) requirements and *soft* (nice-to-have) requirements.

The fourth challenge is coping with plant uncertainty and plant variations during operation. Plant uncertainty can be handled with μ-synthesis techniques [10], [11] when a detailed uncertainty model is available. Otherwise, a practical if less rigorous alternative consists of tuning the controller against a set of plant models representative of plant variations during operation. This approach is depicted in Figure 2 where one set of control elements must now be tuned against a family of closed-loop models.

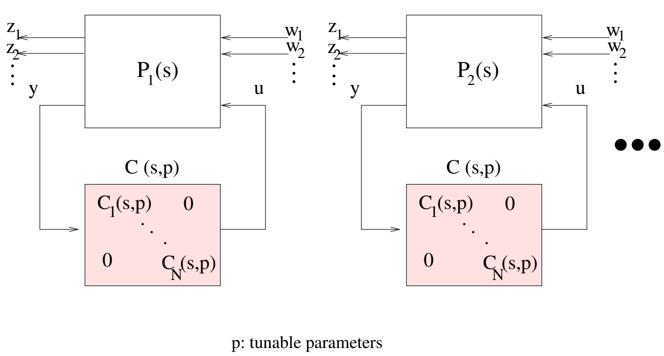

Fig. 2. Synthesis against multiple requirements and models

Addressing these various challenges naturally leads to an optimization problem of the form

$$\begin{array}{ll} \underset{p}{\text{minimize}} & \max_{i,k} \left\{ \|T^{(k)}_{w_i \to z_i}(C(s,p))\| \right\} \\ \text{subject to} & \max_{j,k} \left\{ \|T^{(k)}_{w_j \to z_j}(C(s,p))\| \right\} \leq 1\,. \end{array} \quad (1)$$

where p is the vector of tunable parameters, $T^{(k)}_{w \to z}$ denotes the closed-loop map from signal w to signal z for the k-th plant model, and $\|.\|$ denotes either the H_∞ or the H_2 norm, possibly restricted to prescribed frequency intervals. This seeks to minimize the worst-case value of the soft requirements $\|T^{(k)}_{w_i \to z_i}\|$ while enforcing the hard requirements $\|T^{(k)}_{w_j \to z_j}\|$. Note that all terms should be normalized for this formulation to make sense. Also, we dropped terms associated with requirements on pole location for notational simplicity.

Problem (1) is nonlinear, non-convex, and non-smooth because the H_∞ norm is a maximum over frequency. It cannot be tackled with LMI methods without overly conservative relaxations. And conventional constrained minimization [12] tends to stall when differentiability is lost because the peak gain is achieved at two or more frequencies. Specialized non-smooth algorithms are therefore needed. Surprisingly, such algorithms perform quite well in practice, both in terms of execution speed and quality of the solutions [11], [13].

III. NON-SMOOTH SOLVER

Design problems involving requirements of different nature as well as multiple models can be formalized through the general program

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & g(x) \leq 1, \end{array} \quad (2)$$

where $x \in \mathbb{R}^n$ is the decision vector consisting of tunable parameters in the (structured) controller. The functions f and g capture requirements of different nature over a family of models. Each of these requirements is referred to as f_i and g_j for simplicity, and we define requirement aggregates using $\max$ operations as follows

$$f(x) := \max_{i=1,\ldots,n_f} f_i(x), \quad g(x) := \max_{i=1,\ldots,n_g} g_j(x)\,. \quad (3)$$

We also assume that all terms have been adequately normalized so that program (2) makes sense. Weighted sums of squares is a conventional way to combine multiple objectives. However, the $\max$ formulation (3) offers advantages in terms of pruning non-contributing terms, which translates into substantial computational savings. No matter the number of constraints, only nearly active constraints are relevant when solving (2). We say an objective f_{i_a} is nearly active at x whenever $f_{i_a}(x) \geq (1-\kappa) f(x)$, and similarly for constraints g_j's. In our numerical implementation, a typical value for the threshold κ is 0.2 which leads to significant speedup in the early iterations of the non-smooth solver.

The challenges in solving (2) are two-fold. The $\max$ aggregates are non-smooth by construction and some of the components f_i and g_j themselves are non-smooth (H_∞ norm and pole clustering constraints). In addition, the aggregates f and g are non-convex for structurally constrained controllers. So (2) is a non-smooth and non-convex program. For the rich set of control design requirements considered here, the properties of the functions f_i and g_j are well-known [5]. They are Lipschitz and even Clarke regular [14]. An immediate consequence is that Clarke's sub-differentials of f and g are easily computed using convex hull operations over sub-gradients. First-order information is therefore easily accessible to build a specialized and thus highly efficient non-smooth solver.

It remains to discuss how constrained minimization in (2) is addressed. This again remains challenging since constrained non-smooth programming is by no means as well advanced as the unconstrained case. One straightforward option is to combine objectives f and constraints g using the concept of barrier functions. Very often, logarithmic or reciprocal barriers are used to enforce feasibility of constraints along iterations [12], [15]. This is not entirely satisfactory as these techniques suffer from numerical problems when barrier parameters are driven to their limits. Also, they lead to two-phase algorithms in which feasibility is achieved in phase 1 and objective minimization is carried out in phase 2. This is often inefficient since non-smooth constraint boundaries strongly restrict displacements in the search space. The progress function approach introduced in [16] and further explored for control design in [17] overcomes the limitations due to ill-conditioning but it is again a two-phase algorithm for which slow progress is often observed in phase 2. Exact penalty functions [12], [18] somewhat remedy this difficulties. They are of the form

$$p_c(x) = f(x) + c \max(g(x) - 1,\, 0)\,, \tag{4}$$

and for large enough values of the penalty parameter c, local solutions of (2) can be computed by minimizing (4), hence the name exact penalty. However, schemes for selecting adequate values of c can be complex and breakdowns may occur when c becomes very large. Note admissible values for c satisfy $c \geq \lambda^*$, where λ^* is a Lagrange multiplier associated to the Karush-Kuhn-Tucker (KKT) conditions for (2) [12]. Our implementation of the non-smooth solver relies on the related objective function

$$\Phi_\eta(x) := \max\{f(x), \eta g(x)\} \tag{5}$$

and uses a Lagrangian method to adjust η and locally solve the KKT conditions for (2).

The basic principle of this method is as follows (see [19] for a more detailed treatment). If constraints g are not competing with f then we are left with simply minimizing f alone. Otherwise, we infer that constraints g should be active at a local solution, i.e, $g(x^*) = 1$. Solutions x^* are then obtained by minimizing Φ_η for a sequence of η values that lead to saturating the constraint $g(x) \leq 1$. More precisely, η is adjusted by a bisection scheme that increases or decreases η based on constraint feasibility (η is increased when the constraint g is violated and vice versa). The subproblem of minimizing Φ_η for a given η is tackled with the unconstrained algorithm developed in [5] and originally implemented in `hinfstruct` [11]. This algorithm has solid local convergence properties and performs well in practice [13].

This basic principle admits many refinements beyond the scope of this paper. In particular, the aggregate (5) lends itself to pruning of non-active f_i, g_j terms which translates into considerable speedup when dealing with multiple requirements on multiple models. Note that this technique is of exterior type since boundary crossing is allowed and thus potentially larger steps are performed at every iteration. Finally, as with any local method, it is advisable to use several runs with different initial points to weed out unsatisfactory local solutions. The method described above is the core of the `systune` solver in [11].

IV. Fault-tolerant Control

Our first example is an application of multi-model, fixed-structure tuning to reliable flight control. The flight control system is required to maintain stability and adequate performance in both nominal operation and in situations when the aircraft undergoes outages in the elevator and aileron actuators. In particular, wind gusts must be alleviated in all outage scenarios to maintain safety.

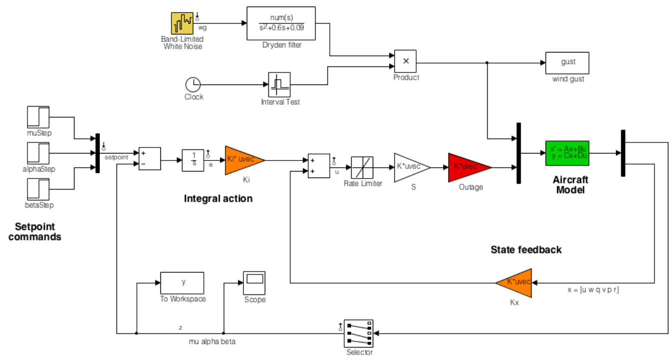

Fig. 3. Synthesis interconnection

The control system is depicted in Fig. 3. The aircraft is modeled as a 6th-order state-space system with body velocities u, v, w and pitch, roll, and yaw rates q, p, r. The state vector is available for control as well as the flight-path bank angle rate μ (deg/s), angle of attack α (deg), and sideslip angle β (deg). The control inputs are the deflections of the right elevator, left elevator, right aileron, left aileron, and rudder. The controller consists of a 3×6 state-feedback gain matrix K_x in the inner loop and a 3×3 integral gain matrix K_i in the outer loop, a total of 27 parameters to tune.

In addition to nominal operation, we consider 8 outage scenarios modeled as a 5×5 diagonal "outage gain" at the aircraft input and summarized in Table I.

TABLE I

Outage scenarios where 0 stands for failure

Outage cases	Diagonal of outage gain				
nominal mode	1	1	1	1	1
right elevator outage	0	1	1	1	1
left elevator outage	1	0	1	1	1
right aileron outage	1	1	0	1	1
left aileron outage	1	1	1	0	1
left elevator and right aileron outage	1	0	0	1	1
right elevator and right aileron outage	0	1	0	1	1
right elevator and left aileron outage	0	1	1	0	1
left elevator and left aileron outage	1	0	1	0	1

The design requirements are as follows:

- Good tracking performance in μ, α, and β with adequate decoupling of the three axes.
- Adequate rejection of wind gusts of 5 m/s.
- Maintain stability and acceptable performance in the face of actuator outage.

The tracking requirement is expressed as an LQG-like cost function that penalizes the integrated tracking error e and the control effort u:

$$J = \lim_{T\to\infty} E\left(\frac{1}{T}\int_0^T \|W_e e\|^2 + \|W_u u\|^2 dt\right). \quad (6)$$

The diagonal weights W_e and W_u provide tuning knobs for trading responsiveness and control effort and balancing the three channels. We use $W_e = \mathrm{diag}(20, 30, 20), W_u = I_3$ for normal operation and $W_e = \mathrm{diag}(8, 12, 8), W_u = I_3$ for outage conditions.

The gust alleviation requirement is treated as a hard constraint limiting the variance of the error signal e due to white noise w_g driving the Dryden wind gust model. Specifically, the variance of e is limited to 0.01 for normal operation and to 0.03 for the outage scenarios.

With the notation of section III, the functions $f(x)$ and $g(x)$ in (2) are given by $f(x) := \max_{i=1,\ldots,9} f_i(x)$ and $g(x) := \max_{i=1,\ldots,9} g_i(x)$, where x is the vector comprised of the entries of K_i and K_x, i is the scenario index, the f_i's are the square roots of J in (6) with appropriate weightings W_e and W_u, and the g_i's are the RMS values of e suitably weighted to reflect variance bounds of 0.01 and 0.03. Note that all f_i and g_i terms measure the H_2 norm of some closed-loop transfer function and are covered by the `Variance` and `WeightedVariance` requirements in [11].

With this setup, we tuned the controller gains K_i and K_x for the nominal scenario only (*nominal design*) and for all 9 scenarios (*fault-tolerant design*). The responses to setpoint changes in μ, α, and β with a gust speed of $5\,m/s$ are shown in Fig. 4 for the nominal design and in Fig. 5 for the fault-tolerant design. As expected, nominal responses are good but noticeably deteriorate when faced with outages. By contrast, the fault-tolerant controller maintains acceptable performance in outage situations. The optimal performance (square root of LQG cost J in (6)) for the fault-tolerant design is only slightly worse than for the nominal design (26 vs. 23). The non-smooth program (2) was solved with `systune` and the fault-tolerant design (9 models, 11 states, 27 parameters) took 30 seconds on Mac OS X with 2.66 GHz Intel Core i7 and 8 GB RAM. See [11] for the model data and additional details.

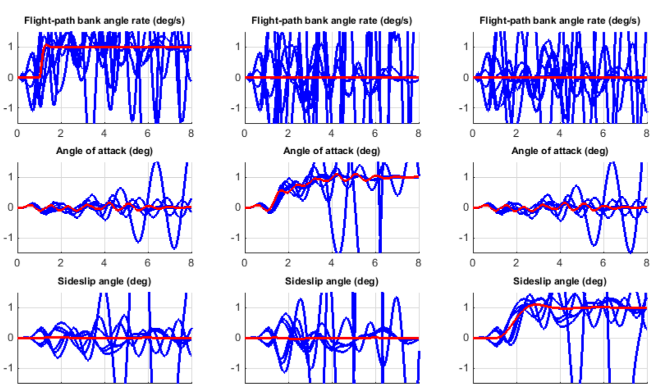

Fig. 4. Responses to step changes in μ, α and β for nominal design.

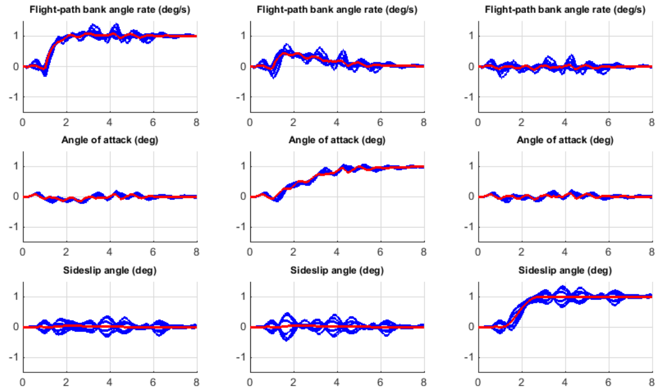

Fig. 5. Responses to step changes in μ, α and β for fault-tolerant design.

V. ACTIVE VIBRATION CONTROL IN THREE-STORY BUILDING

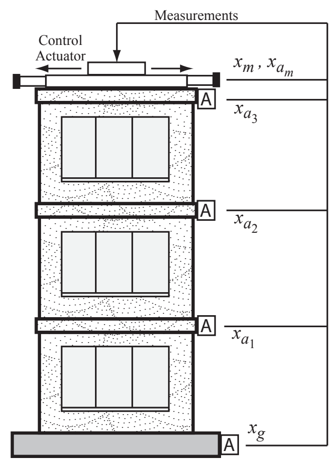

Fig. 6. Active mass driver control system

In this example, we consider an Active Mass Driver (AMD) control system for vibration isolation in a three-story experimental structure. This setup is used to assess control design techniques for increasing safety of civil engineering structures during earthquakes. The structure consists of three stories with an active mass driver on the top floor which is used to attenuate ground disturbances. This application is borrowed from [20] where a 28-state scale model of the building including actuator and sensors was derived from experimental data (see related example in [11] for data). The relevant states for control purpose are shown in Table II. The inputs are the ground acceleration x_{ag} (in g) and the control signal u fed to the actuator. The actuator generates left and right motions of the mass to attenuate ground disturbances. The earthquake acceleration is modeled as a white noise process filtered through a Kanai-Tajimi filter [20], see Figure 7. Bode plots of the transfer functions from control signal u and ground acceleration x_{ag} to the first floor acceleration $x_a(1)$ are shown in Figure 8. The building features a number of structural flexibilities with dominant peaks at 5.80, 17.67 and 28.53 Hz and associated damping 0.3%, 0.23% and

0.30%. Such structural modes may incur serious damage when excited by ground disturbances.

TABLE II

RELEVANT STATES FOR THREE-STORY BUILDING

$x(i)$	displacement of i-th floor relative to the ground (cm)
x_m	displacement of AMD relative to 3rd floor (cm)
$x_v(i)$	velocity of i-th floor relative to the ground (cm/s)
$x_{vm}(i)$	velocity of AMD relative to the ground (cm/s)
$x_a(i)$	acceleration of i-th floor relative to the ground (g)
x_{am}	acceleration of AMD relative to the ground (g)
$d(i)$	$d(1) = x(1),\ d(2) = x(2) - x(1),$ $d(3) = x(3) - x(2),$ inter-story drifts

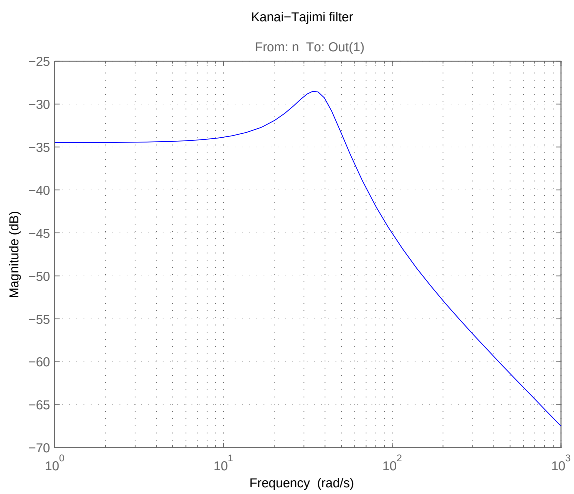

Fig. 7. Bode magnitude of Kanai-Tajimi filter

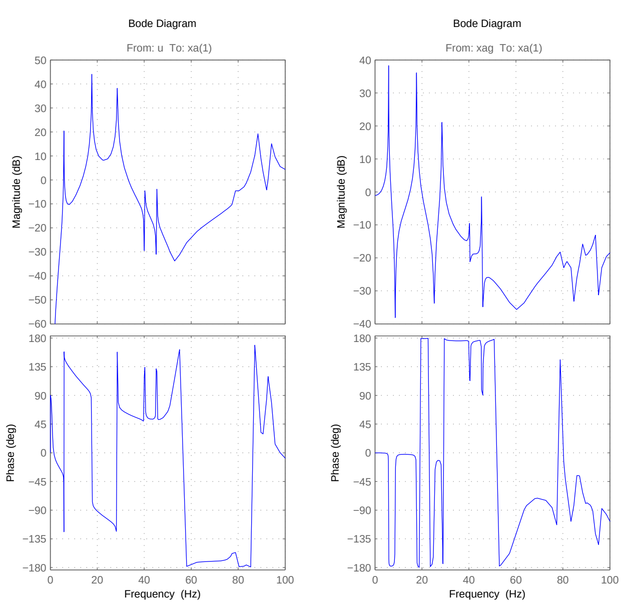

Fig. 8. Bode response from control (left) and ground acceleration (right) to 1st floor acceleration

The open-loop standard deviations of drifts and accelerations in response to white noise colored by the Kanai-Tajimi filter are displayed as the blue bars in the bar plot of Figure 9. Observe that ground disturbances generate large drifts for the first floor and significant accelerations for both the second and third floors which need to be reduced using feedback.

The controller uses four measurements of the accelerations x_a and x_{am} to generate the control signal u. Physically, the control u is an electrical current driving an hydraulic actuator that moves the masses of the AMD. The design requirements include:

- Minimization of the inter-story drifts $d(i)$ and accelerations $x_a(i)$,
- Hard constraints on control effort in terms of mass displacement x_m, mass acceleration x_{am}, and control effort u.

All design requirements are assessed in terms of standard deviations of the corresponding signals. Each variable is scaled by its open-loop standard deviation to achieve uniform relative improvement in all variables. The design problem is expressed as a constrained non-smooth program (2) where f and g are defined as

$$f(x) := \max\{ \max_{i=1,2,3} \frac{\sigma_{d(i)}}{\sigma_{d_0(i)}},\ \max_{i=1,2,3} \frac{\sigma_{x_a(i)}}{\sigma_{x_{a0}(i)}} \},$$

and

$$g(x) := \max\{ \frac{\sigma_{x_m}}{3},\ \frac{\sigma_{x_{am}}}{2},\ \frac{\sigma_u}{1} \}$$

where $\sigma_{d_0(i)}$ and $\sigma_{x_{a0}(i)}$ are the open-loop standard deviations of the drifts and accelerations for each floor.

As mentioned before, the controller complexity is a design parameter in our approach and we can therefore adjust it by trial-and-error, starting with sufficiently high order to gauge the limits of performance, then reducing the order until a noticeable performance degradation is observed. In this example, a 5th-order controller with no feedthrough term was found sufficient. This controller was optimized with `systune` in 11 seconds on a Mac OS X with 2.66 GHz Intel Core i7 and 8 GB RAM. An overall reduction of 40% in standard deviations was achieved while meeting all hard constraints. Figure 9 compares the resulting open- and closed-loop standard deviations. Finally, we simulated the response of the three-story structure to an earthquake-like excitation in both open and closed loop. The earthquake acceleration is modeled as before as a white noise process colored by the Kanai-Tajimi filter. Simulations results appear in Figures 10 and 11. More details on this example can be found in [11].

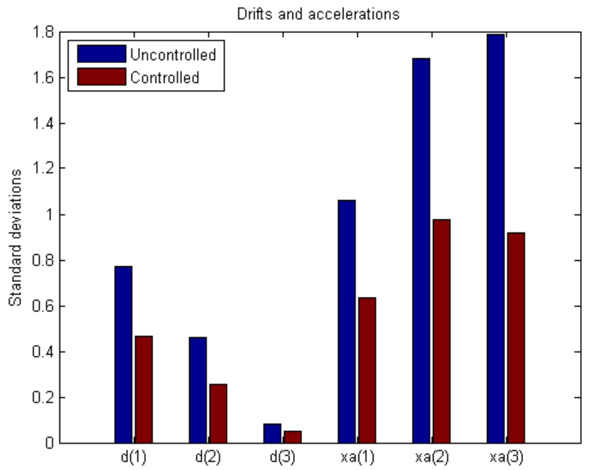

Fig. 9. Drifts and accelerations for uncontrolled and controlled structure

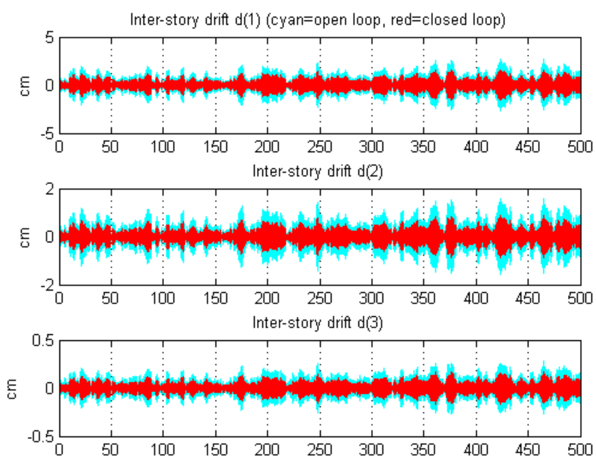

Fig. 10. Inter-story drifts in open and closed loop

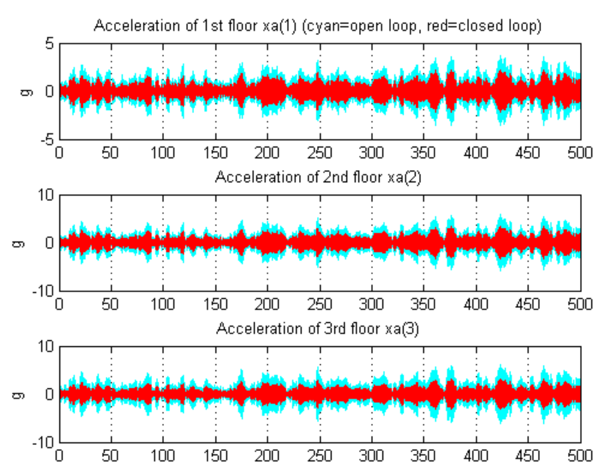

Fig. 11. Accelerations in open and closed loop

Conclusion

We have presented a non-smooth programming technique for solving control problems with realistic sets of requirements, constraints on the controller structure, and robustness goals. A core ingredient for tackling problems with soft and hard requirements is the use of a driving function whose critical points are KKT points of the original problem. This can be implemented very efficiently by exploiting basic sub-differential properties of $\max$ functions. This technique has been fully implemented in the `systune` software and proven effective for a wide range of applications. We believe this tool will help control engineers leverage the full power of frequency-domain design techniques within the practical constraints of their application.

References

[1] J. M. Maciejowski, *Multivariable Feedback Design*. Addison-Wesley, 1989.

[2] J. Doyle, K. Glover, P. Khargonekar, and B. A. Francis, "State-space solutions to standard H_2 and H_∞ control problems."

[3] S. Boyd, L. ElGhaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in Systems and Control Theory*, ser. SIAM Studies in Applied Mathematics. Philadelphia: SIAM, 1994, vol. 15.

[4] C. Scherer, "Multi-objective control without Youla parameterization," in *Perspectives in robust control*, ser. Lecture Notes in Control and Information Sciences, S. O. Moheimani, Ed. Springer Berlin / Heidelberg, 2001, vol. 268, pp. 311–325.

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

[6] J. V. Burke, D. Henrion, A. S. Lewis, and M. L. Overton, "HIFOO - a MATLAB package for fixed-order controller design and H_∞ optimization," in *5th IFAC Symposium on Robust Control Design*, Toulouse, France, July 2006.

[7] P. Gahinet and P. Apkarian, "Decentralized and fixed-structure H_∞ control in MATLAB," in *Proc. IEEE Conf. on Decision and Control*, dec. 2011, pp. 8205 –8210.

[8] S. Skogestad and I. Postlethwaite, *Multivariable Feedback Control - Analysis and Design*. Wiley, 1996.

[9] J. D. Blight, R. L. Dailey, and D. Gangsaas, "Practical control law design for aircraft using multivariable techniques," *Int. J. Control*, vol. 59, no. 1, pp. 93–137, 1994.

[10] G. Stein and J. Doyle, "Beyond singular values and loopshapes," *AIAA Journal of Guidance and Control*, vol. 14, pp. 5–16, 1991.

[11] *Robust Control Toolbox 5.0*. MathWorks, Natick, MA, USA, Sept 2013.

[12] R. Fletcher, *Practical Methods of Optimization*. John Wiley & Sons, 1987.

[13] P. Apkarian, "Internet pages," http://pierre.apkarian.free.fr, 2013.

[14] F. H. Clarke, *Optimization and Nonsmooth Analysis*, ser. Canadian Math. Soc. Series. New York: John Wiley & Sons, 1983.

[15] D. P. Bertsekas, *Nonlinear Programming*. Belmont, Mass.: Athena Scientific, USA, 1995.

[16] E. Polak, *Optimization : Algorithms and Consistent Approximations*. Applied Mathematical Sciences, 1997.

[17] P. Apkarian, D. Noll, and A. Rondepierre, "Mixed H_2/H_∞ control via nonsmooth optimization," *SIAM J. on Control and Optimization*, vol. 47, no. 3, pp. 1516–1546, 2008.

[18] D. P. Bertsekas, *Constrained optimization and Lagrange multiplier methods*. Academic Press, London, 1982.

[19] P. Apkarian and D. Noll, "Nonsmooth optimization for multiband frequency domain control design," *Automatica*, vol. 43, no. 4, pp. 724–731, April 2007.

[20] B. F. Spencer, S. J. Dyke, and H. S. Deoskar, "Benchmark problems in structural control: part 1 - active mass driver system," *Earthquake Engineering & Structural Dynamics*, vol. 27, no. 11, pp. 1127–1139, 1998.