# Optimization-Based Control Design Techniques and Tools

Pierre Apkarian[1], Dominikus Noll[2]
[1]*ONERA - The French Aerospace Lab, Toulouse, France*
[2]*Université de Toulouse, Institut de Mathématiques, Toulouse, France*
*Pierre.Apkarian@onera.fr, noll@mip.ups-tlse.fr*

## Abstract

Structured output feedback controller synthesis is an exciting new concept in modern control design, which bridges between theory and practice in so far as it allows for the first time to apply sophisticated mathematical design paradigms like $H_\infty$- or $H_2$-control within control architectures preferred by practitioners. The new approach to structured $H_\infty$-control, developed during the past decade, is rooted in a change of paradigm in the synthesis algorithms. Structured design may no longer be based on solving algebraic Riccati equations or matrix inequalities. Instead, optimization-based design techniques are required. In this essay we indicate why structured controller synthesis is central in modern control engineering. We explain why non-smooth optimization techniques are needed to compute structured control laws, and we point to software tools which enable practitioners to use these new tools in high technology applications.

## Keywords and Phrases

Controller tuning, $H_\infty$ synthesis, multi-objective design, nonsmooth optimization, structured controllers, robust control

## Introduction

In the modern high technology field control engineers usually face a large variety of concurring design specifications such as noise or gain attenuation in prescribed frequency bands, damping, decoupling, constraints on settling- or rise-time, and much else. In addition, as plant models are generally only approximations of the true system dynamics, control laws have to be robust with respect to uncertainty in physical parameters or with regard to un-modeled high frequency phenomena. Not surprisingly, such a plethora of constraints presents a major challenge for controller tuning, due not only to the ever growing number of such constraints, but also because of their very different provenience.

The dramatic increase in plant complexity is exacerbated by the desire that regulators should be as simple as possible, easy to understand and to tune by practitioners, convenient to hardware implement, and generally available at low cost. Such practical constraints explain the limited use of black-box controllers, and they are the driving force for the implementation of *structured* control architectures, as well as for the tendency to replace hand-tuning methods by rigorous algorithmic optimization tools.

## 1 Structured Controllers

Before addressing specific optimization techniques, we introduce some basic terminology for control design problems with structured controllers. A state-space description of the given $P$ used for design is given as

$$P:\left\{\begin{array}{rcl} \dot{x}_P &=& Ax_P + B_1w + B_2u \\ z &=& C_1x_P + D_{11}w + D_{12}u \\ y &=& C_2x_P + D_{21}w + D_{22}u \end{array}\right. \qquad (1)$$

where $A$, $B_1$, ... are real matrices of appropriate dimensions, $x_P \in \mathbb{R}^{n_P}$ is the state, $u \in \mathbb{R}^{n_u}$ the control, $y \in \mathbb{R}^{n_y}$ the measured output, $w \in \mathbb{R}^{n_w}$ the exogenous input, and $z \in \mathbb{R}^{n_z}$ the regulated output. Similarly, the sought output feedback controller $K$ is described as

$$K:\left\{\begin{array}{rcl} \dot{x}_K &=& A_Kx_K + B_Ky \\ u &=& C_Kx_K + D_Ky \end{array}\right. \qquad (2)$$

with $x_K \in \mathbb{R}^{n_K}$, and is called *structured* if the (real) matrices $A_K, B_K, C_K, D_K$ depend smoothly on a design parameter $\mathbf{x} \in \mathbb{R}^n$, referred to as the vector of tunable parameters. Formally, we have differentiable mappings

$$A_K = A_K(\mathbf{x}), B_K = B_K(\mathbf{x}), C_K = C_K(\mathbf{x}), D_K = D_K(\mathbf{x}),$$

and we abbreviate these by the notation $K(\mathbf{x})$ for short to emphasize that the controller is structured with $\mathbf{x}$ as tunable elements. A structured controller synthesis

problem is then an optimization problem of the form

$$\begin{array}{ll} \text{minimize} & \|T_{wz}(P,K(\mathbf{x}))\| \\ \text{subject to} & K(\mathbf{x}) \text{ closed-loop stabilizing} \\ & K(\mathbf{x}) \text{ structured, } \mathbf{x}\in\mathbb{R}^n \end{array} \qquad (3)$$

where $T_{wz}(P,K) = \mathcal{F}_\ell(P,K)$ is the lower feedback connection of (1) with (2) as in Fig. 1 (left), also called the Linear Fractional Transformation [Varga and Looye, 1999]. The norm $\|\cdot\|$ stands for the $H_\infty$-norm, the $H_2$-norm, or any other system norm, while the optimization variable $\mathbf{x}\in\mathbb{R}^n$ regroups the tunable parameters in the design.

Standard examples of structured controllers $K(\mathbf{x})$ include realizable PIDs, observer-based, reduced-order, or decentralized controllers, which in state-space are expressed as:

$$\left[\begin{array}{cc|c} 0 & 0 & 1 \\ 0 & -1/\tau & -k_D/\tau \\ \hline k_I & 1/\tau & k_P+k_D/\tau \end{array}\right], \left[\begin{array}{c|c} A-B_2K_c-K_fC_2 & K_f \\ \hline -K_c & 0 \end{array}\right],$$

$$\left[\begin{array}{c|c} A_K & B_K \\ \hline C_K & D_K \end{array}\right], \left[\begin{array}{c|c} \operatorname*{diag}_{i=1}^{q} A_{Ki} & \operatorname*{diag}_{i=1}^{q} B_{Ki} \\ \hline \operatorname*{diag}_{i=1}^{q} C_{Ki} & \operatorname*{diag}_{i=1}^{q} D_{Ki} \end{array}\right].$$

In the case of a PID the tunable parameters are $\mathbf{x} = (\tau, k_P, k_I, k_D)$, for observer-based controllers $\mathbf{x}$ regroups the estimator and state-feedback gains $(K_f, K_c)$, for reduced order controllers $n_K < n_P$ the tunable parameters $\mathbf{x}$ are the $n_K^2 + n_K n_y + n_K n_u + n_y n_u$ unknown entries in $(A_K, B_K, C_K, D_K)$, and in the decentralized form $\mathbf{x}$ regroups the unknown entries in $A_{K1}, \ldots, D_{Kq}$. In contrast, full-order controllers have the maximum number $N = n_P^2 + n_P n_y + n_P n_u + n_y n_u$ of degrees of freedom and are referred to as unstructured or as *black-box* controllers.

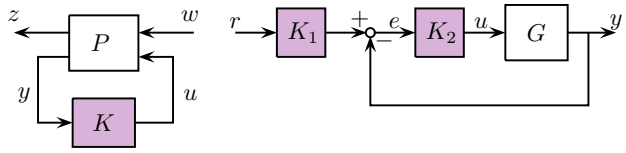


Figure 1: Black-box full-order controller $K$ on the left, structured 2-DOF control architecture with $K = \text{block-diag}(K_1, K_2)$ on the right.

More sophisticated controller structures $K(\mathbf{x})$ arise form architectures like for instance a 2-DOF control arrangement with feedback block $K_2$ and a set-point filter $K_1$ as in Fig. 1 (right). Suppose $K_1$ is the 1st-order filter $K_1(s) = a/(s+a)$ and $K_2$ the PI feedback $K_2(s) = k_P + k_I/s$. Then the transfer $T_{ry}$ from $r$ to $y$ can be represented as the feedback connection of $P$ and $K(\mathbf{x})$ with

$$P := \left[\begin{array}{c|ccc} A & 0 & 0 & B \\ \hline C & 0 & 0 & D \\ 0 & I & 0 & 0 \\ -C & 0 & I & -D \end{array}\right], K(\mathbf{x}) := \left[\begin{array}{cc} K_1(s) & 0 \\ 0 & K_2(s) \end{array}\right],$$

where $K(\mathbf{x},s)$ takes a typical block-diagonal structure featuring the tunable elements $\mathbf{x} = (a, k_P, k_I)$.

In much the same way arbitrary multi-loop interconnections of fixed-model elements with tunable controller blocks $K_i(\mathbf{x})$ can be re-arranged as in Fig. 2, so that $K(\mathbf{x})$ captures all tunable blocks in a decentralized structure general enough to cover most engineering applications.

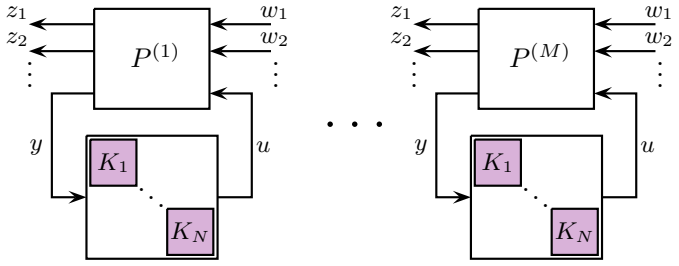


Figure 2: Synthesis of $K = \text{block-diag}(K_1, \ldots, K_N)$ against multiple requirements or models $P^{(1)}, \ldots, P^{(M)}$. Each $K_i(\mathbf{x})$ can be structured.

The structure concept is equally useful to deal with the second central challenge in control design: *system uncertainty*. The latter may be handled with $\mu$-synthesis techniques [Stein and Doyle, 1991] if a parametric uncertain model is available. A less ambitious but often more practical alternative consists in optimizing the structured controller $K(\mathbf{x})$ against a finite set of plants $P^{(1)}, \ldots, P^{(M)}$ representing model variations due to uncertainty, aging, sensor and actuator breakdown, un-modeled dynamics, in tandem with the robustness and performance specifications. This is again formally covered by Fig. 2 and leads to a multi-objective constrained optimization problem of the form

$$\begin{array}{ll} \text{minimize} & f(\mathbf{x}) = \max\limits_{k\in\text{SOFT}, i\in I_k} \|T_{w_i z_i}^{(k)}(K(\mathbf{x}))\| \\ \text{subject to} & g(\mathbf{x}) = \max\limits_{k\in\text{HARD}, j\in J_k} \|T_{w_j z_j}^{(k)}(K(\mathbf{x}))\| \le 1 \\ & K(\mathbf{x}) \text{ structured and stabilizing} \\ & \mathbf{x}\in\mathbb{R}^n \end{array} \qquad (4)$$

where $T_{w_i z_i}^{(k)}$ denotes the $i$th closed-loop robustness or performance channel $w_i \to z_i$ for the $k$-th plant model $P^{(k)}(s)$. The rationale of (4) is to minimize the worst-case cost of the soft constraints $\|T_{w_i z_i}^{(k)}\|$, $k \in \text{SOFT}$,

while enforcing the hard constraints $\|T^{(k)}_{w_j z_j}\| \leq 1$, $k \in$ HARD. Note that in the mathematical programming terminology, soft and hard constraints are classically referred to as objectives and constraints. The terms soft and hard point to the fact that hard constraints prevail over soft ones and that meeting hard constraints for solution candidates is mandatory.

## 2 Optimization Techniques Over the Years

During the late 1990s the necessity to develop design techniques for structured regulators $K(\mathbf{x})$ was recognized [Fares et al, 2001], and the limitations of synthesis methods based on algebraic Riccati equations (AREs) or linear matrix inequalities (LMIs) became evident, as these techniques can only provide black-box controllers. The lack of appropriate synthesis techniques for structured $K(\mathbf{x})$ led to the unfortunate situation, where sophisticated approaches like the $H_\infty$ paradigm developed by academia since the 1980s could not be brought to work for the design of those controller structures $K(\mathbf{x})$ preferred by practitioners. Design engineers had to continue to rely on heuristic and ad-hoc tuning techniques, with only limited scope and reliability. As an example: post-processing to reduce a black-box controller to a practical size is prone to failure. It may at best be considered a fill-in for a rigorous design method which directly computes a reduced-order controller. Similarly, hand-tuning of the parameters $\mathbf{x}$ remains a puzzling task because of the loop interactions, and fails as soon as complexity increases.

In the late 1990s and early 2000s, a change of methods was observed. Structured $H_2$- and $H_\infty$-synthesis problems (3) were addressed by bilinear matrix inequality (BMI) optimization, which used local optimization techniques based on the augmented Lagrangian method [Fares et al, 2001; Noll et al, 2002; Kocvara and Stingl, 2003], sequential semidefinite programming methods [Fares et al, 2002; Apkarian et al, 2003], and non-smooth methods for BMIs [Noll et al, 2009; Lemaréchal and Oustry, 2000]. However, these techniques were based on the bounded real lemma or similar matrix inequalities, and were therefore of limited success due to the presence of Lyapunov variables, i.e. matrix-valued unknowns, whose dimension grows quadratically in $n_P + n_K$ and represents the bottleneck of that approach.

The epoch-making change occurs with the introduction of non-smooth optimization techniques [Noll and Apkarian, 2005; Apkarian and Noll, 2006b, 2007, 2006c] to programs (3) and (4). Today non-smooth methods have superseded matrix inequality-based techniques and may be considered the state-of-art as far as realistic applications are concerned. The transition took almost a decade.

Alternative control-related local optimization techniques and heuristics include the gradient sampling technique of [Burke et al, 2005], derivative-free optimization discussed in [Kolda et al, 2003; Apkarian and Noll, 2006a], particle swarm optimization, see [Oi et al, 2008] and references therein, and also evolutionary computation techniques [Lieslehto, 2001]. The last three classes do not exploit derivative information and rely on function evaluations only. They are therefore applicable to a broad variety of problems including those where function values arise from complex numerical simulations. The combinatorial nature of these techniques, however, limits their use to small problems with a few tens of variable. More significantly, these methods often lack a solid convergence theory. In contrast, as we have demonstrated over recent years, [Apkarian and Noll, 2006b; Noll et al, 2008] specialized non-smooth techniques are highly efficient in practice, are based on a sophisticated convergence theory, capable of solving medium size problems in a matter of seconds, and are still operational for large size problems with several hundreds of states.

## 3 Non-smooth optimization techniques

The benefit of the non-smooth casts (3) and (4) lies in the possibility to avoid searching for Lyapunov variables, a major advantage as their number $(n_P + n_K)^2/2$ usually largely dominates $n$, the number of true decision parameters $\mathbf{x}$. Lyapunov variables do still occur implicitly in the function evaluation procedures, but this has no harmful effect for systems up to several hundred states. In abstract terms, a non-smooth optimization program has the form

$$\begin{array}{ll} \text{minimize} & f(\mathbf{x}) \\ \text{subject to} & g(\mathbf{x}) \leq 0 \\ & \mathbf{x} \in \mathbb{R}^n \end{array} \tag{5}$$

where $f, g : \mathbb{R}^n \to \mathbb{R}$ are locally Lipschitz functions and are easily identified from the cast in (4).

In the realm of convex optimization, non-smooth programs are conveniently addressed by so-called bundle methods, introduced in the late 1970s by Lemaréchal [Lemarechal, 1975]. Bundle methods are used to solve difficult problems in integer programming or in stochastic optimization via Lagrangian re-

laxation. Extensions of the bundling technique to non-convex problems like (3) or (4) were first developed in [Apkarian and Noll, 2006b, 2007, 2006c; Apkarian et al, 2008; Noll et al, 2009], and in more abstract form, in [Noll et al, 2008].

Fig. 3 shows a schematic view of a non-convex bundle method consisting of a descent-step generating inner loop (yellow block) comparable to a line search in smooth optimization, embedded into the outer loop (blue box), where serious iterates are processed, stopping criteria are applied, and the model tradition is assured. At the core of the interaction between inner and outer loop is the management of the proximity control parameter $\tau$, which governs the stepsize $\|\mathbf{x}-\mathbf{y}^k\|$ between trial steps $\mathbf{y}^k$ at the current serious iterate $\mathbf{x}$. Similar to the management of a trust region radius or of the stepsize in a linesearch, proximity control allows to force shorter trial steps if agreement of the local model with the true objective function is poor, and allows larger steps if agreement is satisfactory.

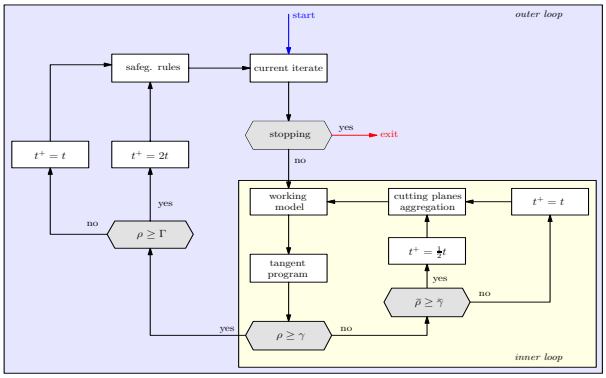

Figure 3: Flow chart of proximity control bundle algorithm

Oracle-based bundle methods traditionally assure global convergence in the sense of subsequences under the sole hypothesis that for every trial point $\mathbf{x}$ the function value $f(\mathbf{x})$ and a Clarke subgradient $\phi \in \partial f(\mathbf{x})$ are provided. In automatic control applications it is as a rule possible to provide more specific information, which may be exploited to speed up convergence.

Computing function value and gradients of the $H_2$-norm $f(\mathbf{x}) = \|T_{wz}(P,K(\mathbf{x}))\|_2$ requires essentially the solution of two Lyapunov equations of size $n_P + n_K$, see [Apkarian et al, 2007; Rautert and Sachs, 1997]. For the $H_\infty$-norm, $f(\mathbf{x}) = \|T_{wz}(P,K(\mathbf{x}))\|_\infty$, function evaluation is based on the Hamiltonian algorithm of [Benner et al, 2012; Boyd et al, 1989]. The Hamiltonian matrix is of size $n_P + n_K$, so that function evaluations may be costly for very large plant state dimension ($n_P > 500$), even though the number of outer loop iterations of the bundle algorithm is not affected by a large $n_P$ and generally relates to $n$, the dimension of $\mathbf{x}$. The additional cost for subgradient computation for large $n_P$ is relatively cheap as it relies on linear algebra [Apkarian and Noll, 2006b].

# 4 Computational Tools

The novel non-smooth optimization methods became available to the engineering community since 2010 via the MATLAB Robust Control Toolbox [Robust Control Toolbox 4.2, 2012; Gahinet and Apkarian, 2011]. Routines `HINFSTRUCT`, `LOOPTUNE` and `SYSTUNE` are versatile enough to define and combine tunable blocks $K_i(\mathbf{x})$, to build and aggregate design requirements $T_{wz}^{(k)}$ of different nature, and to provide suitable validation tools. Their implementation was carried out in cooperation with P. Gahinet (MathWorks). These routines further exploit the structure of problem (4) to enhance efficiency, see [Apkarian and Noll, 2007] and [Apkarian and Noll, 2006b].

It should be mentioned that design problems with multiple hard constraints are inherently complex. It is well known that even simultaneous stabilization of more than 2 plants $P^{(j)}$ with a structured control law $K(\mathbf{x})$ is NP-complete, so that exhaustive methods are expected to fail even for small to medium problems. The principled decision made in [Apkarian and Noll, 2006b], and reflected in the MATLAB routines, is to rely on local optimization techniques instead. This leads to weaker convergence certificates, but has the advantage to work successfully in practice. In the same vein, in (4) it is preferable to rely on a mixture of soft and hard requirements, for instance, by the use of exact penalty functions [Noll and Apkarian, 2005]. Key features implemented in the mentioned MATLAB routines are discussed in [Apkarian, 2013; Gahinet and Apkarian, 2011; Apkarian and Noll, 2007].

# 5 Design example

Design of a feedback regulator is an interactive process, in which tools like `SYSTUNE`, `LOOPTUNE` or `HINFSTRUCT` support the designer in various ways. In this section we illustrate their enormous potential by solving a multi-model, fixed-structure reliable flight control design problem.

In reliable flight control one has to maintain stability and adequate performance not only in nominal operation, but also in various scenarios where the air-

craft undergoes outages in elevator and aileron actuators. In particular, wind gusts must be alleviated in all outage scenarios to maintain safety. Variants of this problem are addressed in [Liao et al, 2002].

The open loop F16 aircraft in the scheme of Fig. 4 has 6 states, the body velocities $u, v, w$, pitch, roll, and yaw rates $q, p, r$. The state is available for control as is the flight-path bank angle rate $\mu$ (deg/s), the angle of attack $\alpha$ (deg), and the sideslip angle $\beta$ (deg). Control inputs are the left and right elevator, left and right aileron, and rudder deflections (deg). The elevators are grouped symmetrically to generate the angle of attack. The ailerons are grouped anti-symmetrically to generate roll motion. This leads to 3 control actions as shown in Fig. 4. The controller consists of two blocks, a $3 \times 6$ state-feedback gain matrix $K_x$ in the inner loop, and a $3 \times 3$ integral gain matrix $K_i$ in the outer loop, which leads to a total of $27 = \dim \mathbf{x}$ parameters to tune.

In addition to nominal operation, we consider 8 outage scenarios shown in Table 1.

Table 1: 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 different models associated with the outage scenarios are readily obtained by pre-multiplication of the aircraft control input by a diagonal matrix built from the rows in Table 1.

The design requirements are as follows:

- Good tracking performance in $\mu$, $\alpha$, and $\beta$ 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.

Tracking is addressed by an LQG-cost [Maciejowski, 1989], which penalizes integrated tracking error $e$ and control effort $u$ via

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

Diagonal weights $W_e$ and $W_u$ provide tuning knobs for trade-off between responsiveness, control effort, and balancing of 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. Model-dependent weights allow to express the fact that nominal operation prevails over failure cases. Weights for failure cases are used to achieve limited deterioration of performance or of gust alleviation under deflection surface breakdown.

The second requirement, wind gust alleviation, is treated as a hard constraint limiting the variance of the error signal $e$ in response to white noise $w_g$ driving the Dryden wind gust model. In particular, 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 3, the functions $f(\mathbf{x})$ and $g(\mathbf{x})$ in (5) are $f(\mathbf{x}) := \max_{k=1,\ldots,9} \|T_{rz}^{(k)}(\mathbf{x})\|_2$ and $g(\mathbf{x}) := \max_{k=1,\ldots,9} \|T_{w_g e}^{(k)}(\mathbf{x})\|_2$, where $r$ denotes the set-point inputs in $\mu$, $\alpha$ and $\beta$. The regulated output $z$ is

$$z^T := \left[\begin{array}{cc} (W_e^{1/2} e)^T & (W_u^{1/2} u)^T \end{array}\right]^T,$$

with $\mathbf{x} = (\mathrm{vec}(K_i), \mathrm{vec}(K_x)) \in \mathbb{R}^{27}$. Soft constraints are the square roots of $J$ in (6) with appropriate weightings $W_e$ and $W_u$, hard constraints the RMS values of $e$, suitably weighted to reflect variance bounds of 0.01 and 0.03. These requirements are covered by the `Variance` and `WeightedVariance` options in [Robust Control Toolbox 4.2, 2012].

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 $\mu$, $\alpha$, and $\beta$ with a gust speed of $5\,m/s$ are shown in Fig. 5 for the nominal design and in Fig. 6 for the fault-tolerant design. As expected, nominal responses are good but notably deteriorate when faced with outages. In contrast, the fault-tolerant controller maintains acceptable performance in outage situations. 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 (5) 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. The reader is referred to [Robust Control Toolbox 4.2, 2012] or higher versions, further examples, and additional details.

## Future directions

From an application viewpoint, non-smooth optimization techniques for control system design and tuning will become one of the standard techniques in the engineer's toolkit. They are currently studied in major European aerospace industries.

Future directions may include

- Extension of these techniques to gain-scheduling in order to handle larger operating domains.
- Application of the available tools to integrated system/control when both system physical characteristics and controller elements are optimized to achieve higher performance. Application to fault detection and isolation may also reveal as an interesting vein.

## Cross References

Controller tuning, optimization-based design, robust synthesis, non-smooth optimization, $H_\infty$ synthesis, multi-objective synthesis, structured controllers

## Recommended reading


Apkarian P (2013) Tuning controllers against multiple design requirements. In: American Control Conference (ACC), 2013, pp 3888–3893

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

Apkarian P, Noll D (2006b) Nonsmooth $H_\infty$ synthesis. IEEE Trans Aut Control 51(1):71–86

Apkarian P, Noll D (2006c) Nonsmooth optimization for multidisk $H_\infty$ synthesis. European J of Control 12(3):229–244

Apkarian P, Noll D (2007) Nonsmooth optimization for multiband frequency domain control design. Automatica 43(4):724–731

Apkarian P, Noll D, Thevenet JB, Tuan HD (2003) A spectral quadratic-SDP method with applications to fixed-order $H_2$ and $H_\infty$ synthesis. European J of Control 10(6):527–538

Apkarian P, Noll D, Rondepierre A (2007) Mixed $H_2/H_\infty$ control via nonsmooth optimization. In: Proc. of the 46th IEEE Conference on Decision and Control, New Orleans, LA, pp 4110–4115

Apkarian P, Noll D, Prot O (2008) A trust region spectral bundle method for nonconvex eigenvalue optimization. SIAM J on Optimization 19(1):281–306

Benner P, Sima V, Voigt M (2012) $L_\infty$-norm computation for continuous-time descriptor systems using structured matrix pencils. IEEE Trans Aut Control 57(1):233–238

Boyd S, Balakrishnan V, Kabamba P (1989) A bisection method for computing the $H_\infty$ norm of a transfer matrix and related problems. Mathematics of Control, Signals, and Systems 2(3):207–219

Burke J, Lewis A, Overton M (2005) A robust gradient sampling algorithm for nonsmooth, nonconvex optimization. SIAM J Optimization 15:751–779

Fares B, Apkarian P, Noll D (2001) An augmented lagrangian method for a class of LMI-constrained problems in robust control theory. Int J Control 74(4):348–360

Fares B, Noll D, Apkarian P (2002) Robust control via sequential semidefinite programming. SIAM J on Control and Optimization 40(6):1791–1820

Gahinet P, Apkarian P (2011) Structured $H_\infty$ synthesis in MATLAB. In: Proc. IFAC World Congress, Milan, Italy, pp 1435–1440

Kocvara M, Stingl M (2003) A code for convex nonlinear and semidefinite programming. Optimization Methods and Software 18(3):317–333

Kolda TG, Lewis RM, Torczon V (2003) Optimization by direct search: new perspectives on some classical and modern methods. SIAM Review 45(3):385–482

Lemarechal C (1975) An extension of Davidon methods to nondifferentiable problems. In: Balinski ML, Wolfe P (eds) Math. Programming Study 3, Nondifferentiable Optimization, North-Holland, pp 95–109

Lemaréchal C, Oustry F (2000) Nonsmooth algorithms to solve semidefinite programs. *SIAM Advances in Linear Matrix Inequality Methods in Control series, ed L El Ghaoui & S-I Niculescu*

Liao F, Wang JL, Yang GH (2002) Reliable robust flight tracking control: An LMI approach. IEEE Trans Control Sys Tech 10:76–89

Lieslehto J (2001) PID controller tuning using evolutionary programming. In: American Control Conference, vol 4, pp 2828 –2833 vol.4

Maciejowski JM (1989) Multivariable Feedback Design. Addison-Wesley

Noll D, Apkarian P (2005) Spectral bundle methods for nonconvex maximum eigenvalue functions: first-order methods. Mathematical Programming Series B 104(2):701–727

Noll D, Torki M, Apkarian P (2002) Partially augmented lagrangian method for matrix inequality constraints. submitted Rapport Interne , MIP, UMR 5640, Maths. Dept. - Paul Sabatier University

Noll D, Prot O, Rondepierre A (2008) A proximity control algorithm to minimize nonsmooth and nonconvex functions. Pacific J of Optimization 4(3):571–604

Noll D, Prot O, Apkarian P (2009) A proximity control algorithm to minimize nonsmooth and nonconvex semi-infinite maximum eigenvalue functions. Jounal of Convex Analysis 16(3 & 4):641–666

Oi A, Nakazawa C, Matsui T, Fujiwara H, Matsumoto K, Nishida H, Ando J, Kawaura M (2008) Development of PSO-based PID tuning method. In: International Conference on Control, Automation and Systems, pp 1917 –1920

Rautert T, Sachs EW (1997) Computational design of optimal output feedback controllers. SIAM Journal on Optimization 7(3):837–852

Robust Control Toolbox 42 (2012) The MathWorks Inc. Natick, MA, USA

Stein G, Doyle J (1991) Beyond singular values and loopshapes. AIAA Journal of Guidance and Control 14:5–16

Varga A, Looye G (1999) Symbolic and numerical software tools for LFT-based low order uncertainty modeling. In: In Proc. CACSD'99 Symposium, Cohala, pp 1–6

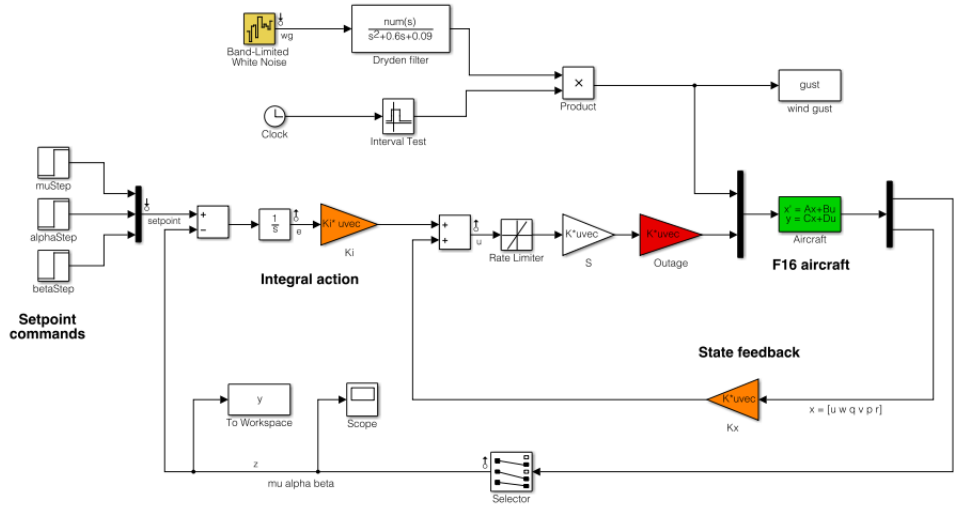


Figure 4: Synthesis interconnection for reliable control

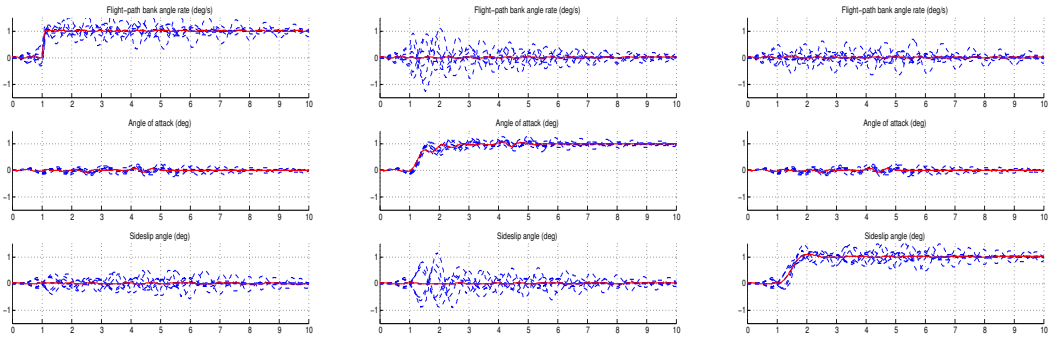


Figure 5: Responses to step changes in $\mu$, $\alpha$ and $\beta$ for nominal design.

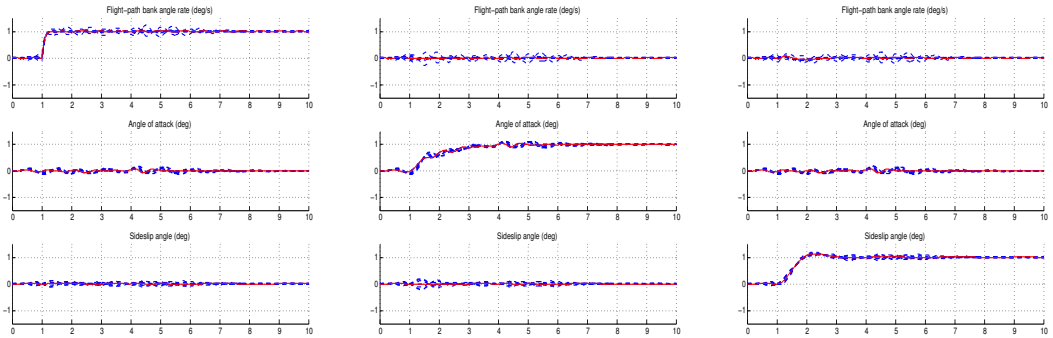


Figure 6: Responses to step changes in $\mu$, $\alpha$ and $\beta$ for fault-tolerant design.