Time-Domain Control Design

a Nonsmooth Approach

Pierre Apkarian * Dominikus Noll † Alberto M. Simões ‡

Abstract

We present a method to efficiently compute locally optimal feedback controllers for synthesis problems formulated in the time-domain. We minimize a time-domain performance objective subject to state or input time-domain constraints. The possibility to include state or input constraints in the design is very appealing from a practical point of view, in particular for plants subject to operational limits as input saturations. Our method is based on a nonsmooth minimization technique, which can handle time-domain constraints as hard constraints. For model-based designs, a stability constraint can also be handled as a hard constraint. The validity and efficiency of the approach are demonstrated through a variety of numerical tests with comparisons with a state-of-the-art technique in constrained optimization.

Keywords: Nonsmooth optimization, structured controllers, PID tuning, time-domain synthesis.

1 Introduction

In traditional frequency based feedback control design for linear time-invariant systems, closed-loop performance specifications like limited overshoot, short settling- or rise-times cannot be addressed directly. Instead, experienced designers know how to handle these specifications heuristically by introducing suitable frequency-domain performance channels, which are then optimized using classical loop-shaping design techniques or more recent H_∞ or H_2 synthesis methods.

Another approach allowing to handle time-domain constraints more directly uses closed-loop system responses $z(t)$ to fixed test input signals $w(t)$ such as steps, ramps or other inputs. The idea is then to find a controller that minimizes the discrepancy between system responses and a given

*ONERA-CERT, Centre d'études et de recherches de Toulouse, Control System Department, 2 av. Edouard Belin, 31055 Toulouse, France - and - Institut de Mathématiques, Université Paul Sabatier, Toulouse, France - Email: apkarian@cert.fr - Tel: +33 5.62.25.22.11 - Fax: +33 5.62.25.27.84

†Université Paul Sabatier, Institut de Mathématiques, 118, route de Narbonne, 31062 Toulouse, France - Email: noll@mip.ups-tlse.fr - Tel: +33 5.61.55.86.22 - Fax: +33 5.61.55.83.85.

‡ONERA-CERT, Centre d'études et de recherche de Toulouse, Control System Department, 2 av. Edouard Belin, 31055 Toulouse, France - Email: alberto.simoes@cert.fr - Fax: +33 5.62.25.27.64.

expected behavior. We follow this line here, and discuss state and control constraints for system response trajectories $z(t)$ and $u(t)$ to control not only overshoot, settling- and rise-time, but also actuator saturation and other operational limits on the system.

A substantial body of work addressing the fixed input design problem uses the Youla parametrization, see [14, 5]. This method solves a recurring difficulty with optimization-based methods: how to efficiently handle the internal stability specification. Unfortunately, it is no longer suited if general structural constraints on the controller have to be satisfied.

A difficulty with the above l_∞- or L_∞-norm formulations is the nonsmoothness of the resulting optimization problem. A recent trend in feedback control - referred to as *iterative feedback tuning* (IFT) [10] - follows the least squares approach. IFT techniques handle time-domain specifications as soft constraints by penalizing L_2 integrals of the constraint violation [8] and use penalization when constraints are present. Penalization strategies raise important and critical questions like how to initialize and update the penalty parameter and how to avoid the inherent ill-conditioning of these techniques for asymptotic values of the penalty parameter. Also, it may lead to unsatisfactory execution times since an unconstrained nonlinear problem must be solved to completion for each value of the parameter.

Our design method computes *locally* optimal structured controllers using a nonsmooth optimization technique. Despite their local nature, these solutions prove to be very useful in practice as demonstrated on a variety of examples. The present work is an extension of our previous work on time-domain synthesis [4] to the considerably more difficult problem where explicit time-domain constraints are present. The paper is organized as follows. Section 2 discusses the time-domain shaping design problem. Our nonsmooth optimization technique is briefly presented in Section 3. Section 4 covers several challenging applications.

Notation

We use concepts from nonsmooth analysis covered by [7]. For a locally Lipschitz function $f : \mathbb{R}^n \to \mathbb{R}$, $\partial f(x)$ denotes its Clarke subdifferential at x while $f'(x;h)$ stand for its directional derivative at x in the direction h. For functions of two variables $f(x,y)$, $\partial_1 f(x,y)$ will denote the Clarke subdifferential with respect to the first variable. For differentiable functions f of two variables x

and y the notation $\nabla_x f(x,y)$ stands for the gradient with respect to the first variable. The symbol $[\cdot]_+$ denotes the threshold function $[x]_+ = \max\{0, x\}$.

2 Structured controllers synthesis in time-domain

Consider a plant P in state-space form

$$P(s): \quad \begin{bmatrix} \dot{x} \\ z \\ y \end{bmatrix} = \begin{bmatrix} A & B_1 & B_2 \\ C_1 & D_{11} & D_{12} \\ C_2 & D_{21} & D_{22} \end{bmatrix} \begin{bmatrix} x \\ w \\ u \end{bmatrix}, \tag{1}$$

where $x \in \mathbb{R}^n$ is the state vector of P, $u \in \mathbb{R}^{m_2}$ the vector of control inputs, $w \in \mathbb{R}^{m_1}$ is a test signal, $y \in \mathbb{R}^{p_2}$ the vector of measurements and $z \in \mathbb{R}^{p_1}$ the controlled or performance vector.

We consider control laws of the form $u = K(s)y$ with state-space realization

$$K(s) = C_K(sI - A_K)^{-1}B_K + D_K, \qquad A_K \in \mathbb{R}^{k\times k}, \tag{2}$$

where the case $k = 0$ of a static controller $K(s) = D_K$ is included. We develop the formulas for static controllers, which allows to unify the setup notationally and facilitates implementation. Formulas for dynamic controllers are then obtained by a prior standard dynamic augmentation of the plant $P(s)$, so that dynamic controller for $P(s)$ becomes static

$$\mathcal{K} := \begin{bmatrix} A_K & B_K \\ C_K & D_K \end{bmatrix} \in \mathbb{R}^{(k+m_2)\times(k+p_2)}. \tag{3}$$

for the augmented system [1]. Structural constraints on the controller may now be defined by a matrix-valued mapping $\mathcal{K}(\cdot)$ from a parameter space $\mathbb{R}^q$ to $\mathbb{R}^{(k+m_2)\times(k+p_2)}$. That is $\mathcal{K} = \mathcal{K}(\kappa)$, where $\kappa \in \mathbb{R}^q$ denotes the independent variables in the controller parameter space $\mathbb{R}^q$. For the time being we will consider free variation $\kappa \in \mathbb{R}^q$, but the reader will be easily convinced that parameter restrictions under the form of mathematical programming constraints $g_I(\kappa) \leq 0, g_E(\kappa) = 0$ could be added if needed. We will assume throughout that the mapping $\mathcal{K}(.)$ is continuously differentiable, but otherwise arbitrary.

The focus is on time-domain synthesis with structured controllers $\mathcal{K}(\kappa)$ for the plant in (1). We want to find $\kappa \in \mathbb{R}^q$ such that

- **Internal stability:** $\mathcal{K}(\kappa)$ stabilizes the original plant $P(s)$ in closed-loop.

- **Performance:** For all stabilizing $\mathcal{K}(\kappa)$ with that structure, the closed-loop time response $z(\kappa, t)$ to an input test signal $w(t)$ with controller $\mathcal{K}(\kappa)$ satisfies the envelope constraints

$$z_{i,min}(t) \leq z_i(\kappa, t) \leq z_{i,max}(t), \ \forall t \geq 0, \ i \in I := \{1, \ldots, p_1\}. \tag{4}$$

The constraints in (4) with upper and lower envelopes define in some sense templates for shaping the closed-loop responses $z(t)$. Typical cases will be illustrated in Section 4, where envelope or shape constraints on overshoot, damping, rise-time, settling-time and steady-state accuracy are imposed to closed-loop responses. Yet, the approach offers the flexibility to incorporate any deterministic input of practical interest such as ramps, sinusoids, stair sequences, etc.

It is also possible and useful to formulate amplitude and rate constraints for the control signal $u(t)$. A standard technique to handle these constraints amounts to augmenting the plant with inputs as new states, as shown schematically in Figure 1. The original sought control law is then easily recovered afterward. The order and structure of the controller are slightly altered in this formulation. Control signal constraints arise regularly in practical designs, and this has generated intensive research in the past decade. Our approach differs from anti-windup schemes and is closer in spirit to the saturation avoidance philosophy. Admittedly with some sacrifice of performance, we try to keep signals at levels where the system dynamics remain linear.

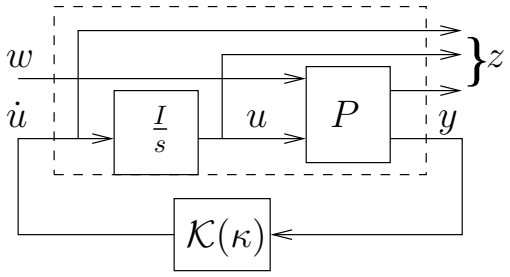


Figure 1: Augmentation of standard form

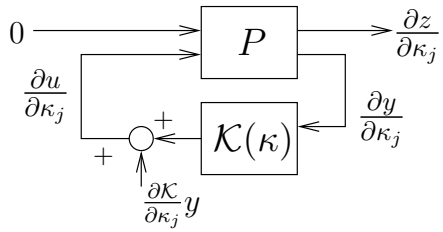


Figure 2: Interconnection for gradient computation

There exist various optimization strategies to handle the specifications in (4). Consider for instance a partition of I into disjoint subsets S and H, i.e., $I = S \cup H$, $S \cap H = \emptyset$, where we think of S as the soft constraints, H the hard constraints. With

$$e_i(\kappa, t) := \max \{z_i(\kappa, t) - z_{i,max}(t), \ z_{i,min}(t) - z_i(\kappa, t)\} \tag{5}$$

a possible form of the program is now

$$\begin{aligned} \underset{\kappa\in\mathbb{R}^q}{\text{minimize}} \quad & f(\kappa) := \max_{i\in S}\max_{t\geq 0}[e_i(\kappa,t)]_+ \\ \text{subject to} \quad & g(\kappa) := \max_{i\in H}\max_{t\geq 0} e_i(\kappa,t) \leq 0\,. \end{aligned} \tag{6}$$

Notice that program (6) has nonsmooth semi-infinite objective and constraints, which do not admit closed-form expressions via space-state representations. For that reason, the max operations involving the closed-loop system responses have to be performed explicitly. In a *model-based* design, time responses are generated from the state-space model (1) through numerical *simulation*, which can be performed using the classical discrete state-propagation approach or a general-purpose ordinary differential equation solver. Yet, they can also be obtained from *experiments* carried out online with the real system. The latter approach is often referred to as the *model-free* approach and forms the basis of the IFT method. In both cases, and also from a practical point of view, a finite horizon for simulation or data acquisition has to be selected, so only a limited number of samples $t \in T = \{t_0, ..., t_k\}$ are considered at each iteration, where the set T may in principle differ at each iteration.

An equivalent more classical formulation for (6) is the following cast:

$$\begin{aligned} \underset{\gamma\in\mathbb{R},\kappa\in\mathbb{R}^q}{\text{minimize}} \quad & \gamma \\ \text{subject to} \quad & \begin{cases} z_i(\kappa,t) - z_{i,max}(t) - \gamma \leq 0 \\ z_{i,min}(t) - z_i(\kappa,t) - \gamma \leq 0 \end{cases} \quad ,\forall i \in S, t \in T \\ & \begin{cases} z_i(\kappa,t) - z_{i,max}(t) \leq 0 \\ z_{i,min}(t) - z_i(\kappa,t) \leq 0 \end{cases} \quad ,\forall i \in H, t \in T\,. \end{aligned} \tag{7}$$

When a fixed sampling time is used to generate the set T throughout the iterations sequence, then program (7) becomes a smooth constrained nonlinear program, since for each fixed time $t \in T$ the constraints are differentiable with respect to the parameters κ of the controller.

Even though state-of-the-art smooth constrained optimization techniques are available for program (7) or the least squares formulation [8], we privilege a nonsmooth semi-infinite optimization algorithm that solves program (6) directly for several reasons:

- First of all, time-domain specifications can be handled as hard constraints hence dispensing with the often critical management of barrier or penalty parameters. The nonsmooth algorithm is more in line with exact penalization techniques where solutions to the original problem are obtained with a single minimization of an appropriate progress function.

- Classical approaches including the state-of-the-art sequential quadratic programming (SQP) of Matlab (function FGOALATTAIN in the Optimization Toolbox) and the least square approach require sampling every trajectory in (7) hence leading to a discretized problem with so many constraints that it might reveal impractical for currently available codes. An illustration of this difficulty is discussed in Applications 4.1 and 4.2.

In sharp contrast, the nonsmooth technique relies solely on active times to generate descent steps. Active times are those times where the max of f and g in (6) are attained which leads to a reduced size discretized problem and therefore enhances efficiency. The proposed technique also offers the flexibility to update the simulation or experiment horizon as well as the sampling time along the iterations to further improve execution times.

3 Nonsmooth minimization technique

We give now a brief presentation of our optimization method and emphasize the main ingredients. For a more detailed discussion we refer the reader to [1, 12, 2]. Following an idea in [12, 2], we introduce the so-called progress function for (6):

$$F(\kappa^+, \kappa) = \max\{f(\kappa^+) - f(\kappa) - \mu g(\kappa)_+; g(\kappa^+) - g(\kappa)_+\}, \tag{8}$$

where $\mu > 0$ is some fixed parameter. We think of κ as the current iterate, κ^+ as the next iterate or as a candidate to become the next iterate. We must search for points $\bar{\kappa}$ satisfying $0 \in \partial_1 F(\bar{\kappa}, \bar{\kappa})$, because this is a necessary condition for a local minimum of (6), see [2] for the proof. Excluding practically rare cases where $\bar{\kappa}$ is a critical point of the constraint violation $g(\bar{\kappa}) \geq 0$, critical points $\bar{\kappa}$ of $F(\cdot, \bar{\kappa})$ will also be critical points of the original program (6).

Approximating a point $\bar{\kappa}$ with $0 \in \partial_1 F(\bar{\kappa}, \bar{\kappa})$ is based on an iterative procedure. Suppose the current iterate κ is such that $0 \notin \partial_1 F(\kappa, \kappa)$. Then it is possible to reduce the function $F(\cdot, \kappa)$ in

a neighborhood of κ, that is, to find κ^+ such that $F(\kappa^+, \kappa) < F(\kappa, \kappa)$. Replacing κ by κ^+, we repeat the procedure. Unless $0 \in \partial_1 F(\kappa^+, \kappa^+)$, in which case we are done, it is possible to find κ^{++} such that $F(\kappa^{++}, \kappa^+) < F(\kappa^+, \kappa^+)$, etc. The sequence $\kappa, \kappa^+, \kappa^{++}, \ldots$ so generated is expected to converge to the sought local minimum $\bar{\kappa}$ of (6).

Finding the descent step κ^+ away from the current κ is based on solving the tangent program at κ. Its name is derived from the fact that a first-order approximation $\widehat{F}(\cdot, \kappa)$ of $F(\cdot, \kappa)$ is built, which provides a descent direction $d\kappa$ at κ, that is, $d_1 F(\kappa, \kappa; d\kappa) < 0$, where $d_1 F$ denotes the directional derivative of $F(\cdot, \kappa)$ at κ in direction $d\kappa$. The next iterate is then $\kappa^+ = \kappa + d\kappa$, or possibly $\kappa^+ = \kappa + \alpha d\kappa$ for a suitable stepsize $\alpha \in (0, 1)$ found by a backtracking line search.

The choice of the progress function in (8) leads to a so-called phase I/phase II method. As long as the constraint $g(\kappa) > 0$ is not satisfied, the right hand term in F is dominant and reducing F amounts to reducing constraint violation. This is phase I, which ends successfully as soon as a feasible iterate has been found. Now phase II begins, and from now on iterates stay (strictly) feasible, and the objective function is minimized at each step. In that case the algorithm converges towards a critical point of (6). The choice of the constant $\mu > 0$ may have an influence on the behavior of the method in phase I, but has been fixed to $\mu = 1$ in our implementation.

In order to define the initial iterate, a stabilizing controller is computed from scratch using the nonsmooth technique in [3]. For model-based designs, a spectral abscissa constraint is added to the original hard constraints g in (6) whenever the solution of the nonsmooth algorithm (6) is not internally stabilizing. The spectral abscissa α is defined as the maximum real part of closed-loop eigenvalues. The constraint in program (6) then becomes $\max\{\alpha - \hat{\alpha}, g(\kappa)\} \leq 0$, where $\hat{\alpha} < 0$ represents a prescribed largest acceptable spectral abscissa.

In order to generate a first-order approximation $\widehat{F}(\cdot, \kappa)$ of $F(., \kappa)$ around κ, we need the set of active times for f: $T_f(\kappa) := \{t \geq 0 : \exists i \in S, [e_i(\kappa, t)]_+ = f(\kappa)\}$. $T_g(\kappa)$ is defined analogously for g. Let us consider the case where $f(\kappa) > 0$, because for $f(\kappa) = 0$ there is nothing to optimize. As the active sets may be small, we consider finite extensions T_f^e and T_g^e of the sets T_f and T_g, respectively. The idea here is that enriched sets capture more information on the closed-loop responses which results in a better tangent model. The proposed technique offers great flexibility

to build such extensions, while guaranteeing convergence [2]. A general characterization is $\forall t \in T_f^e, \exists i \in S, [e_i(\kappa,t)]_+ > 0$. For all such t, the functions $[e_i(\kappa,t)]_+$ are differentiable in a neighborhood of κ. Indeed, $[e_i(\kappa,t)]_+ = \max\{z_i(\kappa,t) - z_{i,max}(t), z_{i,min}(t) - z_i(\kappa,t), 0\}$, and only one component is active in this expression. We have

$$\nabla_\kappa [e_i(\kappa,t)]_+ = \begin{cases} \nabla_\kappa z_i(\kappa,t) & \text{if } z_i(\kappa,t) > z_{i,max}(t) \\ -\nabla_\kappa z_i(\kappa,t) & \text{if } z_{i,min}(t) > z_i(\kappa,t) \end{cases}$$

For all $t \in T_f^e$, we collect all pairs $(\phi_f, \Phi_f) := ([e_i(\kappa,t)]_+, \nabla_\kappa [e_i(\kappa,t)]_+)$ and denote this finite set as $\mathcal{W}_f$. The set $\mathcal{W}_g$ is constructed analogously.

All signals in (1) are differentiable with respect to controller entries, so $\nabla_\kappa z_i(\kappa,t)$ can be obtained by differentiating the state-space equations with respect to κ_j. It follows that the partial derivative of the output signal $\frac{\partial z}{\partial \kappa_j}(\kappa,t)$ corresponds to the output of the interconnection in Figure 2, where the exogenous input w is held at 0, and the vector $\frac{\partial \mathcal{K}}{\partial \kappa_j} y$ is added to the controller output signal. One readily infers that q experiments or simulations are required to form the sought gradients.

For SISO controllers, however, the linear operators $\frac{\partial \mathcal{K}}{\partial \kappa_j}$ and the closed-loop transfer on Figure 2 commute, so instead of filtering y with $\frac{\partial \mathcal{K}}{\partial \kappa_j}$ and then injecting the result in the closed-loop system, one may alternatively inject y only and then filter the system output with $\frac{\partial \mathcal{K}}{\partial \kappa_j}$. Consequently, only one experiment or simulation involving the plant is required for gradient computation no matter the order and structure of the controller. This allows to reduce the experimental overhead in model-free designs and to speed-up computations for the model-based case. We refer the reader to [11] and references therein for a discussion on how to reduce the number of experiments in the MIMO case.

With this preparation, a first-order (tangent) approximation is obtained as

$$\widehat{F}(\kappa + h, \kappa) := \max\left\{ \max_{(\phi_f,\Phi_f)\in\mathcal{W}_f} \phi_f - f(\kappa) - \mu g(\kappa)_+ + \Phi_f^T h, \max_{(\phi_g,\Phi_g)\in\mathcal{W}_g} \phi_g - g(\kappa)_+ + \Phi_g^T h \right\},$$

where h is the displacement in the controller parameter space $\mathbb{R}^q$. This gives the tangent program

$$\underset{h\in\mathbb{R}^q}{\text{minimize}}\ \widehat{F}(\kappa + h, \kappa) + \tfrac{\delta}{2} \|h\|^2 . \tag{9}$$

Program (9) can be turned into a standard convex quadratic program (CQP), and can be efficiently

solved using currently available codes. Current state-of-the-art CQP codes solve problems involving several hundreds of variables and constraints in less than a second.

4 Applications

4.1 Step following with input amplitude and rate constraints

We start our experiments with a simple step following problem borrowed from [9]. Consider the standard negative feedback interconnection of the plant $G(s) = \frac{s+0.5}{s(s-2)}$ and controller $K(s)$ in Figure 3. As in the original problem, we do not use a prefilter in this preliminary study, i.e. $F(s) = I$. The closed-loop system must follow a step reference command with minimum overshoot.

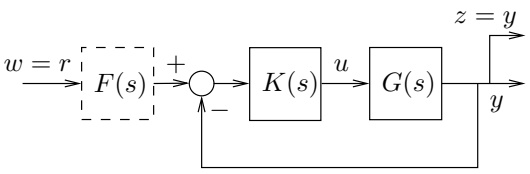


Figure 3: Standard interconnection

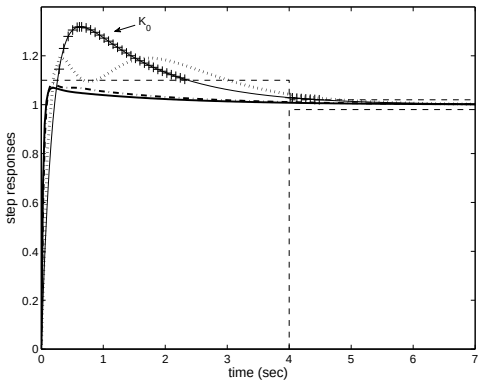


Figure 4: Comparison of step-responses: K_{ns} (solid), K_s (dash-dot), [9] (dot)

The specified time-domain constraints define a settling-time of 4 seconds with worst-case overshoot of 10% and steady-state error of ±2%. The corresponding envelope constraints are drawn as dashed lines in Figure 4. We seek a second-order controller meeting the above constraints.

An initial stabilizing controller is computed as $K_0(s) = \frac{7.93s^2+79.78s+805}{s^2+9.972s+99.55}$. The corresponding closed-loop response $y(t)$ is depicted in Figure 4. Simulation step and and sample time are selected according to the closed-loop system bandwidth using standard Matlab routines. We also display times ('+' symbols) where envelope constraints are violated. These samples are selected to build the tangent subproblem (9) for computation of the descent direction with the nonsmooth technique. In the present case, a globally optimal solution meeting all template constraints has been obtained for problem (6) with zero value of the cost function, and the associated second-order controller is

described as $K_{ns}(s) = \frac{39.02s^2+928s+6408}{s^2+24.49s+157.9}$. The nonsmooth algorithm takes 3.2 seconds cputime on a 2.8GHz Pentium D processor with 1Gb RAM, performing 15 evaluations of (8) and requiring a total of 87 simulations, including those for subgradients computation.

For the sake of comparison, a controller is also designed using the smooth approach. Program (7) is solved using the FGOALATTAIN routine from the Optimization Toolbox from Matlab, which implements a SQP method. The same simulation routines were used, as well as the initial controller K_0. The smooth program needs 15.5 seconds to finds a feasible controller $K_s(s) = \frac{28.92s^2+293.1s+3364}{s^2+9.339s+111.6}$, performing 119 function evaluations (8) and a total of 1111 simulations. Figure 4 depicts the corresponding closed-loop response together with the results for the third-order controller in [9].

With the same example, we now consider a more realistic set-up, where the step following problem is combined with hard constraints on both control input amplitude and rate. This is easily formulated via (6) and the scheme in Figure 1. The additional constraints are $|u(t)| \leq 5$ and $|\dot{u}(t)| \leq 15$. Constraints on the closed-loop step response $y(t)$ are considered as soft constraints. In order to avoid injecting pure step commands, which may result in unduly conservative designs when rate restrictions are present, we use the prefilter $F(s) = \frac{1}{0.3s+1}$ in Figure 3.

The proposed nonsmooth method finds a locally optimal solution in 22 seconds cputime, performing 132 function evaluations and 618 simulations. The associated controller is described as $K_{15}(s) = \frac{90.31s^2+6114s+1079}{s(s^2+64.9s+626.6)}$. The corresponding time-domain simulations including step responses, control signal $u(t)$ and control input derivative $\dot{u}(t)$ are presented in Figure 5.

The input rate constraint turns out to be severe, and the computed controllers do *not* meet the shape constraints. The rate constraint is in some sense exhausted in the transient part of the output response, and we have exactly $\max_{t\geq 0} |\dot{u}| = 15$. Relaxing the rate constraint to 20 allows to satisfy all time-domain specifications, as shown in Figure 5. The associated controller is then $K_{20}(s) = \frac{114.2s^2+9402s+2270}{s(s^2+81.52s+691.4)}$.

4.2 Power system oscillation damping

We discuss now the control of a large dimension system, the oscillation damping of the power system presented in [13]. The system response oscillation is due mainly to a lightly-damped resonant mode,

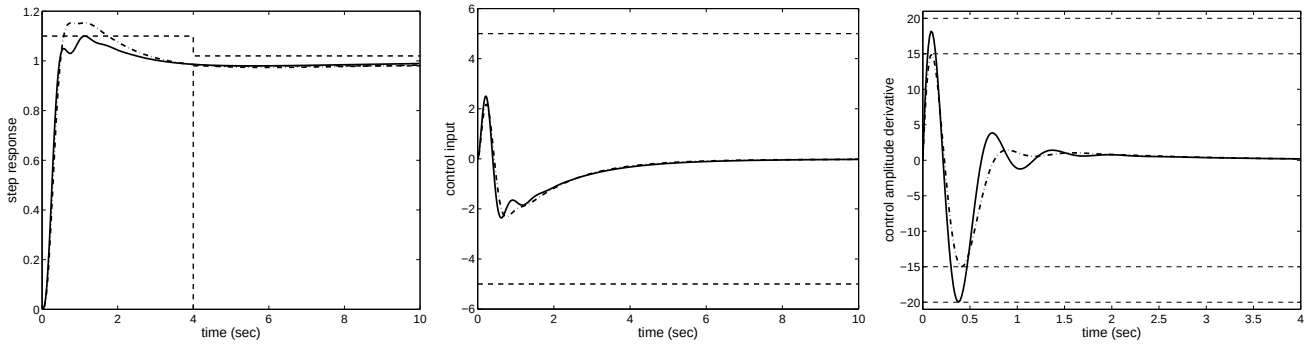


Figure 5: Responses with control amplitude and rate constraints: K_{15} (dash-dot), K_{20} (solid)

known as the NS (north-south) mode, which resulted from the interconnection of the Brazilian north and south sub-systems. In the closed-loop block diagram shown in Figure 6, the measured and also the controlled output $y \in \mathbb{R}$ corresponds to the active power deviation, the control input $u \in \mathbb{R}$ represents the susceptance deviation, and the disturbance $w \in \mathbb{R}$ stands for the deviation in the mechanical power of a plant located at the north side of the interconnection.

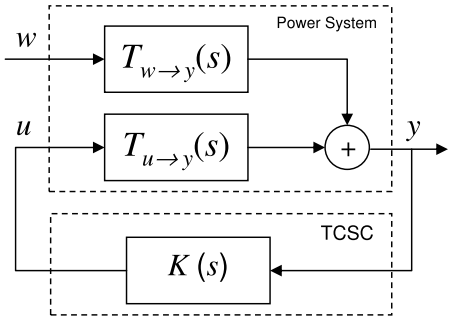


Figure 6: Closed-loop system block diagram representation

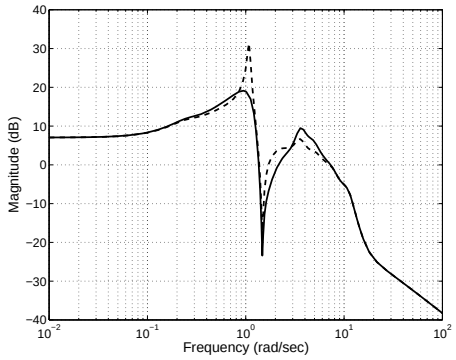


Figure 7: Frequency response for $T_{w\to y}$ (dashed: open-loop, solid: final closed-loop)

We have used a model with 90 states corresponding to the worst-damped scenario in [13]. The NS mode presents a natural frequency of 1.08 rad/s and 3% damping, dominating the transient phase of the open-loop step response. This is confirmed by the magnitude of the frequency response for the open-loop transfer function $T_{w\to y}$ as displayed in Figure 7.

This problem imposes some challenging design specifications. Firstly, from a performance perspective, the primary control objective is to guarantee oscillation damping with the lowest possible overshoot in the presence of disturbance. This must be achieved with a limited control effort deviation to avoid saturation of the Thyristor Controlled Series Compensator(TCSC) components.

Secondly, a reduced-order controller must be sought, given the large dimension of the system. It is also desirable that the controller possesses washout filtering properties to eliminate bias. Therefore, the controller structure was chosen of the form $K(s) = \frac{s}{s+p}\,\hat{K}(s)$, where $\hat{K}(s)$ is a 5th-order strictly proper transfer function, and the position of the real washout pole $-p$ is also a decision variable of the optimization program. Altogether, this gives a controller parametrization $\mathcal{K}(\kappa)$ with free parameters $\kappa \in \mathbb{R}^{36}$.

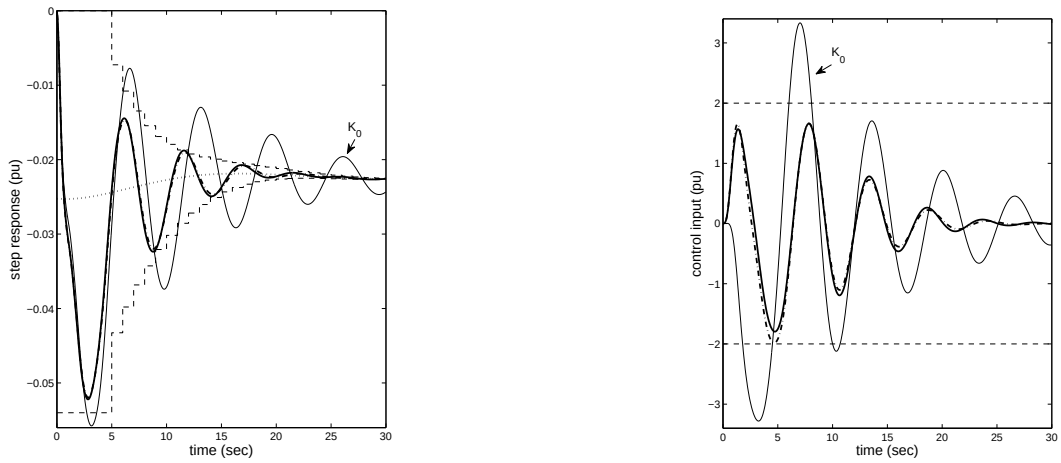


Figure 8: Step and control input responses (dark solid: nonsmooth, dash-dot: smooth)

The closed-loop step response and control input with the computed initial controller $K_0(s)$ are shown in Figure 8. In order to achieve the desired level of oscillation damping, time-domain constraints were constructed as a piecewise constant approximation of a decaying exponential corresponding to 20% damping for the NS mode frequency. These shape constraints appear as step functions in Figure 8. The exponential envelope has an offset equal to the asymptotic value of the open-loop step response, since the controller incorporates a washout filter. Regarding the control effort, constraints were introduced to limit the peak value, which avoids saturating TCSC components. The nonsmooth algorithm needs 31 seconds, 64 function evaluations and 964 simulations to find a feasible controller

$$K(s) = \frac{s(22.02s^4 + 652.4s^3 + 6440s^2 + 6920s + 392.2)}{s^6 + 12.43s^5 + 60.57s^4 + 144.8s^3 + 195.5s^2 + 168.5s + 64.21}.$$

By feasible we mean that the closed-loop step response satisfies both response and control input

constraints, see Figure 8. The closed-loop transfer function $T_{w\to y}$ is drawn in Figure 7. A feasible controller has also been found using the smooth SQP approach of Matlab based on (7). Contrasting with the nonsmooth approach, it required 445 seconds, 623 function evaluations and 22497 simulations.

Figure 8 reveals that the envelope constraints for the output response has been chosen to accommodate a low frequency oscillatory component, which is caused by an almost uncontrollable mode with natural frequency of 0.26 rad/s (dotted line in the figure 8). By definition such a phenomenon cannot and should not be compensated by feedback. As it is very flexible, the proposed design technique can take such plant characteristics into account to avoid unrealistic solutions.

4.3 Model-free design for a process with large dead time

Most design methods are model-based and may perform poorly when confronted with the actual plant. An appealing feature of model-free approaches is that they rely only on experimental data and consequently inaccuracies in the mathematical model are no longer harmful. Moreover, the fact that there is no need to open the control loop is another attractive feature of IFT. A reasonable strategy appears to combine both model-based and model-free strategies. A first controller is computed using an identified model of the plant. If matching the result with experimental data turns out unacceptable, a model-free design is performed to further improve the controller. With this procedure, a complex accurate model is no longer needed and initializing the model-free design with a sensible controller should reduce the number of iterations in the tuning phase.

The process we consider to emulate experimental data is taken from [6] and is described by

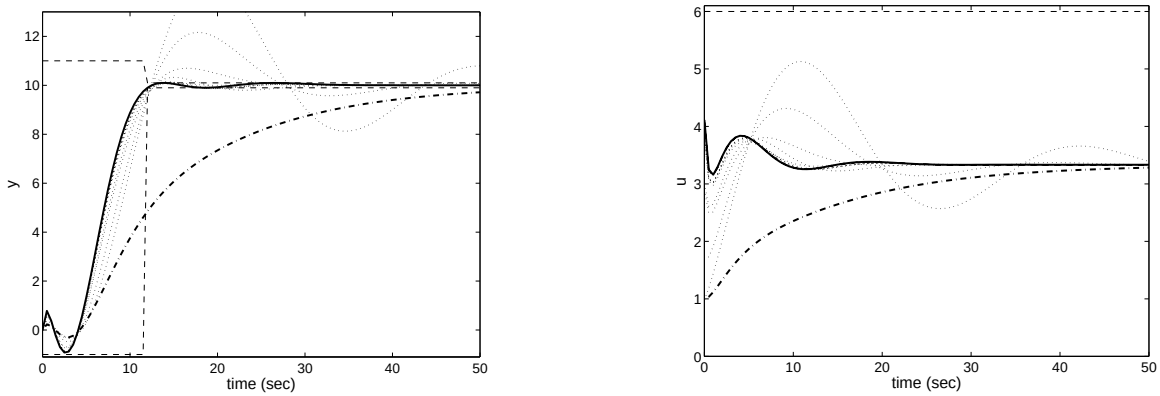
$$y(s) = \frac{3}{3s+1}\frac{7}{s+7}e^{-6s}u(s) + v, \tag{10}$$

where v is white Gaussian noise with zero mean and variance $\sigma^2 = 0.01$. The true dynamics (10) of the process are supposed unknown, and will be used solely as a black-box to generate experimental data of the real system during the re-tuning phase. For the model-based synthesis, a simple model of the process is constructed as a first-order transfer function in series with a 2nd-order Padé approximation for the dead time. The steady-state gain and bandwidth of the system

are accurately modeled, but the dead-time has been underestimated to 5 seconds:

$$y(s) = \frac{3}{3s+1}\left(\frac{s^2 - 1.2s + 0.48}{s^2 + 1.2s + 0.48}\right) u(s). \tag{11}$$

We are seeking a PID controller $K(s) = K_p + \frac{K_i}{s} + \frac{K_d s}{1+\varepsilon s}$ with the classical feedback interconnection shown in Figure 3 (with $F(s) = I$). Parameters for the initial controller $K_0(s)$ are chosen as $K_p = 0.09$, $K_i = 0.02$, $K_d = 0.01$ and $\varepsilon = 1$. Figure 9 shows the closed-loop system responses and control signal for model (11) with controller $K_0(s)$.



(a) (b)

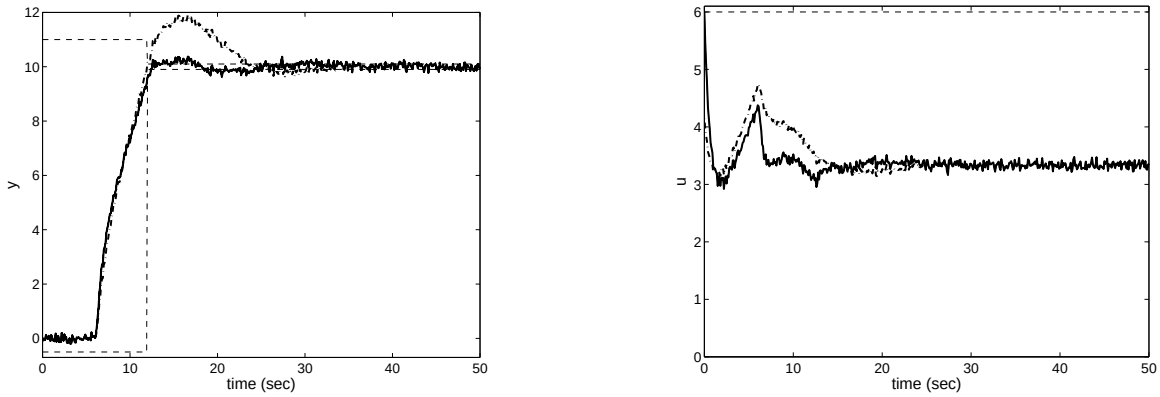
Figure 9: Closed-loop responses with model (11) (dash-dot: K_0, solid: K_1)

Notice in Figure 9 that the process dead time is easily captured by defining appropriate templates. A control effort constraint is also introduced, although it remains initially inactive. Evolution of the step responses along a few final algorithm iterations are shown as dotted-lines in Figure 9. Also shown are the closed-loop responses for a model-based controller $K_1(s)$ computed using the nonsmooth technique. This controller meets all time-domain constraints, and is described by $K_p = 0.199$, $K_i = 0.045468$, $K_d = 0.22304$ and $\varepsilon = 1.0507$.

In the next step, controller $K_1(s)$ is tested with the true noisy process (10). The corresponding responses are shown in Figure 10. Due to the discrepancy between model (11) and the true process (10), $K_1(s)$ performs poorly. Compare with Figure 9.(a).

This leads us to re-tune the controller using experimental data and identical time constraints. The model-free synthesis automatically adjusts to the true time-delay and no further information is required. The (model-free) PID parameters are obtained as $K_p = 0.1994$, $K_i = 0.0395$, $K_d = 0.2872$

and $\varepsilon = 0.7274$. The initial overshoot has been significantly reduced, see Figure 10.(a). The specified control effort constraint becomes active, as can be seen from Figure 10.(b). The final constraints violation falls below $\gamma = 0.27495$, that is 2.7% and hence becomes acceptable.



(a) (b)

Figure 10: Closed-loop responses with true process (dash-dot: K_1, solid: K_2)

5 Conclusion

We have described a nonsmooth algorithm to compute locally optimal solutions to time-domain synthesis problems. The approach is flexible as it applies to many different scenarios and can capture any controller structure of practical interest. The proposed technique expands on our previous results which were restricted to the minimization of a single cost function without trajectory (hard) constraints [4]. In terms of execution times, our technique outperforms the standard SQP approach as illustrated on a variety of examples.

References

[1] P. Apkarian and D. Noll, *Nonsmooth H_∞ synthesis*, IEEE Trans. Aut. Control, 51 (2006), pp. 71–86.

[2] P. Apkarian, D. Noll, and A. Rondepierre, *Nonsmooth optimization algorithm for mixed H_2/H_∞ synthesis*, in Proc. of the 46th IEEE Conference on Decision and Control, New Orleans, LA, 2007, pp. 4110–4115.

[3] V. Bompart, P. Apkarian, and D. Noll, *Nonsmooth techniques for stabilizing linear systems*, in Proc. American Control Conf., New York, NY, July 2007, pp. 1245–1250.

[4] ——, *Control design in the time- and frequency-domain using nonsmooth techniques*, Syst. Control Letters, 57 (2008), pp. 271–282.

[5] S. Boyd and C. Barratt, *Linear Controller Design: Limits of Performance*, Prentice-Hall, 1991.

[6] F. D. Bruyne, *Iterative feedback tuning for internal model controllers*, Control Engineering Practice, 11 (2003), pp. 1043–1048.

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

[8] K. Hamamoto, T. Fukuda, and T. Sugie, *Iterative feedback tuning of PID parameters: comparison with classical tuning rules*, Control Engineering Practice, 11 (2003), pp. 1061–1068.

[9] D. Henrion, S. Tarbouriech, and V. Kucera, *Control of linear sytems subject to time-domain constraints with polynomial pole placement and LMIs*, IEEE Trans. Aut. Control, 50 (2005), pp. 1360–1364.

[10] H. Hjalmarsson, M. Gevers, S. Gunnarsson, and O. Lequin, *Iterative Feedback Tuning: theory and applications*, IEEE Control Syst. Mag., 18 (1998), pp. 26–41.

[11] J. Jansson and H. Hjalmarsson, *Gradient approximations in Iterative Feedback Tuning of multivariable processes*, Int. J. Adaptive Contr. and Sig. Process., 18 (2004), pp. 665–681.

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

[13] D. C. Savelli, P. C. Pellanda, N. Martins, N. J. P. Macedo, A. A. Barbosa, and G. S. Luz, *Robust signals for the* TCSC *oscillation damping controllers of the brazilian north-south interconnection considering multiple power flow scenarios and external disturbances*, Proceedings of the IEEE PES General Meeting, (2007).

[14] M.-G. Yoon, *Sign-weighted peak minimization problem for feedback systems*, IEEE Trans. Aut. Control, 46 (2001), pp. 943–948.