A Convex Control Design (CCD) Toolbox with frequency-domain specifications

Gilles FERRERES
ONERA–DCSD
System Control and Flight Dynamics Department
BP 4025, F–31055 Toulouse Cedex, France
ferreres@onera.fr

Abstract

The CCD Toolbox proposes Matlab computational tools for the convex design of LTI feedback controllers and the convex design of gain-scheduled LFT feedback controllers, using (extended) Youla parameterization.

Contents

1 Introduction **4**

2 Toolbox structure and software requirements **5**

3 Principles of convex control design with Youla parameterization **6**

3.1 Youla parameterization 6

3.2 Convexity of design specifications 7

3.3 Problems to be solved 8

4 The design procedure **9**

4.1 Introduction of the open loop model 9

4.2 Design of an initial controller 9

4.3 Choice of the basis of filters 10

4.4 Definition of the problem 10

4.4.1 Use of the Simulink file *sys_closed_loop.mdl* 10

4.4.2 Definition of *specif* and *puls_specif* 13

4.4.3 H_∞ constraint 13

4.4.4 H_∞ minimization 14

4.4.5 H_2 minimization 16

4.4.6 H_2 constraint 17

4.4.7 Summary 17

4.5 Convex optimization with a cutting planes method 18

4.5.1 Principle of the cutting planes method 18

4.5.2 The algorithm 19

4.5.3 One-shot vs progressive design 20

4.5.4 On screen 20

4.5.5 Advanced tuning of the optimization algorithm 21

4.6 Description of *obs_state_feedback.mat* 22
4.7 Description of *youla_file.mat* 23

5 Convex design of a gain-scheduled LFT controller 23

1 Introduction

The issue of convex control design [5] is twofold. The first is to check the feasibility of design specifications: does there exist an LTI controller, whose order is free, which satisfies a set of design specifications on an LTI closed loop? The second is to translate in a *direct* way a set of specifications into the design procedure. Using Youla-parametrization many design specifications (most nominal performance specifications and unstructured robustness ones) can be translated as convex constraints on the design parameters, or as the minimization of a convex objective.

The convexity of the optimization problem is crucial since it enables to check the feasibility of these specifications. A great deal of work has been devoted to the non-convex design of fixed-order controllers, which satisfy a set of design specifications. Despite the interest of these methods it is worth emphasizing that the solution depends on the initialization, and if no solution is found this does not mean that it does not exist.

The principles of convex control design are exposed in the pioneering work of [5]. Nevertheless, many practical problems remained unsolved, and especially the computational requirement. Because of the industrial interest for the issue of checking the feasibility of design specifications, and thanks to a regain of interest for convex control design since the end of the 90's the subject is now more mature. Our contribution here is to propose computational tools for the convex design of feedback controllers. The toolbox also contains a realistic design example of a flight control system for a flexible aircraft, whose numerical data are extracted from [6]. See also [12] for an other application. The design procedure is the following:

- Design of an initial controller, which is then put under the form of an observed-state feedback controller [3, 4, 2], or direct design of an observed-state feedback controller.
- Convex optimization over the Youla parameter $Q(s) = \sum_i \theta_i Q_i(s)$. Filters $Q_i(s)$ are fixed, while the θ_i are the design parameters. The feedback controller $K(s)$ is deduced from the initial controller and from the optimal value of $Q(s)$. H_2 and H_∞ specifications can be accounted for.

A specific effort was made on the computational requirement when solving the convex optimization problem [10]. A first solution is to use a classical LMI solver: H_2 and H_∞ specifications can be translated into state-space LMI constraints. However the practical application of this classical solution is (very) difficult:

- When applying e.g. the KYP lemma on the closed loop system, its order is $2n_G + n_Q$, where n_G (resp. n_Q) is the order of the open loop plant model

(resp. of $Q(s)$). Even if n_G is low a large basis of filters $Q_i(s)$ is necessary to check the feasibility of design specifications, so that the order of the closed loop system is usually (very) high, which leads to an excessive computational time when using an LMI solver.

- An analytic expression is necessary for H_∞ templates when using state-space solutions, noting that many design specifications are directly expressed in the frequency domain (e.g. a constraint on the magnitude of a closed loop transfer function over a limited frequency interval). Deducing from such a specification an approximate analytic template is not so easy, and the order of the closed loop system is increased by the order of this template.

We have thus developed a frequency-domain solution in the spirit of [11]. The convex optimization problem is solved on a frequency gridding, and design specifications are then checked between the points of the gridding. If necessary the gridding is refined and a new convex optimization problem is solved. Moreover convex constraints at each point of the gridding are approximated as linear ones, using a cutting planes method. More precisely, the idea is to solve in an exact way the convex optimization problem as a series of approximate LP problems.

The toolbox also contains a realistic engineering application, namely the design of a flight control system for a flexible aircraft whose numerical data is extracted from [6]. The paper is organized as follows. Section 2 first presents the structure of the toolbox and software requirements. The principles of convex control design are exposed in section 3, while the whole design procedure is detailed in section 4. Last, section 5 deals with the extension of the method to the convex design of a gain-scheduled LFT controller, with an application to a nonlinear missile model.

2 Toolbox structure and software requirements

The directory *Demo_LTI_flexible_airplane*, which presents the design of a feedback controller on a flexible airplane model [6], contains the main files:

1. *script_1_design_initial_controller.m*: the initial observed state-feedback controller is computed using a modal method. The design is willingly very simple, with only one tuning parameter. The file *open_loop_model.mat*, which contains the open loop model, is used as the main input data. The file *obs_state_feedback.mat* is produced.

2. *script_2_calc_youla_parameterization.m*: on the basis of the initial controller in *obs_state_feedback.mat*, the Youla parameterization is computed and saved in *youla_file.mat*. The Simulink file *sys_closed_loop.mdl* is used.

3. *script_3_design_youla_parameter.m*: convex design of the optimal value of the Youla parameter. The trade-off between the design specifications can be studied. The design spec. are defined in *design_spec.m*. The tuning parameters of the optimization algorithm are defined in *design_tuning.m*. The optimal controller is saved in *controller.mat*.

The routines of the CCD Toolbox are gathered in the directory *Routines_CCD*, the main routines being *design_convex.m* and *validation.m*. The Optimization Toolbox, the System Control Toolbox and Simulink are required.

3 Principles of convex control design with Youla parameterization

3.1 Youla parameterization

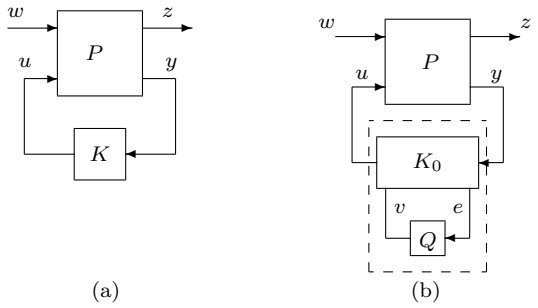

Figure 1: The design problem (a) and Youla parameterization (b).

Consider the standard design problem of figure 1.a, where $P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix}$ is an augmented plant. The closed loop transfer matrix $F_l(P,K) = P_{11} + P_{12}K(I - P_{22}K)^{-1}P_{21}$ is a highly nonlinear function of controller K. Suppose an initial stabilizing controller K_0, whose order is equal to the order of P_{22}, is available. Additional inputs and outputs v and e are introduced in K_0 (see figure 1.b), with the key constraint that the closed loop transfer matrix between v and e is zero (a solution to achieve this property is to put K_0 under the form of an observed state feedback controller). When connecting a free stable transfer matrix Q to these additional inputs and outputs, $F_l(P,K)$ can be rewritten as $T_1 + T_2QT_3$, where fixed transfer

matrices T_i depend on P and K_0, while Q is the design parameter. Indeed, the closed loop transfer matrix between (w, v) and(z, e) is $\begin{bmatrix} T_1 & T_2 \\ T_3 & 0 \end{bmatrix}$.

3.2 Convexity of design specifications

Generally speaking, a norm constraint on the closed loop transfer matrix $T_1+T_2QT_3$ is convex with respect to $Q = \sum_i \theta_i Q_i$, where filters Q_i are fixed while the θ_i are the design parameters. As a consequence, when constraining or minimizing the norm of various parts of the closed loop transfer matrix $T_1 + T_2QT_3$, a convex optimization problem with convex constraints is obtained. Optimal values of the design parameters θ_i are computed, $Q(s)$ is deduced as well as $K(s)$ (see figure 1.b).

Remarks:
(i) The order of the controller $K(s)$ is generally equal to the sum of the orders of the plant model and Youla parameter $Q(s)$.
(ii) Convex optimization offers two main advantages with respect to H_∞ control. It is possible to *separately* minimize or constrain H_∞ norms of different SISO or MIMO transfer functions of $F_l(P, K)$ instead of minimizing a unique large MIMO transfer matrix $F_l(P, K)$. Moreover, a limited frequency domain can be considered and analytic expressions of the templates are not required.

Consider now an H_∞ constraint:

$$\overline{\sigma}\left[T_1(j\omega) + T_2(j\omega)\left(\sum_i \theta_i Q_i(j\omega)\right) T_3(j\omega)\right] \leq \alpha(\omega) \tag{1}$$

or the minimization of γ under the constraint:

$$\overline{\sigma}\left[T_1(j\omega) + T_2(j\omega)\left(\sum_i \theta_i Q_i(j\omega)\right) T_3(j\omega)\right] \leq \gamma\alpha(\omega) \tag{2}$$

where $\alpha(\omega)$ is a fixed frequency-domain template. The above constraints are convex with respect to the θ_i. They can be translated at each frequency as LMI constraints, but we prefer in this toolbox to solve them with a computationally more efficient cutting planes method.

Note finally that the square of the H_2 norm of the transfer function $T(s, \theta) = T_1(s) + T_2(s)(\sum_i \theta_i Q_i(s))T_3(s)$ on a finite frequency interval $[\underline{\omega}, \overline{\omega}]$:

$$\frac{1}{2\pi}\int_{\underline{\omega}}^{\overline{\omega}} Trace(T^*(j\omega, \theta)T(j\omega, \theta))d\omega \tag{3}$$

can be approximated as a quadratic criterion $f_0 + f^T\theta + \theta^T Q\theta$ using a fine enough frequency gridding. An H_2 constraint can then be considered in the toolbox:

$$\sqrt{\frac{1}{2\pi}\int_{\underline{\omega}}^{\overline{\omega}} Trace(T^*(j\omega,\theta)T(j\omega,\theta))d\omega} \leq \alpha \tag{4}$$

where the positive scalar α is fixed. When minimizing the H_2 norm of different transfer functions (e.g. $T_1(s)$ and $T_2(s)$) the following criterion is minimized:

$$p_1\int_{\underline{\omega}_1}^{\overline{\omega}_1} Trace(T_1^*(j\omega)T_1(j\omega))d\omega + p_2\int_{\underline{\omega}_2}^{\overline{\omega}_2} Trace(T_2^*(j\omega)T_2(j\omega))d\omega \tag{5}$$

where p_1 and p_2 are weighting factors.

3.3 Problems to be solved

Despite the pioneering work of [5] many practical problems remained unsolved:

- How to choose the basis of filters? A large basis of filters $Q_i(s)$ is necessary to check the feasibility of design specifications, this basis should even be theoretically infinite.
- Computational requirement, as explained in the introduction: despite its convexity the optimization problem which we solve is not an easy one, since it has an infinite number of constraints and a (very) large number of optimization parameters.
- The order of the controller can be (very) large, necessarily greater than the order of the open loop plant model.
- Structured robustness specifications are non-convex, as well as multi-model design.

4 The design procedure

4.1 Introduction of the open loop model

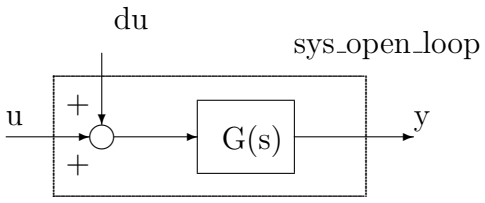

Figure 2: Definition of the open loop model.

The file *open_loop_model.mat* contains the open loop model *sys_open_loop* (with ss format) as well as $feedback\_inputs$ (resp. $feedback\_outputs$) which describes the list of the inputs (resp. outputs) used for feedback. Some inputs or outputs can be used for design specifications, but not for feedback. THE PLANT MODEL SEEN BY THE FEEDBACK CONTROLLER, i.e. the transfer matrix $sys\_open\_loop(feedback\_outputs, feedback\_inputs)$, MUST BE STRICTLY PROPER.

In the context of our flexible aircraft example *sys_open_loop* has 4 inputs and 9 outputs, but only the first 2 inputs and first 4 outputs are used for feedback. Thus:

```
feedback_inputs=1:2;
feedback_outputs=1:4;
```

The 2 additional inputs correspond to additive disturbances on the control inputs, as indicated on the figure. *sys_open_loop* contains actuators as well as rigid and flexible models of the aircraft (4 rigid states and 6 bending modes). See [6] for more details.

Remark: the 9 outputs of *sys_open_loop* are ny, p, r, ϕ + rigid and flexible ny + actuator rates + β. The 4 inputs are aileron and rudder deflections + additive disturbances on these control inputs.

4.2 Design of an initial controller

A first solution is to directly synthesize an observed-state feedback controller, as done in *script_1_design_initial_controller.m*. Otherwise, any controller whose order

is at least equal to the order of the plant model can be put under the form of an observed-state feedback controller, with possibly an initial value $Q_0(s)$ of the Youla parameter [3, 4, 2].

In *script_1_design_initial_controller.m* the computation of the observed state feedback controller is willingly rough. The issue is just to satisfy with an automatic LQ method a stability degree requirement for the closed loop poles. Remember the stability degree of a state-matrix A is defined as:

$$\alpha = -\max_i \Re(\lambda_i) \tag{6}$$

where λ_i is an eigenvalue of matrix A. Thus, the degree of stability is positive if the plant is stable. Other more sophisticated techniques could be used. *Remember the better the initial controller is, the simpler the convex design of Q(s) is.*

4.3 Choice of the basis of filters

Remember $Q(s) = \sum_i \theta_i Q_i(s)$, where filters $Q_i(s)$ are fixed. The orthonormal basis of [1] is used:

$$Q_i(s) = \frac{\sqrt{2Re(a_i)}}{s + a_i} \Pi_{k=1}^{i-1} \frac{s - \overline{a}_k}{s + a_k} \tag{7}$$

The poles, which are chosen on the basis of our physical knowledge of the plant, are fixed. The use of an orthonormal basis reduces numerical problems which can be encountered in practice (especially when solving the LP problems in the cutting planes method, see below).

4.4 Definition of the problem

The design specifications are defined with the Simulink file *sys_closed_loop.mdl* and with variables $specif$ and $puls\_specif$, specified in the file *design_spec.m*.

4.4.1 Use of the Simulink file *sys_closed_loop.mdl*

sys_closed_loop.mdl is to be modified according to the structure of the problem. Remember $feedback\_inputs$ and $feedback\_outputs$ contain the list of plant inputs and outputs used for feedback, but that other I/O can be used for design specifications.

In the context of our flexible aircraft example, figure 3 presents the corresponding file *sys_closed_loop.mdl*. Input # 1 corresponds to the disturbances in the control

inputs, which were integrated inside the open loop model *sys_open_loop*. Input # 2, named v, corresponds to additional inputs of the augmented controller. In the same way, outputs # 1 and # 2 correspond to all plant outputs and to the controller outputs, while output # 3, named e, corresponds to additional outputs of the augmented controller. These additional I/O v and e will be used to connect the Youla parameter $Q(s)$. Last note that "Selector 2" is used to select only the outputs used for feedback.

When changing sys_closed_loop.mdl and especially when adding or removing I/O, the issue is to place the additional I/O of the augmented controller as the last ones. A way to test the modifications of *sys_closed_loop.mdl* is to check whether the affine closed loop form $T_1 + T_2QT_3$ is well obtained, by checking that the transfer matrix between all inputs and outputs of the Simulink file is under the form $T(s) = \begin{bmatrix} T_1(s) & T_2(s) \\ T_3(s) & 0 \end{bmatrix}$.

This is done in *script_3_design_youla_parameter.m* by choosing a basis $Q(s) = \sum_i \theta_i Q_i(s)$, a frequency gridding and a random value of θ. The corresponding value of $Q(s)$ is computed and applied to $T(s)$ (i.e. connected to the additional I/O of the augmented controller). The closed loop frequency response $F(j\omega)$ is computed (in the context of figure 3 $F(s)$ is the transfer function between input # 1 and outputs # 1 and # 2, after input # 2 and output # 3 were connected to $Q(s)$). In the same way, the closed loop frequency responses $F_i(j\omega)$ are computed by connecting $T(s)$ to each $Q_i(s)$. The quantity:

$$F(j\omega) - T_1(j\omega) - \sum_i \theta_i (F_i(j\omega) - T_1(j\omega))$$

should be very small, ideally 0. Indeed, $F(j\omega)$ is expected to be:

$$F(j\omega) = T_1(j\omega) + T_2(j\omega)(\sum_i \theta_i Q_i(j\omega))T_3(j\omega)$$

while $F_i(j\omega)$ is expected to be $T_1(j\omega) + T_2(j\omega)Q_i(j\omega)T_3(j\omega)$.

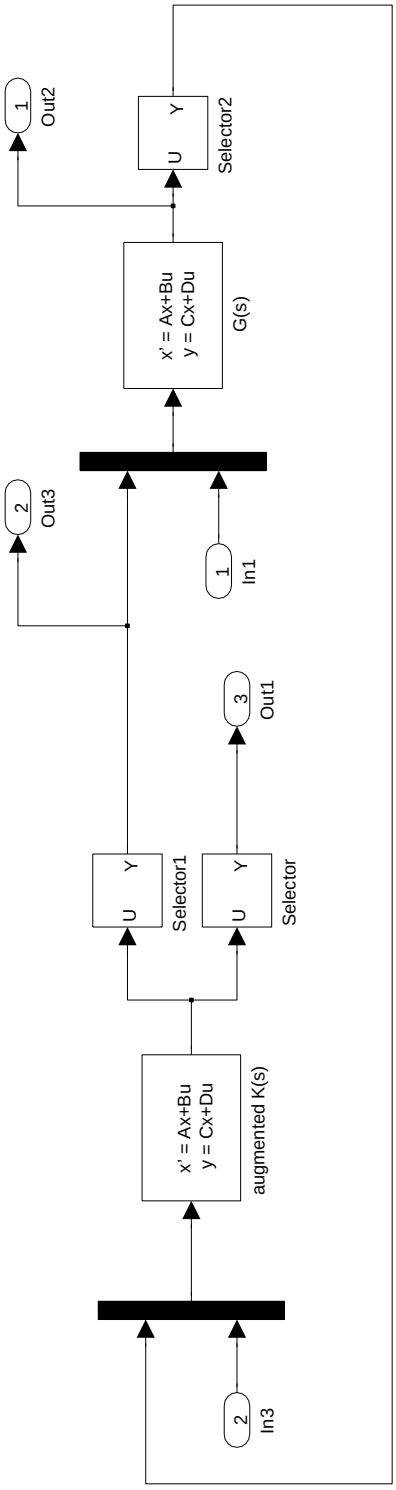

Figure 3: The simulink file *sys_closed_loop.mdl*.

4.4.2 Definition of $specif$ and $puls\_specif$

We explain in the following how to define the structured cells $specif$ and the row vector $puls\_specif$. We first give some generalities:

- The length of the cell is equal to the number of design specifications. If there are two specifications, then $specif\{1\}$ and $specif\{2\}$ have to be defined.
- The nature of spec $\#i$ is defined by $specif\{i\}.obj$.
- $specif\{i\}.input$ and $specif\{i\}.output$ contain the list of inputs and outputs in $sys\_closed\_loop.mdl$, which describe the SISO or MIMO transfer function to be shaped or minimized.
- A unique vector $puls\_specif$ is used for all specifications. Here:

  ```
  puls_specif=[5 25 0 logspace2(1e-2,1e3,100)]
  ```

  Note that $logspace2(1e-2, 1e3, 100)$ means 100 logarithmically spaced points between 10^{-2} and 10^{3} rad/s.

4.4.3 H_∞ constraint

This one corresponds to $specif\{i\}.obj = 0$. Consider the following example where $Gain\_roll\_off$ and $omega\_roll\_off$ (in rad/s) are tuning parameters:

```
%
% Hinf constraint between two disturbance inputs and two feedback controller outputs
%
num=Gain_roll_off;
den=conv([1/omega_roll_off 1],[1/omega_roll_off/omega_roll_off 1.4/omega_roll_off 1]);
[a,b,c,d]=tf2ss(num,den);
sys=ss(a,b,c,d);
%
specif{3}.input=[1 2];
specif{3}.output=[10 11];
specif{3}.list=3:length(puls_specif);
specif{3}.template=squeeze(abs(freqresp(sys,puls_specif(3:end))));
specif{3}.obj=0;
specif{3}.lin_log_x=1;
specif{3}.lin_log_y=1;
specif{3}.interp=1;
```

With reference to figure 3, the above specification defines a roll-off constraint between input # 1 and output # 2 (the size of output # 1 is 9).
$specif\{3\}.list$ describes the list of specified frequencies. Remember:

```
puls_specif=[5 25 0 logspace2(1e-2,1e3,100)]
```

so that:

```
specif{3}.list=3:length(puls_specif)
```

means that all frequencies are used except the first 2 ones. The size of $specif\{3\}.template$ must be the same as the one of $specif\{3\}.list$, i.e. the template has to be defined only on the frequency points of interest, not on the whole initial frequency gridding. The template, which must only contain positive real values, can be defined in two different ways between the points of the frequency gridding, according to $specif\{3\}.interp = 0$ (rectangular interpolation, see figure 4.a) or $specif\{3\}.interp = 1$ (linear interpolation, see figure 4.b). Note finally that $specif\{3\}.lin\_log\_x$ (resp. $specif\{3\}.lin\_log\_y$) defines the linear or logarithmic nature of the x (resp. y) axis for visualization.

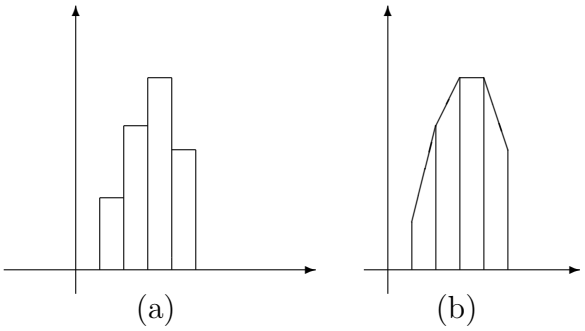

Figure 4: Definition of the interpolation method.

4.4.4 H_∞ minimization

The fields (input, output, list...) are the same as in the previous subsection, except that $specif\{i\}.obj = 1$. Note also that the simultaneous minimization of different transfer functions is possible, as in the following example:

```
%
% Hinf min. of the transfer function between the 1st disturbance input and ny
```

```
%
specif{1}.input=1;
specif{1}.output=1;
specif{1}.list=[1 2];
specif{1}.template=1e-3*[1 1];
specif{1}.obj=1;
specif{1}.lin_log_x=0;
specif{1}.lin_log_y=0;
specif{1}.interp=0;
%
% Hinf min. of the transfer function between the 2d disturbance input and ny
%
specif{2}.input=2;
specif{2}.output=1;
specif{2}.list=[1 2];
specif{2}.template=1e-3*[1 1];
specif{2}.obj=1;
specif{2}.lin_log_x=0;
specif{2}.lin_log_y=0;
specif{2}.interp=0;
```

With reference to figure 3, the above specification defines the minimization of both SISO transfer functions between the 2 scalar inputs in input # 1 and the first scalar output in output # 1, which corresponds to an acceleration ny.

template defines the relative weight, with which each transfer function has to be minimized. In the above example, if T_1 (resp. T_2) is the transfer function between the first (resp. second) disturbance input and the first output, the issue is to minimize γ under the following constraints on the frequency interval [5 rad/s, 25 rad/s]:

$$|T_1(j\omega)| \leq 0.001\gamma \quad (8)$$

$$|T_2(j\omega)| \leq 0.001\gamma \quad (9)$$

Note that the templates could depend on frequency, and that two different weights could have been chosen. If the following specification was added:

```
num=Gain_roll_off;
den=conv([1/omega_roll_off 1],[1/omega_roll_off/omega_roll_off 1.4/omega_roll_off 1]);
[a,b,c,d]=tf2ss(num,den);
sys=ss(a,b,c,d);
%
specif{3}.input=[1 2];
specif{3}.output=[10 11];
```

```
specif{3}.list=3:length(puls_specif);
specif{3}.template=squeeze(abs(freqresp(sys,puls_specif(3:end))));
specif{3}.obj=1;  % AN ADDITIONAL MINIMIZATION OBJECTIVE
specif{3}.lin_log_x=1;
specif{3}.lin_log_y=1;
specif{3}.interp=1;
```

The issue would be to minimize γ under the following constraints:

$$|T_1(j\omega)| \leq 0.001\gamma \quad (10)$$
$$|T_2(j\omega)| \leq 0.001\gamma \quad (11)$$
$$\overline{\sigma}(T_3(j\omega)) \leq \gamma\alpha(\omega) \quad (12)$$

where $T_3(s)$ and $\alpha(\omega)$ are defined by $specif\{3\}$.

4.4.5 H_2 minimization

It corresponds to $specif\{i\}.obj = 2$. Here is an example:

```
specif{4}.input=1;
specif{4}.output=1;
specif{4}.wmin=5;
specif{4}.wmax=25;
specif{4}.weight=1;
specif{4}.obj=2;
%
specif{5}.input=2;
specif{5}.output=1;
specif{5}.wmin=5;
specif{5}.wmax=25;
specif{5}.weight=2;
specif{5}.obj=2;
```

$specif\{i\}.wmin$ and $specif\{i\}.wmax$ define the frequency interval, on which the H_2 norm is computed. $specif\{i\}.weight$ allows to weight the minimization of the H_2 norm of each transfer function, when different transfer functions are simultaneously considered. In the above example if T_1 (resp. T_2) is the transfer function between the first (resp. second) disturbance input and the first output, the following quantity is minimized:

$$\int_5^{25} |T_1(j\omega)|^2 d\omega + 2\int_5^{25} |T_2(j\omega)|^2 d\omega \quad (13)$$

$specif\{i\}.weight$ must be a positive scalar. Nevertheless, let J be the above quantity: note that the "cumulative H2 norm", which is displayed on screen is $\frac{\sqrt{J}}{2\pi}$, i.e. it's homogeneous to an H_2 norm.

4.4.6 H_2 constraint

The only differences with the above subsection are $specif\{i\}.template$, which replaces $specif\{i\}.weight$, and $specif\{i\}.obj = 3$. Here is an example:

```
specif{4}.input=1;
specif{4}.output=1;
specif{4}.wmin=5;
specif{4}.wmax=25;
specif{4}.obj=3;
specif{4}.template=0.13;
```

The corresponding constraint is:

$$\sqrt{\frac{1}{2\pi}\int_5^{25} |T_1(j\omega)|^2 d\omega} \leq 0.13 \tag{14}$$

$specif\{i\}.template$ must be a positive scalar.

4.4.7 Summary

The following table summarizes the design specifications:

nature of the spec.	obj	list of required fields in specif
H_∞ constraint	0	obj, input, output, list, template, lin_log_x, lin_log_y, interp
H_∞ minimization	1	obj, input, output, list, template, lin_log_x, lin_log_y, interp
H_2 minimization	2	obj, input, output, wmin, wmax, weight
H_2 constraint	3	obj, input, output, wmin, wmax, template

Several H_∞ (resp. H_2) minimization objectives can be accounted for, but it is not possible to simultaneously minimize H_∞ and H_2 objectives. The special case of the H_2 minimization of an objective under H_2 constraints is not allowed.

4.5 Convex optimization with a cutting planes method

To simplify the exposition we assume that the problem is to minimize γ under the constraints:

$$\begin{aligned} \overline{\sigma}(T_1(j\omega,\theta)) &\leq \gamma\ \alpha_1(\omega) && (15)\\ \overline{\sigma}(T_2(j\omega,\theta)) &\leq \alpha_2(\omega) && (16) \end{aligned}$$

where $T_i(j\omega,\theta)$ are affine functions of θ. Thus, there is an H_∞ minimization objective and an H_∞ constraint. The case of H_2 specifications is simpler to explain.

4.5.1 Principle of the cutting planes method

First note that the convex constraint (16) can be approximated at $\theta = \theta_0$ by an affine one:

$$\overline{\sigma}(T_2(j\omega,\theta_0)) + S^T(\theta-\theta_0) \leq \alpha_2(\omega) \qquad (17)$$

where S is called a subgradient. Indeed, for all θ:

$$\overline{\sigma}(T_2(j\omega,\theta_0)) + S^T(\theta-\theta_0) \leq \overline{\sigma}(T_2(j\omega,\theta)) \qquad (18)$$

When approximating (16) at different points $\theta = \theta_i$ and at different frequencies $\omega = \omega_i$ all these affine constraints (17) can be stacked into an LP constraint $A\theta \leq b$. In the same way, the convex constraint (15) can be approximated at $\theta = \theta_0$ by:

$$\overline{\sigma}(T_1(j\omega,\theta_0)) + S^T(\theta-\theta_0) \leq \gamma\ \alpha_1(\omega) \qquad (19)$$

Let γ^* be the minimal value of γ under constraints (15,16) on a frequency interval. Using the above approximations a lower bound of γ^* can be computed as the minimal value of γ under the LP constraints:

$$A_{aug} \begin{bmatrix} \theta \\ \gamma \end{bmatrix} \leq b_{aug} \qquad (20)$$

An upper bound of γ^* is also computed, noting that any value of θ which satisfies the H_∞ constraint (16) provides a value of γ. Thus, the principle of the cutting planes algorithm is as follows:

1. Compute the value $\tilde{\theta}$ of θ which minimizes γ under the LP constraint (20). Let γ_{lb} be the associated minimal value of γ.

2. Compute an upper bound γ_{ub} of γ^*. If the gap between the bounds is close enough stop. Otherwise approximate convex constraints (15,16) at $\theta = \tilde{\theta}$ and at the critical frequencies, where constraints are the most violated. Return to step 1.

Thus, the issue is to refine the affine approximation of the initial convex optimization problem, until γ^* is computed with a satisfactory accuracy.

The case of an H_2 minimization objective or constraint is simpler (see the end of section 3.2), since it reduces either to the minimization of γ under the constraint:

$$\theta^T Q \theta + f^T \theta + f_0 \leq \gamma$$

or to the constraint (C is fixed):

$$\theta^T Q \theta + f^T \theta + f_0 \leq C$$

In the same way as $\overline{\sigma}$ constraints these quadratic constraints can be approximated as affine ones.

4.5.2 The algorithm

Here again, we first consider the simplified optimization problem (15,16). The H_∞ specifications are defined on a continuum of frequencies. Nevertheless, *the poles of the closed loop transfer function* $T_1(s)+T_2(s)Q(s)T_3(s)$ *are a priori known*, so that it is possible to define a fine frequency gridding which is a satisfactory approximation of the continuum (i.e. if the constraints are satisfied on the gridding, they are most probably satisfied on the continuum).

The simplest solution would be to directly apply the cutting planes algorithm on this fine frequency gridding, but it can be time consuming, especially in the case of flexible systems with numerous bending modes since the required frequency gridding can be very large. A more efficient solution is to iteratively refine the frequency gridding, which is used by the cutting planes algorithm:

1. *Initialization*: an a priori fixed fine frequency gridding is defined for validation of H_∞ specifications (and also for H_2 ones if any). A second rough frequency gridding is defined as the initial design gridding.

2. *Convex optimization over the design frequency gridding*: at the end of the optimization H_∞ specifications (15,16) are satisfied at each point of the design frequency gridding for $\gamma = \gamma^*$, where γ^* is the minimized value of the H_∞ objective.

3. *Validation of H_∞ specifications over the fixed fine frequency gridding*: the issue is to check whether the H_∞ specifications (15,16) are still satisfied for $\gamma = \gamma^*$ at each point of the fine frequency gridding (with some numerical tolerance, e.g. 1 %). If yes stop. Otherwise just choose for each H_∞ constraint or minimization objective the critical frequency, where the constraint or minimization objective is the most violated. Add this set of critical frequencies to the design gridding and go back to step 2.

It is worth emphasizing that the final design gridding is usually much smaller than the one used for validation. The value of γ^* increases at each iteration, since a larger number of frequencies and thus of constraints is considered. Moreover, subgradients are kept from one optimization to an other: if the LP constraints $A_1\theta \leq b_1$ were obtained at the end of the first optimization, with a rough frequency gridding, the initial value of the LP constraints at the beginning of the second one will still be $A_1\theta \leq b_1$, in order to keep all information from the first optimization. This trick enables to save a large amount of computational time.

The case of H_2 specifications is very simple, since these ones already correspond to the fine frequency gridding. The associated quadratic criterion corresponds to a single minimization objective or to a single quadratic constraint (which is independent of frequency). Note finally that it would be possible to avoid the use of a fine frequency gridding using a frequency sweeping technique described in [8]. But remember the poles of the closed loop transfer function $T_1(s) + T_2(s)Q(s)T_3(s)$ are a priori known, so that it is possible to refine as much as necessary the gridding around the natural frequencies of the *closed loop* flexible modes, if any.

4.5.3 One-shot vs progressive design

A progressive design is possible for the case of a large number of optimization parameters. If there are e.g. 80 optimization parameters, only the first 8 optimization parameters will be used. Then the first 16 optimization parameters are used... Until all 80 optimization parameters are used. The decrease of the minimized objective is visualized as a function of the number of optimization parameters. Subgradients are kept from an optimization to an other in order to save a large amount of computational time.

4.5.4 On screen

We just explain with a few words what appears on screen during convex optimization:

```
SEARCH OF A FEASIBLE POINT

    1: constr_max = 3.233e+000 (8 variables)
    2: constr_max = 3.206e+000 (8 variables)
    3: constr_max = 3.311e+000 (8 variables)
    4: constr_max = 2.272e+000 (8 variables)
    5: constr_max = 1.282e+001 (8 variables)
    6: constr_max = 5.959e+000 (8 variables)
    7: constr_max = 8.121e+000 (8 variables)
    8: constr_max = 1.321e+001 (8 variables)
    9: constr_max = 8.985e-001 (8 variables)


MINIMIZATION OF THE OBJECTIVE

      1: 0.000e+000 <= gamma <= 3.835e+001 (8 variables)
      2: 1.375e+001 <= gamma <= 3.835e+001 (8 variables)
      3: 2.697e+001 <= gamma <= 3.835e+001 (8 variables)
      4: 2.982e+001 <= gamma <= 3.835e+001 (8 variables)
      5: 3.072e+001 <= gamma <= 3.478e+001 (8 variables)
      6: 3.121e+001 <= gamma <= 3.478e+001 (8 variables)
      7: 3.169e+001 <= gamma <= 3.478e+001 (8 variables)
      8: 3.185e+001 <= gamma <= 3.478e+001 (8 variables)
      9: 3.190e+001 <= gamma <= 3.478e+001 (8 variables)
     10: 3.215e+001 <= gamma <= 3.392e+001 (8 variables)
     11: 3.220e+001 <= gamma <= 3.392e+001 (8 variables)
     12: 3.238e+001 <= gamma <= 3.392e+001 (8 variables)

Minimization objective between 32.379 and 33.921 (4.546e-002 percent)
```

A feasible point is first computed. A convex optimization problem without constraints is solved. *constr_max* represents the maximal value of the normalized H_∞/H_2 constraints. The feasibility algorithm stops as soon as *constr_max* is less than 1, which means that the constraints are feasible. The objective is now minimized. An interval is computed for the minimized objective "gamma", see e.g. "$1.375e+001 <= gamma <= 3.835e+001$".

4.5.5 Advanced tuning of the optimization algorithm

Advanced tuning parameters can be found in the file *design_tuning.m*, and especially:

- Display of intermediate results inside the convex optimization algorithm.
- Minimal gap between the lower and upper bounds of the minimized objective. The algorithms stops as soon as this gap is achieved.

- *eps_cvge_cutting_plane*: ideally it should be zero. A small positive value may increase the speed of the algorithm, but it's dangerous. The algorithm may conclude that the constraints are not feasible, even if they are feasible. The idea is indeed to slightly over-constraint the problem in the LP solver.
- Maximal infinity norm of the vector of optimization parameters: ideally it should be ∞, but it's dangerous for the algorithm. Moreover, if big numerical problems occur with the LP solver, constraining this norm may be a solution. In any case, at the end of the optimization the infinity norm of the optimal vector of optimization parameters is given. If it is strictly less than the maximal norm this means that this constraint was not active.

4.6 Description of *obs_state_feedback.mat*

This file contains an initial controller under an observed-state feedback form. It must contain:

- *sys_open_loop*, $feedback\_inputs$, $feedback\_outputs$: see section 4.1.
- K, L: stabilizing state-feedback and observer gains.
- *sys_Q*: an initial value of the Youla parameter.
- *sys_K*: the corresponding value of the feedback controller $K(s)$.

Let:

```
A=sys_open_loop.a
B=sys_open_loop.b(:,feedback_inputs)
C=sys_open_loop.c(feedback_outputs,:)
D=sys_open_loop.d(feedback_outputs,feedback_inputs)
```

D must be zero since the open loop transfer matrix between the plant inputs and outputs used by feedback must be strictly proper. The closed loop state-matrices $A - BK$ and $A - LC$ are supposed to have all their eigenvalues in the Left Half Plane. *sys_Q* is supposed to be a STABLE transfer matrix, whose I/O dimensions are the same as those of the feedback controller. In our example the zero matrix gain *sys_Q* was computed as:

```
sys_Q=ss([],[],[],zeros(length(feedback_inputs),length(feedback_outputs)));
```

Note that the corresponding initial controller *sys_K* uses a POSITIVE feedback.

4.7 Description of *youla_file.mat*

This file contains an initial value *sys_Q* of the Youla parameter, a structure *youla* which will be described below, and *specif_inputs* and *specif_outputs* which are the lists of the open loop plant I/O used by the design specifications. These 2 lists are used to compute the Youla parameterization in *youla*. In our example:

```
specif_inputs=3:4;
specif_outputs=1:9;
```

If all plant outputs may be used by the design spec., as in the example above, the plant inputs used by the feedback controller cannot be used by the spec. The fields of the structure *youla* are:

1. *youla.sys_T*: a state-space representation of $T(s) = \begin{bmatrix} T_1(s) & T_2(s) \\ T_3(s) & 0 \end{bmatrix}$.
2. *youla.idxI2* and *youla.idxO2* are the lists of inputs and outputs of T(s), corresponding to the feedback with $Q(s)$.
3. *youla.idxI1* and *youla.idxO1* are the lists of inputs and outputs of T(s), corresponding to the design specifications.
4. *youla.puls_valid* is a fine frequency gridding that will be used for validation and H_2 spec. See *Routines_CCD/comp_puls_valid.m* for its computation.

As for the design specifications, sub-transfer matrices will be shaped inside $T_1(s) + T_2(s)Q(s)T_3(s)$, so that it's necessary to know the meaning of these closed loop inputs and outputs, i.e. to which open loop plant inputs and outputs they correspond.

5 Convex design of a gain-scheduled LFT controller

This section presents Matlab computational tools for computing a gain-scheduled feedback controller under an LFT form, on the basis of an open loop LFT model, using an extension of the Youla-parameterization to the LFT case. The technique, which is described in [7], is illustrated on a nonlinear missile example [13] in the directory *Demo_LFT_missile*, which contains 3 main files:

- *script_1_LFT_initial_controller.m*: design of an LFT state-feedback observer with *calc_state_feedback_obs_lft_one_shot.m* and *calc_state_feedback_obs_lft_valid.m* (in *Routines_CCD/**). The main input data is the open loop missile LFT model, loaded from *missile_LTI_open_loop.mat*, which describes the linearizations at a continuum of trim points parameterized by Mach and the angle of attack α [1]. The Simulink file *open_loop_missile.mdl* is also used. The result is saved in *sf_obs_controller.mat*.

- *script_2_LFT_Youla_parameterization.m*: computation of the Youla parameterization on the basis of the initial controller, loaded from *sf_obs_controller.mat*. The file *closed_loop_youla.mdl* is used. The result is saved in *youla.mat*.

- *script_3_LFT_design_Q.m*: design of the Youla parameter $Q(s)$ on the LFT model. $Q(s)$ is synthesized on a gridding of models using *youla_lft_design_one_shot.m* (in *Routines_CCD/**). The associated tuning parameters are extracted from *design_tuning.m*. The Youla parameterization is loaded form *youla.mat*. $Q(s)$ is validated on a (larger) gridding of models and on the continuum of models using *youla_lft_validation_1/2.m* (in *Routines_CCD/**).

The Optimization Toolbox, the System Control Toolbox, the Robust Control Toolbox and Simulink are required, as well as the LMI Control Toolbox for the synthesis of the initial observed state feedback LFT controller.

A few additional remarks:

- The LFT continuum of closed loops is validated using *mu_margin.m*, which is extracted from the Skew Mu Toolbox [9].

- The missile example is a bit complexified by the presence of an integrator on the output α: since the state of the integrator can be measured, we have chosen to introduce the integrator in the state-feedback design, but not in the observer model. A simpler solution would be to introduce this integrator in the observer model, i.e. an LFT observed state-feedback controller could be computed on the basis of a generic augmented model with the additional output $\int \alpha$.

- Since the overall design is convex, it would be possible to iteratively compute the LFT feedback controller with guaranteed convergence properties: the controller is synthesized on a finite set of design models, then it is validated on the

[1] *missile_LTI_open_loop.mat* contains *sys_M* and *blk*. *blk* describes the structure of the model perturbation Δ (musyn format). *sys_M* is a state-space representation of $M(s)$, whose first I/O correspond to the interconnection with Δ. The last input is the physical one, i.e. the control input, while the last 3 outputs are Nz, q, α.

LFT continuum (i.e. $\forall \delta \in D$, where δ is the vector of normalized scheduling parameters and D is the unit hypercube). If spec. are not satisfied on the continuum, a worst-case model is added to the set of design models...
But validating the controller on the LFT continuum is (very) time-consuming, so that our choice was a one-shot design on a finite set of models, followed by a validation on the continuum a posteriori. In this context, it is worth strengthening the spec. in the one-shot design, with respect to the validation step. Another possibility is to choose worst-case models inside kD, with $k > 1$, during the multi-model design stage, and then to check if the spec. are satisfied $\forall \delta \in D$. For an example, see the design of the observer gain in $Demo\_missile/script\_1\_LFT\_initial\_controller.m$.

About the specifications:

- When synthesizing the Youla parameter on the gridding of models, $specif$ and $puls\_specif$ are defined in the same way as for an LTI feedback controller: H_2 and H_∞ spec. can be accounted for. There's only an additional field $specif\{i\}.models$ which gives the list of models, on which spec. $\# i$ must be satisfied. Logically, this spec. is to be satisfied on all models of the gridding since it is to be satisfied on the continuum. Nevertheless, if an additional aim is to retune the gain-scheduled controller at specific trim points, additional spec. may be added which are to be satisfied only at specific points described by the field $specif\{i\}.models$.
- When validating the Youla parameter on the LFT continuum, only H_∞ spec. can be accounted for. Computing a worst-case H_2 performance level is a much more difficult problem than computing a worst-case H_∞ performance level.

References

[1] H. Akcay and B. Ninness. Orthonormal basis functions for continuous-time systems and Lp convergence. *Mathematics of Control, Signals and Systems*, 12(3):295–305, 1999.

[2] D. Alazard and P. Apkarian. Observer-based structures of arbitrary compensators. *International Journal of Robust and Nonlinear Control*, 9(2):101–118, february 1999.

[3] D.J. Bender and R.A. Fowell. Computing the estimator-controller form of a compensator. *International Journal of Control*, 41(6):1565–1575, 1985.

[4] D.J. Bender and R.A. Fowell. Some considerations for estimator-based compensator design. *International Journal of Control*, 41(6), 1985.

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

[6] G. Ferreres. *A practical approach to robustness analysis with aeronautical applications*. Springer Verlag, 1999.

[7] G. Ferreres and P. Antoinette. Convex gain-scheduled control of an LFT model. *Proc. of the ECC*, 2009.

[8] G. Ferreres and J.M. Biannic. A μ analysis technique without frequency gridding. *Proc. of the ACC*, 4:2294–2298, 1998.

[9] G. Ferreres and J.M. Biannic. A Skew Mu Toolbox (SMT) for robustness analysis. *available on the authors' homepages*, 2003-2009.

[10] G. Ferreres and G. Puyou. Flight control law design for a flexible aircraft: limits of performance. *Journal of Guidance, Control and Dynamics*, 29(4):870–878, July-August 2006.

[11] C.Y. Kao, A. Megretski, and U.T. Jonsson. An algorithm for solving optimization problems involving special frequency dependent LMIs. *Proceedings of the ACC*, pages 307–311, 2000.

[12] G. Puyou, G. Ferreres, C. Chiappa, and P. Menard. Multiobjective method for flight control law design. *Proc. of the AIAA GNC Conference*, 2004.

[13] R.T. Reichert. Dynamic scheduling of modern robust control autopilot design for missiles. *IEEE Control System Magazine*, 12(5):35–42, October 1992.