A Matlab Toolbox for System Analysis via Integral Quadratic Constraints

Jönsson, Ulf; Rantzer, Anders

1997

Document Version:
Publisher's PDF, also known as Version of record

ISSN 0280–5316
ISRN LUTFD2/TFRT--7556--SE

A Matlab Toolbox for System Analysis via Integral Quadratic Constraints

Ulf Jönsson
Anders Rantzer

Department of Automatic Control
Lund Institute of Technology
January 1997

Department of Automatic Control
Lund Institute of Technology
Box 118
S-221 00 Lund Sweden

Document name
INTERNAL REPORT

Date of issue
January 1997

Document Number
ISRN LUTFD2/TFRT--7556--SE

Author(s)
Ulf Jönsson and Anders Rantzer

Supervisor

Sponsoring organisation

Title and subtitle
A Matlab Toolbox for System Analysis via Integral Quadratic Constraints

Abstract

This report is a manual for a Matlab toolbox for system analysis with integral quadratic constraints.

Key words
Matlab, LMI, Stability, Performance, Integral Quadratic Constraint

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title
0280–5316

ISBN

Language
English

Number of pages
43

Recipient's notes

Security classification

1. Introduction

This aim of this report is to describe a Matlab toolbox consisting of a set of commands for robustness analysis with Integral Quadratic Constraints (IQC). The first version of the toolbox was developed in the late summer of 1994. Since then there have been a continuous development of the toolbox, which has been driven by our research needs and by the development of new versions of the LMI-Lab part of the Matlab LMI Control Toolbox, [3]. The data structures and the command library in the toolbox are to some extent inspired by μ-**Tools**. The toolbox will for this reason go under the name IQCtools in this report.

The toolbox supports the use of IQCs for stability analysis and performance analysis of control systems. The basic framework behind the toolbox is the particular use of IQCs for robustness analysis that was suggested by Megretski in [8].

How to use the Toolbox

The toolbox is built on top of LMI-Lab and μ-**Tools**, see [3] and [2], respectively. These packages are necessary when using the toolbox. The IQCtools commands are provided in two directories:

`IQCtools/commands`: Contains the IQCtools user commands

`IQCtools/subs`: Contains subroutines used by the IQCtools commands

General information on the commands in IQCtools can be obtained with the Matlab commands `help IQCtools/commands` and `help IQCtools/subs`.

The directories `IQCtools/commands` and `IQCtools/subs` can at the Department of Automatic Control, Lund, Sweden, be put in your `MATLABPATH` with the commands

```
>>path(path,'/home/ulfj/matlab/IQCtools/commands');
>>path(path,'/home/ulfj/matlab/IQCtools/subs');
```

A Note

The toolbox should in the present form be regarded mainly as a research tool and not a fully developed software package. It may be subject to future development and improvements.

About this report

This report is a manual for IQCtools. Section 2 contains a brief survey of basic IQC-based robustness analysis. Then Section 3 describes the ideas that form the basis for the implementation of the toolbox. The reader who is familiar with, for example, [9], [5], and [4] can skip these two sections without any essential loss. Section 4 contains a tutorial for the toolbox. The main part of the tutorial consists of three numerical examples. Comments and suggestions for future improvements are given in Section 5, and finally, Section 7 contains the command reference for the toolbox.

Notation

M^*	Hermitian conjugation.
$\lvert\cdot\rvert$	The Euclidean norm $\lvert x\rvert = \sqrt{x^T x}$.

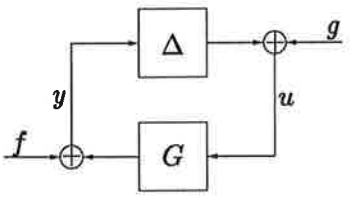

Figure 1 System consisting of a nominal plant G and a bounded causal perturbation Δ.

$\mathbf{RL}_\infty^{m\times m}$ The proper real rational matrix functions with no poles on the imaginary axis. If $H \in \mathbf{RL}_\infty^{m\times m}$ then the adjoint is defined as $H^*(s) = H(-s)^T$.

$\mathbf{RH}_\infty^{m\times m}$ The subspace of $\mathbf{RL}_\infty^{m\times m}$ consisting of functions with no poles in the closed right half plane.

P_T The projection operator defined by $P_T u(t) = u(t)$ when $t \leq T$ and $P_T u(t) = 0$ when $t > T$.

$\mathbf{L}_2^m[0,\infty)$ The space of $\mathbf{R}^m$ valued square integrable functions with norm

$$\|u\|^2 = \int_0^\infty |u(t)|^2 dt.$$

$\mathbf{L}_{2e}^m[0,\infty)$ The vector space of functions f with $P_T f \in \mathbf{L}_2^m[0,\infty)$ for all $T > 0$.

An operator $H : \mathbf{L}_{2e}^m[0,\infty) \to \mathbf{L}_{2e}^m[0,\infty)$ is said to be causal if $P_T H P_T = P_T H$ for all $T \geq 0$. This means that the value at a certain time instant does not depend on future values of the argument. A causal operator H on $\mathbf{L}_{2e}^m[0,\infty)$ is bounded if $H(0) = 0$ and if the gain defined as

$$\|H\| = \sup_{\substack{u\in\mathbf{L}_2^m[0,\infty)\\ u\neq 0}} \frac{\|Hu\|}{\|u\|} \tag{1}$$

is finite, see [11] for a more detailed discussion.

An operator H with transfer function in $\mathbf{RH}_\infty$ is causal and bounded.

2. Introduction to IQC Based Robustness Analysis

We will in this section give a brief introduction to IQC-based robustness analysis. Our presentation will be limited to the concepts that are used in the toolbox. For further details on the material in this and the next section we refer to [9] and [4].

We consider systems consisting of a bounded causal operator G in a positive feedback interconnection with a bounded causal operator Δ, see Figure 1. We will assume that G has a rational transfer function.

The idea behind the IQC approach for robustness analysis is to find a description of Δ in terms of multipliers Π, that satisfy

$$\int_{-\infty}^{\infty} \begin{bmatrix} \widehat{y}(j\omega) \\ \widehat{v}(j\omega) \end{bmatrix}^* \Pi(j\omega) \begin{bmatrix} \widehat{y}(j\omega) \\ \widehat{v}(j\omega) \end{bmatrix} d\omega \geq 0, \tag{2}$$

for all square integrable y and $v = \Delta(y)$. Here $\widehat{y}$ and $\widehat{v}$ denotes the Fourier transforms of y and v, respectively. It is assumed that Π is a rational transfer function matrix satisfying the condition $\Pi(s) = \Pi(-s)^T$ for all $s \in \mathbf{C}$. This can also be formulated as $\Pi = \Pi^* \in \mathbf{RL}_\infty^{(l+m)\times(l+m)}$, where l and m are the number of inputs and outputs of Δ, respectively.

This way of defining multipliers excludes the use of the often very useful Popov multipliers. We will therefore generally consider multipliers consisting of a proper part $\Pi_B = \Pi_B^* \in \mathbf{RL}_\infty^{(l+m)\times(l+m)}$ and a nonproper Popov multiplier on the form

$$\Pi_P(j\omega) = \begin{bmatrix} 0 & -j\omega\Lambda^T \\ j\omega\Lambda & 0 \end{bmatrix}, \tag{3}$$

where $\Lambda \in \mathbf{R}^{m\times l}$. Popov multipliers give useful descriptions of static nonlinearities, parametric uncertainties and of slowly time-varying parameters.

We use the Popov multiplier in (3) to define constraints involving the integral

$$\int_0^\infty 2(\Delta(y))^T \Lambda \dot{y} dt. \tag{4}$$

We can now combine the bounded multiplier Π_B with a Popov multiplier to get the following definition of IQC

DEFINITION 1
We say that Δ satisfies the IQC defined by $\Pi = \Pi_B + \Pi_P$ if there exists a positive constant γ such that

$$\int_{-\infty}^{\infty} \begin{bmatrix} \widehat{y}(j\omega) \\ \widehat{v}(j\omega) \end{bmatrix}^* \Pi_B(j\omega) \begin{bmatrix} \widehat{y}(j\omega) \\ \widehat{v}(j\omega) \end{bmatrix} d\omega + \int_0^\infty 2v^T \Lambda \dot{y}\, dt \geq -\gamma |y(0)|^2,$$

for all y and $v = \Delta(y)$ such that $y, \dot{y}, v \in \mathbf{L}_2^m[0,\infty)$. □

We note that in order to use the Popov multiplier we must ensure differentiability of y. This enforces a condition on strict properness of the nominal transfer function that would be overly restrictive in applications when Λ is a sparse matrix. This problem can be overcome. In fact, it is only necessary to differentiate the components of y that contributes to the integral (4). This means that the Popov multiplier imposes the following condition on strict properness of the nominal plant

$$\Lambda G(\infty) = 0. \tag{5}$$

For more details, see [4].

Combination of Multipliers

In order to obtain the most accurate robustness condition we need to find as many multipliers for the perturbation Δ as possible. The next two properties can be used to combine multipliers for this purpose.

Property 1 Assume that Δ satisfies the IQCs defined by $\Pi_1, \ldots, \Pi_n$. Then Δ also satisfies the IQC defined by the conic combination $\sum_{i=1}^n \alpha_i \Pi_i$, where $\alpha_i \geq 0$, $i = 1, \ldots, n$.

Property 2 Assume that $\Delta = \mathrm{diag}(\Delta_1, \ldots, \Delta_n)$, and that, for $i = 1, \ldots, n$, Δ_i satisfies the IQC defined by

$$\Pi_i = \begin{bmatrix} \Pi_{i(11)} & \Pi_{i(12)} \\ \Pi_{i(12)}^* & \Pi_{i(22)} \end{bmatrix},$$

where the block partitions are consistent with the number of inputs and outputs of the perturbations Δ_i. Then Δ satisfies the IQC defined by

$$\mathrm{daug}(\Pi_1, \ldots, \Pi_n) = \left[\begin{array}{ccc|ccc} \Pi_{1(11)} & & & \Pi_{1(12)} & & \\ & \ddots & & & \ddots & \\ & & \Pi_{n(11)} & & & \Pi_{n(12)} \\ \hline \Pi_{1(12)}^* & & & \Pi_{1(22)} & & \\ & \ddots & & & \ddots & \\ & & \Pi_{n(12)}^* & & & \Pi_{n(22)} \end{array}\right].$$

This is easily seen by writing out the expression for the IQC in Definition 1.

It follows from the first property that the set of multipliers that describe Δ is a convex cone. We use the notation Π_Δ for any such convex cone of multipliers that describe Δ. In applications it is important to find an as accurate as possible description of Δ. A description of Δ can be improved by addition and augmentation of convex cones of multipliers.

Addition: If Δ is described by the convex cones $\Pi_{i\Delta}$, $i = 1, \ldots, n$, then Δ is also described by $\sum_{i=1}^n \Pi_{i\Delta} = \{\sum_{i=1}^n \Pi_i : \Pi_i \in \Pi_{i\Delta}\}$.

Diagonal Augmentation: If, for $i = 1, \ldots, n$, Δ_i is described by the convex cone Π_{Δ_i}, then $\Delta = \mathrm{diag}(\Delta_1, \ldots, \Delta_n)$ is described by the convex cone

$$\mathrm{daug}(\Pi_{\Delta_1}, \ldots, \Pi_{\Delta_n}) = \{\mathrm{daug}(\Pi_1, \ldots, \Pi_n) : \Pi_i \in \Pi_{\Delta_i}\}.$$

We illustrate with an example

EXAMPLE 1
Let $\Delta = \mathrm{diag}(\varphi, \delta I)$, where

1. φ is a memoryless nonlinearity satisfying the sector condition $0 \leq \varphi(x)x \leq x^2$, for all $x \in \mathbf{R}$,
2. δI is an repeated uncertain real parameter with $\delta \in [-1, 1]$.

It is easy to see that the sector nonlinearity φ satisfies the IQCs defined by the multipliers

$$\Pi_{1\varphi} = \left\{ x \begin{bmatrix} 0 & 1 \\ 1 & -2 \end{bmatrix} : x \geq 0 \right\}, \quad \Pi_{2\varphi} = \left\{ \begin{bmatrix} 0 & -j\omega\lambda \\ j\omega\lambda & 0 \end{bmatrix} : \lambda \in \mathbf{R} \right\}.$$

For the repeated uncertain parameter we use

$$\Pi_{1\delta I} = \left\{ \begin{bmatrix} X(j\omega) & 0 \\ 0 & -X(j\omega) \end{bmatrix} : X(j\omega) = X(j\omega)^* \geq 0 \right\},$$

and

$$\Pi_{2\delta I} = \left\{ \begin{bmatrix} 0 & Y(j\omega) \\ Y^*(j\omega) & 0 \end{bmatrix} : Y(j\omega) = -Y(j\omega)^* \right\}.$$

We note that the multipliers in $\Pi_{1\delta I}$ also can be used if δ is a norm bounded complex uncertainty.

The combination $\Pi = \text{daug}(\Pi_{1\varphi} + \Pi_{2\varphi}, \Pi_{1\delta I} + \Pi_{2\delta I})$ gives a convex cone of multipliers that describes the diagonal operator $\Delta = \text{diag}(\varphi, \delta I)$. It is easy to see that every $\Pi \in \Pi_\Delta$ is on the form

$$\Pi(j\omega) = \begin{bmatrix} 0 & 0 & x - j\omega\lambda & 0 \\ 0 & X(j\omega) & 0 & Y(j\omega) \\ x + j\omega\lambda & 0 & -x & 0 \\ 0 & Y^*(j\omega) & 0 & -X(j\omega) \end{bmatrix}.$$

□

Stability Analysis

Robust stability analysis of the system in Figure 1 can now be formulated as a feasibility problem on the following form:

ROBUSTNESS TEST 1—INFINITE-DIMENSIONAL FEASIBILITY TEST
Find $\Pi \in \Pi_\Delta$ such that

$$\begin{bmatrix} G(j\omega) \\ I \end{bmatrix}^* \Pi(j\omega) \begin{bmatrix} G(j\omega) \\ I \end{bmatrix} < 0,$$

for all $\omega \in [0, \infty]$. □

Here Π_Δ corresponds to a convex cone of multipliers that gives an IQC description of the operator Δ.

There are some technical well-posedness conditions that must be verified before using Robustness Test 1. However, apart from the strict properness condition in (5), these conditions are trivially satisfied in most practical applications. We refer to [9] and [4] for further details.

Robust Performance Analysis

For robust performance analysis we consider the system described by the equations

$$\begin{bmatrix} y \\ z \end{bmatrix} = G \begin{bmatrix} u \\ w \end{bmatrix}, \qquad u = \Delta(y). \tag{6}$$

Here Δ is a bounded causal operator and G is a bounded causal operator with rational transfer function. We assume that G is block partitioned according to the size of the signals

$$G = \begin{bmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{bmatrix} \in \mathbf{RH}_\infty^{(l+p)\times(m+q)}.$$

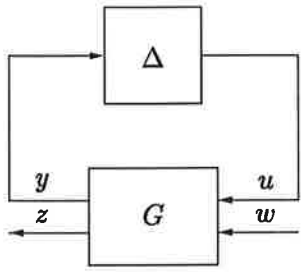

Figure 2 System setup for robust performance analysis.

The system (6) can be illustrated with the block diagram in Figure 2.

The performance of the system in (6) is generally measured in terms of disturbance attenuation. A measure of this attenuation can, for example, be obtained in terms of the energy ratio of the *error signal* z and the *disturbance input* w. It is important to exploit the spectral characteristics of the disturbance w when studying such a performance measure. We will as it has been suggested in, for example, [7], [8], and [10], study performance criteria in terms of disturbance inputs from a set $\mathcal{W}_{\text{inp}} \subset \mathbf{L}_2^q[0,\infty)$. This set is assumed to be defined by the convex cone $\Upsilon_{\text{inp}} \subset \mathbf{RL}_\infty^{q\times q}$ in the following way

$$\mathcal{W}_{\text{inp}} = \{w \in \mathbf{L}_2^q[0,\infty) : \int_{-\infty}^{\infty} \widehat{w}(j\omega)^* \Upsilon(j\omega)\widehat{w}(j\omega)d\omega \geq 0,\ \forall \Upsilon \in \Upsilon_{\text{inp}}\}.$$

We note that $\mathcal{W}_{\text{inp}} = \mathbf{L}_2^q[0,\infty)$ if $\Upsilon_{inp} = \{0\}$. If we in a certain application have a set of signals $\mathcal{W}$ then we need to find a convex cone $\Upsilon_{\text{inp}} \subset \mathbf{RL}_\infty^{q\times q}$ such that $\mathcal{W} \subset \mathcal{W}_{\text{inp}}$. We consider robust performance in terms of a weighted induced $\mathbf{L}_2$-norm according to the following definition

DEFINITION 2
Assume that the signal w is in the set $\mathcal{W}_{\text{inp}} \subset \mathbf{L}_2^q[0,\infty)$ defined by the convex cone Υ_{inp}. The system in (6) is said to have robust weighted $\mathbf{L}_2$-performance level $\gamma_{\mathbf{L}_2} = \sqrt{\gamma}$ if

(i) the feedback interconnection of G_{11} and Δ is stable,

(ii) the inequality

$$\int_{-\infty}^{\infty} \begin{bmatrix} \widehat{z}(j\omega) \\ \widehat{w}(j\omega) \end{bmatrix}^* \begin{bmatrix} W(j\omega)^*W(j\omega) & 0 \\ 0 & -\gamma I \end{bmatrix} \begin{bmatrix} \widehat{z}(j\omega) \\ \widehat{w}(j\omega) \end{bmatrix} \leq 0$$

holds for all possible $z = (G_{22}+G_{21}\Delta(I-G_{11}\Delta)^{-1}G_{12})w$, with $w \in \mathcal{W}_{\text{inp}}$.

Here $W \in \mathbf{RH}_\infty^{p\times p}$ is used to obtain appropriate weighting of the output channels. □

The performance level $\gamma_{\mathbf{L}_2} = \sqrt{\gamma}$ in Definition 2 can be found by solving the optimization problem

ROBUSTNESS TEST 2—ROBUST PERFORMANCE TEST

$$\inf \gamma \quad \text{subject to} \tag{7}$$

$$\begin{cases} \exists \Pi \in \Pi_\Delta, \Upsilon \in \Upsilon_{\text{inp}}, \quad \text{such that} \\ \begin{bmatrix} G(j\omega) \\ I \end{bmatrix}^* \text{daug}(\Pi, \begin{bmatrix} W^*W & 0 \\ 0 & \gamma I + \Upsilon \end{bmatrix})(j\omega) \begin{bmatrix} G(j\omega) \\ I \end{bmatrix} < 0, \quad \forall \omega \in [0,\infty], \end{cases}$$

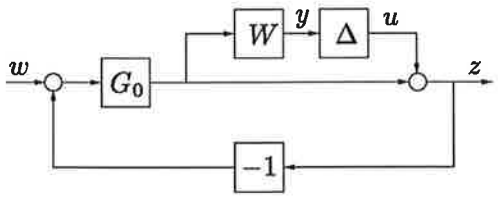

Figure 3 System for Example 2.

□

REMARK 1
The convex cone Π_Δ contains multipliers that describes the operator Δ. Each multiplier from Π_Δ is generally on the form $\Pi = \Pi_B + \Pi_P$. If Λ is the Popov parameter in Π_P then the following constraint needs to be satisfied

$$\Lambda \left[G_{11}(\infty) \quad G_{12}(\infty) \right] = 0.$$

□

REMARK 2
There are technical well-posedness conditions that should be verified before using the optimization problem in (7) to compute the performance level $\gamma_{\mathbf{L}_2} = \sqrt{\gamma}$. See, Theorem 3.2 and Remark 3.4 in [4]. □

REMARK 3
Note that it is in general only a suboptimal value of the performance level γ that can be obtained from the optimization problem in (7). □

We illustrate robust performance analysis with an example

EXAMPLE 2
We will here illustrate robust performance analysis with a simple example. Consider the system in Figure 3. The perturbation Δ is assumed to be a dynamic uncertainty with $\|\Delta\|_\infty \leq 1$, where $\|\cdot\|_\infty$ denotes the $\mathbf{H}_\infty$-norm. We assume that w belongs to the set

$$\mathcal{W}_H = \left\{ w \in \mathbf{L}_2^1[0,\infty) : |\widehat{w}(j\omega)|^2 = \frac{\|w\|^2}{\|H\|_2^2} |H(j\omega)|^2 \right\}, \tag{8}$$

where $H \in \mathbf{RH}_\infty^{1\times 1}$ is strictly proper and where the $\mathbf{H}_2$-norm is defined by

$$\|H\|_2^2 = \frac{1}{2\pi} \int_{-\infty}^{\infty} |H(j\omega)|^2 d\omega.$$

We can regard $\mathcal{W}_H$ as a set of filtered deterministic noise signals. We want to compute the performance measure

$$\sup_{\substack{w\in\mathcal{W}_H,\ \|w\|\leq 1 \\ \|\Delta\|_\infty \leq 1}} \|G_{\mathrm{cl}} w\|, \tag{9}$$

where the closed loop system operator is defined as

$$G_{\rm cl} = \frac{(I + W\Delta)G_0}{I + (I + W\Delta)G_0}.$$

The optimization problem in (9) is infinity if G_{cl} is unbounded for some Δ.

If we assume that the nominal system is stable then the system in Figure 3 can be put into the form in Figure 2 with

$$G = \frac{1}{1+G_0}\begin{bmatrix} -WG_0 & WG_0 \\ 1 & G_0 \end{bmatrix} \in \mathbf{RH}_\infty^{2\times 2}.$$

An upper bound for the optimal value in (9) can be obtained as $\gamma^{1/2}$, where γ is the solution to the optimization problem in (7) when

$$\Upsilon_{\rm inp} = \left\{ \Upsilon \in \mathbf{RL}_\infty^{1\times 1} : \int_{-\infty}^{\infty} \Upsilon(j\omega)|H(j\omega)|^2\, d\omega \geq 0 \right\}. \tag{10}$$

and

$$\Pi_\Delta = \left\{ \begin{bmatrix} x(j\omega) & 0 \\ 0 & -x(j\omega) \end{bmatrix} : x(j\omega) \geq 0,\ \forall \omega \right\}.$$

It is easy to verify that $\mathcal{W}_H \subset \mathcal{W}_{\rm inp}$, when $\Upsilon_{\rm inp}$ in (10) is used to define $\mathcal{W}_{\rm inp}$.

The resulting optimization problem is infinite-dimensional. This follows since both Π_Δ and $\Upsilon_{\rm inp}$ are infinite-dimensional convex cones. We will in the next section formulate a methodology that can be used to obtain suboptimal solutions to this optimization problem. Numerical computations for this example will be given in Section 4 □

3. LMI Formulations of Robustness Problems

We will in this section discuss a format for finite-dimensional restrictions of Robustness Test 1 and Robustness Test 2. An application of the Kalman-Yakubovich-Popov (KYP) lemma then gives an equivalent LMI formulation the restricted robustness tests. The toolbox is nothing but an interface to LMI-Lab that implements the ideas of this section.

Our means of obtaining finite-dimensional restrictions of Robustness test 1 is to introduce a format for a finite-dimensional parametrization of a convex subcone of Π_Δ. The subcone is defined in terms of a basis multiplier and a convex cone of parameters according to the following definition.

DEFINITION 3
Let

1. $\Psi = [\,\Psi_a \quad \Psi_b\,]$ be a *basis multiplier*, where Ψ_a and Ψ_b are $N \times l$ and $N \times m$ rational transfer functions,
2. $\Phi_1, \ldots, \Phi_K$ be proper rational transfer functions, and let $\Phi = [\,\Phi_1, \ldots, \Phi_K\,]$,
3. $M_{\rm struc} \subset \mathbf{R}^{N\times N}$ be a *structure matrix* with parameters,
4. $M_0 \in \mathbf{R}^{N\times N}$ be a constant matrix.

Then $\mathtt{Pi}_\Delta(\Psi, \Phi, M_{\text{struc}}, M_0) \subset \Pi_\Delta$ denotes the subcone defined as

$$\mathtt{Pi}_\Delta(\Psi, \Phi, M_{\text{struc}}, M_0) = \{\Psi^*(M_{\text{struc}} + M_0)\Psi : \\ \Phi_k(j\omega)^*(M_{\text{struc}} + M_0)\Phi_k(j\omega) < 0, \ \forall \omega \in [0, \infty], \forall k\}.$$

We note that M_{struc} contains the free parameters of $\mathtt{Pi}_\Delta(\Psi, \Phi, M_{\text{struc}}, M_0)$. □

REMARK 4
Note that the constraints on M_{struc} are defined as strict inequalities. It would in many applications be preferable with non-strict inequalities. However, the strict inequalites can be verified as strict LMIs, which are easier to treat numerically than the corresponding non-strict LMIs. □

We note that Ψ in general is nonproper due to the Popov multipliers. If we for a given $M = M_{\text{struc}} + M_0$ write out the expression for the corresponding multiplier we see that it has the structure

$$\Pi = \begin{bmatrix} \Pi_{11} & \Pi_{12} \\ \Pi_{12}^* & \Pi_{22} \end{bmatrix} = \begin{bmatrix} \Psi_a^* M \Psi_a & \Psi_a^* M \Psi_b \\ \Psi_b^* M \Psi_a & \Psi_b^* M \Psi_b \end{bmatrix}.$$

Before we give an example we notice that addition and diagonal augmentation of subcones on the the form in Definition 3 can be done in a simple way.

Addition: Let

$$\mathtt{Pi}_{1\Delta} = \mathtt{Pi}_{1\Delta}(\Psi_1, \Phi_1, M_{1\text{struc}}, M_{10}),$$

and

$$\mathtt{Pi}_{2\Delta} = \mathtt{Pi}_{2\Delta}(\Psi_2, \Phi_2, M_{2\text{struc}}, M_{20}).$$

Then addition is performed as

$$\mathtt{Pi}_{1\Delta} + \mathtt{Pi}_{2\Delta} = \mathtt{Pi}_\Delta\left(\begin{bmatrix} \Psi_1 \\ \Psi_2 \end{bmatrix}, \begin{bmatrix} \Phi_1 & 0 \\ 0 & \Phi_2 \end{bmatrix}, \begin{bmatrix} M_{1\text{struc}} & 0 \\ 0 & M_{2\text{struc}} \end{bmatrix}, \begin{bmatrix} M_{10} & 0 \\ 0 & M_{20} \end{bmatrix}\right).$$

Diagonal Augmentation: Let

$$\mathtt{Pi}_{\Delta_1} = \mathtt{Pi}_{\Delta_1}(\Psi_1, \Phi_1, M_{\text{struc}_1}, M_{0_1}),$$

and

$$\mathtt{Pi}_{\Delta_2} = \mathtt{Pi}_{\Delta_2}(\Psi_2, \Phi_2, M_{\text{struc}_2}, M_{0_2}).$$

Then diagonal augmentation is performed as

$$\text{daug}(\mathtt{Pi}_{\Delta_1}, \mathtt{Pi}_{\Delta_2}) = \mathtt{Pi}_\Delta\left(\Psi, \begin{bmatrix} \Phi_1 & 0 \\ 0 & \Phi_2 \end{bmatrix}, \begin{bmatrix} M_{\text{struc}_1} & 0 \\ 0 & M_{\text{struc}_2} \end{bmatrix}, \begin{bmatrix} M_{0_1} & 0 \\ 0 & M_{0_2} \end{bmatrix}\right)$$

where Ψ is defined as follows. If $\Psi_1 = [\,\Psi_{1a} \quad \Psi_{1b}\,]$ and $\Psi_2 = [\,\Psi_{2a} \quad \Psi_{2b}\,]$, then

$$\Psi = \begin{bmatrix} \Psi_{1a} & 0 & \Psi_{1b} & 0 \\ 0 & \Psi_{2a} & 0 & \Psi_{2b} \end{bmatrix},$$

We will next give some simple examples of parametrizations on the form in Definition 3.

EXAMPLE 3
It is easy to see that $\Pi_{1\varphi}, \Pi_{2\varphi}, \Pi_{1\delta I}$, and $\Pi_{2\delta I}$ in Example 1 have the following finite dimensional parametrizations

- The set $\Pi_{1\varphi}$ can be formulated as $\texttt{Pi}_{1\varphi}(\Psi, \Phi_1, M_{\text{struc}}, M_0)$, where

$$\Psi = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix}, \quad \Phi_1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad M_{\text{struc}} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad M_0 = 0.$$

- The set $\Pi_{2\varphi}$ can be formulated as $\texttt{Pi}_{2\varphi}(\Psi, [\,], M_{\text{struc}}, M_0)$, where

$$\Psi = \begin{bmatrix} j\omega & 0 \\ 0 & 1 \end{bmatrix}, \quad M_{\text{struc}} = \begin{bmatrix} 0 & \lambda \\ \lambda & 0 \end{bmatrix}, \quad M_0 = 0.$$

  Note that there are no constraints, i.e., Φ is empty. This follows since λ is free to take any value in $\mathbf{R}$.

- The set $\Pi_{1\delta I}$ is infinite dimensional since X can be chosen from an infinite dimensional set of proper rational transfer functions. We use the restriction $X(j\omega) = R(j\omega)^* U R(j\omega)$, where $R \in \mathbf{RH}_\infty^{N\times m}$ and where $U \in \mathbf{R}^{N\times N}$ satisfy $U = U^T \geq 0$. Here R is chosen by the user, U is the parameters that the toolbox should optimize, m is the number of repetitions of δ, and N is some integer. We get the parametrization $\texttt{Pi}_{1\delta I}(\Psi, \Phi_1, M_{\text{struc}}, M_0)$, where

$$\Psi = \begin{bmatrix} R & 0 \\ 0 & R \end{bmatrix}, \quad \Phi_1 = \begin{bmatrix} 0 \\ I \end{bmatrix}, \quad M_{\text{struc}} = \begin{bmatrix} U & 0 \\ 0 & -U \end{bmatrix}, \quad M_0 = 0.$$

- The set $\Pi_{2\delta I}$ is also infinite dimensional. Here we use the finite-dimensional restriction $Y(j\omega) = VS(j\omega) - S(j\omega)^* V^T$, where $S \in \mathbf{R}^{N\times m}$ and $V \in \mathbf{R}^{m\times N}$ for some integer N. We get $\texttt{Pi}_{2\delta I}(\Psi, [\,], M_{\text{struc}}, M_0)$, where

$$\Psi = \begin{bmatrix} I & 0 \\ 0 & I \\ 0 & S \\ -S & 0 \end{bmatrix}, \quad M_{\text{struc}} = \begin{bmatrix} 0 & 0 & V & 0 \\ 0 & 0 & 0 & V \\ V^T & 0 & 0 & 0 \\ 0 & V^T & 0 & 0 \end{bmatrix}, \quad M_0 = 0.$$

□

REMARK 5
The transfer functions $\Phi = [\,\Phi_1, \ldots, \Phi_K\,]$ that defined the constraints on the parameters in M_{struc} are often constant matrices. We have also seen that Φ may be empty, i.e., there are no constraints. □

REMARK 6
The constant matrix M_0 is often zero. This is due to the conicity of the set of multipliers Π_Δ. It can sometimes be useful to either fix some parameters in M_{struc} to a constant value or to limit the range of the parameters in M_{struc}. In both of these cases we need a nonzero M_0. □

If we restrict Robustness Test 1 to $\texttt{Pi}_\Delta(\Psi, \Phi, M_{\text{struc}}, M_0) \subset \Pi_\Delta$, then the resulting feasibility problem becomes

ROBUSTNESS TEST 3—FINITE DIMENSIONAL FEASIBILITY PROBLEM
Find a parameter matrix M_{struc} such that

$$\Phi_k^*(j\omega)(M_{\text{struc}} + M_0)\Phi_k(j\omega) < 0, \quad \forall\omega \in [0,\infty], \quad k = 0,\ldots,K,$$

where

$$\Phi_0 = \Psi \begin{bmatrix} G \\ I \end{bmatrix}.$$

□

It is possible to obtain an LMI formulation of this feasibility problem as follows. Introduce representations

$$\Phi_k = \left[\begin{array}{c|c} A_k & B_k \\ \hline C_k & D_k \end{array}\right], \quad k = 0,\ldots,K.$$

Then by the KYP lemma, the finite dimensional feasibility problem is equivalent to the following LMI problem.

ROBUSTNESS TEST 4—FEASIBILITY TEST WITH LMIS
Find a parameter matrix M_{struc} and matrices $P_k = P_k^T \in \mathbf{R}^{n_k \times n_k}$, $k = 0,\ldots K$, where $n_k = \dim(A_k)$, such that

$$\mathcal{N}_k^T \mathcal{A}_k(P_k, M_{\text{struc}})\mathcal{N}_k < 0, \quad k = 0,\ldots,K$$

where

$$\mathcal{N}_k = \left[\begin{array}{cc} A_k & B_k \\ I & 0 \\ \hline C_k & D_k \end{array}\right], \quad \mathcal{A}_k(P_k, M_{\text{struc}}) = \left[\begin{array}{cc|c} 0 & P_k & 0 \\ P_k & 0 & 0 \\ \hline 0 & 0 & M_{\text{struc}} + M_0 \end{array}\right].$$

□

REMARK 7
The usual situation is that $\Phi_k(s)$ is a constant matrix (for k=1, ... ,K). In that case, the k^{th} LMI becomes

$$\Phi_k^T(M_{\text{struc}} + M_0)\Phi_k < 0.$$

□

We can apply exactly the same ideas as above when considering robust performance problems as in Robustness Test 2. The only difference is that we need to introduce a finite dimensional restriction also for the input description Υ_{inp}. Exactly as in Definition 3, we define $\mathbf{Ups}_{\text{inp}}(\Psi_{\text{inp}}, \Phi, M_{\text{struc}}, M_0) \subset \Upsilon_{\text{inp}}$ to consist of the multipliers

$$\{\Psi_{\text{inp}}^*(M_{\text{struc}} + M_0)\Psi_{\text{inp}} : \Phi_k(j\omega)^*(M_{\text{struc}} + M_0)\Phi_k(j\omega) < 0, \ \forall\omega \in [0,\infty], \forall k\}. \tag{11}$$

A total finite dimensional multiplier description is now obtained as

$$\mathtt{Pi}_{\mathrm{Tot}} = \mathrm{daug}(\mathtt{Pi}_{\Delta}, \mathtt{Pi}_{\mathbf{L}_2} + \widehat{\mathtt{Ups}}_{\mathrm{inp}}),$$

where $\mathtt{Pi}_{\mathbf{L}_2}(\Psi, [\,], M_{\mathrm{struc}}, M_0)$ has

$$\Psi = I_{p+q}, \quad M_{\mathrm{struc}} = \begin{bmatrix} 0 & 0 \\ 0 & -\gamma I_q \end{bmatrix}, \quad M_0 = \begin{bmatrix} I_p & 0 \\ 0 & 0 \end{bmatrix},$$

and where $\widehat{\mathtt{Ups}}_{\mathrm{inp}}([\,0 \quad \Psi_{\mathrm{inp}}\,], \Phi, M_{\mathrm{struc}}, M_0)$ has the same $\Psi_{\mathrm{inp}}, \Phi, M_{\mathrm{struc}}$, and M_0 as in (11).

4. Tutorial

There are three levels of commands in the toolbox. This is illustrated in Figure 4. The purpose of the commands on each level is defined as follows:

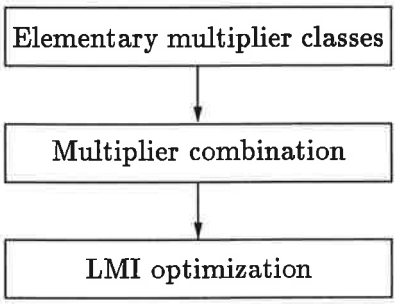

Figure 4 The three command levels in the toolbox

Elementary multiplier classes: This level of commands in the toolbox can be viewed as a library of IQCs. Each command on this level defines a finite-dimensional set of multipliers parametrized as in Definition 3.

The following commands are currently implemented at this level:

Multipliers for operators	
complexpar	uncertain complex parameter
LTIuncert	LTI dynamic uncertainty
delay	uncertain delay operator
harmonic	multiplication with harmonic oscillation
Popov	Generalized Popov multipliers
realpar	uncertain real parameter
sectorNL	sector bounded nonlinearity
slopeNL	slope restricted nonlinearity
slowtvpar	slowly time-varying parameter
tvpar	arbitrarily time-varying real parameter

Multipliers for signals	
domharmonic	signals with dominant harmonics
wspectr	signal with given spectrum

Multiplier combination: The commands on this level are used to combine multipliers from the elementary multiplier classes into a multiplier description of a diagonally structured perturbation Δ. The level consists of the following two commands:

Multiplier combination	
IQCadd	addition of multipliers
IQCdaug	diagonal augmentation of multipliers

LMI optimization: This level contains the commands for the actual stability and performance analysis. The commands in the table below transforms the analysis problem into a convex optimization problem in terms of LMIs as described in Section 3.

LMI optimization	
IQCfeas	feasibility test in terms of IQCs
IQCopt	optimization problem in terms of IQCs
IQCperf	performance analysis in terms of IQCs

We will illustrate the operation of the toolbox with a couple of simple examples. The first example is simply an application of the Popov criterion to a nonlinear systems example.

EXAMPLE 4
Consider the system in Figure 5.

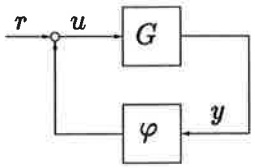

Figure 5 Feedback interconnection of a linear time-invariant plant G with a memoryless nonlinearity φ.

We assume that

$$G(s) = -\frac{s-10}{s^2+s+10}, \tag{12}$$

and that the memoryless nonlinearity satisfies the sector condition

$$0 \leq x\varphi(x) \leq x^2, \quad \forall x \in \mathbf{R}.$$

In order to obtain a multiplier description of φ, we combine the following two sets of multipliers

$$\Pi_{1\varphi} = \left\{ x \begin{bmatrix} 0 & 1 \\ 1 & -2 \end{bmatrix} : x > 0 \right\}, \text{ and } \Pi_{2\varphi} = \left\{ \begin{bmatrix} 0 & -j\omega\lambda \\ j\omega\lambda & 0 \end{bmatrix} : \lambda \in \mathbf{R} \right\}.$$

Stability for the system in Figure 5 can now be investigated by use of Robustness Test 1 with $\Pi_\Delta = \Pi_{1\varphi} + \Pi_{2\varphi}$. We obtain the stability test

ROBUSTNESS TEST 5—THE CLASSICAL POPOV CRITERION
Find $x > 0$ and $\lambda \in \mathbf{R}$ such that

$$\begin{bmatrix} G(j\omega) \\ I \end{bmatrix}^* \begin{bmatrix} 0 & x - j\omega\lambda \\ x + j\omega\lambda & -2x \end{bmatrix} \begin{bmatrix} G(j\omega) \\ I \end{bmatrix} < 0, \quad \forall \omega \in [0, \infty].$$

□

We can use the toolbox to search for suitable values of x and λ (if there exists any). The following sequence of commands does the job:

```
>> b = -[1 -10];
>> a = [1 1 10];
>> [A B C D] = tf2ss(b,a);
>> G = pck(A,B,C,D);
>>
>> Pi_1phi = sectorNL(0,1);
>> Pi_2phi = Popov(1,'0');
>> Pi_Delta = IQCadd(Pi_1phi,Pi_2phi);
>> [Mstruc,tmin] = IQCfeas(G,Pi_Delta,'FDI0');
ndec =
     5

 Solver for LMI feasibility problems L(x) < R(x)
    This solver minimizes  t  subject to  L(x) < R(x) + t*I
    The best value of t should be negative for feasibility

 Iteration    :    Best value of t so far

     1                      0.011768
     2                  5.363203e-04
     3                  5.363203e-04
     4                  5.363203e-04
     5                  2.664274e-04
     6                  7.818501e-05
     7                  7.818501e-05
     8                  1.845912e-05
     9                  1.845912e-05
    10                  2.767277e-06
    11                  2.767277e-06
    12                  5.613782e-07
    13                  5.613782e-07
    14                  1.315827e-07
```

```
    15                    2.836331e-08
    16                    9.103451e-09
* switching to QR
    17                    9.103451e-09
    18                    1.991093e-09
    19                    1.496203e-10
    20                    6.305374e-11
    21                    1.238635e-11
    22                    6.014747e-12
    23                    8.735706e-13
    24                    4.126056e-13
    25                    2.673200e-14
    26                    2.673200e-14
    27                    2.673200e-14
    28                    2.673200e-14
    29                    -5.420680e-14
***                 new lower bound:    -0.095000
    30                    -6.072912e-14

 Result:  feasible solution
          f-radius saturation:  2.336% of R =  1.00e+09
 Termination due to SLOW PROGRESS:
          t was decreased by less than 10.000% during
          the last 10 iterations.


Mstruc =
   1.0e+05 *
         0    2.4675         0         0
    2.4675         0         0         0
         0         0         0   -1.5831
         0         0   -1.5831         0
tmin =
  -6.0729e-14
>>
```

Next follows an explanation the command sequence. We refer to the command reference section for more detailed explanation of the various commands.

- The first four commands defines the transfer function G in (12) as a μ-**Tools** SYSTEM matrix data structure. All transfer functions in the IQC toolbox must be given in the SYSTEM matrix data structure of μ-**Tools**.
- The command `Pi_1phi = sectorNL(0,1)` defines the multipliers $\Pi_{1\varphi}$. Here the numbers 0 and 1 in the first and second argument of the command simply defines the lower and upper sector bound. The resulting variable `Pi_1phi` is a certain data structure that contains all necessary information to define the set $\texttt{Pi}_{1\varphi}(\Psi, \Phi_1, M_{\text{struc}}, M_0)$ in Example 3.
- The command `Pi_2phi = Popov(1,'0')` defines the Popov multipliers in $\Pi_2\varphi$ in the format $\texttt{Pi}_{2\varphi}(\Psi, [\,], M_{\text{struc}}, M_0)$ as defined in Example 3.
- The command `Pi_Delta=IQCadd(Pi_1phi,Pi_2phi)` defines the multipliers $\Pi_\Delta = \Pi_{1\varphi} + \Pi_{2\varphi}$ in the format in Definition 3.

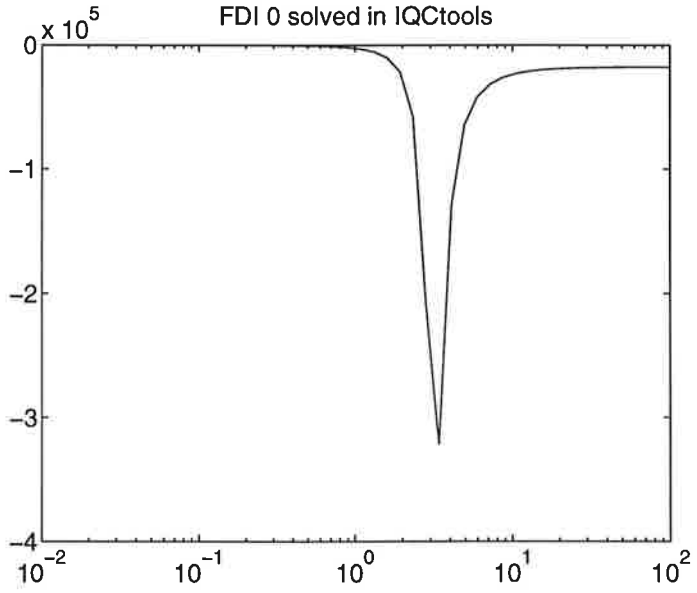

Figure 6 A plot of the inequality in Robustness Test 5 as a function of frequency.

- The command `[Mstruc,tmin] = IQCfeas(G,Pi_Delta,'FDI0')` solves Robustness Test 5 by applying the ideas in Section 3. The output from matlab is the following
  - `ndec=5` shows that there are five variables two solve for in the LMI that corresponds to Robustness Test 5. Two of these are x and λ and the other corresponds to the P_0 matrix in Robustness Test 4.
  - Then follows the the trace of execution from LMI-Lab. We see that we obtain the value `tmin`$=-6.0729 \cdot 10^{-14}$ after four iterations. This means that Robustness Test 5 is feasible, i.e., there exists suitable values for x and λ.
  - We also see that the total M_{struc} matrix becomes

$$M_{\text{struc}} = \begin{bmatrix} 0 & 2.4675 & 0 & 0 \\ 2.4675 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1.5831 \\ 0 & 0 & -1.5831 & 0 \end{bmatrix} 10^5$$

    This means that LMI-Lab found the values $x = 2.4675 \cdot 10^5$ and $\lambda = -1.5831 \cdot 10^5$.

  The third input argument of the command `IQCfeas` is a request for a plot of the largest eigenvalue of the inequality in Robustness Test 5 as a function of frequency. It should be less than zero for all frequencies if the test is feasible. The plot in Figure 6 shows that this is indeed the case in this example.

□

The next example is a continuation of Example 2 in Section 2.

EXAMPLE 5
We assume that the nominal transfer functions is

$$G_0(s) = \frac{100}{s^2 + 2s + 100},$$

and that the weighting filter for the dynamic uncertainty is

$$W(s) = \frac{s + 15}{s + 150}.$$

Furthermore, the spectral characteristic for the input signal w is assumed to be defined by the filter

$$H(s) = \frac{1}{s^2 + 0.1s + 1}.$$

Three steps need to be taken when using the toolbox to obtain an upper bound for (9) in Example 2, see also Figure 7.

1. The first step is to transform the system to the nominal form for robust performance analysis. This amounts to obtaining the transfer function

$$G = \frac{1}{1 + G_0} \begin{bmatrix} -WG_0 & WG_0 \\ 1 & G_0 \end{bmatrix}$$

as a μ-**Tools** SYSTEM matrix. This is done in the first 11 lines of code below.

2. The next step is to use the commands from the first level to obtain finite-dimensional multiplier sets $\mathtt{Pi}_\Delta \subset \Pi_\Delta$ and $\mathtt{Ups}_{\mathrm{inp}} \subset \Upsilon_{\mathrm{inp}}$. For this we use the commands

   - `Pi_Delta=LTIuncert(1,1)` which gives the multipliers

$$\mathtt{Pi}_\Delta = \left\{ \begin{bmatrix} x & 0 \\ 0 & -x \end{bmatrix} : x \geq 0 \right\}$$

   - `Ups_inp=wspectr(Psi0,H)`, where

$$\Psi_0(s) = \begin{bmatrix} \dfrac{10s}{s^2 + 2s + 100} \\ \dfrac{2}{s + 2} \end{bmatrix}.$$

   This gives the multipliers

$$\mathtt{Ups}_{\mathrm{inp}} = \left\{ \Psi_0(j\omega)^* U \Psi_0(j\omega) : \int_{-\infty}^{\infty} \Psi_0^*(j\omega) U \Psi_0(j\omega) |H(j\omega)| d\omega \geq 0 \right\},$$

   where $U = U^T \in \mathbf{R}^{2\times 2}$.

3. Finally the command `[Mstruc,L2P] = IQCperf(G,Pi_Delta,1,Ups_inp)` solves Robustness Test 2 for the case when $W = 1, \Pi_\Delta = \mathtt{Pi}_\Delta$, and $\Upsilon_{\mathrm{inp}} = \mathtt{Ups}_{\mathrm{inp}}$. The optimal value for this finite-dimensional restriction is $\mathtt{L2P} = \sqrt{\gamma} = 0.598$. Lower values of `L2P` may be obtained by using other basis multipliers than $R = 1$ for the command `LTIuncert` and the Ψ_0 defined above for the command `wspectr`. However, it is easy to verify that the optimal value in the case when there is no uncertainty, i.e., when $W = 0$, is $\mathtt{L2P} = 0.5025$. This means that our choice of basis is quite good.

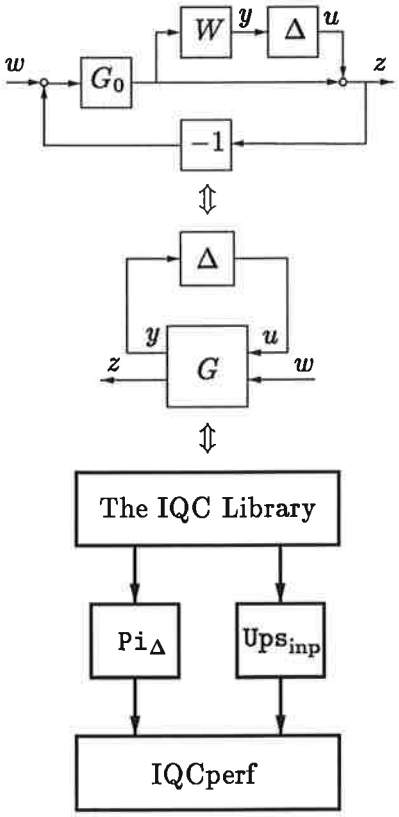

Figure 7 The following steps are taken when using the toolbox for computing an upper bound for (9) in Example 2. The first step is to transform the system to the nominal form for robust performance analysis. The next step is to use the commands from the first level of commands to obtain finite dimensional multiplier descriptions $\texttt{Pi}_\Delta$ and $\texttt{Ups}_{\texttt{inp}}$. We here call the first level of commands the IQC library. The last step is to use the command `IQCperf` with $\texttt{Pi}_\Delta$ and $\texttt{Ups}_{\texttt{inp}}$ as arguments.

```
>> %%% Open loop transfer function (G0):
>> b=[100];
>> a=[1 2 100];
>> [A B C D]=tf2ss(b,a);
>> G0=pck(A,B,C,D);
>>
>> %%% Weighting matrix for dynamic uncertainty
>> W=pck(-150,-135,1,1);
>>
>> %%% Nominal transfer funnction for total system (G):
>> G11n=mscl(mmult(W,G0),-1);
>> G12n=mmult(W,G0);
>> G21n=1;
>> G22n=G0;
>> Gden=minv(madd(1,G0));
>> G=mmult(sbs(abv(G11n,G21n),abv(G12n,G22n)),daug(Gden,Gden));
>>
>> %%% Specification of Pi_Delta
>> R=1;
>> Pi_Delta=LTIuncert(1,1);
```

```
>>
>> b=[1];
>> a=[1 0.1 1];
>> [A B C D]=tf2ss(b,a);
>> H=pck(A,B,C,D);
>>
>> %%%%Specification of Ups_inp
>> b=[0 10 0];
>> a=[1 2 100];
>> [A B C D]=tf2ss(b,a);
>> H1=pck(A,B,C,D);
>> H2=pck(-2,1,2,0);
>> Psi0=abv(H2,H1);
>> Ups_inp=wspectr(Psi0,H);
>>
>>[Mstruc,L2P] = IQCperf(G,Pi_Delta,1,Ups_inp);
ndec =
   140

 Solver for linear objective minimization under LMI constraints

 Iterations    :    Best objective value so far

     1
     2
     3
     4
     5                  108.743553
     6                   59.287019
     :
     :
    35                    0.361103
***                 new lower bound:     0.343581
    36                    0.361103
***                 new lower bound:     0.347273
    37                    0.357546
***                 new lower bound:     0.354149

 Result:  feasible solution of required accuracy
          best objective value:     0.357546
          guaranteed absolute accuracy:  3.40e-03
          f-radius saturation:  86.991% of R =  1.00e+09
>> L2P
L2P =
    0.5980
>>
```

□

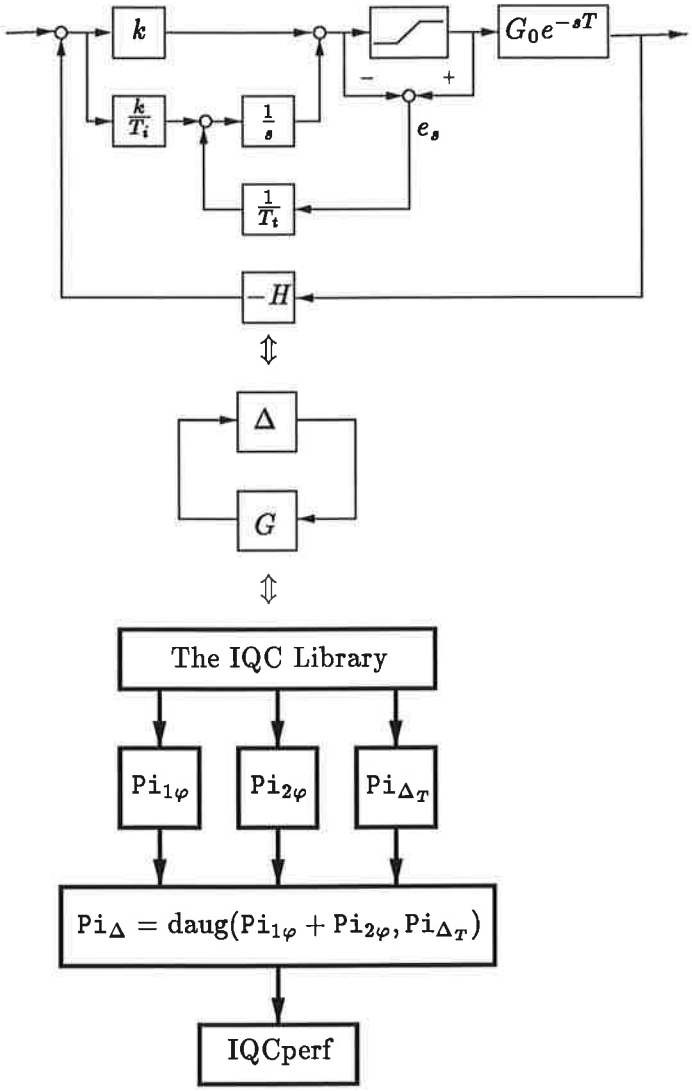

Figure 8 The steps taken in the analysis of the anti-reset windup example.

EXAMPLE 6
Consider the control system in the upper part of Figure 8. A process G_0e^{-sT} is controlled with a PI controller. There is also an anti-reset windup compensator to reduce the effect of integrator windup due to saturation in the actuator. The particular anti-reset windup compensator in this example is of tracking type, see [1] for further details. The idea is to feed the error signal e_s to the integrator when the actuator saturates. With a suitably chosen tracking time constant, T_t, this device prevents the integrator from integrating to a large value during saturation. The filter H corresponds to the sensor dynamics.

We will in this example study stability of the control system in the upper part of Figure 8. In particular we will study how large the time delay T can be before the system becomes unstable. For this purpose we assume that the time delay is uncertain but restricted to be in the interval $[0, T_0]$. We will search for an upper bound on T_0 such that stability is ensured. The following steps are taken when we use the toolbox for the analysis, see also Figure 8.

1. The system is transformed to the nominal form for stability analysis. It

is easy to verify that

$$G(s) = \begin{bmatrix} \dfrac{1 - T_i s G_{PI}(s) H(s) G_0(s)}{sT_t + 1} & -\dfrac{sT_t G_{PI}(s) H(s)}{sT_t + 1} \\ G_0(s) & 0 \end{bmatrix}, \tag{13}$$

$$\Delta = \mathrm{diag}(\varphi, \Delta_T) \tag{14}$$

where $G_{PI}(s) = k + k/(sT_i)$, φ corresponds to the saturation nonlinearity, and $\Delta_T = e^{-sT} - 1$, where $T \in [0, T_0]$. We assume that $k = 0.07$, $T_i = T_t = 0.11$,

$$\varphi(x) = \begin{cases} x, & |x| \le 1, \\ 1, & x > 1, \\ -1, & x < -1, \end{cases}$$

$$G_0(s) = 50 \frac{(s^2 + 0.05s + 1)(s + 200)}{(s^2 + 5s + 50)(s + 10)^2},$$

and

$$H(s) = \frac{50}{s + 50}.$$

2. The next step is to use the first level of commands, which we also call the "IQC Library", to obtain suitable sets of multipliers.

   The saturation nonlinearity, φ, is odd with its slope restricted to the interval $[0, 1]$. This means that it satisfies the IQCs defined by the multipliers from [12], i.e.,

$$\Pi_{1\varphi} = \left\{ \begin{bmatrix} 0 & h_0 + H(j\omega)^* \\ h_0 + H(j\omega) & -2(h_0 + \mathrm{Re} H(j\omega)) \end{bmatrix} : h_0 > 0, \|h\|_1 < h_0 \right\}, \tag{15}$$

   where h is the inverse Fourier transform of H and where the $\mathbf{L}_1$-norm is defined as

$$\|h\|_1 = \int_{-\infty}^{\infty} |h(t)|\, dt.$$

   The first row of the system matrix G in (13) is strictly proper. This means that we also can use a Popov multiplier for the description of φ, i.e.,

$$\Pi_{2\varphi} = \left\{ \begin{bmatrix} 0 & -j\omega\lambda \\ j\omega\lambda & 0 \end{bmatrix} : \lambda \in \mathbf{R} \right\}.$$

   The operator $\Delta_T = e^{-sT} - 1$, where $T \in [0, T_0]$, is described by the multipliers

$$\Pi_{\Delta_T} = \left\{ x(j\omega) \begin{bmatrix} \Psi_0(j\omega T_0) & 0 \\ 0 & -1 \end{bmatrix} : x(j\omega) \ge 0 \right\}, \tag{16}$$

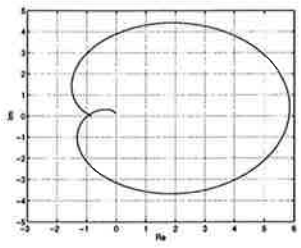

Figure 9 The diagram shows the Nyquist curve for G_{cl}. The stability condition in (17) can be satisfied with proper choice of the Popov parameter λ and the multiplier H.

where

$$\Psi_0(j\omega) = \frac{\omega^2 + 0.08\omega^2}{1 + 0.13\omega^2 + 0.02\omega^4},$$

see [9].

The sets $\Pi_{1\varphi}$ and Π_{Δ_T} are infinite-dimensional and must be restricted. We use the following commands in the toolbox:

- `Pi_1phi=slopeNL([0 -10],0,1)` gives the subset, $\mathtt{Pi}_{1\varphi}$, of (15) that has $H(s) = x/(s-10)$, where $x \in \mathbf{R}$.
- `Pi_2phi=Popov(1,'0')` gives the Popov multipliers in $\Pi_{2\varphi}$.
- `Pi_Delta_T=delay(T_0,R)` gives the subset, $\mathtt{Pi}_{\Delta_T}$, of (16) that has $x(j\omega) = R(j\omega)^* U R(j\omega)$, where $U = U^T > 0$.

3. A multiplier description $\mathtt{Pi}_\Delta = \mathrm{daug}(\mathtt{Pi}_{1\varphi} + \mathtt{Pi}_{2\varphi}, \mathtt{Pi}_{\Delta_T})$ of the diagonal operator in (14) is obtained with the command `Pi_Delta=IQCdaug(IQCadd(Pi_1phi,Pi_2phi),Pi_Delta_T)`.
4. Finally, the command `[Mstruc,tmin] = IQCfeas(G,Pi_Delta)` solves Robustness Test 1 for the case when $\Pi_\Delta = \mathtt{Pi}_\Delta$. The command sequence below shows that `tmin<0`, i.e., the system is stable when $T \in [0, 0.01]$.

The value $T_0 = 0.01$ is in fact close to the optimal bound. To see this consider the transfer function

$$G_{cl}(s) = \frac{T_t s G_{PI}(s) H(s) G_0(s) e^{-sT} - 1}{sT_t + 1}.$$

The control system in the upper part of Figure 8 can be viewed as a negative feedback interconnection of $G_{cl}(s)$ and the saturation nonlinearity φ. It can be shown that the system is stable if

$$\mathrm{Re}[(1 + j\omega\lambda + H(j\omega))(G_{cl}(j\omega) + 1)] > 0, \quad \forall \omega \in [0, \infty]. \tag{17}$$

The Nyquist plot for G_{cl} when $T = 0.011$ is given in Figure 9. The system is very close to instabily for this choice of time delay. In fact, the Nyquist curve is very close to the critical point $(-1, 0)$ for which the system would be unstable even for a unity gain feedback.

```
>> % System transfer function G0(s)
>> num_G0 = 50*conv([1 0.05 1],[1 200]);
>> den_G0 = conv([1 5 50],conv([1 10],[1 10]));
>> % Sensor dynamics
>> num_H = 50;
>> den_H = [1 50];
>> % PI controller
>> k=0.07;
>> Ti=0.11;
>> num_c=[k*Ti k];
>> den_c=[Ti 0];
>> % Tracking time constant
>> Tt = Ti;
>> % System G(s) used in the final loop
>> den_cs = Ti
>> num_sl  = conv(-Tt*num_c,conv(num_G0,num_H));
>> num_G11 = addpoly(conv(den_cs,conv(den_G0,den_H)),num_sl);
>> den_G11 = conv(conv([Tt 1],den_cs),conv(den_G0,den_H));
>> [A_11,B_11,C_11,D_11] = tf2ss(num_G11,den_G11);
>> G_11 = pck(A_11,B_11,C_11,D_11);
>> num_G12 = conv(-Tt*num_c,num_H);
>> den_G12 = conv([Tt 1],conv(den_cs,den_H));
>> [A_12,B_12,C_12,D_12] = tf2ss(num_G12,den_G12);
>> G_12 = pck(A_12,B_12,C_12,D_12);
>> num_G21 = num_G0;
>> den_G21 = den_G0;
>> [A_21,B_21,C_21,D_21] = tf2ss(num_G21,den_G21);
>> G_21 = pck(A_21,B_21,C_21,D_21);
>> G_22 = 0;
>> G = sbs(abv(G_11,G_21),abv(G_12,G_22));
>>
>> % Multipliers for the saturation nonlinearity
>> Pi_1phi = slopeNL([0 -10],0,1);
>> Pi_2phi = Popov(1,'0');
>>
>> % Multipliers for the time-delay
>> R=1;
>> T_0=0.010;
>> Pi_Delta_T = delay(T_0,R);
>>
>> Pi_Delta = IQCdaug(IQCadd(Pi_1phi,Pi_2phi),Pi_Delta_T);
>>
>> [Mstruc tmin] = IQCfeas(G,Pi_Delta)
ndec =
   141

 Solver for LMI feasibility problems L(x) < R(x)
    This solver minimizes  t  subject to  L(x) < R(x) + t*I
    The best value of t should be negative for feasibility

 Iteration   :    Best value of t so far
```

```
     1                        0.270783
***                 new lower bound:    -0.107595
     2                        0.042401
     3                        0.042401
     :
     :
    61                    -2.648646e-07
    62                    -3.539663e-07
    63                    -3.539663e-07
    64                    -5.108738e-07

 Result:  feasible solution
          f-radius saturation:  92.926% of R =  1.00e+09
 Termination due to SLOW PROGRESS:
          t was decreased by less than 10.000% during
          the last 10 iterations.
>> tmin
tmin =
  -5.1087e-07
>>
```

□

5. Discussion

The two hardest parts when using IQCtools for robustness analysis are generally

- Transforming a complex system consisting of several nonlinear, time-varying or uncertain components into the normal form for stability or performance analysis in Figure 1 and Figure 2, respectively. This can be done by using μ-**Tools** commands for manipulating SYSTEM matrices (`abv, sel, madd, mmult`, e.t.c). This can be rather cumbersome.

  An IQC toolbox with a graphical interface that supports the user in this step has been reported by Megretski, [6].

- It is in general hard to know what basis multipliers are good for a certain application. It is often wise to start with as simple multipliers as possible. Non-dynamic multipliers combined with Popov multipliers can often give a good start. The plots of the frequency domain inequalities that appear in the robustness tests gives information on critical frequencies where multipliers are needed. Add multipliers that are active at these frequencies.

6. References

[1] K. Åström and T. Hägglund. *PID Control: Theory Design and Tuning.*

[2] G. Balas, J. Doyle, K. Glover, A. Packard, and R. Smith. *μ-Analysis and Synthesis Toolbox.* The Math Works Inc, 1993.

[3] P. Gahinet, A. Nemirovskii, A. Laub, and M. Chilali. *LMI Control Toolbox*. The Math Works Inc, 1995.

[4] U. Jönsson. *Robustness Analysis of Uncertain and Nonlinear Systems*. PhD thesis, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden, 1996.

[5] U. Jönsson and A. Rantzer. "A format for multiplier optimization." Technical Report TFRT–7530, Department of Automatic Control, Lund Institute of Technology,, Lund, Sweden, March 1995.

[6] A. Megretski. Personal communication.

[7] A. Megretski. "S-procedure in optimal non-stochastic filtering." Technical Report TRITA/MAT-92-0015, Royal Institute of Technology, Stockholm, Sweden, 1992.

[8] A. Megretski. "Power distribution approach in robust control." In *Proceedings of the IFAC Congress*, pp. 399–402, Sydney, Australia, 1993.

[9] A. Megretski and A. Rantzer. "System analysis via integral quadratic constraints: Part I." Technical Report TFRT–7531, Department of Automatic Control, Lund Institute of Technology, 1995. Accepted for publication in IEEE Transactions on Automatic Control.

[10] F. Paganini. "Analysis of systems with combined time invariant/time varying structured uncertainty." In *Proceedings of the American Control Conference*, pp. 3878–3883, Seattle, Washington, 1995.

[11] J. Willems. *The Analysis of Feedback Systems*. MIT Press, Cambridge, Massachusetts, 1971.

[12] G. Zames and P. Falb. "Stability conditions for systems with monotone and slope-restricted nonlinearities." *SIAM Journal of Control*, **6:1**, pp. 89–108, 1968.

7. Command Reference

Multipliers for operators	
complexpar	uncertain complex parameter
LTIuncert	LTI dynamic uncertainty
delay	uncertain delay operator
harmonic	multiplication with harmonic oscillation
Popov	Generalized Popov multipliers
realpar	uncertain real parameter
sectorNL	sector bounded nonlinearity
slopeNL	slope restricted nonlinearity
slowtvpar	slowly time-varying parameter
tvpar	arbitrarily time-varying real parameter

Multipliers for signals	
domharmonic	signals with dominant harmonics
wspectr	signal with given spectrum

Multiplier combination	
IQCadd	addition of multipliers
IQCdaug	diagonal augmentation of multipliers

LMI optimization	
IQCfeas	feasibility test in terms of IQCs
IQCopt	optimization problem in terms of IQCs
IQCperf	performance analysis in terms of IQCs

complexpar

Purpose

Multipliers corresponding to a repeated complex parameter.

Synopsis

```
Pi_Delta = complexpar(R)
```

Description

Specifies a set of multipliers corresponding to a bounded complex parameter i.e., $\Delta = \delta I$, where δ can be:

1. scalar LTI-system with $\|\delta(jw)\| \leq 1$
2. constant parameter with $|\delta| \leq 1$

The size of the identity matrix corresponds to the column size of R.

The multipliers are defined as

$$\Pi(j\omega) = \begin{bmatrix} X(j\omega) & 0 \\ 0 & -X(j\omega) \end{bmatrix},$$

where $X(j\omega) = R(j\omega)^* U R(j\omega)$, and $U = U^T > 0$.

If δ is real valued then these multipliers should be combined with multipliers obtained with the command **realpar**.

See also

realpar

delay

Purpose

Multipliers corresponding to an uncertain delay operator

Synopsis

```
Pi_Delta = delay(T0,R,S)
```

Description

Specifies a set of multipliers corresponding to an uncertain delay, i.e. $\Delta(s) = e^{-sT} - I$, where $0 \leq T \leq T_0$. We have $\Delta(u)(t) = u(t-T) - u(t)$ for some $T \in [0, T_0]$.

The multipliers are defined as

$$\Pi(j\omega) = x(j\omega) \begin{bmatrix} \Psi_0(j\omega T_0) & 0 \\ 0 & -1 \end{bmatrix} + y(j\omega) \begin{bmatrix} 0 & 1 \\ 1 & -1 \end{bmatrix},$$

where

- $x(j\omega) = R(j\omega)^* U R(j\omega)$, with $U = U^T > 0$
- $y(j\omega) = S(j\omega)^* V S(j\omega)$
- $\Psi_0(j\omega) = H(j\omega)^* H(j\omega)$, with $H(s) = 2s(s + \sqrt{12.5})/(s^2 + as + b)$, $b = \sqrt{50}$, and $a = \sqrt{2b + 6.5}$
- if $R = []$ then only the second term of Π
- if $S = []$ then only the first term of Π

domharmonic

Purpose

Multipliers characterizing a signal with dominant harmonics

Synopsis

```
Ups_inp = domharmonic(a,b,n,H)
```

Description

Gives multiplier description for signals satisfying either of the following conditions:

1. supp $\widehat{w}(j\omega) = [-b,-a] \cup [a,b]$
2. supp $\widehat{w}(j\omega) = [-a,a]$ (if $b = []$, $a > 0$)
3. supp $\widehat{w}(j\omega) = (-\infty,-|a|]) \cup ([|a|,\infty)$ (if $b = []$, $a < 0$)

We use the multipliers

$$\Upsilon(j\omega) = [x|H(j\omega)|^2 - d]I_n,$$

where n is the dimension of the signal.

H is a filter with monotone slopes. For case 1 H should be band-pass and for case 2 and 3 H should be low-pass. The following is conditions on H and default choices of filter for the three cases above

1. $x|H(ja)|^2 - d > 0$, $x|H(jb)|^2 - d > 0$, $x, d > 0$, and the default filter is $H(s) = s/(10s/a + 1)(s/(10b) + 1)$
2. $x|H(ja)|^2 - d > 0$, $x, d > 0$, and the default filter is $H(s) = 1/(s/10a+1)$,
3. $x|H(ja)|^2 - d > 0$, $x, d < 0$, and the default filter is $H(s) = 1/(10s/a+1)$.

See also

`wspectr`

harmonic

Purpose

Multipliers corresponding to multiplication with an harmonic oscillation

Synopsis

`Pi_Delta = harmonic(R,`ω_0`)`

Description

Specifies multipliers corresponding to the operator defined by multiplication with an harmonic oscillation, i.e., $(\Delta u)(t) = \cos(\omega_0 t)u(t)$.

The multipliers are defined as

$$\Pi(j\omega) = \begin{bmatrix} \frac{1}{2}(X(j\omega - j\omega_0)) + X(j\omega + j\omega_0)) & 0 \\ 0 & -X(j\omega) \end{bmatrix},$$

where $X(j\omega) = R^*(j\omega)U\,R(j\omega)$, and $U = U^T > 0$.

These multipliers should be combined with multipliers that take the realness of $\cos(\omega_0 t)$ into account. For example, the multipliers formed by the command `tvpar` could be used.

See also

`tvpar`

IQCadd

Purpose

Addition of multipliers for integral integral quadratic constraints

Synopsis

```
Pi_Delta = IQCadd(Pi_1Delta,...,Pi_NDelta)
```

Description

From given sets of multipliers for integral quadratic constraints, this function generates the set of all sums of such multipliers. Currently limited to 9 input arguments.

See also

`IQCdaug`

IQCdaug

Purpose

Diagonal augmentation of multipliers for integral integral quadratic constraints

Synopsis

```
Pi_Delta = IQCdaug(Pi_Delta_1,...,Pi_Delta_N)
```

Description

From given sets of multipliers for the operators $\Delta_1, \ldots, \Delta_N$, this function generates the corresponding set of multipliers satisfied by the operator

$$\mathrm{diag}(\Delta_1, \ldots, \Delta_N).$$

Currently limited to 9 input arguments.

IQCfeas

Purpose

Feasibility test in terms of integral quadratic constraints

Synopsis

```
[Mstruc,tmin] = IQCfeas(G,Pi_Delta,plot_type,wvec)
```

Description

Feasibility test of the frequency domain inequality (FDI0)

$$\begin{bmatrix} G(j\omega) \\ I \end{bmatrix}^* \Pi(j\omega) \begin{bmatrix} G(j\omega) \\ I \end{bmatrix} < 0, \quad \forall \omega \in [0, \infty],$$

where $\Pi(j\omega)$ is defined by `Pi_Delta`, i.e.,

$$\Pi(j\omega) = \Psi(j\omega)^*(M_{\text{struc}} + M_0)\Psi(j\omega),$$
$$\text{subj to: } \Phi_k(j\omega)^*(M_{\text{struc}} + M_0)\Phi_k(j\omega) < 0, \ \forall \omega \in [0, \infty], \quad (\text{FDI1} \ldots \text{FDIN})$$

If the FDI is feasible then `tmin<0` else `tmin>0`.
Inputs:

`G`	System matrix obtained with the mu-tools command **pck**
`Pi_Delta`	A set of multipliers for IQCs
`plot_type`	Either of the following alternatives
	'none' (default) no plot
	'FDI0' plot largest eigenvalue corresponding to FDI0 above.
	'all FDIs' plot largest eigenvalue corresponding to FDI0,...,FDIN in separate windows.
`wvec`	The FDI plots are calculated at these frequencies. By default we use `wvec=logspace(-2,2)`.

Outputs:

`Mstruc`	The "optimal" value for the parameter matrix
`tmin`	The robustness test is feasible if `tmin<0`

See also

`IQCopt` and `IQCperf`

IQCperf

Purpose

Robust performance analysis in terms of IQCs

Synopsis

```
[Mstruc,L2P] = IQCperf(G,Pi_Delta,W,Ups_inp,plot_type,wvec)
```

Description

Computes the robust weighted $\mathbf{L}_2$ performance of a system when the input is constrained to be in the set described by `Ups_inp`. The optimal performance criterion is given as `L2p`$= \sqrt{\gamma_{\text{opt}}}$, where γ_{opt} is the solution to the following optimization problem

$$\inf \gamma \quad \text{subject to} \tag{18}$$

$$\begin{cases} \exists \Pi \in \texttt{Pi\_Delta}, \Upsilon \in \texttt{Ups\_inp}, \quad \text{such that} \\ \begin{bmatrix} G(j\omega) \\ I \end{bmatrix}^* \operatorname{daug}(\Pi + \begin{bmatrix} W^*W & 0 \\ 0 & \gamma I + \Upsilon \end{bmatrix})(j\omega) \begin{bmatrix} G(j\omega) \\ I \end{bmatrix} < 0, \quad \forall \omega \in [0, \infty], \end{cases}$$

where $\gamma > 0$, and W is the weighting matrices for the output. The sets Pi_Delta and Ups_inp are defined by the relations

$$\begin{aligned} &\Pi(j\omega) = \Phi_0(j\omega)^*(M_{\text{struc}} + M_0)\Phi_0(j\omega), \\ &\text{subj to: } \Phi_k(j\omega)^*(M_{\text{struc}} + M_0)\Phi_k(j\omega) < 0, \ \forall \omega \in [0, \infty], \quad k = 1, \ldots, N_\Delta \end{aligned} \tag{19}$$

$$\begin{aligned} &\Upsilon(j\omega) = \Psi_{\text{inp}}(j\omega)^*(M_{\text{struc}_{\text{inp}}} + M_{0_{\text{inp}}})\Psi_{\text{inp}}(j\omega), \\ &\text{subj to: } \Phi_{k_{\text{inp}}}(j\omega)^*(M_{\text{struc}_{\text{inp}}} + M_{0_{\text{inp}}})\Phi_{k_{\text{inp}}}(j\omega) < 0, \forall \omega \in [0, \infty], k = 1, \ldots, N_{\text{inp}} \end{aligned} \tag{20}$$

Inputs:

`G`	System matrix obtained with the mu-tools command `pck`
`Pi_Delta`	A set of multipliers for IQCs. `Pi_Delta` may be empty ([]).
`W`	weighting matrix.
`Ups_inp`	Input specification in terms of IQCs. May be zero matrix of appropriate size.
`plot_type`	Either of the following alternatives
	'none' (default) no plot
	'FDI0' plot largest eigenvalue corresponding to the FDI in (18) above.
	'all FDIs' plot largest eigenvalue corresponding to the FDIs in (18), (19) and (20) in separate windows.
`wvec`	The FDI plots are calculated at these frequencies. By default we use `wvec=logspace(-2,2)`.

See also

`IQCfeas` and `IQCopt`

IQCopt

Purpose

optimization problem in terms of IQCs

Synopsis

```
[Mstruc,gammaotp] = IQCopt(G,Pi_Delta_1,Pi_Delta_2,n2,plot_type,wvec)
```

Description

Solves the optimization problem

$$\inf \gamma \quad \text{subject to} \tag{21}$$
$$\begin{cases} \exists \Pi \in \texttt{Pi\_Delta}(\gamma) \quad \text{such that} \\ \begin{bmatrix} G(j\omega) \\ I \end{bmatrix}^* \Pi(j\omega) \begin{bmatrix} G(j\omega) \\ I \end{bmatrix} < 0, \quad \forall \omega \in [0, \infty] \end{cases}$$

where `Pi_Delta(`γ`)=IQCdaug(Pi_Delta_1,Pi_Delta_2)`. We assume that every $\Pi \in$ `Pi_Delta(`γ`)` has the structure

$$\Pi(j\omega) = \Psi(j\omega)^* \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & \gamma M_{22} \end{bmatrix} \Psi(j\omega),$$

where M_{22} is the lower right n2×n2 block of M (the total structure matrix). By default we use n2=size(M2)/2, where M2 is the structure matrix of `Pi_Delta_2`.
`Pi_Delta_1` may be empty ([]).
It is assumed that the conditions of Proposition 3.2 in [4] are satisfied.
Inputs:

`G` System matrix obtained with the mu-tools command `pck`

`Pi_Delta_1` A set of multipliers for IQCs. `Pi_Delta_1` may be empty ([]).

`Pi_Delta_2` A set of multipliers for IQCs

`n2` See explanation above.

`plot_type` Either of the following alternatives

- 'none' (default) no plot
- 'FDI0' plot largest eigenvalue corresponding to the FDI in (21) above.
- 'all FDIs' plot largest eigenvalue corresponding to the FDI in (21), and the FDIs corresponding to constraints for the multipliers in `Pi_Delta_1` and `Pi_Delta_2`, in separate windows.

`wvec` The FDI plots are calculated at these frequencies. By default we use `wvec=logspace(-2,2)`.

See also

`IQCfeas` and `IQCperf`

LTIuncert

Purpose

Multipliers corresponding to an LTI dynamic uncertainty

Synopsis

```
Pi_Delta = LTIuncert(R,n)
```

Description

Specifies a set of multipliers corresponding to uncertain LTI dynamics $\Delta(j\omega)$ with $\|\Delta(j\omega)\|_\infty \leq 1$. The integer n corresponds to the number of inputs and outputs of Δ.

The multipliers are defined as

$$\Pi(j\omega) = \begin{bmatrix} x(j\omega)I_n & 0 \\ 0 & x(j\omega)I_{n,} \end{bmatrix},$$

where $x(j\omega) = R(j\omega)^* U R(j\omega)$, and $U = U^T > 0$.

See also

`complexpar, realpar`

Popov

Purpose

Defines generalized Popov multipliers

Synopsis

```
Pi_Delta = Popov(n,sign)
```

Description

Specifies a set of generalized Popov multipliers on the form

$$\Pi(j\omega) = \begin{bmatrix} 0 & -j\omega\Lambda \\ j\omega\Lambda & 0 \end{bmatrix},$$

where Λ is a symmetric $n \times n$ matrix specified as follows

`sign='+'` $\Lambda > 0$

`sign='0'` Λ need not be definite

`sign='-'` $\Lambda < 0$

See also

`sectorNL, slopeNL, realpar, slowtvpar`

realpar

Purpose

Multipliers corresponding to a repeated real parameter

Synopsis

```
Pi_Delta = realpar(S)
```

Description

Specifies multipliers corresponding to a repeated real parameter $\Delta = \delta I$, where the size of the identity matrix corresponds to the column size of the basis multiplier S.

The multipliers are defined as

$$\Pi(j\omega) = \begin{bmatrix} 0 & Y(j\omega) \\ Y(j\omega)^* & 0 \end{bmatrix},$$

where $Y(j\omega) = VS(j\omega) - S(j\omega)^* V^T$. This means that $Y(j\omega)$ is skew-Hermitian.

These multipliers should be combined with the multipliers corresponding to the command `complexpar`.

See also

`complexpar`

sectorNL

Purpose

Multipliers corresponding to a sector bounded nonlinearity

Synopsis

`Pi_Delta = sectorNL(`α`,`β`)`

Description

Specifies a set of multipliers corresponding to a nonlinear and possible time-varying gain, which satisfies the condition

$$\alpha x^2(t) \leq \varphi(x(t),t)x(t) \leq \beta x^2(t), \quad \forall t.$$

It is assumed that $-\infty < \alpha < \beta \leq \infty$.

The multipliers are defined as

$$\Pi(j\omega) = T^T \begin{bmatrix} 0 & x \\ x & 0 \end{bmatrix} T, \quad \text{where} \quad T = \begin{bmatrix} \dfrac{\beta}{\beta-\alpha} & -\dfrac{1}{\beta-\alpha} \\ -\alpha & 1 \end{bmatrix},$$

and where $x > 0$.

slopeNL

Purpose

Multipliers for an odd slope restricted nonlinearity

Synopsis

`Pi_Delta = slopeNL(mult,`α`,`β`)`

Description

Specifies a set of multipliers corresponding to an odd slope restricted nonlinearity, i.e., $\Delta(x)(t) = \varphi(x(t))$, where φ satisfies

(*i*) φ is odd

(*ii*) φ has slope restricted to the interval $[\alpha, \beta]$, where $-\infty < \alpha < \beta \leq \infty$. In other words

$$\alpha \leq \frac{\varphi(x_1) - \varphi(x_2)}{x_1 - x_2} \leq \beta$$

If the command `slopeNL(mult)` is used, then $\alpha = 0$ and $\beta = \infty$.

(*iii*) there exists $k > 0$ such that $|\varphi(x)| \leq k|x|$, for any $x \in \mathbf{R}$

The multipliers are defined as

$$\Pi(j\omega) = T^T \begin{bmatrix} 0 & h_0 + H(j\omega)^* \\ h_0 + H(j\omega) & 0 \end{bmatrix} T$$

where

- the transformation matrix is

$$T = \begin{bmatrix} \dfrac{\beta}{\beta - \alpha} & -\dfrac{1}{\beta - \alpha} \\ -\alpha & 1 \end{bmatrix}$$

- $h_0 > 0$
- $H(s) = \sum_{k=1}^{N} (x_k^+ - x_k^-) H_k(s)$, where $x_k^+, x_k^- > 0$ and

$$H_k(s) = \frac{n_k!}{(s + \lambda_k)^{n_k+1}}.$$

The parameters are defined by the vector `mult=[`n_1 $\lambda_1, \ldots, n_N$ λ_N`]`.

- the following $\mathbf{L}_1$ norm constraint holds

$$\sum_{k=1}^{N} (x_k^+ + x_k^-) \|H_k\|_1 < h_0, \quad \|H_k\|_1 = \frac{n_k!}{|\lambda_k|^{n_k+1}}.$$

See also

`sectorNL`

slowtvpar

Purpose

Multipliers corresponding to a repeated real parameter.

Synopsis

`Pi_Delta = slowtvpar(`M_1`,`M_2`,`α`,`R`)`

Description

Defines multipliers corresponding to a slowly time-varying parametric uncertainty, i.e., $\Delta(t) = \delta(t)I$, where $\delta(t)$ satisfies either of the following two conditions:

1. If the fourth input argument is omitted then
   a. $\epsilon \leq \delta(t) \leq 1/\epsilon$, for some small $\epsilon > 0$.
   b. $-2\alpha\delta \leq \dot{\delta}(t) \leq 2\alpha\delta(t)$, where $\alpha > 0$.
2. The other alternative is to let $R > 1$ (close to 1). Then
   a. $\delta(t) \in [-1, 1]$
   b. $\dot{\delta}(t) \in [-\alpha, \alpha]$, where $\alpha > 0$

The multipliers are defined as

$$\Pi(j\omega) = T^T \begin{bmatrix} 0 & M(j\omega) \\ M(j\omega)^* & 0 \end{bmatrix} T,$$

where $M = UM_1 + M_2^* V^T$, and

a. M_1, M_2 are analytic in Re$s \geq -\alpha$
b. M_1 and/or M_2 may be defined as empty i.e., $M_1 = []$ and/or $M_2 = []$
c. U, V are suitably sized matrices
d. we have the constraints:
   1. for the case with positive $\delta(t)$:
   $$UM_1(j\omega - \alpha) + M_1(j\omega - \alpha)^* U^T > 0$$
   $$VM_2(j\omega - \alpha) + M_2(j\omega - \alpha)^* V^T > 0$$
   2. for the case with uniform bounds on $\delta(t)$:
   $$UM_1(j\omega - 2\alpha/R) + M_1(j\omega - 2\alpha/R)^* U^T > 0$$
   $$VM_2(j\omega - 2\alpha/R) + M_2(j\omega - 2\alpha/R)^* V^T > 0$$
e. the transformation matrix is defined as
   1. for positive $\delta(t)$ we use $T = I$ (identity matrix of suitable size)
   2. for the uniform bounds we use
   $$T = \begin{bmatrix} RI & -I \\ RI & I \end{bmatrix}$$

See also

`tvpar, harmonic`

tvpar

Purpose

Multipliers for an arbitrarily time-varying real parameter

Synopsis

```
Pi_Delta = tvpar(Kmax,Kmin,m)
```

Description

Specifies a set of multipliers corresponding to the operator defined by multiplication by a time-varying parameter, i.e. $\Delta(t) = \delta(t)I_m$, where $\delta(t)$ satisfies: $\texttt{Kmin} \leq \delta(t) \leq \texttt{Kmax}$, for all t.

The multipliers are defined as

$$\Pi(j\omega) = \begin{bmatrix} I_m & 0 \\ \alpha I_m & \beta I_m \end{bmatrix}^T \begin{bmatrix} X & Y \\ Y^T & -X \end{bmatrix} \begin{bmatrix} I_m & 0 \\ \alpha I_m & \beta I_m \end{bmatrix}$$

where

- $\alpha = -(\texttt{Kmax} + \texttt{Kmin})/(\texttt{Kmax} - \texttt{Kmin})$
- $\beta = 2/(\texttt{Kmax} - \texttt{Kmin})$
- $X = X^T > 0$
- $Y = -Y^T$

wspectr

Purpose

Multipliers for signals with given spectral characteristic

Synopsis

`Ups_inp = wspectr(`Ψ_0`,`H`,r)`

Description

Specifies a set of multipliers corresponding to the IQC description of an $\mathbf{L}_2[0,\infty)$ signal with given spectral characteristic, i.e., $v \in W_H$, where W_H is the set

$$W_H = \{v \in \mathbf{L}_2[0,\infty) : |v(j\omega)| = \frac{\|v\|}{\|H\|_2}|H(j\omega)|\}$$

and where $H \in \mathbf{RH}_\infty$ is a given strictly proper filter.

The multipliers are defined as

$$\Upsilon_{\text{inp}} = \{\Upsilon = \Upsilon^* : \int_{-\infty}^{\infty} \Upsilon(j\omega)|H(j\omega)|^2 d\omega \geq 0\},$$

where $\Upsilon(j\omega) = \Psi_0(j\omega)^* U \Psi_0(j\omega)$, and $U = U^T$.

The inputs are defined as follows

- $\Psi_0 \in \mathbf{RH}_\infty$ is a (Nx1) "basis function"
- $H \in \mathbf{RH}_\infty$ is a (1x1) strictly proper filter ("spectral characteristic")
- "precision" for approximation, by default 0.001.

See also

`domharmonic`