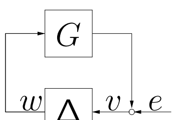
Optimization of IQCs

Ulf Jönsson
Optimization and Systems Theory
Department of Mathematics
Royal Institute of Technology (KTH)
Stockholm, Sweden

Outline

1. Optimization of IQCs
   - Motivation
   - Algorithms
2. The IQC Toolbox
   - Background
   - Environments
   - Tutorial on the IQC part of the toolbox

Optimization of IQCs: Motivation



Theorem 1. *Let* $G \in \mathbf{RH}_\infty$, *and let* Δ *be bounded, causal on* $\mathbf{L}_{2e}[0,\infty)$. *Suppose*

(i) *The system is well-posed*

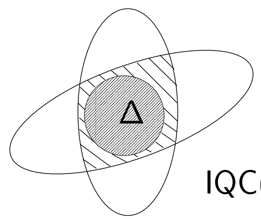
(ii) $\Delta \in \mathrm{IQC}(\Pi)$

(iii) $\Pi_{11} \geq 0$ *and* $\Pi_{22} \leq 0$

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

then the system is stable.

- There are generally many IQCs.
- We use convex parameterizations.
- If $\Delta \in \mathrm{IQC}(\Pi_k)$, $k = 1,\ldots,N$, then also $\Delta \in \mathrm{IQC}(\sum_{k=1}^N \lambda_k \Pi_k)$, where $\lambda_k \geq 0$



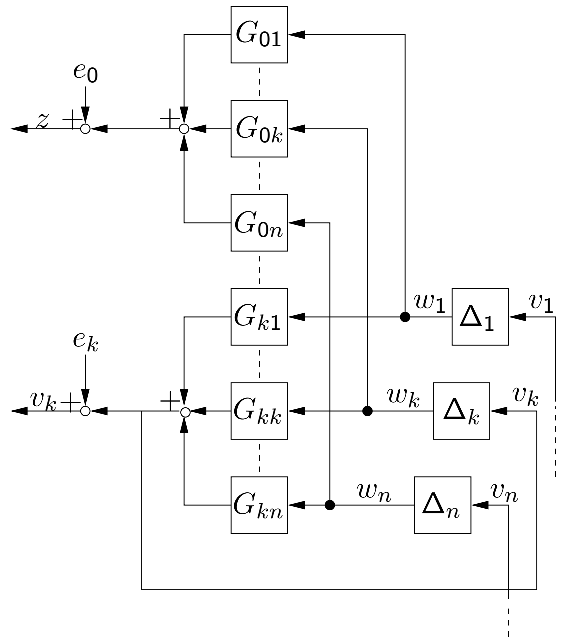
- Resulting stability test

$$\sum_{k=1}^N \lambda_k \begin{bmatrix} G(j\omega) \\ I \end{bmatrix}^* \Pi_k(j\omega) \begin{bmatrix} G(j\omega) \\ I \end{bmatrix} < 0, \qquad \forall \omega \in [0,\infty]$$

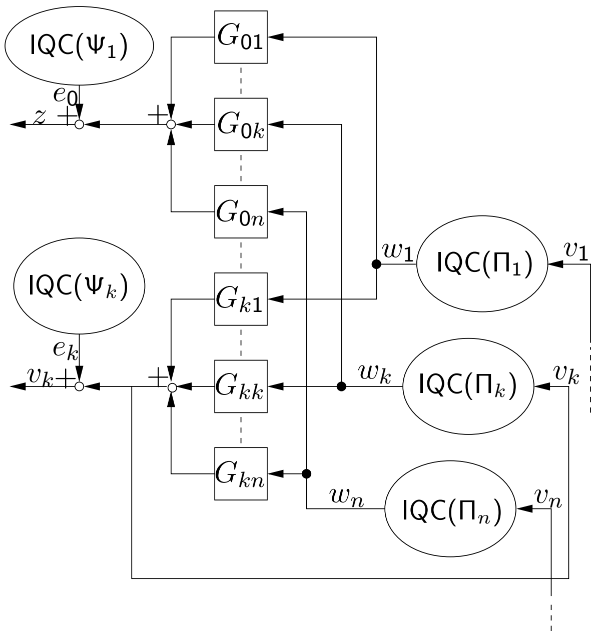
Ideas for semi-automatic systems analysis

- Use IQCs to characterize all uncertain, nonlinear, and time-varying elements.
- Use as many IQCs as possible
- Not necessary to explicitly construct the LFT $\mathcal{S}(G, \Delta)$
- Use efficient algorithms to optimize the parameters of the IQC
- Supply user friendly interface (IQC toolbox)

In analysis of systems with many blocks .....



... we replace all Δ by IQC relations ...



... which leads to the following stability test:

Find $\lambda \in \Lambda$ such that

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

- $G = \begin{bmatrix} G_{01} & \dots & G_{0N} \\ \vdots & & \vdots \\ G_{N1} & \dots & G_{NN} \end{bmatrix}$ stable.
- Π is combination of all IQCs.
- $\Pi(s, \lambda)$ is affine in λ and rational in s, i.e. $\Pi(s,\lambda) = \Pi_0(s) + \sum \lambda_k \Pi_k(s)$
- Λ is convex set of parameters.

Example 1. Consider the IQC with

$$\Pi = \begin{bmatrix} X & Y \\ Y^T & -X \end{bmatrix}, \quad X = X^T \geq 0, \qquad Y = -Y^T$$

If we use the parametrizations (where $N = n + (n-1)^2/2$)

$$X(\lambda) = \begin{bmatrix} \lambda_1 & \lambda_2 & \lambda_3 & \cdots \\ \lambda_2 & \lambda_{n+1} & \lambda_{n+2} & \cdots \\ \lambda_3 & \lambda_{n+2} & \lambda_{2n} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \; Y(\lambda) = \begin{bmatrix} 0 & \lambda_{N+1} & \lambda_{N+2} & \cdots \\ -\lambda_{N+1} & 0 & \lambda_{N+n} & \cdots \\ -\lambda_{N+2} & -\lambda_{N+n} & 0 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

then $\Lambda = \left\{\lambda \in \mathbf{R}^{2N-n} : X(\lambda) \geq 0\right\}$

Example 2.

Consider the IQC with

$$\Pi(j\omega) = \begin{bmatrix} x(j\omega) & 0 \\ 0 & -x(j\omega) \end{bmatrix}, \qquad \text{where } x(j\omega) \geq 0$$

Use the affine parameterization

$$x(j\omega, \lambda) = \lambda_0 + \sum_{k=1}^{N} \left(\frac{\lambda_k}{j\omega + a_k} + \frac{\lambda_k}{-j\omega + a_k} \right)$$

where a_k are given pole locations. Convex parameter set

$$\Lambda = \{\lambda : x(j\omega, \lambda) \geq 0, \ \forall \omega\}$$

IQC Optimization

How to solve the feasibility test:
Find $\lambda \in \Lambda$ such that

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

1. KYP lemma
2. Transformation to Linear Matrix Inequality (LMI)
3. Fast algorithms

KYP Lemma

Lemma 1. *Assume* $j\omega \notin \mathrm{eig}(A)$ *and* (A, B) *stabilizable. Then the following are equivalent*

(i) $$\begin{bmatrix} (j\omega I - A)^{-1}B \\ I \end{bmatrix}^* \begin{bmatrix} Q & S \\ S^T & R \end{bmatrix} \begin{bmatrix} (j\omega I - A)^{-1}B \\ I \end{bmatrix} < 0, \ \forall \omega \in [0, \infty]$$

(ii) *there exists* $P = P^T$ *such that* $$\begin{bmatrix} A^TP + PA + Q & PB + S \\ B^TP + S^T & R \end{bmatrix} < 0$$

(iii) $R < 0$ *and the Hamiltonian* $$\begin{bmatrix} A - BR^{-1}S^T & BR^{-1}B^T \\ Q - SR^{-1}S^T & -A^T + SR^{-1}B^T \end{bmatrix}$$ *has no eigenvalues on the imaginary axis.*

Transformation to LMI

Use state space realization of Π and G to get

$$\begin{bmatrix} G(j\omega) \\ I \end{bmatrix}^* \Pi(j\omega,\lambda) \begin{bmatrix} G(j\omega) \\ I \end{bmatrix} = \begin{bmatrix} (j\omega I - A)^{-1}B \\ I \end{bmatrix}^* \begin{bmatrix} Q(\lambda) & S(\lambda) \\ S(\lambda)^T & R(\lambda) \end{bmatrix} \begin{bmatrix} (j\omega I - A)^{-1}B \\ I \end{bmatrix}$$

where Q, S, R are affine in λ

Equivalent LMI

Feasibility test: Find $\lambda \in \Lambda$ and $P = P^T$ s.t.

$$\begin{bmatrix} A^TP + PA + Q(\lambda) & PB + S(\lambda) \\ B^TP + S(\lambda)^T & R(\lambda) \end{bmatrix} < 0$$

- Standard LMI software works well for mid-size problems
- Computationally demanding if A has high dimension since P becomes large, $\dim(P) = n_A + (n_A - 1)^2/2$

Fast algorithms

Four recent ideas to get around the problem with large size P matrix:

- Cutting plane algorithms (Kao et. al)
  - Use Hamiltonian to obtain hyperplanes
- Outer approximation algorithm (Parrilo)
  - Optimize in frequency domain over a finite frequency grid
  - Use Hamiltonian matrix to update frequency grid
- Special SDP solvers (Hansson, Vandenberghe, Wallin)
  - Exploit structure in the LMI
- Interior path following method with barrier function defined by frequency domain integral (Kao and Megretski)

Cutting Plane Algorithms

- Kelly-type cutting plane algorithm

  (a) C.-Y. Kao. PhD thesis from M.I.T: http://www.math.kth.se/∼cykao/pub.html

  (b) C.-Y. Kao, A. Megretski and U. Jönsson. A Cutting Plane Algorithm for Robustness Analysis of Periodic Systems. *IEEE Transactions on Automatic Control*, 46(4):579–592, April 2001.

- Analytic center cutting plane algorithm

  (a) C.-Y. Kao. PhD thesis

  (b) C.-Y. Kao, A. Megretski and U. Jönsson. Fast Algorithms for IQC Feasibility and Optimization Problems. To appear in *Automatica*

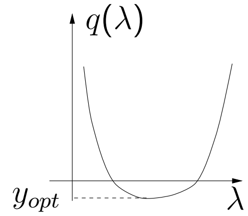
Kelly-type Cutting Plane Algorithm

Let $\Upsilon(j\omega,\lambda) = \begin{bmatrix} G(j\omega) \\ I \end{bmatrix}^* \Pi(j\omega,\lambda) \begin{bmatrix} G(j\omega) \\ I \end{bmatrix}$ and consider

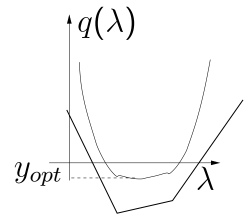
$$\inf y \qquad \text{subject to} \quad \begin{cases} \Upsilon(\lambda) < yI \\ \lambda \in \Lambda \end{cases}$$

If $y_{\text{opt}} < 0$ then we have feasibility of IQC.

Let $q(\lambda) = \inf\{y : \Upsilon(\lambda) < y\}$ "max spectral value function" (convex)



Idea: Use polyhedral approximation of $q(\lambda)$.



Update polyhedral function successively until convergence

repeat

pick test point (y_k, λ_k)

if $\Upsilon(\lambda_k) < y_k I$ *then*

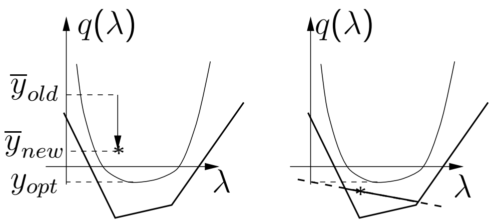
new upper value $\overline{y} := y_k$

else

generate new hyperplane

end

until $y_k < 0$ *or near convergence*



- $\Upsilon(\lambda_k) < y_k$ iff $R_k := R(\lambda_k) - y_k < 0$ and

$$\mathcal{H} = \begin{bmatrix} A - BR_k^{-1}S_k^T & BR_k^{-1}B^T \\ Q_k - S_kR_k^{-1}S_k^T & -A^T + S_kR_k^{-1}B^T \end{bmatrix}$$

has no eigenvalues on the imaginary axis (here $Q_k = Q(\lambda_k)$ e.t.c.).

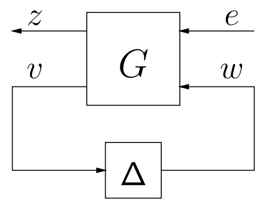
- If $j\omega_k \in \text{eig}(\mathcal{H})$ then $\exists v_k$ s.t.

$$v_k^*(\Upsilon(j\omega_k,\lambda_k) - y_kI)v_k = a^T\lambda_k - y_k + c \geq 0$$

This gives the cutting hyperplane $\{(\lambda, y) : a^T\lambda - y + c = 0\}$.

- Several ways to generate new test point
  - a point between min of polyheder and upper bound
  - analytic center

Analytic Center Cutting Plane Algorithm



Optimal performance problem

$$\inf_{\lambda\in\Lambda} c\lambda \quad \text{subj. to} \quad \Upsilon(\omega,\lambda) < 0,\ \forall\omega\in[0,\infty] \qquad (1)$$

where $\Upsilon(\omega,\lambda)$ is defined on the next slide

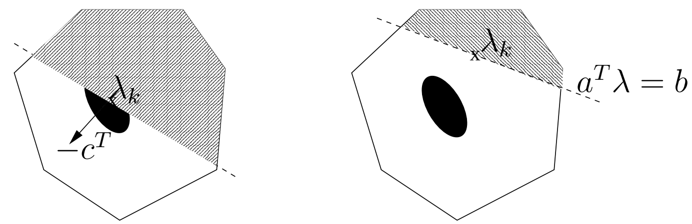
$$\Upsilon(\omega,\lambda) = \begin{bmatrix} G(j\omega) \\ I \end{bmatrix}^* \left[\begin{array}{cc|cc} I & 0 & 0 & 0 \\ 0 & \Pi_{11}(j\omega) & 0 & \Pi_{12}(j\omega) \\ \hline 0 & 0 & -\gamma^2 I + \Psi(j\omega) & 0 \\ 0 & \Pi_{12}^*(j\omega) & 0 & \Pi_{22}(j\omega) \end{array}\right] \begin{bmatrix} G(j\omega) \\ I \end{bmatrix}$$

Note that feasibility problems also can be formulated as (1) since $H(\omega,\lambda) < 0$, $\forall\omega\in[0,\infty]$ iff the following optimization problem have positive optimal objective

$$\inf_{\lambda\in\Lambda, y\in\mathbf{R}} y \quad \text{subj. to} \quad \Upsilon(\omega,\lambda) - yI < 0, \quad \forall\omega\in[0,\infty]$$

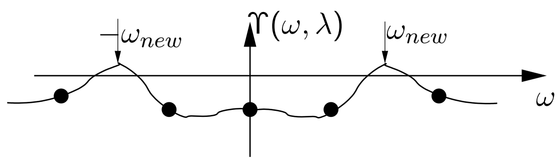
It is indeed enough to find a suboptimal solution $(\bar\lambda,\bar y)$ with $\bar\lambda\in\Lambda$ and $\bar y < 0$.

Main Ideas behind the ACCPA



- Polyhedral outer bound
- Exists oracle such that
  - if the oracle detects that $\Upsilon(\omega,\lambda_k) < 0$ then introduce the cut $c^T(\lambda-\lambda_k)\ge 0$
  - otherwise the oracle generates a cutting hyperplane.

Outer Approximation



1. Solve LMI problem $\min_{\lambda\in\Lambda} \gamma \quad s.t. \quad \Upsilon(\omega_k,\lambda) \le \gamma$ for $\omega_k \in \Omega$.
2. If $\gamma \ge 0$ then stop.
3. Use KYP lemma to check feasibility. If feasible then stop otherwise add new frequencies to the grid Ω.

P. Parillo On the numerical solution of LMIs derived from the KYP lemma IEEE Conference on Decision and Control, 1999.

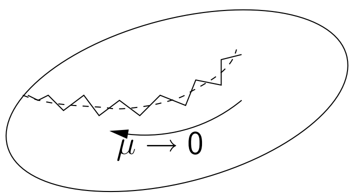
1. *Initialize* $\Omega = \{0\}$
2. $\gamma = \min_{\lambda\in\Lambda} \max_{\omega\in\Omega} \lambda_{\max}(\Upsilon(j\omega,\lambda))$
3. *If* $\gamma \ge 0$ *then unfeasible. Stop!*
4. *If* $j\omega \in \mathrm{eig}(\mathcal{H}(\lambda_{opt}))$ *then* (λ_{opt} *from step 2.)*

   $\Omega := \Omega \cup \{\omega\}$

   else feasible. Stop!
5. *Go to step 2.*

- Can update with worst case frequency in step 4. This requires a little more work. Can prove convergence for this case.
- Need only "dim(Λ)" frequencies in Ω. How to drop frequencies?

Path Following Algorithm



$$\inf c\lambda + \mu B(\lambda)$$

$$B(\lambda) = \log\left(\frac{1}{\pi}\int_{-\infty}^{\infty} tr(\Upsilon(\omega,\lambda)^{-1})\frac{d\omega}{1+\omega^2}\right)$$

C.-Y. Kao and A. Megretski A new barrier function for IQC optimization problems American Control Conference, 2003.

C.-Y. Kao. PhD Thesis.

Structure Preserving Interior Point Algorithms

The LMI from the KYP Lemma has a specific structure

$$\begin{bmatrix} I & 0 \end{bmatrix}^T P \begin{bmatrix} A & B \end{bmatrix} + \begin{bmatrix} A & B \end{bmatrix}^T P \begin{bmatrix} I & 0 \end{bmatrix} + M(\lambda) < 0$$

- Most of the variables are usually in P. Exploit this structure!
- Two different algorithms have been suggested in e.g.

  R. Wallin, A. Hansson, L. Vandenberghe Comparison of two structure-exploiting optimization algorithms for integral quadratic constraints LiTH-ISY-R-2502, Mar 2003.
  - Conjugate gradient based method
  - Reduction of the number of variables

The IQC Toolbox

- History
- Environments
- Tutorial on the IQC part of the toolbox

Background

1. Early versions
   - IQCTools (94-96) by Jönsson and Rantzer, LTH. An interface inspired by the Mu Toolbox.
   - Toolbox (around 94-96) by Megretski and Cygankov. Equipped with a graphical user interface.
2. The IQC toolbox IQCβ (97)
   - Initiated by Megretski. The idea was to exploit new features in Matlab 5 to develop abstract environments for LMI and IQC optimization.
   - The code, installation instructions, and manual can be found at the homepage of Chung-Yao Kao http://www.math.kth.se/ cykao/

Abstract Environments

1. Standard LMI problems

$$\inf c\lambda \text{ subj. to } F(\lambda) := F_0 + \sum_{i=1}^{m} \lambda_i F_i > 0$$

2. Frequency dependent LMI problems

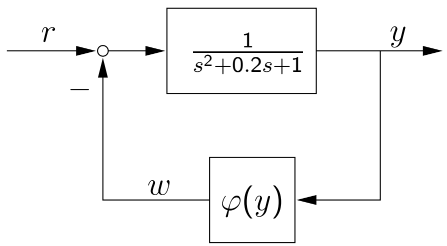
$$\inf c\lambda \text{ subj. to } H(\omega,\lambda) := H_0(\omega) + \sum_{i=1}^{m} \lambda_i H_i(\omega) > 0$$

3. Robustness analysis in the IQC framework

Tutorial

1. Two modes of operation
   - Graphical user interface (GUI)
   - Command line mode
2. The IQC environment
3. Definition of IQC
4. Predefined IQC blocks
5. Optimization Code
6. Example

Example



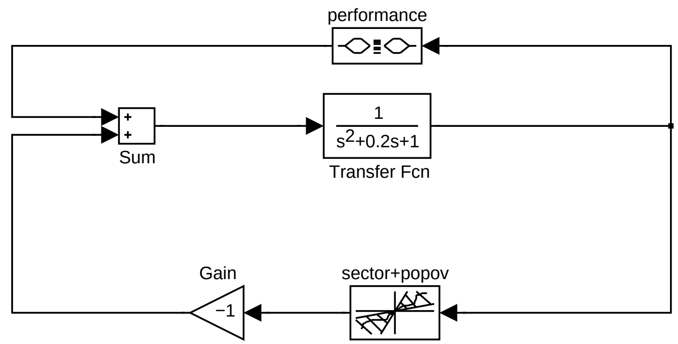
- Compute $\mathbf{L}_2$-gain $r \to y$ (system is stable if finite gain)
- $\varphi \in \text{sector}(0,1)$

$$(y - \varphi(y))\varphi(y) \geq 0$$

- Use Popov IQC

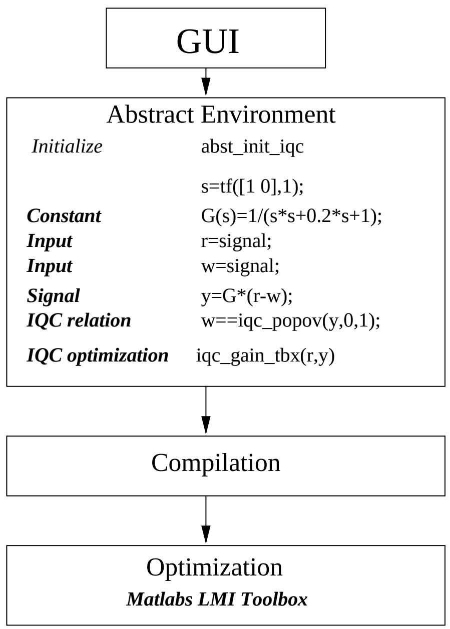
$$\int_0^\infty [x(y - \varphi(y))\varphi(y) + \lambda\varphi(y)\dot{y}]dt \geq 0, \quad x \geq 0, \quad \lambda \in \mathbf{R}$$

Using GUI for our example




```
iqc_lib;
% Draw diagram + define sector constraint
open_system('Ex')
gain=iqc_gui('Ex')
```


Command line mode and structure of toolbox



The IQC Environment and its "abst" Types

Initialization: `abst_init_iqc`

Declaration:

- Constants: "abst type" `constant`
  - `A=[-1 2;2 -3]; G=1/(s+1)`
- Matrix variables: "abst type" `variable`
  - `P=symmetric(2)`
  - `rectangular(n,m)`
  - `diagonal(n)`
  - `skew(n)`
  - `X=variable([1 2;2 3])`
- Inputs (external input): "abst type" `input`
  - `v=signal(2)`

Description :

- Linear matrix inequalities (LMI)
- Frequency dependent LMI
- Integral quadratic constraint (IQC)
- Example of "abst" types to define descriptions
  - `linear`: `A'*P+P*A`
  - `lmi`: `A'*P+P*A<0`
  - `signal`: constant LTI transform `A*v`; `G*v`
  - `varsignal`: variable LTI transform `X*v`
  - `cosignal`: transpose of `signal v'`
  - `varcosig`: transpose of `varsignal v'`
  - `qform`: quadratic form `v'*X*v-w'*X*w`
  - `iqc`: inequality of qforms `v'*X*v-w'*X*w>0`
  - `link`: equivalence of signals `w==v`

Definition of IQCs

Example 1: Compute gain $f \to v$ of

$$v = G(w+f)$$
$$w = \delta v, \quad \delta(t) \in [-1, 1]$$

We can use the IQC

$$\int_0^\infty (v^T X v - w^T X w + v^T Y w)dt \geq 0$$

where $X = X^T \geq 0$ and $Y^T = -Y$.


```
abst_init_iqc;
G=ss(A,B,C,D);
f=signal(n);
w=signal(n);
v=G*(w+f);
X=symmetric(n);
X>0;
Y=skew(n);
v'*X*v-w'*X*w+v'*Y*w>0;
iqc_gain_tbx(f,v)
```


Definition of IQC-Blocks

Example: $w = \Delta v$, $\|\Delta\|_{\mathbf{H}_\infty} \leq 1$
We can use the IQC

$$\int_{-\infty}^\infty (\widehat{v}^* X(j\omega)\widehat{v} - \widehat{w}^* X(j\omega)\widehat{w})d\omega \geq 0$$

where $X(j\omega) = X(j\omega)^* \geq 0$.

We use the following parametrization

$$\chi(s) = X_0 + \sum_k X_k \frac{1}{s+a_k}, \quad X_k = X_k^T \geq 0$$
$$X(s) = \chi(s) + \chi(s)^*$$

The block definition as a Matlab function

```
function w=iqc_uncertainty(v,a)
w=signal(size(v,1));
X=symmetric(n);
for k=1:length(a)
  Xk=symmetric(n);
  X=X+Xk*(1/(s+a(k)));
end
X>0;
v'*X*v-w'*X*w>0;
```


This is used either as

```
w==iqc_uncertainty(v,a);
```

or

```
[wsl,X]==iqc_uncertainty(v,a);
w==wsl;
```


Predefined IQC blocks

Block name	M-file name
dominant harmonics	iqc_domharmonic.m
white noise performance	iqc_white.m

Block name	M-file name
repeated diagonal slope-restricted nonlinearity	iqc_d_slope
odd nonlinearity version of iqc_d_slope	iqc_d_slope_odd
encapsulated deadzone	iqc_dzn_e
encapsulated odd deadzone	iqc_dzn_e_odd
L2 norm-bounded general block	iqc_ltvnorm.m
monotonic with restricted rate	iqc_monotonic.m
rate limiter	iqc_ratelimiter
sector+popov	iqc_popov.m
popov IQC	iqc_popov_vect.m
sector	iqc_sector.m

Block name	M-file name
cdelay	iqc_cdelay.m
uncertain delay	iqc_delay
uncertain delay (simple)	iqc_delay1
harmonic oscillation	iqc_harmonic
unknown constant	iqc_ltigain.m
multi-harmonic oscillation	iqc_multi_harmonic
polytope	iqc_polytope.m
polytope with restrict rate	iqc_polytope_stvp.m
STV scalar	iqc_slowtv.m
TV scalar	iqc_tvscalar.m
window	iqc_window.m

Optimization Code

- The SDP solver in the current version of the Matlab LMI toolbox is too slow for many realistic examples.
- Special purpose codes are being developed
  1. Cutting plane algorithms by C.-Y. Kao
  2. Path following algorithms by C.-Y. Kao
  3. Various structure exploting SDP solvers by Hansson, Vandenberghe and Wallin

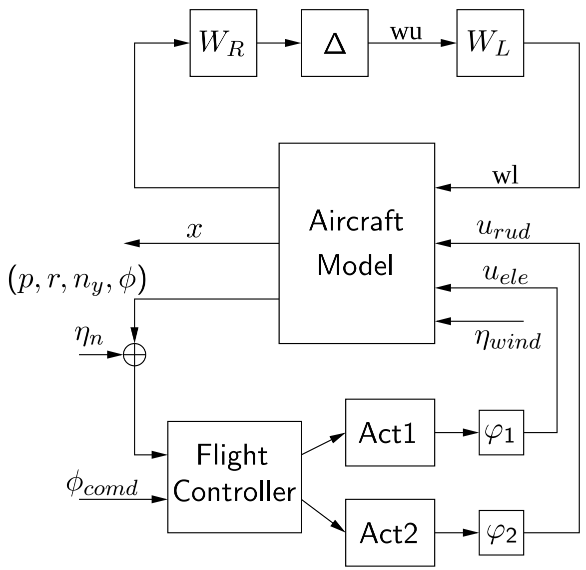
Numerical Example

- Space shuttle lateral axis flight control system
- Adapted from an example in the μ-tools manual.
- The example is taken from

  Kao, Megretski, Jönsson, Rantzer A MATLAB Toolbox for Robustness Analysis

  where it was adapted from

  C.-Y. Kao "Efficient Computational Methods for Robustness Analysis". Department of Mechanical Engineering, MIT. September 2002. http://www.math.kth.se/∼cykao/pub.html




```
abst_init_iqc;

w_unct = signal(9);
w_satu = signal(2);
w_wind = signal;

w_L = WL*w_unct;
output_SS = SC*[w_L;w_satu;w_wind];
v_1 = WR*output_SS(1:3);
v_2 = output_SS(4:7);
output_ctrl = CR*v_2;
output_ac1 = AC1*output_ctrl(1);
output_ac2 = AC1*output_ctrl(2);

for flcnt =1:9
  w_unct(flcnt) == iqc_ltigain(v_1(flcnt));
end
w_satu == iqc_d_slope_odd([output_ac1;...
   output_ac2],[1.2],[0.1];[0.1],[1.2],0,1);
w_wind == iqc_domharmonic(1,1/10,10,2,2);

iqc_gain_tbx(w_wind,v_2);
```

- Choice of basis for the IQC is important for feasibility.
- Advanced IQC were used for the multiple repeated nonlinearity. If

$$\Phi(x) = \begin{bmatrix} \phi(x) \\ \phi(x) \end{bmatrix}$$

  where $\phi \in \text{slope}[0, 1]$ then
  $\sigma_{ZF}(v, \Phi(v) = \langle \Phi(v), (\Upsilon - H)(v - \Phi(v)) \rangle \geq 0$, where
  - $\Upsilon = \Upsilon^T \in \mathbf{R}^{2\times 2}$ has non-positive off diagonal elements
  - $h : \mathbf{R} \to \mathbf{R}^{2\times 2}$ be a symmetric matrix valued function s.t.

$$\Upsilon_{ii} \geq \sum_{j=1, j\neq i}^{n} |\Upsilon_{ij}| + \sum_{j=1}^{n} \|h_{ij}\|_1, \quad \forall i = 1, \ldots, n$$

F. J. D'Amato, et. al. New results for analysis of systems with repeated nonlinearities Automatica 37(5), 2001.

Concluding Remarks

- We do not need to rearrange the system on the G-Δ form before using IQCbeta.
- Performance analysis using IQCs
  - $\mathbf{L}_2$-gain, $\mathbf{L}_2 \to \mathbf{L}_\infty$-gain, robust $\mathbf{H}_2$-performance.
  - IQC characterizations of the input signals.
- Necessity is proven for a few cases using the S-procedure.