# A Toolbox for Robust PID Controller Tuning Using Convex Optimization

**Mehdi Sadeghpour, Vinicius de Oliveira and Alireza Karimi**

*Laboratoire d'Automatique*
*Ecole Polytechnique Fédérale de Lausanne (EPFL)*
*Lausanne, Switzerland*

**Abstract:** A robust PID controller design toolbox for Matlab is presented in this paper. The design is based on linearizing or convexifying the conventional non-convex constraints on the classical robustness margins or $H_\infty$ constraints. Then the existing optimization solvers can be used to compute the controller parameters. The software can be used in a wide range of controller design problems, including multi-model systems and gain-scheduled controllers. The models can be parametric or non-parametric while the software is compatible with the output data of the identification toolbox of Matlab. Three illustrative examples exhibit convenience of working with the developed commands.

*Keywords:* Robust control, PID controller, convex optimization, toolbox

## 1. INTRODUCTION

PID controllers have become the most widely used type of controllers in practice. They are simple, effective controllers able to cover a large area of applications. Although the PID controller has only a three parameters to be tuned, it is surprisingly difficult to find the right tuning for them without systematic procedures and tools. To this end, a quite large variety of techniques and tools has been designed. Most of the techniques are useful in just special applications and cannot be considered a tool. The available tools, however, although having very good attributes, have at least one of these characteristics: 1) Using nonlinear and non-convex optimization problems in design, 2) not supporting multi-model systems, and 3) not considering $H_\infty$ controllers for multi-model, either SISO or MIMO systems.

For instance in [Garpinger and Hagglund (2008)] a software tool for robust PID design is presented. But it uses a non-convex optimization method to design controllers and does not support multi-model cases. In [Oviedo et al. (2006)], a software package is proposed for tuning PID controllers based on constrained nonlinear optimization. The multi-model cases are not considered here either. Gonzalez-Martin et al. (2003) and Harmse et al. (2009) also provide useful PID tuning tools, however, multi-model systems is not supported and the methods are based on non-convex optimizations.

Few software tools have been presented so far that support multi-model cases. Most of them have not got the first and/or third properties mentioned above. An example of these softwares is the one proposed by Bajcinca and Hulin (2004). This Matlab Toolbox designs PID or other three-term controllers based on the method of singular frequencies. But it cannot be used to design multivariable $H_\infty$ controllers. Another example is in Ge et al. (2002) where robust PID controllers are designed via Linear Matrix Inequalities (LMI) approach. This method does not support MIMO systems too.

According to what is said, there is not to the best of authors knowledge a PID controller design tool which designs robust controllers based on $H_\infty$ or classical robustness margins for multi-model (SISO or MIMO) systems. In this paper, a quite comprehensive, yet simple and reliable robust PID controller toolbox for Matlab is presented. This toolbox designs robust controllers in terms of $H_\infty$ performance or classical robustness margins such as the gain and phase margin, for single/multi-model, either SISO or MIMO systems by solving linear or convex optimization problems. The software also supports the design of gain-scheduled PID controllers for LPV (Linear Parameter Varying) systems. The toolbox uses the YALMIP interface Löfberg (2004) for formulation of the optimization problems and the standard optimization solvers.

The paper is organized in five sections. In the next section, a concise theoretical basis for the commands of the software is presented. Section 3 is devoted to the explanation of the commands. In Section 4, three examples are provided, and Section 5 presents some conclusions.

## 2. THEORETICAL BASES

The underlying theories of the developed software are, in fact, the results of Karimi et al. (2007), Karimi and Galdos (2010), and Galdos et al. (2010). We shall briefly describe these results here in order to better understand the commands of the toolbox. Hence, first the class of controllers and models are presented and then different control performances that can be considered by the toolbox are given.

[1] Corresponding author: Alireza Karimi *alireza.karimi@epfl.ch*

### 2.1 Class of models and controllers

The design method needs the frequency response of the plant model in a finite number of frequencies which can be obtained directly from data by spectral analysis or computed from a parametric model. Therefore, high order models with pure time delay and non minimum phase zeros can be considered with no approximation. We consider the set of $m$ models

$$\mathbb{M} = \{G_i(j\omega_k); \quad i = 1, \ldots, m, \quad k = 1, \ldots, N\} \qquad (1)$$

where $G_i(j\omega_k)$ can be a scalar or a matrix representing the frequency response of a SISO or MIMO model, respectively, at $\omega_k$ and N is large enough so that it will give a good approximation of the frequency response of the system. In the sequel a SISO model with a SISO controller is considered. Then, it will be shown how this method can be applied to compute multivariable decoupling controllers.

The class of linearly parameterized controllers $K = \rho^T \phi$, where $\rho$ is the vector of controller parameters and $\phi$ is a vector of basis functions is considered. A continuous-time PID controller belongs to this class with;

$$\rho = [k_p \quad k_i \quad k_d]^T \quad , \quad \phi(s) = [1 \quad \frac{1}{s} \quad \frac{s}{1+\tau s}]^T \qquad (2)$$

where $k_p$, $k_i$, and $k_d$ are, respectively, the coefficients of the proportional, integral, and derivative part of the PID controller and $\tau$ is the known time constant of the derivative part.

For a discrete-time PID controller, we have:

$$\rho = [k_p \quad k_i \quad k_d]^T \quad , \quad \phi(z) = [1 \quad \frac{z}{z-1} \quad \frac{z-1}{z}]^T \qquad (3)$$

It is clear that higher order controllers can also be designed by the proposed approach with choosing for example a set of Laguerre basis functions for $\phi$ (see Karimi and Galdos (2010) for details).

The main interest of linearly parameterized controllers is that every point on the Nyquist diagram of the open loop transfer function is linear with respect to the controller parameters; i.e., $L(j\omega_k) = \rho^T \phi(j\omega_k) G(j\omega_k)$. This property enables us to obtain linear or convex constraints for the optimization problems used in the controller design.

### 2.2 GPhC controller

Gain margin, phase margin and crossover frequency (GPhC) are typical performance specifications for PID controller design in industry. We use these specifications for SISO minimum-phase stable systems if the number of integrators in the open-loop transfer function is less than or equal to 2. Specifying the gain and phase margin defines a straight line in the Nyquist diagram (see $d_1$ in Fig. 1). Now, if the Nyquist curve of the open loop system lies in the right side of $d_1$ the desired values for the gain margin $g_m$ and phase margin $\varphi_m$ will be assured. This can be represented by a set of linear constraints thanks to the linear parameterization of the controller. Now, consider another straight line $d_2$ which is tangent to the middle of the unit circle arc in the sector created by $d_1$ and the imaginary axis. If we call $\omega_x$ the frequency at which the Nyquist curve intersects $d_2$, a crossover frequency greater than or equal to $\omega_x$ can be achieved by a satisfying a set of

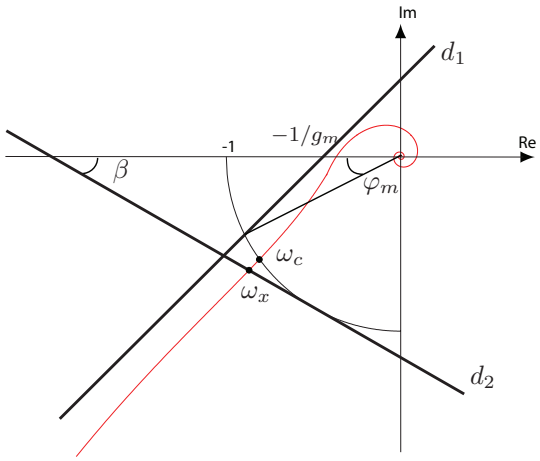


Fig. 1. GPhC specifications converted to linear constraints in Nyquist diagram

linear constraints. In fact, for frequencies greater than $\omega_x$ the Nyquist curve should lie below $d_1$ and above $d_2$ while for frequencies less than $\omega_x$ it should lie below $d_2$.

The control objective is to optimize the load disturbance rejection of the closed-loop. This is, in general, achieved by maximizing the controller gain at low frequencies. For continuous- and discrete-time PID controllers it corresponds to maximizing $k_i$ [Karimi et al. (2007)].

Let us define the set of all points in the complex plane on the line $d$ by $f(x+iy, d) = 0$. Assume that $f(x+iy, d) < 0$ represents the half plane that excludes the critical point. Then, computing a controller that satisfies the desired performance can be carried out by the following linear optimization problem:

$$\max k_i$$

subject to:

$$\begin{aligned} &f(\rho^T \phi(j\omega_k) G_i(j\omega_k), d_1) < 0 \quad \text{for} \quad \omega_k > \omega_x, \\ &f(\rho^T \phi(j\omega_k) G_i(j\omega_k), d_2) > 0 \quad \text{for} \quad \omega_k > \omega_x, \\ &f(\rho^T \phi(j\omega_k) G_i(j\omega_k), d_2) < 0 \quad \text{for} \quad \omega_k \leqslant \omega_x, \\ &\quad \text{for} \quad i = 1, \ldots, m \end{aligned} \qquad (4)$$

In many control problems a constraint on the controller gain at high frequencies can help reducing the large pick values of the control input. This can be achieved by a constraint on the magnitude of the controller gain at frequencies greater than $\omega_h$:

$$|\rho^T \phi(j\omega_k)| < K_u \quad \text{for} \quad \omega_k > \omega_h \qquad (5)$$

This constraint is convex but can be linearized by considering a bound on the real and the imaginary part of $\rho^T \phi(j\omega_k)$ and including it into the above linear programming optimization.

### 2.3 Loop shaping controller :

The performance specification can be defined by a desired open loop transfer function $L_d(j\omega)$. A typical choice for stable systems is $L_d(s) = \omega_c / s$. If a desired reference model $M$ is available, $L_d$ can be computed as $L_d = M/(1-M)$. Then, a PID controller can be designed by minimizing the following quadratic criterion:

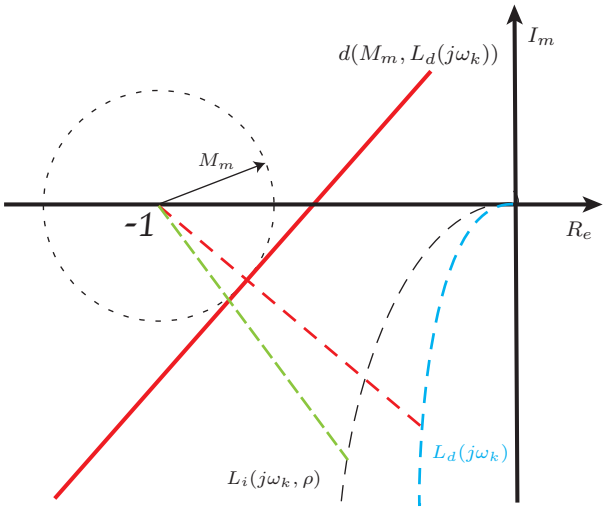


Fig. 2. Loop shaping in Nyquist diagram by quadratic programming

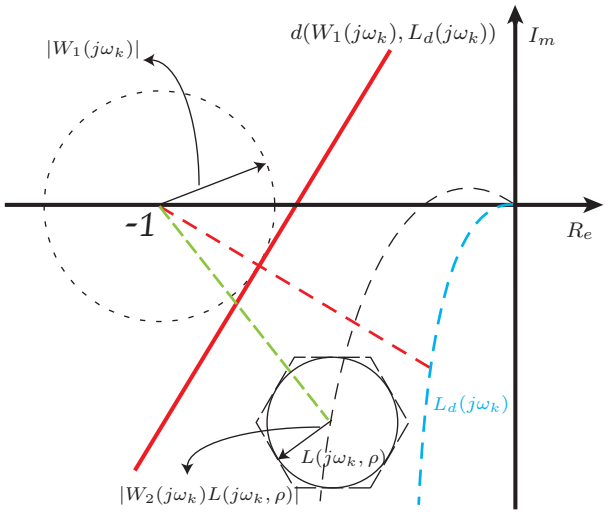


Fig. 3. Expression of the robust performance condition as linear or convex constraints

$$J(\rho) = \sum_{i=1}^{m} \sum_{k=1}^{N} |L_i(j\omega_k, \rho) - L_d(j\omega_k)|^2 \qquad (6)$$

where $L_i(j\omega_k, \rho) = \rho^T \phi(j\omega_k) G_i(\omega_k)$.

The modulus margin, the shortest distance between the Nyquist curve and the critical point, is a better robustness indicator than the classical gain and phase margin [Landau et al. (2011)]. A modulus margin $M_m$ of 0.5 is met if the Nyquist curve does not intersect a circle of radius 0.5 centered at the critical point. This can be achieved if the Nyquist diagram is at the side of $d$, a straight line tangent to the modulus margin circle, that excludes the critical point. This constraint is linear but conservative. The conservatism can be reduced if the slope of this line changes with the frequency. A good choice is a line $d(M_m, L_d(j\omega_k))$ orthogonal to the line that connects the critical point and $L_d(j\omega_k)$ and tangent to the modulus margin circle (see Fig 2). The controller can be designed solving the following quadratic optimization problem :

$$\min J(\rho)$$

$$f(\rho^T \phi(j\omega_k) G_i(j\omega_k), d(M_m, L_d(j\omega_k))) < 0 \qquad (7)$$
$$\text{for} \quad k = 1, \ldots, N \quad , \quad \text{for} \quad i = 1, \ldots, m$$

This approach can be applied to unstable systems if $L_d$ contains the same number of unstable poles as well as the poles of $L_i(s)$ on the imaginary axis (see Karimi and Galdos (2010) for details).

### 2.4 $H_\infty$ controller

Consider a SISO plant model with multiplicative unstructured uncertainty: $\tilde{G}(j\omega) = G(j\omega)[1 + W_2(j\omega)\Delta]$ where $G(j\omega)$ is the plant nominal frequency function, $W_2(j\omega)$ is the uncertainty weighting frequency function, and$\Delta$ is a stable transfer function with $\|\Delta\|_\infty < 1$. In the Nyquist diagram the open loop frequency function will belong to a disk centered at $L(j\omega,\rho)$ with a radius of $|W_2(j\omega)L(j\omega,\rho)|$. This disk can be approximated by a circumscribed polygon with $p > 2$ vertices: $L_i(j\omega,\rho) = K(j\omega,\rho)G_i(j\omega)$ for $i = 1, \ldots, p$, where

$$G_i(j\omega) = G(j\omega)\left[1 + \frac{|W_2(j\omega)|}{\cos(\pi/p)} e^{j2\pi i/p}\right] \qquad (8)$$

Suppose that the nominal performance is defined as $\|W_1 S\|_\infty < 1$, where $S = (1 + KG)^{-1}$ is the sensitivity function and $W_1$ is the performance weighting filter. This condition is satisfied if the Nyquist curve of the nominal model does not intersect the performance disk, a disk centered at the critical point with a radius of $|W_1(j\omega)|$. Therefore, the robust performance is achieved if there is no intersection between the uncertainty and performance disks [Doyle et al. (1992)]. This constraint can be linearized using a straight line $d(W_1(j\omega), L_d(j\omega))$ which is tangent to the performance disk and orthogonal to the line connecting the critical point and $L_d(j\omega)$ [Karimi and Galdos (2010)]. The robust performance is met if $L_i(j\omega,\rho)$ is at the side of $d(W_1(j\omega), L_d(j\omega))$ that excludes the critical point for all $\omega$. This can be represented by the following set of linear constraints:

$$f(\rho^T \phi(j\omega_k) G_i(j\omega_k), d(W_1(j\omega_k), L_d(j\omega_k))) < 0 \qquad (9)$$
$$\text{for} \quad k = 1, \ldots, N \quad , \quad \text{for} \quad i = 1, \ldots, p$$

As a control objective the criterion in (6) can be minimized. An alternative is to define $\|W_1 S\|_\infty < \gamma$ as performance specification and minimizing $\gamma$ with a bisection algorithm.

In the same way, constraints on the weighted infinity norm of other closed loop sensitivity functions can be included in the optimization problem (see more details in Karimi and Galdos (2010)):

$$\|W_3 KS\| < 1 \quad , \quad \|W_4 GS\| < 1$$

where $W_3$ and $W_4$ are weighting filters.

### 2.5 MIMO controller

The performance specifications for SISO systems can also be used for designing MIMO controllers if the open loop system is decoupled. The main idea is to design a MIMO decoupling controller such that the open-loop transfer matrix $\mathbf{L}(j\omega)$ becomes diagonally dominant. For this reason a diagonal desired open loop transfer matrix $\mathbf{L_d}$ is considered and the following quadratic criterion is minimized:

$$J(\rho) = \sum_{k=1}^{N} \|\mathbf{L}(j\omega_k, \rho) - \mathbf{L_d}(j\omega_k)\|_F \qquad (10)$$

where $F$ stands for the Frobenius norm.

MIMO controllers presented by a matrix of transfer functions are considered, where each element $K_{ij}$ of the matrix should be linearly parameterized, i.e., $K_{ij} = \rho_{ij}^T \phi_{ij}$. The controller parameters are obtained by minimizing $J(\rho)$ under some constraints to meet the SISO specifications for each diagonal element.

In MIMO systems, besides the performance constraints, there are other constraints implying the stability of MIMO systems that should be considered. In fact, because the closed-loop system will not be completely diagonal, the stability of dominant loops will not guarantee the stability of the MIMO system. However, a stability condition can be obtained based on Gershgorin bands [Galdos et al. (2010)]:

$$\begin{aligned} &\big|r_q(\omega_k, \rho)[1 + L_{dq}(j\omega_k)]\big| - Re\big([1 + L_{dq}(-j\omega_k)] \\ &\quad \times [1 + L_{qq}(j\omega_k, \rho)]\big) < 0, \\ &\quad \text{for} \quad k = 1, \ldots, N \quad \text{and} \quad q = 1, \ldots, n_o \end{aligned} \qquad (11)$$

where

$$r_q(\omega,\rho) = \sum_{p=1, p\neq q}^{n_o} |L_{pq}(j\omega,\rho)|,$$

$n_o$ is the number of the outputs of the system and $L_{dq}$ is the $q^{th}$ diagonal element of $\mathbf{L_d}$.

### *2.6 Gain-scheduled controller*

All presented robust controller design methods for systems with multimodel uncertainty can be extended to designing gain-scheduled controllers. Suppose that each model in (1) is associated to a value of a scheduling parameter vector $\theta$, which is measured in real time. The controller parameters can be polynomial functions of $\theta$ and be computed by the optimization algorithm. For example, for a PID controller with a scalar scheduling parameter and the vector $\rho$ a second order polynomial of $\theta$ we have [Kunze et al. (2007)]:

$$\rho(\theta) = \begin{bmatrix} k_{p_0} & k_{p_1} & k_{p_2} \\ k_{i_0} & k_{i_1} & k_{i_2} \\ k_{d_0} & k_{d_1} & k_{d_2} \end{bmatrix} \begin{bmatrix} 1 \\ \theta \\ \theta^2 \end{bmatrix}$$

## 3. TOOLBOX COMMANDS

There are three commands that constitute the software structure. The first command determines the controller structure. In the second command the control performance is defined, and finally the last command designs the controller with the predefined structure to meet the performance specifications. In the following comes a description of these three commands.

### *3.1 Controller structure*

The controller structure is defined by the vector of basis function $\phi$. The command is of the form:

```
phi = conphi(ConType,ConPar,CorD,F)
```

`ConType` is a string specifying the type of the controller and can take the following values: `'PI'`, `'PD'`, `'PID'`, `'Laguerre'`, `'General'`, where `'Laguerre'` and `'General'` are used for higher order controller design using Laguerre or generalized orthogonal basis functions and are not discussed in this paper. `ConPar` defines the parameters of the controller structure. For continuous-time PID and PD controllers, it is the time constant of the derivative filter, $\tau$. It can be chosen such that $-1/\tau$ be a high frequency pole (the default value is $\tau = 1.2/\omega_N$).`CorD` is a string taking the values `'s'` or `'z'` showing that `phi` must be a continuous or a discrete-time transfer function, respectively. Finally, `'F'` is a transfer function that can be fixed in the controller. For example, it can be the disturbance model according to the internal model principle.

### *3.2 Control performance*

The control performance is defined by the following command:

```
per = conper(PerType,par,Ld)
```

`PerType` is a string specifying the desired performance of the system. It can be `'GPhC'`, `'LS'` or `'Hinf'`, where LS stands for the loop shaping method.

The `par` for `'GPhC'` is a vector containing gain margin $g_m$, phase margin $\varphi_m$, crossover frequency $\omega_c$, maximum of controller gain $K_u$ and $\omega_h$. No constraint for crossover frequency and maximum gain of controller is used if the last three values are missed. This type of performance can be used only for stable systems. If the desired `Ld` is specified, the criterion (6) will be minimized, else the low frequency gain of the controller will be maximized.

The `par` for `'LS'` is a vector containing $M_m$, $K_u$ and $\omega_h$ and for `'Hinf'` is a structure containing the weighting filters $W_1$, $W_2$, $W_3$ and $W_4$. These filters may be given as continuous- or discrete-time transfer functions or a vector of complex values in discrete frequencies. For `'Hinf'`, `par.gamma` can be defined as a vector containing $\gamma_{\min}$, $\gamma_{\max}$ and $\epsilon$. If this vector is specified the infinity norm of the weighted sensitivity functions will be minimized by a bisection algorithm using the minimum and maximum value of $\gamma$ with a tolerance of $\epsilon$. If this vector is missed, the criterion (6) is minimized.

### *3.3 Controller design*

The third and the last command is the main command in which an optimal controller of the specified structure in `phi` with the closed-loop desired performance characteristics of `per` is obtained. The general form of the command is as follows:

```
K=condes(G,phi,per,options)
```

where `K` is the transfer function of the optimal controller. `G` is the model of the system that can be a `tf`, `ss`, or an `frd` object. In multi-model case, `G` must be defined as a cell array, `G{i}`, which refers to the model $G_i$ for $i = 1, \ldots, m$. `options` is a structure that defines different options for the controller optimization. For example, `options.w` is a vector of frequency points where frequency response of the system is known or is to be evaluated. If it is not specified by the user, a default grid for frequency is created. For gain-scheduled controller design `options.gs=theta`

should be assigned, where `theta` is a vector (or matrix if we have more than one scheduling parameter) of the scheduling parameter values at which the constraints should be met and `options.np=np` is the order of the polynomial for the gain scheduled controller. The number of models in `G` should be equal to the number of grid points in `theta`. The optimization solver can be assigned by `options.solver='linprog'` if linear programming solver should be used.

In MIMO case, `G` is an $n_o \times n_i$ matrix transfer function where $n_i$ is the number of inputs and $n_o$ the number of outputs. The controller will be an $n_i \times n_o$ matrix transfer function. If one wants to define different controller structure for different elements of matrix `K`, one should define `phi` as an $n_o \times n_i$ cell object. Then each element of `phi` can be created using the `conphi` command, for example in a loop. If `phi` is entered just as one vector, then it will be used for all elements of the controller matrix transfer functions. In the same way, `per` is defined for diagonal elements of $L = GK$. So `per{i}{q}` is a structure containing the desired performance characteristics of the $q^{th}$ diagonal element of $L_i = G_i K$ if we have different performance specification for each model $G_i$.

## 4. EXAMPLES

This section presents some examples of the use of the toolbox commands.

*Example 1* Consider a system with three models in three different operating points :

$$G_1(s) = \frac{4e^{-3s}}{10s+1} \quad ; \quad G_2(s) = \frac{e^{-5s}}{s^2+14s+7.5}$$
$$G_3(s) = \frac{2e^{-s}}{20s+1}$$

Compute a PID controller for a gain margin of greater than 3 and a phase margin of at least 60° for all models. Specifying these margins, a controller with maximum performance in terms of load disturbance rejection is designed by entering the series of commands:

```
s=tf('s');
G{1}=exp(-3*s)*4/(10*s+1);
G{2}=exp(-5*s)/(s^2+14*s+7.5);
G{3}=exp(-s)*2/(20*s+1);
phi=conphi('PID');
per=conper('GPhC', [3,60]);
K=condes(G,phi,per)
```

The PID controller is:

$$K(s) = \frac{0.5314s^2 + 0.5319s + 0.05441}{0.0012s^2 + s} \qquad (12)$$

It should be noted that the gain and phase margin of the resulting open-loop system are greater or equal to the specified ones. The Nyquist curves of the three resulting open-loop transfer functions shown in Fig. 4 exhibit satisfaction of the constraints.

*Example 2* Consider the system given in the previous example. The aim is to design a gain-scheduled PID controller with loop shaping method for this system.

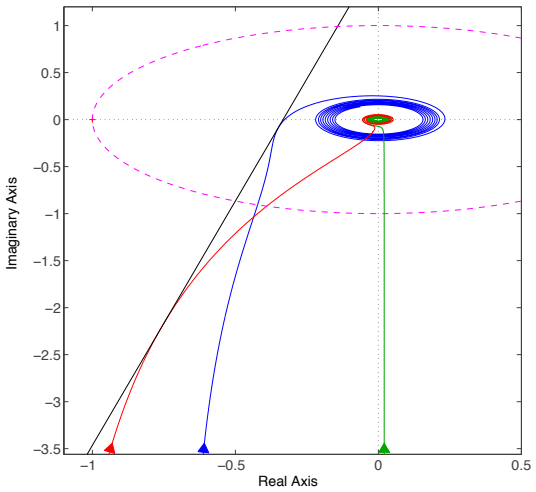


Fig. 4. The Nyquist diagrams of $KG_1$ (blue line), $KG_2$ (green line), and $KG_3$ (red line) in Example 1. The black line is $d_1$.

Suppose that the three models corresponds to a normalized scheduling parameter $\theta \in \{-1, 0, 1\}$. We consider a desired open loop transfer function $L_d(s) = \omega_c/s$ with $\omega_c = 1$ rad/s and a desired modulus margin of $M_m = 0.4$. Assume that the gain-scheduled PID controller is described as follows:

$$K(s) = [k_p \;\; k_i \;\; k_d]\,[1 \;\; \frac{1}{s} \;\; \frac{s}{1+\tau s}]^T$$

where $k_p = k_{p_0} + k_{p_1}\theta$, $k_i = k_{i_0} + k_{i_1}\theta$ and $k_d = k_{d_0} + k_{d_1}\theta$. Using the series of commands:

```
Ld=1/s;Mm=0.4;
phi=conphi('PID');
per=conper('LS',Mm,Ld);
options.gs=[-1;0;1];options.np=1;
K=condes(G,phi,per,options)
```

the coefficients of the gain-scheduled PID controller are :

$$\begin{aligned} k_p &= 1.6781 + 1.1603\,\theta \\ k_i &= 0.3324 + 0.1677\,\theta \\ k_d &= 2.4771 + 1.6792\,\theta \end{aligned} \qquad (13)$$

with $\tau = 0.0012$. In Fig. 5 the Nyquist diagrams of the resulting open-loop transfer functions for three values of $\theta = -1, 0, 1$ are shown. All three curves do not intersect the modulus margin circle and their distance to $L_d$ is minimized.

*Example 3* In this example, a PID controller for the multi-model MIMO process proposed in Bao et al. (1999) is designed. In Bao et al. (1999) a multi-loop PID controller is designed for the process $G_1(s)$ with weighting functions $W_1(s)$ and $W_2(s)$. Afterwards, the designed controller is examined on a slightly different system (denoted by $G_2$ here) to check its robustness. The related transfer functions are:

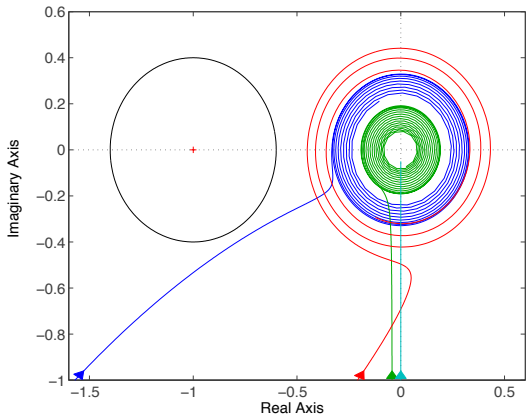


Fig. 5. The Nyquist curves of $K(j\omega,\theta)G(j\omega,\theta)$ for $\theta = -1$ (blue line), $\theta = 0$ (green line), and $\theta = 1$ (red line). The modulus margin of 0.4 is shown with a black circle.

$$G_1(s) = \begin{bmatrix} \frac{-33.89}{(98.02s+1)(0.42s+1)} & \frac{32.63}{(99.6s+1)(0.35s+1)} \\ \frac{-18.85}{(75.43s+1)(0.30s+1)} & \frac{34.84}{(110.5s+1)(0.03s+1)} \end{bmatrix}$$

$$G_2(s) = \begin{bmatrix} \frac{-33.89\,e^{-0.1s}}{(98.01s+1)(0.43s+1)} & \frac{32.63\,e^{-0.1s}}{(98.5s+1)(0.33s+1)} \\ \frac{-18.85\,e^{-0.1s}}{(76.0s+1)(0.31s+1)} & \frac{34.84\,e^{-0.1s}}{(109.5s+1)(0.025s+1)} \end{bmatrix}$$

$$W_1(s) = \frac{s+1000}{1000s+1}I \qquad W_2(s) = \frac{500s+1000}{3s+5000}I$$

Here, the weighting functions $W_1$ and $W_2$ are applied to both systems $G_1$ and $G_2$ where a simple desired open loop function $L_d$ is chosen as shown by the commands below:

```
s=tf('s');
Ld=1/(10*s);
par.W1=(s+1000)/(1000*s+1);
par.W2=(500*s+1000)/(3*s+5000);
G{1}=G1; G{2}=G2;
phi=conphi('PID',0.05,'s');
per=conper('Hinf',par,Ld);
K=condes(G,phi,per)
```

The result is the following controller:

$$K(s) = \begin{bmatrix} \frac{-10.83s^2-17.12s-0.103}{0.05s^2+s} & \frac{-1.301s^2+12.31s+0.1239}{0.05s^2+s} \\ \frac{-2.332s^2-10.54s-0.05149}{0.05s^2+s} & \frac{1.132s^2+17.15s+0.09748}{0.05s^2+s} \end{bmatrix}$$

In Bao et al. (1999) the proposed multi-loop controller could not satisfy the nominal performance condition $\|W_1 S\|_\infty < 1$ while our controller satisfies the robust performance condition for both models.

## 5. CONCLUSIONS

A Matlab toolbox for designing robust PID controllers is presented. This toolbox designs robust PID controllers using linear programming and in some cases convex optimization. When designing controllers based on constraints on classical robustness margins or $H_\infty$ performance, solving linear or convex optimization problems would be a great advantage yielding faster and reliable results. Dealing with multi-model, either SISO or MIMO systems is easy with the presented toolbox. Three main commands are developed for design as shown in examples. A GUI will also be designed for the software to become more convenient to use.

## REFERENCES

Bajcinca, N. and Hulin, T. (2004). Robsin: A new tool for robust design of PID and three-term controllers based on singular frequencies. In *Proceedings of the IEEE International Conference on Control Applications*, 1546–1551. Taipei, Taiwan.

Bao, J., Forbes, J.F., and McLellan, P.J. (1999). Robust multiloop PID controller design: A successive semidefinite programming approach. *Industrial and Engineering Chemistry Research*, 38(9), 3407–3419.

Doyle, C.J., Francis, B.A., and Tannenbaum, A.R. (1992). *Feedback Control Theory*. Mc Millan, New York.

Galdos, G., Karimi, A., and Longchamp, R. (2010). $H_\infty$ controller design for spectral MIMO models by convex optimization. *Journal of Process Control*, 20(10), 1175 – 1182.

Garpinger, O. and Hagglund, T. (2008). A software tool for robust PID design. In *Proceedings of the 17th IFAC World Congress*. Seoul, Korea.

Ge, M., Chiu, M.S., and Wang, Q.G. (2002). Robust PID controller design via LMI approach. *Journal of Process Control*, 12(1), 3 – 13.

Gonzalez-Martin, R., Lopez, I., Morilla, F., and Pastor, R. (2003). Sintolab: the REPSOL-YPF PID tuning tool. *Control Engineering Practice*, 11(12), 14691480.

Harmse, M., Hughes, R., Dittmar, R., Singh, H., and Gill, S. (2009). Robust optimization-based multi-loop PID controller tuning: A new tool and an industrial example. In *7th IFAC International Symposium on Advanced Control of Chemical Processes, ADCHEM'09*, 548–553. Istanbul, Turkey.

Karimi, A. and Galdos, G. (2010). Fixed-order $H_\infty$ controller design for nonparametric models by convex optimization. *Automatica*, 46(8), 1388–1394.

Karimi, A., Kunze, M., and Longchamp, R. (2007). Robust controller design by linear programming with application to a double-axis positioning system. *Control Engineering Practice*, 15(2), 197–208.

Kunze, M., Karimi, A., and Longchamp, R. (2007). Gain-scheduled controller design by linear programming. In *European Control Conference*, 5432–5438.

Landau, I.D., Lozano, R., M'Saad, M., and Karimi, A. (2011). *Adaptive Control: Algorithms, Analysis and Applications*. Springer-Verlag, London.

Löfberg, J. (2004). YALMIP: A toolbox for modeling and optimization in MATLAB. In *CACSD Conference*. URL `http://control.ee.ethz.ch/ joloef/yalmip.php`.

Oviedo, J.J.E., Boelen, T., and Overschee, P.V. (2006). Robust advanced PID control (RaPID): PID tuning based on engineering specifications. *IEEE Control Systems Magazine*, 26(1), 15–19.