Graphical User Interface for Design Stabilizing Controllers

Petr Urban[1,2], *Michael Šebek*[1]

[1]Department of Control Engineering, Faculty of Electrical Engineering,
Czech Technical University, Prague
[2]Institute of Information Theory and Automation,
Czech Academy of Sciences, Prague

Abstract

In this note we show how the problem of finding all stabilizing controllers - PI, PD, PID can be solved using the Hermite stability criterion. We make GUI - Graphical User Interface - for simple design of controllers. This report is application-oriented and therefore it hasn't much theoretical background.

1 Introduction

Robust control theory has certain tools for design of robust controller intended for control of systems with uncertain parameters. Design of static output feedback simultaneous stabilization of scalar plants with uncertain parameters is interesting problem. In these cases when it goes about only one parameter, the results are available now in [1], [3], [4]. For instance, it is possible to compute the stability interval, i.e. an interval of values of the uncertain parameter (gain of the static output feedback) for which the system is stable.

In the case of more then one uncertain parameter, the situation is a lot more difficult from the theoretical point of view. Nonetheless, in a two or three-parameter case it is possible to take advantage of some graphical routines available in common CACSD packages and the possible result is the (2D) or (3D) picture. More n-dimensional picture isn't good from practical aspect, i.e. comprehensive, visual orientation.

In case of control loop analysis can finding parameters represents i.e. uncertain parameters of closed loop characteristic polynomial.

$$\begin{aligned} p(s,q,r) &= a_0(q,r) + a_1(q,r)s + ... \\ &+ a_i(q,r)s^i, \;\; i = 1, ..., N \end{aligned} \tag{1}$$

denote the closed loop characteristic polynomial of degree N, where $a_i(q,r)s^i$ are scalar multivariate polynomials with uncertain parameters q, r. Our objective is to find a region for parameters q, r for which the closed loop characteristic polynomial is stable.

At this point when it goes about design of control loop (controller) can be these finding parameters i.e. individual gain of classical dynamic controllers - PI, PD, PID consequently K_i, K_d, K_p. For which is the control loop stable.

In this report we show how the Hermite polynomial stability criterion in [5] for obtain parameters of PI, PD, PID controllers can be used. Result from these procedure is the set of controllers that are stabilizing the plant. The plant is set with external description - transfer function. The GUI-Graphical User Interface is designed and created using MATLAB see [2] with its basic functions and POLYNOMIAL TOOLBOX see [1].

2 Solution

In solution of our problem we issue from picture Fig. 1 which show general scheme of control loop.

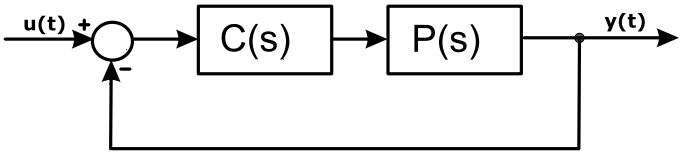

Figure 1: Closed control loop

Closed control loop transfer function is

$$CL(s) = \frac{C(s)P(s)}{1 + C(s)P(s)} \ , \tag{2}$$

where $C(s) = \frac{y(s)}{x(s)}$ is the controller and $P(s) = \frac{p(s)}{q(s)}$ is the plant, $x(s), y(s), p(s), q(s)$ are polynomials in variable s. After substitution and rearrangement equation (2) we obtain transfer function in this form

$$CL(s) = \frac{p(s)y(s)}{p(s)y(s) + q(s)x(s)}$$

denominator of this function is the closed loop characteristic polynomial.

2.1 The Hermite stability criterion

We use the Hermite stability criterion for the stability test and finding uncertain parameters :
A scalar polynomial $r(s) = r_0 + r_1 s + ... + r_n s^n$ is stable if and only if the symmetric $n \times n$ Hermite matrix $H = [h_{ij}]$ defined by

$$\begin{aligned} h_{ij} &= \sum_{k=0}^{i-1} (-1)^{k+i-1} r_k r_{i+j-k-l}, \ \ i \leq j \\ &= 0, \ \ i + j \ \ odd; \ \ i, j = 1, 2, ..., n \end{aligned}$$

is positive definite.
For example, when $p(s, q, r)$ is a polynomial of degree 3 and the stability region is the open left half-plane, matrix H is given by

$$H = \begin{bmatrix} a_2 a_3 & 0 & a_0 a_3 \\ 0 & a_1 a_2 - a_0 a_3 & 0 \\ a_0 a_3 & 0 & a_2 a_3 \end{bmatrix}. \tag{3}$$

where subdeterminants

$$\begin{aligned} D_1 &= a_2(q,r)a_3(q,r) \\ D_2 &= a_2(q,r)a_3(q,r)(a_1(q,r)a_2(q,r) - \\ &- a_0(q,r)a_3(q,r)) \\ D_3 &= a_2(q,r)a_3(q,r)(a_1(q,r)a_2(q,r) - \\ &- a_0(q,r)a_3(q,r))a_2(q,r)a_3(q,r) - \\ &- a_0(q,r)a_3(q,r)(a_1(q,r)a_2(q,r) - \\ &- a_0(q,r)a_3(q,r))a_0(q,r)a_3(q,r) \end{aligned}$$

depend only on coefficients of polynomial $a_i(q,r)$.

2.2 Building closed loop characteristic polynomial

Matlab implementation of building the closed loop polynomial is follow:

- Polynomial for PI controllers
  Controller transfer function is

  $$C(s) = \frac{K_I + K_P s}{s} = \frac{y(s)}{x(s)}$$

  X represents K_P and Y represents K_I

```
a{1} = ['(' '(' p{1} ').* Y' ')'];
while jj <= degree
  if jj == degree
    a{jj+1} = ['(' '(' p{jj} ').*X +(' q{jj} ')' ')'];
  else
    a{jj+1} = ['(' '(' p{jj+1} ').* Y +(' p{jj} ').*X +(' q{jj} ')' ')'];
  end
  jj = jj + 1;
end
```

- Polynomial for PD controllers
  Controller transfer function is

  $$C(s) = K_P + K_D s = \frac{y(s)}{x(s)}$$

  X represents K_P and Y represents K_D

```
a{1} = ['(' '(' q{1} ')+(' p{1} ').*X' ')']; while jj < degree
    a{jj+1} = ['(' '(' q{jj+1} ')+(' p{jj+1} ').*X +(' p{jj} ').*Y' ')'];
    jj = jj + 1;
end
```

- Polynomial for PID controllers
  Controller transfer function is

  $$C(s) = \frac{K_I + K_P s + K_D s^2}{s} = \frac{y(s)}{x(s)}$$

– controller with fixed parametr K_P,
X represents K_I and Y represents K_D

```
a{1} = ['(' '(' p{1} ').*X' ')'];
while jj <= degree
  if jj == degree
    a{jj+1} = ['(' '(' p{jj} ').*' num2str(Kp) '+(' p{jj-1} ').*Y +
             + (' q{jj} ')' ')'];
  elseif jj == 1
    a{jj+1} = ['(' '(' p{jj+1} ').*X +(' p{jj} ').*' num2str(Kp) '+
             + (' q{jj} ')' ')'];
  else
    a{jj+1} = ['(' '(' p{jj+1} ').*X +(' p{jj} ').*' num2str(Kp) '+
             + (' p{jj-1} ').*Y +(' q{jj} ')' ')'];
  end
  jj = jj + 1;
end
```

– controller with fixed parametr K_I,
X represents K_D and Y represents K_P

```
Ki = varargin{4};
a{1} = ['(' '(' p{1} ').*' num2str(Ki) ')'];
while jj <= degree
  if jj == degree
    a{jj+1} = ['(' '(' p{jj} ').*Y +(' p{jj-1} ').*X +(' q{jj} ')' ')'];
  elseif jj == 1
    a{jj+1} = ['(' '(' p{jj+1} ').*' num2str(Ki) '+(' p{jj} ').*Y +
             + (' q{jj} ')' ')'];
  else
    a{jj+1} = ['(' '(' p{jj+1} ').*' num2str(Ki) '+(' p{jj} ').*Y +
             + (' p{jj-1} ').*X +(' q{jj} ')' ')'];
  end
  jj = jj + 1;
end
```

– controller with fixed parametr K_D,
X represents K_I and Y represents K_P

```
Kd = varargin{4};
a{1} = ['(' '(' p{1} ').*X' ')'];
while jj <= degree
  if jj == degree
    a{jj+1} = ['(' '(' p{jj} ').*Y +(' p{jj-1} ').*' num2str(Kd) '+
             + (' q{jj} ')' ')'];
  elseif jj == 1
    a{jj+1} = ['(' '(' p{jj+1} ').*X +(' p{jj} ').*Y +(' q{jj} ')' ')'];
  else
    a{jj+1} = ['(' '(' p{jj+1} ').*X +(' p{jj} ').*Y +
             + (' p{jj-1} ').*' num2str(Kd) '+(' q{jj} ')' ')'];
  end
  jj = jj + 1;
end
```

where degree is max degree$(p(s), q(s))$, $p(s)$ and $q(s)$ are input polynomial. Output of these functions is cell array $a\{:\}$ where $a\{jj\}$ are coefficients of the closed loop characteristic polynomial:

$$cl(s) = a\{1\}s^0 + a\{2\}s^1 + ... + a\{jj+1\}s^{jj}, \text{ where } jj = \text{degree}$$

with uncertain parameters X, Y like controller constants.

2.3 Algorithm

1. In the first step we must compute the Hermite matrix. We compute the determinant D_N and all subdeterminants D_i, where $i = 1, 2, 3, ..., (N-1)$. N is a degree of the Hermite matrix.

2. We obtained all subdeterminants D_i and determinant D_N. Now we are going to find all parameters for which is Hermite matrix positive definite.

3. We make grid for all finding parameters using the standard Matlab function. Then for each value of the grid we evaluate all D_i and D_N. From this we obtain sets (matrixes) in which 1 represents that for this value of polynomial parameter is D_i or D_N stable and 0 represents that for this value of polynomial is D_i or D_N unstable. The sets (matrixes) data type is logical.

4. In this step make the intersection between all sets (matrixes). Hence we obtain the final set (area) that we need graphically visualize.

5. This step consist of drawing the final set (area) - for it we use standard Matlab graphic routines i.e. pcolor, shading. In last procedure of this algorithm we draw the zero curves. Zero curves are boundary where the parameter is crossing from stable to unstable region and v.v.

3 Graphical interface

In this section we show the GUI for design of controllers. The basic figure is show in Fig. 2. Now we describe it

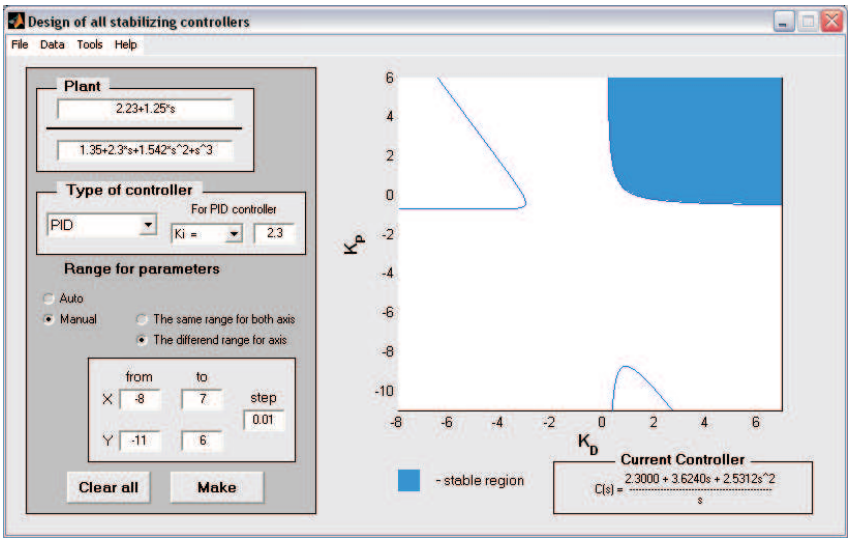

Figure 2: GUI for design of stabilizing controllers

Plant serve for input of the transfer function of the plant for which we will design a controller. Format of input data is polynomial or we can use name of workspace variable (type of variable must be polynomial). Denominator of the transfer function must be monic.

Type of controller with this item we specify the type of designed controller. For PID controller we must specify the fixed parameter too.

Range for parameters has two possibilities

- *Auto* - this is the default value of the range $\langle -10; 10\rangle$ with step 0.05.
- *Manual* - if we will specify other range we have two possibilities:
  - The same range for both axis
  - The different range for axis

Make this is the button for simplify the controller.

Clear All this button serve for erase all input fields.

Current Controller it is only the frame in which is shown the transfer function of the controller.

Figure Menu serve for some operation with:

File is menu item for basic operation with matlab figure (file).

Data in this menu item we can do this

- *Get data from figure*
- *Update data from workspace* - update plant fields if are used workspace variables
- *Export controller data* - export controller to workspace. Before export we must use *Get data from figure* to get controller data.
- *Export region to figure*

Tools here are basic axes function like zoom and edit plot

4 Acknowledgements

This work was supported by the Grant Agency of the Czech Republic grant No. 102/02/0709 and by the Ministry of Education of the Czech Republic under Project ME 546.

5 Conclusion

We try to design simple interactive GUI for design of PI, PD and PID controllers. We notice that all designed controllers are only in theoretical (ideal) form how you can find it in many school books and scripts. In the next version of this GUI we will use other forms of controllers usable in practical cases. Drawing of the final region isn't simple problem and therefore we use standard Matlab functions i.e. meshgrid, pcolor, contour etc. It is time-difficult for small step or large range of parameters. This user interface is always in development and we enjoy every your remarks to this.

References

[1] PolyX, Ltd. *Polynomial Toolbox 3.0, (prerelease)*, <http://www.polyx.com>.

[2] The MathWorks, *Matlab 6, rerelease 13*, <http://www.mathworks.com>.

[3] Henrion, D., Šebek, M., Kučera, V., *An Algorithm for Static Output Feedback Simultaneous Stabilization of Scalar Plants ,Proceedings of the World Congress on Automatic Control, IFAC*, Barcelona, Spain, July 21-26, 2002.

[4] Henrion, D., Šebek, M., *New robust control functions for the Polynomial Toolbox 3.0*, May 17, 2002.

[5] Barnett, S., *Polynomial and Linear Control Systems*.