A Two-Stage Optimization PID Algorithm ⋆

Gísli Herjólfsson * **Anna Soffía Hauksdóttir** *
Sven Þ. Sigurðsson **

* *Department of Electrical and Computer Engineering,University of Iceland (e-mail: gislihe@hi.is and ash@hi.is)*
** *Department of Computer Science, University of Iceland (e-mail: sven@hi.is)*

Abstract: A two-stage PID algorithm is proposed with focus on fulfilling some important general requirements such as settling time, overshoot, size of control signal, disturbance rejection and robustness. The proposed method has two main goals. The first is to create an automatic PID computing algorithm giving decent results for a great variety of systems with different design requirements and different practical requirements. The second goal is to ensure that users with only a basic knowledge of automatic control systems can use the method. The proposed method is tested on 35 well known benchmark examples that have various difficulties of control.

1. INTRODUCTION

Currently, a great majority of control systems in industries are operated by PID controllers and it is estimated that over 90% of all feedback controllers are PID controllers, Åström and Hägglund [2001]. Therefore, a small improvement in PID design could affect industries worldwide and the design of PID controllers remains a very active research topic, O'Dwyer [2003]. A detailed overview of modern PID technology is examined in Ang et al. [2005]. The well known Ziegler and Nichols tuning rule was presented in Ziegler and Nichols [1942] and is still today one of the most used methods when it comes to tuning PID coefficients. The tuning method based on internal model control (IMC) is also well known and is widely used today, see, e.g., Rivera et al. [1986]. Many more tuning methods have been established for PID controllers, see, e.g., Malwatkar et al. [2009] and Shamsuzzoha and Skogestad [2010]. Instead of tuning the PID coefficients experimentally, it is possible to calculate the coefficients from a mathematical model. In Åström et al. [1998], a PI controller is designed based on non-convex optimization. Open-loop PID shaping by directly dealing with frequency-domain inequalities is presented in Hara et al. [2006]. The two-degree-of-freedom PID controller has the ability to control the set point response as well the disturbance rejection in a more decoupled way as opposed to the standard PID controller. This is discussed, e.g., in Carotenuto et al. [2006].

The general problem on how to optimize zero locations, such as to get a system to track a reference system, is reported in Hauksdóttir [2004]. Similar, optimized zero locations have been applied in model reduction in Herjólfsson et al. [2009]. An optimized PID controller has also been developed, tracking a given open loop reference system that effectively includes the design requirements for the corresponding closed loop reference system, Herjólfsson and Hauksdóttir [2003].

In this paper, a two-stage PID algorithm is considered where closed-form zero optimization is combined with an iterative search algorithm in order to minimize a general cost function. The proposed method has two main goals. The first goal is to create an algorithm that automatically computes PID coefficients that give decent results for a variety of systems having different design and practical requirements. The second goal is that users with only a basic knowledge in the field of automatic control systems can use this method, given that a mathematical model of the system to be controlled exist. Thus the method must take into consideration critical factors like settling time, overshoot, undershoot, size of control signal, disturbance rejection and robustness.

2. THE TWO-STEP OPTIMIZATION PID ALGORITHM

The proposed PID optimization algorithm has two main stages, see Figure 1. In both stages, the same general cost function, which is a weighted combination of the critical factors, i.e., settling time, overshoot, undershoot, the magnitude of the PID coefficients, the inverse squared integral coefficient, and error sensitivity, is minimized, but with a different choice of free parameters. In the first stage, the free parameters are two parameters of a second order reference system, directly related to overshoot and settling time. The PID coefficients calculated in each iteration are those obtained by open-loop zero optimization. The zero optimization is based on a closed form expression, resulting in an explicit formula for the optimal PID coefficients. In the second stage, the free parameters are the three PID coefficients themselves. The PID coefficients from the last step of the first stage are used as a starting point for the second stage.

The main reason for the first stage is that, when using an iterative search algorithm for a PID controller, it is vital that the initial PID values result in a stable closed loop, preferably not near a poor local minimum of the cost function. It is not trivial, how to automatically choose such a good and stable starting point when dealing with

⋆ This work was supported in part by the Eimskip Fund of The University of Iceland.

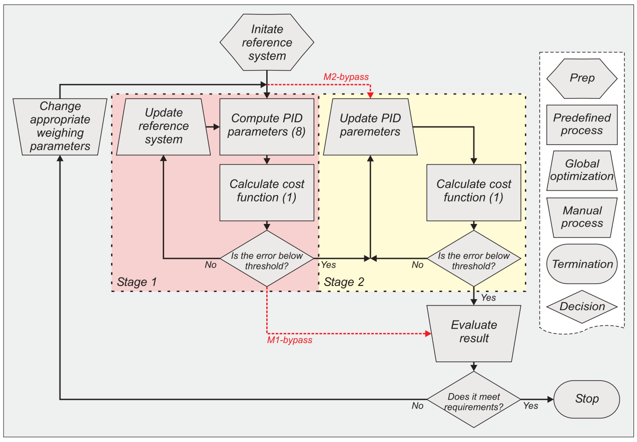

Fig. 1. A flow chart describing the two-stage PID optimization algorithm.

a variety of systems. The zero optimization minimizes the squared integral error between the controlled system and a stable reference response. It is generally easier to select a simple, stable and well behaved initial reference response, rather than finding a stable and well behaved PID controller directly.

The main reason for the second stage, is that in the choice of the second order reference system, the outcome is dominated by the focus on overshoot and settling time, while the relation to design criteria such as robustness and disturbance rejection is less evident. By optimizing directly the PID coefficients, the search space is effectively expanded, in an attempt to further decrease the cost function.

In both these stages, the weight parameters between the various critical factors of the cost function are kept fixed. It may of course be the case that on evaluating the results one may wish to alter these relative weights in order to be able to fulfill the desired design requirements. Thus, e.g., if the control signal is not expensive, one can lower the corresponding weight or if the resulting overshoot is too high, one can increase that corresponding weight parameter.

2.1 The cost function

The cost function is a weighted sum of some properties of the controlled system response, the weights thus give the user the option of changing the behavior of the controller to suit his/her needs. Since different types of systems have different properties we must normalize some terms in the weighted sum of our cost function. As an example two systems can have a similar overshoot but the settling time can differ by a factor of thousands. Therefore, if we would let the cost depend linearly on the settling time, the variations in that term would dominate the cost function and the remaining properties would be neglected, unless the terms are normalized in some way. The following cost function normalizes the cost related to the settling time, overshoot and undershoot with the values of the original uncontrolled system. The cost related to sensitivity to modelling error is uniform for all systems. There are two cost terms, measuring the squared sum of the magnitude of the PID coefficients and the squared inverse of the integral coefficients of the PID controller, that are not normalized. They are however easily changed by the user if needed. The cost function is given by

$$\mathcal{E} = w_T \frac{T_{S,C}}{T_{S,O}} + w_O \frac{O_{S,C}}{O_{S,O}} + w_U \frac{U_{S,C}}{U_{S,O}} + \\ w_P \left(\sum_{i=0}^{2} |c_i| \right)^2 + w_I \frac{1}{c_0^2} + w_S F(R) \qquad (1)$$

where, c_i are the resulting controller coefficients and w denote the weights

- w_T weights the normalized set-point settling time T_S
- w_O weights the normalized set-point overshoot O_S
- w_U weights the normalized set-point undershoot U_S
- w_P balances the sum of the magnitude of the PID coefficients squared
- w_I weights the inverse squared magnitude of the integrator part
- w_S weights the sensitivity to modelling error.

$T_{S,C}$, $O_{S,C}$ and $U_{S,C}$ are the settling time, overshoot and undershoot, respectively, of the controlled system in a closed-loop setup and $T_{S,O}$, $O_{S,O}$ and $U_{S,O}$ are the settling time, overshoot and undershoot of the original system in an open-loop setup, see Figure 2. Here S_L is the settling limit which is defined as 5% of the steady state S_S. The settling time, T_S, is the time it takes for the response to settle inside the settling limit including any time delays. For the case, where the open-loop setup of the original system has a zero pole, the original settling time, $T_{S,O}$, is computed without the integrator in the transfer function. The overshoot is measured as O_S/S_S and the undershoot as U_S/S_S. The overshoot and undershoot are measured in percentage and $O_{S,O}$ and $U_{S,O}$ are forced to be $\geq 1\%$, i.e. $O_{S,O} = max(O_S/S_S \times 100, 1)$. $F(R)$ is a function measuring the sensitivity with respect to model uncertainties.

A standard procedure for measuring the sensitivity to modelling error is to find the inverse of the shortest distance from the Nyquist curve of the loop transfer function to the critical point -1, Åström et al. [1998]. In the above cost function we use

$$F(R) = max(0, R_r - R)^2 \qquad (2)$$

where R is the minimum distance from the Nyquist graph to the critical point and R_r is the radius of a circle around the critical point which we wish to stay outside, see Figure 3. If the system is indeed outside the circle then $F(R) = 0$ and the sensitivity weight does not have any effect, but as we get inside the circle and closer to the critical point, $F(R)$ increases. If the initial point is stable and the step sizes are kept small enough and the weight w_S large enough, in the iterative search algorithm, then the function $F(R)$ will prevent the algorithm from resulting in an unstable controlled system.

Increasing the integrator part of the PID controller affects the load disturbance rejection. We therefore use the weight w_I, multiplied by $\frac{1}{c_0^2}$, to control the size of the integrator factor. The optimization method then tries to increase the integrator part without having too much effect on the set point response. However, increasing only the integrator part may cause an oscillation in the con-

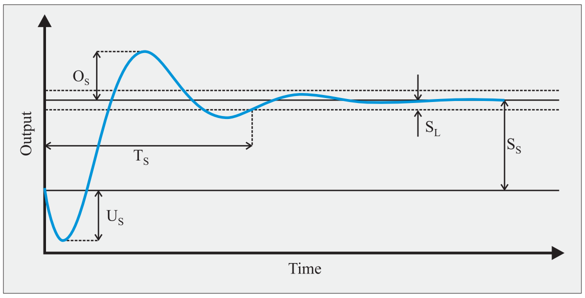

Fig. 2. Explanation of terms used in the cost function (1) for an example of a step response of an underdamped system with undershoot.

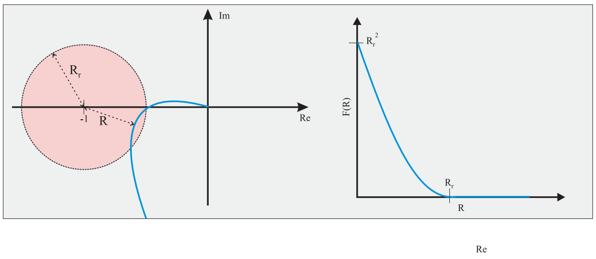

Fig. 3. The left graph shows how R_r and R in (2) are defined. The right graph shows $F(R)$ as a function of R.

trolled system. In that case, it is beneficial to increase all of the PID coefficients, by decreasing the weight w_P. Choosing the weights of the cost function in (1) as $w = [w_T \; w_O \; w_U \; w_P \; w_I \; w_S] = [1\;1\;1\;1\;1\;1\;]$ usually results in a good set point response.

Other terms, e.g. phase- and gain-margins, can be included in the cost function, if desired.

2.2 The iterative search algorithm

The purpose of the iterative search algorithm is to minimize the cost function in Equation (1) by searching for the optimal free parameters in both stages. The free parameters in the first stage are two parameters of an open-loop reference system while in the second stage the free parameters are the PID coefficients themselves. There is a variety of search methods that can be used for this parameter optimization, in this paper we choose to use the Nelder-Mead Simplex Method or Matlab's `fminsearch` function Lagarias et al. [1998], which is widely available.

2.3 The open-loop zero optimization

In the first stage, the PID coefficients that are calculated in each iteration are those that minimize an integral squared error between the controlled system and the reference system in open-loop. A closed form expression for these optimal coefficients is derived as follows. Consider the open-loop stable transfer function

$$G(s) = \frac{b(s)}{a(s)} = \frac{b_m s^m + \cdots + b_1 s + b_0}{s^n + a_{n-1}s^{n-1} \cdots + a_1 s + a_0} \qquad (3)$$

which we wish to control in a closed-loop setup using a standard form PID controller

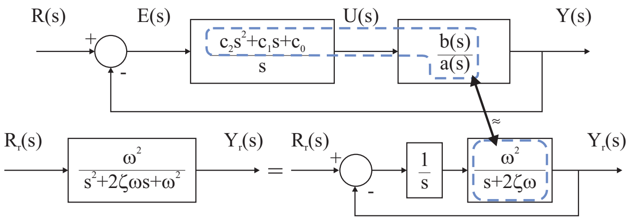

Fig. 4. The controlled system and the reference system in a closed-loop with a single integrator.

$$C(s) = \frac{c(s)}{s} = \frac{c_p s^p + \cdots + c_0}{s} = \frac{K_D s^2 + K_P s + K_I}{s}, \qquad (4)$$

where $p = 2$ is the degree of the polynomial $c(s)$ for a standard PID controller. We introduce an open-loop reference system

$$\frac{b_r(s)}{a_r(s)} = \frac{b_{m_r,r}s^{m_r} + \cdots + b_{0,r}}{s^{n_r} + a_{n_r-1,r}s^{n_r-1} + \cdots + a_{0,r}} = \frac{\omega^2}{s + 2\xi\omega} \qquad (5)$$

corresponding to the standard second order closed-loop system

$$\frac{Y_r(s)}{R(s)} = \frac{\omega^2}{s^2 + 2\xi\omega s + \omega^2}. \qquad (6)$$

The transfer functions systems inside the broken lines in Figure 4 demonstrate the open-loop tracking resulting in the desired closed-loop responses. The PID coefficients that minimize the integrated squared deviation between the transient part of the step response of the controlled system, $\tilde{y}_S(t)$, and the reference system, $\tilde{y}_{S,r}(t)$, in open loop, i.e.,

$$\int_0^\infty \left(\tilde{y}_S(t) - \tilde{y}_{S,r}(t)\right)^2 dt \qquad (7)$$

while constraining the DC-gain of both responses, $\frac{K_I b_0}{a_0}$ and $\frac{\omega}{2\xi}$, respectively, to be the same, can be computed directly by solving a linear system of equations, Herjólfsson et al. [2009]

$$\begin{bmatrix} \mathcal{G} & \dfrac{b_0}{a_0}u_1 \\ \dfrac{b_0}{a_0}u_1^T & 0 \end{bmatrix} \begin{bmatrix} \mathcal{C} \\ \dfrac{\lambda}{2} \end{bmatrix} = \begin{bmatrix} \mathcal{D} \\ \dfrac{\omega}{2\xi} \end{bmatrix}, \qquad (8)$$

where

$$\mathcal{C} = [\, K_I \; K_P \; K_D \,]^T, \qquad (9)$$

$\mathcal{G}$ and $\mathcal{D}$ are a Grammian matrix and a cross Grammian vector that can be obtained from solutions to appropriate Lyapunov and Sylvester equations and u_1 is a unity column vector with the first element set to 1. Here we are essentially computing the PID coefficients such that the controlled system follows the response of the reference system.

There are mainly two reasons why the reference system in the latter half in (5) is used. The first reason is its simplicity, the open-loop only has a single variable pole and a variable DC-gain, in addition to the integrator. The other reason is that the output response of the closed-loop reference system, that we are essentially aiming for, can easily be chosen.

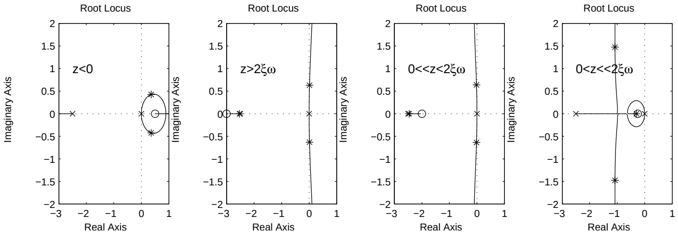

Fig. 5. Different sizes of z when dealing with a double integrator, $\omega = 1$, $\xi = 1.25$ and a dummy pole at -100. Pole locations with unity gain, $K = 1$, are marked with $*$.

2.4 Handling of different system types and controllers

Open-loop causality The zero optimization algorithm requires the controlled and the reference system to be causal, i.e., $m + p < n$ for the controlled system and $m_r < n_r$ for the reference system in an open-loop setup. The simplest way to deal with cases when $m + p \geq n$ and $m_r \geq n_r$ is to add dummy poles, while computing the PID controller, that have a minimum effect on the open loop system response. In general it gives a good result to choose the location of the dummy poles to be approximately 100 times the bandwidth of the system, in rad/s. The original system then becomes

$$\tilde{G}(s) = G(s) \times \left(\frac{1}{s/\lambda_d + 1}\right)^{n_d}, \quad n_d = m+p-n+1 \tag{10}$$

while computing the PID controller where the dummy poles are given by $\lambda_d = 100 \times BW\left(G(s)\right)$.

Time delay There are no stability issues associated with a time delay in an open-loop setup and we note that the open-loop reference system with the added integrator and time delay

$$\frac{b_r(s)}{s a_r(s)} e^{-sT_d} = \frac{\omega^2}{s(s + 2\xi\omega)} e^{-sT_d} \tag{11}$$

can always be selected such that it is stable in closed-loop for a given time delay, T_d. For larger time delays we normally obtain a more stable closed-loop reference system by increasing ξ, thus resulting in a more damped system. It should however be noted, that we need to add the time delay in the outer optimization function when calculating the cost function $\mathcal{E}$, as it is dependent on the closed-loop.

A system integrator It is assumed that the system $\frac{b(s)}{a(s)}$ is strictly stable, i.e. all of the poles are in the left half plane and the same applies for the reference system $\frac{b_r(s)}{a_r(s)}$. There are, however, two ways of dealing with systems that have a single integrator or a zero pole. One is simply to use the given method to calculate a PD controller. We then choose $p = 1$ and include the integrator in the reference system, but move the integrator of the original system to the controller and treat it as a PI controller during the design phase. In other cases, a full PID controller is needed, e.g., for disturbance rejection. In that case, we need to change the reference system such that it has a double integrator like the controlled system. In order for the reference system to be stable in a closed-loop setup, we must add a properly positioned zero. The open-loop reference system, with the double integrator, is now given by

$$\frac{b_r(s)}{s^2 a_r(s)} = \frac{\omega^2(s/z + 1)}{s^2(s + 2\xi\omega)(1/\lambda_d s + 1)} \tag{12}$$

where the zero must be chosen such that $0 < z < 2\xi\omega$ and a dummy pole λ_d is needed for open-loop causality. The root locus of this reference system can be seen in Figure 5 for the cases $z < 0$, $0 < z < 2\xi\omega$ and $z > 2\xi\omega$ and a dummy pole at $\lambda_d = -100$.

Sys.	G(s)	Ind.
1	$\frac{1}{(s+1)^n}$	$n = 1, 2, 3, 4, 8$
2	$\frac{1}{(s+1)(1+\alpha s)(1+\alpha^2 s)(1+\alpha^3 s)}$	$n = 0.1, 0.2, 0.5, 1$
3	$\frac{1-\alpha s}{(s+1)^3}$	$\alpha = 0.1, 0.2, 0.5, 1, 2, 5$
4	$\frac{1}{(1+sT)} e^{-s}$	$T = 0, 0.1, 0.2, 0.5, 2, 5, 10$
5	$\frac{1}{(1+sT)^2} e^{-s}$	$T = 0, 0.1, 0.2, 0.5, 2, 5, 10$
7	$\frac{100}{(s+10)^2}\left(\frac{1}{s+1} + \frac{0.5}{s+0.05}\right)$	
8	$\frac{(s+6)^2}{s(s+1)^2(s+36)}$	
9	$\frac{\omega_0^2}{(s+1)(s^2+2\zeta\omega_0 s+\omega_0^2)}$	$\zeta = 0.1, \omega_0 = 1, 2, 5, 10$

Table 1. A set of the 35 benchmark examples.

Approximate derivative A common practice when implementing a PID controller is to use a low pass pole associated with the derivative term, especially in a noisy environment and in order to reduce spikes in the control signal. It is easy to incorporate the pole into the optimization method, essentially as an additional system pole, such that it is used in the computation of the PID parameters. The PID controller with the low pass pole is given by

$$\frac{\hat{c}(s)}{s} = \frac{\hat{c}_2 s}{(s/\lambda_a + 1)} + \hat{c}_1 + \frac{\hat{c}_0}{s} \tag{13}$$

where λ_a is the low pass pole location. Note that

$$\frac{\hat{c}(s)}{s} = \frac{c_2 s^2 + c_1 s + c_0}{s(s/\lambda_a + 1)} \tag{14}$$

where $\hat{c}_0 = c_0$, $\hat{c}_1 = c_1 - \hat{c}_0/\lambda_a$ and $\hat{c}_2 = c_2 - \hat{c}_1/\lambda_a$. In order to add the pole into the optimization, we let

$$\tilde{G}(s) = \frac{b(a)}{a(s)(s/\lambda_a + 1)} \tag{15}$$

be the original system to be controlled and calculate $c(s)$ and then we can retrieve the $\hat{c}(s)$ parameters to implement a proper PID controller with a low pass pole on the derivative term.

PI and PD controllers The proposed method can easily be altered to compute a PI or PD controller instead of a full PID controller. In the case of a PI controller, we select $p = 1$, i.e., we only compute a single zero and keep the integrator when closing the loop, leaving the reference system the same as before. When computing a PD controller, we let $p = 1$ and skip the integrator in the reference system, unless the original system contains a pure integrator.

3. BENCHMARK EXAMPLES

The two-stage optimization method has been tested on 35 well known benchmark examples, introduced in Åström and Hägglund [2000]. The system's transfer functions are shown in Table 1. The same starting point and weight

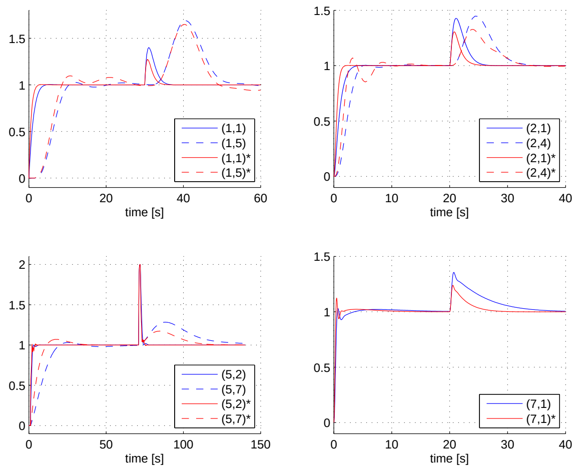

Fig. 6. Unit step responses and response to a unit step disturbance for a subset of the benchmark examples. Results labelled with * were obtained by decreasing the weight w_P and increasing the weight w_I.

parameters are used for all of the benchmark sets. The weighing parameters are set to

$$[w_T \quad w_O \quad w_U \quad w_P \quad w_I \quad w_S] = [1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1], \tag{16}$$

and the open-loop reference system is given by Equation (5), where $\omega = 1$ and $\xi = 20$ are used as the initial values. The radius of the reference circle in $F(R)$ in (2) is set to $R_r = 0.5$. For the case where the benchmark system has a zero pole, the reference system in (12) is used with the same initial values of ω and ξ as before and $z = \xi\omega/20 = 1$ as the initial value. The resulting PID parameters are listed in Table 2. Note that in some cases the resulting PID controller has RHP zeros, see e.g. results in systems $(2, 1)$ and $(2, 2)$. In these examples the undershoot of the controlled systems is not noticeable and the resulting response is very good. However RHP zeros can be prevented, if necessary, by increasing the weight w_U until even the smallest undershoot results in a measured cost. The resulting unit step input and disturbance responses for a subset of the benchmark examples is shown in Figure 6. This selection of weights, all equal to one, normally results in a decent set point response with a low overshoot and good disturbance rejection with moderately sized PID coefficients and control signal. If the size of the control signal is not a vital factor, there may be room to increase the emphasis on the disturbance rejection even further, by allowing larger PID coefficients in the cost function.

The difference between the methods used in each stage has been examined in order to demonstrate what is gained by using the two-stage method instead of using just one of the two stages. We therefore compare the three methods:

- **Method 1 (M1):** Uses only the first stage of the method, i.e. the zero optimization to compute the PID coefficients and the global search method to find the reference system's ξ and ω.
- **Method 2 (M2):** Uses only the second stage, i.e. the global search method is used directly on the PID coefficients.
- **Method 3 (M3):** Is the proposed two-stage method, see Figure 1.

The same starting points apply for M1 and M3. We need to find a starting point for the PID parameters in M2, which is nontrivial if we are to use the same starting point for all 35 systems. After some trials, the following starting point was identified

$$\mathcal{C}_{init} = [K_I \; K_P \; K_D] = [0.1 \quad 0.1 \quad 0] \tag{17}$$

which worked for all systems. Where the system has a zero pole, the integral coefficient is put to zero, i.e., $K_I = 0$. A comparison of the three methods is done in such a way, that for each benchmark system, the output error for each method is divided by the sum of the error of the three methods for the corresponding system. Therefore each method is graded by the following normalized formula

$$Gr(m, n) = \frac{\mathcal{E}(m, n)}{\sum_{k=1}^{3} \mathcal{E}(k, n)}, \qquad m = 1, 2, 3 \text{ and } n = 1, \ldots, 35 \tag{18}$$

where a lower grade gives less error and thus a better result. Here $\mathcal{E}(m, n)$ is the resulting value of the cost function of method m for system n given in (1). By using this grading method, each system weighs the same and the grades are normalized. The average grades for the methods are

$$\begin{aligned} M1 &: \overline{Gr(1, n)} = 0.35 \\ M2 &: \overline{Gr(2, n)} = 0.36 \\ M3 &: \overline{Gr(3, n)} = 0.29 \end{aligned} \tag{19}$$

indicating that using both stages gives better results, than using just one of them. It should also be noted that combined method M3 showed consistency on all of the benchmark systems while the other methods were more inconsistent.

We now look at the effect of changing the weights in the algorithm for a subset of the benchmark examples. We compare the original setup of the weights all set to 1, to a setup with more emphasis on disturbance rejection. Then the weight w_P is decreased from 1 to 0.5 and the weight w_I is increased from 1 to 10. This setup allows the algorithm to increase the size of the PID coefficients, in particular the integrator part. The resulting PID coefficients are shown in Table 2. The resulting unit step input and disturbance responses for a subset of the benchmark examples, are shown in Figure 6. By looking at the resulting PID coefficients we have 20% reduction in the peak disturbance response, on the average for all of the systems. Users can however change the weights as they see fit in order to change the results to suit their needs.

4. CONCLUSION

A two-step optimization PID algorithm has been formulated in this paper. One of the goals was to create a method that works for a variety of linear systems with decent results. The other goal was to make the method such that users with little expertise in control theory can use and fine-tune it to suit their needs. The proposed PID optimization algorithm has two main stages. In both stages, the same general cost function is minimized but with a different choice of free parameters. In the first stage, the free parameters are two parameters of a second order reference response function, effectively reflecting some design requirements. The corresponding PID coefficients are then obtained by open-loop zero optimization in each iteration.

(Sys,Ind)	$[K_D\ K_P\ K_I]$	T_S[s]	O_S[%]	$[K_D\ K_P\ K_I]$*
(1,1)	[0.00 0.78 0.96]	3.4	0.3	[0.00 1.68 1.83]
(1,2)	[0.10 1.12 0.68]	3.2	0.0	[1.03 2.59 1.36]
(1,3)	[0.83 1.38 0.56]	3.9	0.0	[2.28 2.61 1.05]
(1,4)	[1.21 1.27 0.42]	4.5	0.5	[2.99 1.95 0.58]
(1,5)	[1.59 0.80 0.16]	25.4	2.6	[2.28 0.85 0.21]
(2,1)	[-0.05 0.79 0.95]	3.2	0.2	[-0.07 1.70 1.81]
(2,2)	[-0.03 0.86 0.94]	2.9	0.2	[0.08 1.83 1.70]
(2,3)	[0.20 1.05 0.73]	3.0	0.0	[0.94 2.25 1.43]
(2,4)	[1.21 1.27 0.42]	4.5	0.5	[2.99 1.95 0.58]
(3,1)	[0.88 1.35 0.55]	4.0	0.1	[2.13 2.35 0.92]
(3,2)	[0.59 1.19 0.46]	8.3	0.0	[1.66 1.90 0.79]
(3,3)	[0.46 0.93 0.37]	5.0	0.0	[1.22 1.40 0.58]
(3,4)	[0.66 0.89 0.35]	4.3	0.5	[0.95 1.06 0.43]
(3,5)	[0.42 0.58 0.23]	6.4	0.5	[0.49 0.61 0.25]
(3,6)	[0.04 0.18 0.09]	15.4	0.8	[0.01 0.21 0.10]
(4,1)	[0.00 0.26 0.70]	5.6	2.1	[0.00 0.00 0.46]
(4,2)	[0.00 0.28 0.64]	2.4	0.1	[0.00 0.28 0.65]
(4,3)	[0.00 0.30 0.61]	2.5	0.0	[0.00 0.31 0.61]
(4,4)	[0.00 0.40 0.54]	3.0	0.3	[0.00 0.40 0.54]
(4,5)	[0.34 1.38 0.61]	3.0	0.1	[0.59 1.69 0.74]
(4,6)	[0.09 2.12 0.42]	5.0	0.3	[1.46 3.85 0.77]
(4,7)	[0.00 3.03 0.32]	8.1	0.7	[1.82 6.19 0.68]
(5,1)	[0.00 0.26 0.70]	3.0	0.7	[0.00 0.00 0.46]
(5,2)	[0.00 0.29 0.59]	2.7	0.1	[0.08 0.41 0.69]
(5,3)	[0.00 0.32 0.53]	3.1	0.2	[0.13 0.55 0.70]
(5,4)	[0.28 0.74 0.59]	5.1	2.7	[0.33 0.78 0.64]
(5,5)	[1.52 1.58 0.39]	5.0	0.0	[2.63 2.12 0.56]
(5,6)	[3.08 2.00 0.19]	11.0	0.8	[9.49 3.80 0.44]
(5,7)	[5.33 2.28 0.11]	29.9	3.1	[15.79 4.07 0.26]
(7,1)	[0.05 2.67 0.55]	8.9	3.0	[0.36 3.76 1.64]
(8,1)	[4.56 1.79 0.26]	14.5	7.0	[7.01 2.38 0.84]
(9,1)	[0.80 0.19 0.82]	19.0	16.0	[1.74 0.31 1.76]
(9,2)	[0.22 0.19 0.98]	16.1	12.6	[0.62 0.31 1.99]
(9,3)	[0.05 0.44 0.59]	6.5	1.3	[0.04 0.46 0.68]
(9,4)	[0.00 0.70 0.86]	4.0	0.2	[0.06 1.05 1.13]

Table 2. The original PID coefficients and resulting settling time and overshoot from the first examples compared to the PID coefficients with the altered weights (marked with *).

In the second stage, the free parameters are the three PID coefficients themselves. The general cost function is a weighted sum of terms measuring vital factors in the resulting system response. In both stages the weights are kept fixed. The users can however change the weights after evaluating the initial outcome in order to better fulfill their design requirements. The method has been tested on 35 well known benchmark systems, where the same setup was successfully used on all of the models. The two basic stages in the method were compared to the two-staged method to further show how the two-stage method improves each stage. Then a new set of PID coefficients were computed on a subset of the benchmark systems with more emphasis on disturbance rejection, showing a 22% reduction in the peak disturbance response, on the average.

REFERENCES

K. H. Ang, G. Chong, and Y. Li. PID control system analysis, design, and technology. *Control Systems Technology, IEEE Transactions on*, 13(4):559 – 576, Jul. 2005.

K. J. Åström and T. Hägglund. The future of PID control. *Control Engineering Practice*, 9(11):1163 – 1175, 2001.

K. J. Åström, H. Panagopoulos, and T. Hägglund. Design of PI controllers based on non-convex optimization. *Automatica*, 34(5):585 – 601, 1998.

K.J. Åström and T. Hägglund. Benchmark systems for PID control. In *Proceedings of the IFAC workshop, Digital Control: Past, Present and Future of PID Control*, pages 165–166, 2000.

L. Carotenuto, P. Pugliese, and Ya. D. Sergeyev. Maximizing performance and robustness of PI and PID controllers by global optimization. *Control and Intelligent Systems*, 34(3):225–235, 2006.

S. Hara, T. Iwasaki, and D. Shiokata. Robust PID control using generalized KYP synthesis: direct open-loop shaping in multiple frequency ranges. *Control Systems, IEEE*, 26(1):80 – 91, Feb. 2006.

A.S. Hauksdóttir. Optimal zero locations of continuous–time systems with distinct poles tracking reference step responses. *Dynamics of Continuous, Discrete, and Impulsive Systems, Series B: Applications & Algorithms*, 11:353–361, 2004.

G. Herjólfsson and A. S. Hauksdóttir. Direct computation of optimal PID controllers. In *Proceedings of the 42nd IEEE Conference on decision and control*, pages 1120–1125, Maui, Hawaii, Dec. 2003.

G. Herjólfsson, B. Ævarsson, A.S. Hauksdóttir, and S.Þ. Sigurðsson. Closed form $\mathcal{L}_2/\mathcal{H}_2$ optimizing of zeros for model reduction of linear continuous time systems. *International Journal of Control*, 82:555–570, 2009.

J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright. Convergence properties of the Nelder-Mead simplex method in low dimensions. *SIAM Journal of Optimization*, 9:112–147, 1998.

G.M. Malwatkar, S.H. Sonawane, and L.M. Waghmare. Tuning PID controllers for higher-order oscillatory systems with improved performance. *ISA Transactions*, 48 (3):347 – 353, 2009.

A. O'Dwyer. PID compensation of time delayed processes 1998-2002: a survey. In *American Control Conference. Proceedings of the*, volume 2, pages 1494 – 1499, 4-6, 2003.

D. E. Rivera, M. Morari, and S. Skogestad. Internal model control: PID controller design. *Industrial & Engineering Chemistry Process Design and Development*, 25(1):252–265, January 1986.

N. Shamsuzzoha and S. Skogestad. The setpoint overshoot method: A simple and fast closed-loop approach for PID tuning. *Journal of Process Control*, 20(10):1220 – 1234, 2010.

J. G. Ziegler and N. B. Nichols. Optimum Settings for Automatic Controllers. *Transactions of ASME*, 64:759–768, 1942.