When is PID a good choice?

Soltesz, Kristian; Cervin, Anton

Published in:
IFAC-PapersOnLine

DOI:
10.1016/j.ifacol.2018.06.074

2018

Document Version:
Peer reviewed version (aka post-print)

When is PID a good choice?

Kristian Soltesz [*] **Anton Cervin** [*]

[*] *Dept. Automatic Control, Lund University*
P.O. Box 118, SE-221 00, Lund, Sweden.
(e-mail: {kristian,anton}@control.lth.se)

Abstract: A new and freely available model-based PID design tool for Matlab is introduced. It can be used to solve Maximal Integral Gain Optimization (MIGO) and (load) Integral Absolute Error (IAE) problems. Robustness is ensured through $\mathcal{H}_\infty$ constraints on the closed-loop transfer functions. A Youla parameter (Q design) method for comparison with the optimal linear time-invariant (LTI) controller for the considered IAE optimization problem is presented. Several realistic design examples are provided, in which the tool is used to compare achievable PID and optimal LTI controller performance, to illustrate whether PID is a good choice for a given combination of process dynamics and closed-loop robustness requirements.

Keywords: PID control, Youla parametrization, control performance, constrained optimization

1. INTRODUCTION

1.1 Outline

This paper compares achievable performance of PID controllers using PIDopt[1] , which is a free, open-source tool for model-based design of (filtered) PID controllers. Designs solve either of two constrained optimization problems, using local algorithms introduced in Hast and Hägglund (2015); Soltesz et al. (2017). In addition, PIDopt provides a means of investigating how large room for improvement a certain design has, if the controller type is relaxed from PID to arbitrary-order linear time-invariant (LTI).

The paper is organized as follows: The two PID design problems are introduced in Section 1.2 and further detailed in Section 2. A general background with references to prior work is given in Section 1.4. The Q (Youla) design method is introduced in Section 2.2. Section 3 features illustrative comparisons, in which the two PID and the Q design methods are compared in terms of achievable performance. A general discussion is provided in Section 4.

1.2 Design problem formulations

The setting of Figure 1 is considered. The objective is to design a controller K, which stabilizes the process output y in the presence of the load disturbance d, measurement noise n, and some degree of uncertainty in the LTI process dynamics P. The regulator problem is considered, explaining the lack of a reference input in Figure 1. This is sufficient, since the servo problem can be treated separately through two degree-of-freedom designs, once K has been designed, see Hast and Hägglund (2015).

A continuous-time setting is assumed, with asymptotically stable single-input single-output (SISO) dynamics P.

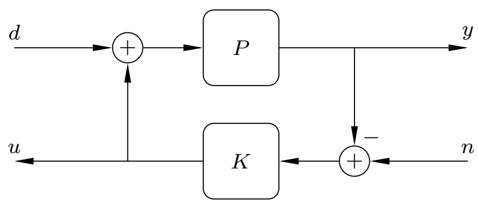

Fig. 1. Closed-loop control system with process P, controller K, process output y, load disturbance d, control signal u, and measurement noise n.

None of these assumptions are a result of limitations: the algorithms are readily portable to discrete-time process and controller dynamics; they can be directly applied to unstable dynamics, provided that they are initiated with the parameters of a stabilizing initial controller; multiple-input multiple-output (MIMO) dynamics can be treated with the same techniques. See Boyd et al. (2015) for a MIMO extension of the IE algorithm, presented below. (Extending the exact formulation used in this paper requires one of several possible generalizations of the cost function to the MIMO setting.)

The two studied design problems are the load step Integral Error (IE) minimization problem, hereafter referred to as the *IE problem*, and the load step Integral Absolute Error (IAE) minimization problem, hereafter referred to as the *IAE problem*. The IE problem is also known as the Maximal Integral Gain Optimization (MIGO) problem, as maximizing PID integral gain leads to minimization of load step IE, whenever the loop transfer function contains at least one integrator, see Panagopoulos et al. (2002). Both problems are of high practical significance and industrial recognition. They are constrained optimization problems of the form

$$\begin{aligned} \underset{\theta}{\text{minimize}} \quad & J(\theta), \\ \text{subject to} \quad & \|S\|_\infty \leq M_s, \\ & \|T\|_\infty \leq M_t, \\ & \|KS\|_\infty \leq M_{ks}, \end{aligned} \tag{1}$$

[*] The authors are members of the LCCC Linnaeus Center and the ELLIIT Excellence Center at Lund University.

[1] PIDopt is available under MIT license at `https://gitlab.control.lth.se/kristian/PIDopt`.

where $S = (1+PK)^{-1}$ is the sensitivity function, $T = 1-S$ is the complementary sensitivity function, and KS is the noise sensitivity function (the transfer function from measurement noise to control signal). Robustness is enforced through the user-specified constraint vector $M = [M_s \; M_t \; M_{ks}]^\top$. Assuming that d is a step disturbance, the IE problem aims to minimize the time integral of the resulting output,

$$J_{IE} = \int y(t)dt, \tag{2}$$

while the IAE problem aims to minimize the time integral of its modulus,

$$J_{IAE} = \int |y(t)|dt. \tag{3}$$

The parameter vector θ of (1) parametrizes the (filtered) PID controller $K = FC$, where

$$C(s) = k_p + \frac{k_i}{s} + k_d s \tag{4}$$

is an ideal PID controller, linear in its parameters $\theta_c = [k_p \; k_i \; k_d]^\top$. A conventional second-order filter,

$$F(s) = \frac{1}{s^2T^2 + 2\zeta sT + 1}, \tag{5}$$

is used to guarantee high-frequency roll-off for the common case of at most one derivator (factor s) in the loop transfer function PK. The PIDopt IE problem algorithm only works for fixed $T = 0$ in (5), i.e., for designing an ideal, unfiltered PID controller $K(s) = C(s)$ with parameter vector $\theta = \theta_c$. For the IAE (3) design, a filtered PID controller is considered, through the addition of the filter time constant, T, to the parameter vector: $\theta = \left[\theta_c^\top \; \theta_f\right]^\top$, where $\theta_f = T$. The relative damping ratio is fixed to $\zeta = 1/\sqrt{2}$, but can readily be treated as a free optimization variable, $\theta_f = [T \; \zeta]^\top$, although the resulting benefit is limited and outweighed by increased execution time as pointed out in Soltesz et al. (2017). Relatedly, it is possible to omit elements of θ in both algorithms. For instance, $k_d = 0$ can be enforced to design a PI type controller.

The IE algorithm has an advantage in that $J(\theta) = -k_i$, is convex (even linear) in θ. This enables the utilization of an efficient convex-concave technique introduced in Yuille and Rangarajan (2003). For practical applications, it is recommended to use a low-pass filter F, which provides high-frequency roll-off, as explained in Hägglund (2012). Unfortunately, the introduction of F voids constraint concavity in the IE problem (upon which the convex-concave technique relies), meaning that F needs to be chosen prior to optimizing C. (This process can be iterated in order to find the optimal K for the IE problem.) Another drawback of the IE problem formulation, exemplified in Section 3.4, is that it can lead to closed-loop systems with oscillatory load responses if robustness is not sufficiently enforced through M.

The IAE problem makes use of a gradient-based technique, enabling simultaneous optimization of C and F. Its main disadvantage is that the evaluation of the Jacobian, needed in each optimization step, makes the algorithm somewhat slower (typical 1 min) than the very fast (typical 2 s) IE algorithm [2]. The quoted times assume that the IE algorithm has been used to provide an initial parameter vector for the IAE algorithm.

In addition to finding the optimal PID controller—in the sense of (1), with objective being either (2) or (3)—it is relevant to know how much room there is for improvement if the class of possible K is expanded beyond (filtered) PID controllers. To this end, PIDopt provides an algorithm for solving (1) with the IAE objective (3) for LTI K of arbitrary order. This is done using a Youla (Q) parameter design problem, hereafter referred to as the *Q problem*, described in Section 2.2 and demonstrated in Section 3.1.

1.3 Practical applicability

The purpose of the developed tool is to answer, for a particular set of constraint levels of (1), whether it is worthwhile to use a more complex (higher-order) LTI controller, for the dynamics of a certain stable SISO LTI process. This might be worthwhile if the performance benefit, in terms of (1), is deemed sufficiently large by the user. The tool could also be used to answer the question of the title in a broader sense—by categorizing for which types of dynamics and which constraint levels the mentioned difference becomes large, although such a study is outside the scope of this paper. The tool is currently in a proof-of-concept state and offers a set of Matlab functions for the various designs described in this paper.

1.4 Prior work

PID tuning techniques can roughly be divided into two categories: heuristic rules of thumb (such as Skogestad (2003)), and fully model-based techniques (such as the ones in this paper). The main advantage of the latter is that they are tailored for a specific process, with specific design objectives. Their main disadvantage is that solving the design problem involves more complex computer software than what is needed for most heuristic tuning rules.

The prospect of designing PID controllers, with or without filter, through solving constrained optimization problems, has been investigated in several works. Particularly for $\mathcal{H}_\infty$ and mixed $\mathcal{H}_\infty/\mathcal{H}_2$ problems, there exist efficient approaches, as implemented in for example Matlab's `hinfstruct` based on Apkarian and Noll (2006), and the toolbox of Sadeghpour et al. (2012). While IE, and particularly IAE, are industrially recognized performance measures, they cannot be minimized within the $\mathcal{H}_\infty$ and $\mathcal{H}_\infty/\mathcal{H}_2$ frameworks, motivating the development of new methods. The problems described herein have been previously studied (see for example Panagopoulos et al. (2002); Garpinger (2015)), but thus far their solutions have relied on slow simplex-type optimization methods. There is also a rich literature on PID design by solving closely related constrained optimization problem, see Kristiansson and Lennartson (2002); Sekara and Matausek (2009) for a few examples.

[2] Both algorithms have been implemented in Matlab, and no optimization for execution speed has been considered. The typical execution times are representative for a normal desktop or laptop computer 2018.

In this paper, we will utilize a set of 124 asymptotically stable processes, which was used to arrive at the AMIGO tuning rule in Åström and Hägglund (2004). (The original set contains 134 processes, of which 10 contain an integrator, making them only marginally stable.)

2. METHOD

2.1 PID design

Detailed descriptions of the IE and IAE algorithms, beyond that of Section 1.2, are provided in Hast et al. (2013), Soltesz et al. (2017), and through the PIDopt git repository [3].

Both algorithms evaluate constraints over a discrete angular frequency grid. The default grid is generated by applying the Matlab `bode` command to P to get a frequency range, which is then extended both upward and downward by one decade (to cater for designs where the closed-loop bandwidth is pushed either up or down by the controller). A grid comprising $N = 10^3$ logarithmically spaced points has proven sufficient for all thus far considered designs.

In all design examples of Section 3, the IE algorithm has first been applied to obtain a parameter vector, with which the IAE algorithm was initialized. For cases where $M_{ks} < \infty$, the IE algorithm has been configured to yield a PI controller, as the lack of high-frequency roll-off (due to $F = 1$) would force $k_d \to 0$ if sufficiently high frequencies were considered in the optimization. The resulting parameter vector was extended with $[k_d\ T]^\top = [0\ 0]^\top$ before it was passed to the IAE algorithm. The result of the IAE design is in turn used as a nominal controller in the Q design, as explained below.

2.2 Q design

Looking beyond PID controllers, the Q parametrization together with convex optimization can be used to search for an LTI controller of arbitrarily high order that solves (1), assuming the IAE criterion (3). The general procedure is outlined in Boyd and Barratt (1991) and is here specialized for the design problem at hand.

Given that P is stable, all possible stable closed-loop systems can be characterized via a stable Q parameter

$$Q = \frac{K}{1 + PK}. \tag{6}$$

The sensitivity functions of interest can be expressed as

$$\begin{aligned} S &= 1 - PQ, \\ T &= PQ, \\ KS &= Q, \end{aligned} \tag{7}$$

while the unit load step response is given by

$$Y(s) = P(s)S(s)s^{-1} = \big(P(s) - P^2(s)Q(s)\big)s^{-1}. \tag{8}$$

The above expressions are affine in Q, which together with the norms in (1) and (3) imply that the problem is convex. The IAE-optimal filtered PID controller, here denoted K_0, is used as a starting point in the optimization.

[3] The PIDopt git repository (subject to change) is found at `git@gitlab.control.lth.se:kristian/QPID.git`.

To facilitate the search for the optimal Q, a discrete-time formulation is used. The sample time is selected as $h = 0.1/\omega_b$, where ω_b is the 3 dB bandwidth of T when the IAE-optimal filtered PID controller is used. This corresponds to a Nyquist frequency about 30 times larger than the closed-loop bandwidth. The sample time is however lower bounded by $L/15$, where L is the deadtime of the process, to avoid sampled models of very high order.

The nominal controller and the process are approximated by their first-order hold equivalents:

$$\begin{aligned} K_{0d}(z) &= \mathcal{FOH}(K_0(s), h), \\ P_d(z) &= \mathcal{FOH}(P(s), h). \end{aligned} \tag{9}$$

The Q parameter is expressed as the discrete-time Ritz approximation

$$Q_d(z) = Q_{0d}(z) + \sum_{k=1}^{N_q} x_k Q_{kd}(z), \tag{10}$$

where x_k are the scalar variables to be optimized, N_q is the order of the approximation, and $Q_{kd}(z) = z^{k-1}$. The constant term is given by

$$Q_{0d}(z) = \frac{K_{0d}(z)}{1 + P_d(z)K_{0d}(z)} \tag{11}$$

and ensures that the Q design will be at least as good as the sampled filtered PID, even with very few terms in (10). The IAE objective (3) is discretized as

$$J_d = h \sum_{k=0}^{N_t - 1} |y(kh)|, \tag{12}$$

where $N_t = \lfloor T_{0.5\%}/h \rfloor$, $T_{0.5\%}$ being the time until the magnitude of the response has settled below 0.5% of its peak value.

Letting $S_{sd}(z) = C_{sd}(zI - A_{sd})^{-1}B_{sd} + D_{sd}$, where the Q parameter appears affinely in B_{sd} and D_{sd}, the discrete-time maximum sensitivity constraint

$$\|S_{sd}(e^{i\omega})\|_\infty \leq M_s \tag{13}$$

can be expressed as a frequency-independent linear matrix inequality (LMI)

$$\begin{bmatrix} P_{sd} & A_{sd}P_s & B_{sd} & 0 \\ P_{sd}^T A_{sd}^T & P_{sd} & 0 & P_{sd}C_{sd}^T \\ B_{sd} & 0 & 1 & D_{sd}^T \\ 0 & C_{sd}P_{sd}^T & D_{sd} & M_s^2 \end{bmatrix} \succ 0, \tag{14}$$

with P_{sd} being a symmetric matrix variable of the same size as A_{sd}, see De Oliveira et al. (2002). Similar LMIs hold for M_t and M_{ks}. (It can be noted that transforming constraints into LMIs for continuous-time PID designs is only possible for delay-free process dynamics, as delays do not have a finite-dimensional continuous-time state-space representation.)

Finally, to guarantee that the IAE converges as $t \to \infty$, the controller must have integral action, implying $S_{sd}(1) = 0$. This can be enforced by adding the steady-state constraint

$$Q_d(1) = P_d^{-1}(1). \tag{15}$$

All above constraints are closed-loop convex, meaning that a solution can be found efficiently. Once the optimal Q_d has been found, the controller is recovered as

$$K_d = \frac{Q_d}{1 - Q_d P_d}. \tag{16}$$

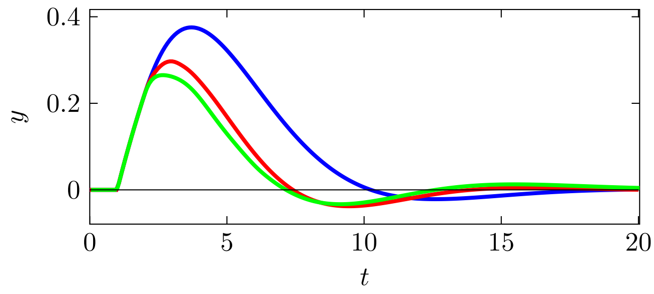

Fig. 2. Load step responses of the closed-loop with process P of (17) and either of the IE-optimal PI (blue), IAE-optimal filtered PID (red), Q-optimal high-order LTI (green) controllers. All controllers honor the constraint levels $M_s = M_t = 1.5$, $M_{ks} = 10$.

The optimization problem is specified and solved in Matlab using CVX, see Grant and Boyd (2014), with the MOSEK solver, see MOSEK ApS (2015), using default settings. Through experiments, it was found that $N_q = 50$ represents a good compromise between optimization time and quality of the approximation. Execution times for finding a controller is around 20–60 seconds and mainly depends on the sampled process model order.

3. COMPARISONS

3.1 A case example

In this example, the process

$$P(s) = \frac{1}{4s+1}e^{-s}, \tag{17}$$

being a lag dominated process with normalized time delay $\tau = 0.2$, is considered.

The $\mathcal{H}_\infty$ constraints on S and T are set to $M_s = M_t = 1.5$, being reasonable values close to those used to arrive at the heuristic SIMC, Skogestad (2003), and AMIGO (approximate MIGO), Åström and Hägglund (2004), tuning rules. While S, and consequently $T = 1 - S$, are both unitless, the noise sensitivity KS is not. Consequently, the choice of M_{ks} is problem instance dependent and it is not possible to recommend a general default. In this example we will use $M_{ks} = 10$, which will be put into context by subsequent design examples.

For each process, a PI controller (blue) was designed by solving the IE problem, a filtered PID controller (red) was designed by solving the IAE problem, and a high-order LTI controller (green) was designed by solving the Q problem. Figure 2 shows the resulting load step responses. Bode plots of the controllers are shown in Figure 3. The Bode magnitudes of S, T and KS are shown together with corresponding constraint levels M_s, M_t, and M_{ks} (black) in Figure 4.

Parameters (rounded to 4 decimals) of the resulting designs are shown in Table 1; resulting IAE and constraint function values in Table 2. Active constraints (in practice) are shown in italic. The filtered PID controller of the IAE design resulted in an IAE decrease of 37%, compared to the PI controller of the IE design. The Q design further decreased IAE by 11%.

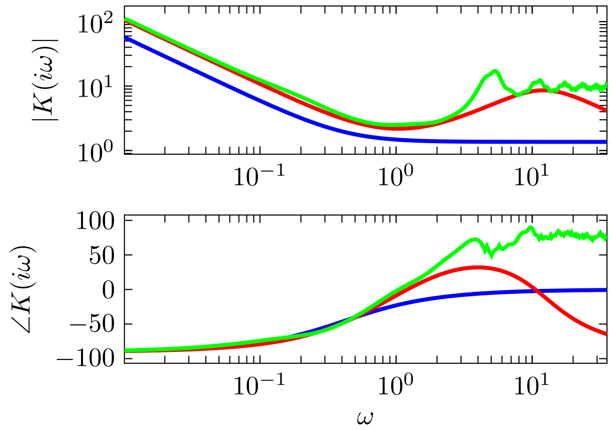

Fig. 3. Controller Bode plots. Controllers and colors as in Figure 2.

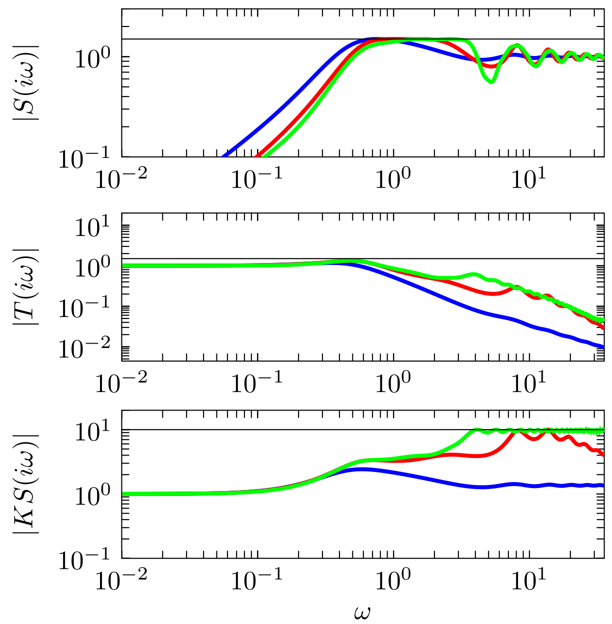

Fig. 4. Bode magnitudes of S, T, and KS. Controllers and colors as in Figure 2. Black lines show $\mathcal{H}_\infty$ constraints.

Table 1. Controller and filter parameters for the IE and IAE design problems for the example of Section 3.1.

Problem	k_p	k_i	k_d	T
IE	1.3620	0.5768	0	0
IAE	2.1786	1.0698	0.9838	0.0817

Table 2. IAE and constraint function values for the IE, IAE, and Q design problems of the example in Section 3.1.

Problem	IAE	$\|S\|_\infty$	$\|T\|_\infty$	$\|KS\|_\infty$
IE	1.9322	*1.5000*	1.1741	2.4232
IAE	1.2194	*1.5000*	1.3336	*10.0000*
Q	1.0843	*1.5000*	1.3278	*9.9997*

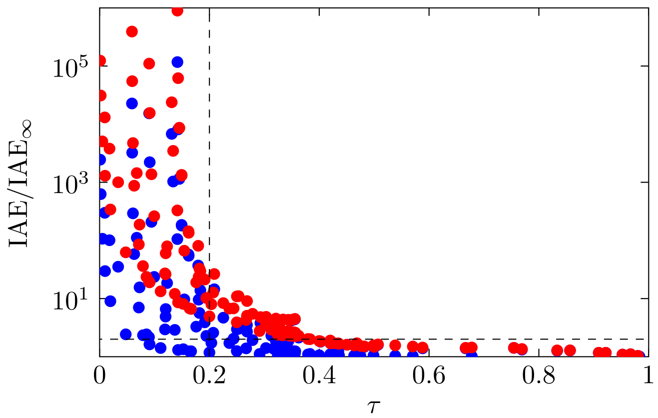

Fig. 5. A representative view of how achievable performance is affected by constraining noise sensitivity for processes of varying relative time delay, τ. Blue marks show performance deterioration, IAE$_{10}$/IAE∞, when going from $M_{ks} = \infty$ to $M_{ks} = 10$. Red marks show IAE$_1$/IAE$_\infty$ (same notation).

3.2 The influence of noise sensitivity

Before further comparing solutions of the IAE and Q problems, it is worthwhile to characterize how the noise sensitivity constraint level, M_{ks}, affects performance of the filtered PID controllers, which solve the IAE problem. To this end, designs for $M_s = M_t = 1.5$ and $M_{ks} \in \{1, 10, \infty\}$ were obtained for the process set mentioned in Section 1.4.

Performance (IAE) was computed for the unconstrained design (IAE$_\infty$), the design with $M_{ks} = 10$ (IAE$_{10}$), and the design with $M_{ks} = 1$ (IAE$_1$). Figure 5 shows IAE$_{10}$/IAE$_\infty$ (blue), and IAE$_1$/IAE$_\infty$ (red), plotted against the normalized time delay, τ, for each process. (For higher-order processes, the generalization $\tau = L/T_{ar}$, where T_{ar} is the average residence time, with $T_{ar} = L + T$ for FOTD processes, has been used.)

One can note—for the considered process set—that M_{ks} plays an imperial role for lag-dominated processes with $\tau < 0.2$, while noise sensitivity affects performance with at most a factor of two for balanced and delay-dominated processes with $\tau > 0.4$ (see dashed lines in Figure 5).

Furthermore, a closer look at the IAE-optimal filtered PID controllers reveals that they are all (within numeric tolerance) equal to their IE-optimal PI counterparts for the most constrained case of $M_{ks} = 1$. This is a combined consequence of the lack of zero crossing oscillations of the IE-optimal design, and that the introduction of derivative action would violate the noise sensitivity constraint.

3.3 PID versus general LTI controllers

To assess whether PID control is a suitable choice for particular process and constraint level combinations, the relative load IAE improvements of the Q design over the filtered PID design are shown for $M_{ks} = 1$ (blue), $M_{ks} = 10$ (red), and $M_{ks} = \infty$ (green) in Figure 6.

For $M_{ks} = 1$, IAE improvement never exceeds 50% over the considered process set. For $M_{ks} = 10$ and $M_{ks} = \infty$ the maximal relative improvements are factors of 2.4 and 21, respectively.

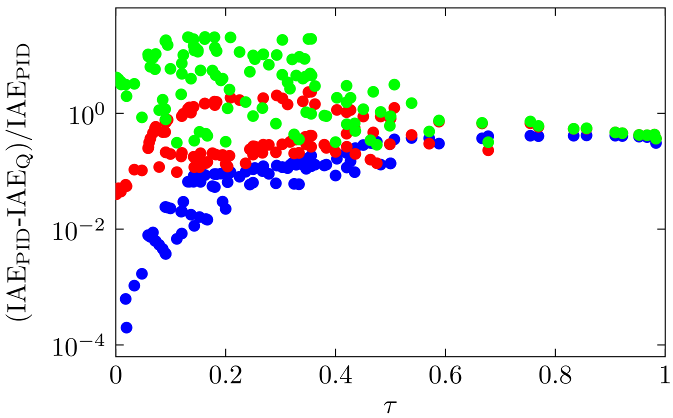

Fig. 6. Relative IAE improvement of Q over IAE design, for $M_{ks} = 1$ (blue), $M_{ks} = 10$ (red), and $M_{ks} = \infty$ (green), plotted against normalized time delay, τ, for the processes considered in Section 3.2.

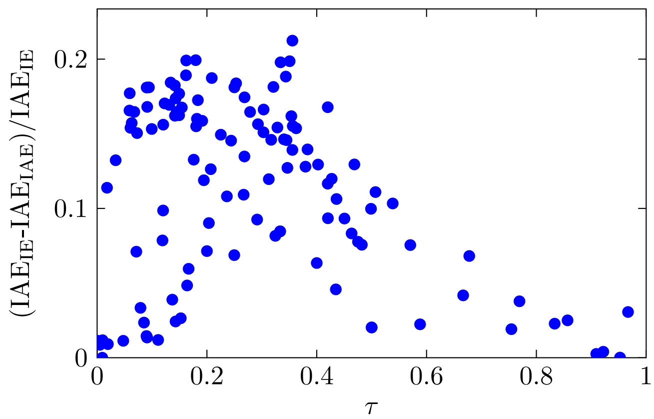

Fig. 7. Relative performance improvement, (IAE$_{\mathrm{IE}}$-IAE$_{\mathrm{IAE}}$)/IAE$_{\mathrm{IE}}$, for $M_{ks} = \infty$, plotted against normalized time delay, τ, for the processes considered in Section 3.2.

3.4 IE versus IAE minimization

This final example investigates how much performance is improved between the solution of the IE and IAE problems. Figure 7 shows the relative IAE improvement, (IAE$_{\mathrm{IE}}$-IAE$_{\mathrm{IAE}}$)/IAE$_{\mathrm{IE}}$ (subscripts denoting design), of the IAE-optimal PID controller over its IE-optimal PI counterpart, for the processes considered in Section 3.2.

The performance improvement is larger for lag-dominated processes ($\tau < 0.5$), with the largest relative improvement, 21%, occuring for the same process yielding the maximal improvement in Figure 8. Larger relative improvements can be expected when relaxing the (complementary) sensitivity constraints, enabling for more aggressive controllers with zero crossing load step response oscillations.

4. DISCUSSION

Two previously published PID design algorithms have been implemented in Matlab together with a new Q design algorithm, and made available through the PIDopt software package. It has been used in this paper to investigate achievable performance for a large set of stable process

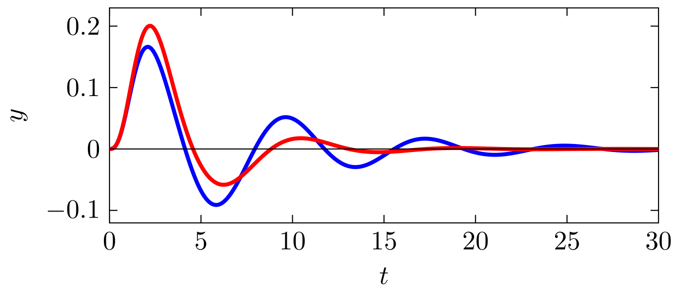

Fig. 8. Load step responses of IE-optimal PI (blue) and IAE-optimal PID (red) controllers corresponding to the maximal relative improvement in Figure 7.

dynamics. Such comparisons could serve to decide whether PID is a suitable controller type choice for a particular combination of process and robustness requirements.

It should be pointed out that both PID design algorithms rely on local optimization, and while they honor robustness constraints, they come with no guarantees of finding the global performance optima, although they have done so in cases where the global optima have been previously published, as mentioned in Soltesz et al. (2017).

The processes on which the methods have been evaluated in this paper are all representative of process industry and are known to be suitable for PID control. An expansion of this work could be to make similar comparisons using other types of dynamics for which PID controllers are typically used, such as (mechanical) systems with oscillatory modes found in robotic or automotive applications, or more exotic dynamics, such as integrator-lead processes or heat transfer dynamics.

While falling outside the scope of this paper, it should also be straightforward to use PIDopt to compare other (filtered) PID designs, obtained through for example tuning rules such as Skogestad (2003) or Åström and Hägglund (2004), with achievable LTI controller performance.

In some applications, it is desirable to limit the $\mathcal{H}_2$ (energy) rather than the $\mathcal{H}_\infty$ (worst case) norm of the noise sensitivity KS. This can be readily achieved for the IAE-optimizing design, as explained in Soltesz et al. (2017). Other natural extension of PIDopt include multi-model designs, and designs for uncertain systems.

ACKNOWLEDGEMENTS

The authors would like to acknowledge Chriss Grimholt and Martin Hast for sharing their code, on which parts of the PIDopt software package relies. We would further like to acknowledge Olof Garpinger and Olof Troeng for inspiration leading up to this work, and Karl Johan Åström for fruitful discussions on PID tuning.

REFERENCES

Apkarian, P. and Noll, D. (2006). Nonsmooth Hinf synthesis. *IEEE Transactions on Automatic Control*, 51(1), 71–86.

Åström, K.J. and Hägglund, T. (2004). Revisiting the Ziegler-Nichols step response method for PID control. *Journal of Process Control*, 14(6), 635–650.

Boyd, S., Hast, M., and Åström, K.J. (2015). MIMO PID tuning via iterated LMI restriction. *International Journal of Robust and Nonlinear Control*, 26(8), 1718–1731.

Boyd, S. and Barratt, C. (1991). *Linear Controller Design: Limits of Performance*. Prentice Hall.

De Oliveira, M.C., Geromel, J.C., and Bernussou, J. (2002). Extended H2 and HInf norm characterizations and controller parametrizations for discrete-time systems. *International Journal of Control*, 75(9), 666–679.

Garpinger, O. (2015). Optimal PI and PID parameters for a batch of benchmark process models representative for the process industry. Technical report 7645, Dept. Automatic Control, Lund University, Sweden.

Grant, M. and Boyd, S. (2014). CVX: Matlab software for disciplined convex programming, version 2.1. `http://cvxr.com/cvx`. Acessed: 2018-01-19.

Hägglund, T. (2012). A unified discussion on signal filtering in PID control. *Control Engineering Practice*, 21(8), 994–1006.

Hast, M., Åström, K.J., Bernhardsson, B., and Boyd, S.P. (2013). PID design by convex-concave optimization. In *2013 European Control Conference*, 4460–4465. Zürich, Switzerland.

Hast, M. and Hägglund, T. (2015). Optimal proportional–integral–derivative set-point weighting and tuning rules for proportional set-point weights. *IET Control Theory & Applications*, 9(15), 2266–2272.

Kristiansson, B. and Lennartson, B. (2002). Robust and optimal tuning of PI and PID controllers. *IEEE Proceedings of Control Theory and Application*, 149(1), 17–25.

MOSEK ApS (2015). *The MOSEK optimization toolbox for MATLAB manual. Version 7.1 (Revision 28).* URL `http://docs.mosek.com/7.1/toolbox/index.html`. Accessed: 2018-01-10.

Panagopoulos, H., Åström, K.J., and Hägglund, T. (2002). Design of PID controllers based on constrained optimisation. *IEEE proceedings of Control Theory and Applications*, 149(1), 32–40.

Sadeghpour, M., de Oliveira, V., and Karimi, A. (2012). A toolbox for robust PID controller tuning using convex optimization. In *Proceedings of IFAC Conference in Advances in PID Control*. Brescia, Italy.

Sekara, T.B. and Matausek, M.R. (2009). Optimization of PID controller based on maximization of the proportional gain under constraints on robustness and sensitivity to measurement noise. *IEEE Transactions on Automatic Control*, 54(1), 184–189.

Skogestad, S. (2003). Simple analytic rules for model reduction and PID controller tuning. *Journal of Process Control*, 13(4), 291–309.

Soltesz, K., Grimholt, C., and Skogestad, S. (2017). Simultaneous design of PID controller and measurement filter by optimization. *IET Control Theory & Applications*, 11(3), 348–348.

Yuille, A.L. and Rangarajan, A. (2003). The convex-concave procedure. *Neural Computation*, 15(4), 915–936.