Preprints, 4th IFAC Conference on
Advances in Proportional-Integral-Derivative Control
Almería, Spain | June 12-14, 2024

A MATLAB virtual laboratory to support learning of auto-tuning PID approaches

J.A. Rossiter * **A. Visioli** ** **S. Dormido** *** **R. Bars** ****

* *Department of Automatic Control and Systems Engineering, University of Sheffield, Sheffield, S1 3JD, UK, e-mail: j.a.rossiter@sheffield.ac.uk*
** *Dipartimento di Ingegneria Meccanica e Industiale, University of Brescia, Italy, e-mail: antonio.visioli@unibs.it*
*** *Universidad Nacional de Educacion a Distancia, Madrid, Spain, e-mail:s.dormido@dia.uned.es*
**** *Department of Automation and Applied Informatics, Budapest University of Technology and Economics, Hungary, email: Bars.Ruth@aut.bme.hu*

Abstract: This paper presents a small number of MATLAB APPs and livescript files designed to help students both understand and implement PID tuning. The paper presents the thinking behind the use of MATLAB and the topic itself before then describing the proposed resources in detail. The resources split into files with detailed mathematical and coding background students can use for self-study and assignments, and a virtual laboratory which is more intuitive and interactive and useful for familiarisation with core concepts. The files were recently added to the control101 toolbox (Rossiter, 2023).

Keywords: Control101 toolbox, PID compensation, virtual laboratories, independent learning.

1. INTRODUCTION

The potential of the internet to facilitate sharing of control resources (Rossiter et al., 2018; Rossiter, 2022; Serbezov et al., 2022) is now well understood and thus, as a community, researchers working in control are asking how they can support each other with high quality open-access resources? Already one excellent example exists (Douglas, 2022), but this site acts more as pointer to resources produced elsewhere rather than the host itself, and there is a need for community members to produce the resources themselves.

Within the specific context of a first course in control (Rossiter et al., 2020), one topic above all others was recognised as top priority due to is prevalence, that is PID controllers. In consequence it is logical for the community to consider what open-access resources are available to support students in familiarisation with both the concepts of PID compensation and also the tuning (Svrcek et al., 2000; Astrom et al., 2006, 2004; Skogestad, 2003; Hagglund, 2019; Buchi, 2022; Rivera et al., 1986; Rojas et al., 2021; Vilanova et al., 2014). The aim here is to supplement existing resources with something novel, accessible and easy to use.

One are of particular interest of late is that of virtual laboratories (Matisak et al., 2023; Cameron, 2009; de la Torre et al., 2013, 2020; Rossiter, 2017). Increasingly computer power and the internet is making the development and distribution of virtual laboratories a straightforward task. Of particular note is the significant effort of the Spanish community (Guzman et al., 2018, 2023) in this venture and their excellent suite of open access resources. They have focused significantly on virtual laboratories for a 1st course in control which provide an interactive interface whereby users can easily and quickly experiment with different choices (parameters, algorithms, etc.) and see the impact almost instantly.

1.1 The use of MATLAB

However, despite the existence of these high quality resources, the first author has taken a slightly different path. In many universities students are exposed to MATLAB throughout their programmes as a software of choice for supporting computation and plotting, across a wide range of topics. Consequently, for consistency and convenience, it is helpful to students if other resources are presented in the same environment as this allows a seamless transition. Moreover, MATLAB provides a coding environment which gives users the flexibility to edit and modify what has been provided to personalise for their own needs, for example in projects and assignments. MATLAB gives the facility for multiple different file types, but in this paper we focus on just two.

(1) Livescript files: These files (Rossiter, 2023), by default come in a two column setting where the left hand column contains explanation in a moderately neat format and MATLAB code. The right hand column contains the results of the code, that is, displays of values, text and figures as desired. The combination of neat notes with code allows a single resource to both explain a topic and illustrate the associated numerical computations and figures and moreover,

the code can be edited by users to allow entry of their own examples and parameters. The code itself serves as an exemplar of how to use MATLAB and this can be copied and modified and extended by students for different scenarios and their projects.

(2) Virtual laboratories or *apps*: MATLAB also has a ready made interactive environment (Rossiter, 2022), in essence to support virtual laboratories. The interface allows users to select buttons, sliders, drop down menus and so forth and the results are presented in figures and text as appropriate. The author generates code which is in the background and hidden from the student and thus these are much less flexible than livescripts, but conversely, they are more intuitive and allow the author to create an environment that focuses on communicating core concepts.

A holistic learning resource may contain both virtual laboratories and livescripts: the virtual laboratories introduce the concepts and allow users to learn by trial and error and also to investigate different concepts in a convenient and simple way. The livescripts provide more technical depth and background and encourage the user to move on to deeper understanding, design and synthesis.

1.2 The "control101" toolbox

Given the arguments above, the first author has recently proposed that there would be significant benefit to the global community in having a MATLAB toolbox focused on a 1st course in control (Rossiter, 2023, 2021) where the priority includes helping students focus on core concepts and visualisation, before getting into detailed coding. A first draft of the toolbox was released at the IFAC world congress in 2023 (Mathworks File Exchange, 2024); users can download this for free using the add-ons option within Matlab. However, the first release was very much that and the hope is that more international colleagues will want to contribute resources to this toolbox so it becomes a shared enterprise which contains all the relevant virtual laboratories and more.

The initial release focused on system behaviours and the potential of simple PI feedback to modify those behaviours and reject uncertainty. Numerous virtual laboratories introduced concepts using animations on common examples such as tank level, house temperature and car velocity. Since that point, more effort is now being focused onto topics that would more likely come towards the end of a 1st course, one of those being systematic tuning of a PID compensator. Hence the purpose of this paper is to introduce the initial resources on PID compensator design and to invite colleagues to share these with their students and of course, offer feedback on improvements and other contributions that would be beneficial.

It is also worth noting that recently Mathworks released the PIDTuner framework. However, the files developed for the control101 toolbox give more detailed explanations and also cover other approaches.

1.3 Paper organisation and contribution

Many effective compensator design methods are based on frequency domain considerations, and indeed these are discussed in detail elsewhere in the toolbox. Nevertheless, in practice it is useful to provide compensator parameter settings based on some simple measurements providing the model of the plant, generally approximating it by a first order lag with dead time. Hence, as stated above, the core contribution of this paper is to summarise the new time domain toolbox functionality on PID compensator design. Section 2 will give a concise summary of the design methods to be deployed. Section 3 introduces, very briefly, the earlier resources which focus more on illustration than design. Then section 4 will describe the new resources and how to use them. The paper finishes with conclusions.

2. BACKGROUND ON PID COMPENSATION

This section gives a brief background on PID compensation and some typical tuning laws that one might want to include in a taught course. The list is of course not comprehensive and cannot reasonably cover all possible alternatives, although we re-emphasise that the authors would like the toolbox to be co-owned by the community and are happy to receive recommendations on how to embellish this further. The focus here is on relatively simple tuning methods which work well on relatively standard model forms, but of course this means that they may be less effective where more nuanced tuning is required.

2.1 Definition of a PID compensator

For simplicity this work focuses only on parallel forms of PID compensation, where $M(s)$, the transfer function of the controller is given as:

$$M(s) = K_p + \frac{K_i}{s} + sK_d; \quad K_i = \frac{K_p}{T_i}; \quad K_d = K_p T_d \quad (1)$$

It is known that in general a pure derivative is not implementable and needs to be modified. For simplicity the toolbox uses the following approximation, where T_D is linked to the derivative time constant (typically about ten times faster).

$$sK_d \to K_d \frac{s}{T_D s + 1} \quad (2)$$

Further discussion on the derivative term is omitted from the introductory resources.

2.2 Characterisation of a system

Many popular auto-tuning methods are based on a characterisation of the open-loop system using the open-loop step response. A typical example is given in Figure 1 where the intention is to estimate: i) effective delay, here denoted as L; ii) effective time constant, here denoted as T and iii) steady-state gain, here denoted as K. Clearly the selection of L, T, K is not entirely objective and precise and small changes in the choices will impact on the resulting PID design.

Remark 1. Some tuning methods also require the user to specify a desired closed-loop time constant. For simplicity the app currently takes this to match the open-loop value of T, but the authors are open to discussion on this point.

2.3 Popular PID tuning methods

It was decided that a tool that was to be of use to students learning about PID tuning should include a variety of

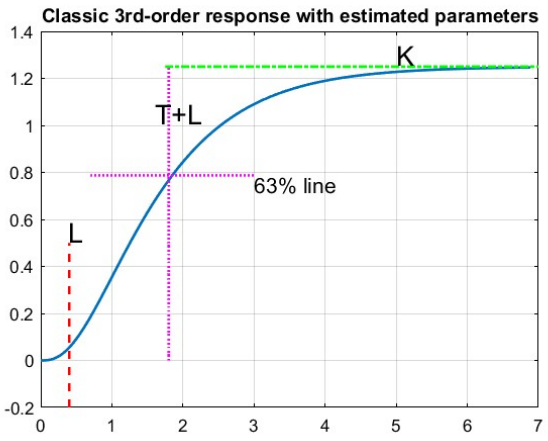

Fig. 1. Characterisation of the open-loop system.

different tuning methods, but for simplicity we did not include the Ziegler-Nichols critical gain method as this does not use Figure 1 and thus would be more cumbersome to include in a compact resource. The methods and the tuning rules are summarised below for completeness as these capture a good range of the behaviours and available thinking in the literature.

Ziegler Nichols reaction curve rules (Astrom et al., 2006)

- Proportional only:
$$K_p = \frac{T}{KL}$$
- Proportional + integral (PI):
$$K_p = \frac{0.9T}{KL}, \quad K_i = \frac{K_p}{3.33L}$$
- PID:
$$K_p = \frac{1.2T}{KL}, \quad K_i = \frac{K_p}{2L}, \quad K_d = \frac{K_p}{2L}$$

Cohen-Coon Method (Cohen et al., 1953)

- Proportional only:
$$K_p = \frac{1.03T}{K(L+0.34)}$$
- Proportional + integral (PI):
$$K_p = \frac{0.9T}{K(L+0.092)}, \quad T_i = \frac{3.33L(T+0.092L)}{(T+2.22L)}$$
- PID:
$$K_p = \frac{1.24T}{K(L+0.129)}, \quad T_i = \frac{2.5L(T+0.185L)}{(T+0.611L)},$$
$$K_d = \frac{0.37K_pLT}{(T+0.185L)}$$

Chien-Hrones-Reswick Method (Svrcek et al., 2000)

- Proportional only:
$$K_p = \frac{0.3T}{LK}$$
- Proportional + integral (PI):
$$K_p = \frac{0.35T}{LK}, \quad T_i = 1.2T$$
- PID:
$$K_p = \frac{0.6T}{LK}, \quad T_i = T, \quad K_d = 0.5LK_p$$

AMIGO tuning rules (Astrom et al., 2004)

- Proportional + integral (PI):
$$K_p = \frac{0.15}{K} + \frac{(0.35 - \frac{LT}{(L+T)^2})T}{KL},$$
$$T_i = 0.35L + \frac{(6.7LT^2)}{(T^2+12TL+7L^2)}$$
- PID:
$$K_p = \frac{1}{K(0.2+0.45T/L)}, \quad T_i = \frac{0.4L+0.8T}{(L+0.1T)L},$$
$$T_d = \frac{0.5LT}{(0.3L+T)}$$

One-third tuning rule (Hagglund, 2019)

- Proportional + integral (PI):
$$K_p = \frac{1}{3K}, \quad T_i = \frac{L}{3} + T$$

Lambda tuning (Smuts, 2011; Veronesi et al., 2020) Define λ as the desired closed-loop time constant.

- Proportional + integral (PI):
$$K_p = \frac{T}{K(\lambda+L)}, \quad T_i = T$$

Skogestad method (Skogestad, 2003)

- Proportional + integral (PI):
$$K_p = \frac{T}{2KL}, \quad T_i = \min(T, 8L)$$

3. EARLY FILES ON BEHAVIOUR AND PI FEEDBACK

The first release of the toolbox in the summer of 2023 contained what might be called illustrations of feedback and opportunities for students to learn through trial and error. In essence the focus here was mostly analysis and some evaluation, but not systematic design. This paper will not discuss those files in detail but rather just give a quick summary.

Several case study driven app files: *cruisecontrol, house-temperature_story, tanklevel_and_PIcontrol, tanklevel_story, pi_tuning* allow users to explore the impact of different PI tuning choices on closed-loop behaviour, both tracking and disturbance rejection. However, there is no explicit guidance on how to tune systematically. The *pi_tuning* app is the most generic of these (see Figure 2) as it allows a wide variety of systems to be explored whereas the case studies apps (Rossiter, 2023) are specific solely to the given scenario and implied model.

4. THE LIVESCRIPT AND APP FILES ON PID COMPENSATION

Clearly, at some point we want to move students on from analysis and evaluation of control loops to systematic

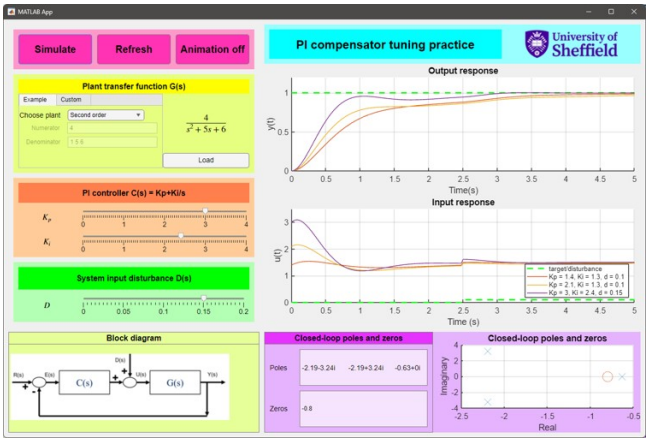

Fig. 2. Interface for the PI tuning app.

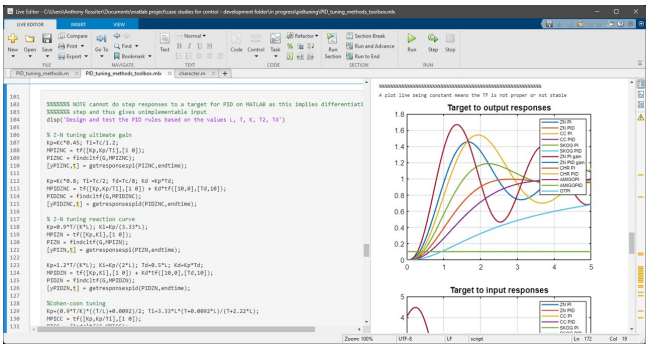

Fig. 3. Screen shot from within the livescript file showing PID definitions and some closed-loop responses.

design. When it comes to PID, a logical first step is to look at the many *tuning rules* in the literature which give guidance on how PID parameters can be chosen in a systematic fashion when a reasonable model takes a relatively simple and low order form. The current toolbox files focus on the methods summarised in section 2.

Remark 2. Users will note that when more complicated models are considered, such as those with sigificant underdamping, the rules are less effective in designing a good PID compensator. A few challenging examples are included in the app so users can observe these failures.

4.1 The livescript file

The livescript file *pid_tuning_methods.mlx* (Figure 3) contains the detailed background on the system characterisation and MATLAB coding for the PID compensators so that users can easily modify for their own needs. It also gives some of the background information on how the different tuning methods are defined (similar to section 2). This resource will be useful to those wishing to develop their own code and examples and to gain a deeper understanding of the methods.

4.2 The virtual laboratory overview

The virtual laboratory hides the fine details, algebra and coding and allows users to focus on concepts and evaluation; how well do these methods work and is the characterisation of section 2.2 effective for underpinning

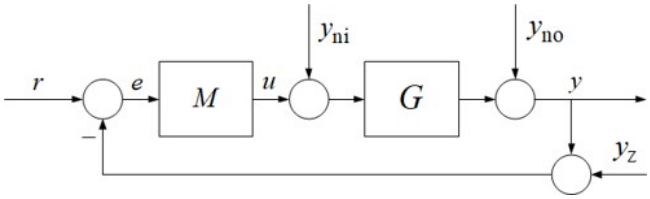

Fig. 4. Typical closed-loop diagram.

the PID tuning? Consequently the app is split into two main tasks.

(1) Task 1 focuses on characterisation.
(2) Task 2 focuses on illustration the closed-loop behaviour.

Task 2 also considers the important facet of disturbance rejection, so taking a closed-loop diagram as given in Figure 4 we are interested not only in tracking of targets, but also the response to input and output disturbances and measurement noise.

Remark 3. For convenience the model options entered all have a steady-state gain close to unity and time constants in a similar range so that the definitions of the sliders have sufficient precision to cover the whole range of examples.

4.3 Characterising the system in the virtual laboratory

When the app is first opened the system characterisation plot is shown, as in Figure 5; this focuses on the characterisation discussed in section 2.2. The user can always return to the characterisation part of the design using the top left yellow and orange boxes and top left pink box.

(1) Users select the system from the drop down options and add a system delay if desired. The current choice is displayed in the text boxes in the bottom middle of the window. Whenever a system is changed, the characterisation plot will appear.
(2) Users select the values for L, T, K using the sliders in the orange box. Their choices appear in the figure so they can fine tune as desired. It is logical to choose K first, then L and finally T. Whenever one of these sliders is changed, the corresponding characterisation plot will appear.
(3) As seen in Figure 5, the corresponding auto-tuned P, PI and PID parameters are displayed in the text boxes below the figure, corresponding to whichever tuning rule is currently, or most recently, active in the green box.
(4) One can return to this step at any point using the pink button 'Characterise system' or by making changes to the options in the yellow and orange boxes or by selecting the appropriate tab in the figure window.

4.4 Closed-loop responses in the virtual laboratory

The pink 'create responses' button allows the user to view the various closed-loop behaviours as seen in Figure 6 corresponding to the various loop inputs r, y_{no}, y_{ni}, y_z seen in Figure 4. The default is that the behaviours shown match the selections in the green drop down menus. If the user selects 'manual choice' in the green drop down, the parameters are taken from the sliders in the bottom left.

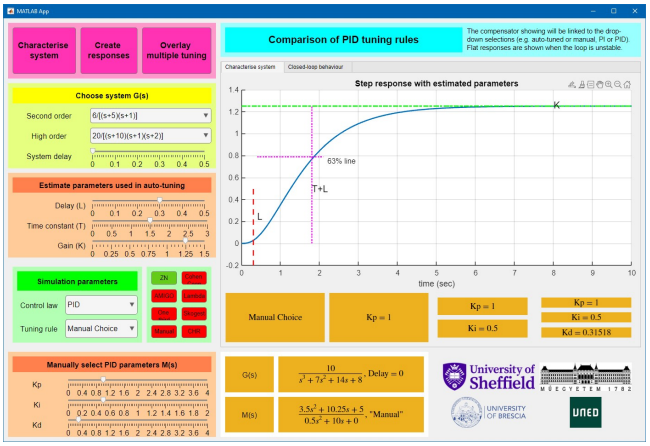

Fig. 5. Open-loop characterisation and corresponding PID parameters.

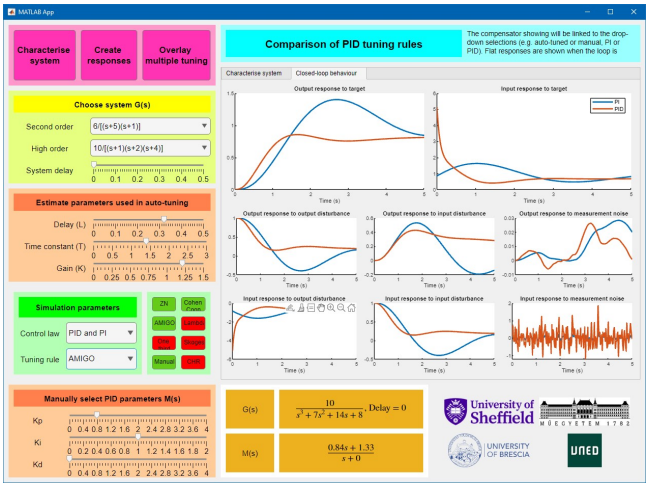

Fig. 6. Overlaying selected closed-loop responses.

(1) Users can select a specific law such as PI or PID and a specific tuning method (Figure 6).
(2) There are also drop down options to compare all the different tuning methods (Figure 7). The small buttons in the green box allow the user to specify which of the methods are overlaid when 'All PI' or 'All PID' is selected.
(3) Selecting the appropriate tab in the figure window will also show the responses and these also update when changes are made to the options in the green drop downs.

Remark 4. It is easy to check (on MATLAB) for stability with a zero dead-time, so when the dead-time is zero, a check is undertaken and divergent responses are replaced by a flat line so as not to detract from the other behaviours. When the dead-time is not zero, this is less straightforward and so the vertical axes adopt default limits.

4.5 Overlaying options to support systematic tuning

It may be convenient or useful to compare and contrast specific choices of control law or characterisations, as illustrated in Figure 8 for example:

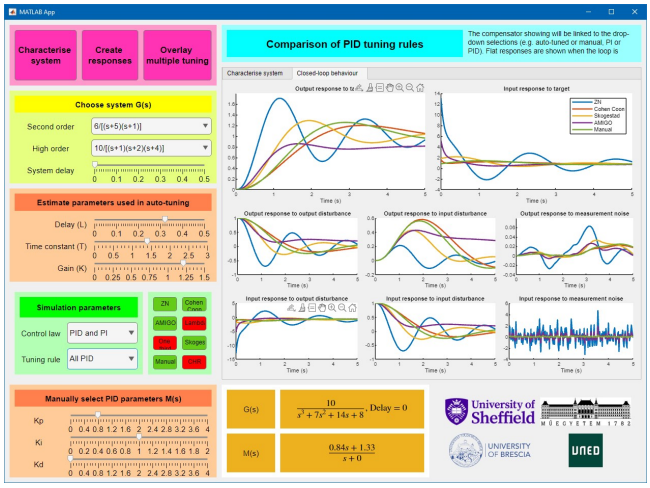

Fig. 7. Overlaying the responses from all the different tuning methods.

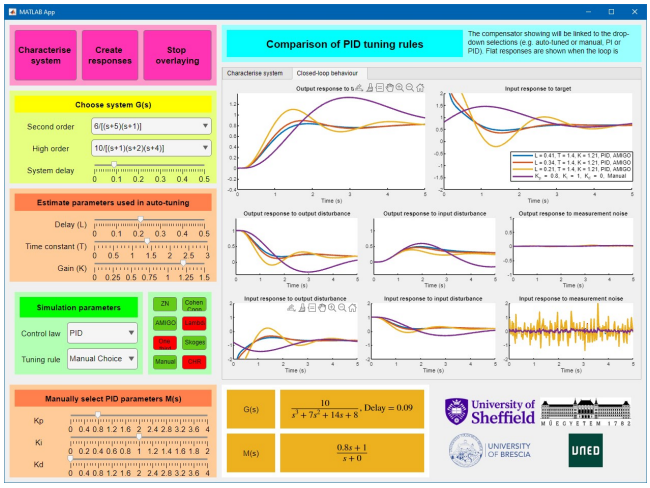

Fig. 8. Overlaying the responses from specified options only.

(1) How does changing the parameter choices for K, L, T effect the auto-tuned compensator and thus closed-loop behaviour?
(2) Can I compare just 2 or 3 tuning methods?
(3) Can I compare PI and PID for 2 different methods simultaneously?

To achieve this select the pink button 'overlay multiple tuning' and now each option will be added to the responses window so you can build up the desired comparison. As seen in Figure 8, the legend keeps track of the options used. A new plot is added when the user selects the create responses button. Press 'stop overlaying' to clear the figure and begin again.

5. CONCLUSIONS

This paper has introduced a new set of tools designed for the control101 MATLAB toolbox which focus on PID design. PID is an important part of introductory control modules, but students may not have time to study design in detail and thus resort to tuning methods. These tools allow students to explore some common tuning methods quickly and easily and perform compare and contrast

exercises. The decision has been made to use MATLAB as that is very commonly used elsewhere in the curriculum and therefore allows relatively seamless integration as the toolbox can be downloaded in about a minute.

A decision has been taken to limit the case studies and options available so as not to over face users with too much detail, but nevertheless the authors are interested in feedback and proposals from the community, for example:

(1) Do the files include the most important tuning methods and if not, what should be added?
(2) Are there other core aspects of PID familiarisation which should be added to the toolbox, perhaps through additional files?
(3) Are there other core topics missing from the toolbox and would colleagues like to collaborate on writing these given the toolbox co-ordinator has limited time and expertise.

It should also be noted that the files are continually in development so what is available at the time of reading in the future may be slightly improved compared to what has been described here.

This paper does not discuss fine details of how the interfaces are designed and coded, but any reader who is interested in developing similar files is welcome to talk to the authors and also is reminded that the source code is all available (Rossiter, 2023) so can be used as templates if desired.

REFERENCES

Astrom, K. J., Hagglund, T., 2006, Advanced PID Control, ISA, The Instrumentation, Systems and Automation Society. ISBN: 978155617-9426

Aström, K.J., and Hagglund, T., 2004, Revisiting the Ziegler-Nichols step response method for PID control, J. of Process Control 14, 635-650.

Büchi, R., 2022, Parameter tables for PID controllers for time delayed systems optimized with a learning method, ASIM Proceedings.

Cameron, I., 2009, Pedagogy and immersive environments in the curriculum, Blended Learning conference, 290-294.

Cohen, G.H. and Coon, G.A., 1953, Theoretical consideration of retarded control. Transactions of ASAME, 75, 827–834.

de la Torre, L., R. Heradio, C. A. Jara, J. Sanchez, S. Dormido, F. Torres, and F. Candelas, 2013, Providing Collaborative Support to Virtual and Remote Laboratories. IEEE Transactions on Learning Technologies.

de la Torre, L., Neustock, L.T., Herring, G.K., Chacon, J., Clemente, F.J.G., and Hesselink, L., 2020, IEEE Transactions on Industrial Informatics, Automatic Generation and Easy Deployment of Digitized Laboratories, 12, 16, 7328-7337, doi = 10.1109/TII.2020.2977113

Douglas, B., 2022, Resourcium, https://resourcium.org/

Mathworks File exchange, https://uk.mathworks.com /matlabcentral/fileexchange/130439-control-101-toolbox

Guzmán, J.L., Piguet, Y., Dormido, S., Berenguel, M., Costa-Castelló, R., 2018, New Interactive Books for Control Education, IFAC papers online, Volume 51, Issue 4, 2018, Pages 190-195.

Guzmán, J.L., Costa-Castelló, R., Berenguel, M., Dormido, S., 2023, Automatic Control with Interactive Tools, Springer, DOI: 10.1007/978-3-031-09920-5

Hagglund, T., 2019, The one-third rule for PI controller tuning, Computers and Chemical Engineering 127, 25-30.

Matisak, J., Pohancenik, M. and Zakova, K., 2023, Platform for Virtual Laboratory of Mechatronic Systems in Augmented Reality, EXPAT 2023 and IEEE Explore

Rivera, D.E., M. Morari, and S. Skogestad, 1986, Internal Model Control 4. PID Controller Design, Industrial Engineering and Chemical Process Design and Development, 25.

Rojas, J.A., Arrieta, O. and Vilanova, R., 2021, Industrial PID Controller Tuning, Springer.

Rossiter, J.A., 2017, Using interactive tools to create an enthusiasm for control in aerospace and chemical engineers, IFAC world congress (ifacpapersonline).

Rossiter, J.A., B. Pasik-Duncan, S. Dormido, L. Vlacic, B. Jones, and R. Murray, 2018, Good Practice in Control Education, European Journal of Engineering Education, http://dx.doi.org/10.1080/03043797.2018.1428530.

Rossiter, J.A., Serbezov, A., Visioli, A., Zakova, K. and Huba, M., 2020, A survey of international views on a first course in systems and control for engineering undergraduates, IFAC Journal of Systems and Control, Vol. 13, Article 100092, 15 pages. https://doi.org/10.1016/j.ifacsc.2020.100092

Rossiter, J.A., 2021, Modelling, dynamics and control website, http://controleducation.group.shef.ac.uk/mainindex.html

Rossiter, J.A., 2022, MATLAB apps to support the learning and understanding of simple system dynamics, IFAC Symposium on Advances in Control Education (ACE 2022).

Rossiter, J.A., Control 101 toolbox, https://controleducation.sites.sheffield.ac.uk/matlabresources/community-control-toolbox or https://github.com/jarossiter/control101

Rossiter, J.A., 2023, A suite of MATLAB livescript files to support learning of elementary control and feedback concepts, IFAC world congress.

Skogestad, S., 2003, Simple analytic rules for model reduction and PID controller tuning, J. of Process Control 13, 635-650.

Serbezov, A., Zakova, K., Visioli, A., Rossiter, J.A., Douglas, B. and Hedengren, J., 2022, Open access resources to support the first course in feedback, dynamics and control, IFAC Symposium on Advances in Control Education.

Smuts, J., 2011, Process Control for Practitioners: How to tune PID controllers and optimize control loops, OptiControls.

Svrcek, W Y., Mahoney, Donald P., Young, Brent R. A Real Time Approach to Process Control, 2nd Edition. John Wiley and Sons, Ltd. Print ISBN:9780470025338 —Online ISBN:9780470029558 —DOI:10.1002/9780470029558

Veronesi, M. and A. Visioli, 2020, On the Selection of Lambda in Lambda Tuning for PI(D) Controllers, IFAC-PapersOnLine, 53, 2, Pages 4599-4604

Vilanova, R. and Visioli, A., 2014, PID Control in the Third Millennium, Springer.