Proceedings Paper:
Rossiter, J.A. (2023) A suite of MATLAB livescript files to support learning of elementary control and feedback concepts. In: Ishii, H., Ebihara, Y., Imura, J. and Yamakita, M., (eds.) IFAC-PapersOnLine. 22nd World Congress of the International Federation of Automatic Control (IFAC2023), 09-14 Jul 2023, Yokohama, Japan. Elsevier, pp. 7555-7560. ISSN: 2405-8963. EISSN: 2405-8963.

https://doi.org/10.1016/j.ifacol.2023.10.657

Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 56-2 (2023) 7555–7560

A suite of MATLAB livescript files to support learning of elementary control and feedback concepts

J. A. Rossiter *

* *Department of Automatic Control and Systems Engineering, University of Sheffield, Sheffield, S1 3JD, UK (e-mail:j.a.rossiter@sheffield.ac.uk*

Abstract: This paper builds on a body of work in the community which is focussed on sharing learning and teaching resources, especially those which might support a first course in control. Here attention is given to some of the mathematical, analytical and numerical computations which are required to support simple system and feedback analysis and design. The aim is to provide resources which allow students to focus on core concepts and understanding so that the numerical computations are not an obstacle to their investigations. More specifically, this paper focuses on a number of MATLAB livescript files which have been produced to help students visualise the impact of parameter and design choices on system behaviour, while simultaneously empowering them to understand the source code and thus upskill them for the future. The paper gives an overview of the livescripts available so users can decide whether these could be useful in their own context; all are freely available on the author's website (Rossiter, 2021).

Keywords: Virtual laboratories, independent learning, visualisation, livescripts.

1. INTRODUCTION

The IFAC technical committee on control education, and indeed its partner IEEE TC, have focussed a lot of effort in recent years on defining an ideal first curriculum in control (Rossiter et al., 2020). Subsequently, they are focussing their efforts on control education resources, namely how do we as a community support each other with open access resources suitable for courses in control (Douglas, 2022; Rossiter, 2021; Albertos, 2017; Quanser, 2022; Serbezov et al., 2022).

This paper focuses more on the laboratory aspects of course delivery. Nevertheless, although access to real hardware is an important part of any course and there are many ideas for making these more accessible (Abdulwahed, 2010; Panza et al., 2021; Rossiter et al., 2019; Zapata-Rivera et al., 2019), this paper is more focussed on particular aspects that would fit into the theme of virtual laboratories. Virtual laboratories are important because they can provide close to authentic interfaces and visualisations alongside important attributes such as:

(1) 24/7 access
(2) Parallel or independent and simultaneous access by an entire cohort, no matter the class size.
(3) Fast run times (essentially instantaneous) hence allowing rapid testing and design and thus learning.

Consequently, there has been a substantial amount of work in the community both creating and sharing virtual laboratories, most notably from the Spanish community (de la Torre et al., 2013, 2020) who have focussed on a particular choice of software and approach. Multiple other approaches have also been used worldwide, e.g. (Cameron, 2009; Goodwin et al., 2011; Rossiter, 2017; Quanser, 2022; Rossiter, 2016).

This paper focuses on the use of the MATLAB software package. While this is controversial with some due to the expense, it is very common for universities to have a site license so that students have free access and indeed, student licenses are relatively cheap where needed. Modern innovations also mean that, through a University license, students can access the software through MATLAB online where they own computer is not suitable. For the author, the excellent functionality and ready access means that there are several core advantages:

(1) It is quick and easy for him to create useful resources for the students to use (Rossiter, 2016; Serbezov et al., 2022).
(2) The environment is largely intuitive and due to the large number of powerful built-in functions, students can use this for problem solving (Lynch and Becerra, 2011), visualisation and learning with relatively little training.
(3) Tools such as MATLAB online and MATLAB web servers mean that students do not necessarily need MATLAB on their own computer and may be able to run these resources via a web page.

When building a virtual laboratory, the educator will be focussing on visualisation of some important engineering concepts so that students can relate what they are learning in class to real applications. The virtual laboratory should of course also allow suitable interactions and design decisions so that students can investigate the impact of changes. However, there is a limit to the efficacy of simple GUI interfaces (de la Torre et al., 2020; Rossiter, 2016,

10.1016/j.ifacol.2023.10.657

2022) as students can only make the changes that are coded, only create the figures that are provided and so forth. As students mature in their engineering understanding, they want more control of the design and implementation, that is, they may want access to source code and to make changes directly in there. As a consequence, we need to prepare students for this by providing them with a proper foundation in programming: in the context of this paper that programming is in MATLAB.

This paper explores the relatively recent MATLAB innovation of *livescript* files (Antunes, 2021; Nevaranta et al., 2019). The thinking behind these is to ease the student exposure to coding and how this is related to the underlying engineering or tasks at hand. This paper introduces a suite of livescript files (accessible from section 6.9 of the website (Rossiter, 2021)) which students can use to learn and apply skills relevant to a first course in control and also some aspects of classical control design. Section 2 summarises what livescript files are and how to code them. Section 3 details a suite of livescripts which are tailored to run alongside a first course in control, or if you like weekly tutorial files. Conversely, section 4 introduces a suite of files which are designed around topics and themes. The paper finishes with some conclusions.

2. AN INTRODUCTION TO MATLAB LIVESCRIPTS

This section illustrates the concept of a MATLAB livescript and thus how they can be useful for student learning. In simple terms, a livescript combines a classical notes format (for example PDFs supplied by the lecturer) with the code and figures produced by MATLAB into a single integrated environment. By doing so, students can see a well presented engineering argument and development alongside suitable code to support computation and solving. This means the association between suitable code and mathematics is more evident.

A second core advantage of the livescript is that that students can change the coding parts in real time and immediately re-run them to update computations and figures, thus gaining an appreciation of the links between coding syntax and mathematical problem solving, as well as the advantage of rapid number crunching and visualisation. The environment for supplying notes and equations works a little like *word*, but with limited functionality. Images can also be added alongside the textual explanation if desirable. Code is embedded in the same document with a different line, font and light gray background so clearly distinguished.

Figure 1 gives an illustration of this for the simple context of solving a 1st order linear ODE. Readers will notice how the left hand window includes a neat presentation of the mathematical context and areas with a light gray background are the associated code. The solution appears in the right hand window (all windows are scalable as appropriate for the context).

One can easily add figures, data displays and other facets to the right hand display as required, as illustrated in figure 2. It will be evident in this second example that the short code snippet of 6 lines (lines 16-21) illlustrates the following MATLAB syntax to the students:

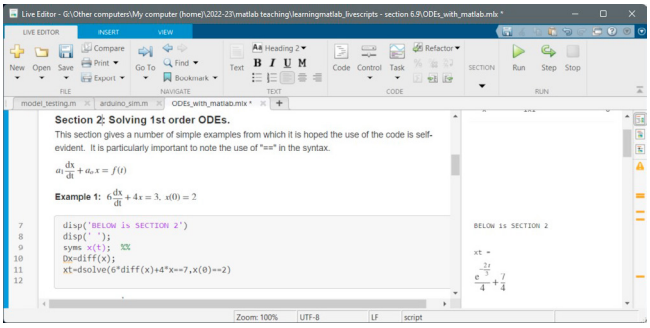

Fig. 1. Illustration of a livescript interface.

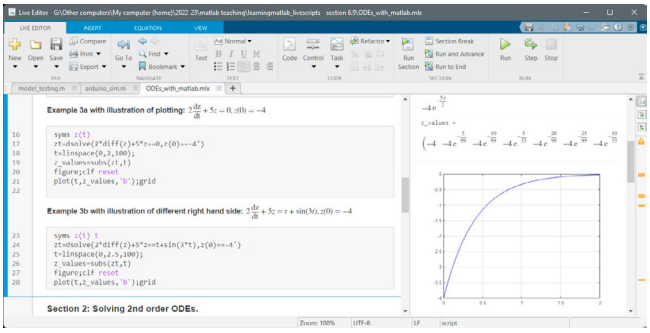

Fig. 2. Illustration of a livescript interface inlcuding a figure.

- Use of symbolic variables.
- Correct use of *dsolve.m* to solve an ODE.
- Conversion of algebraic variables to numbers.
- Plotting (this file assumes students already know how to do this).

The first two of these were covered earlier in the livescript (e.g. figure 1) so in this case the code complexity is increased by adding the numerical computations and plotting. It should be relatively transparent to the user how they can change parameters of their choice such as: (i) the parameters of the ODE and (ii) the time scale for the plot and (iii) the number of numerical values computed and (iv) other options as appropriate.

In summary, the reader should be able to see the potential of the livescript to combine notes and explanation alongside code in a well integrated fashion. This format has the potential to improve student learning of engineering concepts where MATLAB computations and visualisation are helpful and, as a useful aside, to learn how to code in parallel.

3. MATLAB VISUALISATION AND A 1ST COURSE IN CONTROL

A classical way for students to learn control and concepts is through laborious paper and pen computations on simple examples. This is fine to some extent as it helps students develop confidence and important mathematical awareness and dexterity, but conversely it can also be painfully slow for students to appreciate the significance and relevance and the course feels like they are doing more mathematics. Hence, one advantage of suitable software tools is their ability to do, instantaneously, the tedious computations such as computing $f(t)$, $0 \leq t \leq t_{end}$ where, by hand

this would take a long time. Moreover, software can allow rapid visualisation of solutions and steps through analytical solutions, graphs and even animations; such visualisation allows students to engage with concepts and explore questions such as: *what if I change parameter X?*

3.1 Context

The particular aim here is to support learning of a first course in control. A template course is available on the author's website (Rossiter, 2021) and this is divided into roughly 10 segments (denoted as weeks/sections for simplicity). Each week provides some video resources and PDF notes which summarise the core content. Then, in the rightmost columns are suggested learning activities (interactive resources, tutorials, etc.). Part of the interactive resources are a number of livescript files to be used in parallel with tutorial questions and to reinforce learning of core concepts.

3.2 Summary of 1st course tutorial files

For simplicity, the livescripts are organised on a weekly schedule to gradually build student confidence in the basic tools that will be beneficial, thus the livescripts are organised into a weekly set of skills which synchronise up to the notes and other resources on the course website and indeed users will note that some of the sections in the livescipt files specifically relate to tutorial questions in the general activities for the week. The files are summarised briefly in the following list:

- *tutorialweek1.m, tutorialweek1_livescript.mlx*: Elementary matlab notation and simple plotting - see figure 3.
- *tutorialweek2.m, tutorialweek2_livescript.mlx*: Producing nice plots and consolidation of notation.
- *tutorialweek3.m, tutorialweek3_livescript.mlx*: 1st order ODE models in MATLAB and introduction to solutions - see figure 4.
- *tutorialweek4_livescript.mlx*: Solution of 1st order ODEs and plotting.
- *tutorialweek5_livescript.mlx*: 2nd and higher order ODE models in MATLAB.
- *tutorialweek6_livescript.mlx*: Solution of 2nd/higher order ODEs and plotting.
- *tutorialweek7_livescript.mlx*: Laplace transforms and inverse Laplace with MATLAB - see figure 5.
- *tutorialweek8_livescript.mlx*: Transfer functions and characterisation of behaviours.
- *tutorialweek9.m, tutorialweek9_livescript.mlx*: Closed-loop transfer functions and step response plotting - see figure 6.
- *tutorialweek10_livescript.mlx*: Analysis of closed-loop systems using performance criteria.

It is implicit that the weekly livescripts will introduce students to useful functions such as *tf.m, step.m, pzmap.m, feedback.m* and so forth which are core for fundamental control loop analysis. However, the emphasis is on simple rather than advanced usage, that is, enough for students to make progress on the module; in essence the MATLAB tool is a time saver as it does the tedious numerical computations and produces nice plots quickly. This will

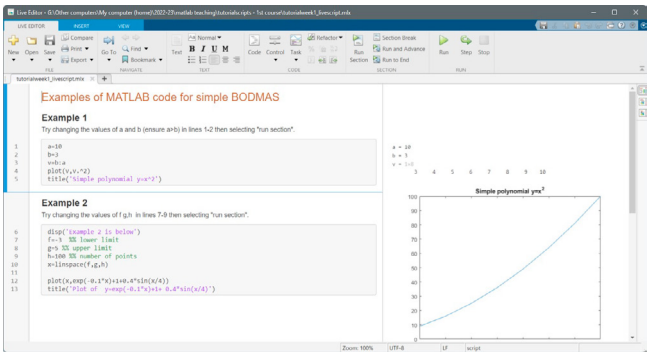

Fig. 3. Illustration of a livescript interface *tutorialweek1_livescript.mlx* to introduce basic MATLAB notation and plotting.

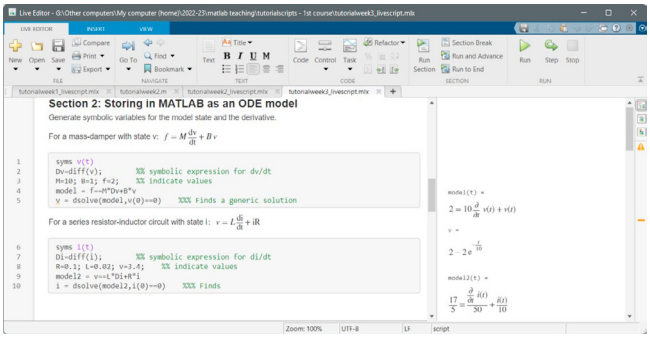

Fig. 4. Illustration of a livescript interface *tutorialweek3_livescript.mlx* for introducing ODE handling.

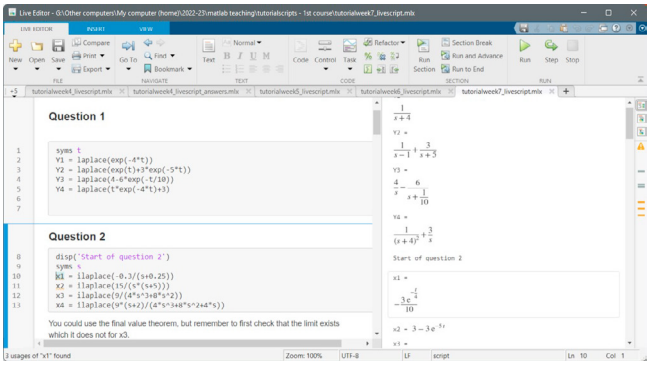

Fig. 5. Introduction to handling of Laplace transforms (*tutorialweek7_livescript.mlx*).

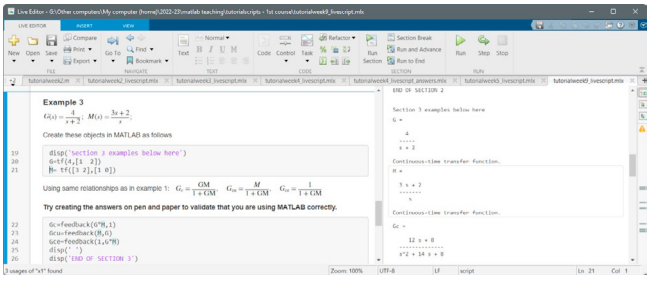

Fig. 6. Introduction to closed-loop transfer functions and their usage (*tutorialweek9_livescript.mlx*).

not happen if students need to spend hours to get to grips with MATLAB itself so the templates offered within the livescript format should provide rapid induction and familiarisation.

It is emphasised that ultimately, more detailed and advanced MATLAB usage should be covered in a programming course or through independent learning. Nevertheless, the expectation is that the livescripts, by providing clear code exemplars, will be useful in support of that.

4. LIVESCRIPTS FOCUSSED ON TOPICS

Section 3 summarised a set of livescripts specifically designed to be used in tutorial sessions alongside a 1st course in control. An alternative requirement could be a student who wants to develop skills in a particular theme of topic.

Due to space limitations, this section will illustrate just a few of the available files; these focus on core concepts and thus may stand outside a given course delivery.

4.1 Summary of supported topics

A question the lecturer might wish to ask is, which engineering concepts would be usefully supported by software visualisation? Of course the answer to this will vary hugely based on the local context, student background, course aims and so forth, so here we set out some reasonable tasks and skills which are useful to many doing a 1st course in control but by no means is this list intended to be exhaustive. For this set of resources we assume that, elsewhere in the degree programme, students are exposed to simple programming concepts and thus have a basic understanding of variables, arrays, functions, plotting and so forth as well as MATLAB. Hence the focus is on how to use MATLAB to support learning of control. Some of the files currently available for download are shown in fig 7, the topic list below and following subsections.

- Simple system dynamics and the link to the solution of ODEs including the impact of parameter changes.
- Characterisation of system behaviours, for example: time constant, gain , overshoot, oscillation frequency, damping, rise-time and more.
- Laplace transforms, inverse Laplace transforms and partial fractions.
- Transfer functions and open-loop behaviours.
- Feedback loop analysis.
- The impact of proportional feedback and systematic design.
- PID compensator design.
- Bode diagrams and frequency response.
- Gain and phase margins and links to closed-loop behaviour.
- Lead and lag compensator design.

4.2 A focus on understanding the time constant form with a 1st order ODE

The livescript file *firstordermodels_and_responses.mlx* focuses on the definition of 1st order models and demonstration of their behaviours (see eqn. (1)).

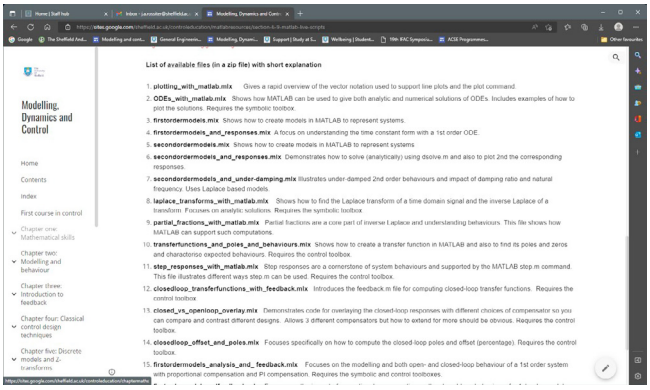

Fig. 7. Website page showing summary of available topic based livescript files.

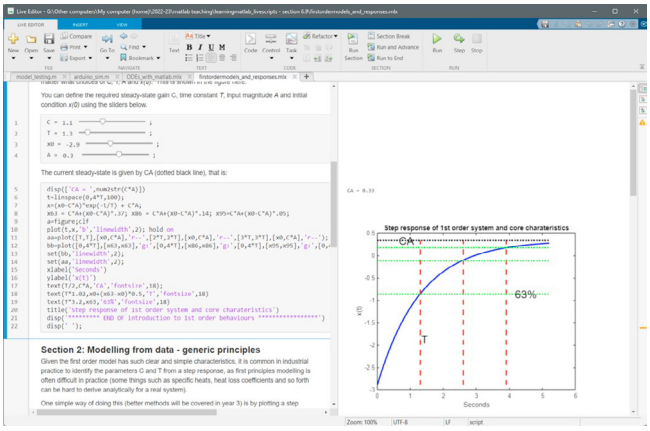

Fig. 8. Estimating time constant from limited step response data.

$$\{Cu(t) = T\frac{dx}{dt} + x(t); \quad x(0) = x_0, \quad u(t) = A\}$$
$$\Rightarrow \quad x(t) = CA(1 - e^{-t/T}) + x(0)e^{-t/T} \quad (1)$$

First order models have simple dynamics and thus are a good tool to introduce students to the concepts of behaviours and how parameter and other design choices can impact on behaviour. The most basic skill discussed in section 1 of the file *firstordermodels_and_responses.mlx* is to link time constant T and steady-state gain C to a step response. It is common to use the time instants $T, 2T, 3T$ to show the convergence speed (63%,86% and 95% movement) - see fig 8.

Section 2 of the file shows how to estimate time constant and gain from a step response with randomly generated examples each time the section is run. Then, in section 3 of the file (figure 9), the livescript can generate an arbitrary example and ask the student to estimate the time constant and gain and compare their estimate with the true behaviour.

A second part is quite similar, but focuses a little more on the mathematical equation of (1) and asks: can you still estimate the time constant if the data you have is not placed at precisely $t = T$ but rather some other time, marked in the figure by the pink cross? In essence it is focussed on the observation that the movement is given by:

$$CA - x(t) = (CA - x(0))e^{-\frac{t}{T}}; \quad (2)$$

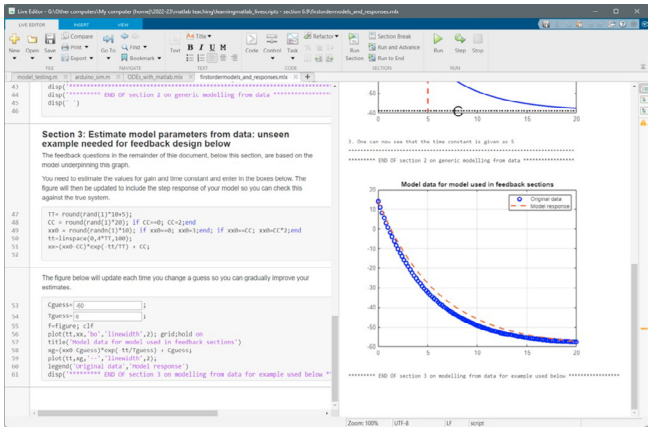

Fig. 9. Estimating time constant and gain from a step response.

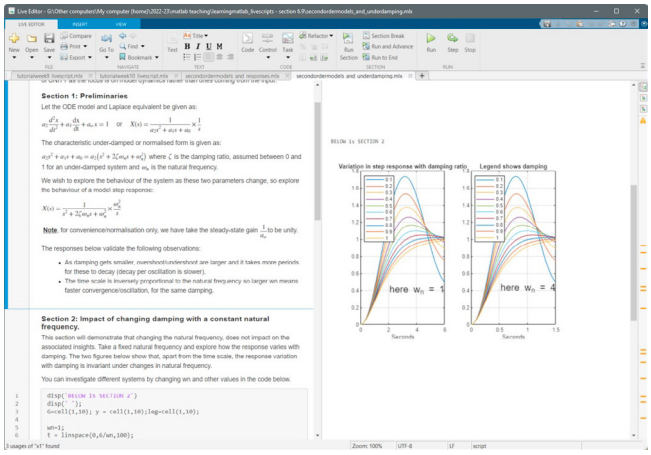

Fig. 10. Part of the livescript on underdamped 2nd order responses.

The time constant can be estimated from (2), assuming the other values are known. Again, students enter their estimate and an overlay of their estimate with the true response is provided.

4.3 Second order behaviours and overshoot

Within second order systems, the most interesting behaviours are those associated to under-damping. The damping ratio affects convergence speed, oscillation frequency, maximum overshoot and also the rate at which the oscillations converge. It is useful for students to be able to explore how model parameter changes impact on these behaviours and so again, a livescript file forms a nice balance between supporting notes and guidance, alongside numerical computations and plotting. Figure 10 shows part of the livescript file *secondordermodels_and_underdamping.mlx* which emphasises how the textual information and plotting is incorporated alongside any MATLAB coding.

4.4 Proportional and lead compensator design using phase margins

While it is accepted that *sisotool* is useful, sometimes students benefit from a slower and more careful presentation

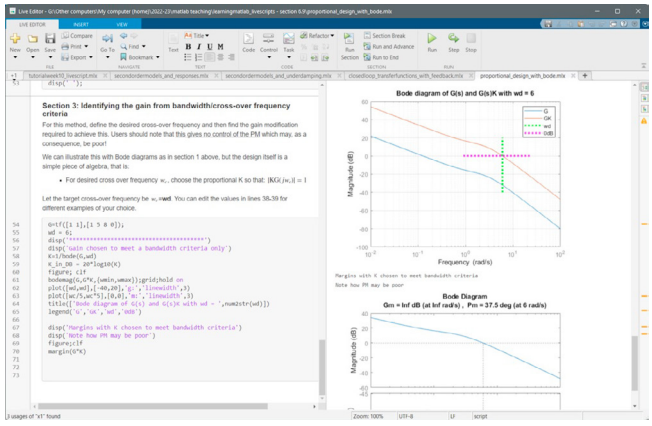

Fig. 11. Proportional design using Bode diagrams and margins.

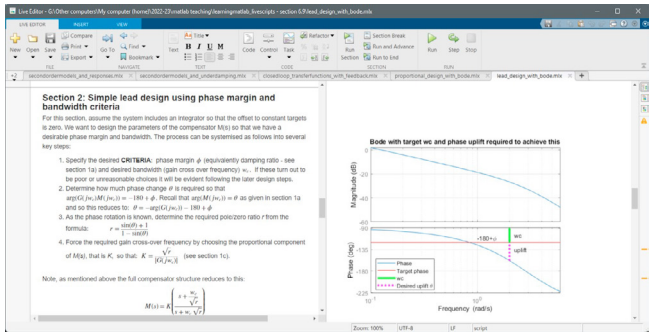

Fig. 12. Lead compensator design using Bode diagrams and margins.

of the core concepts in frequency response design. Examples of this are the files *proportional_design_with_bode.mlx, lead_design_with_bode.mlx*. The advantage of a livescript file is that you can present the conceptual steps alongside the MATLAB code and figures supporting any design computations. Figure 11 shows part of this livescript emphasising the gain cross-over frequency and gain changes required whereas figure 12 shows a livescript which leads students through the steps of a mechanistic lead compensator design with a useful visualisation on the right.

5. ADOPTION IN TEACHING, CONCLUSIONS AND REFLECTIONS

This paper has given an overview of a number of MATLAB livescripts which have been produced to help students with a first course in control, and a few more advanced topics. The nice thing about livescripts is that they allow a combination of neat theoretical and mathematical presentation of concepts and algorithms alongside the corresponding MATLAB code, figures and computations. Students can edit single lines of code, such as a model definition or control specification, and immediately see the impact on figures and core values. Thus the environment provides two core advantages:

(1) They act a little like virtual laboratories or GUIs in allowing simple USER interactions alongside powerful visualisation of the consequences of those interactions. Indeed, they are more powerful than con-

ventional GUIs because students can easily edit and embellish them should that be desirable.

(2) As the code is clearly visible and marked, it also adds visibility to source coding which enables students to develop their MATLAB programming skills; such a skill development will be essential for independent work and later modules and projects.

In summary, the author's view is that appropriate MATLAB livescripts provide a useful supplement to more traditional learning resources and thus are useful support to student learning. The livescripts presented here are openly available on the author's website Rossiter (2021) (Chapter 6) and thus free to use and edit.

The livescripts are being adopted actively in teaching from semester 1 of 2022 (that is October-December). As such evidence of the efficacy will take a while to accumulate and is too early for this submission. Nevertheless, the anecdotal evidence from the first tutorial sessions are that students are using these quite actively compared to other conventional resources and finding them very useful. We hope to have more evidence and are planning a more formal evaluation to be available by the time of the world congress.

REFERENCES

Abdulwahed, M, 2010, Towards enhancing laboratory education by the development and evaluation of the trilab concept, PhD Thesis, University of Loughborough

Albertos, P., 2017, MOOC in dynamics and control, http://personales.upv.es/ palberto/

Antunes, D.J., Using MATLAB Live Scripts to Teach Optimal Control and Dynamic Programming Online, 2021, https://es.mathworks.com/company/newsletters/articles/using-matlab-live-scripts-to-teach-optimal-control-and-dynamic- programming-online.html

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

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/

Goodwin G.C., A. M. Medioli, W. Sher, L. B. Vlacic, and J. S. Welsh, 2011, Emulation-based virtual laboratories: A low-cost alternative to physical experiments in control engineering education, IEEE Transactions on Education, 54:48-55.

Lynch, S. and V. Becerra, 2011, MATLAB assessment for final year modules. In *The use of MATLAB within engineering degrees*. HEA workshop and seminar series.

Nevaranta, N., Jaatinen, P., Gräsbeck, K. and Pyrhönen, O., 2019, Interactive Learning Material for Control Engineering Education Using Matlab Live Scripts, IEEE 17th International Conference on Industrial Informatics, pp. 1150-1154, doi: 10.1109/INDIN41052.2019.8972282.

Panza, S, Invernizzi, D., Giurato, M., Yang, G., Chen, K. and Parisini, T., 2021, Integration of experimental activities into remote teaching using a quadrotor test-bed, 21st IFAC Workshop on Aerospace Control Education.

Quanser, 2022, Quanser experience controls take home app, https://www.quanser.com/experience-controls.

Rossiter, J.A., Giaouris, D., Mitchell, R., and McKenna, P., 2008, Typical control curricula and using software for teaching/assessment: a UK perspective, IFAC world congress.

Rossiter, J.A., 2016, Low production cost virtual modelling and control laboratories for chemical engineering students, IFAC symposium on Advances in control education.

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., Pope, S.A., Jones, B. Ll. and Hedengren, J.D., 2019, Evaluation and demonstration of take home laboratory kit, IFAC Symposium on Advances on Control Education, Philadelphia.

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).

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.

Zapata-Rivera, L.F., Larrondo-Petrie, M.M. and Weinthal, C.P., 2019, Generation of multiple interfaces for hybrid online laboratory experiments based on smart laboratory learning objects, IEEE Frontiers in Education Conference (FIE), 1-8, doi=10.1109/FIE43999.2019.9028421