
Proceedings Paper:
Rossiter, J. orcid.org/0000-0002-1336-0633, Drummond, R., Su, L. et al. (1 more author) (2025) State space files for the control101 toolbox. In: IFAC-PapersOnLine. 14th IFAC Symposium on Advances in Control Education (IFAC ACE 2025), 17-21 Jun 2025, Budapest, Hungary. Elsevier, pp. 54-59. ISSN: 2405-8971. EISSN: 2405-8963.

https://doi.org/10.1016/j.ifacol.2025.08.022

State space files for the control101 toolbox

J.A. Rossiter [*], **R. Drummond** [*], **L. Su** [*], **R.Bars** [**]

[*] *School of Electrical and Electronic Engineering, University of Sheffield, Sheffield, S1 3JD, UK. E-mails: {j.a.rossiter, ross.drummond, lanlan.su}@sheffield.ac.uk*
[**] *Budapest University of Technology and Economics, Hungary. E-mail: bars.ruth@aut.bme.hu*

Abstract: There was somewhat a split in opinion (Rossiter et al., 2020) on whether state space methods should be part of an introductory control course. Consequently, it makes sense for the control101 toolbox (Rossiter, 2024) to include state space resources, to support those places where this is considered fundamental. In view of this, the authors have been developing a set of resources to support an introduction to state space methods in control and this paper gives an overview of those resources (released in late summer 2024), alongside some supporting background.

Keywords: Independent learning, state space models, Control101.

1. INTRODUCTION

It has now been well reported that the control community, supported by the relevant IFAC and IEEE technical committees, in recent years has been looking at what constitutes an ideal first course in control (Silverstein et al., 2015; Rossiter et al., 2020). While of course there were differences of priorities in the middle areas, there is a good consensus on the most important aspects, for example:

(1) Focus on motivation and concepts with interesting case studies and of course laboratories.
(2) Do not over-emphasis mathematics and rigorous analysis or numerical computations.
(3) Help students understand the importance of feedback control and behaviours and indeed the power of feedback.
(4) Use software to support computation and illustration, thus avoiding the need for excessive and slow paper based computation.

Of course this is somewhat of a simplification of the findings (Rossiter et al., 2020) and, in the middle ground there was more diversity of opinion on what could or should be included. One particular area where the vote was more evenly split is the divide between Laplace Transforms and frequency response methods as opposed to time domain state space methods. In consequence, any set of resources designed to support a generic first course in control would, ideally, support both approaches to reflect the local needs of each institution.

1.1 Shared open-access resources

It is not enough to have agreement on a conceptual curriculum. The community would also like to share learning and teaching resources as these both ensure global consistency and also make it far more efficient for us to provide high quality learning experiences to our students; herein however lies a challenge. There is wealth of high quality resources, e.g. (Rossiter, 2021; Albertos, 2017; Quanser, 2022; Guzman et al., 2023; de la Torre et al., 2013, 2020; Dormido et al., 2011, 2012; Goodwin et al., 2011; Cameron, 2009; Rossiter et al., 2018; Murray et al., 2004; Matisak et al., 2023; Rossiter et al., 2024) and staff do not know where to start, which ones to use and more. Moreover, adopting a resource developed elsewhere can equally be cumbersome and time consuming due to differences in software, notation, starting assumptions and more. The community (Serbezov et al., 2022) is continuing to work on this issue with the hope of providing more streamlined and coherent resources to our colleagues, and indeed one early example is (Douglas, 2022).

1.2 The control101 MATLAB toolbox

In parallel with the above discussions, it was felt that a set of learning resources within MATLAB would be a useful contribution to the community, because any resources in this environment are likely to fit relatively seamlessly into any curriculum where MATLAB usage is already assumed. Moreover:

(1) It is widely used in the control community and by students on control so students may already be familiar with the syntax and use.
(2) The download of toolboxes is quick and easy, thus ameliorating some of the clumsiness that can come with adopting resources from elsewhere.
(3) Any shared resources are in a transparent environment which allows easy and often self-evident editing to meet the local needs of both staff and students.
(4) It allows 24/7 access and simultaneous independent access by large numbers and with effectively instantaneous run times.
(5) As an aside, although not central to this paper, the environment allows quick and easy development and editing of interactive resources (Rossiter, 2016, 2017;

Koch et al., 2020; Nevaranta et al., 2019; Rossiter, 2022, 2024).

Remark 1. It should also be emphasised that Mathworks has recently made access to MATLAB online (https://uk.mathworks.com/products/matlab-online.html) free for anyone and toolboxes can be downloaded and used with in this environment as well as users having some limited local storage space.

The focus of this paper is on resources that one might argue are motivational and have less depth in analytical detail in the first instance, that is ones which allow users to do quick and easy testing of concepts. The initial resources in the toolbox were designed to focus on supporting two core aspects.

(1) Quick and easy testing and illustration of concepts, for example:
   - How will the system behaviour change if I increase parameter A?
   - What is the impact of different feedback terms on the behaviour?
   - How is the behaviour affected by uncertainty?
(2) Exemplars of code for doing straightforward but tedious number crunching (e.g. closed-loop poles, inverse Laplace, step responses, etc.).

The initial resources (Rossiter, 2024) were officially launched at the IFAC world congress in 2023 and covered the bare bones of a first course, largely being focused on Laplace transform methods and analysis. Since then efforts have focused on extending the resources to cover more breadth. More recent additions involving a range of international colleagues include frequency response, PID design (Rossiter et al., 2024), SIMULINK, non-linear behaviours and, specifically discussed in this paper, state space methods. Nevertheless, it is worth re-iterating that the authors recognise there will still be gaps and opportunities to improve and extend these resources and welcome anyone wishing to collaborate on this.

In summary then, this paper introduces the new state space resources which were in the summer 2024 release of the `control101` toolbox. Section 2 will give some brief discussion on the chosen coding environment and how users can edit these for their own purposes, if desired. Section 3 summarises the files produced alongside a discussion of how they might be used. Section 4 gives some provisional evaluation of the use of the resources with a student cohort in the Autumn of 2024. The paper then finishes with some conclusions.

2. AN INTRODUCTION TO THE MATLAB AND SIMULINK ENVIRONMENT

Before introducing the resources in detail, it is helpful to summarise the environment these resources are coded in and to illustrate how users can engage with them. For state space methods, most of the early resources are in the form of livescripts but the later resources involving feedback and observers also make use of SIMULINK. As with all the code in the toolbox, the code is open source and thus users are free to download and utilise however they need; source code is available from github (`https://github.com/jarossiter/control101`).

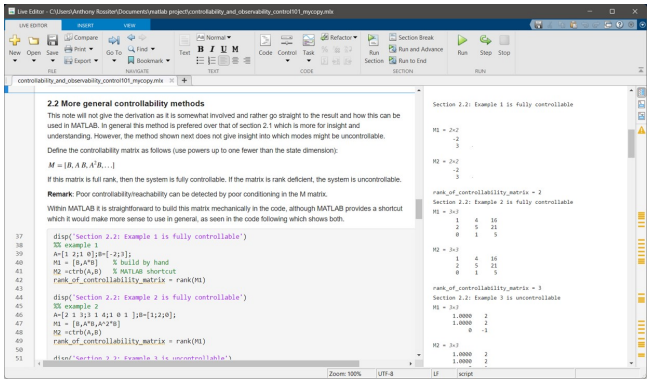

Fig. 1. Illustration of the livescript environment: computation of controllability matrices.

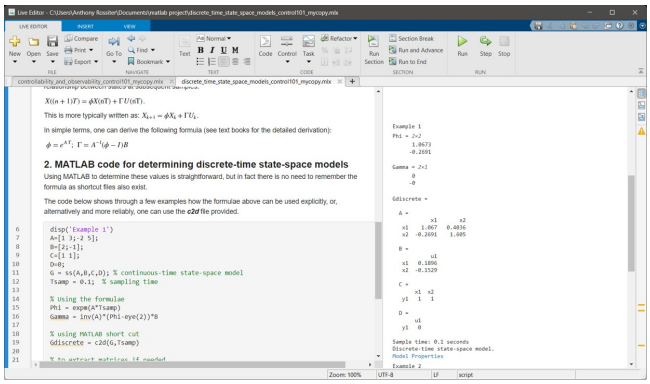

Fig. 2. Illustration of the livescript environment: computation of discrete state space parameters.

2.1 *The livescript environment*

The livescript environment is one which combines both notes and code within an integrated setting. Essentially one is able to provide detailed explanation in a relatively professional format and then supplement this with the code required to do the associated computations. This format is illustrated in Figure 1-2 where the reader can see the explanation in the top left, the code in the lower left (shaded light grey) and the results in the right hand column.

Livescripts can be organised into sections, so that users only need to run the code snippets within a single section at a time. Users can change the numbers, and indeed code if they know how, and then simply select *run section* button in the top banner to update the computations.

Consequently, the resources have been largely written using livescripts because these allow detailed note like explanations to be provided, followed in short sections by code snippets to do the relevant computations. Users need only edit, for example, the state space matrices in the top of the code snippet in order to perform whatever numerical computations are required.

It should be re-emphasised here, the aim is to use MATLAB for the number crunching which in this case is tedious, has little intellectual value and for large dimensions is not amenable to paper and pen computation. This does not mean students are discouraged from understanding properly the associated analysis and indeed the note aspect

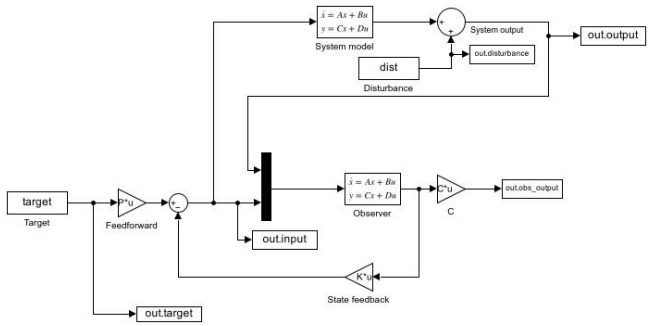

Fig. 3. Illustration of the SIMULINK model to represent observer behaviour with uncertainty.

of the livescript encourages engagement with this and expects students to relate this to the code.

2.2 The SIMULINK environment

While m-files and code are effective for handling simple analysis and simulation of linear systems, including state space models, there are some aspects which are much more clumsy to code and therefore better or more clearly represented using block diagrams. One example of this is the handling of parameter uncertainty alongside an observer. For this reason, it makes sense for the last few items covering state space methods to exploit the SIMULINK environment as it was felt, this would make the resources easier to understand and modify.

A simple example of this is given in Figure 3 which demonstrates the difference between an observer output and the true system output. Not only is it possible for the state space parameters A, B, C, D to be different in each block, but also one can easily incorporate unknown disturbances. The model and observer parameters data can be defined and using a short script which also runs the file, collects the output data and produces the relevant plots. For example:

```
A=[-0. -0.4 -0.6 0.2;0.4 1 .7 .5;-0.3 1 1 0;...];
B=[1 0;-1 1;0 0.3;-1 2];
...
% Control design
Q = [C'*C]; R=eye(2);
K=lqr(A,B,Q,R);
...
% simulation
Data= sim('filename.slx');
y=Data.output.signals; % system output
d=Data.disturbance.signals; % disturbance signal
r=Data.target.signals; % target
yo = Data.obs_output.signals; % observer output
u = Data.input.signals; % input
```

Remark 2. SIMULINK models are not backwardly compatible. The models currently in the toolbox run on versions 2023 and later. If you run in a later version, you may get an alert.

Remark 3. A strategic decision was taken not to run any files using the options directly within the SIMULINK editor, such as the run button. Most models have a large number of parameters and signals which need to be defined. In order for all these parameters to be flexible for the user to change without having to open and close numerous SIMULINK blocks, it is easier to define the values in a code snippet with values then imported into SIMULINK. Similar advantages come from exporting the output data and plotting directly with a code snippet.

The livescripts which utilise SIMULINK are organised into sections a little like a workbook so that users can step through these and gradually engage with more complex concepts and simulations. The sections all have the relevant code snippets and plots which can be run independently through the run section button. Code snippets can be copied to a simple m-file and run through that medium should the user find this more convenient.

3. FILES SUPPORTING STATE SPACE METHODS IN THE CONTROL101 TOOLBOX

This section will summarise the available files and scenarios within the toolbox. The files given here are a first draft and will be used actively with students during the Autumn 2024 to gain some feedback and evaluation data. The intention is to modify and extend these as appropriate in due course. It is also worth repeating that the authors would like the wider community to share this toolbox and feel at ease proposing edits and additions which perhaps reflect the needs and foci of their institutions or indeed a broader perspective.

A full list and concise summary is given in Table 1. The following section explains the current thinking in more detail and gives some illustrations.

3.1 State space basics

The first set of files focuses on basic definitions and analysis tools, so things such as:

(1) What is a state space model?
(2) Links between state space models and transfer function models.
(3) Simulation/behaviour, state transformation matrices and discretisation.
(4) Analysis of the dynamics (poles, phase plane, etc.).
(5) Non-uniqueness and equivalent forms.

In all cases, the files have some explanatory text and analysis followed by illustrative MATLAB code that can be used to do the onerous computational aspects that may be needed, for example as shown in Figure 4.

3.2 Feedback design, observers, feedforward and integral action

The next set of files focuses on how one might change behaviour using state feedback. It is quickly noted that state feedback does not handle tracking and also assumes one has access to the states, thus different files deal with both theory of these aspects and also, how one might code and simulate the corresponding scenarios (e.g. Figure 5).

3.3 Model uncertainty and SIMULINK

The final group of files exploit SIMULINK as this allows clearer and simpler incorporation of disturbances, uncertainty, integral action and tracking (e.g Figure 6). The

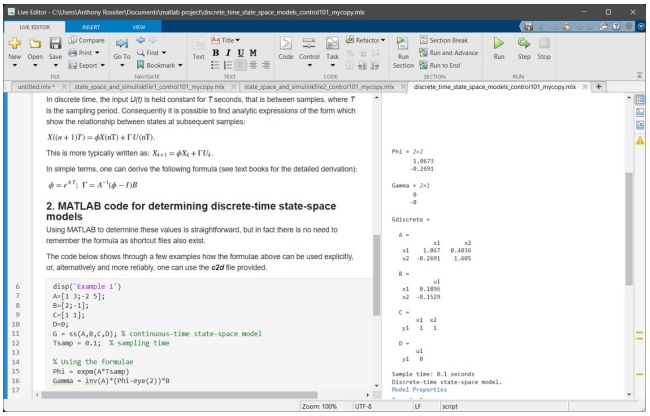

Fig. 4. Screen dump from the livescript focusing on discretisation.

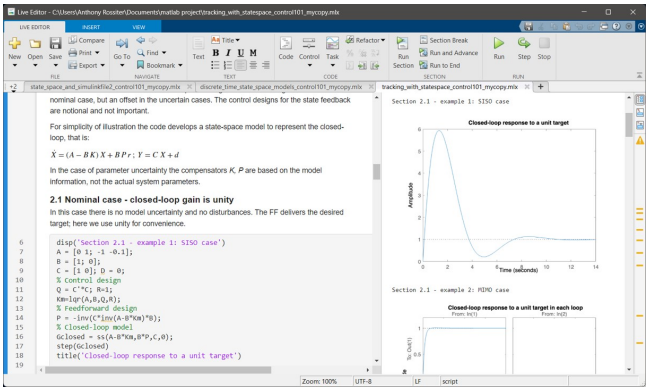

Fig. 5. Screen dump from the livescript focusing on feedforward design.

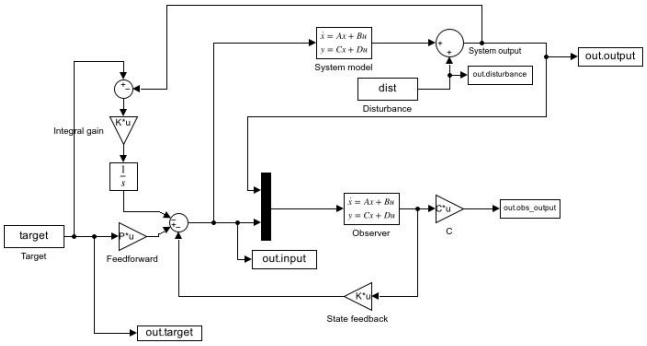

Fig. 6. Screen dump from the SIMULINK file handling uncertainty with observers and integral action.

SIMULINK files allow the user to see and understand the structure of the controller, while the livescript code snippets provide a convenient mechanism for selecting state parameters and doing the state feedback and observer designs. Similarly, the livescript file provides detailed explanation and analysis of the required design steps.

4. EVALUATION OF USE OF RESOURCES WITH STUDENTS

As the state-space files for the `control101` toolbox were only released in late Summer 2024, a quantitative evaluation of their impact is still ongoing. However, initial responses from students have (so far) been encouraging. During Autumn 2024, the toolbox was included as an additional learning resource within a third-year course on "*State-Space Control Design*" at the University of Sheffield, and feedback from the students on the toolbox have (so far) generally been positive. Particular emphasis was given to the benefits in translating the mathematical concepts the students had learned in lectures into practically implementable code. Many students noted that, after some self-study, they could understand the mathematics of the state-space concepts and implement the methods in toy examples by hand (e.g. computing eigenvalues of A matrices) but struggled to relate them to practical problems– such as when observability/reachability could be used in real systems. Addressing this "*concept-to-practice*" gap was one of the main benefits of the codes, at least from what could be gathered from student feedback.

It was also observed through student interactions that the files often instilled the students with new confidence in coding control methods. Again, after some self-study, the students were able to understand the state-space concepts mathematically, but not how to implement them in practice. The files helped give the students a "*leg-up*" in this regard, as they provided them with a set of simple codes with explanations that could, in very few lines, implement those core concepts using off-the-shelf toolboxes and functions. Doing so seemed to help demystify the concepts such that students could start implementing them on their own. In particular, it was observed that many of the students cannibalised aspects of the code for their Individual research projects (including those on controlling drones and filtering). This cannibalisation was encouraged, as it increased engagement in state-space control and accelerated student progress on their projects. Students appeared to have made more rapid progress through the "initial" phases of their projects towards the more advanced/interesting objectives, such as adapting the codes for linear quadratic regulators to drone control problems.

Another recurring comment from the student feedback was appreciation for the live script format. Specifically, students noted the benefits of embedding the mathematical concepts and implementable code in the same file, as it meant they did not have to flip back and forth through notes or keep on downloading separate files from browsers. Whilst combining the text and the code in the same files is a seemingly trivial detail, by making the learning experience of the students slightly easier, it seemed to improve engagement. A secondary benefit that appeared from student interactions was that the files showed students how an engineer would embed code into engineering workflows. As in, the files encouraged the students to not think of coding as a separate activity, but more of a tool to shape an argument and justify conclusions with numerical results. Whilst this perspective had been pushed to students in previous assignments, it had been left to student to develop their own arguments. With the toolbox, this work flow was more direct and prescribed. Whereas these secondary benefits were harder to quantify in the feedback, from discussions it appeared that the logical breakdown gained from embedding the code within the text seemed to reduce barriers (mostly in the gap between the mathematical concepts and the practical implementation) in learning state-space concepts.

5. CONCLUSIONS AND REFLECTIONS

This paper has give a rapid introduction to the new state space resources that have been included recently in the control101 toolbox. Some brief background was given on the toolbox thus the rationale for adding state space resources. The paper then gave an overview of the coding environment to help readers understand how students and staff could engage quickly and easily with the proposed resources. This was followed by a more detailed discussion of the specific resources that are currently available to support student learning and application of state space methods. Finally, the resources were deployed with a single cohort in the Autumn of 2024 and some evaluation of their experiences is given.

The general feeling is that these resources have helped students enormously, especially with reinforcing and applying their learning to a broader range of systems (e.g. high order) where paper and pen computation is not viable and/or too slow. Naturally, this is a first pass and the authors would encourage colleagues to help improve and extend these resources which are freely available to all.

REFERENCES

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

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

S. Dormido, J. Sanchez-Moreno, H. Vargas, L. de la Torre, and R. Heradio. UNED labs: a network of virtual and remote laboratories. In Javier Garca Zuba and Gustavo R. Alves, editors, Using Remote Labs in Education: Two Little Ducks in Remote Experimentation, pages 253-270, Bilbao, 2011.

S. Dormido, H. Vargas, J. Sanchez, 2012, AutomatL@bs Consortium: A Spanish Network of Web-Based Labs for Control Engineering Education, Internet Accessible Remote Laboratories: Scalable E-Learning Tools for Engineering and Science Discipline, 11, 206-225, A. Azad, M. E. Auer, V. J. Harward (Ed), IGI Global.

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

Ferrari, M., Visioli, A., 2023, An Educational Interactive Software Tool to Learn Cascade Control Design, IFAC world congress.

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.

Guzman, J.L., Costa-Castello, R., Berenguel, M. and Dormido, S., 2023, Automatic control with interactive tools, Springer. https://doi.org/10.1007/978-3-031-09920-5

Koch, A., Lorenzen, M., Pauli, P. and Allgower, F., 2020, Facilitating learning progress in a first control course via Matlab apps?, IFAC world congress.

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

Murray, R.M., Waydo, S., Cremean, L. and Mabuchi, H., 2004, A new approach to teaching feedback, IEEE Control Systems Magazine, 24, 38-42

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.

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

Rossiter, J.A. and L. Gray, 2010, Supporting development of independent learning skills. Engineering Education.

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

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., Evaluation and demonstration of take home laboratory kit, IFAC Symposium on Advances on Control Education, Philadelphia, 2019.

Rossiter, J.A., Serbezov, A., Visioli, A., Zakova, K. and Huba, M., 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, 2020. 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., 2024, A novel MATLAB toolbox for Control101 courses, European Journal of Control, 2024. https://doi.org/10.1016/j.ejcon.2024.101041

Rossiter, J.A., Visioli, A., Dormido, S. and Bars, R., 2024, A MATLAB virtual laboratory to support learning of auto-tuning PID approaches, 4th IFAC Conference on Advances in Proportional-Integral-Derivative Control.

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.

D.L. Silverstein, Vigeant, M.A and Staehle, M., 2016, How do we teach process control: 2015 survey results, ASEE Annual Conference.

File name	File summary
statespace_models_control101.m	Advanced approaches to compensator design to take more explicit account of sensitivity to uncertainty.
statespace_models_from_tf_control101.m	This file focuses on how equivalent state space models can be determined from transfer function models, both continuous and discrete.
tf_models_from_statespace_control101.m	This file focuses on how equivalent transfer function models can be determined from state space models, both continuous and discrete.
discrete_time_state_space_models_control101.m	This file focuses on how to determine a discrete time state space model, given one already has the continuous time model. Some brief theoretical background is given before focussing on the code.
statespace_models_from_tf_control101.m	This file focuses on how equivalent state space models can be determined from a transfer function model, both continuous and discrete.
openloop_behaviour_statespace_control101.m	This file focuses on how to infer and explore the expected open-loop behaviour of a state space models, both continuous and discrete.
phaseplane_behaviour_statespace_control101.m	This file focuses on how to infer the expected open-loop behaviour of a state space models, within the phase plane, that is the behaviour of the states rather than the inputs and the outputs. It is interesting to note how this behaviour can be linked to the eigenvalue/vector decomposition of the state matrix.
equivalent_state_space_formulations_control101.m	This file focuses on equivalent forms, that is, state space models which appear different but have the same input-output behaviour. The file gives a brief overview of the underlying relationships and then gives MATLAB code for showing the relationships between two equivalent forms.
controllability_and_observability_control101.m	Controllability and observability are important properties of state space models which determine the extent to which effective state feedback compensators and observers can be implemented.
state_feedback_control101.m	When a system is expressed as a state space model, it is normal to design the closed-loop compensator as a state feedback. This file gives an introduction to the topic and relevant MATLAB code to support the associated design and analysis.
optimal_state_feedback_control101.m	When a system is expressed as a state space model, it is normal to design the closed-loop compensator as a state feedback. This file gives an introduction to the popular method of 'optimal control'.
observer_design_control101.m	In practice the states are not known and need to be inferred or estimated from available measurements; an observer is used for this process. This file introduces the concept of the observer quickly and then focuses on how to use MATLAB to support his approach.
tracking_with_statespace_control101.m	This resource introduces simple methods for dealing with non-zero constant targets and then looks more systematically at how integral action can be incorporated into a state space framework, with an observer and model uncertainty.
state_feedback_and_estimation_cont_case_study.m	The purpose here is to undertake case studies showing how the different analysis and design tools above can be used to formulate a complete control feedback design. The design method used here is pole placement.
state_feedback_and_state_estimation_discrete_case_study.m	The purpose here is to undertake case studies showing how the different analysis and design tools above can be used to formulate a complete control feedback design for a discrete system. The design method used here is pole placement.
state_space_and_simulinkfile1_control101.m	Introduction to using simulink with state space models. Simple scenarios only. Simulink files: state_space_simulinkfile_openloop_control101.slx, state_space_simulink_feedback_control101.slx, state_space_simulink_feedforward_control101.slx.
state_space_and_simulinkfile2_control101.m	Introduces the concepts of feedforward to improve tracking and observers to estimate states. Uses simulink file state_space_simulinkfile_observer_control101.slx
state_space_and_simulinkfile3_control101.m	Introduces integral action into the feedback loop along with an observer. Demonstrates efficacy in presence of uncertainty. Uses Simulink file: state_space_simulink_integral_control101.slx

Table 1. Summary of files covering state space methods in the control101 toolbox (https://github.com/jarossiter/control101).