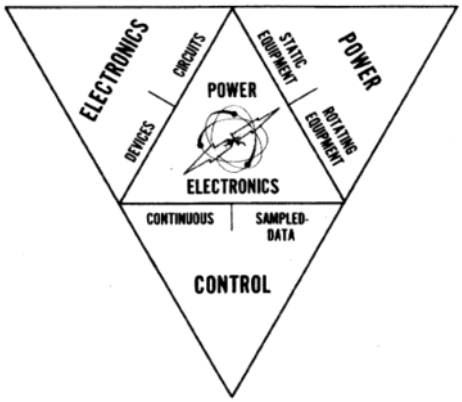
Lecture 6: Modeling and controller design

Dr. Yee Wei Law ⟨yeewei.law@unisa.edu.au⟩

Saturday, 15 May, 2021 10:27 AM

Definition: Power electronics

... is the study of electronic circuits intended to *control* the flow of electrical energy, or equivalently the conversion of electric power, and which handle power flow at levels much higher than individual device ratings.



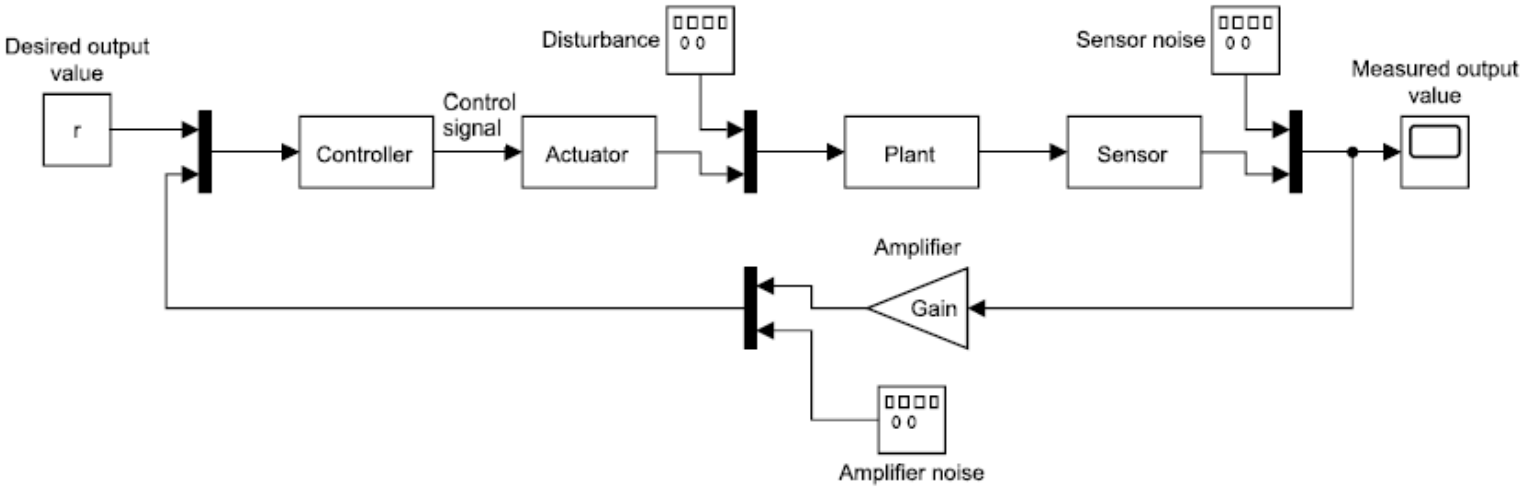
In the definition above,

- We can see three areas of electrical engineering: *electronics*, *power*, and *control*.

Definition: Control

... is the action of adjusting inputs to obtain desirable outputs.

Simplest control architecture:



This lecture requires existing knowledge in classical control theory and basic modern control theory, which you would have learnt from EEET 3046 Control Systems.

The "Control and power electronics" section of the Cyber Engineering Knowledge Base contains some relevant revision material tagged with "control":

GLOSSARY ENTRIES

- Applications of control: examples
  Control and power electronics
  Cyber Engineering Knowledge Base
- Control systems: introduction
  Control and power electronics
  Cyber Engineering Knowledge Base
- PID control
  Control and power electronics
  Cyber Engineering Knowledge Base
- Frequency response
  Control and power electronics
  Cyber Engineering Knowledge Base
- Time response
  Control and power electronics
  Cyber Engineering Knowledge Base

Show only tagged Glossary entries More »

GLOSSARY ENTRIES

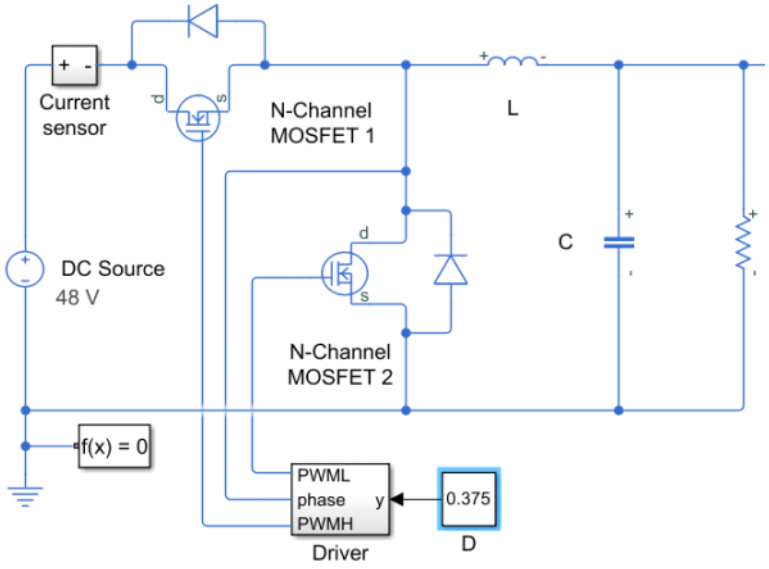
- State-space equations
  Control and power electronics
  Cyber Engineering Knowledge Base
- System dynamics and modelling
  Control and power electronics
  Cyber Engineering Knowledge Base
- Stability
  Control and power electronics
  Cyber Engineering Knowledge Base

« Back Show only tagged Glossary entries

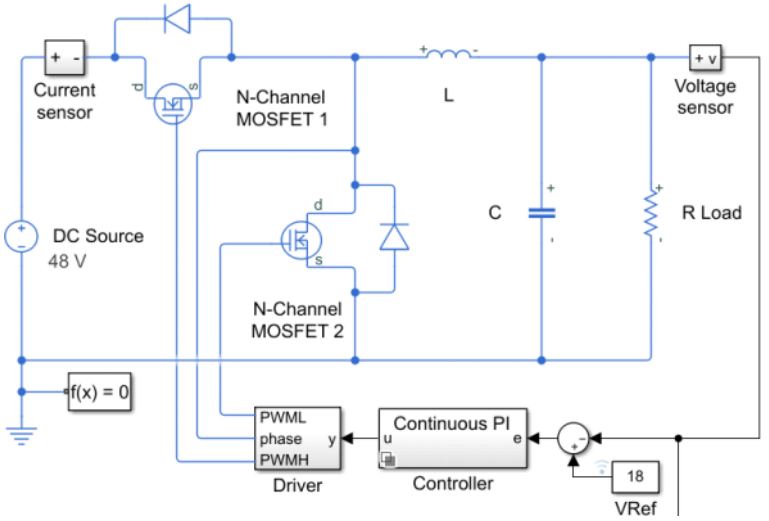
In Lecture 4 “DC-DC step-down conversion”, we have witnessed the importance of *control*.

- DC choppers using the calculated duty ratio cannot track the desirable output voltage exactly.

Open-loop



Closed-loop



In general, control objectives include tracking a set output value and maintain good power quality to meet certain standards.

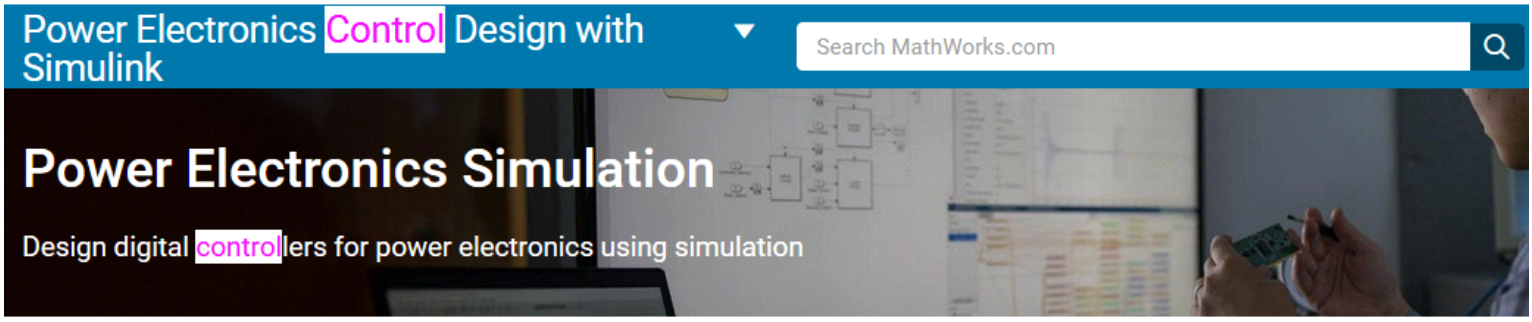
- Example 1: For switched-mode power supplies, the main objective is to track a constant DC voltage — with voltage variations contained within certain limits around the rated value — irrespective of the load.
- Example 2: Rectifiers usually impose a dual goal, namely to regulate the output voltage, while extracting a controllable reactive power.
- Example 3: Stand-alone inverters are required to maintain output voltage and frequency irrespective of load conditions.

The mainstream approach to achieving control is ***model-based control***.

- Given a system, describe the behavior of the system using a mathematical model.
- Power electronic circuits are intrinsically nonlinear due to presence of switches, diodes, potentially discontinuous currents, etc., but luckily models used for control purposes are generally simpler than those used for circuit design or simulation [BMB14, p. 3].
- A controller is designed or synthesized based on the model.
- If the model is single-input single-output (SISO), then ***classical control*** is sufficient.
- If the model is multivariable (either multi-input or multi-output), then ***modern control*** based on state-space techniques is necessary.
- Usually ***linear control*** is sufficient [Kaz16, p. 398].

Efficient control system development requires the support of ***model-based design*** tool.

- See https://au.mathworks.com/solutions/power-electronics-control/power-electronics-simulation.html



Power conversion requires control of IGBTs, power MOSFETs, and other solid-state power electronics. Designing a digital controller with simulation can help ensure stability, improve power quality, optimize dynamic performance, and handle fault conditions. Power electronics simulation provides insight into interaction of digital control algorithms, power semiconductors, and the balance of the electrical system early during development, before hardware testing begins. For battery management systems and power electronics–based systems such as motor drives, power converters, and inverters, fast closed-loop simulation enables power electronics engineers to evaluate and verify their design choices before a controller is implemented.

Power electronics simulation should be considered for the following tasks:

Power electronics simulation should be considered for the following tasks:

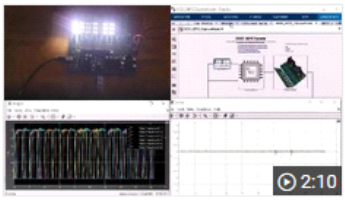
- Designing and validating new topologies and control strategies
- Optimizing system behavior using model libraries of energy sources, power semiconductors, passive circuit elements, and machines such as PMSM and induction motors
- Analyzing system response to faults and abnormal conditions
- Eliminating design problems found through simulation before moving to implementation
- Reusing models to speed up design iterations and next-generation projects

How to Develop DC-DC Converter Control in Simulink

Learn how to model and simulate a DC-DC converter in Simulink® and Simscape Electrical™. Get a demonstration of SEPIC circuit topology and how to model and simulate a DC-DC converter that powers a strip of LEDs. MathWorks engineers show how to use Simulink and Simscape Electrical to develop, simulate, and implement a controller that maintains desired output voltage in the presence of input voltage variations and load changes to achieve a fast and stable response. See how to use control algorithms to generate embedded code optimized for implementing on a Texas Instruments™ C2000™ microcontroller. Also explore hardware-in-the-loop (HIL) testing of the microcontroller using a Speedgoat® real-time target machine.

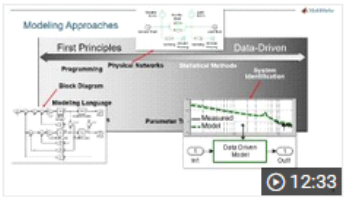
Highlights:

- Modeling and simulating passive circuit elements, power semiconductors, and varying power sources and loads
- Simulating the converter in continuous and discontinuous conduction modes
- Determining power losses and efficiency of the converter
- Tuning the controller to meet rise time, overshoot, and settling time
- Generating C code from the controller model for implementation on a Texas Instruments C2000 microcontroller
- Using a Simscape Electrical model deployed to an FPGA implemented in a Speedgoat real-time target machine for HIL testing



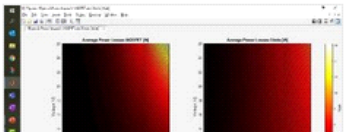
Part 1: Introduction and Demo

Get a quick introduction to the topic of DC-DC converter controls, including a customer reference story and demonstration of the entire system working as desired.



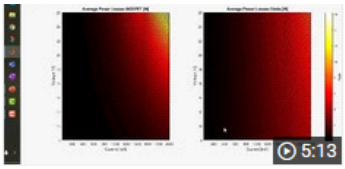
Part 2: Converter Modeling and Efficiency Considerations

Learn how to model a DC-DC converter in Simscape and use simulation results to generate efficiency maps for the diode and the power switch.

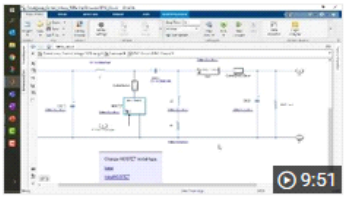


Part 3: Power Losses Investigation

Learn how to use Simscape Electrical functions to generate maps of heat losses to embed in a dedicated model for fast simulations of thermal behavior and sizing of cooling systems.

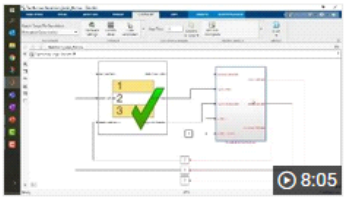


Learn how to use Simscape Electrical functions to generate maps of heat losses to embed in a dedicated model for fast simulations of thermal behavior and sizing of cooling systems.



Part 4: Voltage Control Design

Learn how to design and tune a digital PID controller for a DC-DC converter. Using System Identification Toolbox, engineers can simplify the tuning of any power electronics converter without needing to average converter equations.



Part 5: Supervisory Logic Design and Testing

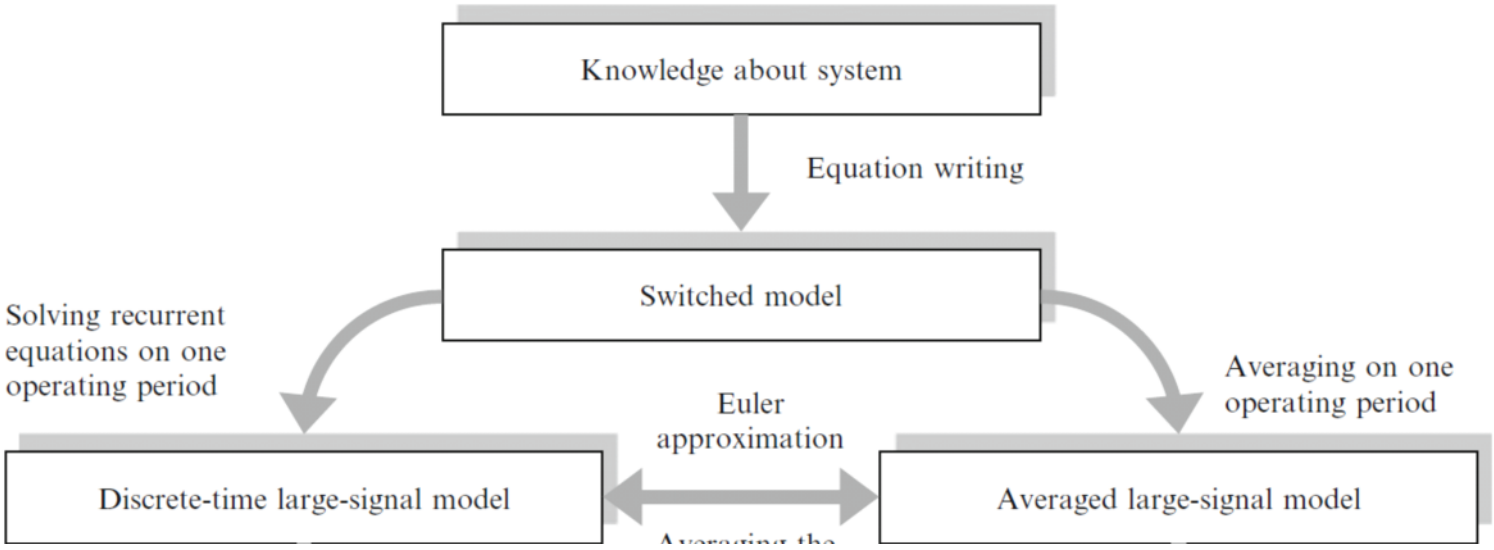
Learn how to use Stateflow to design supervisory logic state machines that manage your converter desired operating mode.

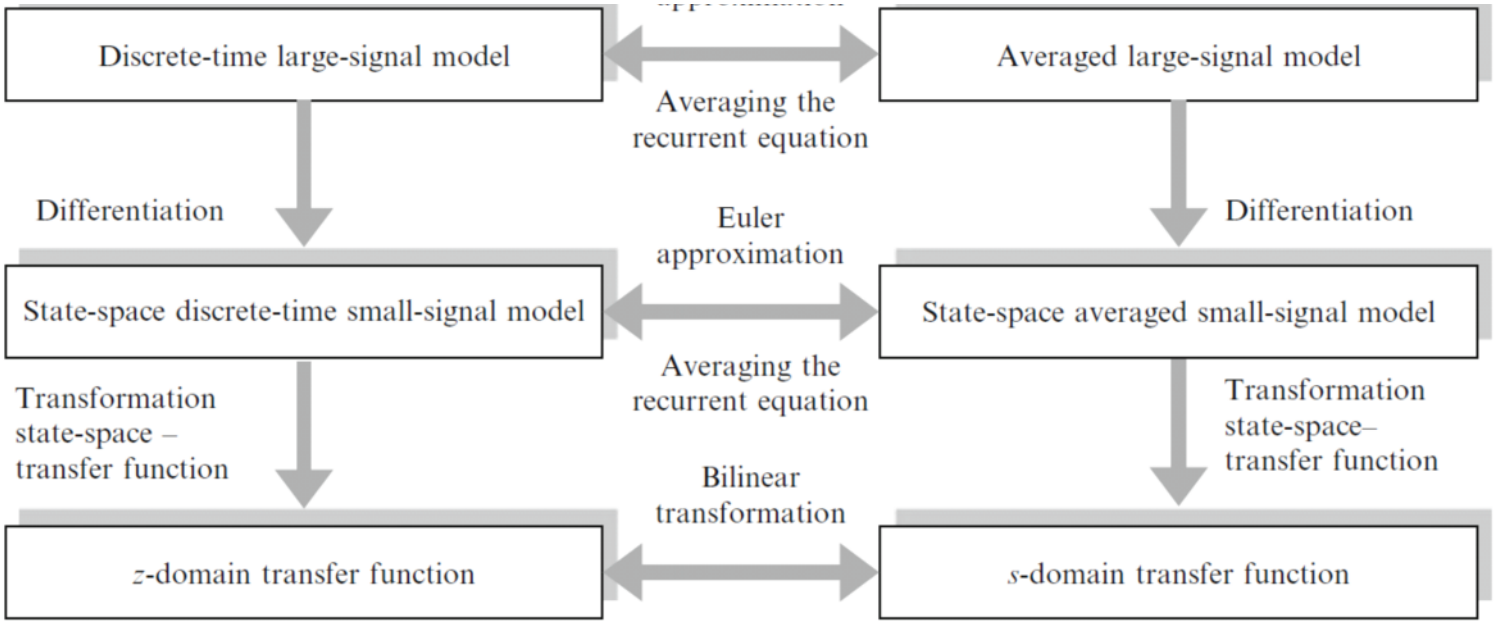
Part 6: Automatic Code Generation and Conclusions

Learn how to automatically generate C code from your model and use the TI C2000 Hardware Support Package to build, compile and run the application on target hardware.

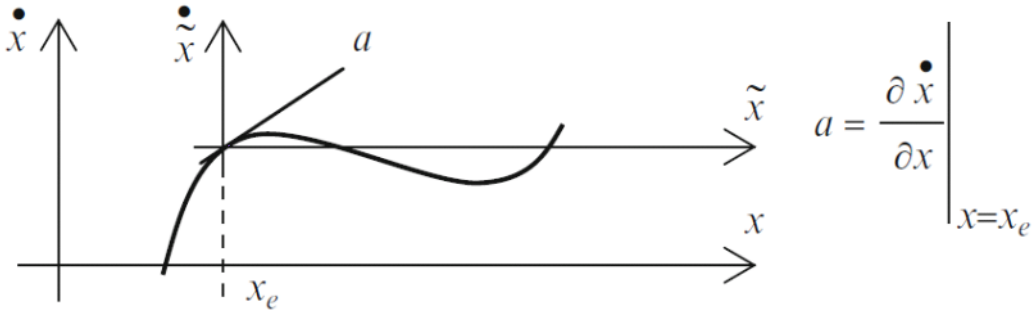
The first step of control system design is of course modeling, which is a focus of this lecture.

- This lecture focuses on modeling approaches specific to power electronics. We will start with ***differential-algebraic models*** (through Simscape Electrical), then continue to ***switched models***, ***averaged models*** (equivalently, *average models*) [BSB14, p. 23].



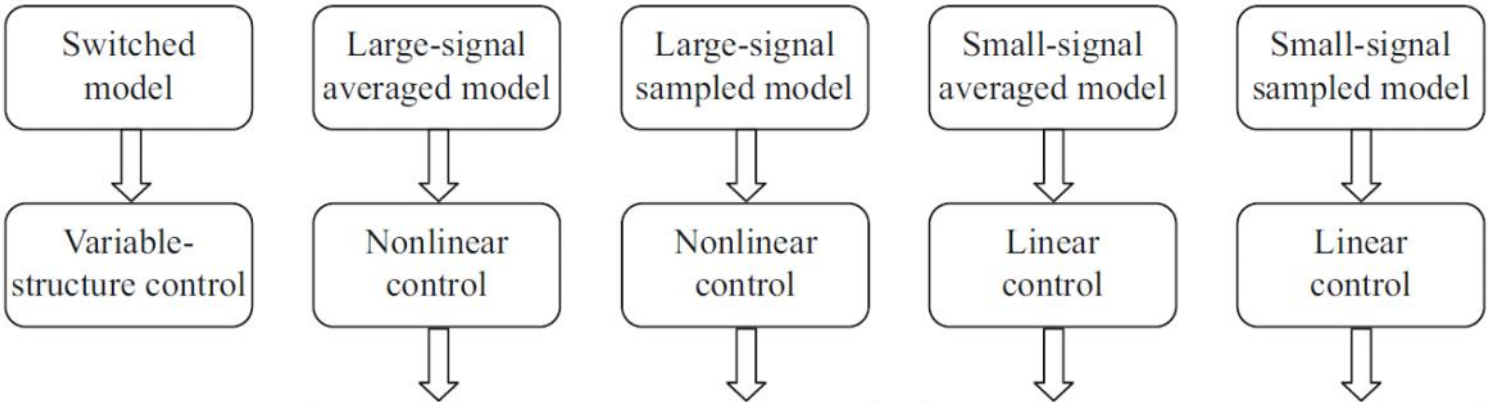


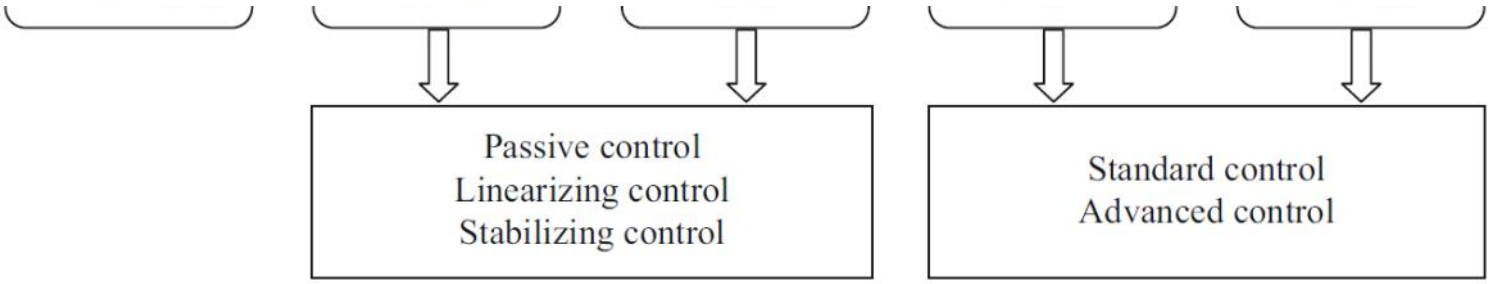
Above, ***large-signal*** models are models that are valid for all values of the state variables, whereas ***small-signal*** models are ***linearized*** models. A linearized model, also called a ***tangent linear*** model, is a model derived from a large-signal model using the DC and first-order terms of the Taylor's series expansion of the large-signal model about some *operating point*.



This is why the further away the system states are away from the operating point, the less accurate the small-signal model is.

- The choice of model informs the choice of control scheme and vice versa [BSB14, p. 23].





We will only use linear control schemes in this lecture, so our focus is small-signal averaged models. We cover switched models because they are the most intuitive and accurate.

- The linear control system design methods used here are covered in EEET 3046.

2. DAE model and PID control

Tuesday, 25 May, 2021 5:52 AM

In general, Simscape Electrical models are a set of differential-algebraic equations (DAEs).

A set of DAEs in fully implicit form takes the form

$$\begin{cases} \boldsymbol{F}(\dot{\boldsymbol{x}}, \boldsymbol{x}, t) = \boldsymbol{0}, \\ \boldsymbol{g}(\boldsymbol{x}, t) = \boldsymbol{0}, \end{cases}$$

where the second equation represents a set of algebraic equations that have no derivatives of the dependent variables (namely $\boldsymbol{x}$).

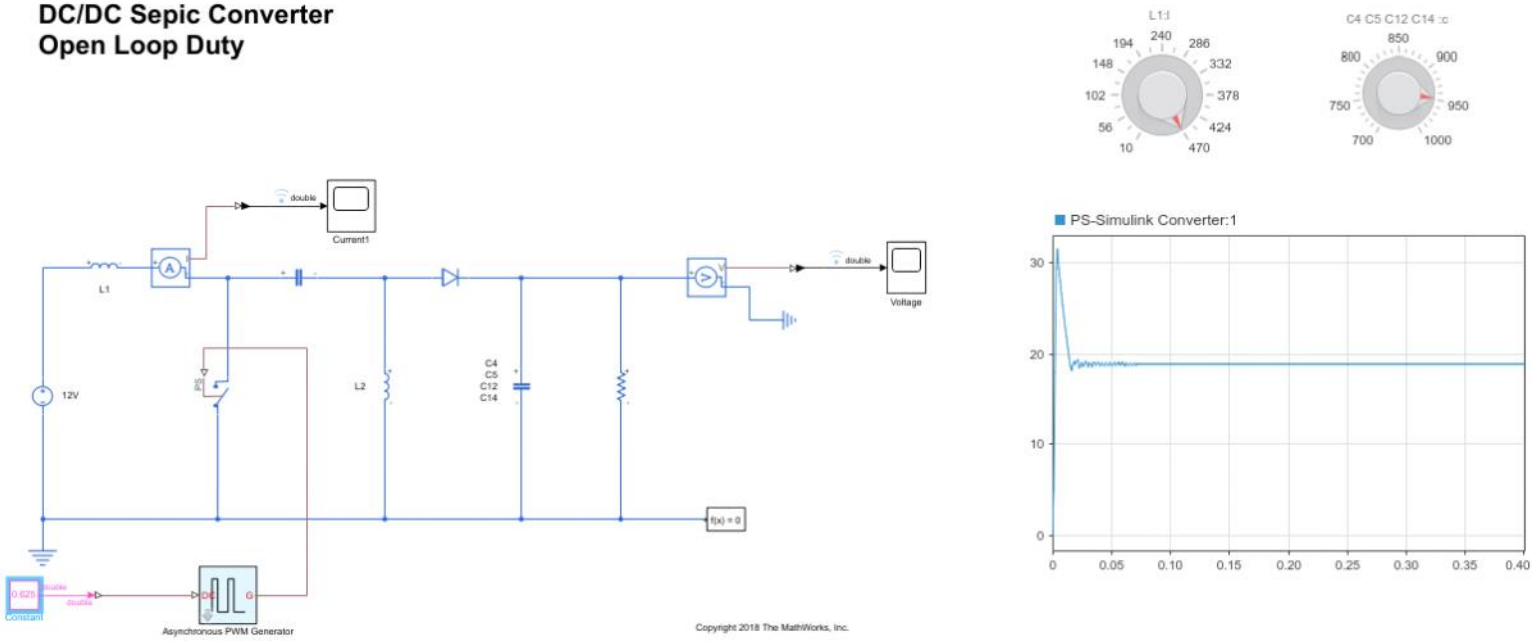
We will actually not be looking at any differential-algebraic equation here. Instead, we will be learning how to *use* DAE models through Simscape Electrical.

This section is based on an earlier version of MathWorks' webinar entitled "Part 4: Voltage Control Design", as part of their "How to Develop DC-DC Converter Control in Simulink" webinar series.

- Latest source files are available from File Exchange, but they have bugs.
- Old source files work better but are no longer available online except learnonline; we are using these in the following discussion.

2.1 Open loop

The `Sepic_new_openloop` model looks like this:

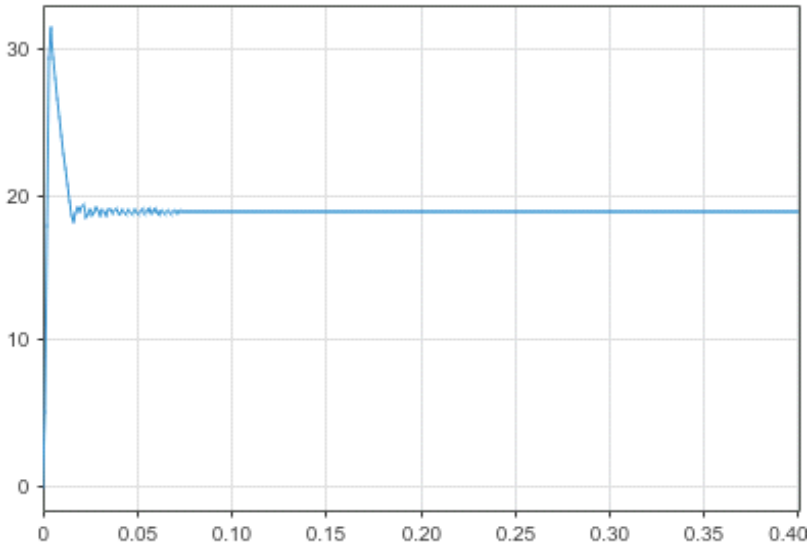


Notice the switch model is greatly simplified model (but not idealized, because it is specified to have an on-state resistance of 0.0011 Ω).

The SEPIC DC/DC converter has not been covered in this course yet, but here is a quick introduction.

- SEPIC is one of the classical converters (see Lecture 4).
- It has the same function as the buck-boost (see Lecture 5), i.e., it can produce a voltage that is either greater or less than the input, *except* without polarity reversal.
- Even the input-output voltage relationship is the same as that of the buck-boost, i.e.,
$$\frac{V_O}{V_I} = \frac{D}{1-D}, \qquad D = \frac{V_O}{V_I + V_O}.$$
- However, it does require one more inductor and one more capacitor than the buck-boost for a smoother output.

Suppose we are after an output of 20 V from an input of 12 V, the required duty cycle is $D = \frac{20}{20+12} = 0.625$. Using a fixed duty cycle of 0.625 in an open loop results in the following output voltage curve, that not only fails to track 20 V, but also has an excessive overshoot.



Let us fix this problem using a proportional-integral (PI) controller (see [knowledge base entry]).

Quiz

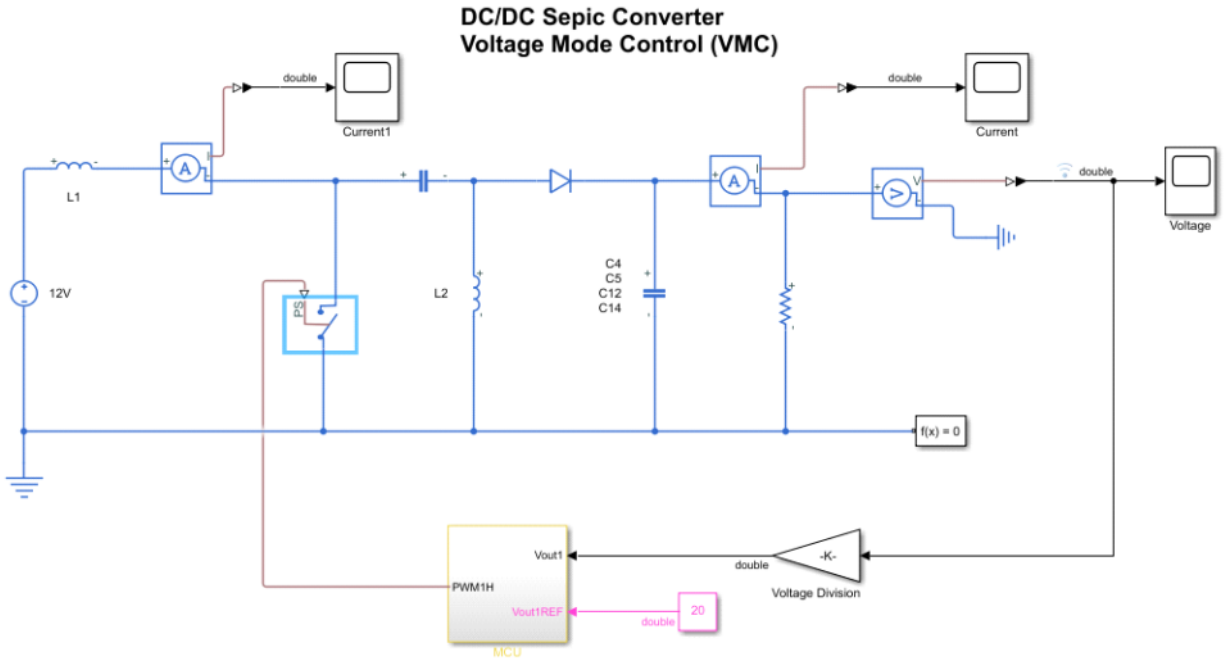
Let us have a quick refresher of the basic concepts of PID control:

- What are rise time, settling time and overshoot?
- The P term of a PI controller is for decreasing rise time and which other performance measure?
- What is the main purpose of using the I term, and what undesirable side effects does it tend to cause?

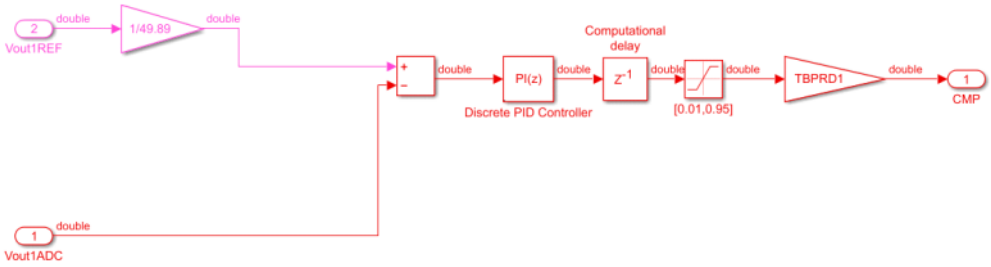
The next two subsections are dedicated to closed-loop control.

2.2 Closed loop (state space)

The `Sepic_new_closedloop_tune` model looks like this:

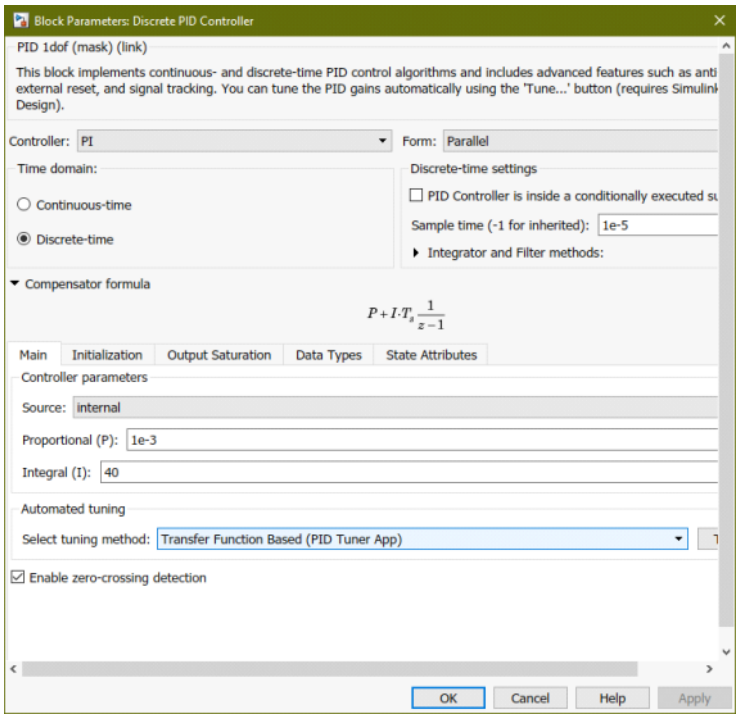


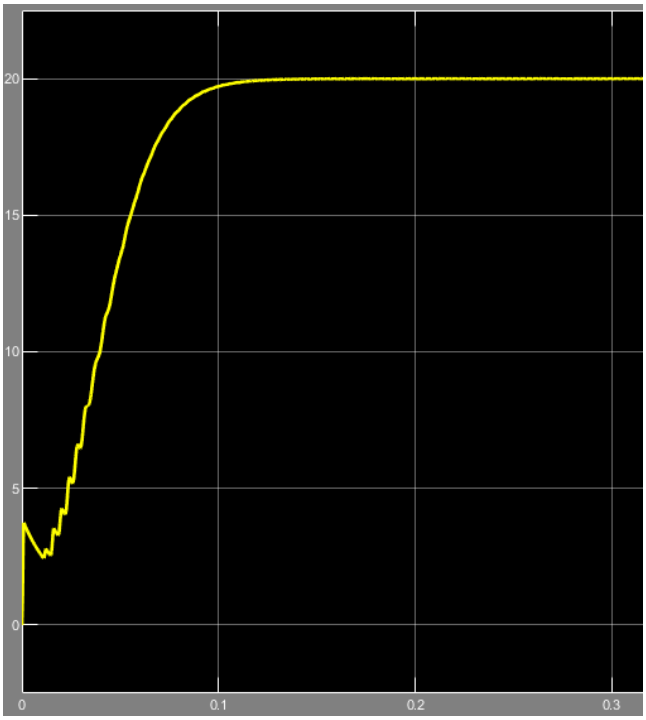
Within the MCU block, there is a Software block, and within the Software block, there is a Discrete PID Controller block:



The Discrete PID Controller block comes preconfigured as a PI controller, with some manually preset gain values ↓

Out of the box, tracking works, but the output response looks overdamped, and can use some improvement, through tuning of the PI controller ↓

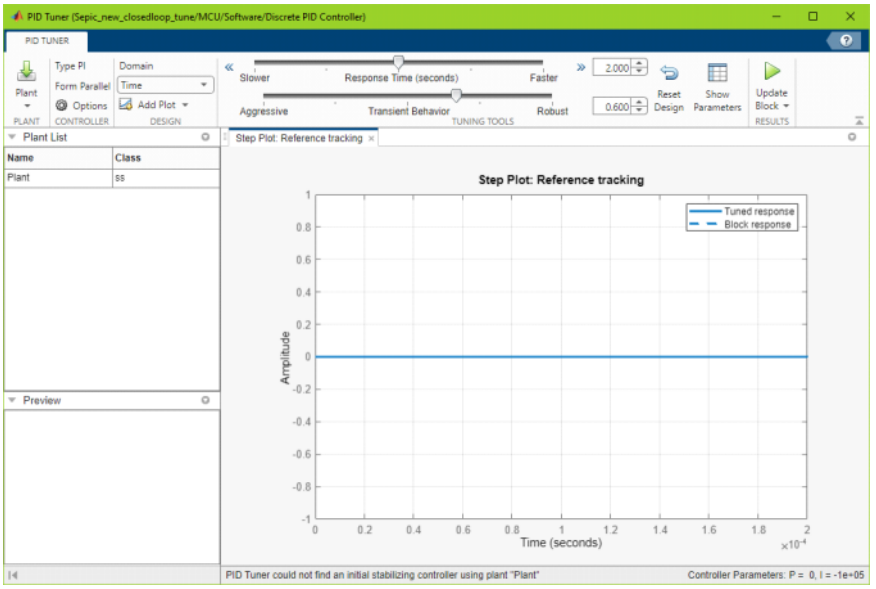


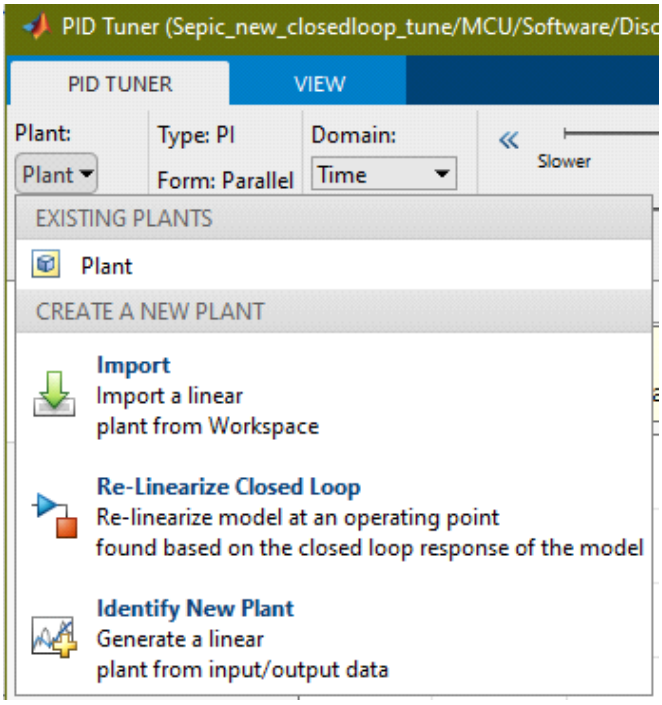


Clicking on the "Tune…" button in the dialog box above will open the **PID Tuner app** and trigger *linearization* of the circuit model.

However, linearization will fail → due to discontinuity in the circuit (caused by switching).

Older versions of the app displayed the message "Plant cannot be linearized" whereas the current version (R2022b) displays "PID Tuner could not find an initial stabilizing controller..."





To work around this expected failure, activate the **PID TUNER** tab and choose **Identify New Plant** — a process that will generate a linearized model based on simulation-based input/output data.

This is a mathematical process that falls under the umbrella of ***system identification***.

System identification is so widely useful, as attested by a 2017 industry survey [Sam17] that ranks system identification as the *third* most impactful control technology, after "PID control" and "Model predictive control".

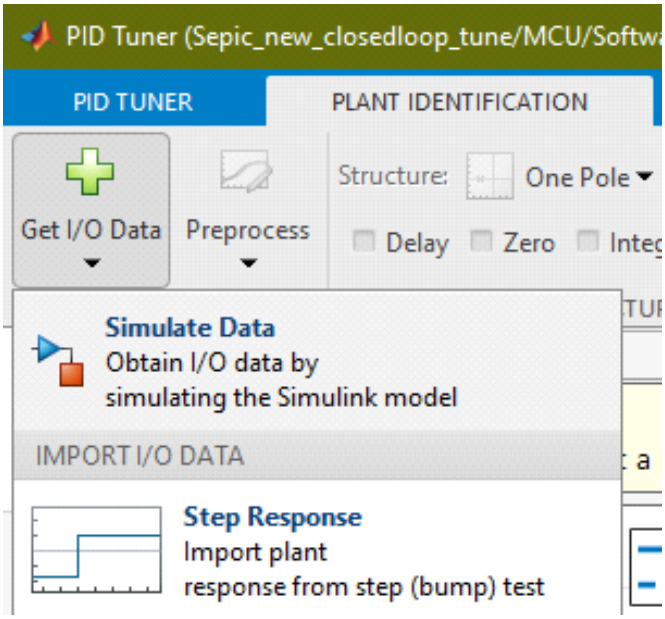
For the purpose of this course, let us treat system identification as a black-box algorithm.

After clicking **Identify New Plant**, the **PLANT IDENTIFICATION** tab will be activated.

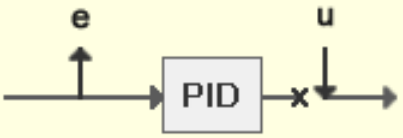
We need to run a simulation of the model to extract input and output (I/O) data, so go to the **PLANT IDENTIFICATION** tab, and choose **Simulate Data** to activate the **SIMULATE I/O DATA** tab.

There is no I/O data we can import anyway.

The process
- disconnects the PID controller from the feedback loop;
- requires the plant input $\boldsymbol{u}$ to be specified (see below);
- collects the control error $\boldsymbol{e}$, through simulation, as would have been seen by the controller.

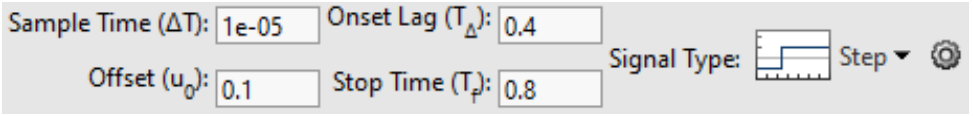


Specify **u**, the plant input.
Simulate the model to obtain **e**, the output of the plant seen by the controller.
After simulation, click Apply & Close to return to Plant Identification.



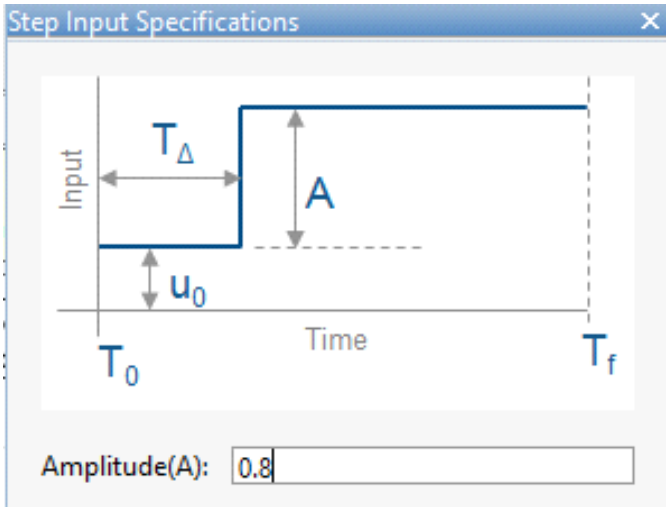
Note:
- control error = set-point − plant output.
- control error ≠ plant output.

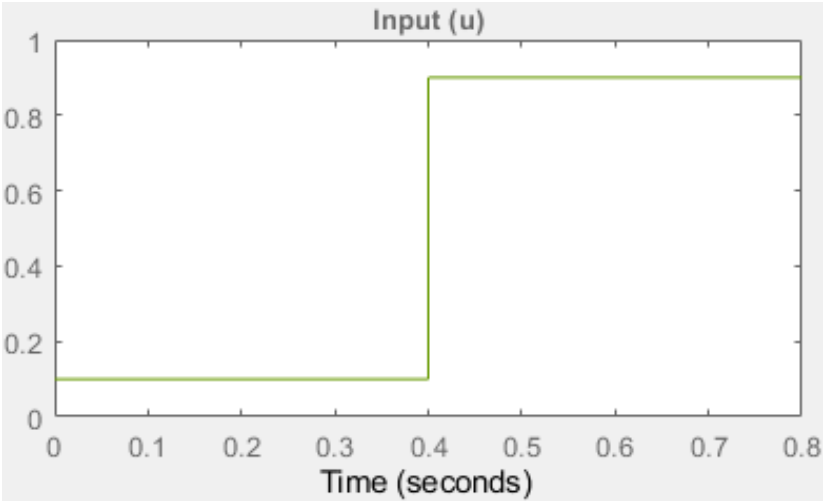
The set-point (i.e., duty ratio) already exists in the circuit model.



For specifying ***u***, the default onset lag is chosen to be half of the simulation time.

The default offset is 0, but let us specify the initial duty ratio as 0.1 and the final duty ratio as 0.9 (thus A=0.9-0.1=0.8 below), within the extremes.





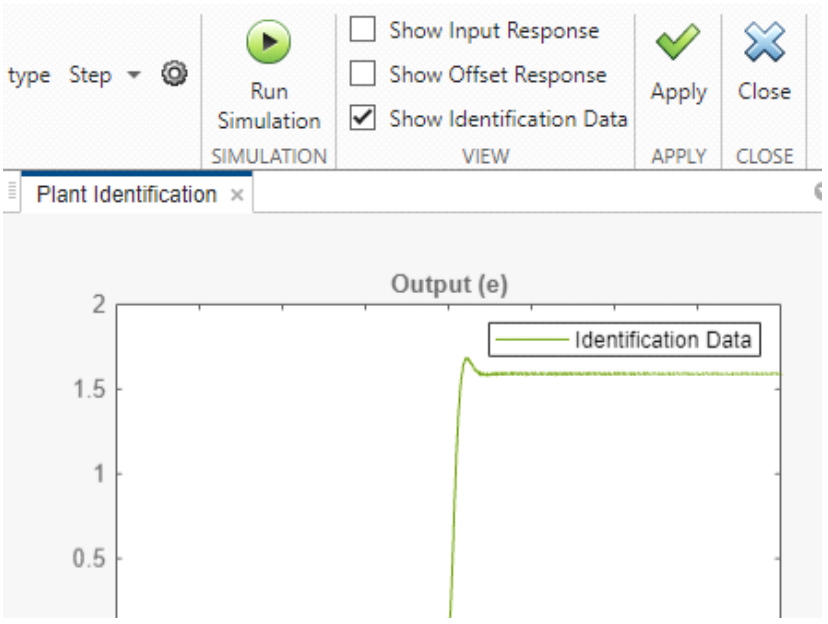
After specifying ***u***, click **Run Simulation**.

Output suggestive of a second-order (or higher) system can be expected →

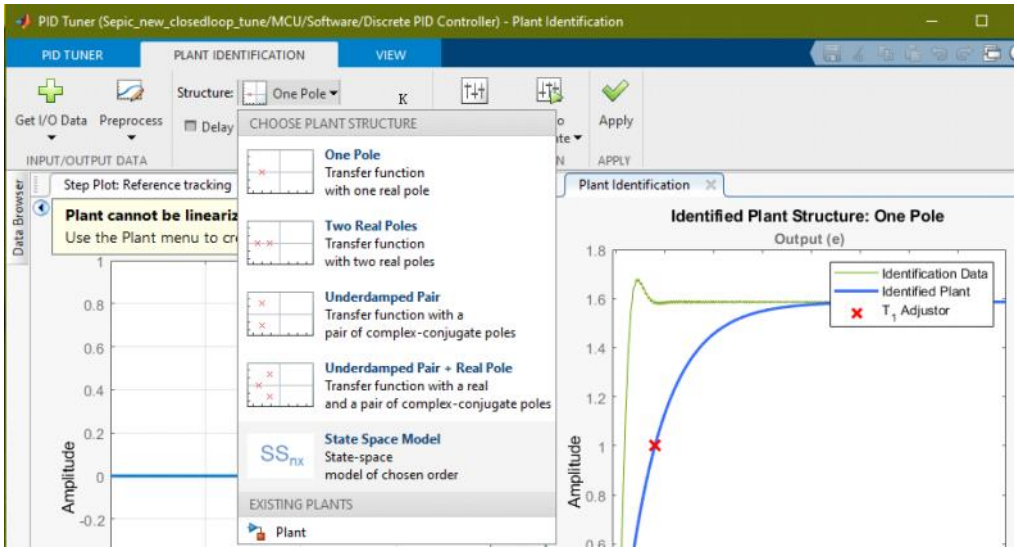
Click **Apply** to apply system identification to the I/O data.

Clicking **Close** in the **SIMULATE I/O DATA** tab will bring us back to the **PLANT IDENTIFICATION** tab.

The **PLANT IDENTIFICATION** tab shows we can treat the model to be estimated



The **PLANT IDENTIFICATION** tab shows we can treat the model to be estimated as either a *transfer function* (one to three poles) or a *state-space model*.

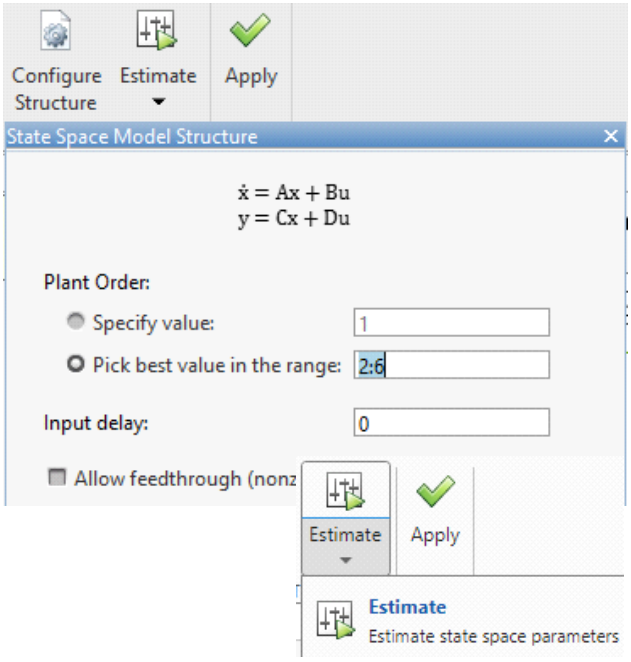


Quiz

Why can we not assume the model to be a first-order transfer function, or a second-order transfer function with two real poles?

A state-space model is more general, so let us try estimating a state-space model.

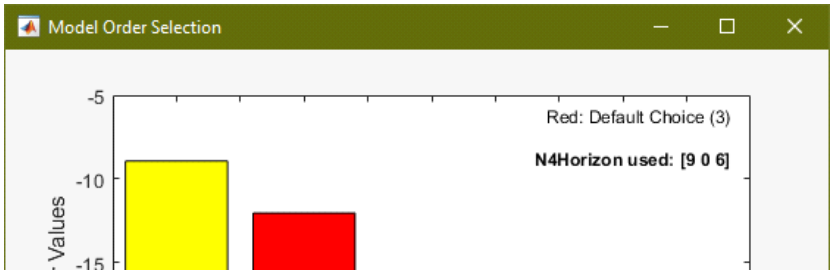
Click **Configure Structure**, and specify the state-space model structure as shown →



The **Plant Order** option "Pick best value in the range" means a range of state-space models will be estimated.

The input "2:6" means a 2nd-, a 3rd-, …, a 6th-order state-space model will be estimated — a total of 5 state-space models will be estimated.

Clicking the **Estimate** menu item triggers model estimation.



The plot shows the Hankel singular values for models of different orders; the lower the value, the better fitting the model is, but notice the trend of diminishing return.

diminishing return.

A default choice is suggested in the plot, but rather than the default choice, choose *the highest order above which there is no significant improvement in the fit*, to avoid overfitting.

The ***principle of parsimony*** advises against overfitting by preferring a model of lower order whenever its fit is good enough [DDS14].

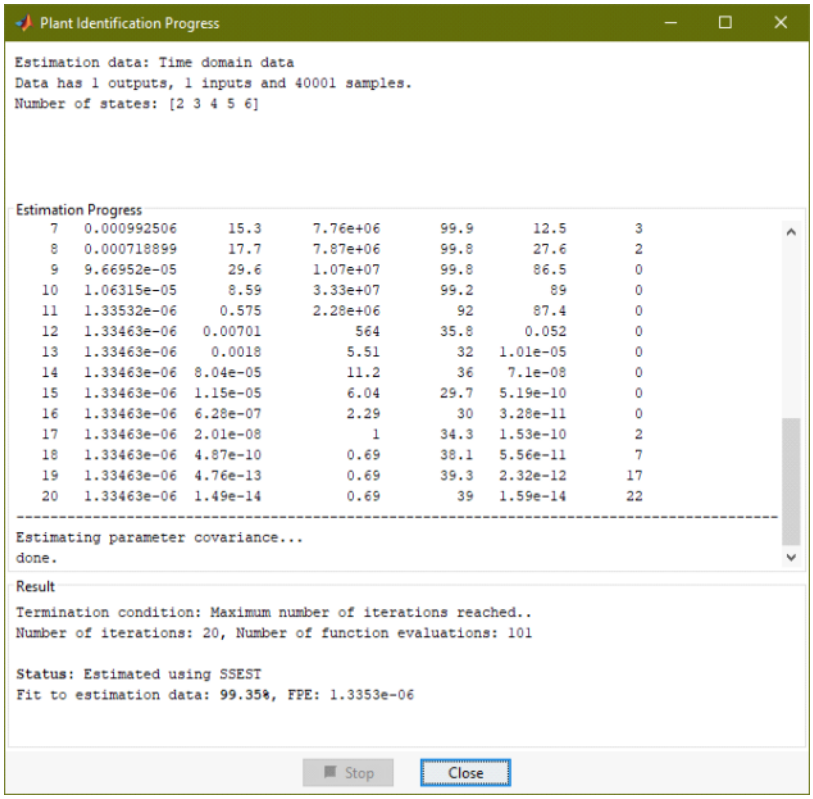
Seeing order 5 gives a more significant improvement to order 4, than order 6 does to order 5, we should choose order 5.

Choosing and applying order 5 will result in an estimated 5th-order state-space model with a fit of 99.35% →

The line "Status: Estimated using SSEST" suggests the estimator `ssest` was used.

Commensurate with the high percent fit of the model, the step response of the estimated model can be seen to overlap with the identification data ↓

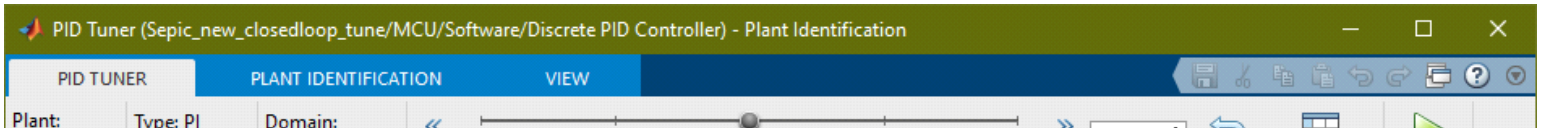
Click **Apply** to apply the estimated model.

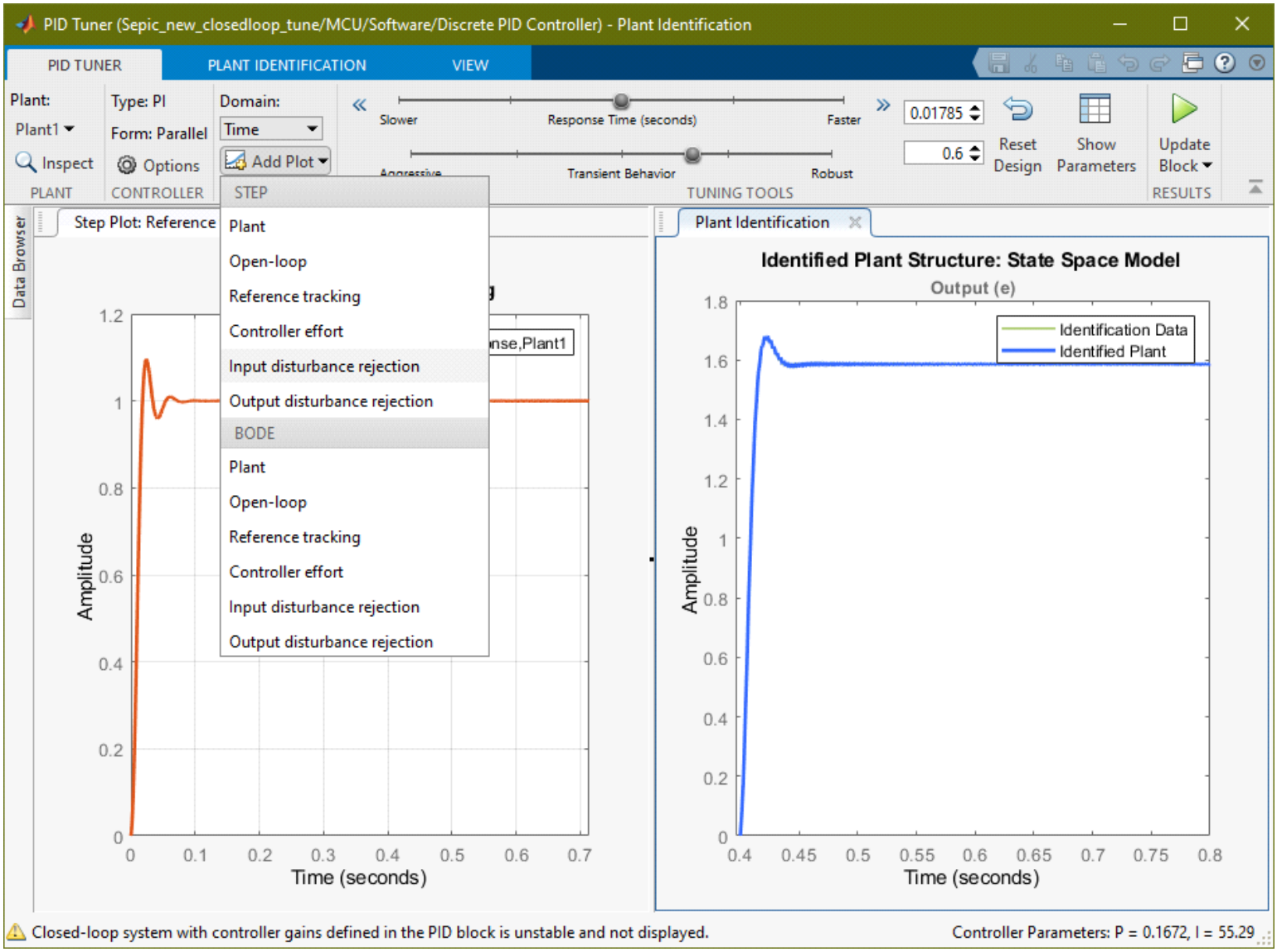


Clicking Apply will bring us back to the **PID TUNER** tab, but if not manually activate the tab.

By default, a **Reference tracking** step plot will be shown, but disturbance rejection is also of interest, so choose **Input disturbance rejection** as shown ↓

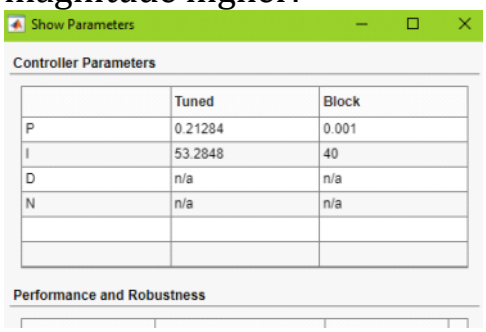
An example of input disturbance is the fluctuation in the input voltage.





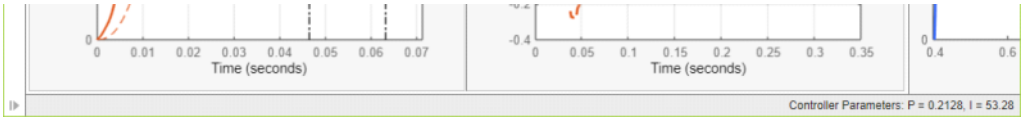
A reasonable PI design can be achieved by increasing the robustness of the transient behaviour.

Notice I gain is higher and P gain is two orders of magnitude higher:



Performance and Robustness

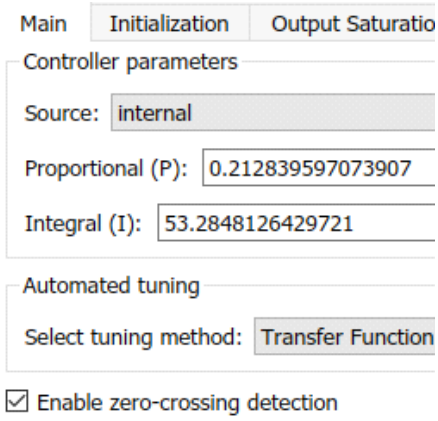
	Tuned	Block
Rise time	0.0123 seconds	0.0147 seconds
Settling time	0.0465 seconds	0.0632 seconds
Overshoot	4.95 %	12.7 %
Peak	1.05	1.13
Gain margin	7.68 dB @ 2.31e+03 rad/s	11.1 dB @ 199 rad/
Phase margin	65.4 deg @ 112 rad/s	57.2 deg @ 79.2 ra

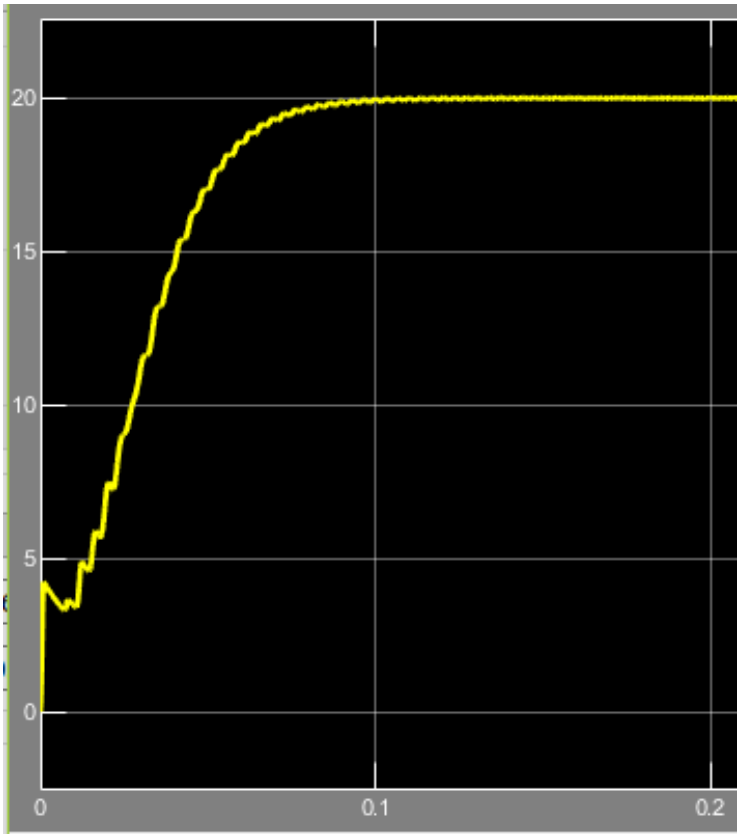


Note improved disturbance rejection ☝
🤙 and note reduced overshoot (less than 5%) and reduced settling time.

Click **Update Block** will update the Discrete PID Controller block with the new P and I gain values shown above.

- Warning: Delay is substantial in R2022a.



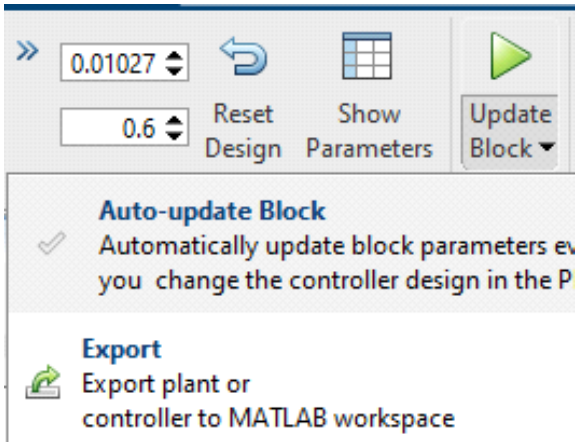


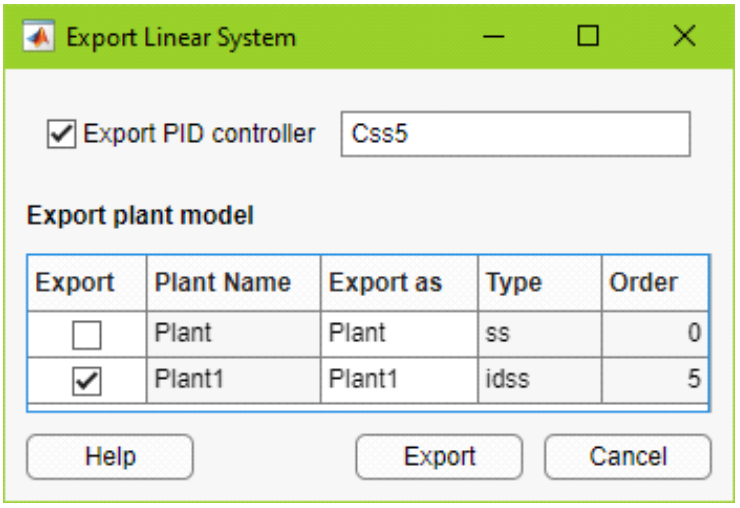
Running a fresh simulation of the model with the updated PID controller will show a slightly improved transient response.

For a more quantitative comparison, we can use the MATLAB function `stepinfo` to get the timing characteristics of the step responses *before* and *after* controller tuning.

The estimated state-space model and tuned controller gains can be saved using the **Export** menu item under **Update Block**.

If specified according to the screenshot below, the PID controller will be saved in a variable called `Css5`, while the estimated model will be saved in a variable called `Plant1`.





We should see what the 5th-order state-space model `Plant1` looks like:

```
>> Plant1

Plant1 =
  Continuous-time identified state-space model:
      dx/dt = A x(t) + B u(t) + K e(t)
       y(t) = C x(t) + D u(t) + e(t)

  A =
              x1          x2          x3          x4          x5
   x1     -174.4       -1063      -305.6        4840       -1691
   x2     -77.81       269.6       -2875   3.544e+04        9632
   x3      23.34        1955      -283.5        1819       -1945
   x4     -16.56       7.572         -12      -14.66   8.551e+04
   x5    0.07921      -2.086      -0.3467      -268.6   -3.91e+05

  B =
       Input (u)
   x1  3.347e+05
   x2  1.059e+06
   x3  2.117e+05
   x4  -5.44e+06
   x5  2.486e+07

  C =
                      x1          x2          x3          x4          x5
   Output (e)  -0.004232   0.0005997   -0.002232     -0.0685    -0.01472

  D =
                 Input (u)
   Output (e)            0
```

Some entries are many orders of magnitude larger than the others, e.g., the 5th column of the A matrix and the B matrix.

Such a model is called *unbalanced*, and is not robust — a controller designed against such a model tends to behave quite differently on the actual system, and a slight change in the actual system can have significant impact on control performance.

There are ways to balance a model, e.g., using the function `balreal`.

The function `balreal` uses Gramian matrices to transform a model from an unbalanced representation to a balanced representation.

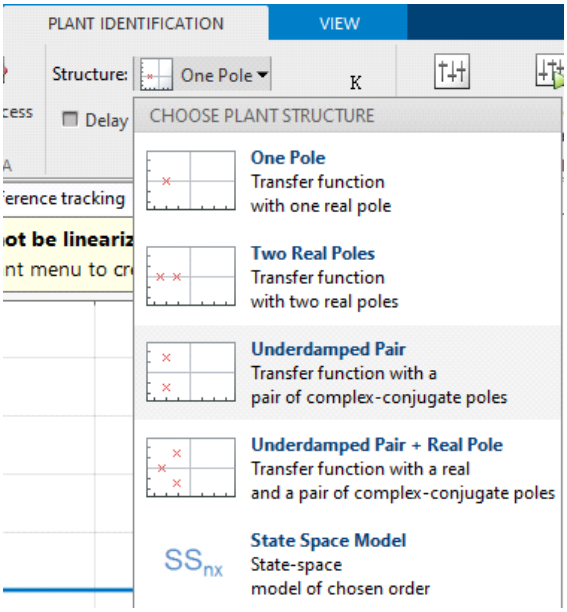
Using `balreal` requires us to break off from the online tuning method of PID Tuner.

2.3 Closed loop (transfer function)

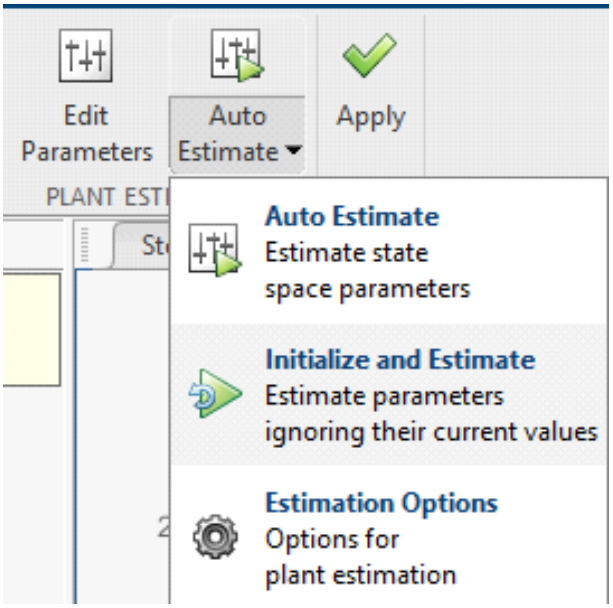
In the previous subsection, we estimated a state-space model for the system. An equivalent valid approach is estimating a transfer function.

In an earlier quiz, you would have got the idea that only two transfer function types recognized by PID Tuner are applicable, namely “Underdamped Pair” and “Underdamped Pair + Real Pole” applicable.

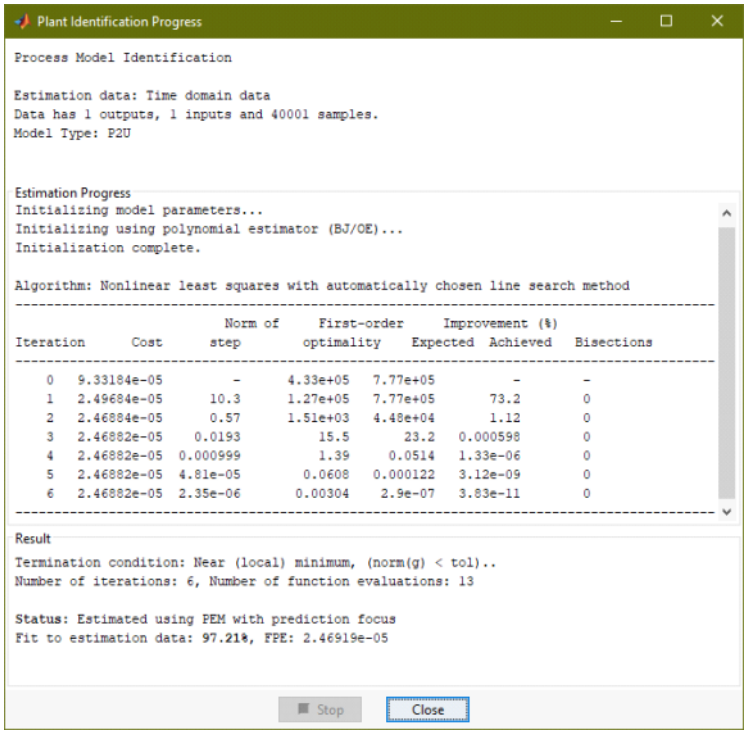
Let us try “Underdamped Pair” in the **PLANT IDENTIFICATION** tab. If the fit for this model is good enough, there is no need to add a third pole ↓

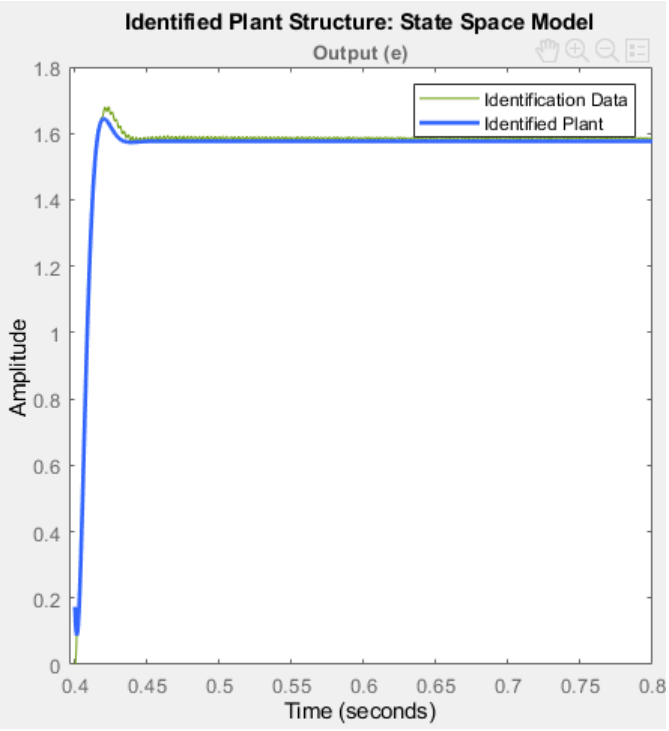


Subsequently choosing the **Initialize and Estimate** menu item will kick start the model estimation process ↓

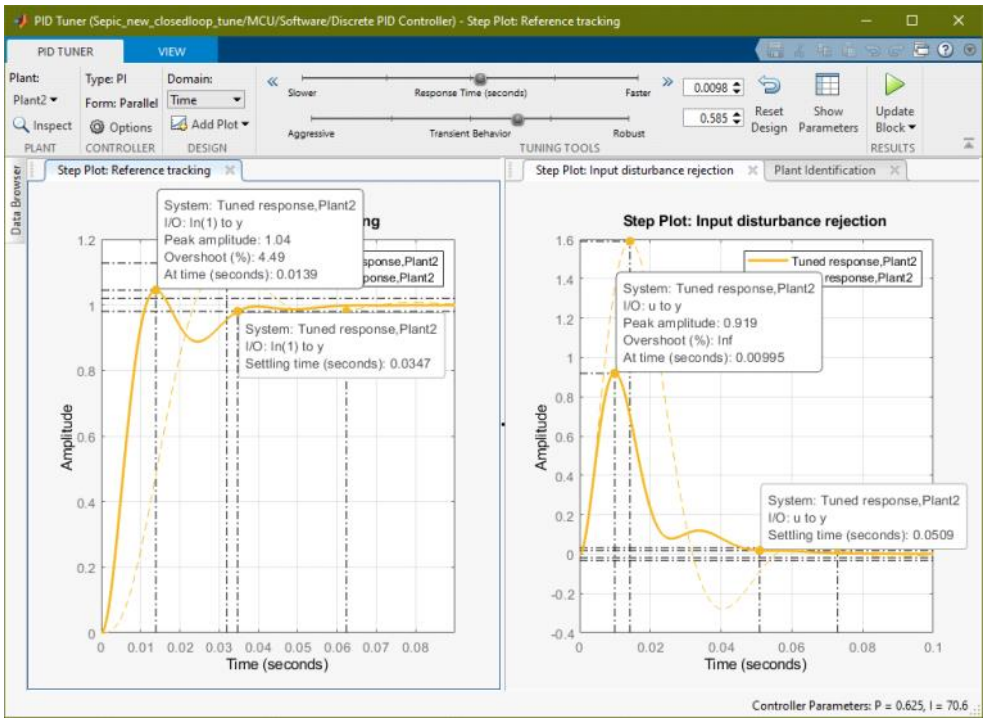


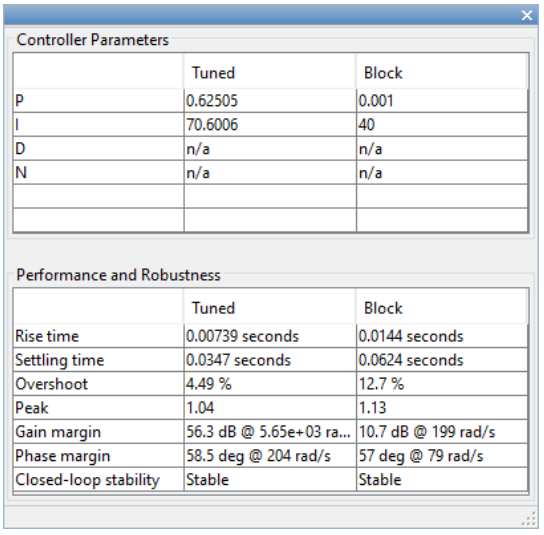
The resultant fit is above 97% — indication of a good model.





Click **Apply** and go back to the **PID TUNER** tab. The initial tuning result looks good enough, but can be slightly improved ↓





Controller Parameters

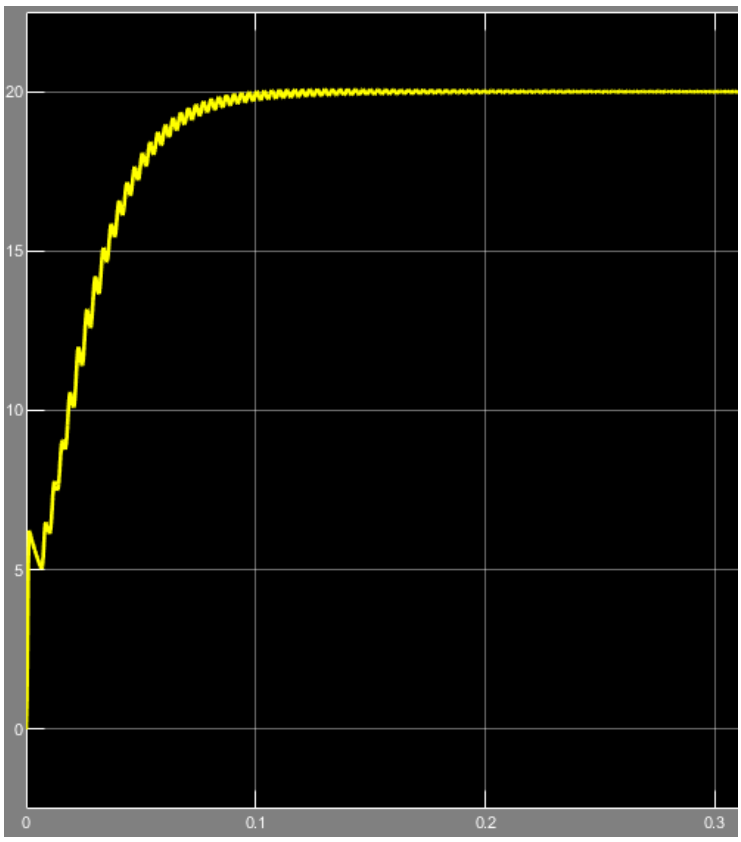
	Tuned	Block
P	0.62505	0.001
I	70.6006	40
D	n/a	n/a
N	n/a	n/a

Performance and Robustness

	Tuned	Block
Rise time	0.00739 seconds	0.0144 seconds
Settling time	0.0347 seconds	0.0624 seconds
Overshoot	4.49 %	12.7 %
Peak	1.04	1.13
Gain margin	56.3 dB @ 5.65e+03 ra...	10.7 dB @ 199 rad/s
Phase margin	58.5 deg @ 204 rad/s	57 deg @ 79 rad/s
Closed-loop stability	Stable	Stable

Unsurprisingly, the closed-loop response resembles the previous ↓

The tuning result and closed-loop responses are close to the previous results, even though we only used a 2nd-order model, rather than a 5th-order.



responses are close to the previous results, even though we only used a 2nd-order model, rather than a 5th-order.

This implies the circuit is dominated by 2nd-order dynamics.

The second-order transfer function can be studied after it is exported to the workspace:

```
>> Plant2

Plant2 =
Process model with transfer
function:
                    Kp
  G(s) = ---------------------
         1+2*Zeta*Tw*s+(Tw*s)^2

           Kp = 1.9827
           Tw = 0.0050478
         Zeta = 0.68674
```

$$G(s) = \frac{K}{s^2 + 2\zeta\omega_n s + \omega_n^2} = \frac{K/\omega_n^2}{\left(\frac{s}{\omega_n}\right)^2 + \frac{2\zeta}{\omega_n}s + 1}$$

Above, `Tw` is the inverse of the natural frequency, called the *natural period*.

`Zeta` (ζ) is of course the damping ratio.

The 2nd-order transfer function is more balanced than the 5th-order state-space model — considering this and tuning results, for this SEPIC converter, a 2nd-order transfer function is preferable.

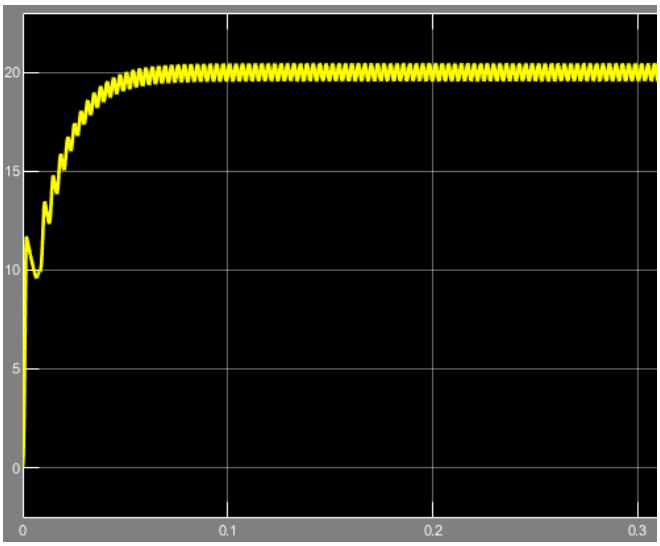
Recall the principle of parsimony — if a lower-order model provides just as good model accuracy as a higher-order model, the lower-order model should be preferred.

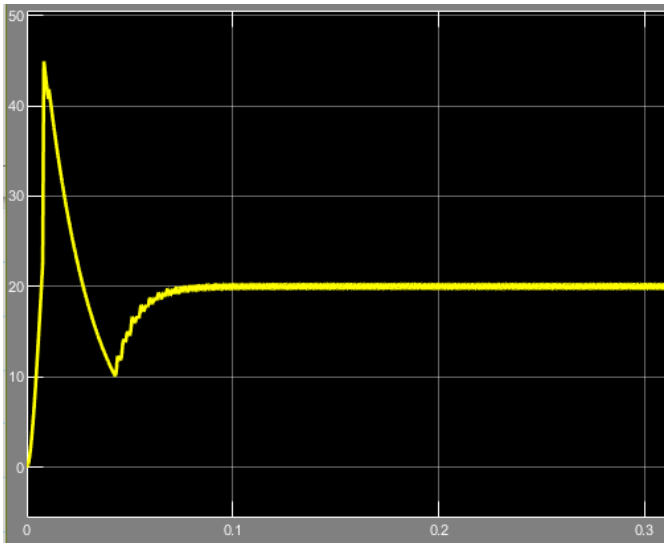
However, keep in mind both models are empirical and linearized models, so both are limited.

One more thing before we wrap up: the PI controller gains can be further *manually* fine-tuned.

Doubling the PI gains leads to a visibly shorter settling time ↓

However, increasing the gains 10-fold leads to an excessive overshoot suggestive of integrator windup ↓

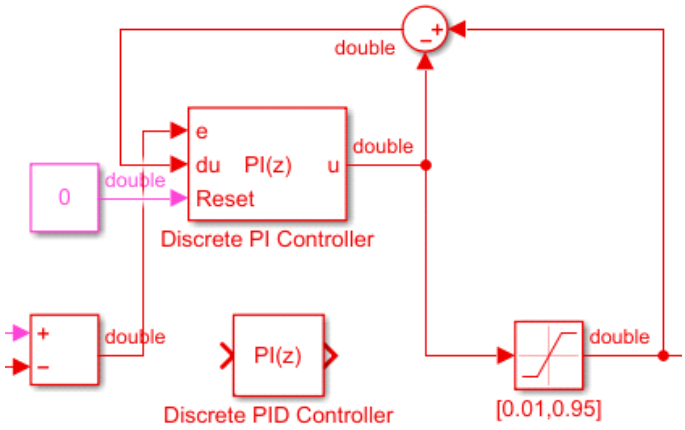


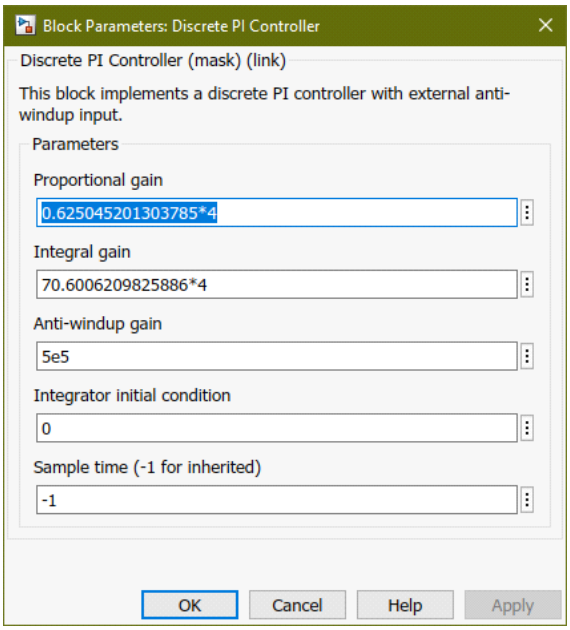


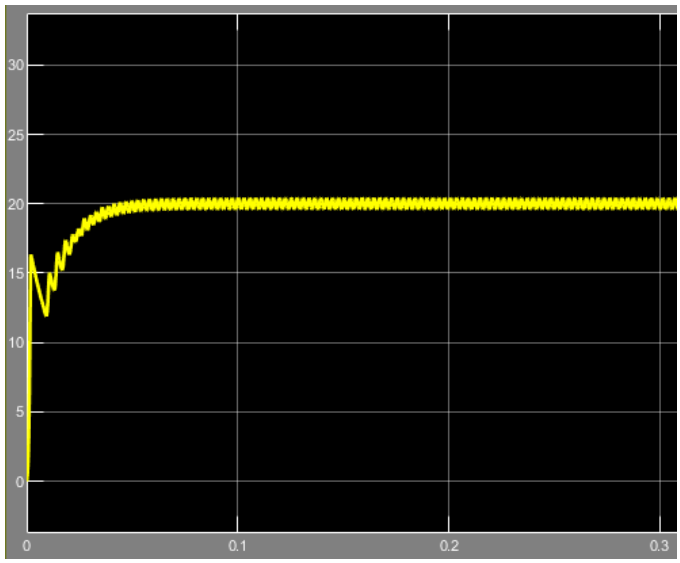
Increasing the PI gains 4-fold, but coupled with a [back-calculation anti-windup] of back-calculation gain 50k (yes, that high) leads to a slightly improved response than before.

The [knowledge base entry] on PID control has details on anti-windup.

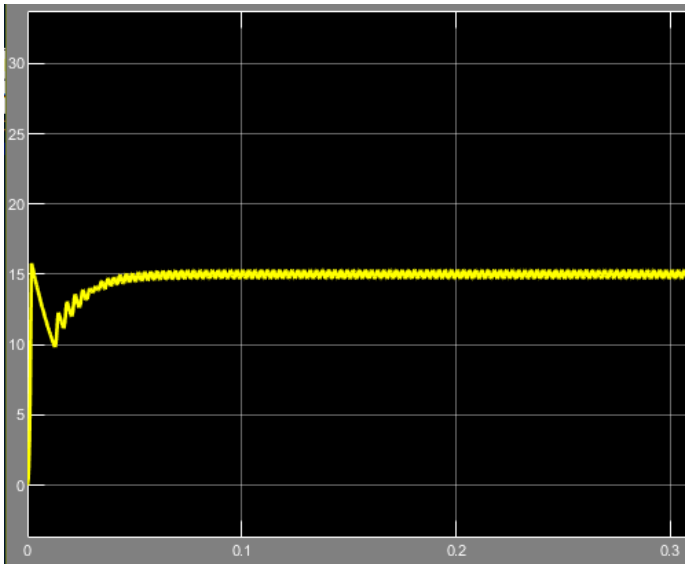
Considering the additional complexity, the smaller PI gains above maybe preferred.

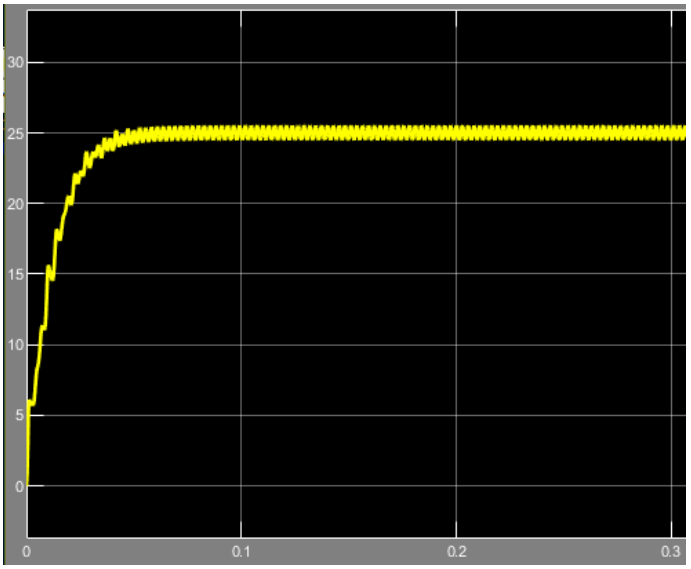






Changing the set-point to 15 V and then to 25 V gives us the step responses below:





In general, a unique linear controller is not likely to maintain control performance (dynamic response, stability, etc.) for the entire operating range [BMB14, Sect. 7.2].

Solutions include *gain scheduling*, i.e., adapting gain values to the operating point, which requires the gain values to be predesigned for optimal operation at select operating points prior to deployment.

Closed-loop system robustness and sensitivity to parameter variations is an open issue in power electronic converter control [BMB14, Sect. 7.2].

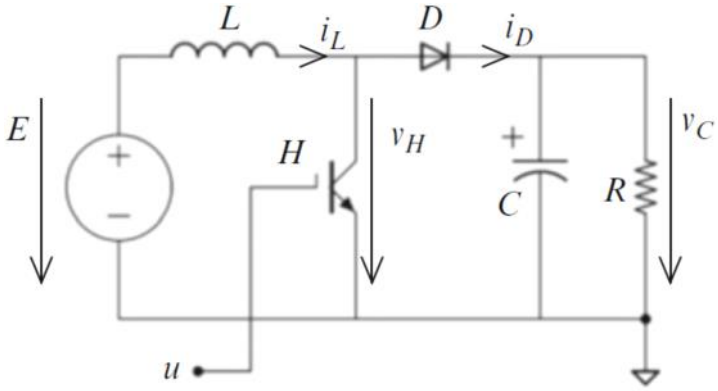
3. Switched model

Friday, 21 May, 2021 10:49 PM

Power electronics is defined by the switching action — whenever a switch turns on or off, the circuit topology changes, and the differential equations describing the circuit change accordingly.

- In other words, whenever a switch turns on or off, the model switches from one set of differential equations to another set of differential equations — hence the name ***switched model***.
- A set of differential equations in this context is called a ***configuration***.

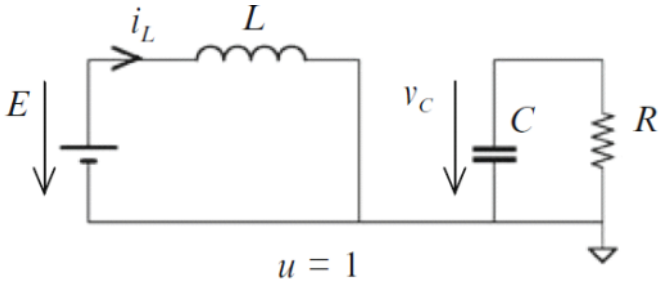
We shall introduce the method of switched models through an example based on the boost converter [BSB14, pp. 36-39].



One switch ⇒ two configurations.

Define $u = \begin{cases} 1, & \text{switch } H \text{ is turned on;} \\ 0, & \text{switch } H \text{ is turned off.} \end{cases}$

u is called the ***switching function***.



Configuration 1: When the switch is closed, i.e., $u = 1$.

KVL gives us

$$L\frac{di_L}{dt} = E \Rightarrow i_L' = \frac{E}{L},$$

using Lagrange's notation for differentiation.

KCL gives us

$$Cv_C' = -\frac{v_C}{R} \Rightarrow v_C' = -\frac{v_C}{RC}.$$

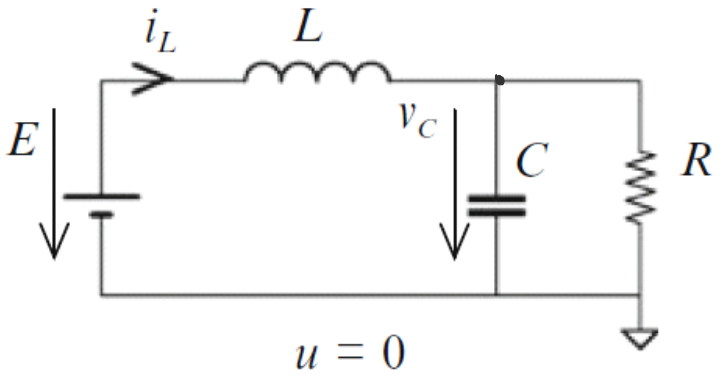
Define state vector as $\boldsymbol{x} = \begin{bmatrix} i_L & v_C \end{bmatrix}^{\mathsf{T}}$, then

state equation:

$$\underbrace{\begin{bmatrix} i_L' \\ v_C' \end{bmatrix}}_{\dot{\boldsymbol{x}}} = \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & -\frac{1}{RC} \end{bmatrix}}_{\text{System matrix}} \underbrace{\begin{bmatrix} i_L \\ v_C \end{bmatrix}}_{\boldsymbol{x}} + \underbrace{\begin{bmatrix} \frac{1}{L} \\ 0 \end{bmatrix}}_{\text{Input matrix}} \underbrace{E}_{\boldsymbol{u}}.$$

Configuration 2: When the switch is open, i.e., $u = 0$.

Configuration 2: When the switch is open, i.e., $u = 0$.



KVL gives us

$$Li_L' = E - v_C \Rightarrow i_L' = -\frac{v_C}{L} + \frac{E}{L}.$$

KCL gives us

$$i_L = Cv_C' + \frac{v_C}{R} \Rightarrow v_C' = \frac{i_L}{C} - \frac{v_C}{RC}.$$

State equation: $= \text{either } -\frac{1}{L} \text{ or } 0$

$$\begin{bmatrix} i_L' \\ v_C' \end{bmatrix} = \begin{bmatrix} 0 & -\frac{1}{L} \\ \frac{1}{C} & -\frac{1}{RC} \end{bmatrix} \begin{bmatrix} i_L \\ v_C \end{bmatrix} + \begin{bmatrix} \frac{1}{L} \\ 0 \end{bmatrix} E.$$

$-\frac{1}{L}(1-u) = -\frac{1}{L}$ for config 2; 0 for config 1

← The terms in red are unique to Configuration 2.

An obvious way to combine the state equations for Configurations 1 & 2 is to write the state equation as

$$\begin{bmatrix} i_L' \\ v_C' \end{bmatrix} = \begin{bmatrix} 0 & -\frac{1}{L}(1-u) \\ \frac{1}{C}(1-u) & -\frac{1}{RC} \end{bmatrix} \begin{bmatrix} i_L \\ v_C \end{bmatrix} + \begin{bmatrix} \frac{E}{L} \\ 0 \end{bmatrix},$$

which gives us the Configuration-1 state equation when $u = 1$, and the Configuration-2 state equation when $u = 0$.

The equation above *looks like* a linear state equation, but the u terms make the state equation *nonlinear*.

If we rewrite the state equation above as

$$\begin{bmatrix} i_L' \\ v_C' \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & -\frac{1}{L} \\ \frac{1}{C} & -\frac{1}{RC} \end{bmatrix}}_{\mathbf{A}} \begin{bmatrix} i_L \\ v_C \end{bmatrix} + \underbrace{\begin{bmatrix} 0 & \frac{1}{L} \\ -\frac{1}{C} & -\frac{1}{RC} \end{bmatrix}}_{\mathbf{B}} \begin{bmatrix} i_L \\ v_C \end{bmatrix} u + \underbrace{\begin{bmatrix} \frac{E}{L} \\ 0 \end{bmatrix}}_{\boldsymbol{d}},$$

the state equation now has the ***bilinear form***:

$$\dot{\boldsymbol{x}} = \mathbf{A}\boldsymbol{x} + \sum_{k=1}^{p} (\mathbf{B}_k \boldsymbol{x} + \boldsymbol{b}_k) u_k + \boldsymbol{d}.$$

← input in here

Switching function, not input

The bilinear form is useful because it can be linearized, and once linearized, it can be used for linear controller design [BSB14, Sect. 3.1.2].

If not linearized, it can be used for variable-structure controller design, e.g., sliding-mode controller design.

What we have done above is deriving a switched model, which is intuitive, but controller design is not straightforward.

We will continue with averaged models, which are not as intuitive, but they facilitate more straightforward linear controller designs.

4. Averaged model

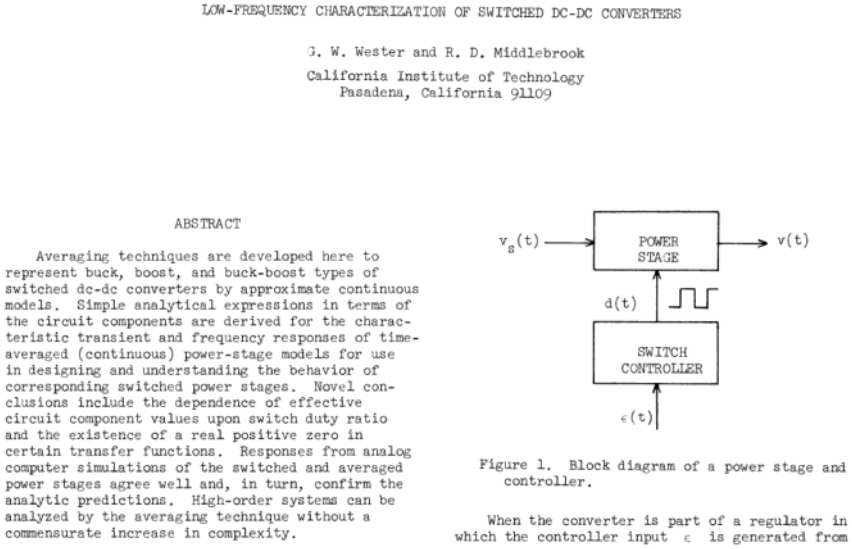
Saturday, 22 May, 2021 3:09 PM

The approach of averaged modeling dates back to the 1970s (see [BSB14, p. 56], [WM73]).

Averaged modeling focuses on capturing the low-frequency behavior of power electronic converters, while neglecting high-frequency variations due to circuit switching.

This is consistent with the fact that most converters come with filters for limiting current and/or voltage ripples.

Averaged models are *continuous-time* models. We will also be investigating the *discretized* versions of averaged models (maybe next year).



LOW-FREQUENCY CHARACTERIZATION OF SWITCHED DC-DC CONVERTERS

G. W. Wester and R. D. Middlebrook
California Institute of Technology
Pasadena, California 91109

ABSTRACT

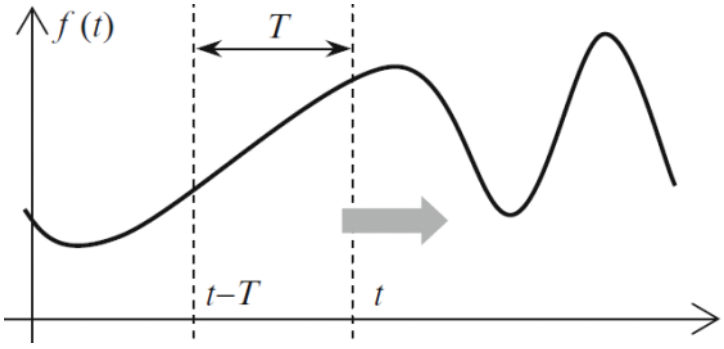
Averaging techniques are developed here to represent buck, boost, and buck-boost types of switched dc-dc converters by approximate continuous models. Simple analytical expressions in terms of the circuit components are derived for the characteristic transient and frequency responses of time-averaged (continuous) power-stage models for use in designing and understanding the behavior of corresponding switched power stages. Novel conclusions include the dependence of effective circuit component values upon switch duty ratio and the existence of a real positive zero in certain transfer functions. Responses from analog computer simulations of the switched and averaged power stages agree well and, in turn, confirm the analytic predictions. High-order systems can be analyzed by the averaging technique without a commensurate increase in complexity.

Figure 1. Block diagram of a power stage and controller.

When the converter is part of a regulator in which the controller input ϵ is generated from

Let us explore the nature of averaging before we derive any averaged model.

When applied in practice, ***sliding average*** (equivalently, *moving average* or *local average*) is more practical than overall average.



← Consider this not necessarily cyclic signal called $f(t)$, where T denotes the width of the sliding window.

The sliding average of $f(t)$, denoted $\langle f(t)\rangle_0(t)$, is defined as

$$\langle f(t)\rangle_0(t) \triangleq \frac{1}{T}\int_{t-T}^{t} f(\tau)d\tau\,. \tag{4.1}$$

- Subscript 0 denotes zeroth-order harmonic.
- In *generalized averaged models*, designed to

In this lecture, we will use the definition in (4.1), and omit the subscript for clarity.

A fundamental property of the sliding average is

$$\frac{d}{dt}\langle f(t)\rangle(t) = \left\langle \frac{d}{dt} f(t)\right\rangle(t). \tag{4.2}$$

- Subscript 0 denotes zeroth-order harmonic.
- In *generalized averaged models*, designed to handle averages of higher-order harmonics, the definition in (3.1) is extended to *kth-order sliding harmonic*, defined as

$$\langle f(t)\rangle_k(t) \triangleq \frac{1}{T}\int_{t-T}^{t} f(\tau)e^{-jk\omega\tau}d\tau\,.$$

- Some authors [EM20] define the sliding average with respect to the switching period, and use the notation $\langle f(t)\rangle_T$ where the subscript T denotes the switching period.

(4.2) $$\frac{d}{dt}\langle f(t)\rangle(t) = \left\langle \frac{d}{dt} f(t)\right\rangle(t).$$

i.e., time-differentiation and slide-averaging are commutative.

Therefore, the general state equation

$$\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}, \boldsymbol{u}, t)$$

is equivalent to

$$\frac{d}{dt}\langle \boldsymbol{x}(t)\rangle(t) = \langle f(\boldsymbol{x}, \boldsymbol{u}, t)\rangle(t).$$

Let us see how we can apply averaging to modeling the effect of switches.

The output voltage, S, of the adjacent circuit can be written as

$$S = E \cdot u,$$

because when $u = 1, S = E$, and when $u = 0, S = 0$.
When averaged,

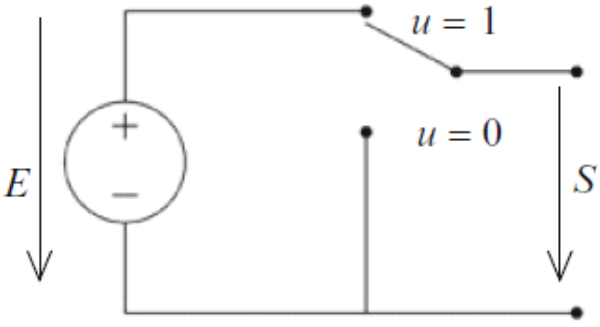
$$\langle S\rangle = \langle E \cdot u\rangle.$$

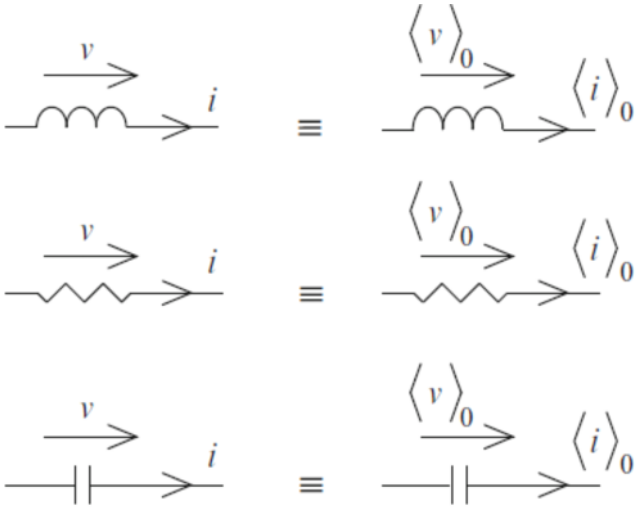
When E is constant,

$$\langle S\rangle = E \cdot \langle u\rangle.$$

When E is not constant but only has a small ripple,

$$\langle S\rangle \approx \langle E\rangle \cdot \langle u\rangle.$$





Applying (4.2), we get
For the inductor,

$$v(t) = L \cdot \frac{di(t)}{dt} \Rightarrow \langle v\rangle = L \cdot \frac{d}{dt}\langle i\rangle.$$

For the resistor,

$$v(t) = R \cdot i(t) \Rightarrow \langle v\rangle = R \cdot \langle i\rangle.$$

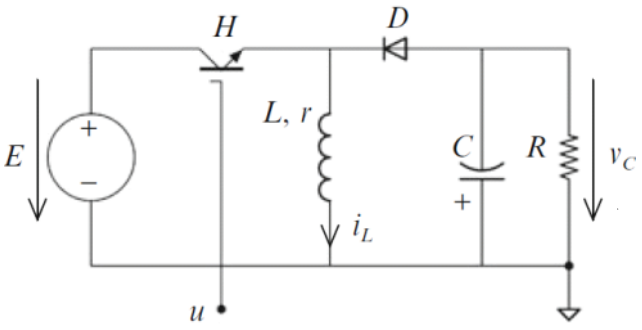
For the capacitor,

$$i(t) = C \cdot \frac{dv(t)}{dt} \Rightarrow \langle i\rangle = C \cdot \frac{d}{dt}\langle v\rangle.$$

Example 4.1

Example 4.1

After laying down some groundwork, we shall now introduce the method of averaged models through an example based on the buck-boost converter [BSB14, Sect. 4.6].



Define the switching function as

$$u = \begin{cases} 1, & \text{switch } H \text{ is turned on;} \\ 0, & \text{switch } H \text{ is turned off.} \end{cases}$$

KVL gives us

$$Li_L' + ri_L = Eu - v_C(1-u)$$
$$\Rightarrow i_L' = -\frac{r}{L}i_L - \frac{1}{L}v_C(1-u) + \frac{1}{L}Eu$$
$$\Rightarrow \langle i_L' \rangle = -\frac{r}{L}\langle i_L \rangle - \frac{1}{L}\langle v_C \rangle(1-\alpha) + \frac{1}{L}E\alpha,$$

where α is the duty ratio.

KCL gives us

$$Cv_C' = -\frac{v_C}{R}u + \left(i_L - \frac{v_C}{R}\right)(1-u)$$
$$\Rightarrow v_C' = \frac{1}{C}i_L(1-u) - \frac{1}{RC}v_C$$
$$\Rightarrow \langle v_C' \rangle = \frac{1}{C}\langle i_L \rangle(1-\alpha) - \frac{1}{RC}\langle v_C \rangle.$$

The differential equations above are nonlinear because of the products $\langle v_C \rangle(1-\alpha)$ and $\langle i_L \rangle(1-\alpha)$.

Before we can proceed with linear controller design, the differential equations need to be linearized.

4.1 Linearization

Linearization refers to the linear approximation of a model in the neighborhood of a chosen operating point using Taylor's series expansion.

Linearization is explained in detail in EEET 3046 Control Systems. The coverage here serves only as a revision.

Consider the continuous-time state-space equations:

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}\big(\boldsymbol{x}(t), \boldsymbol{u}(t)\big) = \begin{bmatrix} f_1\big(\boldsymbol{x}(t), \boldsymbol{u}(t)\big) \\ \vdots \\ f_n\big(\boldsymbol{x}(t), \boldsymbol{u}(t)\big) \end{bmatrix},$$
$$\boldsymbol{y}(t) = \boldsymbol{g}\big(\boldsymbol{x}(t), \boldsymbol{u}(t)\big),$$

and let $\boldsymbol{u}_e(t)$ be an input that leads the system to equilibrium state/point $\boldsymbol{x}_e$, i.e.,

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}\big(\boldsymbol{x}_e(t), \boldsymbol{u}_e(t), t\big) = \boldsymbol{0},$$

where $\boldsymbol{f}$ is a nonlinear but linearizable function.

Equilibrium state (x_e)

… is by definition the state in which system has zero $\dot{\boldsymbol{x}}$. If not explicitly specified, the system input $\boldsymbol{u}$ is assumed to be zero when the system is in its equilibrium state.

The linearized state equation is given by

(4.3) $$\Delta\dot{\boldsymbol{x}}(t) = \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right|_{\substack{\boldsymbol{x}=\boldsymbol{x}_e \\ \boldsymbol{u}=\boldsymbol{u}_e}} \Delta\boldsymbol{x}(t) + \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\right|_{\substack{\boldsymbol{x}=\boldsymbol{x}_e \\ \boldsymbol{u}=\boldsymbol{u}_e}} \Delta\boldsymbol{u}(t),$$

where $\Delta\boldsymbol{x}(t) = \boldsymbol{x}(t) - \boldsymbol{x}_e$, $\Delta\boldsymbol{u}(t) = \boldsymbol{u}(t) - \boldsymbol{u}_e$, $\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}$ and $\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}$ are Jacobians.

- A Jacobian is the multivariable equivalent of a scalar derivative, defined as

(4.4) $$\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}} \triangleq \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}.$$

- We follow [Kel13] to call $\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}$ and $\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}$ the *state Jacobian* and *input Jacobian* respectively, although these terms are curiously uncommon.

Rejoining the example of the buck-boost converter from earlier, let us see how we can linearize the equations in Example 3.1.

Example 4.2

Define the state vector as $\boldsymbol{x} = \begin{bmatrix} \langle i_L \rangle & \langle v_C \rangle \end{bmatrix}^\top$, then

$$\langle i_L' \rangle = f_1(\boldsymbol{x}, \alpha) = -\frac{r}{L}\langle i_L \rangle - \frac{1}{L}\langle v_C \rangle(1-\alpha) + \frac{1}{L}E\alpha,$$

$$\langle v_C' \rangle = f_2(\boldsymbol{x}, \alpha) = \frac{1}{C}\langle i_L \rangle(1-\alpha) - \frac{1}{RC}\langle v_C \rangle.$$

Next comes the important realisation that E can be treated as a constant — with any change to it treated as a disturbance — while α should be treated as an input.

The equilibrium states are such that

$$-r\langle i_L \rangle_e - \langle v_C \rangle_e (1-\alpha_e) + E\alpha_e = 0,$$

$$\langle i_L \rangle_e (1-\alpha_e) - \frac{1}{R}\langle v_C \rangle_e = 0 \Rightarrow \langle v_C \rangle_e = R\langle i_L \rangle_e (1-\alpha_e).$$

Substituting $\langle v_C \rangle_e = R\langle i_L \rangle_e (1-\alpha_e)$ into

$$-r\langle i_L \rangle_e - \langle v_C \rangle_e (1-\alpha_e) + E\alpha_e = 0 \Rightarrow -r\langle i_L \rangle_e - R\langle i_L \rangle_e (1-\alpha_e)^2 + E\alpha_e = 0$$

$$\Rightarrow \langle i_L \rangle_e = \frac{E\alpha_e}{r + R(1-\alpha_e)^2}.$$

Thus given the equilibrium duty ratio α_e, the equilibrium $\langle i_L \rangle$ and $\langle v_C \rangle$ are

$$\langle i_L \rangle_e = \frac{E\alpha_e}{r + R(1-\alpha_e)^2}, \qquad \langle v_C \rangle_e = \frac{RE\alpha_e(1-\alpha_e)}{r + R(1-\alpha_e)^2}.$$

The Jacobians are readily derivable:

$$\left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right|_{\substack{\boldsymbol{x}=\boldsymbol{x}_e \\ \alpha=\alpha_e}} = \begin{bmatrix} -\frac{r}{L} & \frac{\alpha_e - 1}{L} \\ \frac{1-\alpha_e}{C} & -\frac{1}{RC} \end{bmatrix}, \qquad \left.\frac{\partial \boldsymbol{f}}{\partial \alpha}\right|_{\substack{\boldsymbol{x}=\boldsymbol{x}_e \\ \alpha=\alpha_e}} = \begin{bmatrix} \frac{\langle v_C \rangle_e}{L} + \frac{E}{L} \\ -\frac{\langle i_L \rangle_e}{C} \end{bmatrix}.$$

Therefore, the linearized state equation is

$$\Delta\dot{\boldsymbol{x}} = \begin{bmatrix} -\frac{r}{L} & \frac{\alpha_e - 1}{L} \\ \frac{1-\alpha_e}{C} & -\frac{1}{RC} \end{bmatrix} \Delta\boldsymbol{x} + \begin{bmatrix} \frac{\langle v_C \rangle_e}{L} + \frac{E}{L} \\ -\frac{\langle i_L \rangle_e}{C} \end{bmatrix} \Delta\alpha.$$

Verify the above in [BMB14, (4.36)-(4.38)].

The manual derivation above can be efficiently computerized using the Symbolic Math Toolbox. See accompanying Live Script for code.

A linearized state equation can be converted to a transfer function for linear controller design methods such as those based on root locus and frequency response.

Given linearized state-space equations for a SISO model:

$$\Delta\dot{\boldsymbol{x}} = \mathbf{A}\Delta\boldsymbol{x} + \mathbf{B}\Delta\alpha,$$
$$\Delta y = \mathbf{C}\Delta\boldsymbol{x},$$

the transfer function is given by

(4.5) $G(s) = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}$.

References

Friday, 14 May 2021 4:01 PM

[BMB14] S. Bacha, I. Munteanu, and A. I. Bratcu, *Power electronic converters modeling and control: with case studies*, *Advanced Textbooks in Control and Signal Processing*, Springer-Verlag London, 2014, more than 250 Google Scholar citations. https://doi.org/10.1007/978-1-4471-5478-5.

[CLL05] R. Y. Chen, R. L. Lin, T. J. Liang, J. F. Chen and K. C. Tseng, *Current-fed full-bridge boost converter with zero current switching for high voltage applications*, Fourtieth IAS Annual Meeting. Conference Record of the 2005 Industry Applications Conference, 2005, pp. 2000-2006 Vol. 3, doi: 10.1109/IAS.2005.1518722. Available at https://ieeexplore-ieee-org.access.library.unisa.edu.au/abstract/document/1518722

[CM10] F. Chen and D. Maksimović, *Digital Control for Improved Efficiency and Reduced Harmonic Distortion Over Wide Load Range in Boost PFC Rectifiers*, IEEE Transactions on Power Electronics, 25(10):2683-2692, Oct. 2010, doi: 10.1109/TPEL.2010.2050702. Available at https://ieeexplore.ieee.org/abstract/document/5466192

[DDS14] R. Doraiswami, C. Diduch, M. Stevenson, *Identification of Physical Systems: Applications to Condition Monitoring, Fault Diagnosis, Soft Sensor and Controller Design*, Wiley, 2014. URL: http://onlinelibrary.wiley.com.access.library.unisa.edu.au/doi/10.1002/9781118536483.ch7/summary

[EM20] R. W. Erickson and D. Maksimović, *Fundamentals of Power Electronics*, 3rd ed., Springer Nature Switzerland AG, 2020. Available at https://doi-org.access.library.unisa.edu.au/10.1007/978-3-030-43881-4

[Har11] D. W. Hart, *Power Electronics*, international ed., McGraw-Hill Education, 2011.

[JC08] S. Jalbrzykowski and T. Citko, *Current-Fed Resonant Full-Bridge Boost DC/AC/DC Converter*, 55(3):1198-1205, Mar 2008. More than 100 Google Scholar citations. Available at https://ieeexplore-ieee-org.access.library.unisa.edu.au/document/4401139

[JJ07] Y. Jang and M. M. Jovanovic, *Interleaved Boost Converter With Intrinsic Voltage-Doubler Characteristic for Universal-Line PFC Front End*, IEEE Transactions on Power Electronics, 22(4):1394-1401, July 2007, doi: 10.1109/TPEL.2007.900502. Available at https://ieeexplore-ieee-org.access.library.unisa.edu.au/document/4267757

[Kaz16] M. K. Kazimierczuk, *Pulse-width modulated DC-DC power converters*, 2nd ed., John Wiley & Sons, 2016, more than 1000 Google Scholar citations. https://doi.org/10.1002/9781119009597.

[Kel13] A. Kelly, *Mobile Robotics: Mathematics, Models, and Methods*, Cambridge University Press, 2013.

[Kre15] P. T. Krein, *Elements of Power Electronics*, 2nd ed., Oxford University Press, 2015.

[LY18] F.L. Luo and H. Ye, *Power Electronics: Advanced Conversion Technologies*, 2nd edition, CRC Press, 2018. Available at https://www.taylorfrancis.com/books/power-electronics-fang-lin-luo-hong-ye/10.1201/9781315186276

[MZ07] D. Maksimovic and R. Zane, *Small-Signal Discrete-Time Modeling of Digitally Controlled PWM Converters*, IEEE Transactions on Power Electronics, 22(6):2552-2556, Nov. 2007, doi: 10.1109/TPEL.2007.909776. Available at https://ieeexplore-ieee-org.access.library.unisa.edu.au/abstract/document/4371555

[Ons20] ON Semiconductor, *6.6 kW On Board EV Charger Reference Design*, reference design TND6320/D, July 2020. Available at https://www.onsemi.com/pub/Collateral/TND6320-D.PDF

[Ras14] M. H. Rashid, *Power Electronics: Devices, Circuits, and Applications*, Pearson Education Limited, 2014.

[Sam17] T. Samad, *A Survey on Industry Impact and Challenges Thereof* [*Technical Activities*], IEEE Control Systems Magazine, 37(1):17-18, Feb. 2017. Available at https://ieeexplore-ieee-org.access.library.unisa.edu.au/document/7823045

[SS06] H. Sira-Ramírez and R. Silva-Ortigoza, *Control Design Techniques in Power Electronics Devices*, Springer-Verlag London Limited 2006. Available at https://doi-org.access.library.unisa.edu.au/10.1007/1-84628-459-7

[Trz16] A. M. Trzynadlowski, *Introduction to modern power electronics*, 3rd ed., John Wiley & Sons, 2016.

[WM73] G. W. Wester and R. D. Middlebrook, *Low-Frequency Characterization of Switched dc-dc Converters*, IEEE Transactions on Aerospace and Electronic Systems, vol. AES-9, no. 3, pp. 376-385, May 1973, doi: 10.1109/TAES.1973.309723.

[Xia21] W. Xiao, *Power Electronics Step-By-Step: Design, Modeling, Simulation, and Control*, McGraw Hill, 2021.