Modelling and Simulation of a DC Motor Drive

1 Introduction

A simulation model of the DC motor drive will be built using the Matlab/Simulink environment. This assignment aims to familiarise you with basic features of Simulink and to demonstrate modelling approaches applicable to electric drives. After this assignment, you will be able to:

1. Model and simulate a DC motor drive in the Matlab/Simulink environment;
2. Tune the current and speed controllers using a model-based approach;
3. Explain effects of the pulse-width modulation (PWM) on control performance.

The model of the DC motor is first built and tested in Section 2. Detailed step-by-step instructions are given for creating the model. In Section 3, the motor model is augmented with models for the DC-DC converter and PWM. In Section 4, the current and speed controllers are added to the model.

A report is to be written on this assignment in groups of two (or alone). Submit your report as a PDF file to the MyCourses portal (`mycourses.aalto.fi`) no later than on *Wednesday, 25.11.2020, at 16:00*.

In your report, answer *briefly* the questions given inside this kind of framed boxes. The report should be clearly and consistently written. The requested figures describing the models and simulation results should be included in the report. Submit also the requested Simulink models to MyCourses. These models will be used to check that you have built the models yourself.

Guidance is available at the Teams platform on

- Wednesday, 4.11.2020, at 10:15–12:00
- Wednesday, 18.11.2020, at 10:15–12:00

The assignment will be graded on a scale of 0...20 (two points per problem). You are encouraged to discuss with other students but copying solutions from other groups is not allowed! The reports and models will be checked for plagiarism.

2 DC Motor Model

2.1 Dynamic Equations

First, a simulation model of the permanent-magnet DC motor will be built. The following state equations are taken as a starting point:

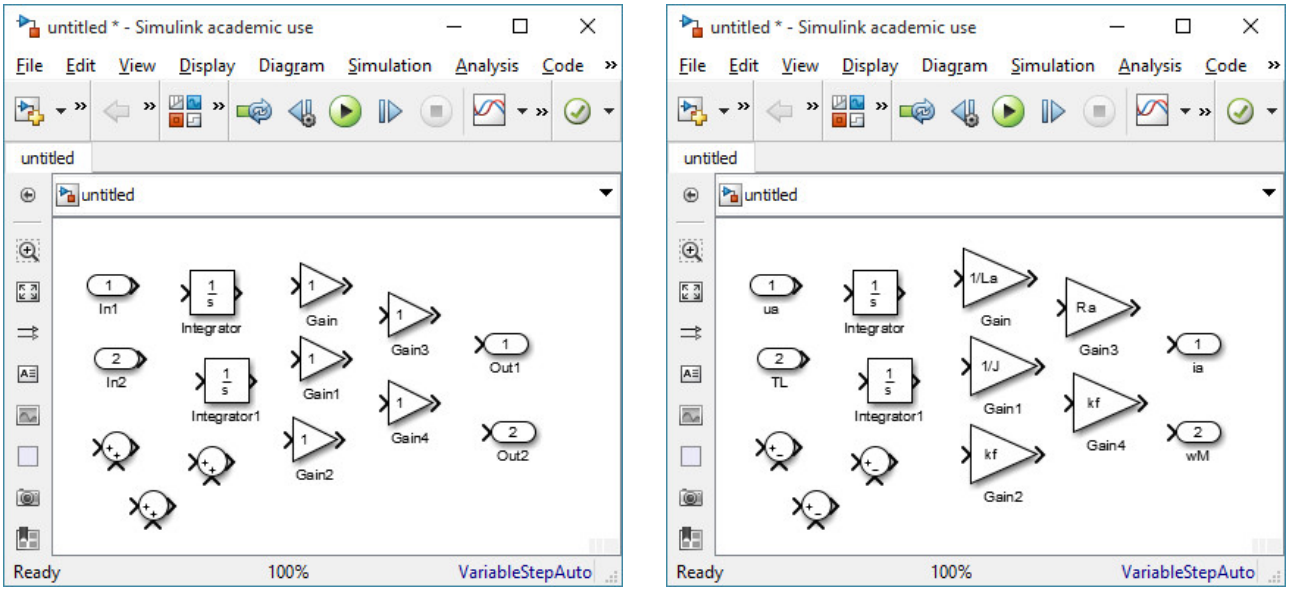
$$L_\mathrm{a}\frac{\mathrm{d}i_\mathrm{a}}{\mathrm{d}t} = u_\mathrm{a} - R_\mathrm{a}i_\mathrm{a} - k_\mathrm{f}\omega_\mathrm{M} \tag{1a}$$

$$J\frac{\mathrm{d}\omega_\mathrm{M}}{\mathrm{d}t} = k_\mathrm{f}i_\mathrm{a} - T_\mathrm{L} \tag{1b}$$

The armature voltage u_a and the load torque T_L are the inputs of the model. The armature current i_a and the angular rotor speed ω_M are the outputs of the model. Rating and parameters of the motor are given in Table 1.

Table 1: Rating and parameters of the DC motor.

Rated voltage U_N	120 V
Rated current I_N	20 A
Rated speed n_N	3000 r/min
Rated torque T_N	7 Nm
Armature resistance R_a	0.5 Ω
Armature inductance L_a	2.5 mH
Flux constant k_f	0.35 Vs
Total moment of inertia J	0.001 kgm^2



(a) (b)

Figure 1: Blocks needed to build the DC motor model: (a) Drag and drop these blocks from the Library Browser to your model; (b) Rename the input and output signals by selecting their names and typing new names. Double click the Sum and Gain blocks and specify them according to the figure.

2.2 Building the Model

Start Simulink by writing the command `simulink` in the Matlab Command Window. The Simulink Library Browser opens up, and you can see the blocks available. Create a new Simulink model using the button or the menu (File→New→Model). For the DC motor model, you will need the blocks shown in Fig. 1(a). Drag and drop these blocks from the Library Browser to your new model. Then, rename the input and output signals by selecting their names and typing new names according to Fig. 1(b). Double click the Sum and Gain blocks and specify them according to Fig. 1(b).

Next, connect the blocks according to the equations given in (1). Use the current i_a and the speed ω_M as state variables in your model. This means that the output of one integrator block should be the current i_a and the output of the other integrator block should be the speed ω_M. The inputs of these integrator blocks should be $\mathrm{d}i_\mathrm{a}/\mathrm{d}t$ and $\mathrm{d}\omega_\mathrm{M}/\mathrm{d}t$, which you can solve from (1).

Create the subsystem from your model. You can create the subsystem by selecting all the blocks and then right-clicking one of the selected blocks. This opens a menu similar to one shown in Fig. 2, where you should choose Create Subsystem from Selection. You can rename the new subsystem as `DC Motor` (click its name and type the new name).

Next, the subsystem will be masked. Open the Mask Editor by right-clicking the subsystem and choosing Mask→Create Mask. In the Mask Editor, choose the pane Parameters & Dialog. Mask the subsystem according to Fig. 3. After you have masked it, you can set numerical values for the model parameters by double-clicking the subsystem, cf. Fig. 4.

Open the Configuration Parameters window, e.g., using the menu: Simulation→Model Configuration Parameters. Set the values `0.5` and `1e-4` for the parameters Stop time and Max step size, respectively, according to Fig. 5. Remember to save your model regularly.

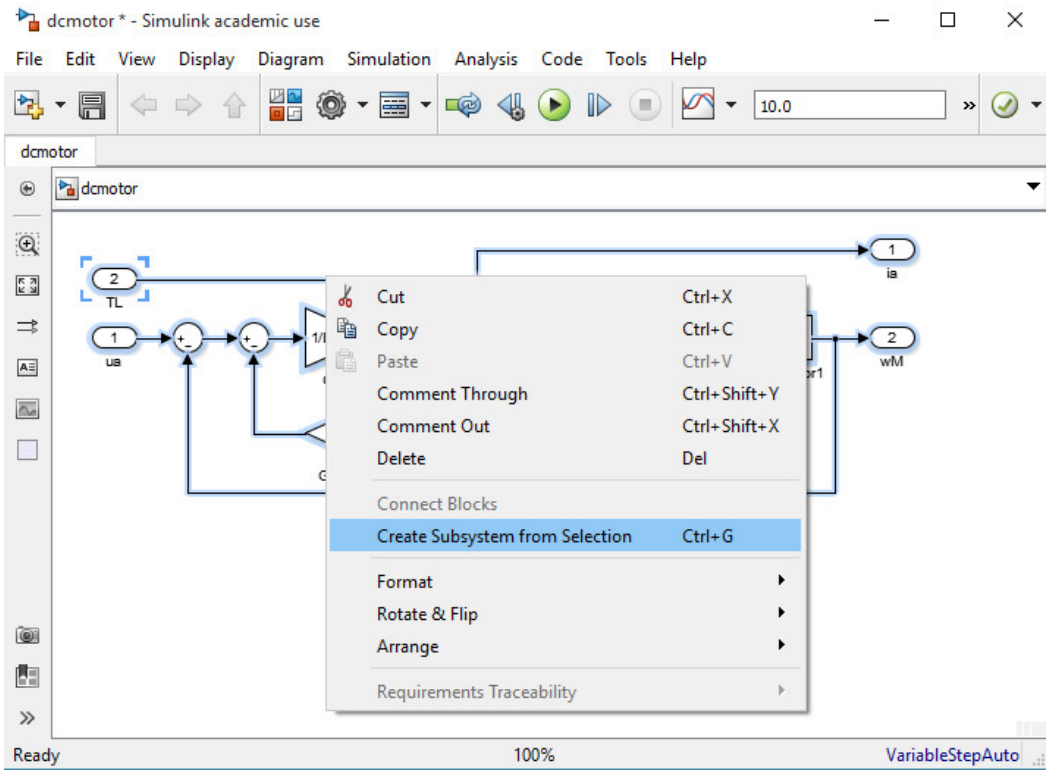


Figure 2: Creating a subsystem.

2.3 Testing the Model

Before starting to build the control system, the motor model will be tested. Connect the blocks Step and To Workspace to the motor model according to Fig. 6. Specify the names of the workspace variables to which the To Workspace block writes the data according to

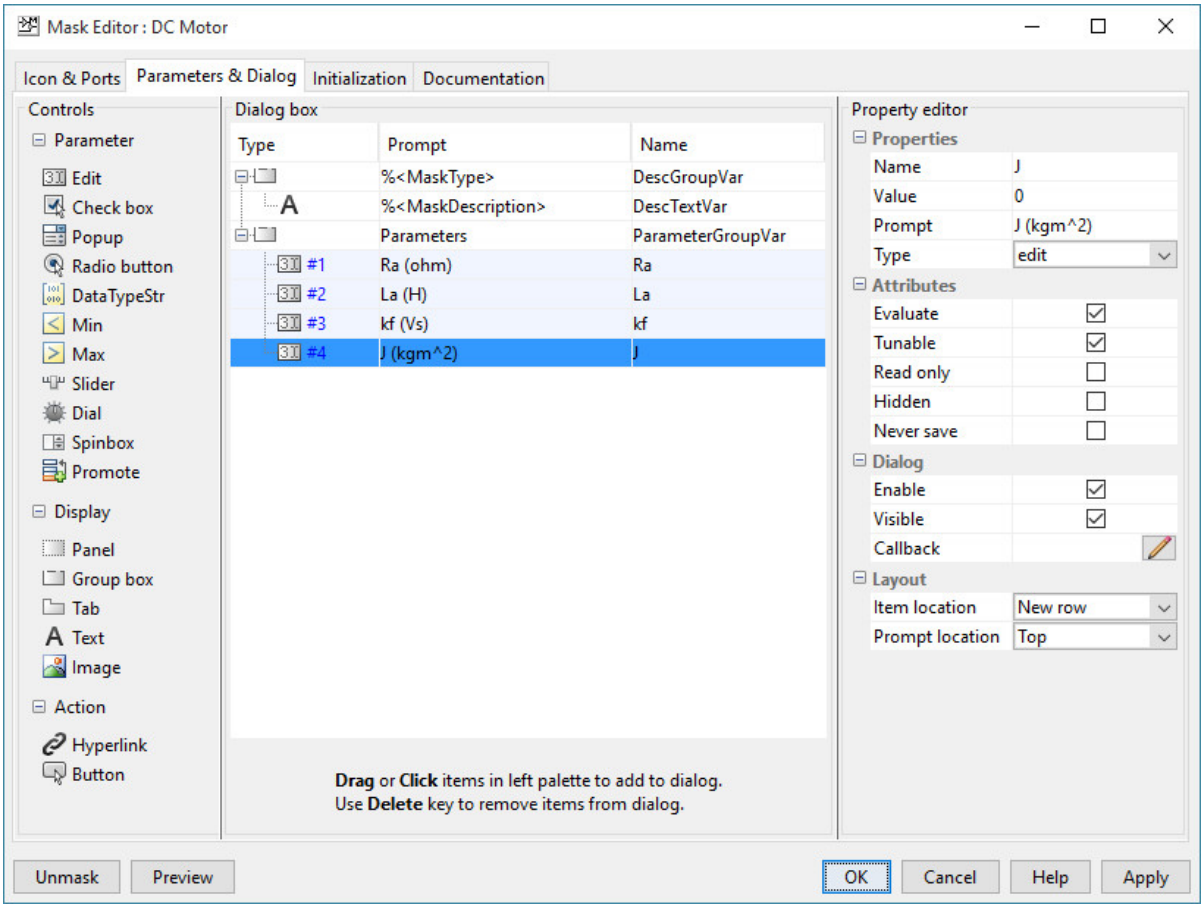


Figure 3: Masking the subsystem.

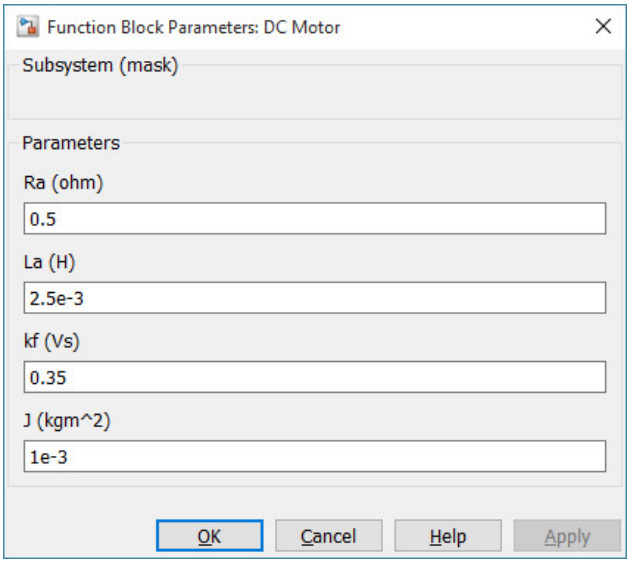


Figure 4: Giving numerical values for parameters.

the figure. Specify the voltage step block so that the voltage is stepped to its rated value 120 V at $t = 0.1$ s. The load torque should be stepped to its rated value at $t = 0.3$ s.

Simulate the model, for example, using the menu: Simulation→Run. After the simulation, you can plot the results in the Matlab workspace using the following commands:

```
subplot(2,1,1);                 % Divides the figure to two subplots
plot(ia.time,ia.data); grid on;
%IN = 20;                       % Rated current
%plot(ia.time,ia.data/IN);      % This would plot the p.u. current
xlabel('Time (s)'); ylabel('Current (A)');
subplot(2,1,2);
plot(wM.time,wM.data); grid on;
xlabel('Time (s)'); ylabel('Speed (rad/s)');
```

It is practical to write and save these commands as a script (e.g., using the name `fig.m`) and then run the script by typing its name (`fig`) in the workspace. This way you can easily edit and reuse your scripts. The simulation results should look similar to those shown in Fig. 7. If they look different, you should debug your model.

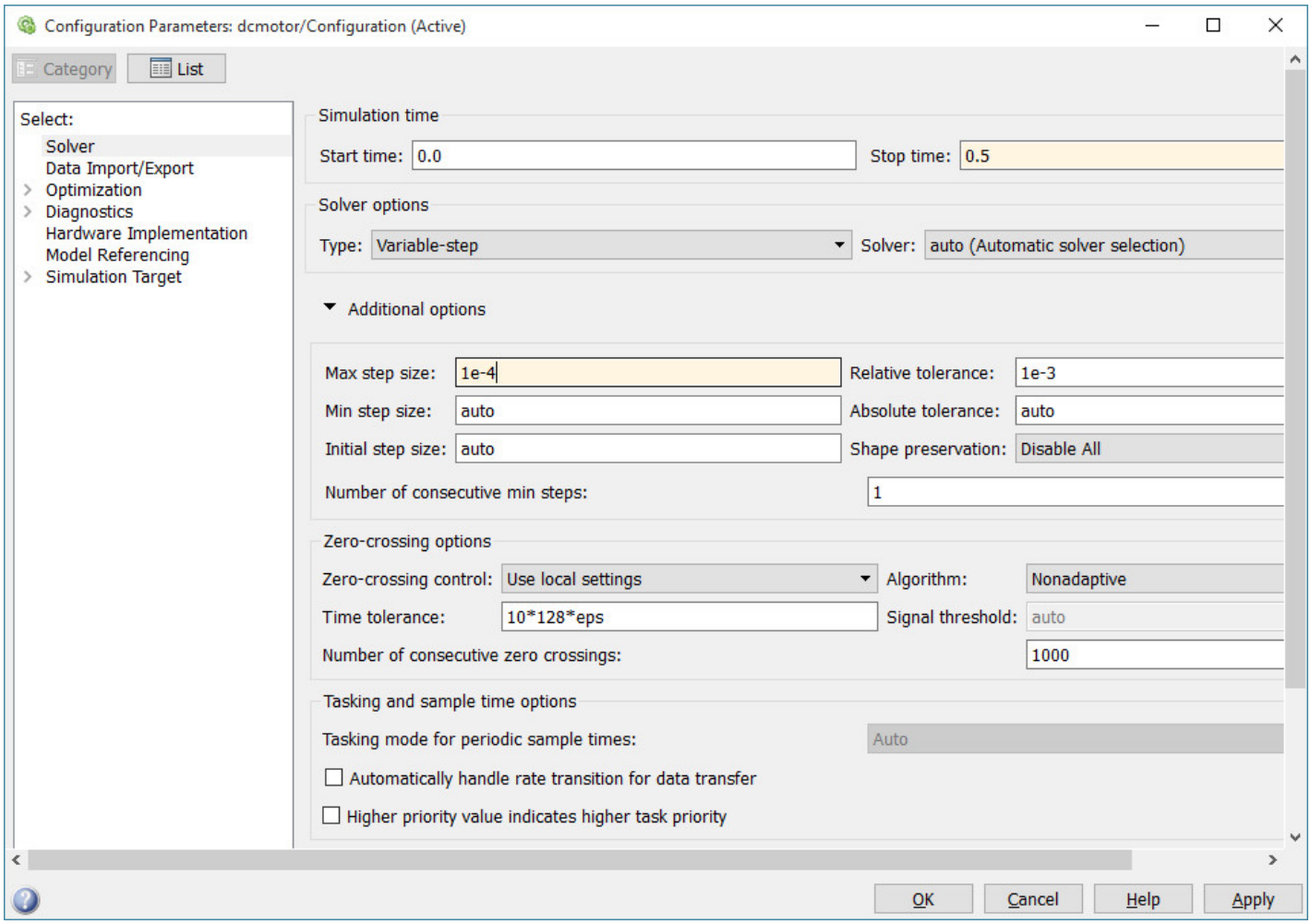


Figure 5: Configuration parameters. Set the new values for the parameters Stop time and Max step size.

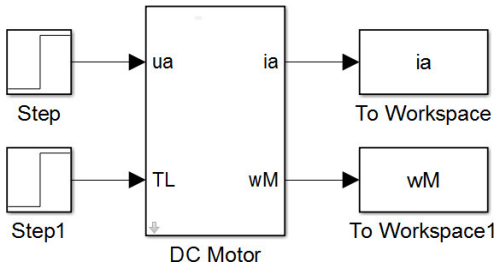


Figure 6: Step inputs for the voltage and load torque.

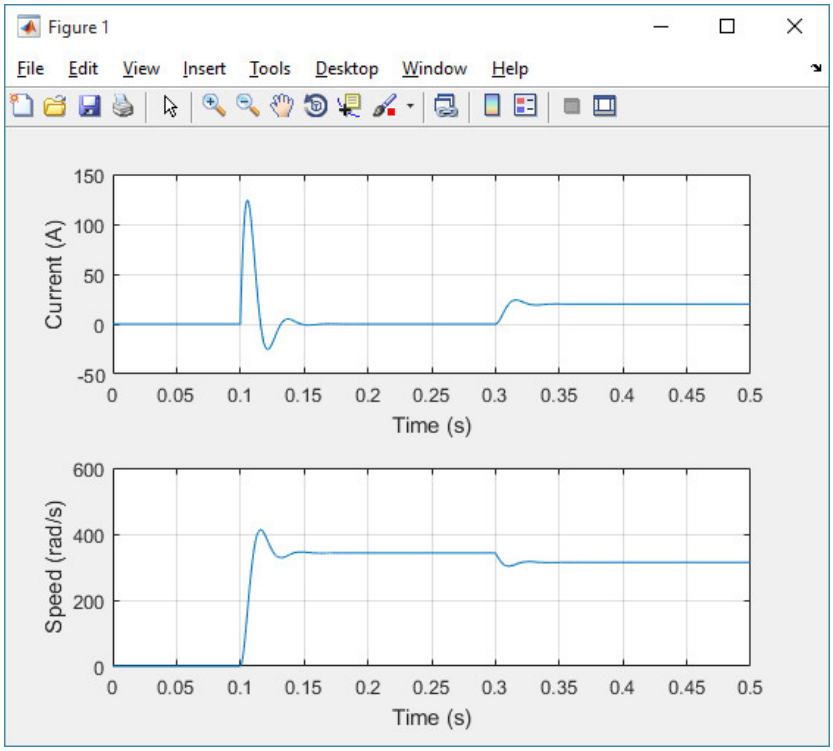


Figure 7: Rated voltage step at $t = 0.1$ s and rated load torque step $t = 0.3$ s.

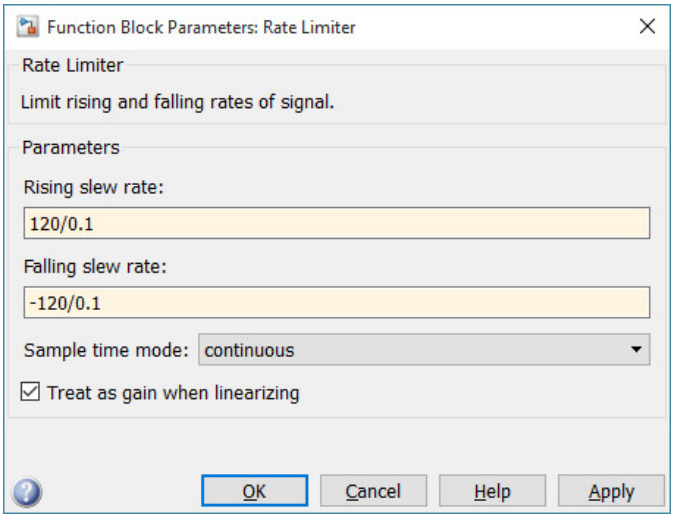


Figure 8: Rate limiter. Select the continuous sample time mode.

1. Simulate the sequence corresponding to Fig. 7. Modify the plotting script so that the per-unit current and the per-unit speed are plotted (use their rated values as base values and do not normalize time). Show this result in your report. Remember to change the axis labels. Explain why there is a very large peak in the current after the voltage step is applied.
2. Using the analytical motor model, calculate the values for the current i_a and the rotor speed ω_M in the steady state, when the voltage $u_a = U_N$ and the load torque $T_L = T_N$. Compare these values to your simulation results.
3. Limit the rising rate of the voltage to 120 V/0.1 s using the Rate Limiter block, cf. Fig. 8. Place this block between the voltage step and the motor model. Simulate the model and show the results in your report. Briefly comment on the current and speed responses.

3 DC-DC Converter and Unipolar PWM

The motor is fed from a four-quadrant DC-DC converter, whose DC-bus voltage is $U_{dc} = 140$ V. Ideal power switches are assumed. Hence, the converter can be modelled using the equivalent circuit shown in Fig. 9(a).

The switching states of the two bi-positional switches are denoted by q_A and q_B. The value of the switching state is 1 if the switch is connected to the positive potential of the DC bus and otherwise 0. The instantaneous output voltage of the converter is $u_a = u_{AN} - u_{BN}$, where u_{AN} is the voltage between potentials A and N and u_{BN} is the voltage between potentials B and N. Hence, the instantaneous output voltage can be expressed as

$$u_a = (q_A - q_B)U_{dc} \tag{2}$$

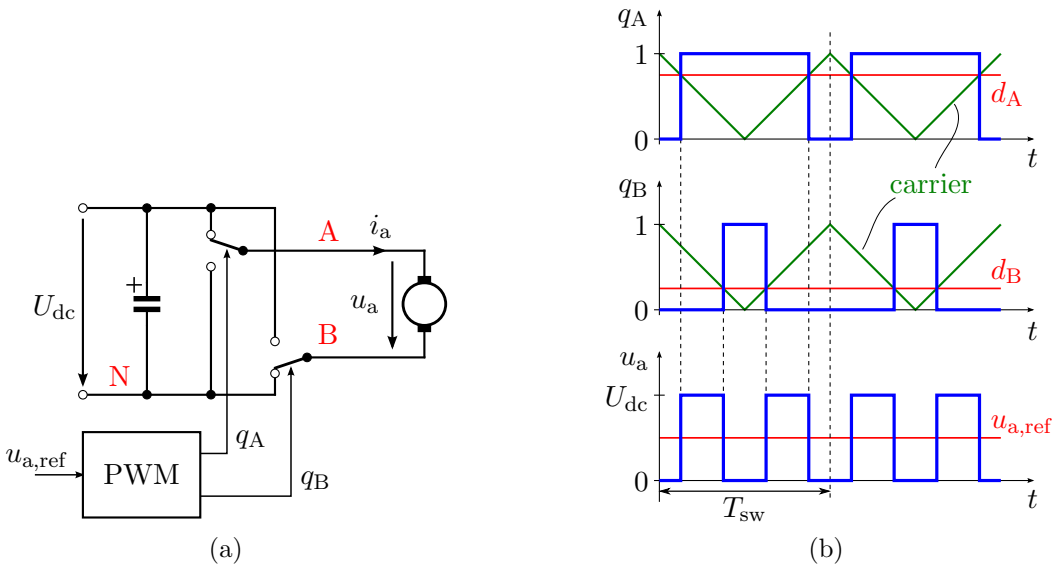


Figure 9: (a) Equivalent circuit of the four-quadrant DC-DC converter. The potentials A, B, and N are marked in the circuit. The positions of the bi-positional switches in the figure correspond to the switching states $q_A = 1$ and $q_B = 0$, and the output voltage is $u_a = U_{dc}$. (b) Example waveforms in unipolar PWM.

The average voltage over the switching cycle is

$$\overline{u}_{\mathrm{a}} = (d_{\mathrm{A}} - d_{\mathrm{B}})U_{\mathrm{dc}} \tag{3}$$

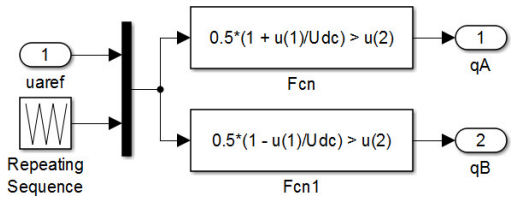
where $0 \leq d_{\mathrm{A}} \leq 1$ and $0 \leq d_{\mathrm{B}} \leq 1$ are the duty cycles.

The switching states of the chopper are to be generated using unipolar PWM, whose operating principle is illustrated in Fig. 9(b). The references for the duty cycles are determined by

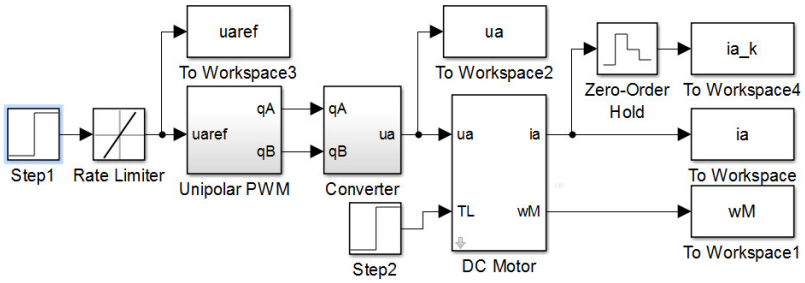
$$d_{\mathrm{A}} = \frac{1}{2}\left(1 + \frac{u_{\mathrm{a,ref}}}{U_{\mathrm{dc}}}\right), \qquad d_{\mathrm{B}} = \frac{1}{2}\left(1 - \frac{u_{\mathrm{a,ref}}}{U_{\mathrm{dc}}}\right) \tag{4}$$

where $u_{\mathrm{a,ref}}$ is the reference voltage. The duty ratios are compared to the carrier signal, which is a triangular wave alternating between 0 and 1 and having a period of $T_{\mathrm{sw}} = 200\ \mu$s. When the duty ratio is higher than the carrier, the corresponding switching state is 1 and otherwise 0. An implementation of unipolar PWM is shown in Fig. 10(a).

For simulating fast switching phenomena with good accuracy, the solver time step should be a few decades shorter than the switching period (or, alternatively, the solver should be informed about the switching instants). Open the Configuration Parameters window and set the value `1e-6` for the parameter Max step size (cf. Fig. 5). Naturally, the simulation becomes slower due to the shorter time step.



(a)



(b)

Figure 10: (a) Simulink implementation of unipolar PWM. (b) Motor model with unipolar PWM and converter model. Specify the Repeating Sequence block by setting `Tsw*[0 0.5 1]` for the Time values and `[1 0 1]` for the Output values. The Zero-Order Hold block represents sampling synchronised to the PWM (set the value `Tsw/2` for the Sampling period parameter). You can assign the values for the variables `Udc` and `Tsw` either via masking the subsystem (as was done in the case of the DC motor) or simply via the Matlab workspace (i.e., type `Udc = 140` and `Tsw = 200e-6` in the workspace).

4. Augment your simulation model with unipolar PWM and converter models. Your model should look similar to the model in Fig. 10(b). Simulate the model and show the results in your report. Briefly comment on differences compared to the previous simulation, where an ideal voltage source was assumed. Submit this version of your simulation model to MyCourses.
5. Plot the waveforms of the actual current `ia` and the synchronously sampled current `ia_k` in the same subplot. Show also the waveform of the voltage `ua`. You can plot the results using the following script:

```
subplot(2,1,1)
plot(ia.time,ia.data); grid on; hold on;
stairs(ia_k.time,ia_k.data,'r');     % Discrete signal
axis([0.15 0.1504 9 11]);            % Controls axis scaling
xlabel('Time (s)'); ylabel('Current (A)');
subplot(2,1,2)
plot(ua.time,ua.data); grid on;
axis([0.15 0.1504 -10 150]);
xlabel('Time (s)'); ylabel('Voltage (V)');
```

   Show the results in your report and briefly comment on them.

4 Cascaded Control

The control system of the DC motor is shown in Fig. 11. The control structure is cascaded: the outer loop is the speed control loop and the inner loop is the current-control loop. For simplicity, the switching harmonics due to the PWM will be omitted in the following (but the voltage saturation will be taken into account).

Save the simulation model you have made, and copy it using a new name for the following changes. You can remove the PWM and converter models. In order to speed up your simulations, open the Configuration Parameters window and set `1e-4` for the Max step size parameter.

4.1 Current Control

Augment your simulation model according to Fig. 12(a). An implementation of the 2DOF PI current controller with the anti-windup scheme is shown in Fig. 12(b). You can tune the

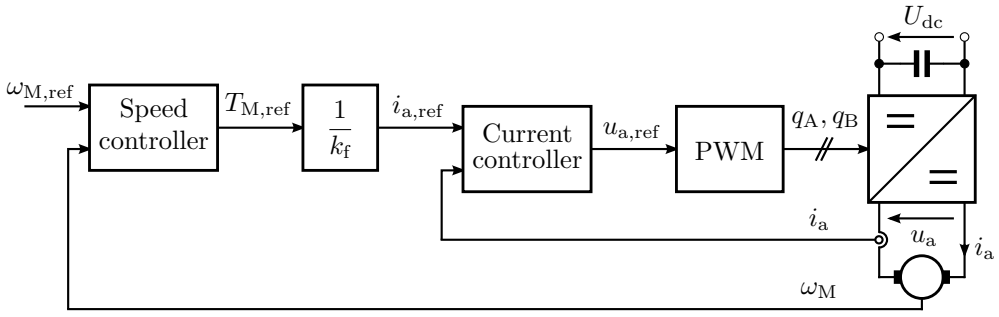


Figure 11: Cascaded control system of the DC motor. In the following, PWM and converter models are omitted for simplicity.

controller using the following script, which you should run before starting the simulation:

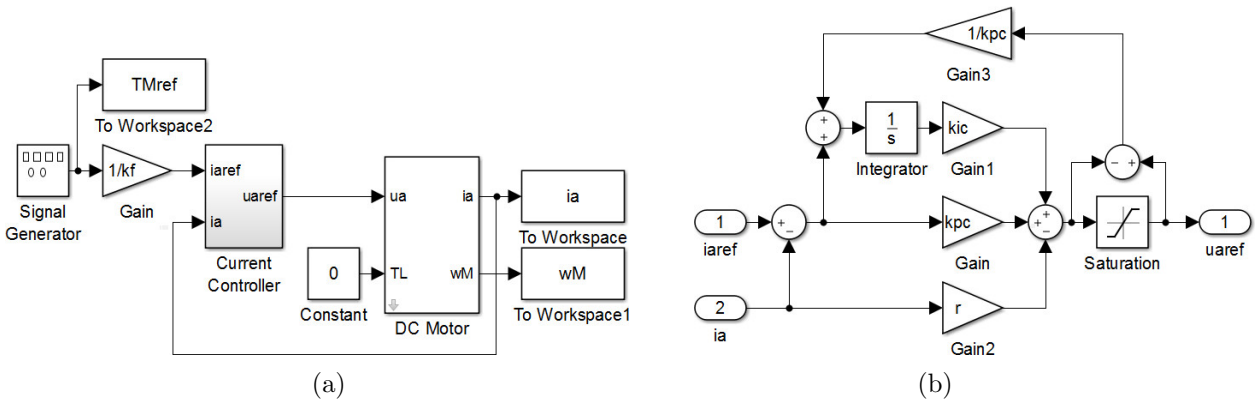
```
clear;                          % Removes variables from the workspace
%% Parameter estimates
Ra = 0.5;                       % Armature resistance
La = 2.5e-3;                    % Armature inductance
kf = 0.35;                      % Flux constant
J = 1e-3;                       % Moment of inertia
%% Gains of the 2DOF PI current controller
alphac = 2*pi*500;              % Closed-loop bandwidth
kpc = alphac*La;                % Proportional gain
kic = alphac^2*La;              % Integral gain
r = kpc - Ra;                   % Active resistance
Umax = 140;                     % Saturation: upper limit
Umin = -140;                    % Saturation: lower limit
%% Gains of the 2DOF PI speed controller (for Section 4.2)
%alphas = 0.1*alphac;           % Closed-loop bandwidth
%kps = ...;                     % Proportional gain
%kis = ...;                     % Integral gain
%b = ...;                       % Active damping
%TN = 7;                        % Rated torque
%Tmax = 2*TN;                   % Saturation: upper limit
%Tmin = -2*TN;                  % Saturation: lower limit
```

Test your model using a 1-Nm 100-Hz square-wave torque reference. The results should look similar to those in Fig. 13.

6. Calculate the theoretical rise time of the torque and compare it to the simulated rise time.

4.2 Speed Control

Finally, the 2DOF PI speed controller will be implemented and tuned. This controller has a similar structure as the 2DOF PI current controller. An implementation of the control system is shown in Fig. 14.



(a) (b)

Figure 12: (a) Torque control. For testing the model, specify the Signal Generator block to generate the square wave with an amplitude of 1 Nm and a frequency of 100 Hz. (b) 2DOF PI current controller with anti-windup. Set `Umax` for the upper limit and `Umin` for the lower limit in the Saturation block.

7. A term proportional to the measured speed is subtracted from the output $T'_{\mathrm{M,ref}}$ of the PI speed controller, i.e. $T_{\mathrm{M,ref}} = T'_{\mathrm{M,ref}} - b\omega_{\mathrm{M}}$, where b is the active damping constant. Derive the transfer function $G'(s)$ from $T'_{\mathrm{M,ref}}$ to the angular speed ω_{M} assuming the current control to be ideal (i.e., the torque equals its reference). Based on this result, give the tuning equations for the 2DOF PI speed controller.
8. Tune the speed controller of your simulation model for the closed-loop bandwidth $\alpha_{\mathrm{s}} = \alpha_{\mathrm{c}}/10$. Test your model using the square-wave speed reference, whose amplitude is 160 rad/s and frequency is 4 Hz. Generate the rated load torque step at $t = 0.3$ s. Show results of this simulation in your report. Show also the figures describing the main level of your simulation model and the implemented speed controller. Submit this version of your Simulink model to MyCourses (including your initialisation script).
9. This problem aims to illustrate the robustness of the closed-loop control scheme against parameter errors. Generally, resistances depend on temperature (about 0.4%/K) and inductances may vary due to the magnetic saturation. Change the actual armature resistance in the motor model to 150% of its original value and the actual armature inductance to 70% of its original value, but do not change the values in the control system. Simulate the model. Show the results and comment on them in your report. After this problem, restore these parameter values back to their original values.
10. This problem aims to illustrate the importance of the anti-windup scheme. Remove the anti-windup in the speed controller (but do not remove the saturation of the controller output). Show results of your simulation and comment on them.

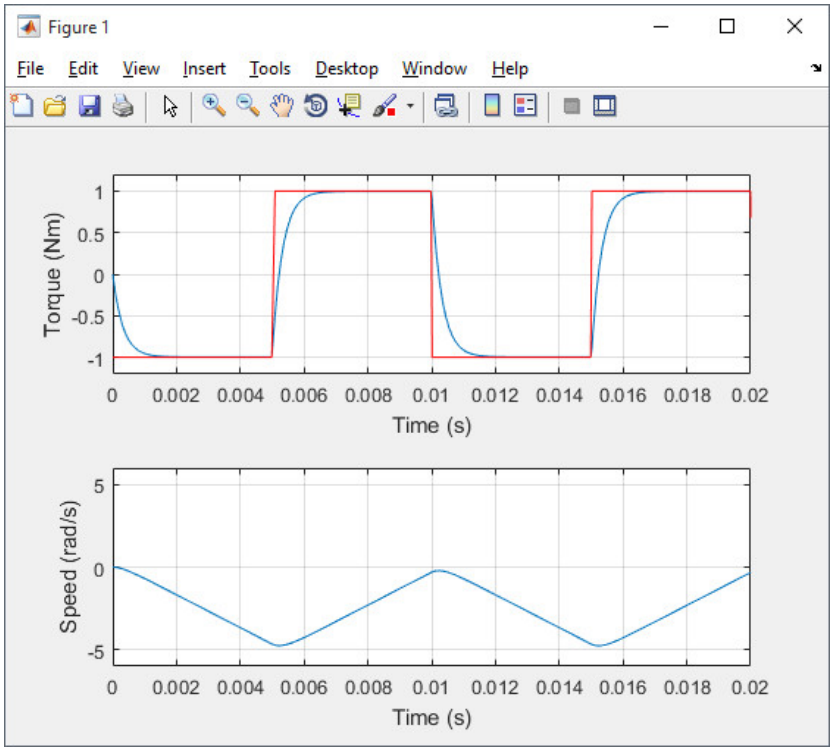


Figure 13: Testing of the torque controller with a 1-Nm 100-Hz square-wave torque reference.

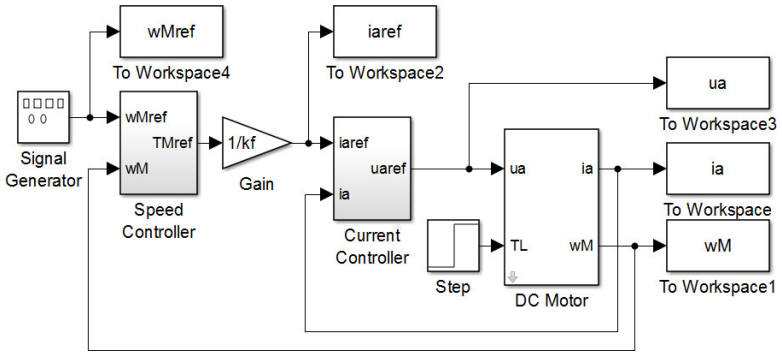


Figure 14: Cascaded control. Specify the Signal Generator block to generate the square wave with an amplitude of 160 rad/s and a frequency of 4 Hz. Specify the Step block to generate the rated torque step at $t = 0.3$ s.

Give Us Feedback

In order to improve this assignment, please give us feedback. In order to estimate the student workload, we would also be happy to know how many hours did you use to do this assignment. All other comments are also welcome.