Technische Universität Berlin

Faculty for Electrical Engineering and Computer Science

Professor Energieversorgungsnetze und Integration Erneuerbarer Energien

Renewable Energy Technology for WS 20/21

Laboratory Wind Assignment

Renewable Energy Technology WS 20/21

Professor: Kai Strunz

Submission date: 21 February 2021

Name	Matriculation number
Heesun Joo	451995
Hutomo Rachmat Saleh	461980

Table 1.

Structure

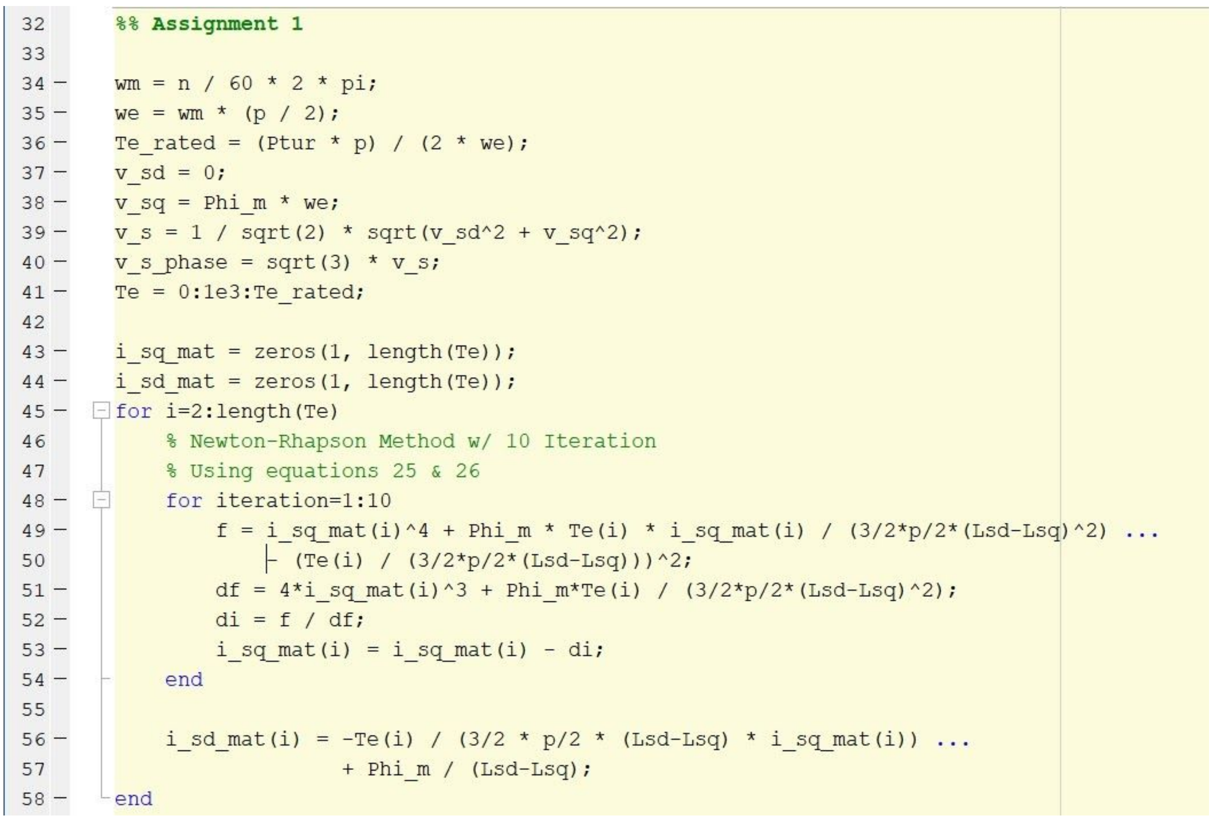
Task 1: Electric System Modeling and Current Command Synthesizer

Description:

Our first task is to model the electric system and current command synthesizer. Here, we calculate the rated electrical angular frequency $\boldsymbol{\omega_e}$, rated electrical torque $\mathbf{T_e}$ as well as rated generator voltage $\mathbf{v_s}$. We then want to observe the changes in the current for the corresponding d- and q-Axis vs the varying Te. Lastly we are to transform the normal current into the dq-Frame.

Task 1.1:

The formulas used are all obtained from lecture and can be seen in the following code:



```matlab
%% Assignment 1

wm = n / 60 * 2 * pi;
we = wm * (p / 2);
Te_rated = (Ptur * p) / (2 * we);
v_sd = 0;
v_sq = Phi_m * we;
v_s = 1 / sqrt(2) * sqrt(v_sd^2 + v_sq^2);
v_s_phase = sqrt(3) * v_s;
Te = 0:1e3:Te_rated;

i_sq_mat = zeros(1, length(Te));
i_sd_mat = zeros(1, length(Te));
for i=2:length(Te)
    % Newton-Rhapson Method w/ 10 Iteration
    % Using equations 25 & 26
    for iteration=1:10
        f = i_sq_mat(i)^4 + Phi_m * Te(i) * i_sq_mat(i) / (3/2*p/2*(Lsd-Lsq)^2) ...
            + (Te(i) / (3/2*p/2*(Lsd-Lsq)))^2;
        df = 4*i_sq_mat(i)^3 + Phi_m*Te(i) / (3/2*p/2*(Lsd-Lsq)^2);
        di = f / df;
        i_sq_mat(i) = i_sq_mat(i) - di;
    end

    i_sd_mat(i) = -Te(i) / (3/2 * p/2 * (Lsd-Lsq) * i_sq_mat(i)) ...
                + Phi_m / (Lsd-Lsq);
end
```

Task 1.2:

The resulting plot of the code above are shown in Figure 1.2.1

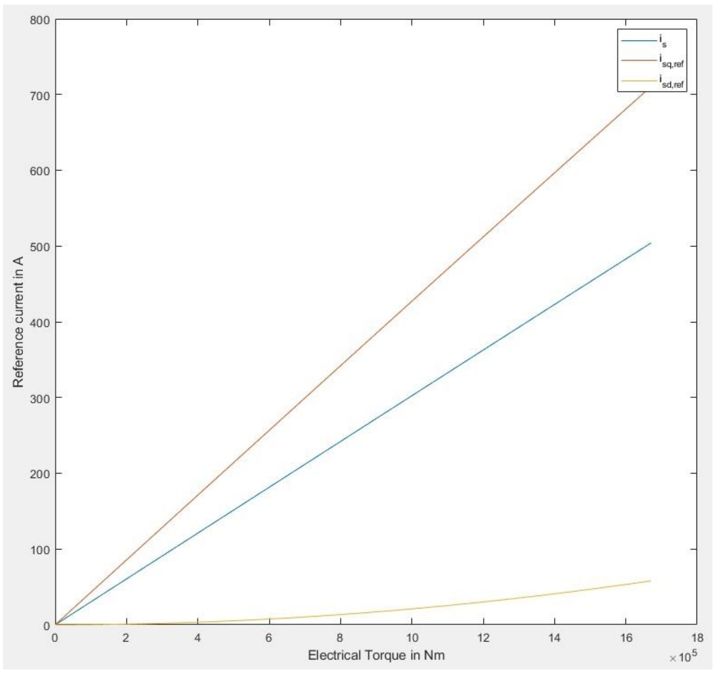


Figure 1.2.1. Plot of i_s, $i_{sd,ref}$ and $i_{sq,ref}$ over time

Task 1.3:

In this task, we applied the derivation shown in the Clarke Transform in slide 7.

Transformation to d-q-Frame :

$$\begin{pmatrix} i_{sd} \\ i_{sq} \end{pmatrix} = \frac{2}{3} \begin{pmatrix} \cos\theta & \cos\left(\theta - \frac{2\pi}{3}\right) & \cos\left(\theta + \frac{2\pi}{3}\right) \\ -\sin\theta & -\sin\left(\theta - \frac{2\pi}{3}\right) & -\sin\left(\theta + \frac{2\pi}{3}\right) \end{pmatrix} \begin{pmatrix} i_{sa} \\ i_{sb} \\ i_{sc} \end{pmatrix}$$

$\Rightarrow$ assume $\theta = 0°$

$$\begin{pmatrix} i_{sd} \\ i_{sq} \end{pmatrix} = \begin{pmatrix} \frac{2}{3} & -\frac{1}{3} & -\frac{1}{3} \\ 0 & \frac{1}{\sqrt{3}} & -\frac{1}{\sqrt{3}} \end{pmatrix} \begin{pmatrix} i_{sa} \\ i_{sb} \\ i_{sc} \end{pmatrix}$$

$$\begin{pmatrix} i_{sd} \\ i_{sq} \end{pmatrix} = \begin{pmatrix} \frac{2}{3} i_{sa} & -\frac{1}{3} i_{sb} & -\frac{1}{3} i_{sc} \\ & \frac{1}{\sqrt{3}} i_{sb} & -\frac{1}{\sqrt{3}} i_{sc} \end{pmatrix}$$

$$\Rightarrow \frac{3}{4}(i_{sd} + i_{sq}) = \frac{3}{4}\left(\frac{2}{3} i_{sa} + \left(\frac{1}{\sqrt{3}} - \frac{1}{3}\right) i_{sb} - \left(\frac{1}{\sqrt{3}} + \frac{1}{3}\right) i_{sc}\right)$$

$$= \frac{1}{2} \cdot \frac{3}{2}\left(\frac{2}{3} i_{sa} + \frac{3 - \sqrt{3}}{3\sqrt{3}} i_{sb} - \frac{3 + \sqrt{3}}{3\sqrt{3}} i_{sc}\right)$$

$$\frac{3}{4}(i_{sd} + i_{sq}) = \frac{1}{2}\left(i_{sa} + \frac{6 - 2\sqrt{3}}{\sqrt{3}} i_{sb} - \frac{6 + 2\sqrt{3}}{\sqrt{3}} i_{sc}\right)$$

$$\Rightarrow \frac{3}{4}(m_d i_{sd} + m_q i_{sq}) = \frac{1}{2}\left(m_a i_{sa} + m_b i_{sb} + m_c i_{sc}\right) = i_{dc}$$

Task 2: Control and Stability in Wind Energy Converters Using Transfer Function Analysis

Description:

As our second task, we focus on building controller models in simulink. We are to calculate the parameters of the controller **Kd,i**, **Kd,p**, **Kq,i**, **Kq,p** for **τ**i = 3ms. After that we are to build a simple controller model as shown in the project overview. The simulink model is shown in Figure 2.1.1.

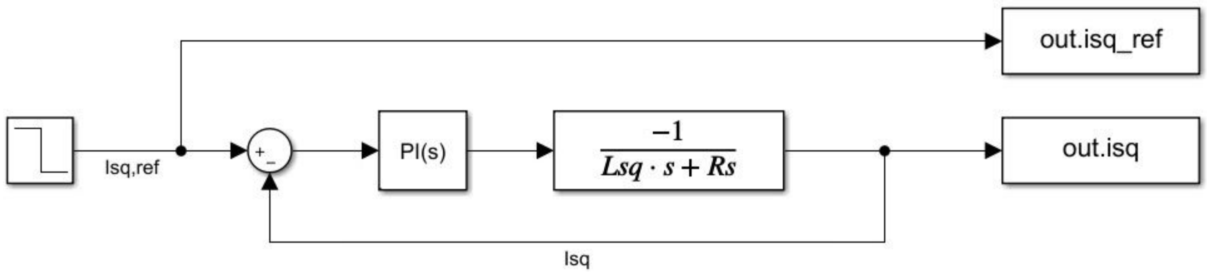


Figure 2.1.2. Plot of isq and isq,ref over time

Task 2.1)

For the simulink model we followed the Figure shown in the assignment overview. The $i_{sq,ref}$ is fed using a step up block with an initial value of $i_{sq,ref}$ and a final value of $i_{sq,ref}$ - Δi_{sq}. The values of the PI Controller are first calculated in code and exported to our simulink model. The resulting plot is shown in Figure 2.1.2.

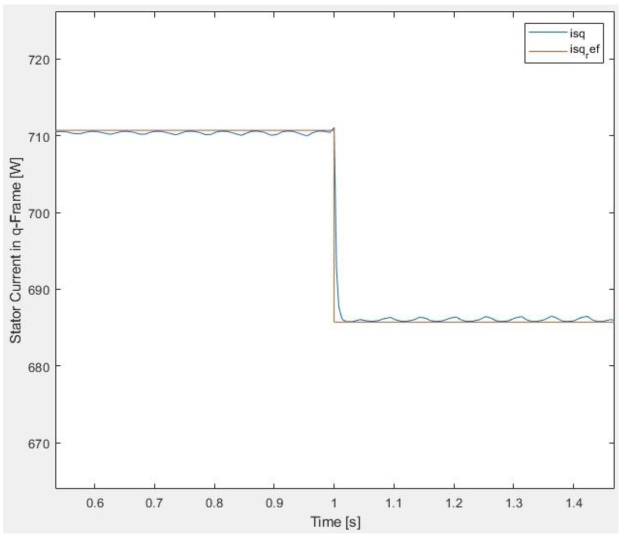
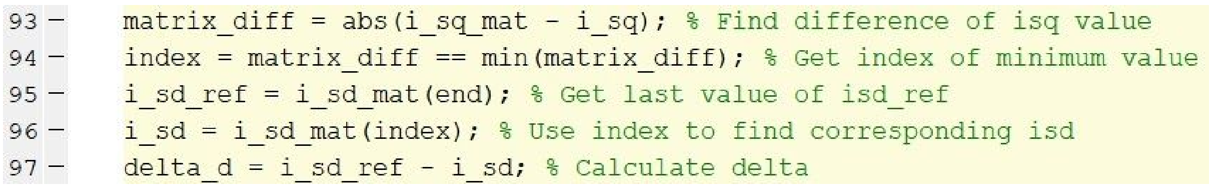


Figure 2.1.2. Plot of isq and isq,ref over time

b) Since both isq and isd has been calculated simultaneously in Task 1, each isq is coupled with isd. Using the find() function, the index can be found from the isq array from Task 1 and thus the corresponding value of isd can be also be found. The code is shown below.



```
matrix_diff = abs(i_sq_mat - i_sq); % Find difference of isq value
index = matrix_diff == min(matrix_diff); % Get index of minimum value
i_sd_ref = i_sd_mat(end); % Get last value of isd_ref
i_sd = i_sd_mat(index); % Use index to find corresponding isd
delta_d = i_sd_ref - i_sd; % Calculate delta
```

c) Figure 2.1.3 shows an enlarged image of the previous plot and has been modified to show the τ_i and the change of current during this time step. The value measured shown in the figure is as expected.

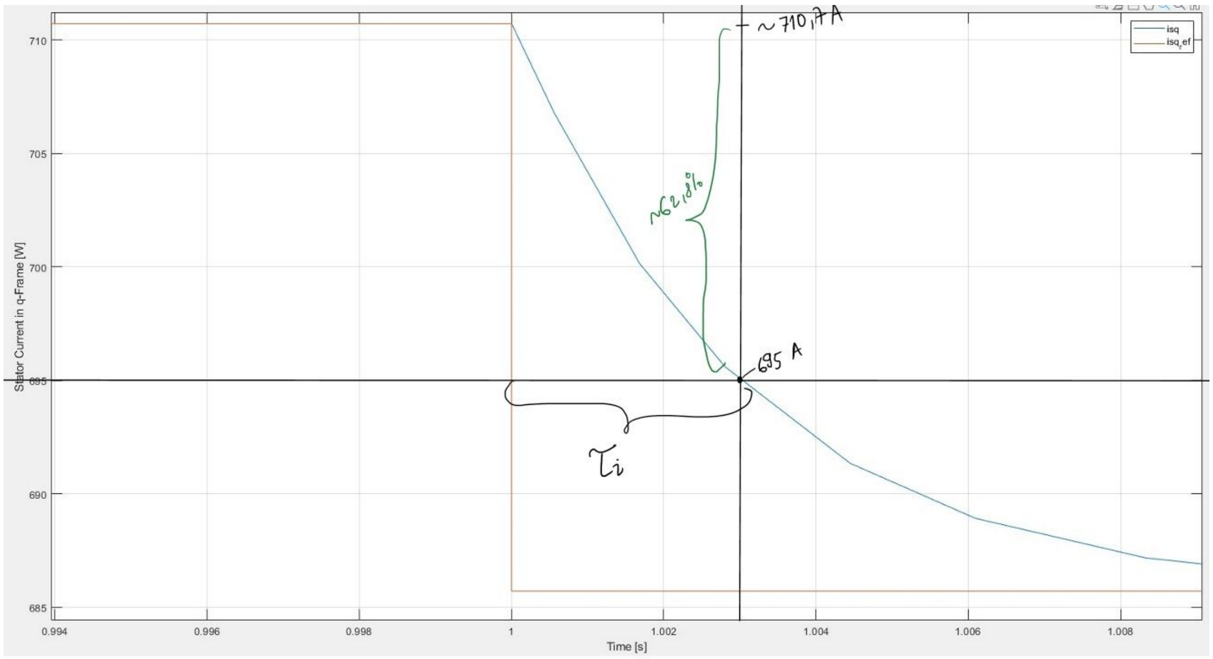


Figure 2.1.3. Enlarged view of isq

d) No because as shown in equation (17) in the paper. The controller parameters are strictly dependent on the values of Rs and Lsd for d-Frame and Lsq for q-Frame.

Task 2.2:

In this task, our main task is to recreate the block diagram of current control. All the variables are imported from the .m code and the results of m_d, m_q and v_{sq}, v_{sd}.

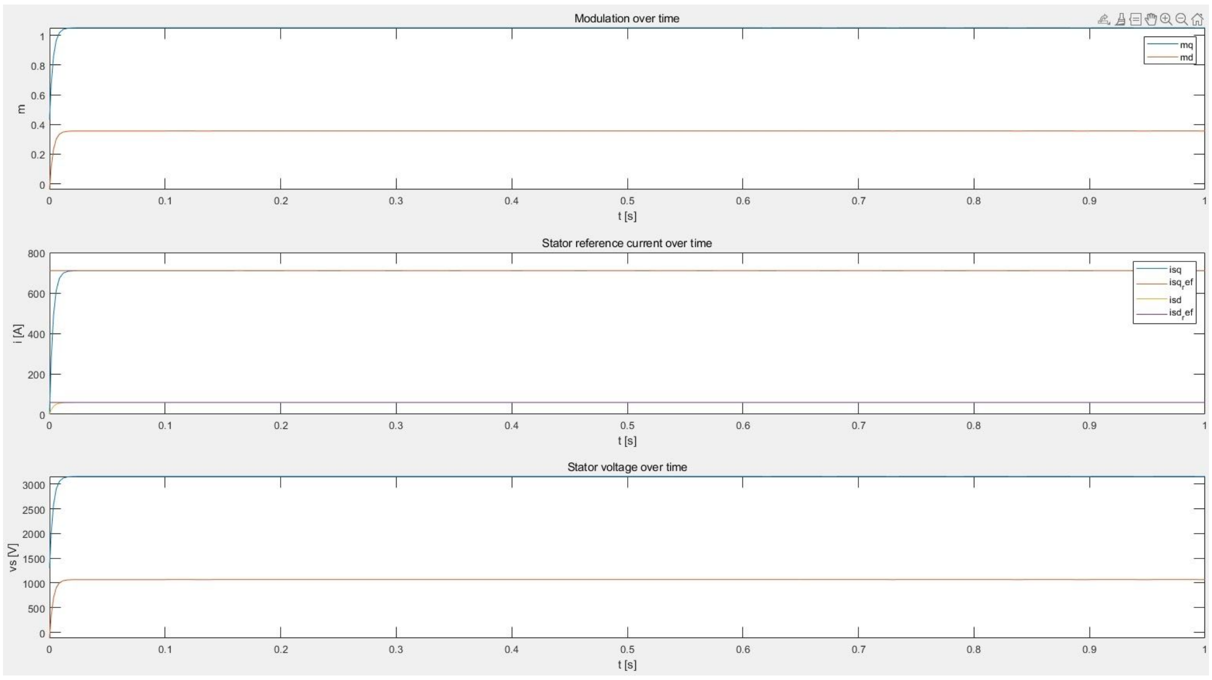


Figure 2.2.1. Plot of i_{sq}, i_{sd}, m_q, m_d, v_{sq} & v_{sd}

c) As we can observe in the plot diagram in Figure 2.2.2. The difference between the output Power calculated with the produced Idc and the Ptur given converges to zero. This shows that the expected power generation is as expected.

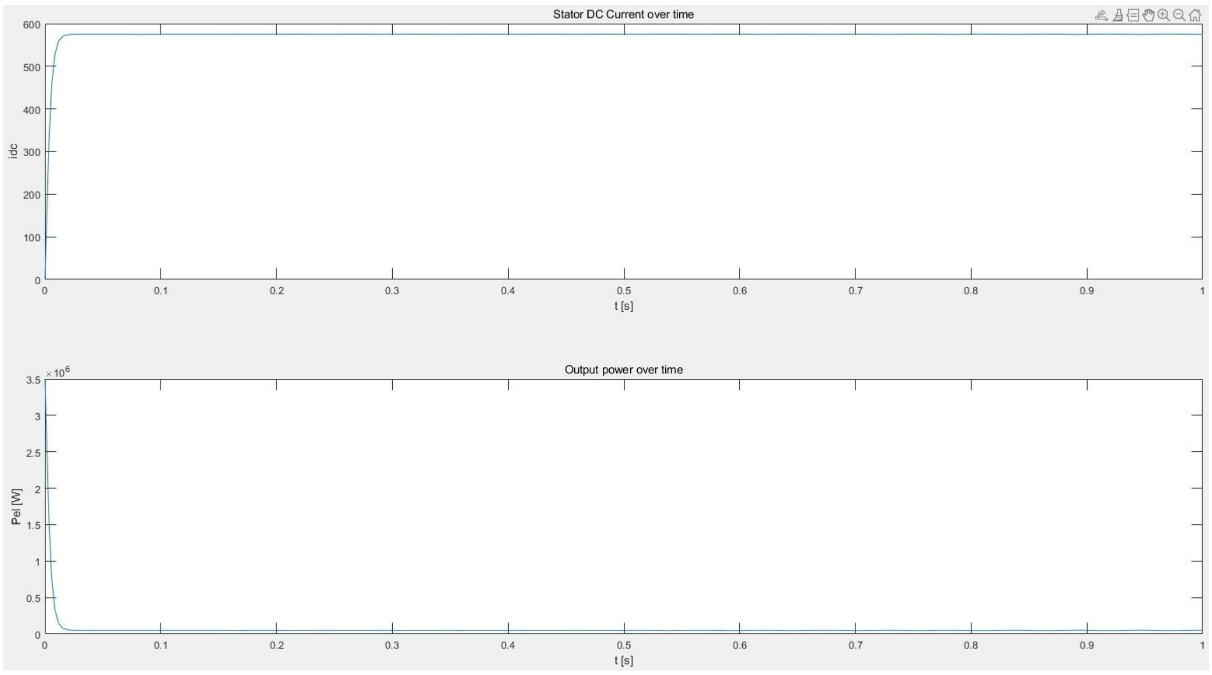


Figure 2.2.2. Course of i_{dc} and P_{el} over time

Task 3: Torque Generation and Turbine-Rotor Interaction Process

Description:

Interpretation:

Task 4: Nonlinear Wind Turbine Model

Description:

Using our models and data in assignments 1-3, our last task is to combine them and run simulations. The subsystem power synthesizer and controller are to be constructed in 4.1 and 4.2. In Task 4.3 and 4.4 the models are combined and run with static as well as varying wind speeds. The graphs are to be observed and explained.

Task 4.1:

The simulink models of the power controller are shown in Fig. 4.1.1

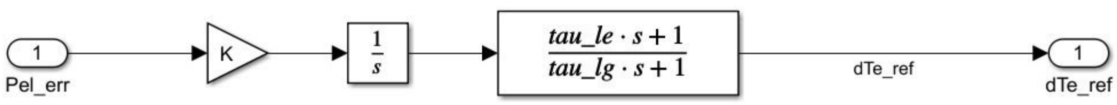


Figure 4.1.1. Simulink Block of Power Controller

The lead τ_{le} and lag τ_{lg} time constant compensates the process due to τ_{ω} and τ_{z}. Which is why the value is the same. The closed-loop time constant is given as 5% of the mechanical lag time $\tau_{le} = \tau_{\omega}$.

We then apply a sudden change of reference power of -50kW to the turbine. We coupled the power controller with the turbine-rotor interaction process block made in our previous task. As input we used a step function which introduces our spike in power at t=5s. The whole block is shown in Fig. 4.1.2

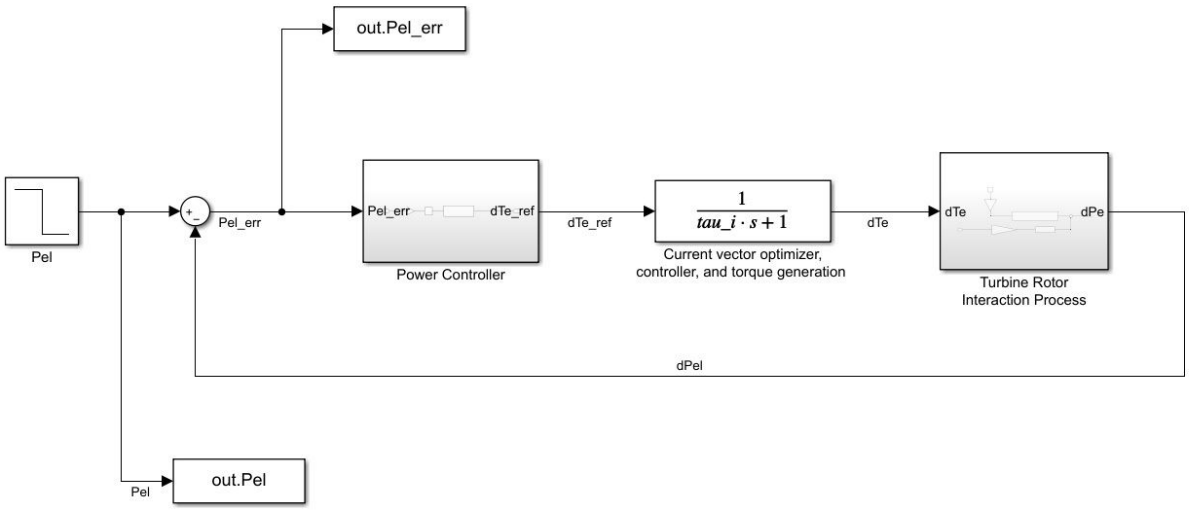


Figure 4.1.2. Complete Simulink Block of Task 4.1

As we can see in Figure 4.1.3 the error converges to 0 as it should be. We can observe a small spike at the time of t = 5s. This is due to the sudden power change specified for this task. The controller then restabilizes the value and the error converges to zero again.

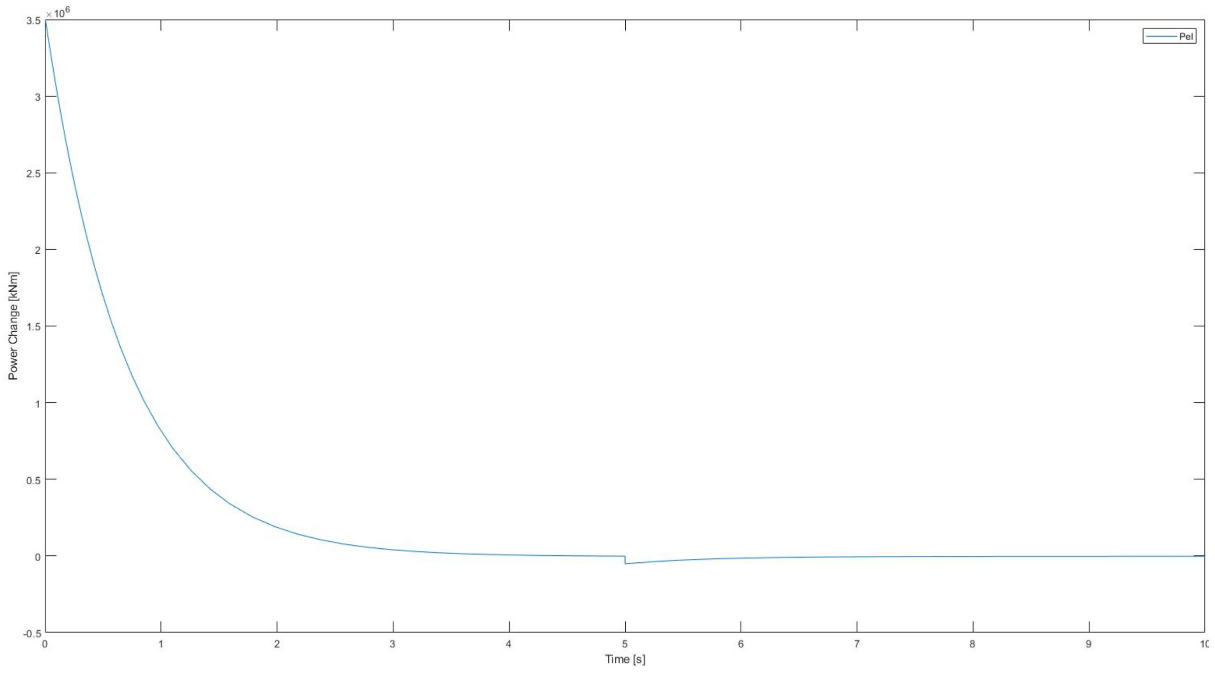


Figure 4.1.3. Change in Pel over time

Task 4.2:

In this task, we built the power command synthesizer and power measurement in Simulink. There is no simulation involved. The simulink blocks are shown in Figure 4.2.1 and 4.2.2

blocks are derived from the equations shown on slides 54 and 58 from chapter 8 respectively.

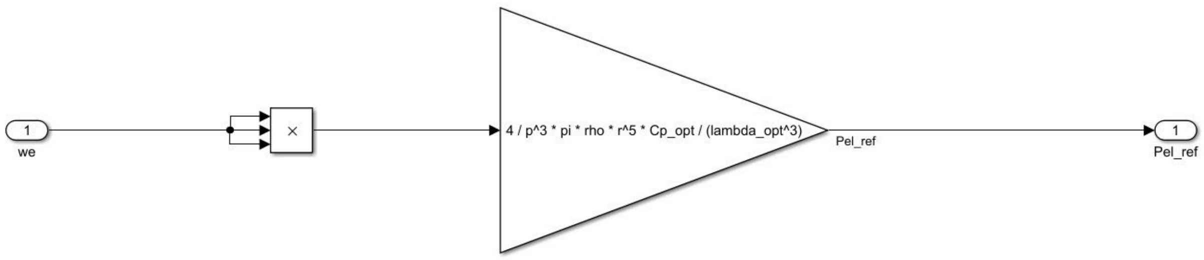


Figure 4.2.1. Simulink Block of Power Command Synthesizer

With the assumption of non-changing i_{sd} and i_{sq} given from the task, the PL becomes zero. Thus the Power P_e is measured only from the difference between P_{ter} and P_L.

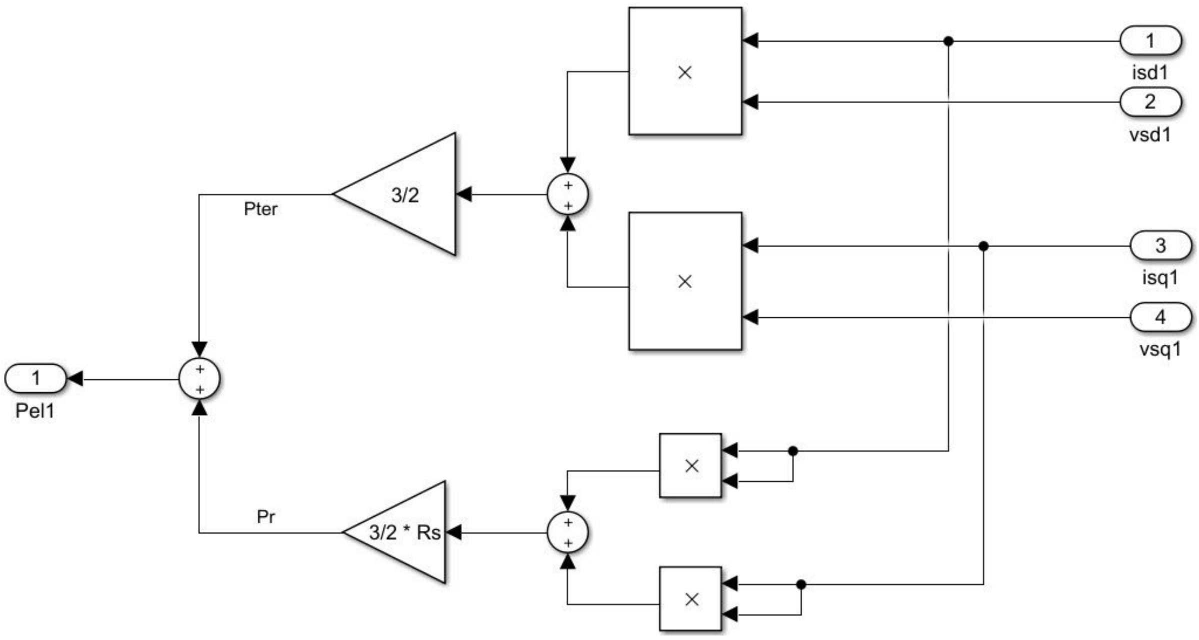


Figure 4.2.2. Simulink Block of Power Measurement

Task 4.3:

Using the blocks made from the previous tasks, we now build the whole wind turbine system. The complete simulink model is shown in Figure 4.3.1.

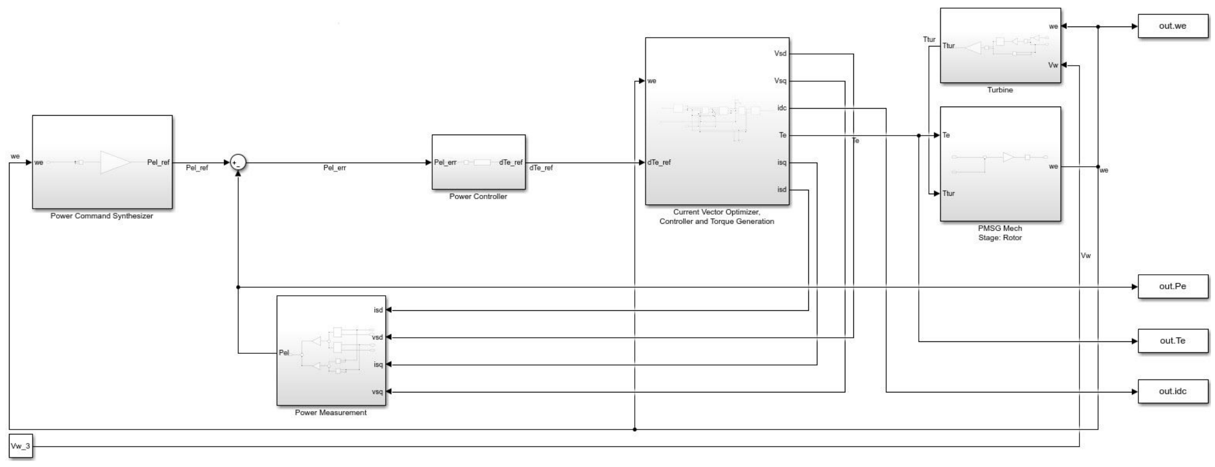


Figure 4.3.1. Complete Wind Turbine System in Simulink

The block from Task 4.1 and 4.2 is added with the simulink block from 3.3. The simulation is to be executed with a constant wind speed of 10 m/s. We are to observe the course of power P_e, electric angular velocity ω_e and torque T_e over 300s. The resulting plot is shown in Figure 4.3.2.

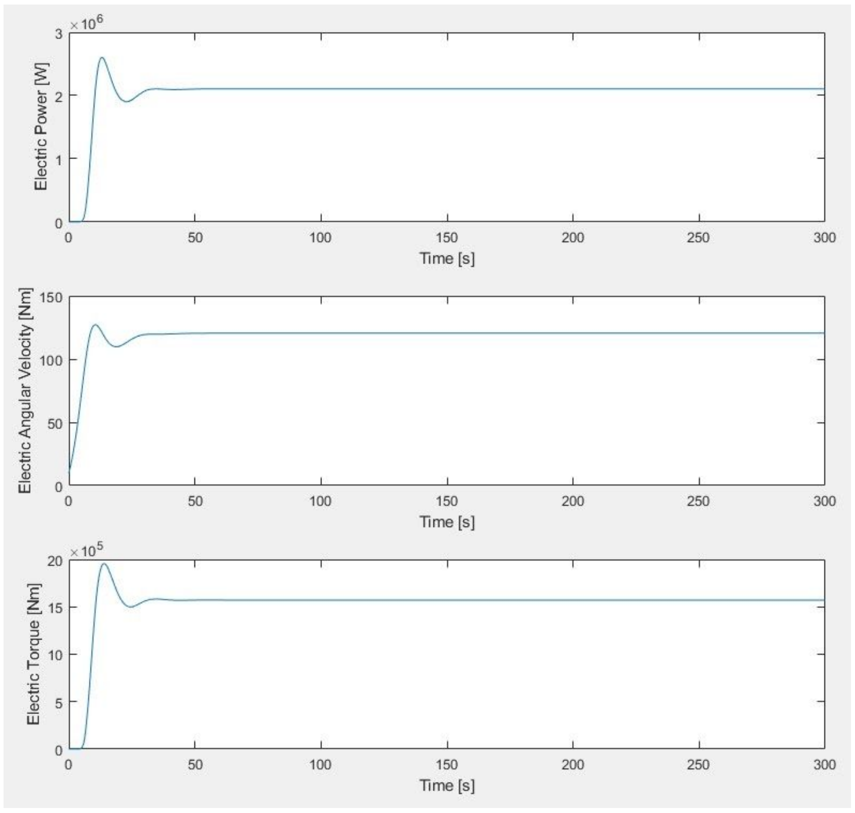


Figure 4.3.2. Plot of P_e, ω_e and T_e

Task 4.4:

Different from the simulation on Task 4.3, we are to simulate with a fluctuating wind speed around 10 m/s. The fluctuation is created via code using a random generated number with rand() and we set up the lower and upper limit to ±0.5 m/s. To control the result of the RNG, we used the default seed from matlab. We set the wind speed course for 300s, converted the array into a time-series format and exported it into a .mat file. The .mat file is then imported to our simulink model. The simulink block stays almost identical to Figure 4.3.1, the only difference being the Wind Speed block is changed into a "from File" block. The result of the simulation can be seen in Figure 4.4.1

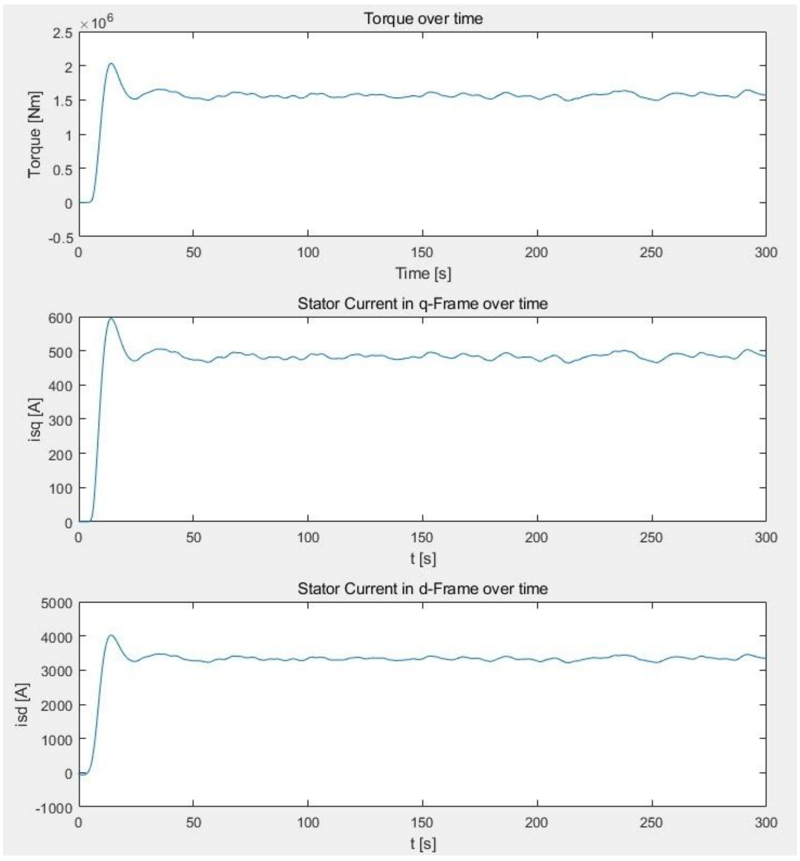


Figure 4.4.1. Plot of P_e, ω_e and T_e

In comparison to Figure 4.3.2, the resulting plot with fluctuation wind speed results in a fluctuation of values. Nonetheless, throughout the whole course the value stays relatively to the desired value. The Figure 4.4.2 shows the relation between P_e with ω_e as well as C_p with T_e.

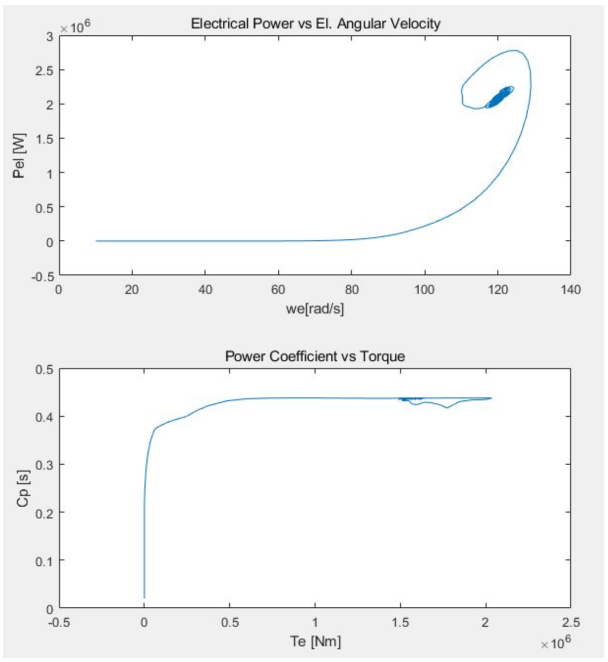


Figure 4.4.2. Plot of P_e, ω_e and T_e

The figure shows that both P_e with ω_e as well as C_p and T_e converges to a desired value. But due to the fluctuations no definite point and a spiral due to the values moving back and forth is observed.

For the task 4.4.e we are to compare the minimal and maximal value of Pe with the expected Pe from theory. In order to find the minimal and maximal value of the wind speed, we would have to find it using the min() and max() function. The resulting min and max value of the wind speed are **9.5046** and **10.4961** m/s. We then use the same method to find the value of P_e, the only difference is that we need to start searching the value after the transient has passed. The resulting min and max P_e are **1.923** and **2.239** MW. To check whether the min/max wind speed value does not appear during transient, we looked up the index using find().

The P_e from theory can be achieved by following the formula stated in Slide 54. The resulting min and max from theory are **1.6739** and **2.4229** MW. The results are quite far from each other.