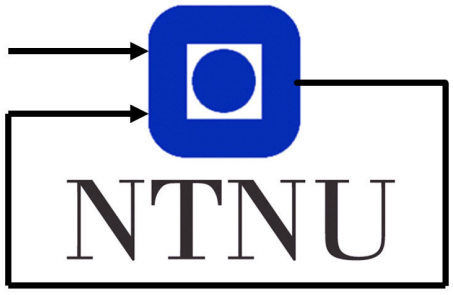
Discrete Kalman Filter Applied to a Ship Autopilot

Group 3
Henrik Dobbe Flemmen - 477564
Jan Fijalkowski - 478412
Kevin Kaldvansvik - 478460

November, 2018

TTK4115 – Linear System Theory



Department of Engineering Cybernetics

Abstract

This report is dedicated to the boat lab in TTK4115, Linear System Theory. We have used Simulink and MATLAB to simulate and control a boat using the discrete Kalman filter.

Contents

1 Problem 5.1 - Identification of the boat parameters

1.1 a) The transfer function from δ to ψ

The system model for the ship can be stated as eq. (1) using the Nomoto-approximation to simplify the yaw dynamics in the model.

$$\dot{\xi}_w = \psi_w \tag{1a}$$
$$\dot{\psi}_w = -\omega_0^2 \xi_w - 2\lambda\omega_0\psi_w + K_w w_w \tag{1b}$$
$$\dot{\psi} = r \tag{1c}$$
$$\dot{r} = -\frac{1}{T}r + \frac{K}{T}(\delta - b) \tag{1d}$$
$$\dot{b} = w_b \tag{1e}$$
$$y = \psi + \psi_w + v \tag{1f}$$

Where ψ is the average heading, the ψ_w is a high-frequency component due to wave disturbance and the r is the rate of change for the average heading. The w_w is the wave disturbance, w_b is the current disturbance, the b is the bias to the rudder angle and K is a gain.

To calculate the transfer function from δ to ψ parameterized by T and K, we substituted the average heading rate given in eq. (1c) in to the equation for the rate of change of heading given in eq. (1d) which gave the following equation

$$\ddot{\psi} = -\frac{1}{T}\dot{\psi} + \frac{K}{T}(\delta - b) \tag{2}$$

Assuming that there are no disturbances in the model means that we can set the rudder bias b equal to zero. By taking the laplace transform of eq. (2) and rearranging the equation to give the transfer function from δ to ψ yields

$$\mathcal{L}\{\ddot{\psi}(t)\}(s) = \mathcal{L}\{-\frac{1}{T}\dot{\psi}(t) + \frac{K}{T}\delta(t)\}(s) \tag{3a}$$
$$(s^2 + \frac{1}{T}s)\psi(s) = \frac{K}{T}\delta(s) \tag{3b}$$
$$\frac{\psi}{\delta}(s) = \frac{K}{T}\frac{1}{s^2 + \frac{1}{T}s} \tag{3c}$$
$$H(s) = \frac{K}{s(Ts+1)} \tag{3d}$$

1.2 b) Identifying the boat parameters T and K

Applying the sine input with $\omega_1 = 0.005(rad/s)$ and $\omega_1 = 0.05(rad/s)$ we got the two responses shown in fig. 1 and fig. 2.

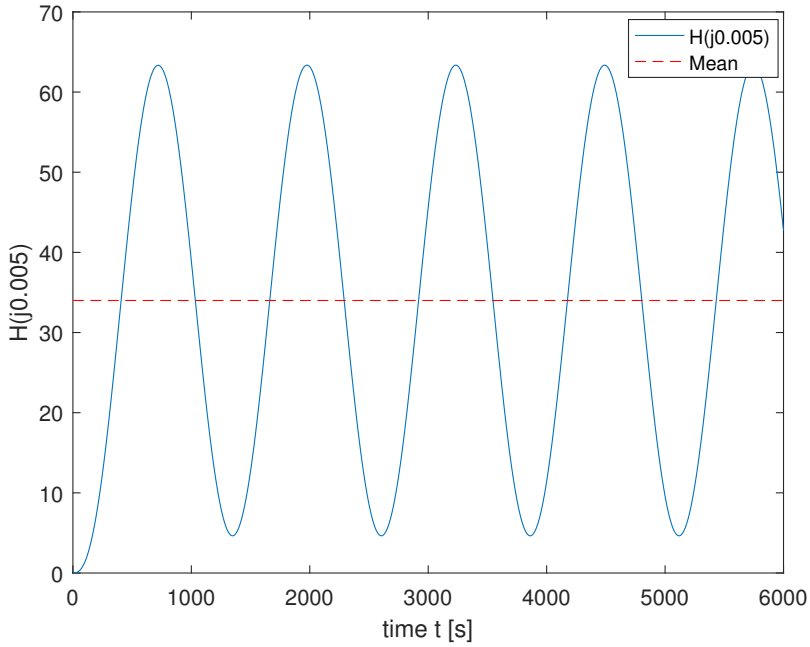


Figure 1: Response with $\omega_1 = 0.005(rad/s)$

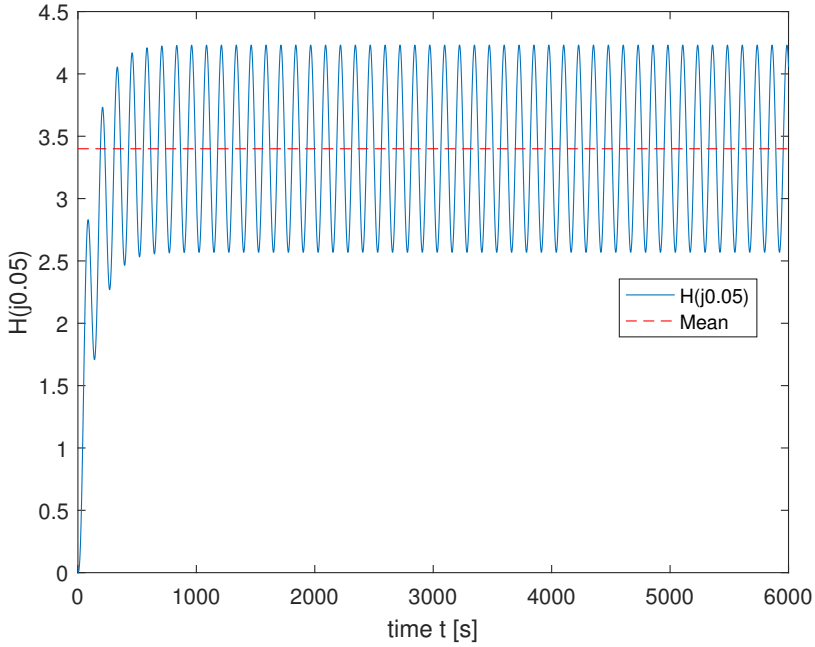


Figure 2: Response with $\omega_2 = 0.05(rad/s)$

In order to find the parameters T and K in eq. (3d), determining the amplitude of the responses is needed. Extracting the amplitude from the response was done using the code shown in appendix B.1. The code first opens the logged data from the file called $5b\_05.mat$. Then by accessing the data, calculation of the amplitude can be done by subtracting the max value of the signal in the oscillatory region by the mean value of the signal.

This gave us the following values for the amplitude which equals to $|H(j\omega)|$

$$A_1 = |H(j\omega_1)| = 29.3582 \tag{4a}$$
$$A_2 = |H(j\omega_2)| = 0.8310 \tag{4b}$$

With the corresponding values ω_1 and ω_2 the parameters T and K can now be found as shown

$$A_1 = \left| \frac{K}{j\omega_1(Tj\omega_1 + 1)} \right| \tag{5a}$$
$$= \frac{K}{\sqrt{\omega^2(T^2{\omega_1}^2 + 1)}} \tag{5b}$$
$$= \frac{K}{\omega_1\sqrt{(T^2{\omega_1}^2 + 1)}} \tag{5c}$$
$$\tag{5d}$$

The same result applies for the amplitude A_2

$$A_2 = \frac{K}{\omega_2\sqrt{(T^2{\omega_2}^2 + 1)}} \tag{6}$$

Solving the equations in eq. (5) with respect to the desired parameters T and K yields

$$K = A_1\omega_1\sqrt{{\omega_1}^2(T^2{\omega_1}^2 + 1)} \tag{7a}$$
$$T = \frac{\sqrt{K^2 - {A_2}^2{\omega_2}^2}}{A_2{\omega_2}^2} \tag{7b}$$

If we now insert eq. (7a) into eq. (7b) we get the following equation for the parameter T

$$T = \frac{\sqrt{{A_1}^2{\omega_1}^2(T^2{\omega_1}^2 + 1) - {A_2}^2{\omega_2}^2}}{A_2{\omega_2}^2} \tag{8a}$$
$$T^2{A_2}^2{\omega_2}^4 = {A_1}^2{\omega_1}^2(T^2{\omega_1}^2 + 1) - {A_2}^2{\omega_2}^2 \tag{8b}$$
$$T = \sqrt{\frac{{A_1}^2{\omega_1}^2 - {A_2}^2{\omega_2}^2}{{A_2}^2{\omega_2}^4 - {A_1}^2{\omega_1}^4}} \tag{8c}$$

Inserting the amplitudes A_1 and A_2 from eq. (10) in to eq. (8c) and then substituting this value for T in to eq. (7a) gives us the following values for the parameters T and K

$$T = 72.4391(s) \tag{9a}$$
$$K = 0.1561 \tag{9b}$$

This time constant is reasonable as it takes a lot of time to change the course to a big boat.

The simulink diagram for simulating the sine wave and recording the data is shown in figure A.1.

1.3 c) Identifying T and K with waves and measurement noise

Applying the sine input with With $\omega_1 = 0.005(rad/s)$ and $\omega_1 = 0.05(rad/s)$ with waves and measurement noise turned on, resulted in the two responses shown in fig. 3

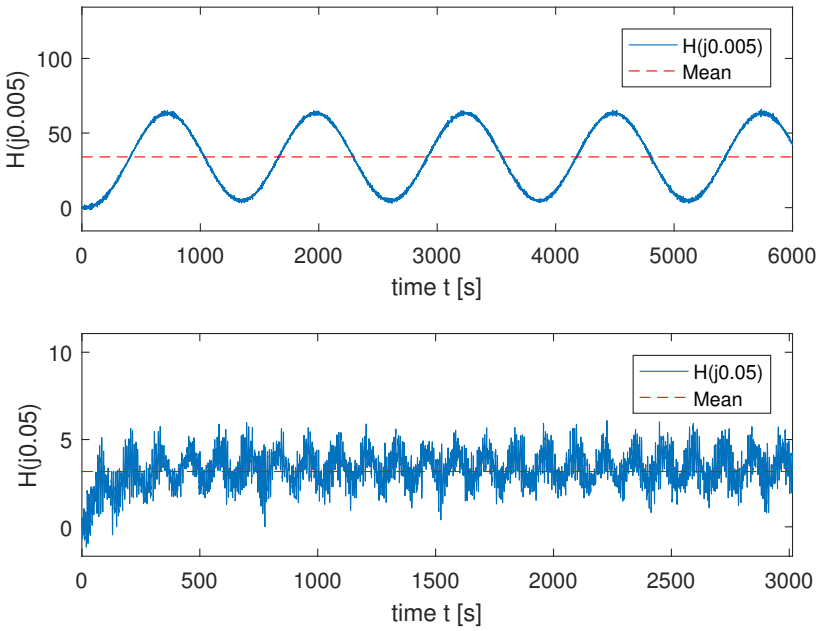


Figure 3: Responses with respectively $\omega_1 = 0.005(rad/s)$ and $\omega_2 = 0.05(rad/s)$ with waves and measurement noise

Using the same code as earlier (appendix B.1) to determine the amplitudes of the responses, gave us the following new values for the amplitudes which equals to $|H(j\omega)|$

$$A_1 = |H(j\omega_1)| = 30.9924 \tag{10a}$$
$$A_2 = |H(j\omega_2)| = 3.1688 \tag{10b}$$

Substituting these new values for A_1 and A_2 in to eq. (8) gave us the following values for T and K

$$T = 4.1876j(s) \tag{11a}$$
$$K = 0.1549 \tag{11b}$$

The signal is very noisy, which in turn makes the amplitude somewhat undefined. If we take the maximum value minus the minimum value, we would get a different amplitude than we did without the noise. We could try to pass the signal through some kind of low pass filter, but any non-ideal filter would also interfere with the actual signal. Therefore would the resulting amplitude depend on what kind of filter we would use. From all this, it seems rather hard to read meaningful amplitudes of the noisy signal. We tried estimating T and K from the amplitude from the max of the sum of the noise and the signal. We then got complex results from the calculations, which makes no sense. From all this we can conclude that it is not possible to get good, consistent estimates for the boat parameters from the readings with noise.

A solution to this problem might be to include a low pass filter to the response. This would remove the high and undesired frequencies, but to do this we would have to find out which frequencies the signal consists of. This could be done by taking the fast Fourier transform of the discrete signal and then by looking at the magnitude of the Fourier transform we could find which frequencies the response consists of. From this we could design a low pass filter to filter out the found high undesired frequencies.

1.4 d) Discussion of the model

The step response on the rudder angle δ of the boat was simulated using a step block in simulink with the step time set to zero, initial value set to 0 and final value set to 1. The data was exported to Matlab to plot the result.

The implementation of the simplified model is shown in B.3. The function tf lets us specify that the transfer function depends on the variable s, and the function step gives us the unit step response of the system transfer function.

The two responses on respectively the boat and the model is shown in fig. 4

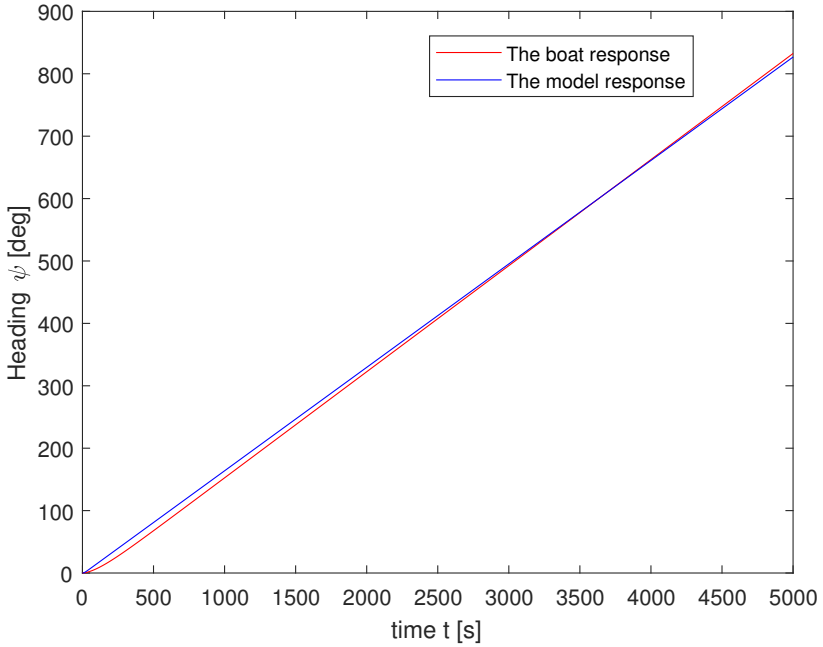


Figure 4: The response on the average heading ψ for the boat and the simplified model when a step input of $\delta = 1(deg)$ is applied to the rudder angle.

Looking at fig. 4 the model response has a minor deviation in the start until about $t_1 = 2500(s)$. From t_1 until $t_2 = 4500(s)$ the curve seems to fit rather perfectly. The boat response seems to lag behind the model in the start which is reasonable due to physical restrictions based on a limited possible input to the motors that drive the rudder to a desired angle. The nomoto-approximation seems to be a rather good approximation for the yaw dynamics of the ship, if we assume that this is a big ship or perhaps even a big cruise ship.

From now on, measurement noise are included in all simulations.

2 Problem 5.2 - Idendification of wave spectrum model

2.1 a) Estimate of the Power Spectral Density

The power spectrum density (PSD) of the heading noise $S_{\psi_w}(\omega)$ can be estimated using the matlab function pwelch as seen in appendix B.2. In the matlab script it is shown how the data were converted and handled to give the correct PSD with the desired unit power s/rad.

The result is shown in fig. 5 where the PSD is plotted against the angular frequencies $\omega \in [0, 2\pi]$.

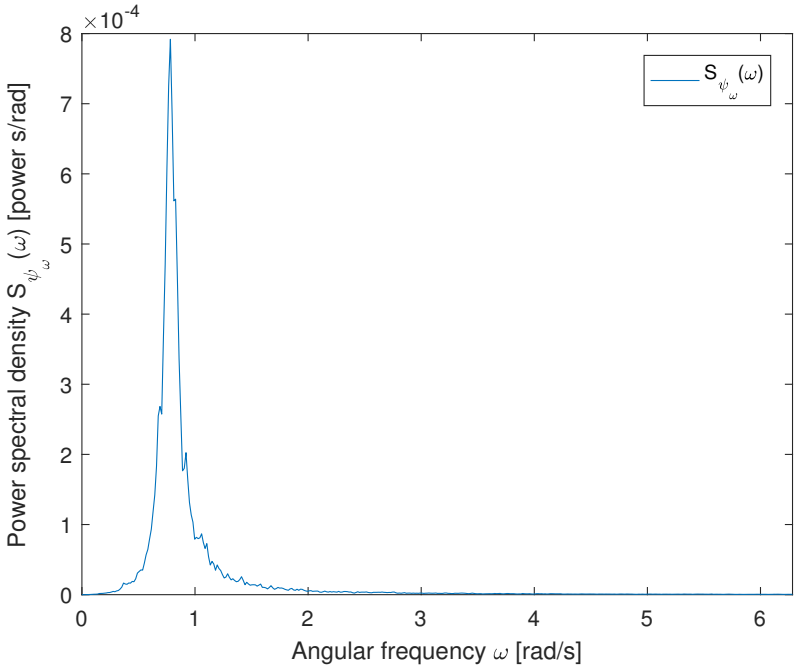


Figure 5: The estimated power spectrum density function of ψ_w, $S_{\psi_w}(\omega)$.

The PSD is a very powerful tool to determine which angular frequency of the waves that are most common and impacts the ship. From the figure we can see that an angular frequency of about $0.8(rad/s)$ is where we have waves that interferes with the heading ψ the most.

2.2 b) The analytical expression

To find the transfer function from w_w to ψ_w which gives us the possibility to analyze the wave response model, we have to analyze the two dynamic equations 1a and 1b. Taking the Laplace transform of these two equations and substituting

the first equation in to the second equation yields

$$s\psi_w = -\frac{{\omega_0}^2\psi_w}{s} - 2\lambda\omega_0\psi_w + K_w \tag{12a}$$

$$\psi_w(s + \frac{{\omega_0}^2}{s} + 2\lambda\omega_0) = K_w w_w \tag{12b}$$

$$\frac{\psi_w}{w_w} = \frac{K_w}{s + \frac{{\omega_0}^2}{s} + 2\lambda\omega_0} \tag{12c}$$

$$H(s) = \frac{K_w s}{s^2 + 2\lambda\omega_0 s + {\omega_0}^2} \tag{12d}$$

The analytic expression for the power spectral density function of ψ_w is denoted as $P_{\psi_w}(\omega)$. To find this analytic expression we use the following relationship between the input and the output of a transfer function $H(j\omega)$ [3, p. 330].

$$P_{\psi_w}(\omega) = \left|H(\omega)\right|^2 P_{w_w}(\omega) = H(\omega)H^*(\omega)P_{w_w}(\omega) \tag{13}$$

In order to find the PSD, the transformation $s = j\omega$ is applied to the transfer function $H(s)$ in eq. (12d), which gives us the transfer function in the frequency domain

$$H(\omega) = \frac{K_w j\omega}{(j\omega)^2 + 2\lambda\omega_0 j\omega + \omega_0^2} \tag{14}$$

The complex conjugate of the transfer function is denoted as $H^*(\omega)$ which is the same as $H(-\omega)$.

The waves are considered as a disturbance to the system and are defined as a zero mean white noise process with unity variance. White signals has the following property for auto-correlation R_{ww} [3, p. 332]

$$R_{ww}(\tau) = \sigma_w^2\delta(\tau) \tag{15}$$

If we take the Fourier transform of eq. (15) it gives us the following formula for the PSD of the white noise process

$$P_{ww}(\omega) = \sigma_w^2 \tag{16}$$

Thus the PSD of of the wave disturbances with unity variance is given by

$$P_{w_w}(\omega) = 1(powers/rad) \tag{17}$$

The PSD for ψ_w is now given by

$$P_{\psi_w}(\omega) = H(\omega)H^*(\omega) \tag{18a}$$

$$= \frac{K_w j\omega}{(j\omega)^2 + 2\lambda\omega_0 j\omega + {\omega_0}^2} \cdot \frac{K_w(-j\omega)}{(-j\omega)^2 + 2\lambda\omega_0(-j\omega) + {\omega_0}^2} \tag{18b}$$

$$= \frac{K_w^2\omega^2}{({\omega_0}^2 - \omega^2)^2 + 4\lambda^2{\omega_0}^2\omega^2} \tag{18c}$$

2.3 c) Identifying w_0 and intensity σ^2

From the estimated power density spectrum S_{ψ_w} shown in fig. 5, ω_0 can be found by locating the frequency that maximizes the PSD function S_{ψ_w}. The corresponding intensity σ^2 can be found by finding the corresponding S_{ψ_w} amplitude at that frequency ω_0. The code for determining these two values is shown in appendix B.4. The code calculates the modal peak frequency ω_0 by using the max function which returns the index at which we can find the corresponding ω_0. It also returns the maximal value of the PSD which is the intensity σ^2. The script gave us the following values for ω_0 and σ^2

$$\omega_0 = 0.7823 rad/s \tag{19a}$$

$$\sigma^2 = 0.0281^2 (powers/rad) \tag{19b}$$

These values can also be seen and verified in fig. 5.

2.4 d) Identifying the damping factor

The last step to find the complete model is to find the damping parameter λ. This is done by tweaking the parameter λ in the expression for $P_{\psi_w}(\omega)$ until the numerical PSD matches the estimated PSD $S_{\psi_w}(\omega)$ shown in fig. 5. Since tweaking this by hand can be a tedious task, using the least square method for data fitting is a good option. This method lets us choose the parameter λ such that the sum of the residuals is minimal.

The parameter K_w is now defined as $2\lambda\omega_0\sigma$ which gives us the following transfer function

$$H(s) = \frac{2\lambda\omega_0\sigma s}{s^2 + 2\lambda\omega_0 s + {\omega_0}^2} \tag{20}$$

The curve-fitting Matlab function *lsqcurvefit* was used to determine the best possible λ. The syntax is given by *lsqcurvefit(fun,x0,xdata,ydata)*. The parameter *fun* is the function of which we wanted to curve-fit, namely $P_{\psi_w}(\omega)$. The implementation of the function and the curve-fit is shown in appendix B.5. The resulting value of λ that made the best possible fit for the estimated PSD is

$$\lambda = 0.0928 \tag{21}$$

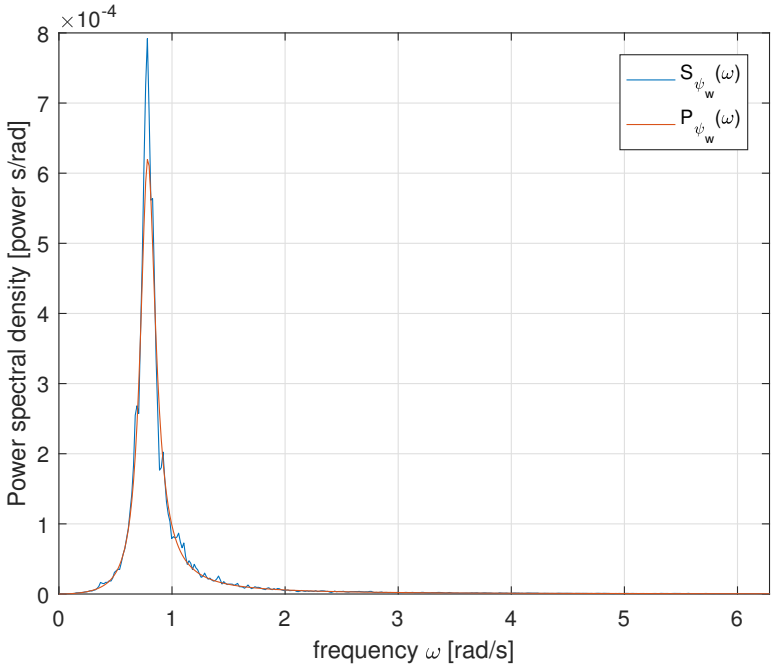


Figure 6: The estimated power spectrum density function $S_{\psi_w}(\omega)$ plotted against $P_{\psi_w}(\omega)$

The analytical PSD found by using least square method is plotted against the estimated PSD in fig. 6. By the figure one can notice that the analytic PSD are not able to completely follow the estimated $S_{\psi_w}(\omega)$, which results in a lower intensity for the analytic PSD. The new values for ω_0 and σ^2 is given by

$$\omega_0 = 0.7823(rad/s) \tag{22a}$$

$$\sigma^2 = 0.0249^2(powers/rad) \tag{22b}$$

The modal peak frequency ω_0 is the same, but the intensity σ^2 is a bit smaller than before. This is reasonable because the numerical PSD is smooth and does not carry any oscillations, which means that the variance is smaller.

Since the amplitude carries very important information about the intensity of the signal, we tried to reduce the dampening factor λ even further to force the analytic PSD $P_{\psi_w}(\omega)$ to follow the maximum intensity of $S_{\psi_w}(\omega)$. This did not work however, the resulting PSD did not manage to improve the intensity.

3 Problem 5.3 - Control system design

3.1 a) Design of PD controller

To control the ship to a desired course angle ψ_r we use a limited PD controller on the form

$$H_{pd}(s) = K_{pd}\frac{1+T_d s}{1+T_f s} \tag{23}$$

The rest of this task is to determine the constants in the PD controller, namely K_{pd}, T_d and T_f.

Using the transfer function $H(s)$ based on no disturbances, from the rudder angle δ to the heading angle ψ in eq. (3d), gives us the following serial compensation equation for $H_0(s) = H_{pd}(s)H(s)$

$$H_0(s) = \frac{KK_{pd}(1+T_d s)}{s(1+Ts)(1+T_f s)} \tag{24}$$

Getting rid of the large inconvenient time constant T in the system can be done by setting $T_d = T$. In section 1.2 we found $T = 72.4391$. Canceling out equal factors gives us the following transfer function for $H_0(s)$

$$H_0(s) = \frac{KK_{pd}}{s(1+T_f s)} \tag{25}$$

The value of K was also found in section 1.2 and was $K = 0.1561$. In the task it is given that the cut-off frequency of H_0, ω_c and the phase margin ϕ should be chosen respectively to be approximately $\omega_c = 0.10$ (rad/s) and $\phi = 50°$. To determine the K_{pd} and T_f, we insert the wanted ω_c and phase margin ϕ into our model of the closed loop system eq. (25). We will calculate the amplitude and the phase of the transfer function, and thereby get two different equations width the two last unknowns.

By inserting $j\omega$ for s in eq. (25) we get

$$H_0(j\omega) = \frac{KK_{pd}}{j\omega(1+T_f j\omega)} \tag{26}$$

We will multiply with the complex conjugate in the numerator and the denominator, in order to eliminate the complex number in the numerator. We then get:

$$H_0(j\omega_c) = \frac{KK_{pd}(-T_f {\omega_c}^2 - j\omega_c)}{\omega_c^2(1+T_f^2\omega_c^2)} \tag{27}$$

To get the expression for the phase angle we take the arcus tangens of the imaginary part divided by the real part of the expression. All constants and variables that appear in both, will cancel against each other and we get:

$$\angle H_0(j\omega_c) = arctan(\frac{-\omega_c}{-T_f {\omega_c}^2}) = arctan(\frac{1}{T_f\omega_c}) \tag{28}$$

By algebraic manipulation we can get an expression for T_f as shown:

$$T_f = \frac{1}{\omega_c \cdot tan(\angle H_0(j\omega_c))} \tag{29}$$

We now have an expression for T_f given solely by known or given constants. In order to deduce an expression for K_{pd}, we will need to calculate the amplitude of $H_0(s)$.

$$\left|H_0(j\omega_c)\right| = \left|\frac{KK_{pd}}{j\omega_c(1+T_f j\omega_c)}\right| = \frac{KK_{pd}}{\left|j\omega_c + T_f {\omega_c}^2\right|} = \frac{KK_{pd}}{\sqrt{{\omega_c}^2 + ({T_f {\omega_c}^2})^2}} \tag{30}$$

We can sort this expression to get the expression for K_{pd}, knowing that since we are computing the absolute value of the transfer funtion H_0 evaluated at ω_c we know that $|H_0(j\omega_c)| = 1$.

$$K_{pd} = \frac{\omega_c\sqrt{1-(T_f\omega_c)^2}}{K} \tag{31}$$

Now the equations eq. (29) and eq. (31) gives the correct values and it is just to insert the constants and requirements in the equations. The phase margin, ϕ, was required to be $50°$. The corresponding phase angle at cutoff frequency will be given by

$$\angle H_0(j\omega_c) = \phi + (-180°) \tag{32}$$

which numerically means that $\angle H_0(j\omega_c) = -130$. The ω_c is given in the assignment to be set to $0.1 rad/s$. The K is already calculated in eq. (11b) to be 0.1561. By first calculating T_f from eq. (29), and then inserting the result into eq. (31) we get both T_f and K_{pd} as in eq. (33).

$$T_f = 8.391(s) \tag{33a}$$
$$K_{pd} = 0.8363 \tag{33b}$$

This yields a controller with the specified requirements. We can verify our calculations by plotting a bode diagram with the Matlab code in appendix B.7. We can see that we have the correct phase margin at the correct cut-off frequency in fig. 7 as specified in the task.

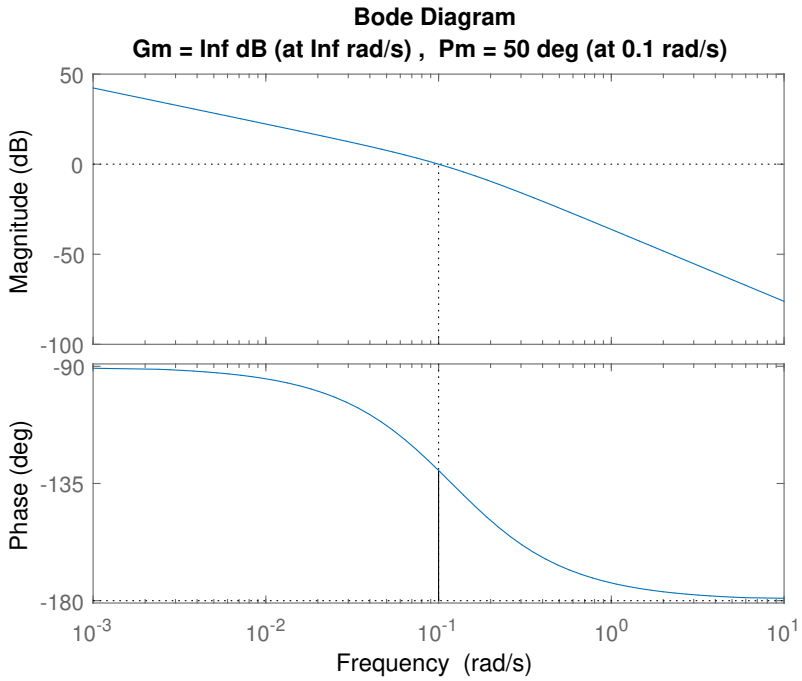


Figure 7: The bode diagram for the complete system $H_0(s)$. From the diagram we can see that the cut-off frequency ω_c is 0.1 rad/s. The resulting phase margin ϕ is 50 °.

3.2 b) Implementation of PD controller and simulation without disturbance

The PD controller developed in section 3.1 is implemented as a transfer function block in simulink shown in appendix A.2. The model is linearized around $\psi = 0$, and large deviations from this point will not be feasible for our model. To make sure that the rudder angle stayed within its limits, a saturation block was added in front of the controller as seen in appendix A.3. This limited the rudder angle to $[-45°, 45°]$.

The step function at the reference was set to 30°at time $t = 0$. All disturbances was turned off except measurement noise, and the result of the simulation is shown in fig. 8.

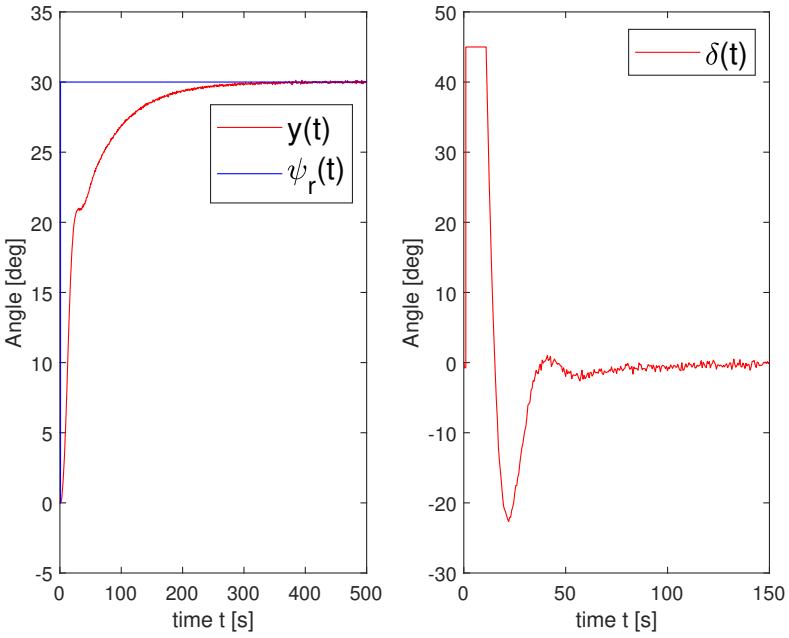


Figure 8: The rudder angle(right) and the heading angle(left) with measuring noise and without disturbances.

We observe in fig. 8 that ψ converges on 30°, as the reference was set, and we can thereby conclude that the autopilot is working for good weather conditions. The heading angle is smooth, which is caused by the controller which is somewhat overdamped. We can see from the graph on the right that the rudder angle δ changes direction before the heading has reached its reference. This is caused by the derivative effect, and without it we would see an overshoot in the course, y. In addition, we use the D effect of the PD-controller to cancel out the very large time constant of the system. This is making the overall system faster and the analysis easier. From eq. (3d) we can observe that the transfer function of the ship has an integrator and a time constant in its denominator. This will cause the amplitude to fall quickly with increasing frequency, limiting our bandwidth. With the PD-controller, we can lift the amplitude(and phase) and thereby increase our bandwidth. Which mean that we now can follow quicker disturbances and heading changes.

The convergence of the heading on the reference, is somewhat slow. It is only after approximately $300(s)$ or $5(min)$ that the angle is relatively close to the desired reference. This is not a large surprise, as large ships takes long time to turn. This effect is further increased because of the limitations in the rudder angle.

3.3 c) Simulation with current disturbance

Now we will run the same simulation as previously, but now we add the current disturbance to the system. The same reference heading is used, i.e $\psi = 30°$.

The resulting response is shown in fig. 9.

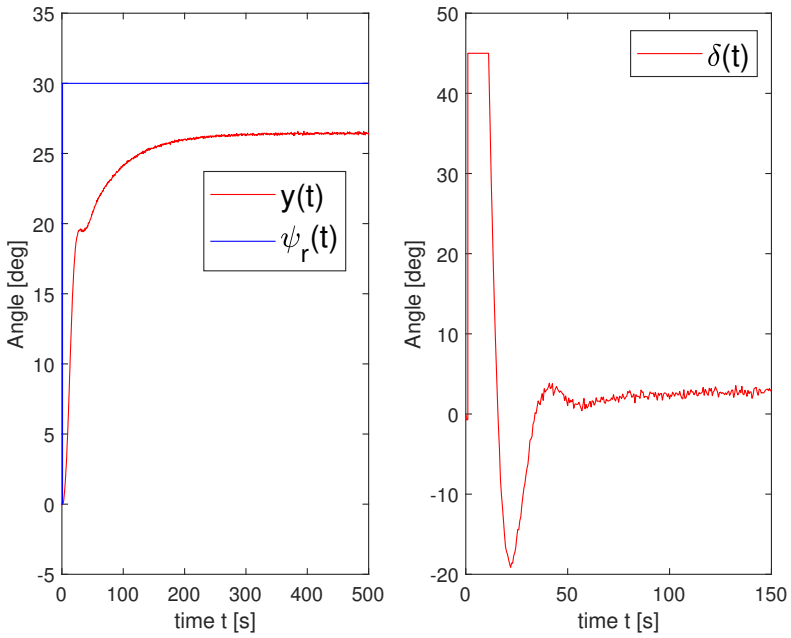


Figure 9: The measured heading y and the reference heading ψ_r shown in the plot to the left. To the right, the rudder angle δ. Measurement noise and current disturbance are included.

In the plot we can see a stationary deviation between the measured heading y and the reference heading ψ_r. This deviation can easily be explained by having a look at eq. (1d). In this equation the influence of the current disturbance is modeled as a bias b on the rudder angle. In the design of our autopilot we negated this rudder bias, thus the autopilot has no idea that this bias exists. The consequence of this is that the autopilot provides a smaller rudder angle reference to the ship than it should. Thus explaining why the autopilot are not able correct the undesired stationary deviation. The autopilot is therefore not working as it should in this case.

To remove this deviation one could try to measure or estimate this bias b and then feed forward it to the rudder angle to cancel the rudder bias term, resulting in the equation we used in the autopilot design having no disturbances.

Another tempting way of removing the stationary deviation is using a PID-controller instead of a PD-controller. The serial compensation transfer function would take the following form

$$H_0(s) = \frac{1+T_i s}{T_i s}\frac{KK_{pd}}{s(1+T_f s)} = \frac{KK_{pd}(1+T_i s)}{s^2 T_i(1+T_f s)} \tag{34}$$

Looking at this transfer function we notice two poles located at origo, which means that the resulting system is unstable. For a system to be considered marginally stable all poles λ_i should have real parts less or equal to 0, where

there are no repeated poles having real part of 0. In our system we have a repeated pole with real part of 0. Thus, the PID controller would result in an unstable system.

3.4 d) Simulation with wave disturbance

Now we will test our controller with no current disturbance but with wave disturbance present. The simulation was still done with a step response at 30°at $t = 0$ at the reference. And we can see the result in fig. 10.

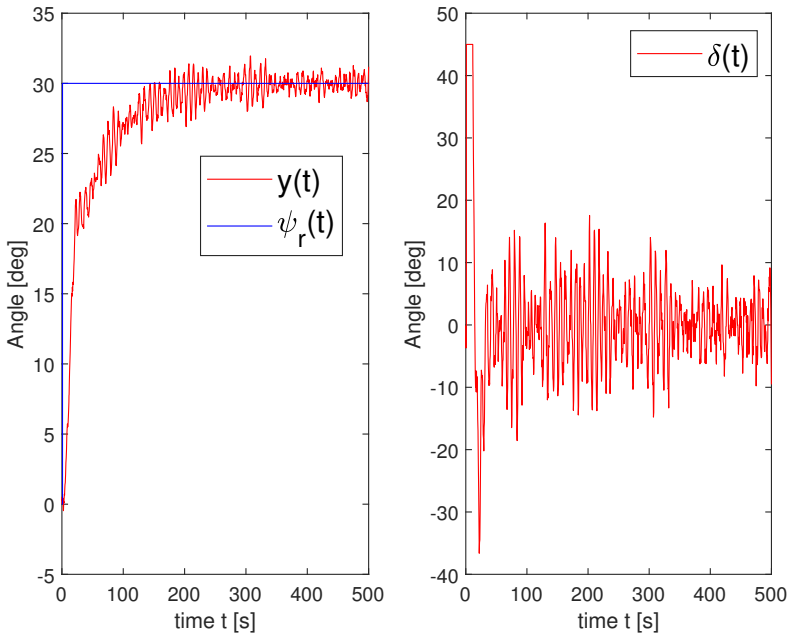


Figure 10: The result of running the PD-controller in a system with wave disturbances. The heading direction to the left and the rudder angle to the right.

We see in fig. 10 that the system is very polluted by wave disturbances. Wave disturbances is, in this model, modeled as white noise summed in the heading. Opposed to what we saw in section 3.3 we now see that the course converges to the correct direction.

We can also observe from that fig. 10 that we have large and quick movements from 15°-30°at the rudder at only 4-5 (s), due to the controllers attempt to compensate for the wave disturbances. This will wear down the rudder and the connected mechanic faster than feasible. In order to go further with our model and controller we will need to filter out the wave disturbance from the measurement, so that the controller does not compensate for the waves.

4 Problem 5.4 - Observability

4.1 a) State space model

We can use the mathematical model eq. (1) to determine the state vector $\mathbf{x}$, the input u, the output y and the noise vector $\mathbf{w}$. They are given as

$$\mathbf{x} = \begin{bmatrix} \xi_w \\ \psi_w \\ \psi \\ r \\ b \end{bmatrix} \quad , \quad y = \psi_w + \psi + v \quad , \quad u = \delta \quad \text{and} \quad \mathbf{w} = \begin{bmatrix} w_w \\ w_b \end{bmatrix} \tag{35}$$

Our system can be written as

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{B}u + \mathbf{E}\mathbf{w} \\ \mathbf{y} &= \mathbf{C}\mathbf{x} + v \end{aligned} \tag{36}$$

and our state space model will have the following form

$$\begin{aligned} \dot{\mathbf{x}} = \begin{bmatrix} \dot{\xi}_w \\ \dot{\psi}_w \\ \dot{\psi} \\ \dot{r} \\ \dot{b} \end{bmatrix} &= \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ -{\omega_0}^2 & -2\lambda\omega_0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -\frac{1}{T} & -\frac{K}{T} \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}}_{\mathbf{A}} \mathbf{x} + \underbrace{\begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{K}{T} \\ 0 \end{bmatrix}}_{\mathbf{B}} u + \underbrace{\begin{bmatrix} 0 & 0 \\ K_w & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}}_{\mathbf{E}} \mathbf{w} \\ \mathbf{y} &= \underbrace{\begin{bmatrix} 0 & 1 & 1 & 0 & 0 \end{bmatrix}}_{\mathbf{C}} \mathbf{x} + v \end{aligned} \tag{37}$$

4.2 b) No disturbance

Without disturbances our model will not include the white noise processes w_w and w_b. In addition we need to exclude the high-frequency component ψ_w from wave disturbance. Our new state vector $\mathbf{x}$, input u and measurement y will then have the following form

$$\mathbf{x} = \begin{bmatrix} \psi \\ r \end{bmatrix} \quad \text{and} \quad y = \psi + v \tag{38}$$

Our system can now be written as

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{B}u \\ y &= \mathbf{C}\mathbf{x} + v \end{aligned} \tag{39}$$

and our state space model yields

$$\begin{aligned}\dot{\mathbf{x}} &= \begin{bmatrix}\dot{\psi}\\ \dot{r}\end{bmatrix} = \underbrace{\begin{bmatrix}0 & 1\\ 0 & -\frac{1}{T}\end{bmatrix}}_{\mathbf{A}}\mathbf{x} + \underbrace{\begin{bmatrix}0\\ \frac{K}{T}\end{bmatrix}}_{\mathbf{B}}u\\ y &= \underbrace{\begin{bmatrix}1 & 0\end{bmatrix}}_{\mathbf{C}}\mathbf{x} + v\end{aligned} \tag{40}$$

The observability matrix $\mathcal{O}$ for a system with n states is given by

$$\mathcal{O} = \begin{bmatrix}\mathbf{C}\\ \mathbf{CA}\\ \mathbf{CA}^2\\ .\\ .\\ \mathbf{CA}^{n-1}\end{bmatrix} \tag{41}$$

In our new state space model we have 2 states.

We can easily get the observability matrix $\mathcal{O}$ by using the MATLAB function $obsv(\mathbf{A}, \mathbf{C})$. The rank can be checked using the function $rank(\mathcal{O})$. How this was done can been seen in detail in appendix B.6. Putting in our $\mathbf{A}$ and $\mathbf{C}$ matrix from eq. (40) we get the following observability matrix

$$\mathcal{O} = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} \tag{42}$$

which clearly has full column rank $n = 2$. This means that all the states are available through linear combinations of the output y. This makes sense as our system does not depend on disturbances. This can also be seen from eq. (1).

4.3 c) Current disturbance

Now we want to examine if the system is observable with the current disturbances. We will add the bias to the rudder angle to the state vector, which implicit gives us the current disturbance w_b by differentiation. We have the same input and measurement as in eq. (38), but a new state vector and a disturbance w.

$$\mathbf{x} = \begin{bmatrix}\psi\\ r\\ b\end{bmatrix} \qquad \text{and} \qquad w = w_b \tag{43}$$

Our new system can be written as

$$\begin{aligned}\dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{B}u + \mathbf{E}w\\ y &= \mathbf{C}\mathbf{x} + v\end{aligned} \tag{44}$$

Our new state space matrix $\mathbf{A}$, the input matrix $\mathbf{B}$, disturbance matrix $\mathbf{E}$ and the output matrix $\mathbf{C}$ will have the form

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{\psi} \\ \dot{r} \\ \dot{b} \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 & 0 \\ 0 & -\frac{1}{T} & -\frac{K}{T} \\ 0 & 0 & 0 \end{bmatrix}}_{\mathbf{A}} \mathbf{x} + \underbrace{\begin{bmatrix} 0 \\ \frac{K}{T} \\ 0 \end{bmatrix}}_{\mathbf{B}} u + \underbrace{\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}}_{\mathbf{E}} w$$
$$y = \underbrace{\begin{bmatrix} 1 & 0 & 0 \end{bmatrix}}_{\mathbf{C}} \mathbf{x} + v \tag{45}$$

Our new state space model will have 3 states. This means that we have to compute $\mathbf{A}^2$ in order to check if the observability matrix has full column rank. We will proceed as we did in section 4.2 in order to compute the observability matrix

$$\mathcal{O} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -0.0138 & -0.0022 \end{bmatrix} \tag{46}$$

This matrix clearly has full column rank $n = 3$. Which means that the system without wave disturbance is observable. This makes sense as the removed wave disturbance does not affect the other states as seen in eq. (1).

4.4 d) Wave disturbance

Now we want to determine if the system is observable with the wave disturbances. We will add the high-frequency component ψ_w to the measurement y, and the disturbance matrix will only contain the wave disturbance. Our new state vector, measurement matrix and disturbance will be defined as

$$\mathbf{x} = \begin{bmatrix} \xi_w \\ \psi_w \\ \psi \\ r \end{bmatrix} \quad , \quad y = \psi_w + \psi + v \quad \text{and} \quad w = w_w \tag{47}$$

Our system will have the same form as in section 4.3, but since we have different vectors and scalers we will have different matrices.

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{\xi_\omega} \\ \dot{\psi_\omega} \\ \dot{\psi} \\ \dot{r} \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 \\ -{\omega_0}^2 & -2\lambda\omega_0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -\frac{1}{T} \end{bmatrix}}_{\mathbf{A}} \mathbf{x} + \underbrace{\begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{K}{T} \end{bmatrix}}_{\mathbf{B}} u + \underbrace{\begin{bmatrix} 0 \\ K_w \\ 0 \\ 0 \\ 0 \end{bmatrix}}_{\mathbf{E}} w$$
$$\mathbf{y} = \underbrace{\begin{bmatrix} 0 & 1 & 1 & 0 \end{bmatrix}}_{\mathbf{C}} \mathbf{x} + v \tag{48}$$

Our new state space model will have 4 states. This means that we have to compute $\mathbf{A}^3$ in order to check if the observability matrix has full column rank. We will proceed as we did in section 4.2 in order to compute the observability matrix

$$\mathcal{O} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ -0.6120 & 0.1452 & 0 & 1 \\ -0.0889 & -0.5909 & 0 & -0.0138 \\ 0.3616 & -0.1747 & 0 & 1.9057 \cdot 10^{-4} \end{bmatrix} \tag{49}$$

The observability matrix has full column rank $n = 4$, which mean that the system without current disturbance is observable. Assuming that the rudder bias b is equal to zero when there is no current disturbances, we can also see from eq. (1) that the current disturbance does not affect the other states.

4.5 e) Wave and current disturbance

Now we want to find if the system is observable with both the current and the wave disturbance. We will use the matrices **A** and **C** from eq. (37) to find the observatillity matrix. We will proceed with the same method as in section 4.2.

$$\mathcal{O} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ -0,6120 & 0,1452 & 0 & 1 & 0 \\ -0,0889 & -0,5909 & 0 & -0,0138 & -0,0022 \\ 0,3616 & -0,1747 & 0 & 1,9057 \cdot 10^{-4} & 2,9748 \cdot 10^{-5} \\ 0,1069 & 0,3363 & 0 & -2,6308 \cdot 10^{-6} & -4,1066 \cdot 10^{-7} \end{bmatrix} \tag{50}$$

The observability matrix has full column rank $n = 5$. This means that for any unknown initial state $x(0)$ there exists a finite $t_1 > 0$ such that the knowledge of the input δ and the output y over $[0, t_1]$ suffices to determine uniquely the initial state $x(0)$. We can therefor determine the behavior of the entire system from the system's outputs and inputs in finit time. We can conclude that the disturbance does not affect the obersvability of this system, and we can construct an observer $\hat{x}(s)$ for any given state. This information is very useful later on in task 5.5.

5 Problem 5.5 - Discrete Kalman filter

5.1 a) Discretization of model

The continuous-time linear state-space model found in section 4 has the following form

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{B}u + \mathbf{E}\mathbf{w} \\ y &= \mathbf{C}\mathbf{x} + v \end{aligned} \tag{51}$$

where v and $\mathbf{w}$ are zero mean white noise processes with the process noise covariance matrix $\mathbf{Q}$ and measurement noise variance σ_v^2

$$\begin{aligned} \mathbf{w}(\mathbf{t}) &\sim \mathbf{N}(\mathbf{0}, \mathbf{Q}) \\ v(t) &\sim N(0, \sigma_v^2) \end{aligned} \tag{52}$$

There are some uncertainties regarding the $\mathbf{Q}$ matrix shown in the lab assignment, whether it is already discretized or not. Thus through this report we assume that this given matrix $\mathbf{Q}$ is in continuous time and is given by

$$\mathbf{Q} = \begin{bmatrix} 30 & 0 \\ 0 & 10^{-6} \end{bmatrix} \tag{53}$$

This model can be exact discretized with a sampling frequency of $f_s = 10(Hz)$, which gives a sampling period of $T = 0.1(s)$. The model then takes the following form

$$\begin{aligned} \mathbf{x}[\mathbf{k}+\mathbf{1}] &= \mathbf{A_d}\mathbf{x}[\mathbf{k}] + \mathbf{B_d}\mathbf{u}[\mathbf{k}] + \mathbf{w}[\mathbf{k}] \\ y[k] &= \mathbf{C_d}\mathbf{x}[\mathbf{k}] + v[k] \end{aligned} \tag{54}$$

where the discretized process noise $\mathbf{w}$[k] and discretized measurement noise $v[k]$ are given by

$$\begin{aligned} \mathbf{w}[\mathbf{k}] &\sim \mathbf{N}(\mathbf{0}, \mathbf{Q_k}) \\ v[k] &\sim N(0, R) \end{aligned} \tag{55}$$

and the discretized model parameters given by

$$\mathbf{A_d} = \mathbf{e}^{\mathbf{AT}} = \mathcal{L}^{-\mathbf{1}}\{(\mathbf{sI} - \mathbf{A})^{-\mathbf{1}}\}\Big|_{t=T} \tag{56a}$$

$$\mathbf{B_d} = \left(\int_{\tau=0}^{T} \mathbf{e}^{\mathbf{AT}}\mathbf{d}\tau\right)\mathbf{B} = \mathbf{A}^{-\mathbf{1}}(\mathbf{A_d} - \mathbf{I})\mathbf{B} \tag{56b}$$

$$\mathbf{C_d} = \mathbf{C} \tag{56c}$$

The reason the discretized system in eq. (54) takes the following form is that we chose to use *Loans method* [2, p. 126] to compute the numerical value for the discretized process noise covariance matrix $\mathbf{Q_k}$. This method takes the $\mathbf{E}$ matrix shown in eq. (51) and the continous process noise covariance matrix $\mathbf{Q}$ shown in eq. (55) and fuses it together in to one discretized process noise covariance matrix $\mathbf{Q_k}$. The Matlab script to compute $\mathbf{Q_k}$ is shown in appendix B.9.

The equations in eq. (56) holds if, and only if the matrix $\mathbf{A}$ is non-singular. This is the same as saying that the matrix should be invertible, and thus the same as saying that the matrix should have a non zero determinant. The $\mathbf{A}$ matrix in our case is singular, which means that we can not compute the exact inverse of $\mathbf{A}$. This can also be confirmed by running the Matlab command *det(A)*, which returns the value 0, thus verifies that the inverse can not be found. However this can be resolved by using the Moore-Penrose pseudoinverse function in Matlab. This function computes the best least squares solution to the inverse and is sufficient for our use. The inverse $\mathbf{A^{-1}}$ can therefore be computed by using the Matlab function *pinv(A)*.

However we can easily compute the exact parameters using the Matlab function *c2d*, which computes the $\mathbf{A_d}$matrix and the $\mathbf{B_d}$. We also verified the output of this function by using the equations in eq. (56) to compute the parameters in Matlab which gave the same result.

The Matlab code for computing these matrices are shown in appendix B.8, where it is shown how the two different solutions are implemented. The code shown first declares the continuous state space matrices. Then on line 28 the direct method of finding the discretized model is shown. To do this the Matlab functions *ilaplace*, *inv*, *diag* and *pinv* were used. The function *ilaplace* computes the inverse laplace of the input, the *inv* function computes the inverse of the input, *diag* computes a diagonal matrix with the specified dimension and value.

A simpler method using *c2d* is shown in line 35. Which converts a given model in continous time into a model in discrete time, hence the name.

The resulting discretized matrices are given by

$$\mathbf{A_d} = \begin{bmatrix} 0.9969 & 0.1006 & 0 & 0 & 0 \\ -0.0616 & 1.0115 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0.0999 & -1.0770 \cdot 10^{-5} \\ 0 & 0 & 0 & 0.9986 & -2.1534 \cdot 10^{-4} \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{57a}$$

$$\mathbf{B_d} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1.0770 \cdot 10^{-5} \\ 2.1534 \cdot 10^{-4} \end{bmatrix} \tag{57b}$$

$$\mathbf{C_d} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \end{bmatrix} \tag{57c}$$

$$\tag{57d}$$

The discretized process noise covariance matrix $\mathbf{Q_k}$ was computed using *Loans method* and are given by

$$\mathbf{Q_k} = \begin{bmatrix} 1.6808 \cdot 10^{-7} & 2.5283 \cdot 10^{-6} & 0 & 0 & 0 \\ 2.5283 \cdot 10^{-6} & 5.0567 \cdot 10^{-5} & 0 & 0 & 0 \\ 0 & 0 & 2.3200 \cdot 10^{-18} & 5.7992 \cdot 10^{-17} & -3.5903 \cdot 10^{-13} \\ 0 & 0 & 5.7992 \cdot 10^{-17} & 1.5463 \cdot 10^{-15} & -1.0770 \cdot 10^{-11} \\ 0 & 0 & -3.5903 \cdot 10^{-13} & -1.0770 \cdot 10^{-11} & 1.0000 \cdot 10^{-7} \end{bmatrix} \tag{58}$$

5.2 b) Estimating the variance of the measurement noise

To find an estimate for the variance of the measurement noise, a data series with only measurement noise is collected. It is obtained by setting the reference heading to zero and turning off the disturbances in the model. In the physical world, this would correspond to forcing the rudder to point straight backward, and remain in calm, undisturbed sea. When obtaining the measurement noise data series one should notice that the Simulink block Measurement noise provides the measurements with already discretized noise with pre-defined sample period of 0.5 seconds. Since we in a later problem want to discretize this measurement noise with sample time 0.1 seconds, we changed the sample time in the measurement noise block to 0.1 seconds. Discretizing a signal twice is probably a bad idea, and thus we want to avoid this.

The discretized variance R of the data series, could be obtained by calculating the mean of the signal(which in this case is zero), and summing up the square difference from the mean to the signal value each time step. Matlabs already implemented *var* function does this and yields

$$R = 1.7326 \cdot 10^{-4} rad^2 \tag{59}$$

The simulink diagram used to collect the data series is shown in appendix A.4. We have turned the disturbances in the model off and applied a constant 0 to the input. The Matlab script that extracts the variance from the data series is displayed in appendix B.10.

This is a reasonable approach for extracting the exact variance of the noise. When the rudder angle is set to zero, and there is no disturbances, the expected value for the heading will be constant zero. Moreover, will the only variance of the signal be the variance from the measurement noise.

5.3 c) Implementation of discrete Kalman Filter

The discrete Kalman filter algorithm is an recursive algorithm with the following four sequential steps

1. Compute Kalman gain $\mathbf{L}[\mathbf{k}]$
2. Update estimate $\hat{\mathbf{x}}[\mathbf{k}]$ with measurement
3. Update error covariance matrix $\mathbf{P}[\mathbf{k}]$
4. Project ahead $\hat{\mathbf{x}}^-[\mathbf{k+1}]$, $\mathbf{P}^-[\mathbf{k+1}]$

The four steps are now computed using the following Kalman filter equations

$$\mathbf{L}[k] = \mathbf{P}^-[k]\mathbf{C_d^T}(\mathbf{C_d}\mathbf{P}^-[k]\mathbf{C_d^T} + R)^{-1} \tag{60a}$$
$$\hat{\mathbf{x}}[\mathbf{k}] = \hat{\mathbf{x}}^-[k] + \mathbf{L}[k](\mathbf{y}[k] - \mathbf{C_d}\hat{\mathbf{x}}^-[k]) \tag{60b}$$
$$\mathbf{P}[k] = (\mathbf{I} - \mathbf{L}[k]\mathbf{C_d})\mathbf{P}^-[k](\mathbf{I} - \mathbf{L}[k]\mathbf{C_d})^T + \mathbf{L}[k]R\mathbf{L}[k]^T \tag{60c}$$
$$\hat{\mathbf{x}}^-[k+1] = \mathbf{A_d}\hat{\mathbf{x}}[k] + \mathbf{B_d}\mathbf{u}[k] \tag{60d}$$
$$\mathbf{P}^-[k+1] = \mathbf{A_d}\mathbf{P}[k]\mathbf{A_d}^T + \mathbf{Q_k} \tag{60e}$$

To enter the Kalman filter algorithm we need to define the following initial a priori estimate error covariance matrix $\mathbf{P}_0^-$ and initial a priori state estimate $\mathbf{x}_0^-$

$$\mathbf{P}_0^- = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0.013 & 0 & 0 & 0 \\ 0 & 0 & \pi^2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 2.5 \cdot 10^{-3} \end{bmatrix} \quad \text{and} \quad \mathbf{x}_0^- = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{61}$$

Now as we have all the necessary parameters we are ready to enter the Kalman filter algorithm. The implementation was done using a the Matlab function block in Sumilink. Inside this block we implemented the Matlab function as seen in appendix B.11. The problem with keeping variables between iterations was solved by declaring the variables which we wanted to keep as persistent. The initialization of the Kalman Filter is done using a function *isempty* which returns zero if the input is one. This makes sure that when we set the init flag to one, we never enter this if-sentence once more. The initialization initializes the discretized process noice covariance matrix $\mathbf{Q_k}$, the discretized measurement noise variance R, the initial a priori state estimate $\mathbf{x}_0^-$ and the discretized matrices shown in eq. (57). The rest of the code follows the four sequential steps discussed earlier and outputs the estimated heading angle ψ without any wave disturbance and current noise, as well as the estimated rudder bias b. The resulting Simulink diagram is shown in appendix A.5. Since the model is discretized with sample period T we need to add zero-order hold blocks with sample time T on the input of the Kalman filter. On the output we need to have memory blocks which remembers and outputs the last value that was sent to the block. This is because we need continous signals later when we want to use the output for control purposes.

5.4 d) Feed forward from estimated bias

We now include a feed forward from the estimated bias $\hat{b}$ of the Kalman filter to the rudder input. This clever trick removes the bias introduced by the current from the rudder input as seen in eq. (1d), thus making it possible for the PD controller to understand that there is still an error which we want to eliminate. Note that from now on, the simulation only holds for small deviations in the compass value ψ, that is $\psi \in [-35, 35]^\circ$. This constraint is implemented in Simulink using a saturation block, and is included in all the following simulations.

Simulating the system with a reference heading ψ_r of 30° with only measurement noise and current disturbance gave the result shown in fig. 11.

In the figure we can see that indeed the heading ψ converges to the desired reference heading ψ_r. This stationary deviation has now been eliminated using feed forward from the estimated rudder bias $\hat{b}$ from the Kalman filter. The *dip* at t = 50s is also removed, this is because the rudder at the same time now has a lower absolute value. One can also notice that the rudder angle δ converges with the rudder bias $\hat{b}$. This means that the rudder counteract the current disturbance on the ship.

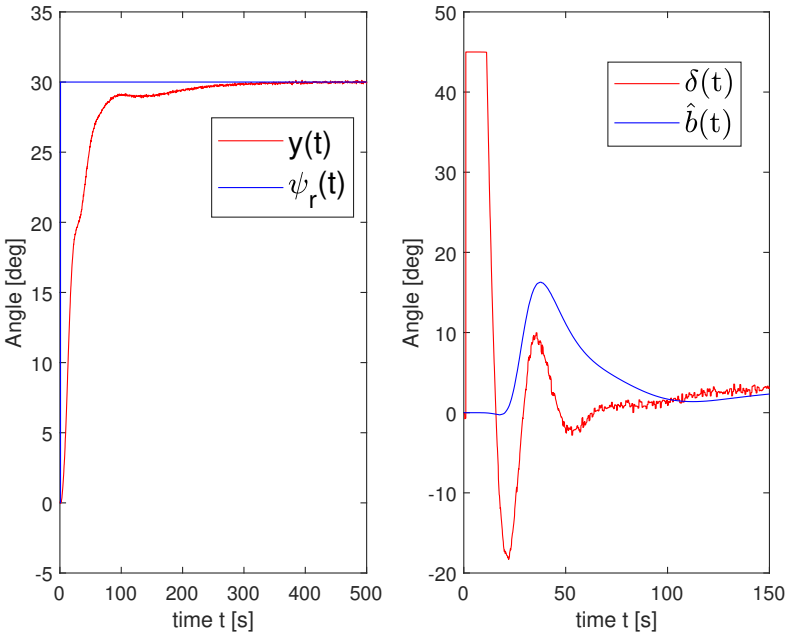


Figure 11: The result of including feed forward from the estimated bias to the rudder input. The compass heading measurement y and the reference heading angle ψ_r are shown in the plot to the left. To the right the rudder input δ is plotted against the estimated rudder bias $\hat{b}$

In problem 3 part c) the resulting autopilot had a stationary deviation, this is because the PD controller had no idea that this rudder bias existed. Since the controller had no idea of the rudder bias, the controller thought that it had reached the desired reference heading ψ_r after some time, when in reality it had not.

The autopilot clearly has a better performance now that we have cancelled the rudder bias using feed forward from the estimated rudder bias $\hat{b}$. Since we have achieved to remove the influence of the bias shown in eq. (1d), the simplification of having no disturbances when designing the PD controller in section 3 now holds. Thus the PD controller can now act as there is no current disturbance at all.

The implementation of the feed forward of the estimated bias $\hat{b}$ to cancel the bias b is shown in appendix A.6.

5.5 e) Feeding the wave filtered heading to the autopilot

Now we use the wave filtered ψ instead of the measured heading y in the autopilot. As previously, the compass measurement has the constraint $\psi \in [-35, 35]^\circ$.

Simulating the system with a reference heading ψ_r of 30° with measurement noise and both wave and current disturbance gave the result shown in fig. 12.

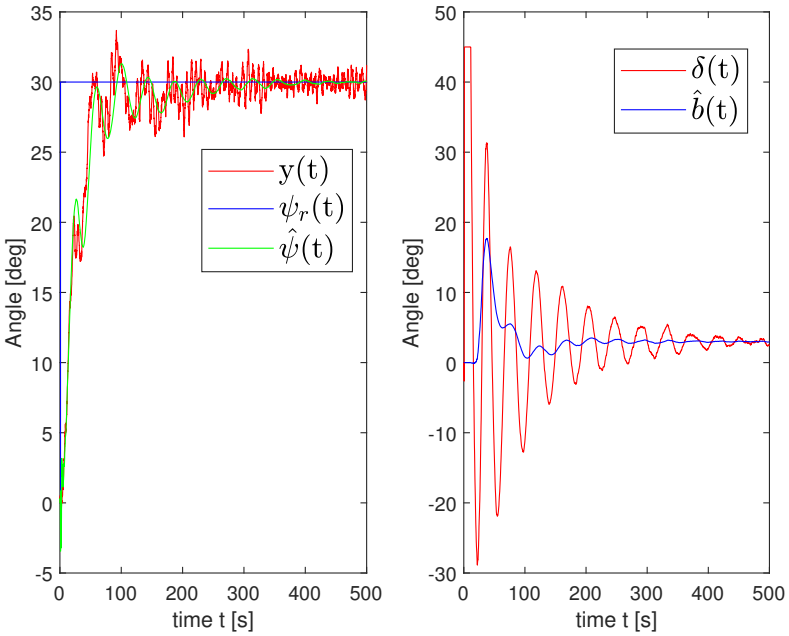
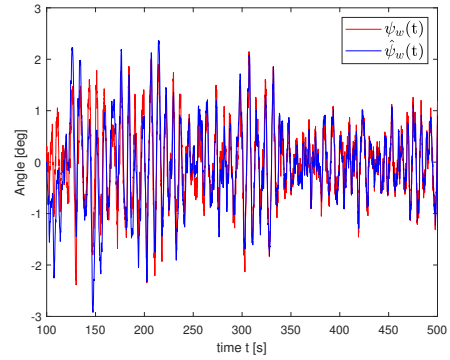


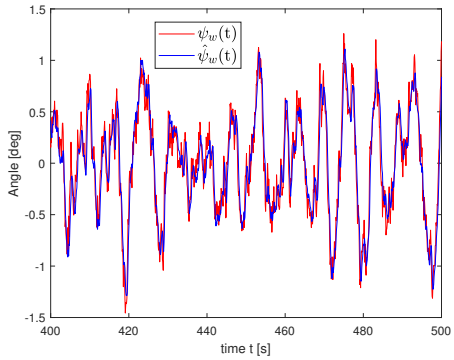
Figure 12: The result using the wave filtered heading $\hat{\psi}$ to the autopilot. The compass heading measurement y, the reference heading angle ψ_r and wave filtered heading $\hat{\psi}$ are shown in the plot to the left. To the right the rudder input δ is plotted against the estimated rudder bias $\hat{b}$

In the figure we can see that the wave filtered heading $\hat{\psi}$ follows the measured heading y but removes the oscillatory noise. The estimate lays perfectly at the mean of the oscillatory measurement y, providing the autopilot a more reasonable value for the actual heading. Looking at the plotted rudder angle δ we can also see that by using the wave filtered ψ we have successfully managed to remove the rather ugly oscillatory rudder angle response shown in fig. 10. The reason why the rudder does not converge to the estimated rudder bias $\hat{b}$, is that the wave filtered heading still has some deviations from the reference heading ψ_r which is perfectly reasonable, due to the wave disturbance.

The wave influence ψ_w and the estimated wave influence $\hat{\psi}_w$ are shown in fig. 13a and a zoomed in version in fig. 13b.



(a)



(b)

Figure 13: (a) The actual wave influence ψ_w in red and the estimated wave influence $\hat{\psi}_w$ in blue when $t \in [100, 500]s$ (b) The actual wave influence ψ_w in red and the estimated wave influence $\hat{\psi}_w$ in blue when $t \in [400, 500]s$

In the figure we can see that we have successfully managed to estimate a smoothed version of the wave influence on the boat. The high frequencies in ψ_w are removed in $\hat{\psi}_w$. This also explains why the estimate is unable to perfectly follow the sudden high variations in ψ_w.

As a conclusion, feeding the wave filtered heading from the Kalman filter yielded better results than previously. Besides having a better estimate of the actual heading, the rudder is now able to safely work while there are waves hitting the boat. Previously the rudder oscillated wildly shown in fig. 10, which is not good for the rudder at all. Now however, the rudder oscillates nice and closes in to the estimated bias to cancel the affect of the currents.

The implementation of feeding the wave filtered heading ψ to the autopilot is shown in appendix A.7.

5.6 f) The Q matrix

5.6.1 Q matrix discussion

The process disturbance covariance matrix **Q** is as the name says, a matrix consisting of all the disturbance covariances in the process. On the diagonal

one can identify the variances of the disturbances given in $\mathbf{w}$. The off-diagonal elements corresponds to the covariances of the disturbances given in $\mathbf{w}$. In our case the disturbances $\mathbf{w}$ is given as

$$\mathbf{w} = \begin{bmatrix} w_w \\ w_b \end{bmatrix} \tag{62}$$

Thus, the $\mathbf{Q}$ matrix should take the following form

$$\mathbf{Q} = \begin{bmatrix} \sigma_w^2 & \sigma_{wb}^2 \\ \sigma_{bw}^2 & \sigma_b^2 \end{bmatrix} \tag{63}$$

Now, if we change any of the values in the $\mathbf{Q}$ matrix we are directly changing the variances in the model. For instance, if we multiply $\mathbf{Q}_{1,1}$, that is σ_w^2 by a factor of 2, we make the system believe that the wave disturbance has a variance of $2\sigma_w^2$. Intuitively, the wave disturbance now has more variations than previously, and are therefore more uncertain than before. To analyze the contribution of the $\mathbf{Q}$ matrix on the Kalman filter, lets have a look at the equation at which the $\mathbf{Q}$ enters the Kalman filter algorithm, *eq.* (64).

$$\mathbf{P}^-[k+1] = \mathbf{A_d}\mathbf{P}[k]\mathbf{A_d}^T + \mathbf{Q_k} \tag{64}$$

Here we can see that the discretized version $\mathbf{Q_k}$ of $\mathbf{Q}$ are added to the predicted error covariance matrix $\mathbf{P}^-[\mathbf{k+1}]$. This intuitively means that the $\mathbf{Q_k}$ matrix adds uncertainty to the predicted error covariance matrix. If we, as discussed earlier, double the first element in the process noise covariance matrix, that means that the we add uncertainty to the predicted error covariance matrix.

Lets say that the elements of the $\mathbf{Q_k}$ matrix has unreasonable high values on the diagonal, thus meaning that the disturbances in the system has high variances. This means that we add big uncertainties to the prediction error. Now if we look at the estimated state $\hat{\mathbf{x}}[\mathbf{k}]$ in eq. (60b), we can see that the prediction is depending on the a priori estimate $\hat{\mathbf{x}}$ which is a prediction from the model, excluding the disturbances. Now whats very interesting about the computation of the state estimate is the term where the Kalman gain $\mathbf{L}$ enters the equation. Intuitively the Kalman gain adjusts how much we should trust the measurement to correct the estimated states, or neglect the correction of the measurement. The elements in the Kalman gain $\mathbf{L}$ can take values from 0 to 1. Lets explain why, by simplyfying things by looking at a monovariable system.

If a given system is monovariable then eq. (60a) reduces to

$$K = \frac{E_{est}}{E_{est} + E_{meas}} \tag{65}$$

Where E_{est} are the error in the estimate and E_{mes} are the error in the measurement. Now, if the error in the estimate is high, then the Kalman gain K will be equal to 1. If the error in the measurement is high, then the Kalman gain K will be equal to 0. This reasoning also applies to multivariable systems, thus in further discussion, we continue to talk about the error in the estimate and the error in the measurement.

Lets have a look at our multivariable system. If the elements on the diagonal of $\mathbf{Q_k}$ is chosen unreasonable high, as discussed earlier, the error in the estimates of all the states should all be high. When the Kalman gain L is computed, it

recognizes that there is a high error in the error covariance matrix $\mathbf{P}$, which corresponds to the E_{est} in the monovariable case. Thus the Kalman gain $\mathbf{L}$ is then chosen to be a diagonal matrix with values of 1, since the error in the measurement are assumed to be minimal compared to the error in the estimate. The estimated state $\hat{\mathbf{x}}$ is now adjusted using the computed Kalman gain, and we can see that the measurement y is fully trusted to update the estimate $\hat{\mathbf{x}}$.

Now if we reduce the values of the elements in $\mathbf{Q_k}$ to be very small, the opposite happens. Using the same reasoning the Kalman filter will now choose a Kalman gain with zeros on the diagonal. The state estimate $\hat{\mathbf{x}}$ is now computed using solely the model, not the measurement, since the measurement is very uncertain compared to the very small uncertainty of process noises.

Now that we have discussed the impact of changing the $\mathbf{Q_k}$ matrix lets have a look at the given $\mathbf{Q}$ matrix shown once more in eq. (67) used for computing the discretized $\mathbf{Q_k}$

$$\mathbf{Q} = \begin{bmatrix} 30 & 0 \\ 0 & 10^{-6} \end{bmatrix} \tag{66}$$

Comparing $\mathbf{Q}$ in eq. (67) and $\mathbf{Q}$ in eq. (63) we can see that for our system the wave disturbance is set to have a variance σ_w^2 equal to 30. The current disturbance is set to have a variance σ_b^2 equal to 10^{-6}. The off diagonal elements are set to 0, which means that the two disturbances are uncorrelated.

Now, the wave disturbance w_w is said in the assignment to be a zero mean white noise process with unity variance. This statement can be found under *4.2.2 Waves* in the assignment. Then obviously, the first element of the diagonal of $\mathbf{Q}$ should be set equal to 1, not 30.

5.6.2 5.5.d discussion

From now on the first element of the diagonal of $\mathbf{Q}$, that is σ_w^2, is set to 1. This gives the following $\mathbf{Q}$ matrix.

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 \\ 0 & \sigma_b^2 \end{bmatrix} \tag{67}$$

Lets do the same simulation as we did in 5.5.d. Since the interesting part about this part is the estimated rudder bias and the resulting measured heading we will try to both reduce and high-en the variance of the current σ_b^2 to respectively $\sigma_{b1}^2 = 10^{-10}$ and $\sigma_{b2}^2 = 1$. Intuitively this means that the Kalman filter should really trust the model when using σ_{b1}^2 for estimation. On the other hand, when using σ_{b2}^2 the Kalman filter should trust the measurement more than previously. Lets look at the result for these two values of σ_b^2. In fig. 14 σ_{b1}^2 was used and in fig. 15 σ_{b2}^2 was used.

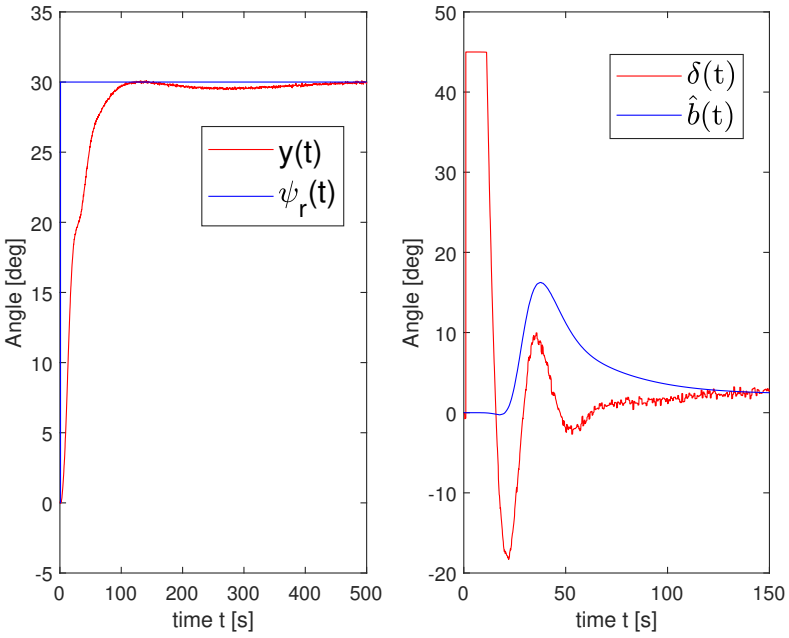


Figure 14: The result of including feed forward from the estimated bias to the rudder input. The compass heading measurement y and the reference heading angle ψ_r are shown in the plot to the left. To the right the rudder input δ is plotted against the estimated rudder bias $\hat{b}$. In this plot σ_b^2 is set equal to 10^{-10}.

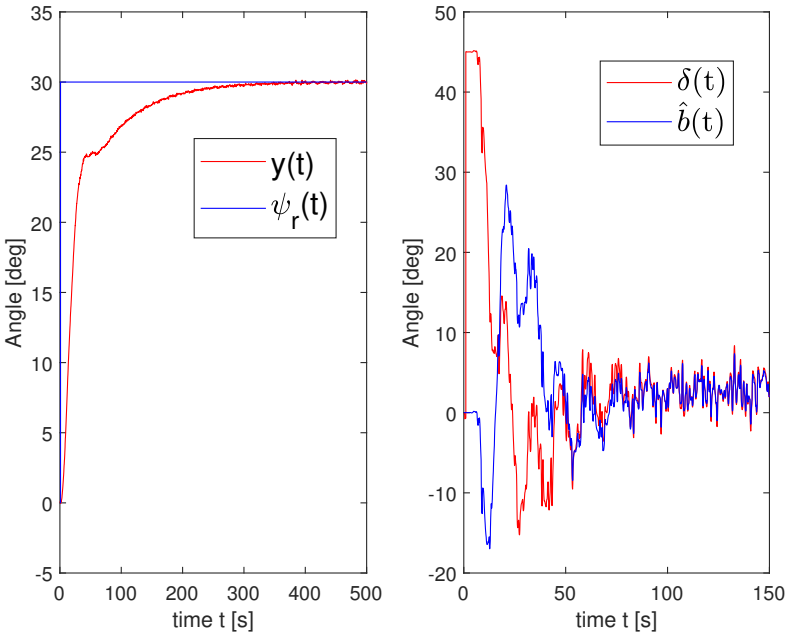


Figure 15: The result of including feed forward from the estimated bias to the rudder input. The compass heading measurement y and the reference heading angle ψ_r are shown in the plot to the left. To the right the rudder input δ is plotted against the estimated rudder bias $\hat{b}$. In this plot σ_b^2 is set equal to 1.

In fig. 14 we can see that the response looks almost identical to the one in fig. 11, this is very reasonable since the variance was already very small in the previous simulation. However, we can see a small improvement where the rudder actually manages to converge better to the estimated bias with $\sigma_b^2 = 10^{-10}$. This means that the Kalman filter trusts the model even more than before.

Now, setting $\sigma_b^2 = 1$ yields the result shown in fig. 15. Now the Kalman filter trusts the model way less than previously, putting more trust in to the measurement to give a good estimate on the rudder bias. Looking at eq. (1f) this is probably a bad idea. The measurement contains no direct information on the current influence on the heading. Using the model would therefore probably be more wise when estimating the rudder bias b. Looking at the chaotic and oscillatory estimation of the rudder bias $\hat{b}$ in fig. 15 makes the rudder oscillate to remove the effect of the bias. This is not a very desirable effect, thus trusting the measurement more on estimating the bias is therefore a bad choice.

In further discussion and simulations we use $\sigma_b^2 = 10^{-6}$, this proved to be a decent choice of the current variance.

5.6.3 5.5.e discussion

Setting σ_w^2 equal to 1 and not 30, intuitively says that the Kalman filter now trusts the model way more than previously. Thus the Kalman gain will now

not trust the measurement as much when estimating $\hat{\mathbf{x}}$. This result is shown in fig. 16. In the figure we can see that the estimated heading $\hat{\psi}$ now is way less oscillatory than before shown in fig. 12. This is because the Kalman filter now believes the model for heading shown in eq. (1c) and eq. (1d) more than before. This is reasonable because in our case, the model is very good with the limitations we have set for the system. As we can see in the plot, the measurement is very noisy and is therefore oscillating quite a bit. Since we trust the measurement less now, the estimated heading should now oscillate less, which is what we see in the figure.

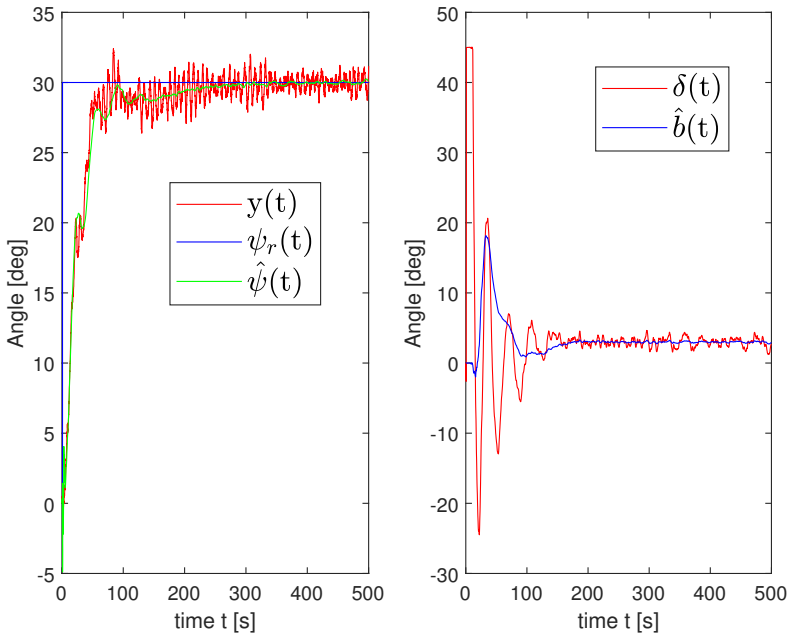
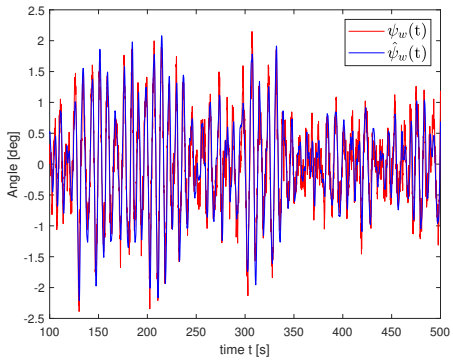


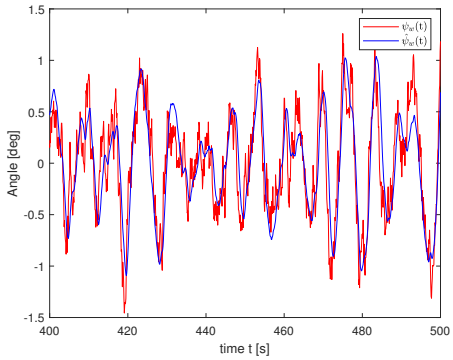
Figure 16: The result using the wave filtered heading $\hat{\psi}$ to the autopilot. The compass heading measurement y, the reference heading angle ψ_r and wave filtered heading $\hat{\psi}$ are shown in the plot to the left. To the right the rudder input δ is plotted against the estimated rudder bias $\hat{b}$. The wave disturbance is now set to have a unity variance, i.e $\sigma_w^2 = 1$

Now lets look at the rudder angle δ and estimated rudder bias $\hat{b}$ in fig. 16. The rudder angle now seems to have way less oscillations with big amplitude. Already at $t = 100s$ we have the same oscillations we had with the previous $\mathbf{Q}$ matrix eq. (53) at $t = 300$. The resulting system is therefore faster in the case where $\sigma_w^2 = 1$.

The wave influence ψ_w and the estimated wave influence $\hat{\psi}_w$ are shown in fig. 17a and a zoomed in version in fig. 17b. Looking at the plot and comparing it with the previous plot shown in fig. 13b we can see that now, the wave influence $\hat{\psi}_w$ follows the actual wave influence more poorly than previously. The new estimate seems to be much smoother however.



(a)



(b)

Figure 17: (a) The actual wave influence ψ_w in red and the estimated wave influence $\hat{\psi}_w$ in blue when $t \in [100, 500]s$ (b) The actual wave influence ψ_w in red and the estimated wave influence $\hat{\psi}_w$ in blue when $t \in [400, 500]s$. The wave disturbance is now set to have a unity variance, i.e $\sigma_w^2 = 1$.

5.6.4 Conclusion

The process disturbance covariance matrix $\mathbf{Q}$ contains the variances of the disturbances on the diagonal and the covariances on the off diagonal. In our case the disturbances are uncorrelated, meaning that the off diagonal elements are set to zero. As we have demonstrated using simulations, changing the $\mathbf{Q}$ matrix gave some interesting and different results. In the case of reducing the wave disturbance variance, we achieved a better result in the wave filtered heading from the Kalman filter. The filtered heading had way less oscillations, which we discussed was because the model was trusted more since the variance of the waves in the model was reduced by a factor of 30. Now, the variance of the current disturbance was already set very low. Thus reducing it further did not give any considerable changes in the response. Increasing σ_b^2 gave worse results. This was discussed to be a consequence of trusting the measurement more, which was not a clever idea.

Since changing the $\mathbf{Q}$ gave different results, and there was an intuitive way

of telling how the system reacted, it is safe so say that the **Q** matrix seems to be some kind of tuning matrix for the Kalman filter. Lets compare the Kalman filter to the Luenberger observer we used in the Helicopter lab. The tuning of the Luenberger observer was a tedious task. The poles had to be manually placed wherever we thought was a good idea to place them. For the Kalman filter however, we do not need to think about placing poles. The poles are automatically placed by the Kalman filter by adjusting the **Q** matrix! This means that we can tune the Kalman filter with only the process disturbance variance matrix, given a perfect measurement noise variance R. The Kalman filter therefore seems to be a better choice if we have information about all the disturbances or noises of the system, or are able to estimate them. However, we are not able to get rid of the tuning aspect.

Appendix

A Simulink Diagrams

A.1 5.1.b System with sine rudder input

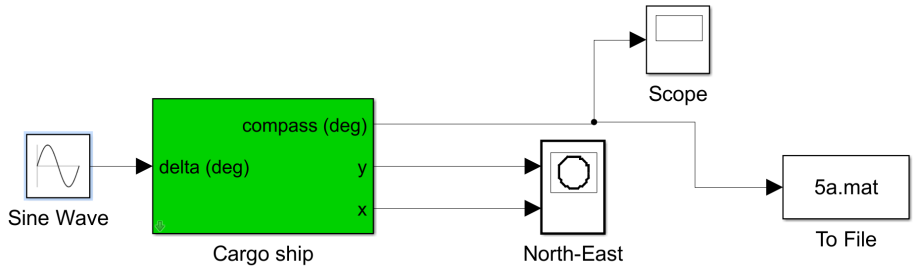


Figure 18: Simulink diagram where a sine wave is applied to the rudder input using the Sine Wave block.

A.2 5.3b System with PD-controller

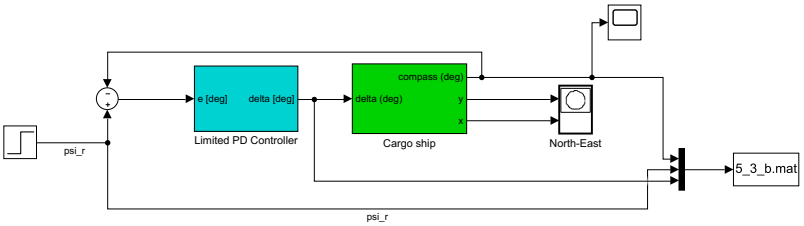


Figure 19: The complete system with our PD-controller. The saturation block in the feedback is avoid using the model for ψ outside $[-35°, 35°]$.

A.3 5.3b The PD -controller

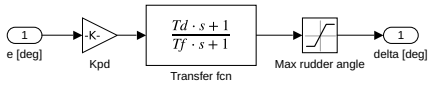


Figure 20: The limited PD-controller used for controlling the ships heading. It is implemented as the transferfunction.

A.4 5.5b Collecting measurement noise data

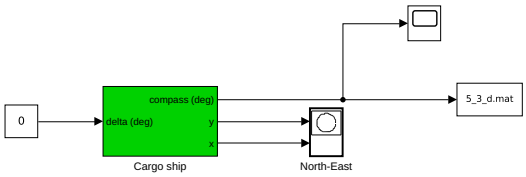


Figure 21: Simulink diagram showing how we extracted the measurement noise data from the compass measurement. Current and wave disturbances is turned of. Measurement noise turned on.

A.5 5.5c System with PD- controller and Kalman filter

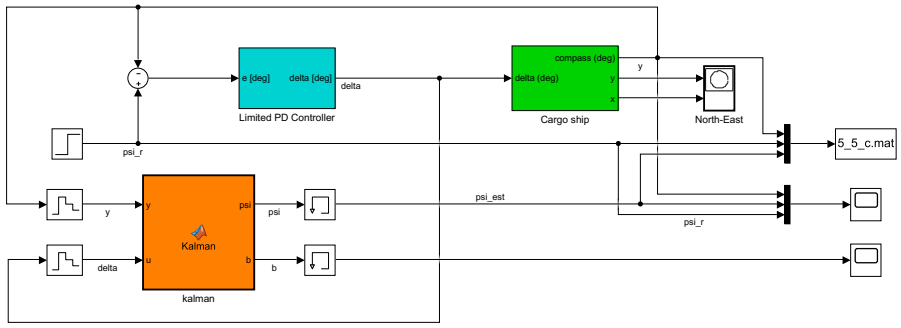


Figure 22: The Simulink diagram of the entire system with the implemented PD- controller and Kalman filter

A.6 5.5d System with feed forward from estimated bias

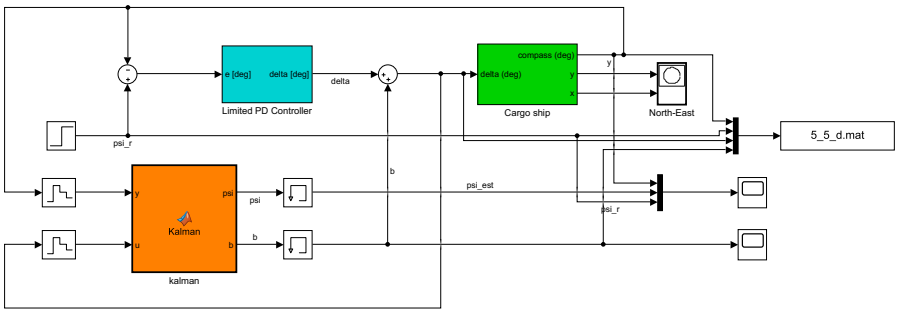


Figure 23: The Simulink diagram of the system with feed forward from the estimated bias to cancel the bias b

A.7 5.5e System when feeding the wave filtered heading to the autopilot

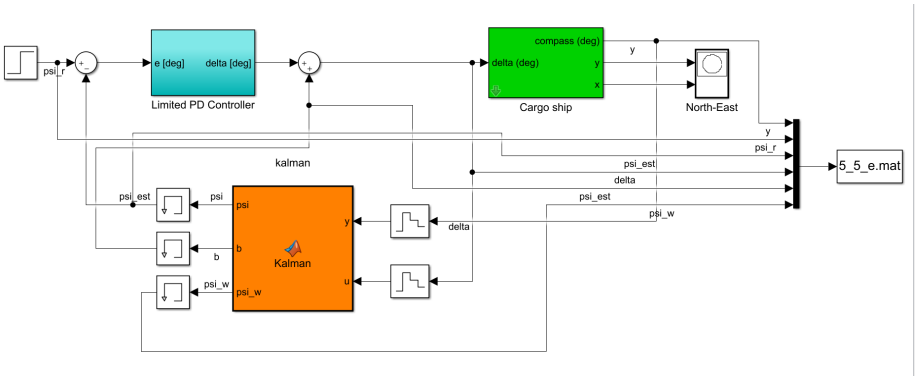


Figure 24: The Simulink diagram where the Kalman filter provides the autopilot with the wave filtered heading ψ

B Matlab code

B.1 Amplitude

```
%Opening the data from file
filename1 = '5b_05.mat';
m1 = matfile(filename1);
array = m1.ans;

%The oscillatory response
y_oscill = array(2,[2000:4000]);
y_max = max(y_oscill);
y_min = min(y_oscill);
y_mean = (y_max + y_min)/2;
y_amplitude = y_max-y_mean;

%Plotting the response and the mean-line
y = array(2,:);
t = array(1,:);
plot(t,y);
hold on;
line([0 6000],[y_mean y_mean],'LineStyle','--','color'
    ,'red');
legend('H(j0.05)','Mean');
xlabel('time t [s]');
ylabel('H(j0.05)');
```

B.2 PSD waves

```
%Opening the wave data from file
filename1 = '../../wave.mat';
m1 = matfile(filename1);
filedata = m1.psi_w;

%Extracting the data and converting [deg] to [rad]
psi_w = filedata(2,:).*(pi/180);

%Calculating the PSD, pxx[power/Hz] with frequencies f
    [Hz]
fs = 10;
[pxx, f] = pwelch(psi_w,4096,[],[],fs);

%Converting frequency f[Hz] to omega[rad/s]
omega = f.*2*pi;

%Converting PSD[power/Hz] to PSD[power s/rad]
p_xx = pxx./(2*pi);

%plotting the PSD
plot(omega,p_xx);
xlim([0 2*pi]);
xlabel('Angular frequency \omega [rad/s]');
ylabel('Power spectral density S_{\psi_\omega}(\omega)
```

```
    [power s/rad]');
legend('S_{\psi_\omega}(\omega)');
```

B.3 Model versus boat

```
%Opening the data from file
filename1 = '5d_step_boat.mat';
m1 = matfile(filename1);
array = m1.ans;

%Plotting the boat response
y = array(2,:);
t = array(1,:);
plot(t,y,'r');
hold on;

%Implementing the model
T = 72.4391; K = 0.1561;
s = tf('s');
H = K/(s*(T*s+1));
stepresponse_model = step(H)/(2*pi);

%Plotting the model response
plot(stepresponse_model,'b');
legend('The boat response','The model response');
xlabel('time t [s]');
ylabel('Heading \psi [deg]');
xlim([0 5000]);
```

B.4 Identifying ω_0 and σ^2

```
%Opening the wave data from file
filename1 = '../../wave.mat';
m1 = matfile(filename1);
filedata = m1.psi_w;

%Extracting the data and converting [deg] to [rad]
psi_w = filedata(2,:).*(pi/180);

%Calculating the PSD, pxx[power/Hz] with frequencies f
    [Hz]
fs = 10;
[pxx, f] = pwelch(psi_w,4096,[],[],fs);

%Converting frequency f[Hz] to omega[rad/s]
omega = f.*2*pi;

%Converting PSD[power/Hz] to PSD[power s/rad]
p_xx = pxx./(2*pi);
```

```
18
19 %plotting the PSD
20 plot(omega,p_xx);
21 xlim([0 2*pi]);
22 xlabel('Angular frequency \omega [rad/s]');
23 ylabel('Power spectral density S_{\psi_\omega}(\omega)
       [power s/rad]');
24 legend('S_{\psi_\omega}(\omega)');
25
26 %Finding the modal peak frequency omega_0 and
      intensity sigma
27 [peak, idx] = max(p_xx);
28 omega_0 = omega(idx);
29 sigma = sqrt(peak);
```

B.5 Identifying the dampening factor λ

```
1  %Opening the wave data from file
2  filename1 = '../../wave.mat';
3  m1 = matfile(filename1);
4  filedata = m1.psi_w;
5
6  %Extracting the data and converting [deg] to [rad]
7  psi_w = filedata(2,:).*(pi/180);
8
9  %Calculating the PSD, pxx[power/Hz] with frequencies f
      [Hz]
10 fs = 10;
11 [pxx, f] = pwelch(psi_w,4096,[],[],fs);
12
13 %Converting frequency f[Hz] to omega[rad/s]
14 omega = f.*2*pi;
15
16 %Converting PSD[power/Hz] to PSD[power s/rad]
17 p_xx = pxx./(2*pi);
18
19 %Finding the modal peak frequency omega_0 and
      intensity sigma
20 [peak, idx] = max(p_xx);
21 omega_0 = omega(idx)
22 sigma = sqrt(peak)
23
24 %Least square method to estimate lambda
25 fun = @(lambda,omega)(omega.^2 * (2*lambda*omega_0*
      sigma).^2)./((omega_0.^2-omega.^2).^2+4*omega*
      lambda.^2*omega_0.^2);
26 lambda = lsqcurvefit(fun, 0.1, omega, p_xx);
27
28 %plotting the estimated PSD
```

```
plot(omega,p_xx);
hold on;

%plotting the PSD using LSQ method
K_w = 2*lambda*omega_0*sigma;
P_ww = (omega.^2 * (2*lambda*omega_0*sigma).^2)./((
    omega_0.^2-omega.^2).^2+4*omega*lambda.^2*omega_0
    .^2);
plot(omega, P_ww);
xlim([0 2*pi]);
grid on;
xlabel('frequency \omega [rad/s]');
ylabel('Power spectral density [power s/rad]');
legend('S_{\psi_w}(\omega)','P_{\psi_w}(\omega)');
hold on;
```

B.6 Observability

```
%Constants
omega_0 = 0.7823;
T = 72.4391;
K = 0.1561;
lambda = 0.0928;

%No disturbances
A = [[0      1    ]
     [0      -1/T]];
C = [1 0];

O_1 = obsv(A,C);
rank_O_1 = rank(O_1);

%With current disturbance
A = [[0      1        0]
    [ 0    -1/T    -K/T]
    [ 0      0       0]];

C = [1 0 0];

O_2 = obsv(A,C);
rank_O_2 = rank(O_2);

%With wave disturbance
A = [[0          1                    0        0        ]
    [-omega_0^2 2*lambda*omega_0      0        0        ]
    [0           0                    0        1        ]
    [0           0                    0        -1/T
       ]];

```

```
C = [0 1 1 0];

rank(obsv(A,C))

%With wave disturbance and current
A = [[0          1                    0        0        0
          ]
     [-omega_0^2 2*lambda*omega_0     0        0        0
                ]
     [0             0                 0        1        0
                ]
     [0             0                 0        -1/T     -K
        /T     ]
     [0             0                 0        0        0
                ]];

C = [0 1 1 0 0];

O_4 = obsv(A,C);
rank_O_4 = rank(O_4);
```

B.7 Bode plots of H_0

```
%Implementation of the open loop transfer function
s = tf('s');
K = 0.1561; T = 72.4391; omega_c = 0.1;

%PD parameters
Kpd = 0.3485; Td = T; Tf = 8.391;

%Transfer function H_0
H_0 = (K*Kpd)/(s*(1+Tf*s));

%Plotting the bode plots of H_0
margin(H_0);
```

B.8 Discretization

```
%Constants
omega_0 = 0.7823; sigma = 0.0281;
T = 72.4391; K = 0.1561;
lambda = 0.0928; K_w = 2*lambda*omega_0*sigma;

%Continous model matrices
A_c = [[0          1                    0        0
    0        ]
       [-omega_0^2 2*lambda*omega_0     0        0
          0        ]
```

```
9         [0            0                       0        1
             0        ]
10        [0            0                       0        -1/T
             -K/T     ]
11        [0            0                       0        0
             0        ]];
12
13  B_c = [0 0 0 K/T 0]';
14
15  C_c = [0 1 1 0 0];
16
17  E_c = [[0 K_w 0 0 0]
18         [ 0 0   0 0 1]]';
19
20  Q_c = [[30       0       ]
21         [0       10^-6    ]];
22
23  %Discretization
24  fs = 10; Ts = 1/fs;
25  syms s;
26  v = [s s s s s];
27
28  % [1] Direct method of finding discretized matrices
29  A_d = ilaplace(inv((diag(v)-A_c)), s, Ts);
30  % Benytter pseudoinvers for   invertere ikke-
       inverterbar A-matrise
31  B_d = pinv(A_c)*(A_d-eye(5))*B_c;
32  C_d = C_c;
33  R_d = 1.7326*10^-4;
34
35  % [2] Using c2d to find discretized A and E matrix
36  [A_d2 B_d2] = c2d(A_c,B_c,Ts);
```

B.9 Loans method

```
1   %       Loans method to aquire Q_k, chen p.126
2   %
       ------------------------------------------------------------

3   %       x_dot = F*x + G*u    same as     x_dot = A_c*x
       + E_c*w
4   %       z = H*x + v          same as     y = C_c*x + v
5
6   % [1] Forming A
7   G = E_c; W = Q_c; F = A_c;
8   A = [-F G*W*G'; zeros(5) F'];
9   % [2] Forming B
10  B = expm(A*Ts);
11  % [3] Transpose of lower-right of B to give Phi
```

```
Phi = B(6:10,6:10)';
% [4] Q_k is now found as
Q_k = Phi * B(1:5,6:10);
```

B.10 Calculating measurement variance

```
filename = '5_3_d.mat';
m = matfile(filename);
filedata = m.ans;

%Extracting data and calculating the variance
%of the measurement noise
noise = filedata(2,:);
var_b = var(noise);
%Converting [deg] to [rad]
var_b = var_b*pi/180;
```

B.11 Kalman filter implementation

```
% Input: m ling y, p drag u
% Output: kalman filter: Estimert psi uten st y og
    rudder bias b
function [psi, b] = Kalman(y, u)

persistent  init_flag A B C E X_ R Q_k P_

% Initialize
if isempty(init_flag)

    init_flag = 1;

    Q_k = [1.68080996498622e-07,2.52832988209693e
        -06,0,0,0;
         2.52832988209693e-06,5.05671543189852e
            -05,0,0,0;
         0,0,2.32004646392527e-22,5.79922670073290e
            -21,-3.59028347101574e-17;
         0,0,5.79922670073289e-21,1.54628259912526e
            -19,-1.07696115999909e-15;
         0,0,-3.59028347101574e-17,-1.07696115999909e
            -15,1.00000000000000e-11];

    R = 0.00017326;

    P_0 =  [[1      0      0      0      0        ]
            [0      0.013  0      0      0        ]
            [0      0      pi^2   0      0        ]
            [0      0      0      1      0        ]
            [0      0      0      0      2.5*10^-3]];
```

```
25
26      X_0 = [0 0 0 0 0]';
27
28      A = [[0.996926739262672     0.100626789614164
                0       0                       0
                                                ]
29          [-0.061582920038110     1.011537233905485
                    0       0                       0
                                                    ]
30          [0                          0
                                          1
               0.099931008253684
               -1.076961159999087*10^-5  ]
31          [0                          0
                                          0
               0.998620482470742
               -2.153426863171962*10^-4  ]
32          [0                          0
                                          0       0
                                      1
                                            ]];
33
34      B = [0 0 1.076961159999087*10^-5
           2.153426863171962*10^-4 0]';
35
36      C = [0 1 1 0 0];
37
38      E = [[2.048850967535938*10^-5
           4.105548994630666*10^-4      0
                                      0
                                        0
           ]
39          [ 0                           0

               -3.590283471015735*10^-7
               -1.076961159999087*10^-5
               0.100000000000000   ]]';
40
41      X_ = X_0;
42      P_ = P_0;
43  end
44  %Kode som kjres etter init av KF
45
46  %Omgjring av input [deg] til [rad]
47  y = y.*(pi/180);
48  u = u.*(pi/180);
49
50  % [1] Computing the kalman gain L
51  L = P_*C.'*inv(C*P_*C.' + R);
52
```

```
53 % [2] Updating the estimate with the measurement y
54 X_hat = X_ + L * (y - C*X_);
55
56 % [3] Updating the error covariance matrix P
57 % This formula holds for both optimal and suboptimal
      gain L
58 P = (eye(5)- L*C) * P_ *(eye(5)-L*C).' + L*R*L.';
59
60 % [4] Projecting ahead
61 X_ = A*X_hat + B*u;
62 P_ = A*P*A.' + Q_k;
63
64 % Oppdaterer output heading psi og rudder bias b
65 % Konverterer ogs fra [rad] til [deg]
66 psi = X_hat(3).*(180/pi);
67 b = X_hat(5).*(180/pi);
68 end
```

References

[1] *Boat_ lab_ assignment.pdf*. https://ntnu.blackboard.com/bbcswebdav/pid-420179-dt-content-rid-18213414_1/courses/194_TTK4115_1_2018_H_1/boat_assignment%282%29.pdf. Accessed: 2018-11-22.

[2] Chi-Tsong Chen. *Linear System Theory and Design*. Oxford University Press, Incorporated, 2014.

[3] Manolakis Dimitris G. Proakis John G. *Digital Signal Processing: Pearson New International Edition*. Pearson Education Limited, 2013.