Linear System Theory
Boatlab

Group 29
Theodor Tveit Husefest 478428
Marius Bæver Larsen 478447
Johan Brun 768630

October 21, 2018

Department of Engineering Cybernetics

Contents

Introduction

In this lab exercise we will simulate a moving ship using a simulink model provided to us. We will observe how the ship behaves when exposed to different types of noise, with and without control applied, and attempt to make an autopilot for the ship. In the end we will apply a discrete Kalman filter in order to neutralize disturbances and noise on the desired heading.

The model for the system is given as:

$$\dot{\xi_w} = \psi_w \tag{0.1a}$$
$$\dot{\psi_w} = -\omega_0^2\xi_w - 2\lambda\omega_0\psi_w + K_w w_w \tag{0.1b}$$
$$\dot{\psi} = r \tag{0.1c}$$
$$\dot{r} = -\frac{1}{T}r + \frac{K}{T}(\delta - b) \tag{0.1d}$$
$$\dot{b} = w_b \tag{0.1e}$$
$$y = \psi + \psi_w + v \tag{0.1f}$$

We can write the system in state-space form.

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}u + \mathbf{E}\mathbf{w},\ y = \mathbf{C}\mathbf{x} + v \tag{0.2}$$

Where $\mathbf{x}$, u and $\mathbf{w}$ are defined as:

$$\mathbf{x} = \begin{bmatrix} \xi_w \\ \psi_w \\ \psi \\ r \\ b \end{bmatrix},\ u = \delta \text{ and } \mathbf{w} = \begin{bmatrix} w_w \\ w_b \end{bmatrix} \tag{0.3}$$

1 Identification of the boat parameters

1.a Transfer Function from δ to ψ

From (0.1c) we get that

$$\dot{\psi} = r \implies \ddot{\psi} = \dot{r} \tag{1.1}$$

And by applying (1.1) to (0.1d) and letting $b = 0$, due to no disturbance, we get

$$\begin{aligned} \ddot{\psi} &= -\frac{1}{T}\dot{\psi} + \frac{K}{T}\delta \\ \overset{\mathcal{L}}{\Rightarrow} s^2\psi &= -s\frac{1}{T}\psi + \frac{K}{T}\delta \\ \delta &= \psi\frac{1}{K}(Ts^2 + s) \\ H(s) = \frac{\psi}{\delta} &= \frac{K}{s(Ts+1)} \end{aligned} \tag{1.2}$$

1.b Identifying Boat Parameters Without Noise

Substituting for $s = j\omega$

$$\begin{aligned} H(j\omega) &= \frac{K}{-T\omega^2 + j\omega} \\ &= K\frac{-T\omega^2 - j\omega}{(-T\omega^2)^2 + \omega^2} \\ \Rightarrow |H(j\omega)| &= \frac{K}{\omega\sqrt{T^2\omega^2 + 1}} \end{aligned} \tag{1.3}$$

The amplitudes A_1 and A_2 of the responses shown in Figure 1, corresponding to $\omega_1 = 0.005$ and $\omega_2 = 0.05$, where calculated in MATLAB using the `min` and `max` functions for timeseries. Then an equation set for T and K where found and calculated in MATLAB by inserting said values, see appendix A.a.

$$A_1 = 29.3582 \tag{1.4}$$

$$A_2 = 0.8310 \tag{1.5}$$

$$K = A_1\omega_1\sqrt{T^2\omega_1^2 + 1} = 0.1561\text{s}^{-1} \tag{1.6}$$

$$T = \frac{\sqrt{K^2 - \omega_2^2 A_2^2}}{\omega_2^2 A_2} = 72.4346\text{s} \tag{1.7}$$

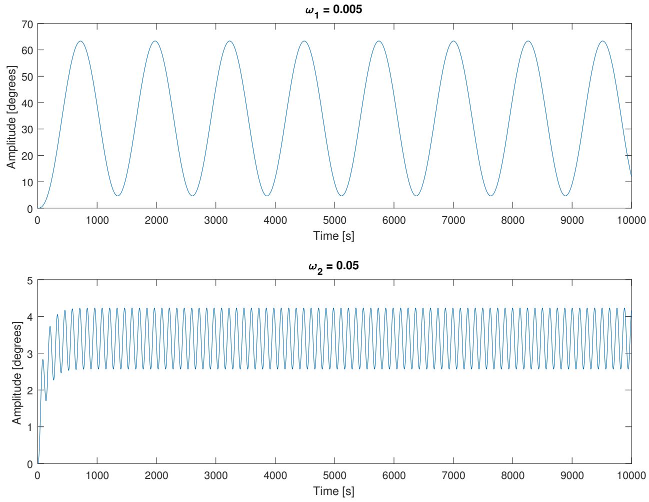

Figure 1: Plot of amplitude when using a sine wave as input. All noise and disturbances turned off.

1.c Identifying Boat Parameters With Noise

In this problem we turned on the measurement noise as well as the wave disturbance. We repeated the approach as in subsection 1.b and calculated the new values for K and T. In Figure 2 we can see that while the response to the sine-wave with $\omega_1 = 0.005$ is usable, albeit a little fuzzy, the response with $\omega_2 = 0.05$ is far too affected by the noise to be usable. Thus it is not possible to get a good estimate for the boat parameters.

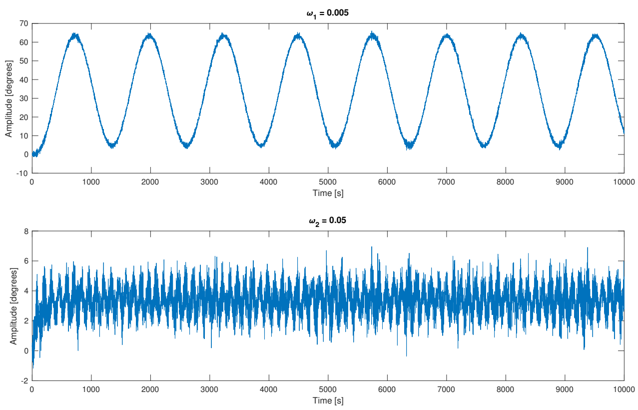

Figure 2: Plot of amplitude when using a sine wave as input, and with wave and measurement noise turned on.

1.d Evaluation Of Model

To examine if the model is a good approximation of the ships dynamics we will apply a step input of one degree to the model and the rudder of the ship. The step-response of both is shown in Figure 3, and we can see that after a long time the model deviates from the ship. Although it might be possible to find a better approximation, on a shorter time-horizon it is a good enough approximation for this exercise.

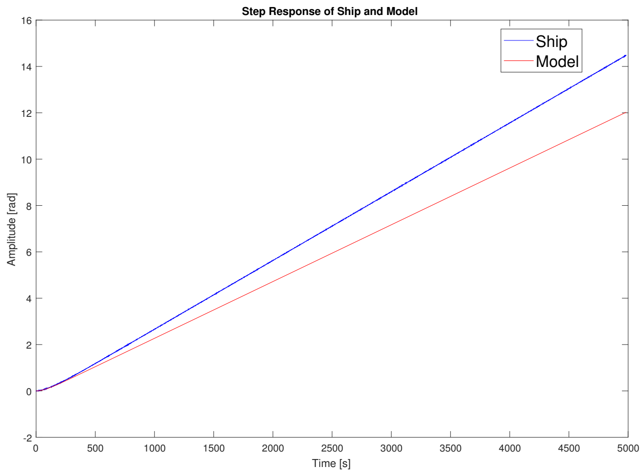

Figure 3: Plot of step-response using both the simulated ship and transfer-function

2 Identification of wave spectrum model

2.a Estimate of Power Spectral Density (PSD)

To better understand how waves affect the compass measurement we will find an estimate for the Power Spectral Density function of ψ_ω, $S_{\psi_\omega}(\omega)$. The PSD describes how the power of a the signal is distributed over each frequency, meaning that in this case it will describe how much the waves affect the measurements at different frequencies.

We found $S_{\psi_\omega}(\omega)$ by using the MATLAB function `[pxx, f] = pwelch(psi_w, window, [], [], fs)` with sampling frequency, fs, equal to 10 Hz and window-size of 4096px. The pwelch function returns the two-sided Welch PSD estimates. As the sampling frequency was given in hertz we had to scale the outputs pxx and f by $\frac{1}{2\pi}$ and 2π respectively. The complete MATLAB code is found in appendix A.b. The estimated function is shown in Figure 4

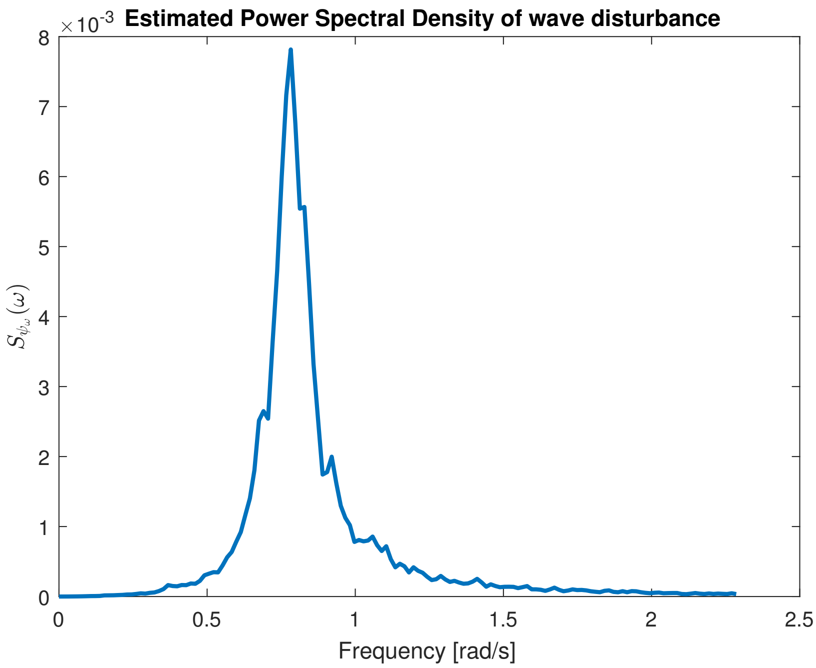

Figure 4: Power Spectral Density estimate of wave-disturbance.

2.b Analytical Expression

We will find the analytical expression by taking the laplace transformation of (0.1a) and inserting it into (0.1b), knowing that the initial conditions of ψ_w are zero we get

$$
\begin{aligned}
\mathcal{L}\left\{\dot{\xi_w} = \psi_w\right\} \\
\xi_w = \frac{\psi_w}{s} \\
s\psi_w(s) = -\omega_0^2\frac{\psi_w(s)}{s} - 2\lambda\omega\psi_w(s) + K_w w_w(s) \\
(s + 2\lambda\omega_0 + \frac{\omega_0^2}{s})\psi_w(s) = K_w w_w(s) \\
\implies H(s) = \frac{\psi_w(s)}{w_w(s)} = \frac{K_w}{s + 2\lambda\omega_0 + \frac{\omega_0^2}{s}} = \frac{K_w s}{s^2 + 2\lambda\omega_0 s + \omega_0^2} \qquad (2.1)
\end{aligned}
$$

In order to find the analytical expression for the power spectrum density denoted as P_{ψ_w} we can use the transfer function

$$P_{\psi_w}(j\omega) = P_{w_w}(j\omega)|H(j\omega)|^2 \qquad (2.2)$$

Since w_w is a zero mean white noise with unit variance we know that its PSD function, $P_{w_w}(j\omega)$, will be equal the variance of w_w, $\sigma_w^2 = 1$. This means that we can simplify the expression for P_{ψ_w} and find its analytical form

$$
\begin{aligned}
P_{\psi_w} &= |H(j\omega)|^2 = H(j\omega)H(-j\omega) && (2.3) \\
&= \frac{jK_w\omega}{(j\omega)^2 + j2\lambda\omega_0\omega + \omega_0}\frac{-jK_w\omega}{(-j\omega)^2 - j2\lambda\omega_0\omega + \omega_0} && (2.4) \\
&= \frac{K_w^2\omega^2}{\omega_0^4 + \omega^4 + 2\omega_0^2\omega^2(2\lambda^2 - 1)} \overset{\mathbb{R}}{=} P_{\psi_w}(\omega) && (2.5)
\end{aligned}
$$

2.c Resonance Frequency

The resonance frequency ω_0 is the frequency at which the wave disturbance affects the ships heading the most. We can find the resonance frequency by finding the maximum value of the power spectral density function S_{ψ_w}, this is easily done in Matlab with the following code:

```
[max_pxx, index] = max(pxx);
omega_0 = omega(index);
```

This gave us $w_0 = 0.7823$

2.d Identifying The Damping Factor λ

In order to complete our model we need to find the damping factor λ. We defined $K_w = 2\lambda\omega_0\sigma$, where σ is the square root of the maximum intensity of S_{ψ_ω}. We plotted the analytical expression against the estimated PSD, and found λ by trial and error. The curve for different values of λ is shown i figure 5. We found $\lambda = 0.08$ to be a very good fitting value.

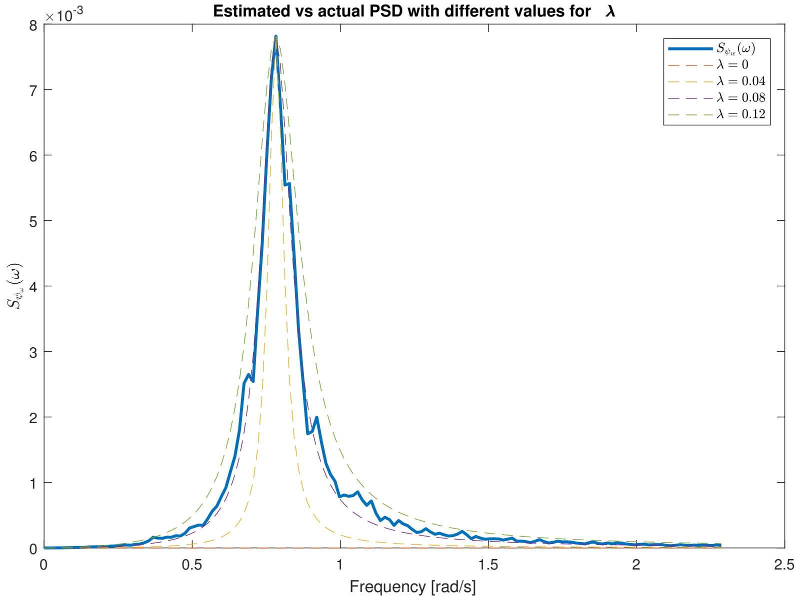

Figure 5: Plot of estimated vs actual PSD with different values for λ

3 Control system design

In this problem we want to control the ship's heading using a PD-controller.

3.a Designing PD-controller

We want to design a PD-controller on the form $H_{pd} = K_{pd}\frac{1+T_d s}{1+T_f s}$ and from section 1.a we know that we can model the dynamics in the plant as $H(s) = \frac{K}{s(Ts+1)}$. If we chose $T_d = T$, we get the open-loop model (3.1) which can be used to find K_{pd} and T_f.

$$\begin{aligned} h(s) = H(s)H_{pd}(s) =& \frac{KK_{pd}(1+T_d s)}{s(1+Ts)(1+T_f s)} \\ h(s) =& \frac{KK_{pd}}{s(1+T_f s)} \end{aligned} \tag{3.1a}$$

We want our phase-margin to be $\theta = 50°$ and $\omega_c = 0.01s^{-1}$. Taking the phase and magnitude of (3.1), we can find T_f and K_{pd}.

$$\begin{aligned} \angle h(j\omega_c) - (-180°) &= \theta \\ \angle\frac{KK_{pd}}{j\omega_c - T_f\omega_c^2} &= 50° - 180° \\ \angle(KK_{pd}) - \angle(j\omega_c - T_f\omega_c^2) &= -130° \\ 0 - \arctan(\frac{w_c}{-T_f\omega_c^2}) &= -130° \\ T_f &= -\frac{1}{\tan(130°)\omega_c} \\ T_f &= 8.39s \end{aligned} \tag{3.2a}$$

$$\begin{aligned} |H(j\omega_c)| = \frac{|KK_{pd}|}{|j\omega_c - T_f\omega_c^2|} &= 1 \\ \frac{KK_{pd}}{\sqrt{\omega_c^2 + (T_f\omega_c^2)^2}} &= 1 \\ K_{pd} = \frac{\sqrt{\omega_c^2 + (T_f\omega_c^2)^2}}{K} & \end{aligned} \tag{3.3a}$$

Using (3.3a), and $T_f = 8.39s$ from (3.2a), $T = 72.43s$ and $K = 0.1516$ from section 1.b, we find $K_{pd} = 0.8361$.

3.b Autopilot with Measurement Noise

In this section we implemented the PD-controller we designed in subsection 3.a in Simulink, and plotted our response with measurement-noise on, shown in Figure 6. We used a saturation-block to avoid an unreasonable rudder input (i.e. within $\pm 35°$) as seen in Figure 18 in Appendix B. There were no disturbance from waves or current.

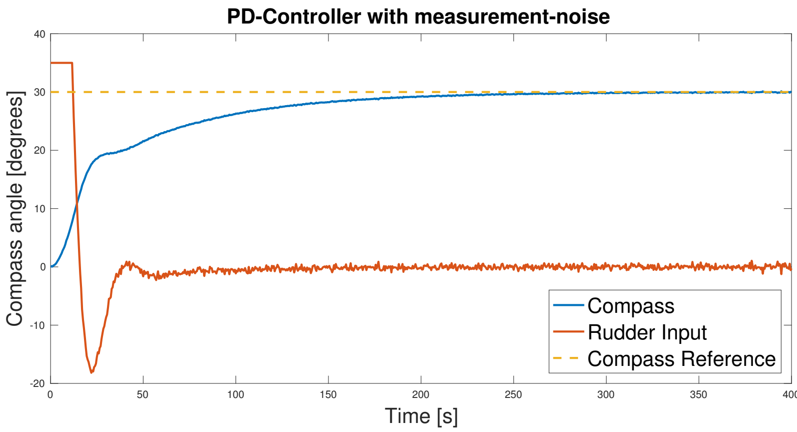

Figure 6: PD-controller with measurement noise.

We observe that the response of the system does not overshoot, and the ship reaches it reference $\psi_{ref} = 30°$ after around 300 seconds or 5 minutes. Because of the high time-constant in the system this is acceptable, but still a slow response.
We also see that the rudder input is at maximum to start with, and the ship is turning quickly. The rudder does however change the angle in negative direction, before it stabilizes at $0°$. This is because of the derivative effect of the PD-controller, which compensates for the rate of change being too high to avoid overshooting. This is what makes the PD-controller preferable to a P-controller.
Another interesting observation is that the rudder movement is more noisy than the measured compass angle. This is because we gain the high-frequency noise when we differentiate the signal, and get a movement in the rudder input that cause the rudder to wear. So even though the ship reaches it's compass heading, we have not yet gained optimal control.

3.c Autopilot with Current-disturbance

In this problem we turned on both the measurement noise and the current-disturbance, but kept the wave-disturbance off. The plot of the response is shown in Figure 7.

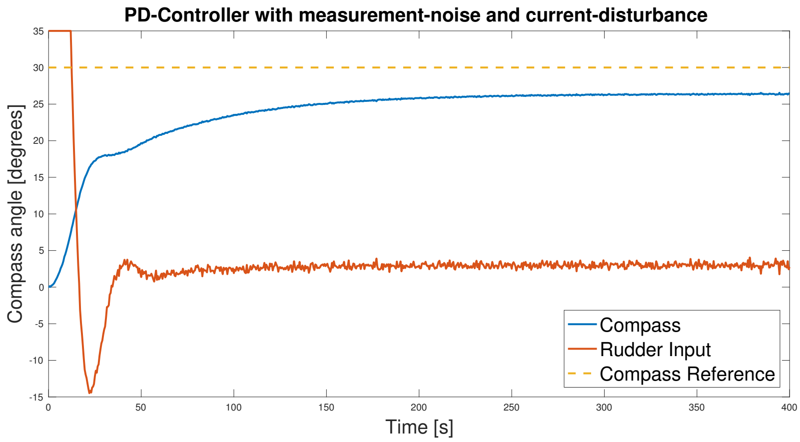

Figure 7: PD-controller with measurement noise and disturbance from current.

We see that the response is quite like to the one in subsection 3.b, except there is an steady-state error in the compass course. We get this because of the current working on the ship. The rudder input finds a stationary value, but unlike before, it's not around zero. The rudder is compensating such that the course-deviation is constant and this is because the PD-controller compensates for the change of rate. However, it is unable to get rid of the error.

3.d Autopilot with Wave-disturbance

We now turned off the current-disturbance, turned on the wave-disturbance and as before kept the measurement noise on. The ships response is shown in Figure 8.

We can observe that the ship reaches it's heading at around the same time as in Figure 6, but there is a lot of unnecessary movement in the ship on the way. The large oscillations in the rudder input seen in Figure 8 is caused by differentiating the signal which is now polluted by a high frequency disturbance. The rudder input has completely unrealistic movement to compensate for the waves, and it will quickly wear out using the PD-controller.

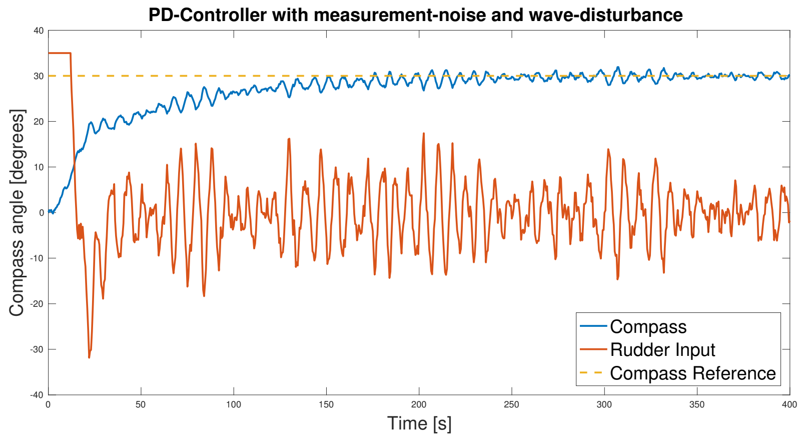

Figure 8: PD-controller with measurement noise and disturbance from waves.

4 Observability

4.a State-space Model

The matrices **A**, **B**, **E** and **C** in the statespace system is found by first differentiating the state vector which is equivalent to the model in (0.1) and solving for the state space model.

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{\xi_w} \\ \dot{\psi_w} \\ \dot{\psi} \\ \dot{r} \\ \dot{b} \end{bmatrix} = \begin{bmatrix} \psi_w \\ -\omega_0^2 \xi_w - 2\lambda\omega_0\psi_w + K_w w_w \\ r \\ -\frac{1}{T}r + \frac{K}{T}(\delta - b) \\ w_b \end{bmatrix}$$

$$\dot{\mathbf{x}} = \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ -\omega_0^2 & -2\lambda\omega_0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -\frac{1}{T} & -\frac{K}{T} \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}}_{\mathbf{A}} \mathbf{x} + \underbrace{\begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{K}{T} \\ 0 \end{bmatrix}}_{\mathbf{B}} u + \underbrace{\begin{bmatrix} 0 & 0 \\ K_w & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}}_{\mathbf{E}} \mathbf{w} \tag{4.1}$$

$$y = \underbrace{\begin{bmatrix} 0 & 1 & 1 & 0 & 0 \end{bmatrix}}_{\mathbf{C}} \mathbf{x} + v \tag{4.2}$$

4.b Observabillity without Disturbance

Without disturbance in the system we are left with the state vector

$$\mathbf{x} = \begin{bmatrix} \psi \\ r \end{bmatrix}$$

Which gives the matrices

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{1}{T} \end{bmatrix} \text{ and } \mathbf{C} = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

The system will be observable if the observability matrix has full rank. We calculate the observability matrix in MATLAB with the function

`Ob = obsv(A, C)` we achieve `Ob` on the form

$$\mathcal{O} = \begin{bmatrix} \mathbf{A} \\ \mathbf{CA} \\ \vdots \\ \mathbf{CA}^{n-1} \end{bmatrix}$$

And inserting values gives

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

where $\mathcal{O}$ has full rank and the system is observable with no disturbance.

4.c Observabillity with Current-disturbance

With current disturbance the bias to the rudder will be applicable, and the state vector will be

$$\mathbf{x} = \begin{bmatrix} \psi \\ r \\ b \end{bmatrix}$$

Which gives the matrices

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -\frac{1}{T} & -\frac{K}{T} \\ 0 & 0 & 0 \end{bmatrix} \text{ and } \mathbf{C} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$$

By calculating the observability matrix in MATLAB, see appendix A.d, as we did in subsection 4.b, we get

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

And $\mathcal{O}$ has full rank and the system is observable with current disturbance.

4.d Observabillity with Wave-disturbance

Due to the wave disturbance, the average heading for the system will encounter disturbance given by ξ_w and ψ_w, giving the state vector

$$\mathbf{x} = \begin{bmatrix} \xi_w \\ \psi_w \\ \psi \\ r \end{bmatrix}$$

Which gives the matrices

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\omega_0^2 & -2\lambda\omega_0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -\frac{1}{T} \end{bmatrix} \text{ and } \mathbf{C} = \begin{bmatrix} 0 & 1 & 1 & 0 \end{bmatrix}$$

By calculating the observability matrix in MATLAB, see appendix A.d, like in subsection 4.b,we get

$$\mathcal{O} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ -0.6120 & -0.1878 & 0 & 1 \\ 0.1149 & -0.5768 & 0 & -0.0138 \\ 0.3530 & -0.2232 & 0 & 0.000 \end{bmatrix}$$

And $\mathcal{O}$ has full rank and the system is observable with wave disturbance.

4.e Observability with both current and wave disturbance

By using $\mathbf{A}$ and $\mathbf{C}$ from (4.1) and (4.2) to calculate the observability matrix,

$$\mathcal{O} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ -0.6120 & -0.1878 & 0 & 1 & 0 \\ 0.1149 & -0.5768 & 0 & -0.0138 & -0.0022 \\ 0.3530 & 0.2232 & 0 & 0.0002 & 0.0000 \\ -0.1366 & 0.3111 & 0 & -0.0000 & -0.0000 \end{bmatrix}$$

with rank $= 5$ which means that the system is observable with both current and wave disturbance as well.

Now that we know that the system is observable for all disturbances we also know that we are able to find its initial states from a measured output. This makes the use of estimators possible which we will be using to improve our regulator with a Kalman filter in part 5.

5 Discrete Kalman filter

5.a Discretization

To improve the PD controller, a Kalman Filter was put to use. By applying Kalman filter to the system we can estimate the states in the system, so that we can filter out the noise-components. This is beneficial for the ship which is heavily affected by disturbances from waves and currents.

The matrices where found by using exact discretization of the model given in subsection 4.a. It is given that $\mathbf{C_d} = \mathbf{C}$ and $\mathbf{D_d} = \mathbf{D}$, and therefor only $\mathbf{A_d}$, $\mathbf{B_d}$ and $\mathbf{E_d}$ is found by using the command `c2d` in MATLAB. The command converts the model from continuous to discrete time, with sampling frequency of 10 Hz which gives step size $T_s = 0.1s$.

$$\mathbf{A_d} = \begin{bmatrix} 0.9970 & 0.0990 & 0 & 0 & 0 \\ -0.0606 & 0.9784 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0.0999 & -0.0000 \\ 0 & 0 & 0 & 0.9986 & -0.0002 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \mathbf{B_d} = \begin{bmatrix} 0 \\ 0 \\ 0.0000 \\ 0.0002 \\ 0 \end{bmatrix}$$

$$\mathbf{C_d} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \end{bmatrix}, \mathbf{D_d} = 0 \text{ and } \mathbf{E_d} = \begin{bmatrix} 0.0000 & 0 \\ 0.0016 & 0 \\ 0 & -0.0000 \\ 0 & -0.0000 \\ 0 & 0.1000 \end{bmatrix}$$

5.b Estimation of Variance

To measure the measurement-noise variance σ^2, the disturbances from current and waves was turned off. We also put the rudder input $\delta = 0$, so that our output was a vector of only measurement noise. The function `var` was used in MATLAB to calculate σ^2.

$$\sigma^2 = 6.1614 * 10^{-7}$$

5.c Implementation of Discrete Kalman Filter

We are given the following in the assignment in order to implement the discrete Kalman filter

$$\mathbf{w} = \begin{bmatrix} w_w & w_b \end{bmatrix}^T, E\{\mathbf{w}\mathbf{w}^T\} = \mathbf{Q} = \begin{bmatrix} 30 & 0 \\ 0 & 10^{-6} \end{bmatrix}$$

$$\mathbf{P}_0^- = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0.013 & 0 & 0 & 0 \\ 0 & 0 & \pi^2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 2.5 \cdot 10^{-3} \end{bmatrix}, \hat{\mathbf{x}}_0^- = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

We decided to write the filter as a MATLAB function and denote the a priori estimate as $\hat{\mathbf{x}}^-$ and the posteriori estimate as $\hat{\mathbf{x}}$. (During the implementation in MATLAB the a priori was denoted $\mathbf{x_m}$ and the a posteriori denoted $\mathbf{x}$.) Furthermore, we implemented the Kalman gain, L, then we updated $\mathbf{x}$ and the error co-variance $\mathbf{P}$. Finally we projected ahead by updating the a priori matrices. The steps in the method can be seen below and the implementation can be seen in appendix appendix A.f.

$$\begin{aligned}
\mathbf{L}[k] &= \mathbf{P}^-[k]\mathbf{C}^T(\mathbf{C}\mathbf{P}^-[k]\mathbf{C}^T + \mathbf{R})^{-1} \\
\hat{\mathbf{x}}[k] &= \hat{\mathbf{x}}^-[k] + \mathbf{L}[k](\mathbf{y}[k] - \mathbf{C}[k]\hat{\mathbf{x}}^-[k]) \\
\mathbf{P}[k] &= (\mathbf{I} - \mathbf{L}[k]\mathbf{C})\mathbf{P}^-[k](\mathbf{I} - \mathbf{L}[k]\mathbf{C})^T + \mathbf{L}[k]\mathbf{R}\mathbf{L}^T[k] \\
\hat{\mathbf{x}}^-[k+1] &= \mathbf{A}\mathbf{x}[k] + \mathbf{B}\mathbf{u}[k] \\
\mathbf{P}^-[k+1] &= \mathbf{A}\mathbf{P}[k]\mathbf{A}^T + \mathbf{E}\mathbf{Q}\mathbf{E}^T
\end{aligned}$$

5.d Bias Feedforward

Using the Kalman Filter we implemented in the last section, we were now able to estimate our bias. By using the bias in a feedforward-loop (still having the measured compass course in our PD-controller), we were able to remove the steady-state error we got in section 3.c. The ship's response is shown in Figure 9, and the `Simulink` diagram is shown in fig. 19.

Figure 9 shows the compass course and the rudder input with and without the feedforwarded bias. We can see that the ship is reaching its desired course after around 300seconds, and by looking at the rudder input we can observe how we got rid of the stationary error. We see that there is very little difference between the the two rudder inputs (with and without $\hat{b}_{feedforward}$, except at the crucial point right before the rudder input stabilizes. At this point, since we add the estimated bias, we get a slightly higher value for the rudder input. This is enough to lift the compass course to the desired course. When the rudder stabilizes the two rudder inputs are equal, meaning that

they both can counteract the current.
We can also observe that we still have some noise on our rudder input, which is undesirable. In the next task we will try to get rid of this, even with wave-disturbance turned on.

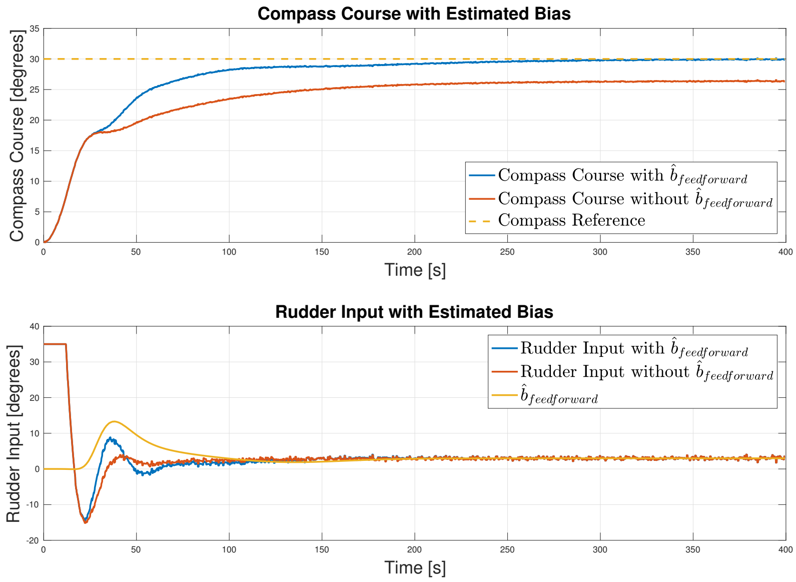

Figure 9: The ship's response using estimated bias to compensate for current.

5.e Kalman Filtered Feedback

In this problem we finally turned on all disturbances and were still able to reach our desired heading. Instead of using our measured compass states, which is polluted with both measurement noise and wave-disturbances, we used the Kalman Filter to estimate our compass state and used this in our PD-controller. The ship's response is shown in Figure 10. The `Simulink` diagram is shown in fig. 20.

If we look at the compass heading, Figure 10, the first observation is that the ship reaches its heading at approximately the same time as before, even though the waves rock the ship. This looks a lot like the response we got in section 3.d, i.e Figure 8. The problem with our system in section 3.d was that the rudder input had a completely unrealistic behaviour, but in Figure 10 we can see that this problem now is almost removed as the input is smooth, although there is quick movement in the beginning. This behaviour in the rudder input is a lot better as it won't cause unnecessary wear in the actual rudder.

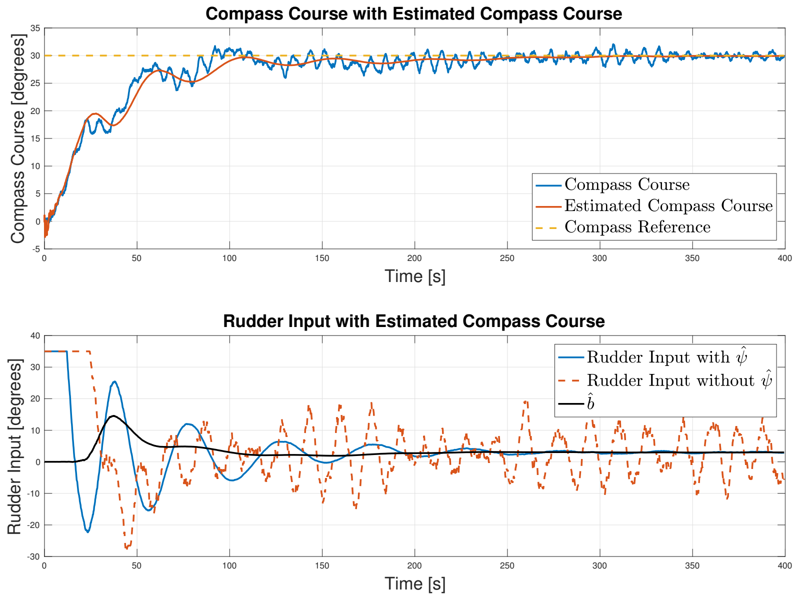

Figure 10: The ships response using estimated compass feedback and the estimated compass course.

To compare the actual and estimated wave-influence, we change the output of our Kalman Filter to give us the estimated wave-influence. We do this by changing `psi = x(3);` to `psi_w = x(2);` in line 25 in appendix A.f. We also set the rudder input to constant 0, and the whole `Simulink` diagam can be seen in appendix B, Figure 21. In Figure 11 we can see that the wave estimate from the Kalman Filter is a little off in the beginning, but once it has corrected for some seconds its very good.

5.f Changing Q

First of all we examine the original Q-matrix. We also repeat the equation for the a priori error co-variance matrix $\mathbf{P}^-$ from Equation 5.1.

$$\mathbf{Q} = \mathbf{E}[\mathbf{w}\mathbf{w}^T] = \mathbf{E}(\begin{bmatrix} w_w^2 & w_w w_b \\ w_b w_w & w_b^2 \end{bmatrix}) = \begin{bmatrix} 30 & 0 \\ 0 & 10^{-6} \end{bmatrix} \tag{5.2}$$

$$\mathbf{P}^-[k+1] = \mathbf{A}\mathbf{P}[k]\mathbf{A}^T + \mathbf{E}\mathbf{Q}\mathbf{E}^T \tag{5.3}$$

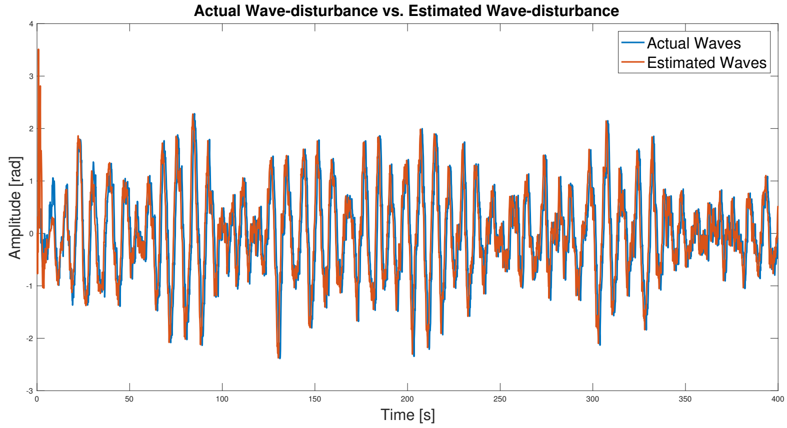

Figure 11: Actual wave-disturbance compared to waves estimated by the Kalman Filter.

First of all we can say that the it makes sense to have the off-diagonal equal to zero, as we assume that the two disturbances are independent of each other. We include Equation 5.3 because this is where $\mathbf{Q}$ is included in the Kalman Filter. This line of in the algorithm is responsible for updating the uncertainty matrix $\mathbf{P}^{-}$ by using our previous uncertainty added with "average" disturbance from the environment. By following the algorithm, that means that a high value in $\mathbf{Q}$ gives a high uncertainty in the model (before we include measurements). This leads to a high Kalman gain, which really means that we should trust our measurements more than the simulation.
First we examine what happens to the system in section 5.d. In Figure 12 we observe that we do in fact reach our destination with a large $\mathbf{Q}$, but it follows a different path and the overall performance is worse than before.

In the opposite case, i.e $\mathbf{Q} = 0$, the Kalman gain is lower and we trust our model to be to quite good. The ships response is shown in Figure 13. Here we see that we get a pretty good response, meaning that our our model-estimation is in fact quite good. This is because the current-disturbance has a very low co-variance $\mathbf{E}[\mathbf{w_w^2}] = 10^{-6}$ to begin with.

Now we examine the what happens to the system from section 5.e The response with very big Q is plotted in Figure 14. We see that the ship still reaches its heading, but with a lot worse performance compared to before. The Kalman Filter is using its measurements, but the measurements are polluted and the trajectory of the ship is affected by the waves.

When $\mathbf{Q}$ is 0 with wave disturbance is on, the Kalman gain is low and we trust our model to be to quite good. The ships response is shown in Figure 15. We can observe that the ship reaches its desired heading, but is

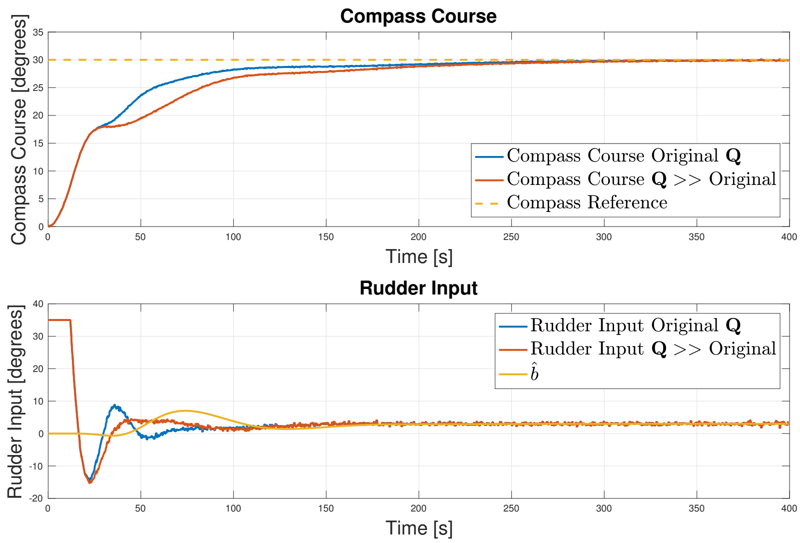

Figure 12: Ships response from section 5.d with very large $\mathbf{Q}$.

unable to keep it. This is because the Kalman Filter is having trouble with estimating the influence of the disturbances, which is quite is not surprising as we say that on average there are no disturbances (by setting $\mathbf{Q} = 0$) and the waves have a high co-variance.

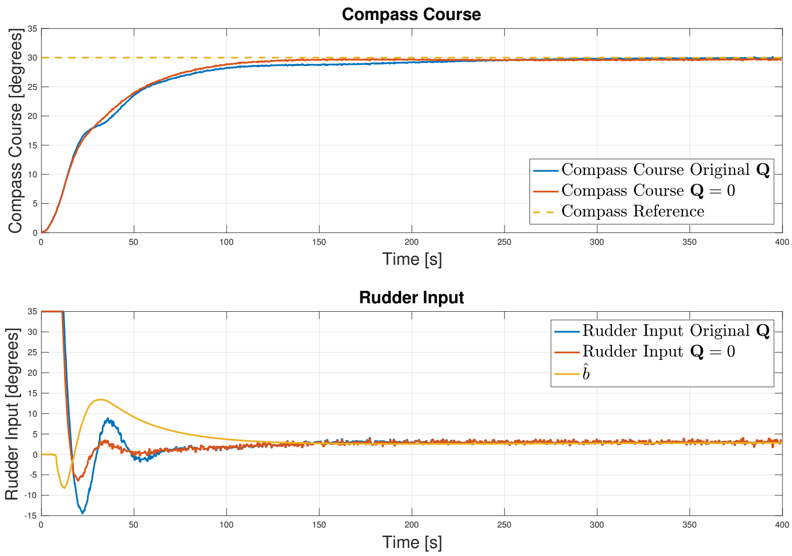

Figure 13: Ships response from section 5.d with $\mathbf{Q} = 0$.

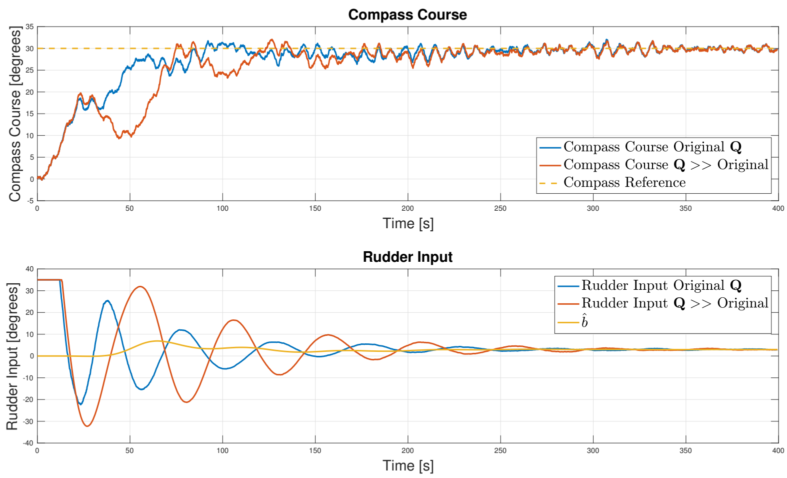

Figure 14: Ships response from section 5.e with very large $\mathbf{Q}$.

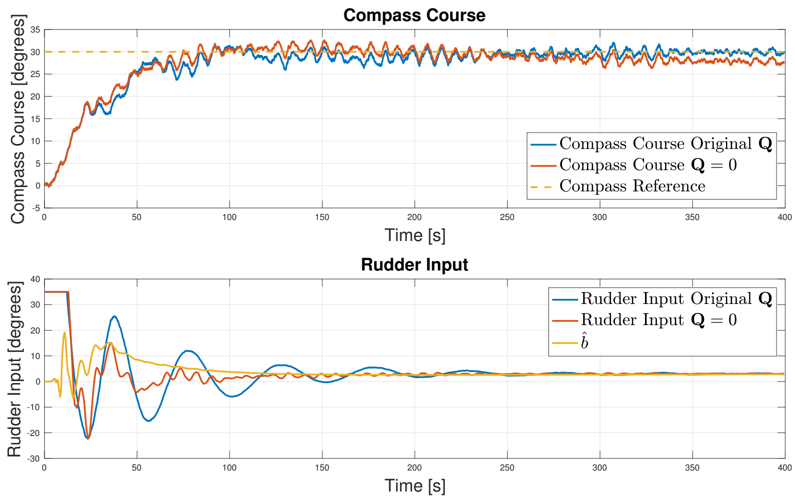

Figure 15: Ships response from section 5.e with $\mathbf{Q} = 0$.

6 Conclusion

In this assignment we have tried to control a ship affected by three types of disturbances, noise, current and waves. We started by identifying the boat-parameters, which we hindsight probably could have been more precise. However, we do not think it lead to worse performance later in the task.

Afterwards we analyzed the wave-disturbance, and modeled it using the power spectral density function. We also implemented a a PD-controller, but observed that it was not good enough to handle disturbances from current or waves. Therefore, we used a Kalman Filter to estimate the bias from the current and to smooth our measurements. Using the Kalman Filter we achieved good control over the system.

A MATLAB Code

A.a Part 1

```
%% 5.1a) Finding boatparamaters using no disturbances

load('amplitude_1_1.mat'); % Load data files
load('amplitude_2_1.mat');

A_1 = (max(amplitude_1_1(2,2000:20000))- ...
    min(amplitude_1_1(2,2000:20000)))/2;
A_2 = (max(amplitude_2_1(2,2000:20000))- ...
    min(amplitude_2_1(2,2000:20000)))/2;  % Calculate amplitude

w_1 = 0.005;
w_2 = 0.05;

% Calculate boatparamaters
T = sqrt((A_1^2*w_1^2 - w_2^2*A_2^2) / ...
    ((w_2^4*A_2^2 -A_1^2*w_1^4)));   % K = 0.1561
K = A_1*w_1 *sqrt(T^2*w_1^2 +1);     % T = 72.43
```

```
%% 5.1b) Repeating 5.1a using disturbances

load A_1_noise.mat;  % Load datafiles
load A_2_noise.mat;

plot(A_2_n(1,2000:20000), A_2_n(2,2000:20000));

A_1 = (max(A_1_n(2,2000:20000))- ...
    min(A_1_n(2,2000:20000)))/2;
A_2 = (max(A_2_n(2,2000:20000))- ...
    min(A_2_n(2,2000:20000)))/2; % Calculate amplitudes

w_1 = 0.005;
w_2 = 0.05;

% Calculate boatparamaters
T = sqrt((A_1^2*w_1^2 - w_2^2*A_2^2) / ...
    ((w_2^4*A_2^2 -A_1^2*w_1^4)));   % T = 9.858
K = A_1*w_1 *sqrt(T^2*w_1^2 +1);    % K = 0.1592
```

```
%% 5.1d) Model vs ship response
P5p1b_init;
```

```
load step_response_deg.mat;
load step_response_tf_deg.mat; % Load datafiles

t = step_res(1,1:10000);
plot(t, step_res(2,1:10000), 'b', t, step_res_tf(2,1:10000), 'r');
xlabel('Time [s]')
ylabel('Amplitude [degrees]')
title('Step Response of Ship and Model')
lg = legend('Ship', 'Model');
lg.FontSize = 16;
```

A.b Part 2

```
%% 5.2a) - Finding estimate of PSD
P5p1b_init;

load('wave.mat');
fs = 10;
[pxx, f] = pwelch(psi_w(2,:).*(pi/180), 4096, [], [], fs);

pxx = pxx./2*pi;
omega = f.*2*pi;

figure();
plot(omega(1:150), pxx(1:150), 'LineWidth', 2);

%% 5.2c) Finding omega 0 and standard deviation from PSD
[max_pxx, index] = max(pxx);
omega_0 = omega(index);
sigma = sqrt(max_pxx);

%% 5.2d Finding a lambda based on PSD
figure()
for lamda = 0.00:0.04:0.12
    K_omega = 2 .* lamda .* omega_0 .* sigma;

    P_psi = ((omega.*K_omega).^2)./ ...
        (omega.^4 + omega_0^4 + 2*omega_0^2.*omega.^2*(2*lamda.^2-1));

    plot(omega(1:150), P_psi(1:150),'--',omega(1:150), pxx(1:150));
    hold on
end
```

A.c Part 3

```
%% 5.3 - PD-controller

P5p1b_init;

%Implementing the PD controller found in 5.3a)
T_d = T; %T from 5.1b)
T_f = 8.39; %s
w_c = 0.1;
psi_r = 30; %deg
K_pd = sqrt(w_c^2 + T_f^2*w_c^4)/K; %K from 5.1b)
```

A.d Part 4

```
%% Init
clear all;
P5p1b_init;
p5p2;

%% 5.4 a) Finding matrices for statespace system
A = [0 1 0 0 0;
    -omega_0.^2 -2*lamda.*omega_0 0 0 0;
    0 0 0 1 0;0 0 0 -1/T -K/T;
    0 0 0 0 0];

B = [0; 0; 0; K/T; 0];

C = [0 1 1 0 0];

E = [0 0; K_omega 0; 0 0; 0 0; 0 1];

%% 5.4 b) Observability without disturbances

A_none = [0 1; 0 -1/T];
C_none = [1 0];

Ob_none = obsv(A_none, C_none);
unob_none = rank(Ob_none) - length(Ob_none),
%unob_none is the amount of unobservable states with no disturbances

%% 5.4 c) Observability with current disturbance
A_c = [0 1 0; 0 -1/T -K/T;0 0 0];
C_c = [1 0 0];
```

```
Ob_c = obsv(A_c, C_c);
unob_c = rank(Ob_c) - length(Ob_c),
%unob_c is the amount of unobservable states with current disturbance

%% 5.4 d) Observability with wave disturbance

A_w = [0 1 0 0;
    -omega_0.^2 -2*lamda.*omega_0 0 0;
    0 0 0 1;
    0 0 0 -1/T];

C_w = [0 1 1 0];

Ob_w = obsv(A_w, C_w);
unob_w = rank(Ob_w) - length(Ob_w),
%unob is the amount of unobservable states with wave disturbance

%% 5.4 e) Observability with both current and wave disturbance
% Use A and C from a)

Ob = obsv(A, C);
unob = rank(Ob) - length(Ob),
%unob is the amount of unobservable states
%with both wave and current disturbances
```

A.e Part 5

```
%% 5.5 - Making the Kalman Filter
p5p2_init;
p5p3b_init;
load('measurement_noise.mat');

% State-space matrices
A = [0 1 0 0 0;
    -omega_0.^2 -2*lamda.*omega_0 0 0 0;
    0 0 0 1 0;
    0 0 0 -1/T -K/T;
    0 0 0 0 0];

B = [0; 0; 0; K/T; 0];
C = [0 1 1 0 0];
E = [0 0; K_omega 0; 0 0; 0 0; 0 1];

```

```
T_s = 0.1; % Sampling Time

% Discretizing matrices
[~, B_d] = c2d(A, B, T_s);
[A_d, E_d] = c2d(A, E, T_s);
C_d = C;

var_measurement = var(v(2,:)); % Variance of measurement-noise

% Initialize matrices to be used in Kalman Filter
Q = [30 0; 0 10^-6];
P_0m = [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];
x_0m = [0; 0; 0; 0; 0];
R = var_measurement/T_s;
I = eye(5);

sys = struct('A_d', A_d, 'B_d', B_d, 'C_d', C_d, ...
    'E_d', E_d, 'Q', Q, 'R', R, 'P_0m', P_0m, 'x_0m', x_0m, 'I', I);
```

A.f Kalman Filter

```
function [b, psi] = kalman_filter(u, y, sys)

persistent init_flag A B C E Q R Pm xm I

if (isempty(init_flag))
    init_flag = 1;

    [A, B, C, E, Q, R, Pm, xm, I] = deal(sys.A_d, sys.B_d, sys.C_d, ...
        sys.E_d, sys.Q, sys.R, sys.P_0m, sys.x_0m, sys.I);

end
% Kalman Gain
    L = (Pm*C')/(C*Pm*C'+R);

% Update Estimate
    x = xm + L*(y-C*xm);

% Update Error
    P = (I - L*C)*Pm*(I-L*C)' + L*R*L';
```

```

% Project ahead
    xm = A*x + B*u;
    Pm = A*P*A' + E*Q*E';

psi = x(3); b = x(5);

end
```

B Simulink Diagrams

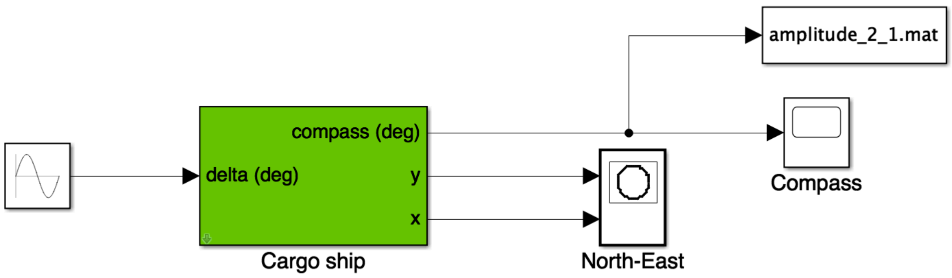

Figure 16: `Simulink` diagram for problem 5.1b and 5.1c.

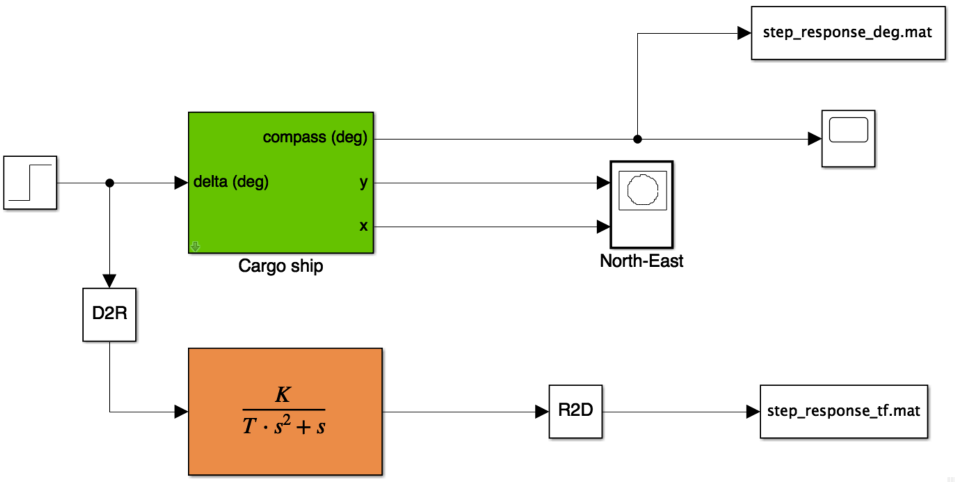

Figure 17: `Simulink` diagram for problem 5.1d

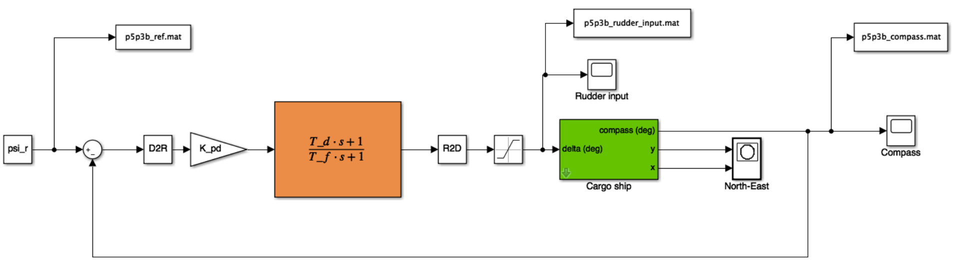

Figure 18: `Simulink` diagram for part 5.3

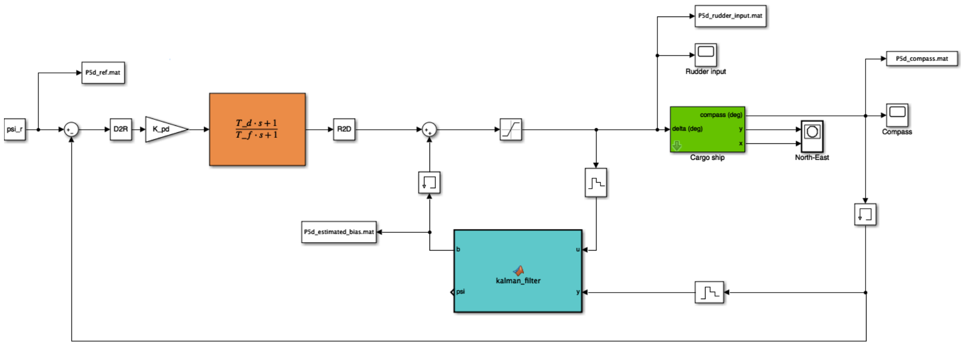

Figure 19: `Simulink` diagram for part 5.5d

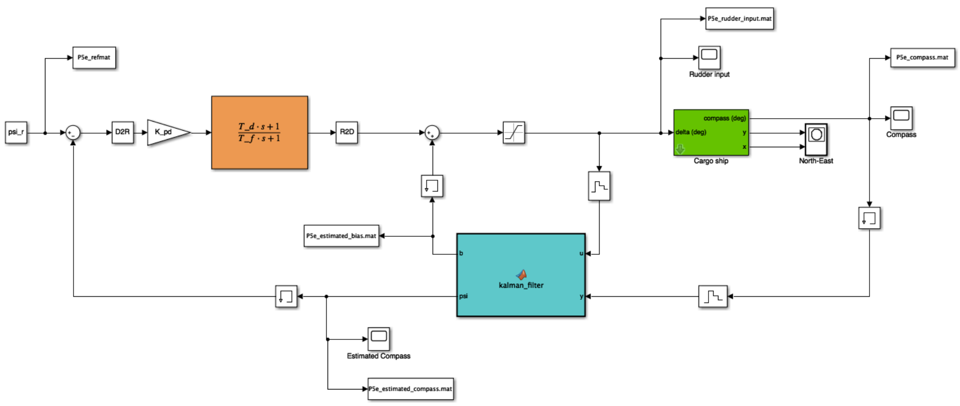

Figure 20: `Simulink` diagram for part 5.5e

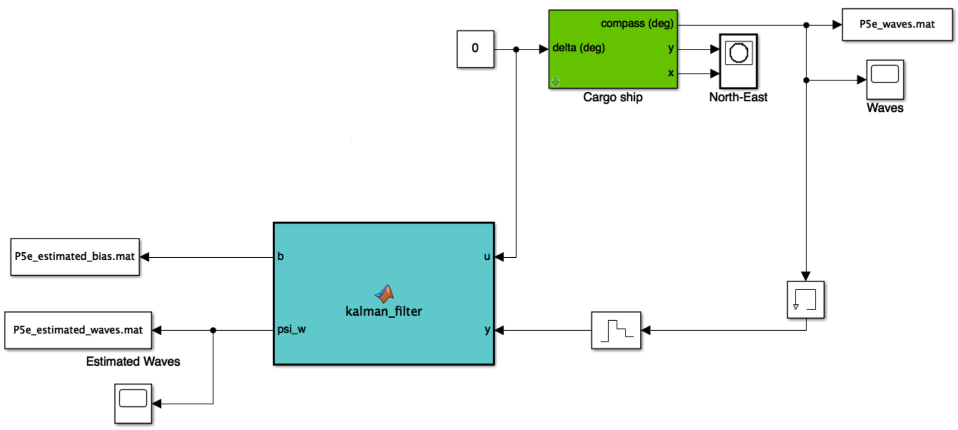

Figure 21: `Simulink` diagram for part 5.5e where we simulate waves