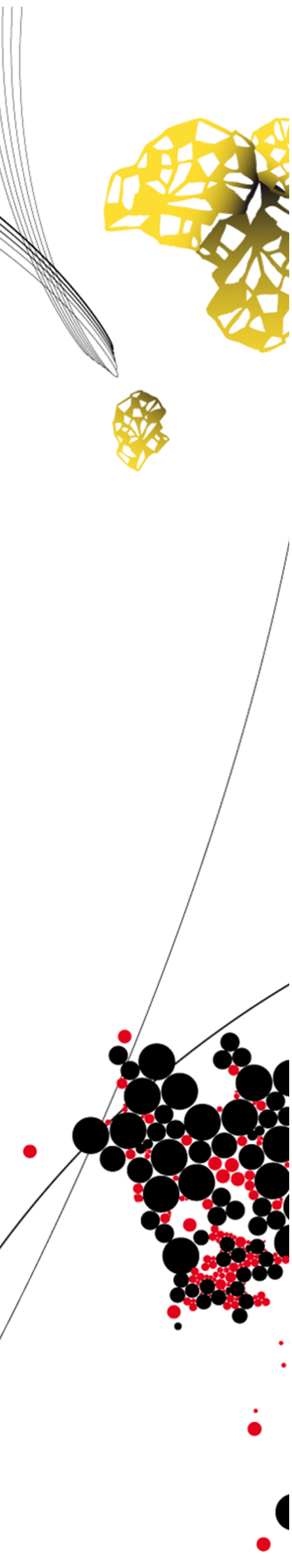


UNIVERSITY OF TWENTE.

Faculty of Electrical Engineering, Mathematics & Computer Science

Parameter Estimation using the Extended Kalman Filter for the ALIA Density Meter

Jasper Marissen
November 2021

Assessment Committee:
prof. dr. A.A. Stoorvogel
dr. ir. M. Beijen (Demcon)
dr. A. Betken

Chair: Hybrid Systems

Contents

1 Introduction

1.1 Background

When transporting a substance through a tube, it is often useful to know what the substance is made of, for example when mining or dredging. A way to find this out is by using a density meter. A density meter is a device that is fitted to the tube to measure the density of the substance inside the tube in (near) real-time, which means that the density is measured within a few seconds of the collection of the data.

The most accurate density meters that are used at this time are nuclear density meters. These work using the transmission of gamma radiation through the flowing substance to measure the mass per unit area. From this we can derive the density of the substance within the measuring tube. There are several drawbacks to the nuclear density meters. The use of any radioactive material is strictly regulated so the transportation, use, and disposal of radioactive material has to be approved by relevant authorities. Besides that, the maintenance of a nuclear density meter is very expensive and radiation hazards are always existent when using radioactive material.

Alia Instruments is a producer of process control equipment in various markets. The Alia Density Meter (ADM) aims to be an alternative to nuclear density meters without all the problems that are associated with radioactive material, as well as reduce the maintenance needed to run the density meters in the field.

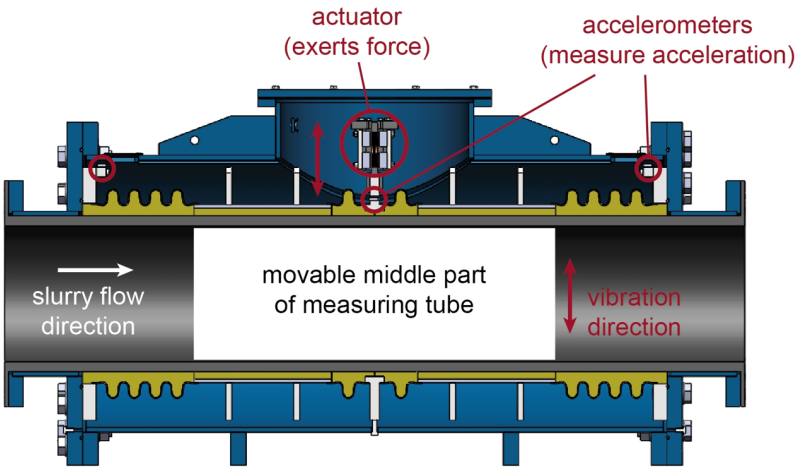


Figure 1.1

Figure 1.1 shows a cross section of the ADM. The ADM consists of an inner tube, or measuring tube, surrounded by a stiff outer tube, which ideally protects the inner tube from outside disturbances to the inner tube. The inner tube is designed such that it only moves perpendicular to the slurry flow. An actuator is placed on top of the inner tube to exert a force on the inner tube and make it vibrate. Three accelerometers are then placed on the tubes, one on the inner tube and two on the outer tube, to measure the acceleration of the

inner- and outer tube. The outer tube accelerometers are needed to account for any outside disturbances that influence the movement of the inner tube.

The movement of the inner tube in response to the force exerted by the actuator will be dependent on the mass of the slurry in the measuring tube. Therefore we can estimate the mass of the slurry by comparing the actuator force and the acceleration of the inner tube. Then we can find the density of the slurry by dividing the mass by the volume of the measuring tube.

The algorithm that is used in the current ADM works in certain situations, but has a few problems. The first is that it works by gathering data for a few seconds and then using that data to estimate the mass. This assumes that the mass does not change during those few seconds. This means that we cannot guarantee a good estimation if the mass changes. The second problem is that the current algorithm assumes that the slurry flows "horizontally" through the tube (i.e. from left to right in figure (1.1)). In reality, there are particles bouncing through the tube constantly, not just from left to right, but also up and down. If these particles disturb the inner tube too much, the current algorithm does not work well.

1.2 Parameter Estimation

In order to estimate the density of the slurry within the measuring tube (figure 1.1), we can look at the measuring tube as a physical system: the inputs of the system will be the actuator force that vibrates the inner tubes, and the vibrations of the outer tube. The relationship between the input and output will be dependent on the parameters of the model, specifically the mass of the slurry in the inner tube. The system can be represented by the following plant model:

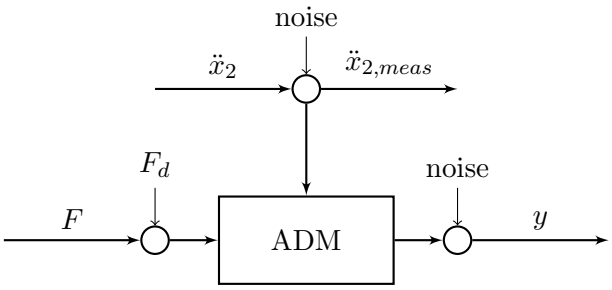


Figure 1.2: ADM model

In this model, F is the force that is generated by the actuator. This input is disturbed by a force F_d before going into the actual ADM model. The other input is the acceleration of the outer tube, which will influence the vibration of the inner tube. The outer tube acceleration is given by $\ddot{x}_2$. The outer tube acceleration is not exactly known, but it will be measured by an accelerometer, which has an measurement error associated with it. For the estimation, the noisy $\ddot{x}_{2,meas}$ will be used. The output y is the acceleration of the inner tube, which is also measured with accelerometers so it will also be disturbed by some measurement noise.

Once we have a model for the ADM with certain set of parameters, we can gather the (noisy) input- and output signals and find the parameters that fit the model "the best",

according to some criterion which we can choose.

1.3 Report lay-out

We will start in section (2) by defining the mathematical model that drives the ADM and stating several assumptions that are included. In section (3) we will formulate the main problem that this report will try to solve. Then in section (4) we will look at a few algorithms that have been used before in the ADM and note the problems with those algorithms to see why a new algorithm might be useful. In section (5) we will explain the algorithms already in use and introduce a new algorithm that will hopefully solve some of the issues that the old algorithms have. In section(6) we will use simulations to test the newly introduced algorithm and investigate its performance relative to the old algorithms. In section(7) we will apply our new estimation algorithm on real data from the field to see how it performs.

2 Modeling

In this section we will define the model used in the ADM for the purpose of parameter estimation, including assumptions on noise terms.

2.1 ADM model

It is important for parameter estimation that we have a good model for the system. The following assumptions deal with the ADM model.

Assumption 1): Between 50Hz and 200Hz, the ADM behaves as a second-order system.
Figure (2.1) shows a measured FRF (from F to $\ddot{x}_2$) of the ADM350, showing the fundamental frequency at around 70Hz and several high frequency modes, which are present due to the mechanics of the steel and rubber in the ADM. The figure shows that the high frequency eigenmodes start at around 200Hz. Between 50Hz and 200Hz, the system behaves as a second-order system. For this reason, we are only interested in the frequency content of the signals between 50 and 200 Hertz. In practice, this means that the the content of all input and output signals below 50Hz and above 200Hz will be filtered out before we do any calculations. The plot of the FRF shows that the main eigenfrequency of the system is located around 70HZ so the frequency band that we are focusing on includes the eigenfrequency and enough frequencies around it.

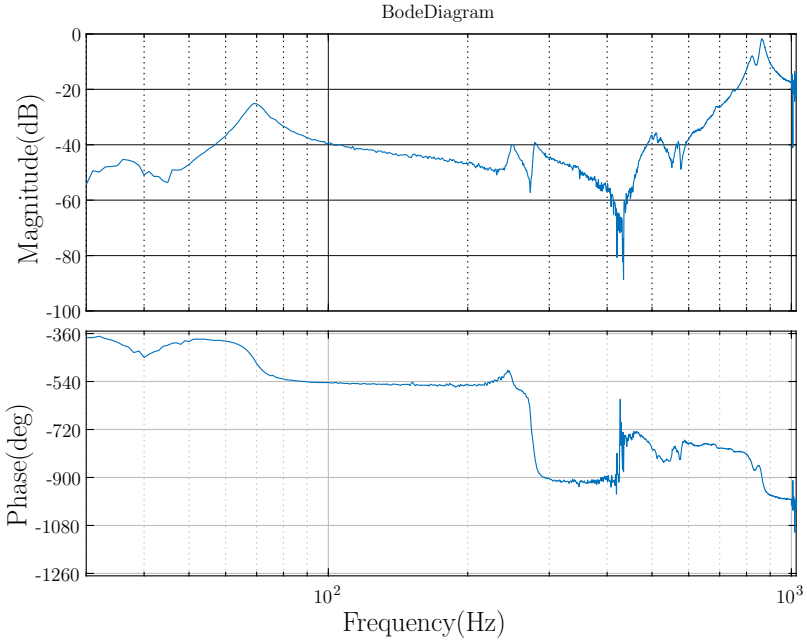


Figure 2.1: FRF measurement of the ADM350 (from F to $\ddot{x}_2$)

The inside of the tube is modelled as a single body of mass connected to the outer tube via the rubber inner tube, which is modelled as a spring. The outer tube is considered to be

infinitely heavy and stiff, so the vibrations of the inner tube do not affect the movement of the outer tube. Since we can measure the vibrations of the outer tube, we can use them as inputs for the system. Figure (2.2) shows the model of the system.

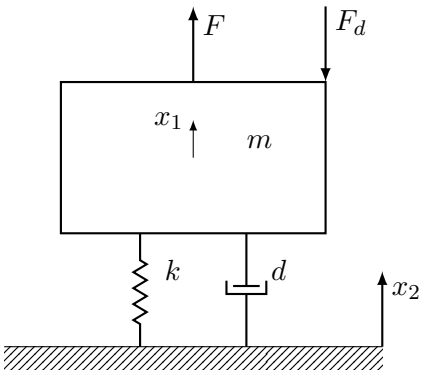


Figure 2.2

The equation of motion can be derived from Figure (2.2) as

$$m\ddot{x}_1(t) + d\dot{x}_1(t) + kx_1(t) = F(t) + d\dot{x}_2(t) + kx_2(t) - F_d(t). \tag{2.1}$$

Here $x_1(t)$ is the position of the inner tube, $x_2(t)$ is the position of the outer tube, m the mass of the inner tube and d and k are the spring constants. The force F that excites the system is a multi-sine consisting of integer frequencies between 50 and 200 Hertz. It is assumed that the input from the multi-sine is exactly known. The disturbance force F_d results from particles in the slurry bouncing against the inner tube. Accelerometers are placed on the inner tube to measure the acceleration of the inner tube and outer tube in discrete time. The system outputs are given by

$$y(t_n) = \ddot{x}_1(t_n) + \ddot{x}_{1,err}(t_n) \tag{2.2}$$

The sample rate used in the measurements is 4096Hz. The terms $\ddot{x}_{1,err}(t_n)$ in equation (2.2) is a measurement noise which is defined under the following assumption:

Assumption 2): The *measurement noise* is a zero-mean stochastic process with known power spectral density.
This can be seen from the sensor's data sheets [1]. The total measurement noise power density function is constant at a height of $1.4 \cdot 10^{-7}(m/s^2)^2/Hz$ at frequencies up to 300Hz. Since all signals are filtered, this means that we can model the measurement noise as uncorrelated noise.

We can write the differential equation (2.1) with the output (2.2) in state-space form using the difference between inner and outer tube as states.

The system input $\mathbf{u}(t)$ is a two dimensional vector made up of measurable random environmental vibrations that enter the system from the outer tube $x_2(t)$ and a random phase multi-sine $F(t)$ that consists of the integer frequencies from 50Hz to 200Hz. We assume that

the multi-sine input is exactly known, but the outer tube vibration measurements are not exact. We can write the actual input as the measured input $\mathbf{u}(t)$ subtracted by the measurement error $\mathbf{u}_{err}(t)$:

$$\mathbf{u}_{exact}(t) = \mathbf{u}(t) + \mathbf{u}_{err}(t) = \begin{bmatrix} \ddot{x}_{2,meas}(t) \\ F(t) \end{bmatrix} + \begin{bmatrix} -\ddot{x}_{2,err}(t) \\ 0 \end{bmatrix} \tag{2.3}$$

Here $\ddot{x}_{2,meas}$ is the measured acceleration and $\ddot{x}_{2,err}$ is the error on the outer tube acceleration measurements. The measurements of the acceleration of the outer tube are done using the same type of accelerometers as are used for the inner tube, but not the same ones. This means that we assume $\ddot{x}_{1,err}$ and $\ddot{x}_{2,err}$ have the same stochastic properties, but are independent of one another.
The state-space model in total is

$$\begin{aligned} \begin{bmatrix} \dot{x}_1(t) - \dot{x}_2(t) \\ \ddot{x}_1(t) - \ddot{x}_2(t) \end{bmatrix} &= \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{d}{m} \end{bmatrix} \begin{bmatrix} x_1(t) - x_2(t) \\ \dot{x}_1(t) - \dot{x}_2(t) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ -1 & \frac{1}{m} \end{bmatrix} \mathbf{u}(t) + \begin{bmatrix} 0 \\ -\frac{1}{m} F_d(t) + \ddot{x}_{2,err}(t) \end{bmatrix}, \\ y(t_n) &= \begin{bmatrix} -\frac{k}{m} & -\frac{d}{m} \end{bmatrix} \begin{bmatrix} x_1(t_n) - x_2(t_n) \\ \dot{x}_1(t_n) - \dot{x}_2(t_n) \end{bmatrix} + \begin{bmatrix} 0 & \frac{1}{m} \end{bmatrix} \mathbf{u}(t_n) - \frac{1}{m} F_d(t_n) + \ddot{x}_{1,err}(t_n). \end{aligned} \tag{2.4}$$

In this state-space model we will differentiate between two types of noise: process noise and output noise. The process noise, which will be noted by $\mathbf{w}(t)$, is the noise that is applied only to the state equations. The output noise, noted by $v(t)$, is the noise that is directly applied to the output equation.

$$\mathbf{w}(t) = \begin{bmatrix} 0 \\ \ddot{x}_{2,err}(t) - \frac{1}{m} F_d(t) \end{bmatrix}, v(t) = \ddot{x}_{1,err}(t) - \frac{1}{m} F_d(t) \tag{2.5}$$

Note that both the process noise and the output noise contain the disturbance force F_d, which means that $w(t)$ and $v(t)$ are correlated.

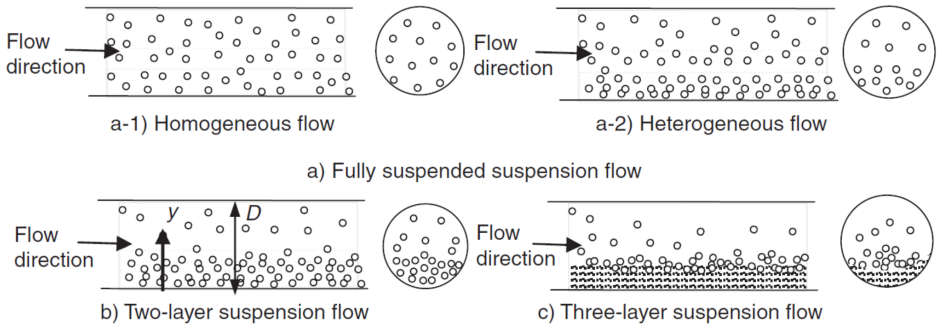


Figure 2.3: Types of slurry flow

Assumption 3): The slurry in the ADM acts as a single mass.
There are two types of slurry: settling and non-settling. In non-settling slurry the solid particles flow with the same speed as the liquid. In that case, the slurry in the ADM can be

assumed to act as a single mass. Within settling slurries, there are four situation that can arise, see Figure 2.3. The assumption made in the ADM is that all settling slurries act as a (pseudo-)homogeneous liquid (a), which means the particles follow the liquid flow completely and the slurry moves as a single mass.

Assumption 4): The *disturbance force* F_d is a zero-mean stochastic process with known power spectral density.
In the ideal case, the slurry flows horizontally through the tube perpendicular to the vibration direction (see figure 1.1). However, particles also move in the vibration direction and bounce against the inner tube. These bounces are modelled as a random disturbance force F_d, as shown in figure (2.2).

The effect of the disturbance force is found by estimating the process noise variance using a spectral analysis method [2, sec. 7.2.3]. Figure (2.4) shows an estimation of the total process noise power spectral density. The process noise consists of the disturbance force F_d (divided by the mass) and the error on the measured input $x_{2,err}$. Figure (2.4) shows an estimation of the PSD of the total process noise between 50 and 200 Hertz. The red line is the estimation, we model it as a flat PSD (the black dotted line at height $3 \cdot 10^{-6}$). We assume that F_d and $\ddot{x}_{2,err}$ are two independent random processes, so the PSD of F_d/m can be found by subtracting the PSD of $\ddot{x}_{2,err}$ from the PSD given in figure (2.4). Since the PSD of $\ddot{x}_{2,err}$ is also flat between 50Hz and 200Hz with a height of $1.4 \cdot 10^{-7}$, the PSD of F_d/m has a height of $2.9 \cdot 10^{-6}$. If we take the mass m as the mass of an empty tube (200kg), this means that the height of the PSD of F_d is $2.9 \cdot 10^{-6}$ at all frequencies between 50Hz and 200Hz. We assume that the spectral density is zero for all frequencies outside this range.

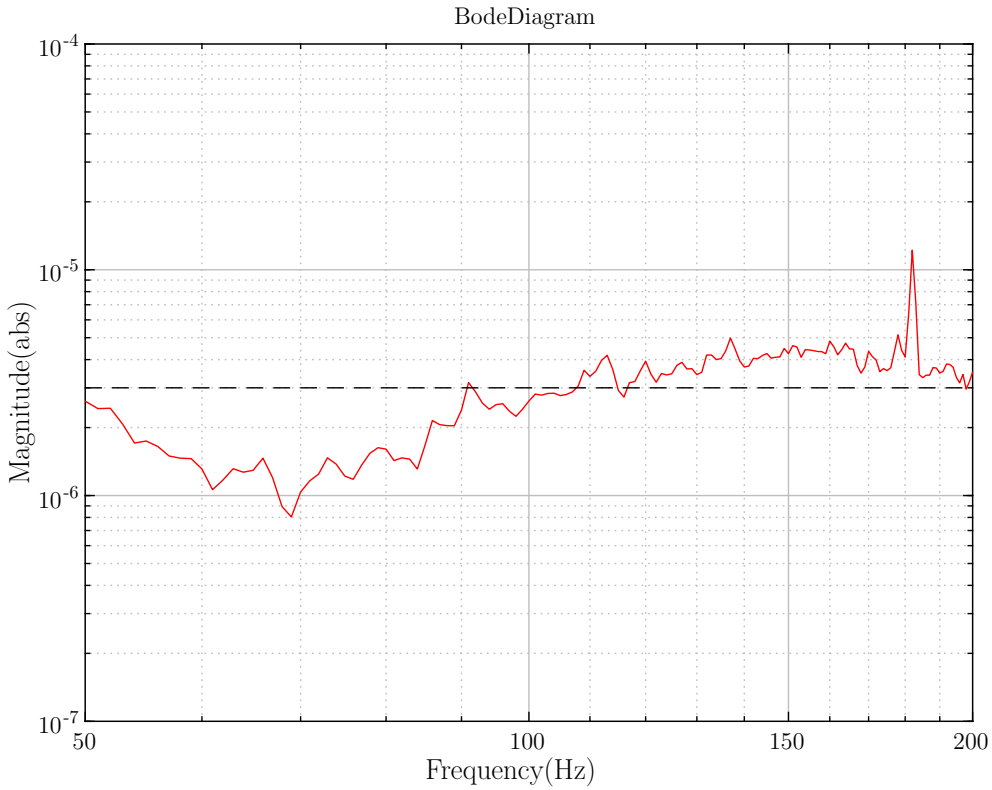


Figure 2.4: Process noise PSD estimate

Assumption 5): The parameters k and d are considered to be constant and unknown.
The parameters k and d can vary very slowly due to changing material properties (e.g. temperature changes and wear and tear). Temperature changes due to weather are at worst in the order of 0.01°C per second, such that the effect of the temperature gradient on the material properties is negligible with respect to the time window in which mass estimation will take place. In applications where the ADM will be used, the effect of erosion of the materials is expected to occur over the course of years. This means that for a measurement time of a few seconds, the variation is negligible, so we assume that the parameters are constant. On top of that, the values of k and d are not known, so they have to be estimated along with m.

In parameter estimations algorithms, we can either choose between a windowing algorithm or a continuously updating algorithm. The advantage of a windowing approach, is that there is less influence of process noise on the parameters, since a lot of data is averaged for estimation. The downside of windowing is that you have to make the assumption that the system dynamics are static throughout the window, which means that any changes in parameters during the window will result in an incorrect estimation and so a slower response time to dynamics changes. For continuously updating algorithms, it is also true that the more data from the past is used for the updating step, the less influence process noise has and the slower

the algorithm will adjust to changes in system dynamics. This trade-off is dependent on the application for which the ADM will be used. The changes that can occur in slurry mass are described as follows:

Assumption 6): The mass can have abrupt changes (at least 2% per second) or change slowly (less than 0.1% per second).
Based on the data from a dredging application, it is assumed that there are abrupt changes of more than 5kg/s (or about 2% per second) and slow variation of less than 0.2kg/s (less than 0.1% per second). The specifications require that the error in the density cannot exceed 0.5% given a window length of 1 second. This means that the slow changes are not detrimental to the requirements, but the abrupt changes exceed the threshold. To improve the existing algorithm, which uses 1 second windows, after the mass abruptly changes, the algorithm needs to adjust the mass estimate to get inside the threshold of 0.5% within 1 second.

3 Main question

In the last section we stated the model of the ADM and the background of the problem. In this section we will state the main question of this report and the problems that go with it.

The current algorithm that is used for estimation is the linear least squares (LLS) algorithm. The problem with the frequency domain LLS estimator are twofold.

- When the mass is constant, the LLS algorithm is inconsistent.
- The LLS algorithm, like all frequency domain parameter estimators, assumes that the parameters are constant during the window over which it estimates. This assumption is not always true in our case, so we cannot guarantee the correctness of the estimation during the periods in which the mass varies.

The first case is specifically in the case of the LLS algorithm, but the second case is inherent to frequency domain estimation algorithms, since frequency domain algorithms assume that parameters are constant during the whole window.

The main goal of this report is: given the model (2.4) for the ADM350, the parameter vector $\theta = (m(t), d, k)^T$ and a series of outputs $\mathbf{y_0}, \mathbf{y_1}, \ldots, \mathbf{y_N}$, improve the estimation of the parameter vector θ obtained from the existing LLS algorithm.

Subquestions:

1. Which parameter estimation algorithms have already been developed in the literature?
2. What type of algorithms are suitable for the ADM, given the assumptions stated in the previous chapter?
3. How does the the chosen algorithm(s) perform compared to the existing LLS algorithm?

The ADM350 will be used as a case study throughout this project, to run simulations and/or test the algorithm in practice. The algorithms will be tested on the following criteria

1. **Consistency**
   An estimator $\hat{\theta}$ is called consistent if it converges in probability to the true parameter value θ as the sample size goes to infinity, i.e. if $\lim_{n\to\infty} \mathtt{P}(|\hat{\theta}(n) - \theta| > \epsilon) = 0$ for all $\epsilon > 0$. Note that this definition is only valid if the estimated parameters are constant.
2. **Tracking fast changes**
   The estimation needs to be able to adapt to sudden changes in mass within a few seconds.

3. **Tracking slow changes**
   In the previous chapter we have seen that in addition to sudden jumps, the value of m can also change slowly. We would like the estimation to stay accurate during these periods, in the sense that the mean square error stays small.

4 Literature

In this section we will look at the frequency domain algorithm that is currently used in the ADM and note the problems that are associated with it. Other frequency domain estimators have been investigated, but they also come with their problems. We will introduce continuous updating algorithms based on Kalman Filters.

4.1 Windowing/frequency domain algorithms

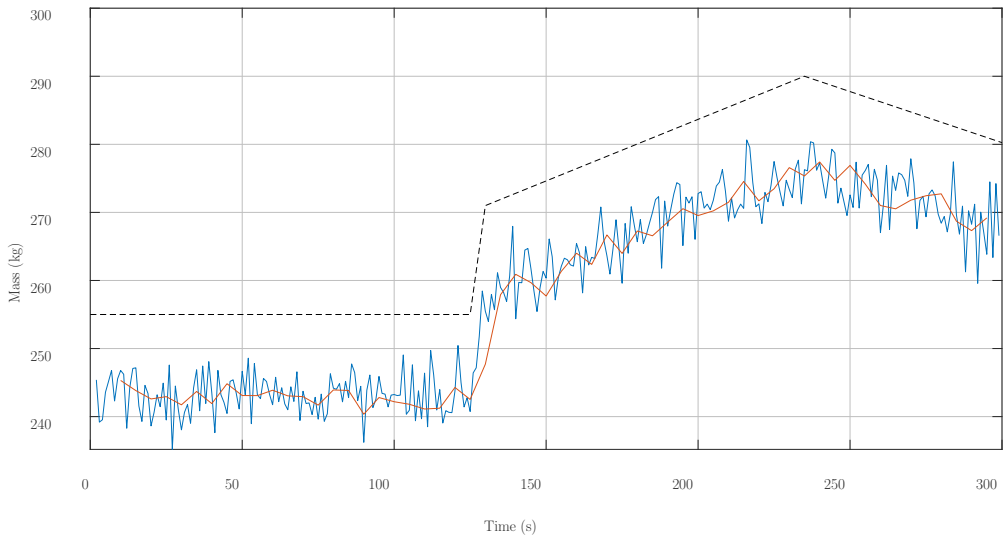


Figure 4.1: Linear Least Squares estimation

The algorithm that is currently being used in the ADM is a frequency domain Linear Least Squares (LLS) algorithm [4, 2]. To show its performance, a simulation has been done that shows all three criteria from the previous chapter: a period of constant mass, a short fast change, and a longer period of slow change. The results are shown in figure (4.1). In this figure, the black dotted line is the real mass in kg over a period of 300 seconds. The results of the estimation done with the LLS algorithm using a window length of one second are given by the red line. The first thing we notice is that the red line lies significantly below the actual mass. The average estimation over the first 125 seconds is 244.4kg, which is almost 11kg lower than the actual mass. Another downside that we see is that the estimation fluctuates a lot, which will increase the error even more. If we simulate 10 times with different random noise terms, these problems are the same in every time. For a window length of one second, the average estimation ranges from 243.3kg to 243.8kg (while the actual mass is 255kg). The variance of the error is very high as well in every simulation, ranging from 7.7 to 9.6. A possible way to solve these problems might be to increase the window length. The blue line in figure shows the results of the LLS algorithm using a window length of five seconds instead. We can see that the fluctuation has decreased, but the bias is still there. Again after repeating this simulation with different random noise series, the bias stays around the same value, but the variance does decrease, in every case. Note that this is consistent with the theory given in [2] which says that the frequency domain LLS estimator is inconsistent in general.

Other frequency domain algorithms are known as well, including Non-linear Least Squares (NLS) and Maximum Likelihood (ML) estimators. While the LLS estimator is a linear estimator and therefore finds a global minimum, the ML and NLS estimators are nonlinear minimizers and so can find local minima if the initial guess of the numerical scheme is not good enough. However, if a global minimum is found, the NLS and ML estimators are consistent. To illustrate this, the NLS algorithm has been applied to the same data as was used for figure (4.1).

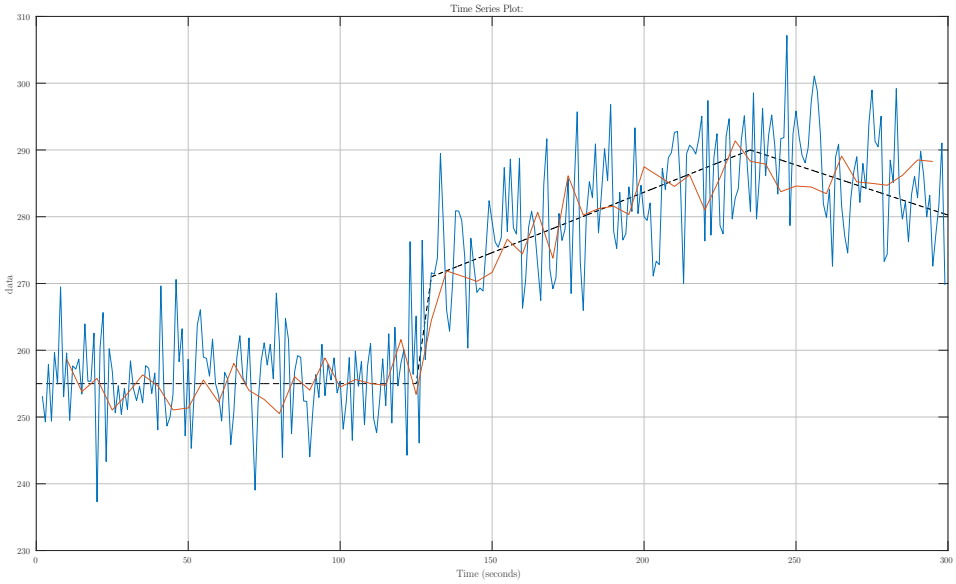


Figure 4.2: Nonlinear Least Squares estimation

Figure (4.2) shows the results of the NLS estimation using the same simulation data as the LLS estimation of figure (4.1). The black dotted line is the actual mass, the blue line is the NLS estimation using a window length of one second and the red line is the NLS estimation using five second windows. We can see that the consistency has improved a lot over the LLS estimation, the average estimation over the first 125 seconds is 255.6kg (the true mass is 255kg), but the fluctuation is still there. In fact, the variance of the NLS estimation over the first 125 seconds (given a window length of one second) is 39.8 while the variance of the LLS estimation over the first 125 seconds is 20.7. We can lower the fluctuation again by increasing the window length, which can be seen in the red plot. The variance of that estimation over the first 125 seconds has been decreased to 7.4 by increasing the window length to five seconds. The variance of the LLS estimation using five second windows is 5.2, so it is still lower than the variance of the NLS estimation.
However, there is a disadvantage of choosing a longer window length. The assumption made in the frequency domain estimators (both in LLS and NLS) is that the system is time-invariant, which means that this assumption is not valid if the mass is changing during a window. Taking a shorter window length has two advantages in that regard. Firstly, a longer window

means that mass fluctuations during the window are more probable and that the changes will have a greater effect. Secondly, any changes in mass during a window can only be seen at the end of the window, so a longer window means that it takes longer for a change to be detected. In the end, a balance needs to be found in the window length, which will depend on the application of the ADM. In applications where it is not important to detect changes quickly, the window length can be made longer to reduce the variance of the estimation and if changes have to be detected quickly, the window length can be shortened.

4.2 Continuous updating

A class of filters used for parameter estimation are extensions on Kalman Filters [3, 6, 9]. Kalman Filters are known to be consistent state estimators if the underlying model is linear, but in the case of non-linear systems, the Extended Kalman Filter used in [3, 6] are known to be inconsistent in general [11], due to the linearization of the model. Despite the fact that the EKF is a non-optimal filter, it can still perform well and is widely used in practice. Alternative Kalman Filters have also been developed for non-linear systems, including the Ensemble Kalman Filter (EnKF) [9] and the Unscented Kalman Filter (UKF) [12, 13]. These are methods that use sampling to find the mean and covariance of state estimations, rather than calculate the (sub)optimal mean and covariance analytically, as the (Extended) Kalman Filter does. These algorithms are less restrictive: the system noises do not need to be Gaussian and they handle non-linearities better than the EKF. This can come at the cost of complexity in the algorithms. The complexity depends on the number of samples needed to calculate a "good" sample mean and covariance.

5 Algorithms

In the previous section, we have derived the model of the ADM with the accompanying assumptions. In this chapter we will discretize the model using forward Euler approximation and derive the Extended Kalman Filter algorithm to estimate the mass m in the system.

5.1 Forward Euler

We are given the following continuous state space model with discrete time measurements:

$$\begin{aligned}\begin{bmatrix}\dot{x}_1(t)-\dot{x}_2(t)\\ \ddot{x}_1(t)-\ddot{x}_2(t)\end{bmatrix} &= \begin{bmatrix}0 & 1\\ -\frac{k}{m} & -\frac{d}{m}\end{bmatrix}\begin{bmatrix}x_1(t)-x_2(t)\\ \dot{x}_1(t)-\dot{x}_2(t)\end{bmatrix} + \begin{bmatrix}0 & 0\\ -1 & \frac{1}{m}\end{bmatrix}\mathbf{u}(t)+\mathbf{w}(t),\\ y(t_n) &= \begin{bmatrix}-\frac{k}{m} & -\frac{d}{m}\end{bmatrix}\begin{bmatrix}x_1(t_n)-x_2(t_n)\\ \dot{x}_1(t_n)-\dot{x}_2(t_n)\end{bmatrix} + \begin{bmatrix}0 & \frac{1}{m}\end{bmatrix}\mathbf{u}(t_n)+v(t_n).\end{aligned} \tag{5.1}$$

All input and output signals are bandpass filtered to eliminate their content below 50Hz and above 200Hz. From figure (2.1) we can see that the eigenfrequency of the system lies between 50Hz and 200Hz, so filtering the signals does not remove the information of the signals around the eigenfrequency.

The Kalman filter used in [3] uses a completely discrete model to estimate the mass at time t_n from a series of estimations done in the past: $\{t_0, t_1, \ldots, t_{n-1}\}$. Therefore it is natural for us to discretize the state space model (5.1).

The first method we will try, is the forward Euler discretization. The forward Euler method is a first order discretization method, which can be useful in practice due to its simplicity, but it can also have some dangers when the state space matrices are ill-posed. Firstly, when using the forward Euler discretization, the stability of the discrete system is not guaranteed. Stability in continuous linear systems is guaranteed if the eigenvalues of A lie in the left half of the complex plane, but stability in the discrete domain is guaranteed if the eigenvalues of the discrete state-space matrix A_d lie within the unit circle. In the case of forward Euler, the discrete state-space matrix is given by $A_d = I + T_s A$, so if λ is an eigenvalue of A, then $\lambda_d = 1 + T_s\lambda$ is an eigenvalue of A_d. Now if λ lies far away from the unit circle, then T_s needs to be very small to push λ_d within the unit circle. It is a well-known fact that the product of the eigenvalues of a matrix is equal to its determinant. In our matrix A the determinant is equal to k/m, so if k is much larger than m, the eigenvalues lie far from the unit circle in the complex plane and T_s needs to be very small. Unfortunately, k is very large in the ADM system (k/m is in the order of 10^5 in our simulations), which means that the system is unstable when using the original sample rate of 4096Hz. In our simulations we will therefore double the original sample rate to 8192Hz, which ensures a stable discrete system.

Another problem one could encounter when using a first-order discretization, is that the discretization error can still be large even when the sample rate is high. For first-order discretizations, the highest order error term will be $T_s^2 A^2$. The largest term (in absolute value) in A^2 is kd/m^2, which can be in the order of 10^7, while T_s^2 is in the order of 10^{-8}, so the error term $T_s^2 A^2$ could still be significant, depending on the parameter values. In fact, because of

the size of k and d compared to m, the value of $\frac{T_s^n A^n}{n!}$ only decreases slowly.

The advantage of a first-order discretizations in the context of Kalman filters is that they maintain the linear character of the original continuous system. The problematic term in the continuous system is $1/m$ in the state-space model, which is manageable as long as m is not too small. When we add higher order terms to the discretization, the non-linearity on the unknown parameters in the discrete system can become more problematic because of the addition of higher powers of $1/m$. So while for the discretization it is better to use higher order terms, it might make the system not suited for Kalman filters. Our starting point will be the forward Euler discretization with a sample rate of 8192Hz. The discrete state space model then becomes:

$$\begin{aligned}\begin{bmatrix} z_1(n+1) \\ z_2(n+1) \end{bmatrix} &= \begin{bmatrix} 1 & T_s \\ -T_s\frac{k}{m} & 1 - T_s\frac{d}{m} \end{bmatrix} \begin{bmatrix} z_1(n) \\ z_2(n) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ -T_s & \frac{T_s}{m} \end{bmatrix} \mathbf{u}(n) + \mathbf{w}(n) \\ y(n) &= \begin{bmatrix} -\frac{k}{m} & -\frac{d}{m} \end{bmatrix} \begin{bmatrix} z_1(n) \\ z_2(n) \end{bmatrix} + \begin{bmatrix} 0 & \frac{1}{m} \end{bmatrix} \mathbf{u}(n) + v(n). \end{aligned} \tag{5.2}$$

The states $z_1(n)$ and $z_2(n)$ approximate the relative position and velocity of the tube at sample n:

$$\mathbf{z}(n) = \begin{bmatrix} z_1(n) \\ z_2(n) \end{bmatrix} \approx \begin{bmatrix} x_1(t_n) - x_2(t_n) \\ \dot{x}_1(t_n) - \dot{x}_2(t_n) \end{bmatrix}. \tag{5.3}$$

Define the following discrete state space matrices:

$$\begin{aligned} A_d(m) &= \begin{bmatrix} 1 & T_s \\ -T_s\frac{k}{m} & 1 - T_s\frac{d}{m} \end{bmatrix}, & B_d(m) &= \begin{bmatrix} 0 & 0 \\ -T_s & \frac{T_s}{m} \end{bmatrix}, \\ C_d(m) &= \begin{bmatrix} -\frac{k}{m} & -\frac{d}{m} \end{bmatrix}, & D_d(m) &= \begin{bmatrix} 0 & \frac{1}{m} \end{bmatrix}. \end{aligned} \tag{5.4}$$

Also define the input vector as

$$\mathbf{u}(n) = \begin{bmatrix} \ddot{x}_2(n) \\ F(n) \end{bmatrix}. \tag{5.5}$$

5.1.1 Noise

To apply a Kalman filter on our system, we need to find the characteristics of the two different stochastic processes: the process noise $\mathbf{w}(t_n)$ and the output noise $v(t_n)$. Specifically, we need to find the covariance matrix $\texttt{cov}(\mathbf{w})$, the variance $\texttt{var}(v)$ and the cross covariance $\texttt{cov}(\mathbf{w}, v)$. This last term is nonzero since the process noise and the output noise both contain the same disturbance force F_d.

The measurement noises $x_{1,err}$ and $x_{2,err}$ are random noise processes with zero mean and a flat power spectral density between 50Hz and 200Hz. The height of the power spectral

density is $1.4 \cdot 10^{-7}$. The variance of the measurement noise can then be calculated as the area under the spectral density curve. This means that, since the spectral density is constant in the bandpass of the filter and zero outside, the variance of the measurement noise after filtering is 150 times the height of the power spectral density function, which is $2.1 \cdot 10^{-5}$. The same thing can be done with the disturbance force F_d. We assume that the disturbance force also has a flat power spectral density between 50Hz and 200Hz. The height of the power spectral density function of the random process F_d is assumed to be 0.12, which means that the variance is 150 times 0.11 which is 16.5.

We defined the process noise $\mathbf{w}$ as a vector of size 2 with each element being the noise term associated with each state. The covariance matrix for the process noise $\mathbf{w}$ is then a 2x2 matrix. Since the first element of $\mathbf{w}$ is zero (there is no noise on the first state), three of the four elements of the covariance matrix are zero too. The only non-zero element of the covariance matrix is the bottom right element, which is the variance of the noise associated with the second state. This consists of the independent random noise processes $\ddot{x}_{2,err}$ and F_d. Since they are independent, we can sum the two variances to get the total variance:

$$\texttt{var}(\ddot{x}_{2,err}) + \texttt{var}(\frac{1}{m}F_d) = 2.1 \cdot 10^{-5} + \frac{16.5}{m^2}. \tag{5.6}$$

The covariance matrix of the process noise is then given by

$$\texttt{cov}(\mathbf{w}) = \begin{bmatrix} 0 & 0 \\ 0 & 2.1 \cdot 10^{-5} + \frac{16.5}{m^2} \end{bmatrix} \tag{5.7}$$

The same thing can be done with the output noise $v(t_n)$. This also consists of the disturbance force $F_d(t_n)$, and measurement noise $\ddot{x}_{1,err}$. Since the measurement noise processes $\ddot{x}_{1,err}$ and $\ddot{x}_{2,err}$ have the same variances, the variance of $v(t_n)$ is the same as the variance of $w(t_n)$.

We can find the cross covariance $\texttt{cov}(\mathbf{w}, v)$ by using the rules for covariances of independent random processes. Again, since there is no noise on the first state equation, the first term of the cross covariance matrix is zero. The second term is

$$\begin{aligned} \texttt{cov}(\frac{1}{m}F_d + \ddot{x}_{2,err}, \frac{1}{m}F_d + \ddot{x}_{1,err}) &= \texttt{cov}(\frac{1}{m}F_d, \frac{1}{m}F_d) \\ &= \frac{16.5}{m^2} \end{aligned} \tag{5.8}$$

5.2 Algorithms

Next we will give a short summary of the Linear Least Squares algorithm, which can also be found in [4], and give a detailed description of the Extended Kalman Filter which we will use for parameter estimation.

5.2.1 Frequency Domain Estimators

We let θ be the vector of our unknown parameters: $\theta = (m, d, k)^T$. The equation of motion (2.1) is converted to the frequency domain via Laplace transform to

$$\begin{aligned} A(s,\theta)X_1(s) &= B_1(s,\theta)F(s) + B_2(s,\theta)X_2(s) + I(s) + F_d(s), \\ A(s,\theta) &= ms^2 + ds + k, \\ B_1(s) &= s^2, \\ B_2(s,\theta) &= ds + k, \\ I(s) &= i_1 s + i_0, \end{aligned} \tag{5.9}$$

where $I(s)$ is a first degree polynomial which is dependent on the initial conditions of x_2 and $\dot{x}_2$. $F_d(s)$ is the Laplace transform of the disturbance force $F_d(t)$. Equation (5.9) is valid for all complex values of s, so specifically for all values $s_f = 2\pi i f_s f/N$ (for sample rate f_s, frequency f in Hz and the number of measured samples N). On these frequencies, the Laplace transform can be approximated by the Discrete Fourier Transform (DFT):

$$\hat{x}(f) = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} x(nT_s) e^{\frac{2\pi i n f}{N}} \approx X(s_f). \tag{5.10}$$

The reason for not using Laplace transforms is that we can only calculate Laplace transforms of continuous signals. Since our measurements of $\ddot{x}_1$ and $\ddot{x}_2$ are done in discrete time, we cannot calculate the Laplace transforms and have to substitute the DFT instead. Using DFTs instead of Laplace transforms keeps equation (5.9) approximately valid at the frequencies s_f except for an aliasing error.
In section 2 we assumed that the input $F(t)$ is exactly known and that $\ddot{x}_2(t)$ is known apart from a measurement error, which means that in the frequency domain we write $X_2(s) = X_{2,meas} - X_{2,err}$ similar to the time domain. Lastly we do not know $x_1(t)$ directly, however we do know from the output equation (2.2) that $y(t_n) = \ddot{x}_1(t_n) + v(t_n)$. Converting the output equation to the frequency domain using DFTs results in $\hat{y}(f) = s_f^2 \hat{x}_1(f) + I_2(s_f) + \hat{v}(f)$, where $I_2(s_f)$ is a second degree polynomial that contains the initial conditions of $x_1(t)$ and $\dot{x}_1(t)$ and $\hat{v}(f)$ is the DFT of the noise v. Combining these together with equation (5.9) gives us the following equation at the frequencies s_f:

$$\begin{aligned} A(s_f,\theta)\left(\frac{\hat{y}(f)}{s_f^2} - \frac{I_2(s_f)}{s_f^2} - \frac{\hat{v}(f)}{s_f^2}\right) &= B_1(s_f)\hat{F}(f) + B_2(s_f,\theta)\hat{x}_{2,meas}(f) + \\ &\quad + I(s_f) - B_2(s_f,\theta)\hat{x}_{2,err}(f) + \hat{F}_d(f) \end{aligned} \tag{5.11}$$

The unknown terms in this equation are the noise terms $\hat{v}$, $\hat{x}_{2,err}$ and $\hat{F}_d$. The known DFTs are the input terms $\hat{F}$, $\hat{x}_{2,meas}$ and the output $\hat{y}$. The polynomials A, B_1, B_2, I and I_2 are also known (see equation (5.9)). The Linear Least Squares algorithm minimizes the quadratic cost function

$$V_{LLS}(\theta) = \sum_{f=50}^{200} \mid A(s_f,\theta)\frac{\hat{y}(f)}{s_f^2} - A(s_f,\theta)\frac{I_2(s_f)}{s_f^2} - B_1(s_f)\hat{F}(f) - B_2(s_f,\theta)\hat{x}_{2,meas}(f) - I(s_f) \mid^2. \tag{5.12}$$

A unique closed form expression exists for the minimum of the LLS cost function .

The Nonlinear Least Squares algorithm attempts to minimize the following cost function:

$$V_{NLS}(\theta) = \sum_{f=50}^{200} \mid \frac{\hat{y}(f)}{F(f)} - \left(\frac{I_2(s_f)}{\hat{F}(f)} + \frac{s_f^2 B_1(s_f)}{A(s_f,\theta)} + \frac{s_f^2 B_2(s_f,\theta)}{A(s_f,\theta)} \frac{\hat{x}_{2,meas}(f)}{\hat{F}(f)} + \frac{s_f^2 I(s_f)}{A(s_f,\theta)\hat{F}(f)} \right) \mid^2. \tag{5.13}$$

Unlike the LLS, the error function is nonlinear in the parameters, so a numerical method (Gauss-Newton scheme) is needed to compute the minimum. To guarantee that the global minimum is found, a good starting value for the parameters need to be used. The starting value will be found by using another algorithm (IWLLS, for details on that algorithm, see [4]).

Lastly, the Maximum Likelihood estimator minimizes the cost function

$$V_{ML}(\theta) = \sum_{f=50}^{200} \frac{\mid A(s_f,\theta)\frac{\hat{y}(f)}{s_f^2} - A(s_f,\theta)\frac{I_2(s_f)}{s_f^2} - B_1(s_f)\hat{F}(f) - B_2(s_f,\theta)\hat{x}_{2,meas}(f) - I(s_f) \mid^2}{\sigma_e^2(f)}. \tag{5.14}$$

The ML estimator uses a similar cost function as the LLS estimator, except that it is weighted by the variance of the equation error used in the LLS where the measurements $X_1(f)$, $X_2(f)$ and $F(f)$ are replaced by the noise of the measurements:

$$\sigma_e^2(f) = \texttt{var}\left(A(s_f,\theta)\frac{\hat{v}(f)}{s_f^2} - B_2(s_f,\theta)\hat{x}_{2,err}(f) + \hat{F}_d(f) \right) \tag{5.15}$$

This means that noisy measurements are weighted less than "clean" measurements.

5.2.2 Kalman filter

Kalman filters are generally known as state estimators for discrete-time linear systems. When we have a (noisy) linear system with noisy measurements $y(n_T s)$, Kalman filters can give an optimal estimation of the state x at time nT_s by using the set of previous estimations and the current measurement. The estimation is usually stated as two distinct steps: the prediction and the update step. However, the two steps can be combined into one step, which we will do. The prediction of the state at the new time is made based on the estimation of the state at the previous step. A residual error is then calculated, which is the difference between the actual measurement $y(nT_s)$ and the predicted measurement based on the estimation of the

state $x(nT_s)$. The residual error is multiplied by a factor K (called the Kalman gain), which is based on the certainty of the measurements and the prediction: the higher the uncertainty of the measurements, the lower K will be, and the higher the uncertainty of the prediction, the higher K will be. Finally the new estimation is computed by adding the predicted state to the residual error times K. This means that the higher K is, the more weight the Kalman filter will give to the measurements, and vice versa. In the case where all the noise terms in the system are Gaussian random processes, the Kalman gain is optimal in the sense that the state estimation minimizes the mean-square error.

To use a Kalman filter to estimate the mass in the discrete system (5.2), we extend the model by adding a state for m and add the difference equation $m(n+1) = m(n)+w_m(n)$. The term $w_m(n)$ is an artificial noise term added to account for mass changes in time. Adding the parameter m to the state space means that the system is no longer linear and the non-linear system becomes

$$\begin{aligned} \begin{bmatrix} \mathbf{z}(n+1) \\ m(n+1) \end{bmatrix} &= \mathbf{f}(\mathbf{z}(n), m(n), \mathbf{u}(n)) + \begin{bmatrix} \mathbf{w}(n) \\ w_m(n) \end{bmatrix}, \\ y(n) &= h(\mathbf{z}(n), m(n)) + v(n), \end{aligned} \tag{5.16}$$

where the functions f and h are given by

$$\mathbf{f}(\mathbf{z}(n), m(n), \mathbf{u}(n)) = \begin{bmatrix} A_d(m(n))\mathbf{z}(n) + B_d(m(n))\mathbf{u}(n) \\ m(n) \end{bmatrix}, \tag{5.17}$$

$$h(\mathbf{z}(n), m(n)) = C_d(m(n))\mathbf{z}(n) + D_d(m(n))\mathbf{u}(n) \tag{5.18}$$

where the matrices A_d, B_d, C_d and D_d are defined in (5.4), which means that they are the first-order approximations of the continuous system. The noise terms $\mathbf{w}$ and v are defined in (2.5) and have the following covariances:

$$Q_w(m) = \mathtt{cov}(\mathbf{w}), \tag{5.19}$$

$$Q_v(m) = \mathtt{cov}(v), \tag{5.20}$$

$$Q_c(m) = \mathtt{cov}(\mathbf{w}, v). \tag{5.21}$$

The original Kalman filter is based on linear systems. However, the Extended Kalman Filter is a version of the Kalman filter which can be applied to non-linear models. In the Extended Kalman Filter, the original matrices used to compute the Kalman gain are replaced by the Jacobian matrices associated with the non-linear state-space functions. This means that the Kalman gain computed in the Extended Kalman Filter are no longer optimal, but it can still be used in practice.

For the Extended Kalman Filter, we need to linearize the functions f and h. For this we need the following Jacobian matrices:

$$M(\mathbf{z}, m, \mathbf{u}) = \frac{\partial}{\partial m}(A_d(m)\mathbf{z} + B_d(m)\mathbf{u}), \tag{5.22}$$

$$N(\mathbf{z}, m, \mathbf{u}) = \frac{\partial}{\partial m}(C_d(m)\mathbf{z} + D_d(m)\mathbf{u}). \tag{5.23}$$

Before stating the Kalman filter, we introduce a few notations. Let

$$\hat{z}(n \mid p) = \mathbb{E}\left[z(n) \mid y(1), \ldots, y(p))\right]$$

be the state estimation of $z(n)$ given the measurements $y(1), \ldots, y(p)$ and

$$\hat{m}(n) = \mathbb{E}\left[m(n) \mid y(1), \ldots, y(p))\right]$$

be the estimate of m given the measurements $y(1), \ldots, y(p)$. Furthermore, let the estimation covariance matrices be given by $P_1(n) = \mathtt{cov}(\hat{\mathbf{z}}(n \mid n) - \mathbf{z}(n))$, $P_2(n) = \mathtt{cov}(\hat{\mathbf{z}}(n \mid n) - \mathbf{z}(n), \hat{m}(n) - m)$ and $P_3(n) = \mathtt{cov}(\hat{m}(n) - m(n))$.

For brevity, the matrices used in the Kalman filter equations will be abbreviated as follows:

$$\begin{aligned}
M_n &= M(\hat{z}(n \mid n), \hat{m}(n), \mathbf{u}(n)),\\
N_n &= N(\hat{z}(n \mid n), \hat{m}(n), \mathbf{u}(n)),\\
A_n &= A_d(\hat{m}(n)),\\
B_n &= B_d(\hat{m}(n)),\\
C_n &= C_d(\hat{m}(n)),\\
D_n &= D_d(\hat{m}(n)),\\
Q_{w,n} &= Q_w(\hat{m}(n),\\
Q_{v,n} &= Q_v(\hat{m}(n),\\
Q_{c,n} &= Q_c(\hat{m}(n),
\end{aligned}$$

The Extended Kalman Filter minimizes

$$e(n+1) = \mathbb{E}\left[||\mathbf{z}(n+1) - \hat{\mathbf{z}}(n+1)||^2 + ||m(n+1) - \hat{m}(n+1)||^2\right]$$

as described in [3], as follows. The predicted state estimate, given the previous estimation is given by

$$\hat{z}(n+1 \mid n) = A_n \hat{z}(n \mid n) + B_n u(n). \tag{5.24}$$

The residual error of the previous estimation is given by

$$y_{res}(n) = y(n) - C_n \hat{z}(n \mid n) - D_n u(n). \tag{5.25}$$

The innovation matrix $S(n)$ is defined as

$$S(n) = C_n P_1(n) C_n^T + C_n P_2(n) N_n^T + N_n P_2^T(n) C_n^T + N_n P_3(n) N_n^T + Q_{v,n}. \tag{5.26}$$

The Kalman gain matrices for the state and parameter estimation update ($K(n)$ and $L(n)$ respectively) are given by

$$K(n) = (A_n P_1(n) C_n^T + M_n P_2^T(n) C_n^T + A_n P_2(n) N_n^T + M_n P_3(n) N_n^T + Q_{c,n}) S^{-1}(n), \quad (5.27)$$

$$L(n) = (P_2^T(n) C_n^T + P_3(n) N_n^T) S^{-1}(n). \quad (5.28)$$

Then the new state and parameter estimation is given by

$$\hat{z}(n+1|n+1) = \hat{z}(n+1 \mid n) + K(n) y_{res}(n), \quad (5.29)$$

$$\hat{m}(n+1) = \hat{m}(n) + L(n) y_{res}(n). \quad (5.30)$$

Finally we calculate the new estimation covariances:

$$P_1(n+1) = A_n P_1(n) A_n^T + A_n P_2(n) M_n^T + M_n P_2^T(n) A_n^T + M_n P_3(n) M_n^T - K(n) S(n) K^T(n) + Q_{w,n}, \quad (5.31)$$

$$P_2(n+1) = A_n P_2(n) + M_n P_3(n) - K(n) S(n) L^T(n), \quad (5.32)$$

$$P_3(n+1) = P_3(n) - L(n) S(n) L^T(n) + Q_m. \quad (5.33)$$

5.3 Initialization of the Kalman Filter

The Kalman Filter, as do any kind of recursive estimation algorithm, require the user to give certain initial parameters. In the case of the EKF, the state estimations have to be initialized. The states in this case are z_1 and z_2 (which are the position and velocity of the inner tube relative to the outer tube), and the mass m. Since we have no information about the position and velocity at any time, we set those parameters at zero at the start. However, we cannot set the mass at zero because of the factor $1/m$ in our model. In practice, we can initialize the mass estimation by using the LLS estimator, which will give us an estimation that is close enough, or we can use the mass of an empty tube, so we do not have to initialize at zero. In the case of the ADM350 this mass is 220kg, which we will use in our simulations.

We also need to initialize the covariances of the estimations, namely $P_1(0)$, $P_2(0)$ and $P_3(0)$. We have no prior information on the initial value of our states, so we also have no information about the covariance of the initial state estimation, so we will set $P_1(0) = 100I$ and $P_2(0) = 0$. We do have to take some care in choosing the initial covariance on the mass $P_3(0)$: if we choose it too high, it is possible that the estimation will initially overshoot and it might end up too close to a local minimum. Especially if the mass is closer to zero (i.e. a smaller ADM), the estimation will be more sensitive to the initial conditions. In our simulation, we will choose $P_3(0) = 70$, but for the real data we will look at some other values.

6 Simulations

In this section we will test the Extended Kalman Filter (EKF) as a parameter estimation algorithm in the ADM by simulating the model in Simulink and applying the EKF the results of the simulation. Firstly we will let the mass be constant to see if the EKF estimation has a bias (or any other uncertainty) in the case of constant mass. After that we will let the mass change both slowly and faster to see how fast the EKF can adapt to changes in mass.

The simulation is done with the other paramaters being: $d = 1.13 \cdot 10^{-4}$ and $k = 5 \cdot 10^{-7}$. We will focus on the mass estimation foremost, so we assume that d and k are known.

6.1 Bias investigation

We have seen in Section 4 (figure (4.1) that the LLS estimation has a significant bias. We want to see if the Extended Kalman Filter (EKF) estimation is biased as well. For this we will apply the EKF to a system with a constant mass. For this situation, the error w_m from equation (5.16) has been removed, since we know that the mass is constant. We can test the EKF both with and without the disturbance force F_d. An example of the estimation without the disturbance force F_d is given in figure (6.1). The black dotted line is the actual mass (255kg) and the blue graph is the estimation done with the EKF. We notice that there is a bias of around 1.2kg in the estimation. This is much smaller than the bias in the LLS estimator, which had a bias of around 11kg. This means that we can already do much better with the Kalman filter estimation than with the LLS. If we repeat the simulation multiple times with different random noise terms, we see that the bias ranges from 1kg to 1.3kg. We can see if we can improve this estimation however. To do this, we first need to find out where this bias comes from. There are three possible explanations for this bias: an error due to the discretisation of the continuous system, an error due to the linearisation in the Extended Kalman Filter or it is an effect of the filtering of the input and output signals.

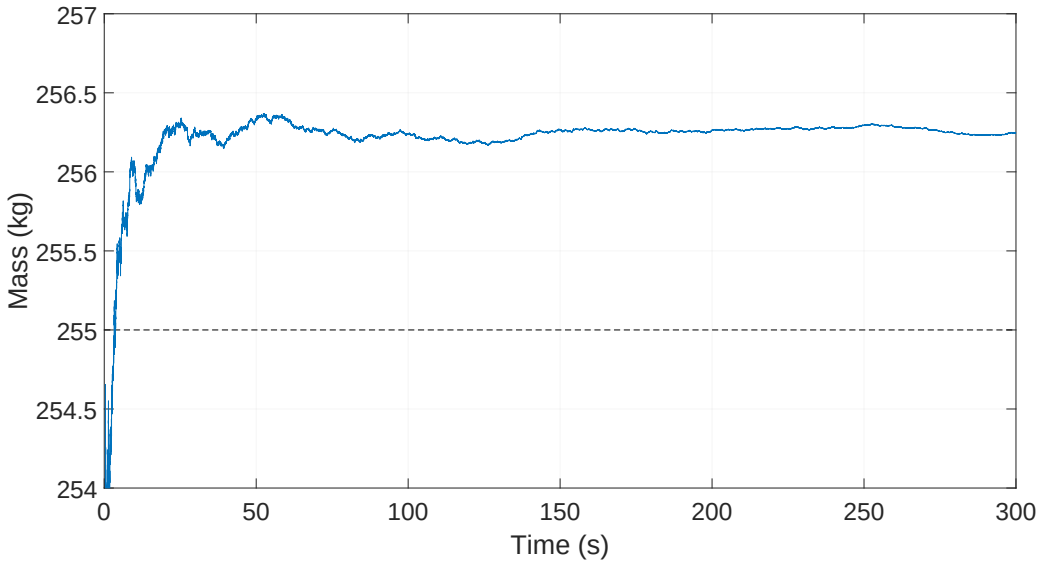


Figure 6.1: Kalman filter mass estimation for constant mass

The first thing we can look at is the discretization error. We have seen with the derivation of the forward Euler discretization that in the ADM model, the state-space matrix A is so ill-posed, that the error in a first-order discretization can be significant, even with a sample rate of 8kHz. This leads us to believe that the discretization error is the most significant factor in the bias that we have.
Both the discretisation error and the linearisation error should decrease when the sample rate of the measurements are increased. To see if this works in our model, we will examine the case where no filtering is used and test the EKF for increasing sample rates. Figure (6.2) gives the result of this simulation. The black dotted line is the exact mass again, the blue line is the estimation for 8kHz, the red line using 16kHz and the yellow line using 32kHz. All three simulations use different random noise terms. The lowering of the bias is consistent when we repeat the simulation using different random noise. We can see that the bias indeed decreases when we increase the sample rate.

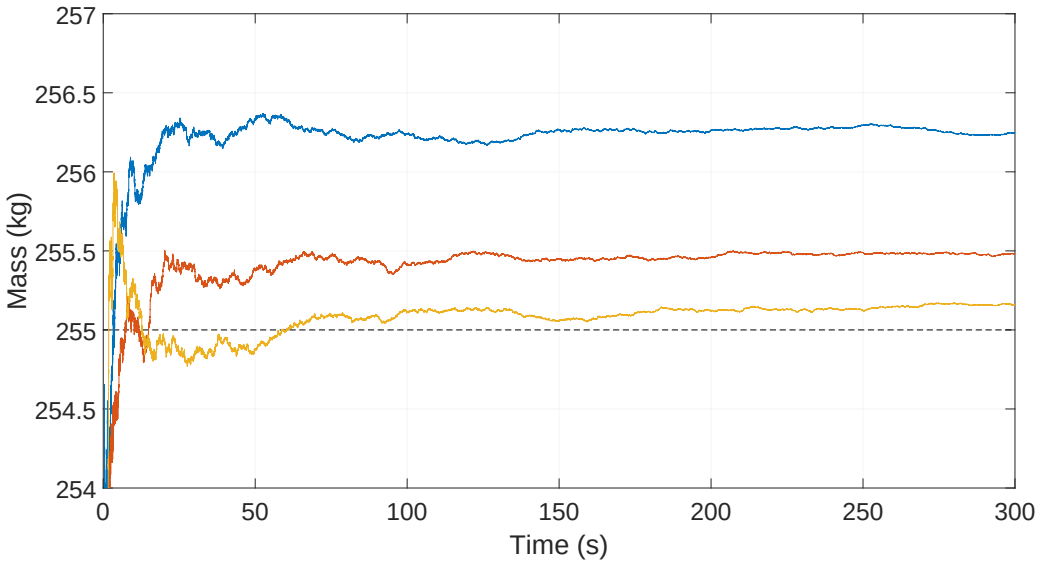


Figure 6.2: Kalman filter mass estimation for constant mass with variable sample rates

This means that we can increase the sample rate to obtain a better estimation with respect to the bias. However, in practice the sample rate is limited by hardware so it might not be possible to increase the sample rate.

An even more certain way to see if the bias indeed comes from the discretization error, is to replace the continuous model of the ADM with the discrete model given by the discrete state-space matrices in (5.2). Figure (6.3) shows an estimation of the mass in a simulation where the continuous system is replaced by the discrete system. We can see that the bias has almost completely been removed (the bias is a few grams), so we can conclude that discretization is the most imnportant factor of the bias as seen in figure (6.2).

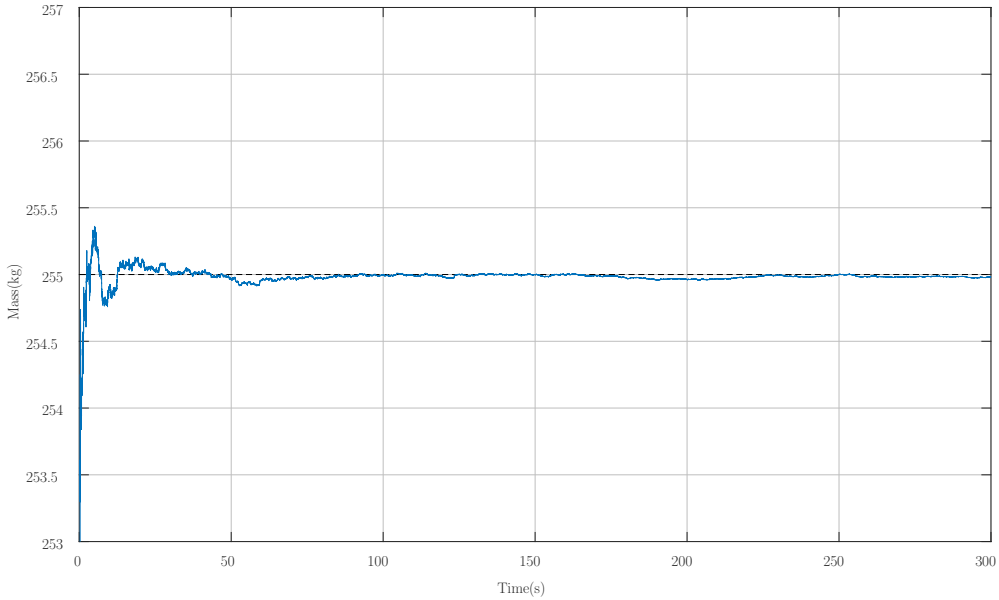


Figure 6.3: Kalman filter mass estimation of a discrete system for constant mass

Since increasing the sample rate can be difficult in practice due to limitations of hardware, we can try to find another way to decrease the discretization error. A way to do this, is to use the exact solution to the differential equation $\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + B\mathbf{u}(t) + Ew(t)$ to find an exact expression for $x(t_{n+1})$ based on $x(t_n)$, instead of an approximation, which is done with forward Euler.

$$\mathbf{x}(t_{n+1}) = e^{A(t_{n+1}-t_n)}\mathbf{x}(t_n) + \int_{t_n}^{t_{n+1}} e^{A(t_{n+1}-\tau)}B\mathbf{u}(\tau)d\tau + \int_{t_n}^{t_{n+1}} e^{A(t_{n+1}-\tau)}Ew(\tau)d\tau. \quad (6.1)$$

Due to the low-pass filtering, we can assume that the input $\mathbf{u}$ and the noise $w(t)$ are constant in the period between t_n and t_{n+1}, since the cut-off frequency of 200Hz lies significantly below the sample frequency of 4096Hz, so we can take $\mathbf{u}(t_n)$ outside the integral and the discrete system with $\mathbf{z}(n) = \mathbf{x}(nT_s)$ and $\mathbf{u}(n) = \mathbf{u}(nT_s)$ becomes:

$$\begin{aligned} \dot{\mathbf{z}}(n+1) &= A_d\mathbf{z}(n) + B_d\mathbf{u}(n) + E_d w(n), \\ y(n) &= C\mathbf{z}(n) + D\mathbf{u}(n) + Ew(n) + v(n). \end{aligned} \quad (6.2)$$

$$
\begin{aligned}
A_d &= e^{A(t_{n+1}-t_n)} = e^{AT_s}, \qquad (6.3)\\
B_d &= \int_{t_n}^{t_{n+1}} e^{A(t_{n+1}-\tau)} d\tau B \\
&= \int_0^{T_s} e^{A\tau} d\tau B \\
&= \left[A^{-1} e^{A\tau}\right]_0^{T_s} B \\
&= A^{-1}\left(e^{AT_s} - I\right) B \\
&= A^{-1}\left(A_d - I\right) B, \\
E_d &= \int_{t_n}^{t_{n+1}} e^{A(t_{n+1}-\tau)} d\tau E \\
&= A^{-1}\left(A_d - I\right) E. \qquad (6.4)
\end{aligned}
$$

Since the matrix A is a 2x2 matrix, it is fairly easy to calculate e^{AT_s} using the eigenvalues of the matrix A. We do not have to worry about instability in this case. If the continuous system is stable, the real part of the eigenvalues of A lie in the left half of the complex plane. If λ is an eigenvalue of A, then $e^{\lambda T_s}$ is an eigenvalue of e^{AT_s}, and since the real part of λ is negative $e^{\lambda T_s}$ lies within the unit circle, which means the discrete system is also stable.

To implement the exact discretization in the Extended Kalman Filter, we need to know the derivative of the state space matrices. We know the derivatives of the matrices A and B with respect to m:

$$
\frac{\partial A}{\partial m} = \begin{bmatrix} 0 & 0 \\ \frac{k}{m^2} & \frac{d}{m^2} \end{bmatrix}, \qquad \frac{\partial B}{\partial m} = \begin{bmatrix} 0 & 0 \\ 0 & -\frac{1}{m^2} \end{bmatrix} \qquad (6.5)
$$

From this we can also derive exact formulas for the Jacobians M and N. To compute the derivative of A_d, we need the derivative of a matrix exponential. The exact formula for this is given in [14]:

$$
\begin{aligned}
\frac{\partial}{\partial m}\left(A_d(m)\right) &= \frac{\partial}{\partial m}\left(e^{A(m)T_s}\right) \\
&= \int_0^1 e^{A(m)T_s(1-\tau)} T_s \frac{\partial A}{\partial m} e^{A(m)T_s\tau} d\tau. \qquad (6.6)
\end{aligned}
$$

In practice, computing this integral at each step of the estimation is not feasible. We could, however, approximate the matrix exponential using the first few terms of the power series and differentiate terms-wise. The approximation of the matrix exponent using the first N terms is

$$
e^{A(m)T_s} \approx I + \sum_{n=1}^{N-1} \frac{T_s^n}{n!} A^n(m). \qquad (6.7)
$$

The derivative of the n-th power (for $n \geq 1$) of the matrix A can be found using the product rule:

$$\frac{\partial}{\partial m}\left(A^n\right) = \sum_{i=0}^{n-1} A^i \frac{\partial A}{\partial m} A^{n-i-1}. \tag{6.8}$$

Combining these gives us an approximation for the derivative of the discretization matrix A_d:

$$\frac{\partial A_d}{\partial m} = \frac{\partial}{\partial m}\left(e^{AT_s}\right) \approx \sum_{n=1}^{N-1} \frac{T_s^n}{n!} \sum_{i=0}^{n-1} A^i \frac{\partial A}{\partial m} A^{n-i} \tag{6.9}$$

In the further text, we will always use the approximation of the exponential matrix to compute the derivative of $A_d(m)$. We find that using $N = 6$ might be sufficient to ensure a good approximation, since the biggest element of the next term of the derivative of $A_d(m)$ is around $4 \cdot 10^{-9}$.

For the derivative of B, we need the derivative of the inverse of a matrix. This can easily be found by using the definition of the inverse: $AA^{-1} = I$ and differentiating both sides:

$$\frac{\partial}{\partial m}\left(AA^{-1}\right) = \frac{\partial A}{\partial m}A^{-1} + A\frac{\partial A^{-1}}{\partial m} = \mathbf{0} \Rightarrow \frac{\partial A^{-1}}{\partial m} = -A^{-1}\frac{\partial A}{\partial m}A. \tag{6.10}$$

Using this identity and the product rule, we can also find an expression for the derivative of B_d:

$$\begin{aligned}
\frac{\partial}{\partial m}\left(B_d(m)\right) &= \frac{\partial}{\partial m}\left(A^{-1}(m)\left(A_d(m) - I\right)B\right) \\
&= \frac{\partial A^{-1}}{\partial m}\left(A_d(m) - I\right)B(m) + A^{-1}\frac{\partial A_d}{\partial m}B(m) + A^{-1}(m)\left(A_d(m) - I\right)\frac{\partial B}{\partial m} \\
&= -A^{-1}(m)\frac{\partial A}{\partial m}A^{-1}(m)\left(A_d(m) - I\right)B(m) + A^{-1}\frac{\partial A_d}{\partial m}B(m) + \\
&\qquad + A^{-1}(m)\left(A_d(m) - I\right)\frac{\partial B}{\partial m}
\end{aligned} \tag{6.11}$$

This means that the Jacobian matrix M can be written as

$$M = \frac{\partial A_d}{\partial m}\mathbf{z} + \frac{\partial B_d}{\partial m}\mathbf{u}. \tag{6.12}$$

We can replace the discretization matrices A_d, B_d, C_d, D_d, and the Jacobian M with these matrices in the Kalman Filter equations and run the simulation again. Since calculating the intergral (6.6) at every time step is quite slow, we will use the series approximation instead. We will approximate it by (6.9) using $N = 6$. A typical resulting estimation is given in figure (6.4). We can see that there is still an error in the estimation. However, when we repeat the simulation using different random noise vectors, we notice that it is not a systemic bias.

From 30 simulations, we see that the mean error ranges from -0.2kg to +0.2kg with an average error of -0.0046kg. A possible explanation for the error is that the algorithm does not have enough time to reach the correct equilibrium. Since we know that the mass is constant in this simulation, we have chosen to exclude the noise term w_m from equation (5.16). This means that the variance of the mass estimation will converge to zero (i.e. the estimator becomes more certain of its estimation the more data it has). It can happen that the variance of the estimation decreases "too fast", which will result in the estimation never reaching the correct value or taking too long to reach the correct value. It is certainly possible that the estimation in our simulation will reach the correct value for m if we let the estimation continue longer than 300 seconds. However, since we ultimately want to use our algorithm in cases where the mass is not constant, it is not very interesting to look beyond the 300 seconds to see if the EKF converges at some point.

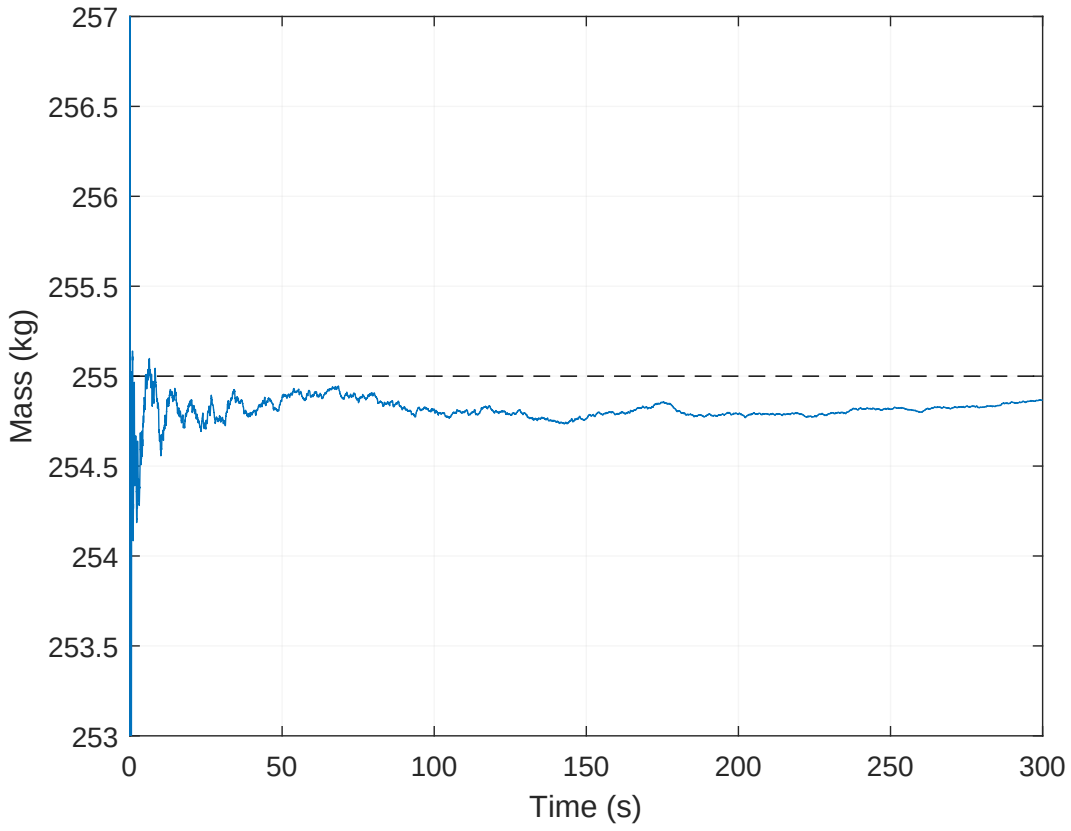


Figure 6.4: Kalman estimation with exact discretization

If it is the case that the variance decreases too quickly for the estimation to reach the correct value for m, then adding the fictitious noise term w_m in the EKF equations will help the estimation converge to the correct value for m. If we set the variance of w_m to non-zero, then the variance of the mass estimation will not converge to zero, which means that the estimation will keep updating, so it won't get stuck in a wrong equilibrium. The disadvantage of creating an artificial noise term is that the estimation will be more noisy (less smooth) as a result. The higher we set the variance of w_m in the EKF, the noisier the estimation will become.

In figure (6.5) we see an estimation where we included a non-zero variance for the noise term ($\texttt{var}(w_m) = 10^{-6}$). We see that the estimation is indeed less smooth than without w_m, which is what we expected. If we repeat this estimation multiple times, then it appears that there is still some uncertainty in the estimation. If we look at the mean error of the estimation during the last 200 seconds, we see that it still ranges from about -0.2kg to +0.2kg, similar to the case without w_m. We can see that the estimation gets pushed back to the real value when it has moved away from it.

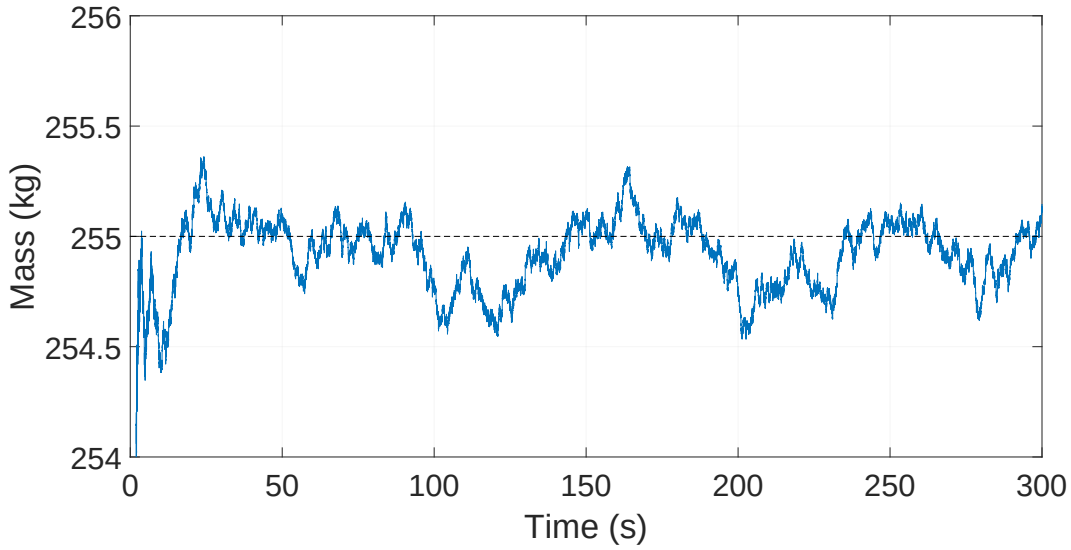


Figure 6.5: Kalman filter mass estimation for constant mass, including mass noise term with variance 10^{-6}

The next thing to look at, is the effect filtering the signals has on our estimation of the mass. Figure (6.6) shows a typical estimation with the EKF when we filter all the input and output signals. The estimation is done without adding the noise w_m to just see the effect of the filtering.

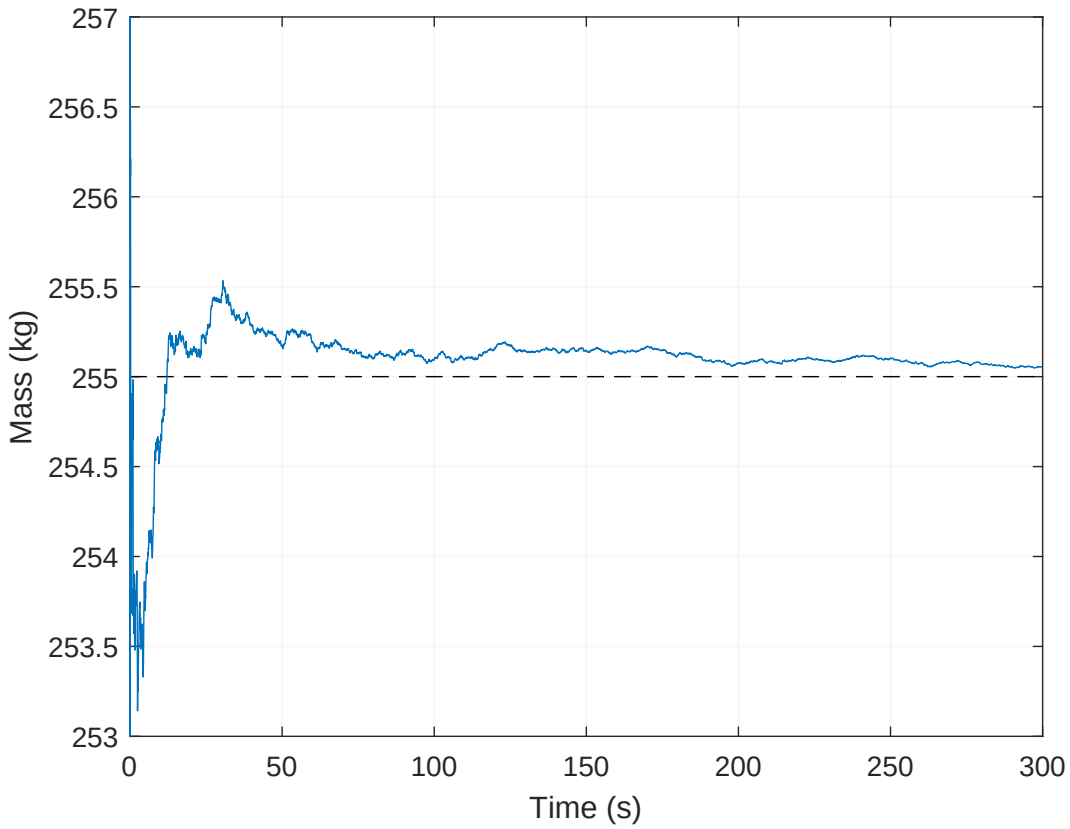


Figure 6.6: Kalman filter mass estimation for constant mass

When we do this simulation several times, we see that the estimations are similar to the situations where we do not filter the signals, i.e. the error in estimation ranges from -0.2kg to +0.2kg. The conclusion is that bandpass filtering does not have any negative effects on the expected results of the EKF estimation.

6.2 Initial mass

For estimation done using Extended Kalman Filters, choosing the initial conditions can be important. Due to the linearization of the model, the EKF is not an optimal filter (unlike the linear Kalman filter), so it can happen that the EKF finds a local minimizer. In the previous simulation the initial mass that has been chosen is 220kg, which was chosen because it is the mass of an empty ADM350 module. We can quickly inspect what happens if we change the initial mass estimate.

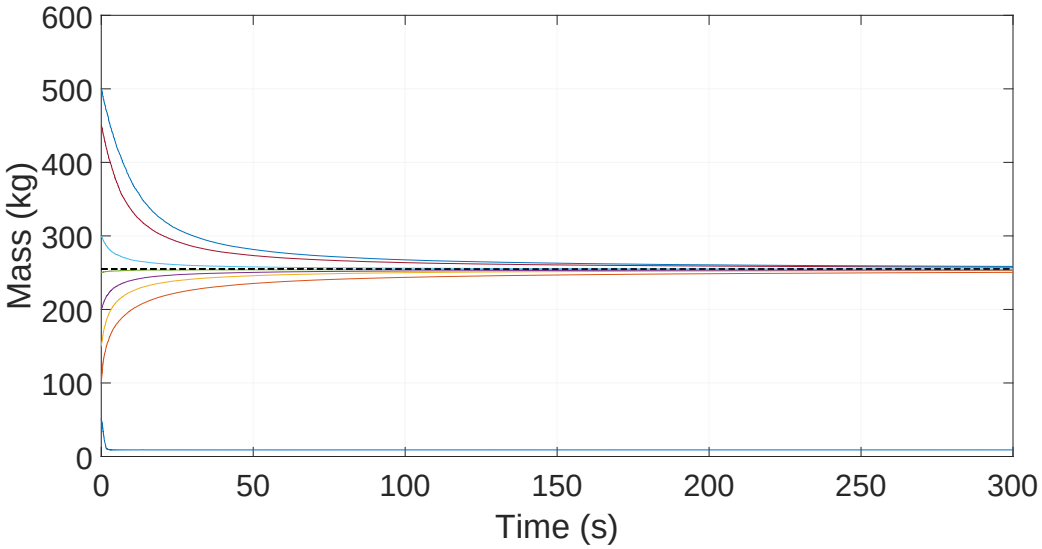


Figure 6.7: Kalman filter estimations using different initial estimations

Figure (6.7) shows a number of estimations done using different initial estimations, from 50kg up to 500kg. We see that there is one estimation which converges to the wrong mass. That estimation starts at 50kg. All estimations that start above 100kg do converge to the correct value for m. So we see that we do have to take some care to choose a good initial value for m. In the case of the ADM350, it is not a big problem, since it only goes wrong below 100kg. In fact, we know that the mass will not go below 220kg, since that is the mass of an empty tube (in the case of the ADM350). However, not all ADM models are the same weight, so if we use a lighter ADM that is below 100kg, there is a possibility that the non-linearity can be a problem. What we can do to get an initial mass that is good enough, is to use another estimation algorithm to choose our initial mass. For this we can use either the LLS or NLS estimation over a period of a few seconds. The LLS algorithm has too much bias to use for regular estimation, but it is good enough to get a decent initial guess for the mass. Another option which we will use in the next section when analysing the real data, is to impose a minimum mass on the EKF.

6.3 Varying mass

The next step is to see how the EKF algorithm adapts to changes in mass. We want to know how quickly the algorithm adapts to sudden changes in mass and also how good the estimator can track slower variations. It is expected that if we do not add the artificial noise w_m to the mass equation, the algorithm cannot adapt to changes in mass. Since there is no noise on the the mass equation, the Kalman gain associated to the mass equation ($L(n)$ in equation 5.28) will converge to zero, or in other words, the algorithm becomes more certain of its estimation the more data it has used. This means that the difference between two consecutive estimations of m will be small. If we add the noise term w_m to the mass equation, however, the Kalman gain $L(n)$ will not converge to zero, so the estimation will put more weight on measurements rather than the previous estimation. This also means that the higher the variance of w_m, the more the estimation will depend on the measurements, so the faster it should adapt to

changes in mass. On the other hand, as we have seen in the previous part, a higher variance on w_m will also mean a greater variance on the estimation, so a balance needs to be found between the estimation variance and adaptability of the algorithm.

The first thing we will look at, is the situation where there is no disturbance force F_d. Figure (6.8) shows the mass estimation if we do not add the artificial noise w_m to the mass equation. We can clearly see that the EKF does not adapt to large changes in mass at all.

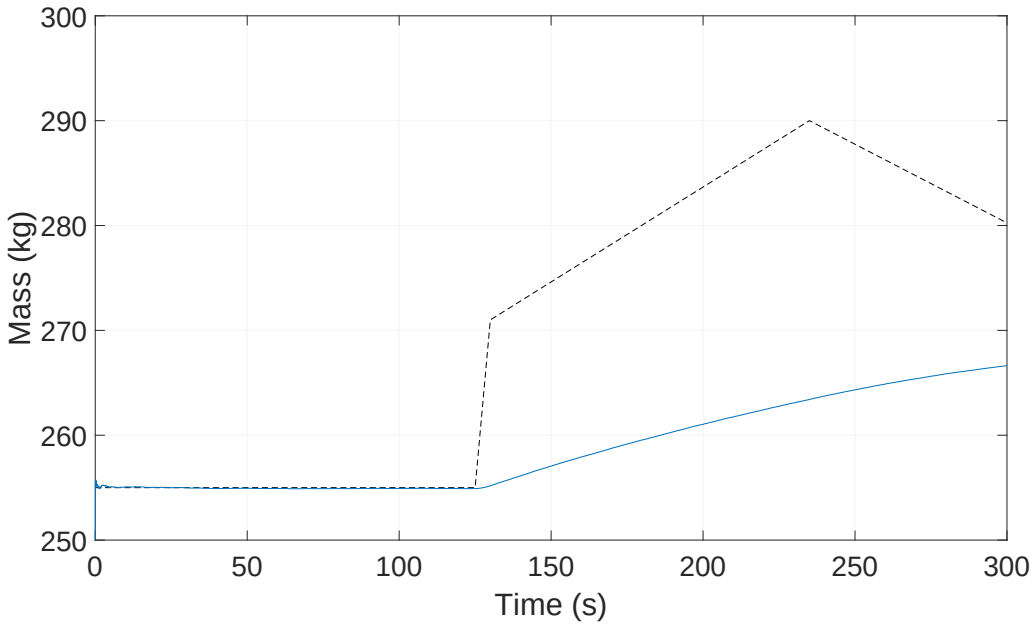


Figure 6.8: Kalman filter mass estimation for varying mass

Once we add the artificial noise w_m to the mass equation, the covariance of the estimation will not converge to zero, so it can adapt to changes in mass. Figure (6.9) shows the estimation using varying values for a covariance of w_m. As we have seen before in the case of constant mass, the higher the variance of w_m, the more variance is in the estimation. On the other hand, we can also see that the higher we set the variance of w_m, the faster the algorithm adapt to high variations in mass. This is because the higher the variance of w_m, the higher the Kalman gain associated with the mass equation will be. The higher the Kalman gain is, the less the estimation trusts the previous estimations and the more the estimation trusts the measurements. And if the mass suddenly changes after a period in which it was constant, then if the algorithm trusts the past estimations more, it will adapt slower to the new value of m.

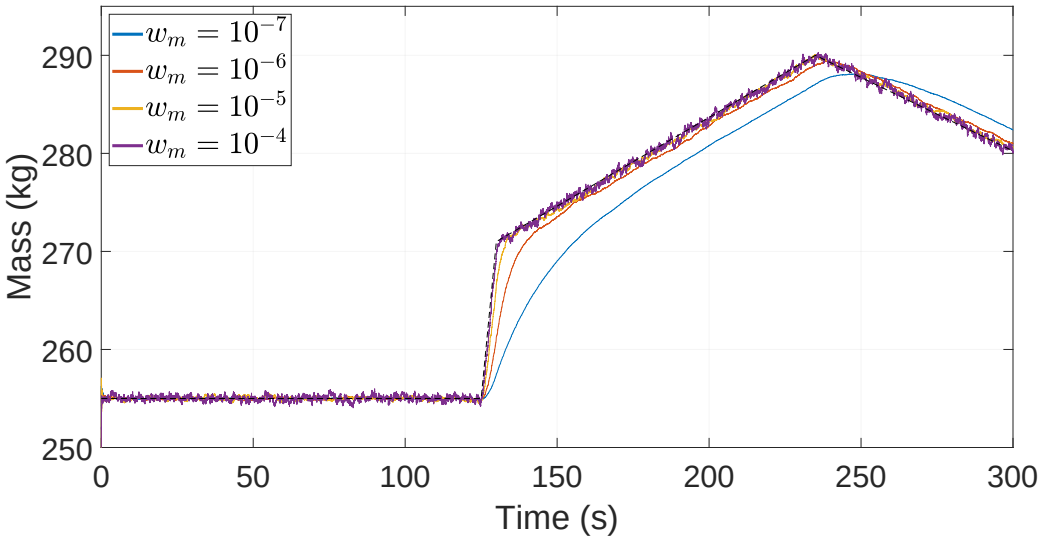
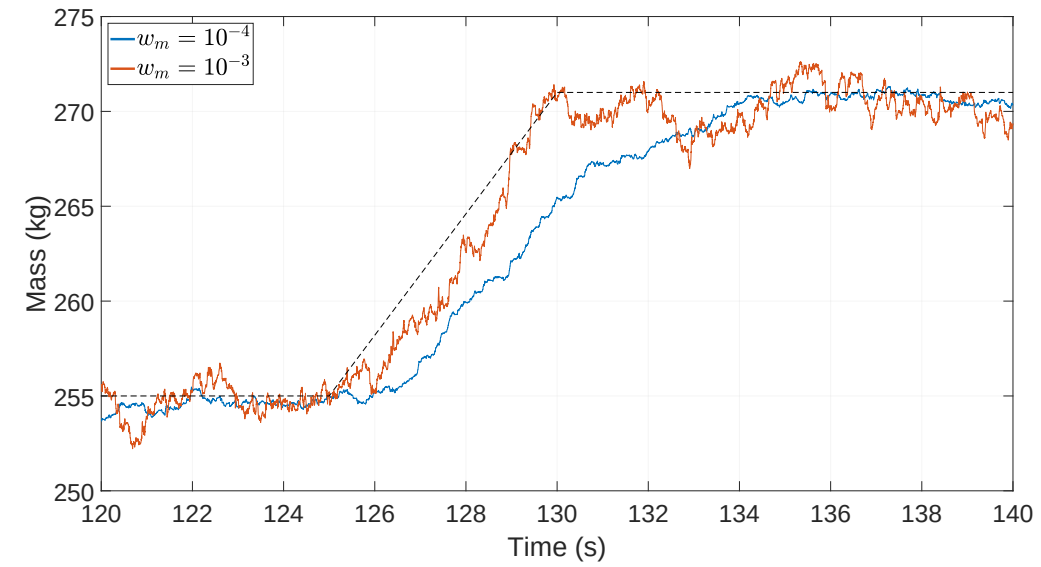


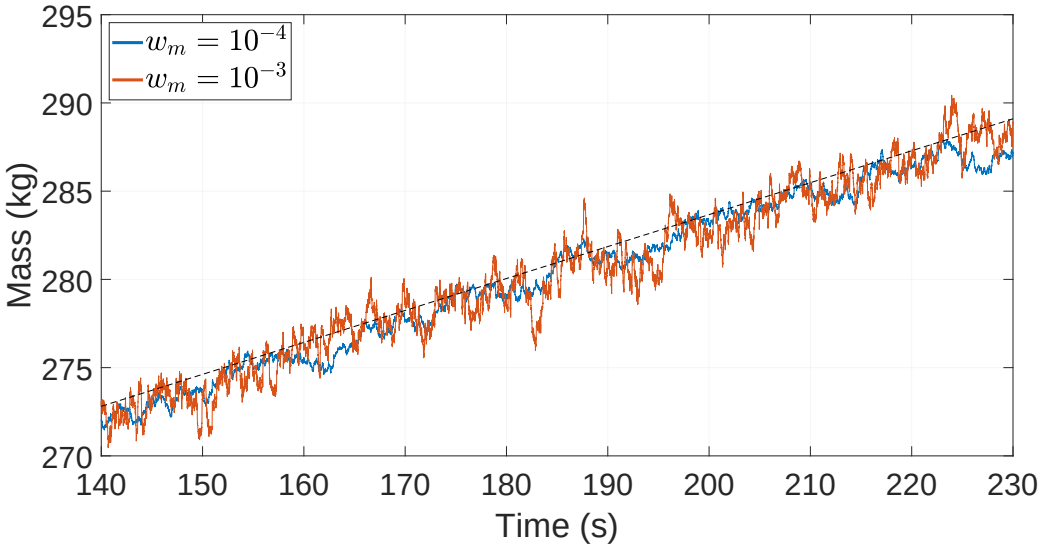
Figure 6.9: Kalman filter mass estimation for varying mass without disturbance force F_d

We can tell from the figure that the blue and red lines are definitely too slow to react to the large changes. Especially in the blue line we can see that it lags behind in the phase after the sudden jump. After the peak at 235 seconds we can see that it keeps climbing even after the actual mass has started decreasing.

Figure (6.10) is the estimation when we add slurry noise, zoomed in to the part where the mass changes. We can see that when we set the variance of the fictitious noise w_m to 10^{-3} (the red line), the estimation follows the changes in mass very quickly. The blue line, which is the estimations where we set the variance of w_m to 10^{-4}, follows the changes in mass more slowly. During the slower changes (seen in the bottom figure), the blue line can still follow the actual mass pretty well. We can also see that faster reaction to mass changes comes at a cost, namely noisier estimation. When we calculate the error of the estimation $(m(t) - \hat{m}(t))$ (during the part where the mass is constant, between 10 and 120 seconds) and compare the variance of the error, we can see it clearly: the variance of the error of the red line is 1.03 and the variance of the error of the blue line is 0.38. For reference, for the frequency domain LLS estimation, the variance of the error is 7.8 with a one second window and 1.24 with a five second window. This means that the Kalman filter estimation is still smoother (has a smaller variance) than the LLS. When we repeat the simulation with the EKF several times, similar result are gotten. The variance of the error ranges from 0.249 to 0.436 when $\texttt{var}(w_m) = 10^{-4}$ and it ranges from 0.860 to 1.088 when $\texttt{var}(w_m) = 10^{-3}$. The variance of the error for the LLS estimation with window length of one second ranges from 7.074 to 11.472, and for the LLS estimation with a window length of 5 seconds it ranges from 1.732 to 3.651. This means that even when we set the variance of the fictitious noise high to account for the changes in mass, the variance of the mass estimation using the EKF is still lower than the variance of the LLS estimation with a five second window. Only when we have a window length of 10 seconds the variance of the error can get below 1kg, but then the reaction time of the algorithm is much slower.



(a)



(b)

Figure 6.10: Kalman filter mass estimation for varying mass with disturbance force F_d

6.4 Note on inputs

In the ADM model, there are two inputs that influence the movement of the inner tube: the actuator force F and the acceleration of the outer tube $\ddot{x}_2$. Of these three, we can measure the actuator force and the outer tube acceleration, which means they can be used as (measurable) inputs of the model.

We can choose the input of F, the force with which we shake the inner tube. The other input, the outer tube acceleration $\ddot{x}_2$ cannot be chosen, because it is dependent on the envi-

ronment in which the ADM is used. For example, if there are machines working on the same floor that the ADM is attached to, the vibrations of the machines can shake the outer tube of the ADM as well. Of course, in our simulation we can choose what $\ddot{x}_2$ will be. Until now, we have chosen it to be a random sequence drawn from a zero-mean normal distribution, as specified in [4].

We can inspect what happens to the estimation when we vary the outer tube movement, or turn it off completely. Figure (6.11) shows an example of this. The blue line is the estimate when there is no outer tube shaking (i.e. $\ddot{x}_2 = 0$ for all time), the red line is the estimation when there is the " normal" shaking of the outer tube (i.e. the intensity of $\ddot{x}_2$ is as specified in [4]) and the yellow line is the estimation when the shaking is very intense, and the dotted line is the real mass. Assuming we keep all the other variables the same, a more intense shaking of the outer tube seems beneficial to the estimation. This is because with larger values for $\ddot{x}_2$ and the same noise variances, the estimation covariance P_3 will go down, i.e. the Kalman Filter will be more certain that the estimation is correct. This results in lower noisiness and in a better tracking of changes in mass. When the estimation covariance is lower, the EKF will correctly assume that any residual error comes from a change of the mass instead of a random error from any of the noise terms.

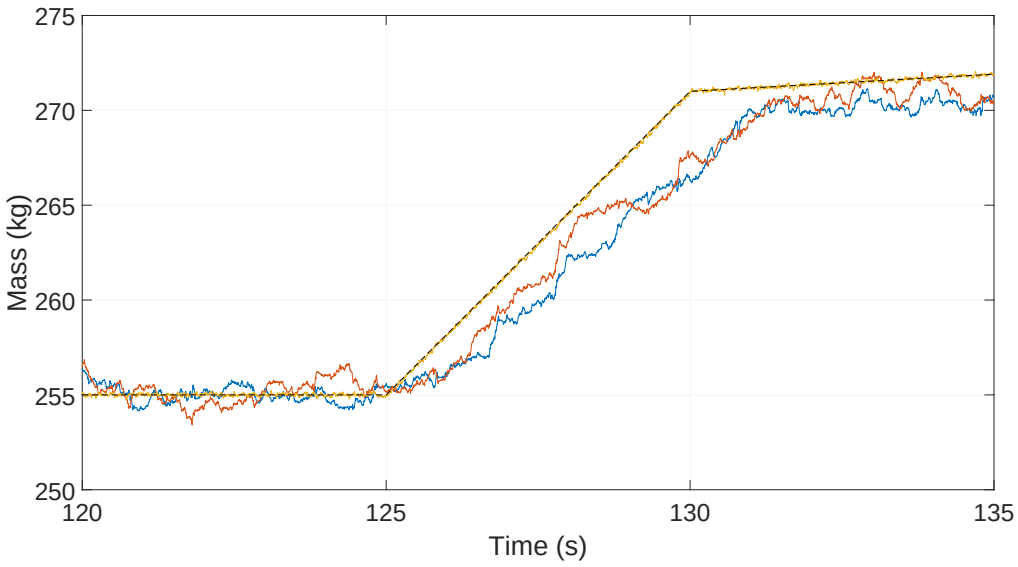


Figure 6.11: Kalman filter estimates with varying intensity of $\ddot{x}_2$

7 Real data

In this section we will apply the Extended Kalman Filter algorithm on data that was gotten from an ADM that is running in the field. The exact mass in the ADM is not exactly known, but we have a benchmark estimation of a constant mass around 66.5kg which we can use to test the EKF estimation. The ADM used to generate this data is a different model of ADM than was used in the simulation. The model of the ADM is still the same, but the ADM model is smaller than the ADM350. This means that the mass of the slurry within the ADM module will also be smaller.

Besides the difference in the models of the ADM, the noise levels in the real data example might also be different than the assumptions we made in earlier sections. The measurement noise $\ddot{x}_{1,err}$ and $\ddot{x}_{2,err}$ are assumed to have the same characteristics, since the same type of accelerometers are used. The disturbance force F_d is different. We can use the same method to estimate the PSD of F_d which we can use to calculate the variance of F_d which in turn determines the covariance matrices used in the Kalman filter equations.

We will focus on a few different things when analysing the EKF on real data. First of all, since the mass of this ADM is smaller, there is a bigger chance that the nonlinear character of the EKF has an effect on the estimation. We have seen in section (6) that the EKF might find a local minimum if the initial value of the mass estimation is too small. Now that the actual mass lies closer to zero, we have to be more careful with our initialization to ensure that we do not end up in the local minimum. Since the model contains a factor of $1/m$, this means that the closer we are to zero, the more sensitive the convergence will be to initial conditions. An example of the sensitivity can be seen in figure 7.1. This shows a number of estimations using different initial guesses (with $var(Q_m = 10^{-5})$. We can see as in the simulations that there are cases in which the EKF converges to a local minimum instead of the absolute minimum.

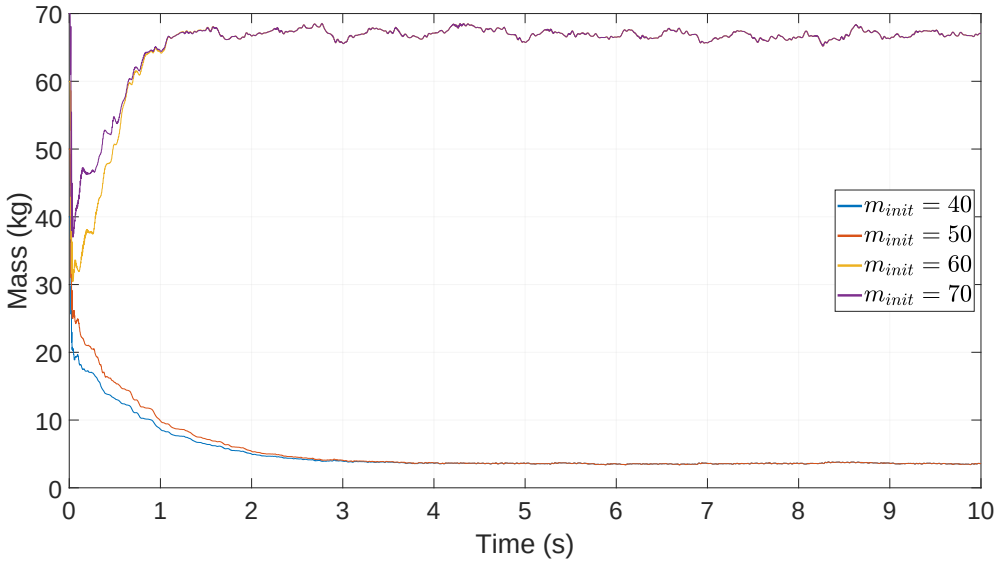


Figure 7.1: Real data with different initial guesses

We can also notice from the estimations that during the first moments of the estimation, the EKF estimation shoots down quite dramatically, which we have not seen in the simulations. Even when the initial mass estimation is 70kg (which is almost the correct value), the estimation shoots down below 40kg before correcting itself. A possible explanation can be found in other initializations. Other than the mass, we also need to give the EKF an initial guess for the other two states, z_1 and z_2, which represent the position and the velocity resp. of the inner tube relative to the outer tube. Since we do not have any information about the position and velocity at the start of the data, we have set them both to zero. It is possible that the actual position and velocity are significantly different than the initial guess, which means that the residual error is big, which in turn results in a bigger Kalman gain for the mass equation. The result is that the mass estimation overshoots until the estimations of z_1 and z_2 have corrected itself. An argument in favor of this theory is that when we start the estimation later (i.e. we throw away the first part of the data), this overshoot does not happen. When we start the estimation at 1 second for example, the EKF estimation does converge to the correct value even when the initial mass estimation is 40kg.

Besides the initial guess for the mass, there are several other parameters we can choose. The covariance of both the state and the mass estimation has to be initialized. It turns out that changing the initial covariance of the states z has no impact on the convergence of the estimation: if we run the EKF twice, once with $P_1(0) = 100I$ and once with $P_1(0) = 10^{-5}$, both estimations of the mass will run together after only a few iterations. If we look at the covariance matrix P_1 during the 10 seconds of estimation, then we see that it shoots down during the first few steps of the Kalman filter. After 5 steps of the Kalman filter, the top left value of P_1 (which is the covariance of the estimation of z_1) is already down to 10^{-5} when we chose $P_1(0) = 100I$. Note that this is using a sample rate of 4096, so after about a very short time, P_1 will be very low, even if we initialize it way too high. The initial covariance of the mass estimation has a greater impact on the convergence. This is because, as we have seen, the EKF estimation seems to shoot during the first period of the estimation and the

higher the initial covariance of the estimation is, the further it shoots down. This means that the lower the initial covariance is, the further the estimation stays away from the local minimum below 10kg. If we look at the value of P_3 during the estimation, we see that it if it starts too high, it will decrease much slower than P_1. A consequence of this is that during the initialization phase, the EKF will overestimate the covariance of the estimation. This means that if there is any error in the estimation, the EKF will overcompensate for that error which can lead to the overshoot we saw. A lower initial estimation covariance does mean that the EKF estimation needs longer to arrive at the correct mass after starting up. A problem that can arise in Kalman Filters, is that the estimation covariance converges to zero too quickly, such that the covariance reaches zero before the right estimation is obtained. This only happens in our case if we set the variance of Q_m to zero, since only then the covariance of the estimation would converge to zero. If the variance of Q_m is non-zero, the covariance of the estimation stays above a certain level, so the EKF will keep updating the mass estimation.

The next thing we can look at, is the influence of the variance of the fictional noise on the mass equation. For $var(Q_m) = 10^{-4}$, the estimation with the initial guess of 50kg does converge to the right minimum, and for $var(Q_m) = 10^{-3}$ all four of the initial guesses lead to a correct equilibrium. So the higher the variance of the fictitious noise Q_m is chosen, the more likely it is to find the correct minimum, even if the initial guess is too low.

We have seen that a big problem in EKF estimation is the nonlinearity of the minimizer, which can lead to the algorithm finding a local minimum. We have seen that there is a local minimum below 10kg which in some cases will be reached. A solution to this problem is to add a minimum mass to the EKF estimation. This can be backed up by a practical reason, since we know the mass will never go below the mass of an empty tube. In the case of the ADM200, the mass of an empty tube is 47kg, so we know that the mass will never go below 47kg. This means that in the update equation 5.30 we can force it to not estimate lower than 47kg.

Figure (7.2) shows the estimation using an initial guess of 50kg (which we have seen is an initial guess that ends up in a wrong minimum). The blue graph is the normal one, the red line is the kalman filter that includes the line that the new estimation is $max(47, \hat{m}(n))$, so the estimation will never go below 47kg. We can see that using this minimum will keep the estimation from reaching the local minimum.

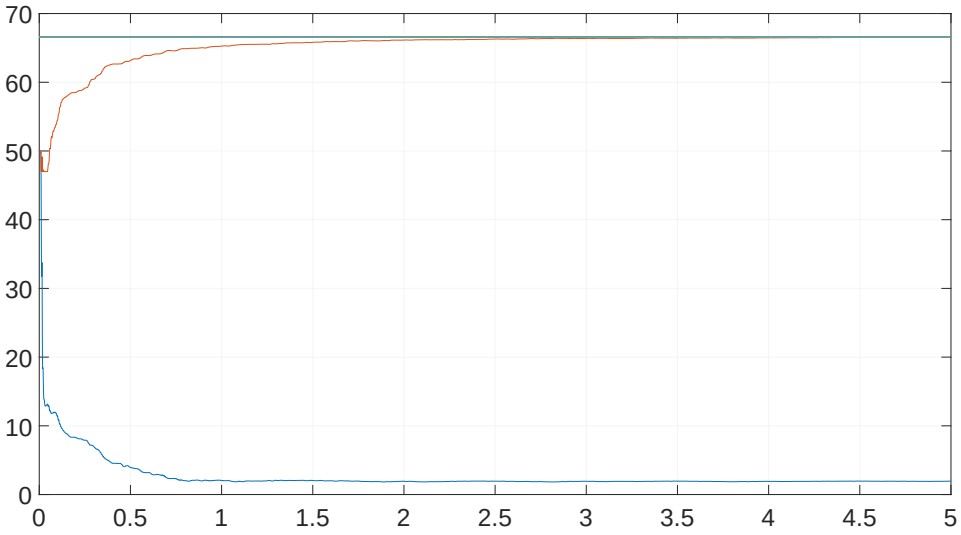


Figure 7.2: Real data with and without a minimum boundary

The next thing we can look at, is the effect of changing the variance of the noise on the mass equation, both on the noisiness of the estimation and the convergence speed. Figure 7.3 shows a number of estimations using different values for the variance of the (fictitious) noise Q_m, using an initial estimation of 60kg. Like in the simulations, a higher variance on Q_m means the estimation is noisier. The mass appears to be constant during this ten second window, but we can see that the estimation with higher variance on the fictitious noise converges faster after initialization.

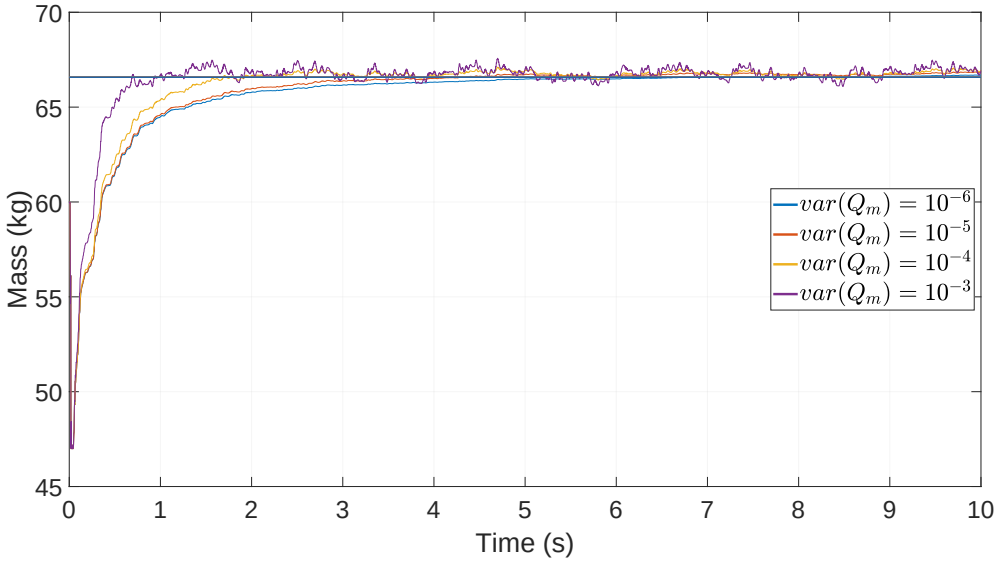


Figure 7.3: Real data with different Qm

8 Conclusion

The goal of this research is to find for a new algorithm to estimate the mass in the Alia Density Meter to fix the problems within the currently used algorithm. Firstly, the current algorithm is biased when the slurry that passes through the ADM is not uniform, i.e. when it contains larger particles that bounce through the tube. Secondly, the current algorithm cannot always adjust to changes in mass quickly enough.

To solve both of those problems, we have chosen to implement the Extended Kalman Filter (EKF). To test the EKF, we have simulated the ADM350 and applied the EKF estimation to data generated from this simulation. From this simulation, we concluded that the EKF has a smaller bias than the currently used algorithm. Using the EKF instead of the currently used algorithm reduces the bias from up to 10kg to only a few tenths of a kilo. Usual parameter estimation with Kalman Filters assumes that the parameter (in this case the mass) is constant. An artificial noise term is therefore added to make sure that the estimation can adjust to changes in mass. The speed in which the EKF can adjust to changes in mass depends on the intensity of the artificial noise: the higher the intensity, the more quickly the estimation adjusts. However, the higher the intensity, the noisier the estimation will be. Thus, the choice of the intensity of the artificial noise has to be chosen to fit the needs of the specific customer.

A problem that can arise due to the nonlinearity of the EKF is that a wrong equilibrium for the mass can be found if the initial conditions of the estimator are not good enough. This happens more often if the mass of the slurry is lower, so smaller and lighter models of the ADM might suffer from this. However, this problem can be solved by giving the EKF a minimum mass which it cannot go under. This minimum mass can be chosen as the mass of an empty tube, since it is known that the mass of the slurry in the ADM cannot go below that at any time.

Lastly we have implemented the EKF on a small sample of real data from the ADM250. The EKF estimation performed well in this test, its estimation did not show much bias compared to the benchmark estimation that we have been given by Demcon.

In conclusion, the EKF estimation algorithm is very suited for the ADM and it can solve the problems with the current algorithm very well.

References

[1] Altheris 4610, *Datasheet*, 15-08-2011 rev B.

[2] Rik Pintelon, Johan Schoukens, *System Identification*, Wiley, Second Edition, 2012.

[3] Lennart Ljung, *Asymptotic Behavior of the Extended Kalman Filter as a Parameter Estimator for Linear Systems*, February 1979 IEEE Transactions On Automatic Control, Vol. AC-24, No. 1.

[4] Gijs Boerrigter, *Development of Advanced Near Real-time Estimation Algorithms for the ALIA Density Meter*, 2018.

[5] Feng Ding, Ling Xu1, Quanmin Zhu, *Performance analysis of the generalised projection identification for time-varying systems*, IET Control Theory Appl., 2016, Vol. 10 Iss. 18, pp. 2506-2514.

[6] Lei Guo, *Estimating Time-Varying Parameters by the Kalman Filter Based Algorithm: Stability and Convergence*, IEEE Transactions On Automatic Control, Vol. 35, No. 2, February 1990.

[7] Xiao-Li Hu, James S. Welsh, *Convergence of a Recursive Least Squares Algorithm for Frequency Domain Identification*, International Federation of Automatic Control, 2011.

[8] Yuvin Adnarain Chinniah, *Fault Detection In the Electrohydraulic Actuator Using Extended Kalman Filter*, University of Saskatchewan, Saskatoon Canada, March 2004.

[9] Andrea Arnold, Daniela Calvetti, Erkki Somersalo, *Parameter estimation for stiff deterministic dynamical systems via ensemble Kalman filter*, IOP Publishing, Inverse Problems, 2004.

[10] Deepak Sridhar, Debarshi Patanjali Ghoshal, Hannah Michalska, *B-Splines in Joint Parameter and State Estimation in Linear Time-Varying Systems* 2018 Annual American Control Conference.

[11] José A. Castellanos, José Neira, Juan D. Tardós, *Limits to the Consistency of EKF-based SLAM*

[12] Simon J. Julier, Jeffrey K. Uhlmann, *A New Extension of the Kalman Filter to Nonlinear Systems*, The Robotics Research Group, Department of Engineering Science, The University of Oxford.

[13] Eric A. Wan, Rudolph van der Merwe, *The Unscented Kalman Filter for Nonlinear Estimation*, Proc. Symp. Adaptive Syst. Signal Process., Commun. Contr., Lake Louise, AB, Canada, Oct. 2000.

[14] T.C. Fung, *Computation of the matrix exponential and its derivatives by scaling and squaring*, Int. J. Numer. Meth. Engng, 2004.