

# Inhoudsopgawe

<b>1. System identification</b>	<b>1</b>
1.1. White-box and black-box techniques . . . . .	1
1.1.1. White-box techniques . . . . .	1
1.1.2. Black-box techniques . . . . .	2
1.2. Plant considered for system identification . . . . .	3
1.3. Parameter estimation . . . . .	3
1.3.1. Predetermined linear model . . . . .	3
1.3.2. Payload mass estimation . . . . .	4
1.3.3. Cable length estimation . . . . .	4
1.4. Dynamic Mode Decomposition with control . . . . .	5
1.5. Hankel Alternative View Of Koopman . . . . .	7
1.6. Implementation and results . . . . .	9
1.6.1. Methodology . . . . .	9
1.6.2. Hyperparameters . . . . .	10
1.6.3. Sample time . . . . .	10
1.6.4. Choice of payload variable in the state vector . . . . .	10
1.6.5. Noise . . . . .	11
1.6.6. Size of training data . . . . .	12
1.6.7. System parameters . . . . .	13
1.6.8. Dynamic payload . . . . .	15
<b>Bibliografie</b>	<b>17</b>
<b>A. Project Planning Schedule</b>	<b>18</b>
<b>B. Outcomes Compliance</b>	<b>19</b>

# Hoofstuk 1

## System identification

System identification is the process of creating mathematical models of a dynamical system by using input and output measurements of that system. Two major approaches are used to represent the dynamics of such a system:

1. A priori mathematical modelling with parameter estimation
2. Data-driven system identification

Models determined from a priori modelling and parameter estimation are referred to as white-box models. In contrast, data-driven system identification methods result in black-box models. This chapter discusses these system identification approaches and describes the differences between them. For each approach, different estimation techniques are explained and applied to the quadrotor and payload system. The results of these techniques are then compared to each other.

### 1.1. White-box and black-box techniques

#### 1.1.1. White-box techniques

The underlying physics of a white-box model is understood by the user because they are determined from first principles. This is done by modelling physical processes with techniques like Lagrangian mechanics or Newton equations. With system identification techniques that use these models, the mathematical relations between system parameters are predefined in the modelling phase. The system identification process is therefore reduced to parameter estimation to determine values for parameters used in the model.

This approach is used by [1] and [2] for swing damping control of a quadrotor with an unknown suspended payload. The system was modelled as two rigid bodies connected by a link and the following assumptions were made regarding the suspended payload:

- The payload is a point mass.
- The link is massless.

- The link is rigid.
- The link is attached to the CoM of the quadrotor.

The only unknown parameters in the quadrotor and payload model is the payload mass and link length. These parameters are first estimated and then inserted into the predefined, linearised model. This model is used by a LQR controller to damp swing angles while also controlling the vehicle.

The approach works well for systems with predictable dynamics, but it is not very adaptable. The payload considered by [1] and [2] is limited to a small rigid mass suspended from the quadrotor by a non-stretching cable. In this use case it was shown that a LQR controller successfully controls a quadrotor while minimising payload swing angles. However, if a payload or cable is used that violates one of the modelling assumptions, the predefined model no longer accurately represent the system. Since the controller is dependent on this model, the mismatch between the model and actual dynamics may result in undesirable controller behaviour.

### 1.1.2. Black-box techniques

Data-driven system identification methods produce black-box models. In contrast to white-box models, black-box models do not require predefined mathematical relations between system parameters. No prior knowledge of the physics of the system are considered and no modelling assumptions are made. Black-box techniques determine the mathematical relationship between inputs and outputs of a system using information from measurement data only.

Black-box models can be categorised as either non-linear or linear models. Non-linear models are often more accurate than linear models because complex, real-world dynamics are better approximated by non-linear systems. The dynamics of a quadrotor and suspended payload are also non-linear. Examples of black box models with quadrotors and payloads in literature ???

However, non-linear models are inherently more complex than linear models. Controllers that use non-linear models are usually more computationally complex than those with linear models. Control architectures for quadrotors used in practical applications are mostly implemented on onboard hardware. Therefore there is value in low-complexity, linear models since these may be simple enough to execute on low cost hardware. trade-off between accuracy and complexity. Non-linear models may require control implementations that are too computationally expensive and may not be practically realisable on the available hardware on a quadrotor.

DMDc and HAVOK are the two data-driven system identification methods investigated in this paper. These are linear regression techniques that produce a linear model to approximate non-linear dynamics. Non-linear data-driven techniques like Neural Networks and SINDy [?] may produce models that are more accurate than linear techniques, but at the cost of greater computational complexity. [Name more techniques](#) DMDc and HAVOK are less computationally complex and their models are suitable for linear MPC, which is significantly faster than non-linear MPC. This is desirable for the quadrotor use case, where onboard computational power is limited.

These techniques and their implementation are explained in the sections below. Each technique is considered for use in a velocity controller in the North direction

**Considered controller** The model identified by DMDc will be used to design a longitudinal velocity controller. As shown in ??, the plant considered for system identification includes the dynamics of the inner loop, attitude controllers. The swing damping controllers which will utilise the identified model act only in the translational velocity loop. Because of the large time-scale separation between the inner and outer loop controllers, the attitude states have a negligible effect on the plant dynamics seen by the velocity controller. As discussed in Section ??, the payload minimally effects the quadrotor attitude because it is attached near the CoM of the vehicle. Therefore the attitude states are excluded from the system identification model.

## 1.2. Plant considered for system identification

**Derived model** Figure 1.1 shows the plant considered for system identification. In Chapter ?? the differential equations that describe the motion of this system are derived with Lagrangian mechanics. From these equations it is clear that the considered plant is defined by the state vector,

$$\mathbf{x} = \begin{bmatrix} V_N & \dot{\theta} \end{bmatrix}^T, \quad (1.1)$$

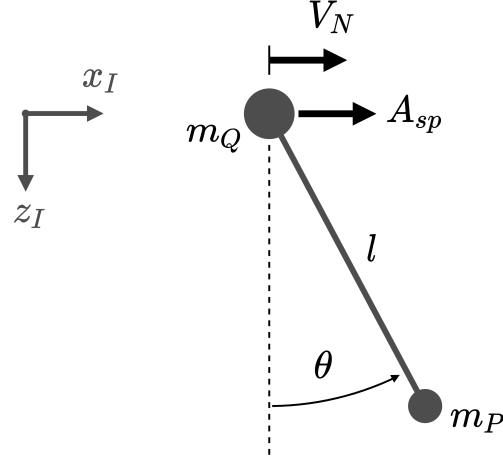
and the input vector,

$$\mathbf{u} = \begin{bmatrix} A_{N,sp} \end{bmatrix}, \quad (1.2)$$

## 1.3. Parameter estimation

### 1.3.1. Predetermined linear model

The motivation for parameter estimation is to determine unknown parameter values required by the predetermined model. This model was derived a priori in Section ??.



**Figuur 1.1:** Floating pendulum model considered for system identification for a North velocity controller

### 1.3.2. Payload mass estimation

RLS Cannot work if quadrotor mass also changes, because assume that  $m_q$  is known.

### 1.3.3. Cable length estimation

The cable length is estimated from the measurement of natural frequency of the swinging payload. As described by [?], the natural frequency is given by:

$$\omega_n = \sqrt{\frac{g}{l} \cdot \frac{m_q + m_p}{m_q}} \quad (1.3)$$

The natural frequency is measured by performing a FFT on the payload swing angle response after a position step by the quadrotor. The dominant frequency identified by the FFT during free swing is the natural frequency of the payload.

?? shows the payload swing angle after the system is stimulated by a position step setpoint. As shown in ?? the first few seconds of the step response are not used in the FFT. This is to minimise the effect of the quadrotor controllers on the swing angle frequency by excluding the transient response in the FFT.

?? shows the resulting amplitude spectrum of the payload swing angle response. The dominant frequency is clearly identified as ??. Since  $m_q$  and  $g$  is known, and  $m_p$  and  $\omega_n$  has been estimated,  $l$  can now be determined from 1.3. In this case the estimated length is ??, compared to the actual length of ??.

Frequency resolution ?? error for different lengths??

## 1.4. Dynamic Mode Decomposition with control

**Intro** DMD is a linear regression technique that can be used to approximate a non-linear dynamical system [3]. It uses temporal measurements of system outputs to reconstruct system dynamics without prior modelling assumptions. DMDc is an adaptation of DMD that also accounts for control inputs [4]. This section provides an overview of the specific implementation of DMDc used in this paper. Note that this implementation is a slight adaptation of DMDc, and includes time-delay-embedding of multiple observables. [5] and [6] use time-delay-embedding in their DMD adaptations in similar ways.

**State space model** DMD produces a linear, discrete state-space model of the system dynamics. Discrete measurements,  $\mathbf{x}_k$ , of the continuous time observable,  $\mathbf{x}(t)$ , are used, where  $\mathbf{x}_k = \mathbf{x}(kT_s)$ , and  $T_s$  is the sampling time of the model. Delay-coordinates (i.e.  $\mathbf{x}_{k-1}, \mathbf{x}_{k-2}$ , etc.) are also included in the state-space model to account for input delay and state delay in the system. Input delay refers to the time delay involved with transporting a control signal to a system, whereas state delay refers to time-separated interactions between system variables [7]. Hence, we define an state delay vector as:

$$\mathbf{d}_k = \begin{bmatrix} \mathbf{x}_{k-1} & \mathbf{x}_{k-2} & \cdots & \mathbf{x}_{k-q} \end{bmatrix}^T, \quad (1.4)$$

$\mathbf{d}_k \in \mathbb{R}^{(n_x)(q)}$  and where  $q$  is the number of delay-coordinates used in the model.

The discrete state-space model is therefore defined as:

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{A}_d\mathbf{d}_k + \mathbf{B}\mathbf{u}_k, \quad (1.5)$$

$\mathbf{A} \in \mathbb{R}^{n_x \times n_x}$  is the system matrix,  $\mathbf{A}_1 \in \mathbb{R}^{(q \cdot n_x) \times (q \cdot n_x)}$  is the state delay system matrix and  $\mathbf{B} \in \mathbb{R}^{n_x \times n_u}$  is the input matrix.

**Training data** The training data consists of full-state measurements,  $\mathbf{x}_k$ , and corresponding inputs,  $\mathbf{u}_k$ , taken at regular intervals of  $\Delta t = T_s$ , during a simulated flight with Cascaded PID control. In a practical flight, these time-series measurements need to be saved in memory because it is used as a single batch by DMD. Note that DMD can be applied in a recursive manner as described in [8], However this implementation is not considered because memory size will not be a limitation since a companion computer will be used.

**Data matrices** The training data is collected into the following matrices:

$$\begin{aligned} \mathbf{X}' &= \begin{bmatrix} \mathbf{x}_{q+2} & \mathbf{x}_{q+3} & \mathbf{x}_{q+4} & \cdots & \mathbf{x}_{w+q+1} \end{bmatrix}, \\ \mathbf{X} &= \begin{bmatrix} \mathbf{x}_{q+1} & \mathbf{x}_{q+2} & \mathbf{x}_{q+3} & \cdots & \mathbf{x}_{w+q} \end{bmatrix}, \\ \mathbf{X}_d &= \begin{bmatrix} \mathbf{x}_q & \mathbf{x}_{q+1} & \mathbf{x}_{q+2} & \cdots & \mathbf{x}_{w+q-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{x}_2 & \mathbf{x}_3 & \mathbf{x}_4 & \cdots & \mathbf{x}_{w+1} \\ \mathbf{x}_1 & \mathbf{x}_2 & \mathbf{x}_3 & \cdots & \mathbf{x}_w \end{bmatrix}, \\ \mathbf{\Upsilon} &= \begin{bmatrix} \mathbf{u}_q & \mathbf{u}_{q+1} & \mathbf{u}_{q+2} & \cdots & \mathbf{u}_{w+q-1} \end{bmatrix}, \end{aligned} \quad (1.6)$$

where  $w$  is the number of columns in the matrices,  $\mathbf{X}'$  is the matrix  $\mathbf{X}$  shifted forward by one time-step,  $\mathbf{X}_d$  is the matrix with delay states, and  $\mathbf{\Upsilon}$  is the matrix of inputs. Equation (1.5) can be combined with the matrices in Equation (1.6) to produce:

$$\mathbf{X}' = \mathbf{A}\mathbf{X} + \mathbf{A}_d\mathbf{X}_d + \mathbf{B}\mathbf{\Upsilon}. \quad (1.7)$$

Note that the primary objective of DMDC is to determine the best fit model matrices,  $\mathbf{A}$ ,  $\mathbf{A}_d$  and  $\mathbf{B}$ , given the data in  $\mathbf{X}'$ ,  $\mathbf{X}$ ,  $\mathbf{X}_d$ , and  $\mathbf{\Upsilon}$  [4]. In order to group the unknowns into a single matrix, (1.5) is manipulated into the form,

$$\mathbf{X}' = \begin{bmatrix} \mathbf{A} & \mathbf{A}_d & \mathbf{B} \end{bmatrix} \begin{bmatrix} \mathbf{X} \\ \mathbf{X}_d \\ \mathbf{\Upsilon} \end{bmatrix} = \mathbf{G}\mathbf{\Omega}, \quad (1.8)$$

where  $\mathbf{\Omega}$  contains the state and control data, and  $\mathbf{G}$  represents the system and input matrices.

**SVD** A SVD is performed on  $\mathbf{\Omega}$  resulting in:  $\mathbf{\Omega} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ . Often, only the first  $p$  columns of  $\mathbf{U}$  and  $\mathbf{V}$  are required for a good approximation of the dynamics [9]. Talk about Reduced Order Modelling?? POD modes?? In many cases, the truncated form results in better models than the exact form when noisy measurements are used. This is because the effect of measurement noise is mostly captured by the truncated columns of  $\mathbf{U}$  and  $\mathbf{V}$ . By truncating these columns, the influence of noise in the regression problem is reduced. [explain this better](#) hence the SVD is used in the truncated form:

$$\mathbf{\Omega} \approx \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^T, \quad (1.9)$$

where  $\sim$  represents rank- $p$  truncation. [maybe insert colour pictures showing matrices](#)

By combining (1.9) with the over-constrained equality in (1.8), the least-squared solution,  $\mathbf{G}$ , can be found with:

$$\mathbf{G} \approx \mathbf{X}'\tilde{\mathbf{V}}\tilde{\Sigma}^{-1}\tilde{\mathbf{U}}. \quad (1.10)$$

By reversing 1.8,  $\mathbf{G}$  can now be separated into:  $\mathbf{G} = [\mathbf{A} \ \mathbf{A}_d \ \mathbf{B}]$ . according to the required dimensions of each matrix. Thereby, the state-space model approximated by DMDc is complete.

## 1.5. Hankel Alternative View Of Koopman

$q$  = number of delays, from here up

HAVOK is a data-driven, regression technique that provides a connection between DMD and Koopman operator theory [?, 9]. We have adapted the standard HAVOK algorithm slightly to account for the effect of control and to extract a discrete, linear model that approximates the behaviour of a controlled dynamical system. In this section, a brief overview is provided for this implementation and expansion of HAVOK.

The extracted discrete state-space model is defined as:

$$\mathbf{a}_{k+1} = \tilde{\mathbf{A}}\mathbf{a}_k + \tilde{\mathbf{B}}\mathbf{u}_k, \quad (1.11)$$

where  $\mathbf{a}_k$  is the state vector previously defined in Section 1.4,  $\tilde{\mathbf{A}} \in \mathbb{R}^{(q \cdot n_x) \times (q \cdot n_x)}$  is the system matrix, and  $\tilde{\mathbf{B}} \in \mathbb{R}^{(q \cdot n_x) \times n_u}$  is the input matrix. Here,  $\sim$  is used to differentiate these matrices from  $\mathbf{A}$  and  $\mathbf{B}$  used in DMDc.

The original HAVOK algorithm, developed by [?], constructs a Hankel matrix from output variables only. In order to incorporate the effect of control, an extended Hankel matrix,  $\mathbf{\Pi}$ , is created by appending a matrix of inputs to a Hankel matrix of measurements:

$$\mathbf{\Pi} = \begin{bmatrix} \mathbf{a}_q & \mathbf{a}_{q+1} & \mathbf{a}_{q+2} & \cdots & \mathbf{a}_{w+q-1} \\ \mathbf{u}_q & \mathbf{u}_{q+1} & \mathbf{u}_{q+2} & \cdots & \mathbf{u}_{w+q-1} \end{bmatrix}, \quad (1.12)$$

where  $w$  is the number of columns in  $\mathbf{\Pi}$ . A truncated SVD of this Hankel matrix results in following approximation:

$$\mathbf{\Pi} \approx \tilde{\mathbf{U}}\tilde{\Sigma}\tilde{\mathbf{V}}^T, \quad (1.13)$$

where  $\sim$  represents rank- $p$  truncation. It is important to note that the model extracted by HAVOK depends on the choice of hyperparameters,  $p$  and  $q$ . The number of samples in the training data,  $N_{train} = w + q - 1$ , also influences the accuracy of the model.



The columns of  $\tilde{\mathbf{V}}$  are the most significant principal components of the system dynamics [?]. This matrix,  $\tilde{\mathbf{V}}$ , can be considered to contain a time-series of the pseudo-state,  $\mathbf{v}$ , such that  $\tilde{\mathbf{V}}^T = [\mathbf{v}_q \ \mathbf{v}_{q+1} \ \cdots \ \mathbf{v}_w]$ , characterises the evolution of the actual dynamics in an eigen-time-delay coordinate system [?]. Consider the following discrete, state-space formulation:

$$\mathbf{v}_{k+1} = \mathbf{\Lambda} \mathbf{v}_k. \quad (1.14)$$

Recall that DMDc finds a best fit linear operator that directly maps  $\mathbf{a}_k$  to  $\mathbf{a}_{k+1}$ . Similarly, HAVOK determines the best fit linear operator  $\mathbf{\Lambda}$  that maps the pseudo-state  $\mathbf{v}_k$  to  $\mathbf{v}_{k+1}$ . So, in order to setup an over-determined equality for (1.14),  $\tilde{\mathbf{V}}^T$  is divided into two matrices:

$$\begin{aligned} \mathbf{V}_1 &= \begin{bmatrix} \mathbf{v}_q & \mathbf{v}_{q+1} & \cdots & \mathbf{v}_{w-1} \end{bmatrix}, \\ \mathbf{V}_2 &= \begin{bmatrix} \mathbf{v}_{q+1} & \mathbf{v}_{q+2} & \cdots & \mathbf{v}_w \end{bmatrix}, \end{aligned} \quad (1.15)$$

where  $\mathbf{V}_2$  is  $\mathbf{V}_1$  advanced a single step forward in time. The matrices from Equation (1.15) are now combined with Equation (1.14) and the best fit  $\mathbf{\Lambda}$  is determined with the Moore-Penrose pseudoinverse:

$$\mathbf{V}_2 = \mathbf{\Lambda} \mathbf{V}_1 \quad \Rightarrow \quad \mathbf{\Lambda} \approx \mathbf{V}_1 \mathbf{V}_1^\dagger \quad (1.16)$$

It can be shown from Equation (1.13) that Equation (1.14) is transformed from the eigen-time-delay coordinate system to the original coordinate system as the following:

$$\begin{bmatrix} \mathbf{a}_{k+1} \\ \mathbf{u}_{k+1} \end{bmatrix} = (\tilde{\mathbf{U}} \tilde{\mathbf{\Sigma}}) \mathbf{\Lambda} (\tilde{\mathbf{U}} \tilde{\mathbf{\Sigma}})^\dagger \begin{bmatrix} \mathbf{a}_k \\ \mathbf{u}_k \end{bmatrix}. \quad (1.17)$$

This form is used to extract  $\tilde{\mathbf{A}}$  and  $\tilde{\mathbf{B}}$  from the matrix,  $(\tilde{\mathbf{U}} \tilde{\mathbf{\Sigma}}) \mathbf{\Lambda} (\tilde{\mathbf{U}} \tilde{\mathbf{\Sigma}})^\dagger$ , in the following way:

$$\begin{bmatrix} \mathbf{a}_{k+1} \\ \mathbf{u}_{k+1} \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{A}} & \tilde{\mathbf{B}} \\ (discarded) \end{bmatrix} \begin{bmatrix} \mathbf{a}_k \\ \mathbf{u}_k \end{bmatrix}. \quad (1.18)$$

Note that the matrix entries in (1.18) that map  $\mathbf{u}_k$  to  $\mathbf{u}_{k+1}$  are meaningless for our purposes and are discarded. Similarly to DMDc, some matrix entries in  $\tilde{\mathbf{A}}$  and  $\tilde{\mathbf{B}}$  are known a priori due to the relative positions of delay coordinates. These are forced to 1 or 0 to improve the prediction performance of the model.

[merge these paragraphs](#) Since the state vector,  $\mathbf{a}$ , includes delay-coordinates, some matrix entries are known a priori and are independent of the dynamics. For example, the values of  $\mathbf{x}_k$  should be mapped from their position in  $\mathbf{a}_k$  to specific indices in  $\mathbf{a}_{k+1}$ . Due to the least-squares fitting and coordinate transformation, DMDc will not produce these exact values in  $\mathbf{A}$  and  $\mathbf{B}$ . By forcing each of these matrix entries to 1 or 0, the

state-prediction performance of the model is improved.

## 1.6. Implementation and results

### 1.6.1. Methodology

#### Simulation environment

**Method overview** [Maybe convert this to a flow diagram](#)

1. Takeoff and hover
2. Command a series of velocity step inputs with random step sizes and time intervals
3. Measure and save input and output data
4. Apply algorithm to data and generate model

**Steps and intervals** For the training period, different velocity step inputs are commanded with varying time intervals between step commands. A algorithm schedules these velocity step commands, by assigning random step values and time-intervals within a specified range. The velocity range is determined in simulation by iteratively increasing the maximum velocity step to a safe value where the quadrotor and payload system remain in stable flight. The maximum time-interval is set to a value that allows the payload swing to reach a steady-state condition. This ensures that the identified model includes transient and steady-state dynamics.

**Why velocity steps?** Velocity step commands are used in the training period because this Frequency decomposition stimulates the system for a large range of frequencies.

**Testing data** Cross-validate

**Error metric** Each state error signal is scaled by the reciprocal of the maximum value of that state variable in the training data. This is to ensure that a scale difference in the variable types create a bias in the error metric. For example, the quadrotor velocity reaches values of 3 m/s but the payload swing angle has a maximum of only 0.526 rad. The velocity prediction error is therefore inherently larger than the payload angle prediction error and will bias the error metric towards favouring models with good velocity predictions. The proposed scaled error metric ensures that the MAE of each state variable can be compared

to each other. It also provides an error metric that is better and unbiased representative of the model prediction performance across all state variables.

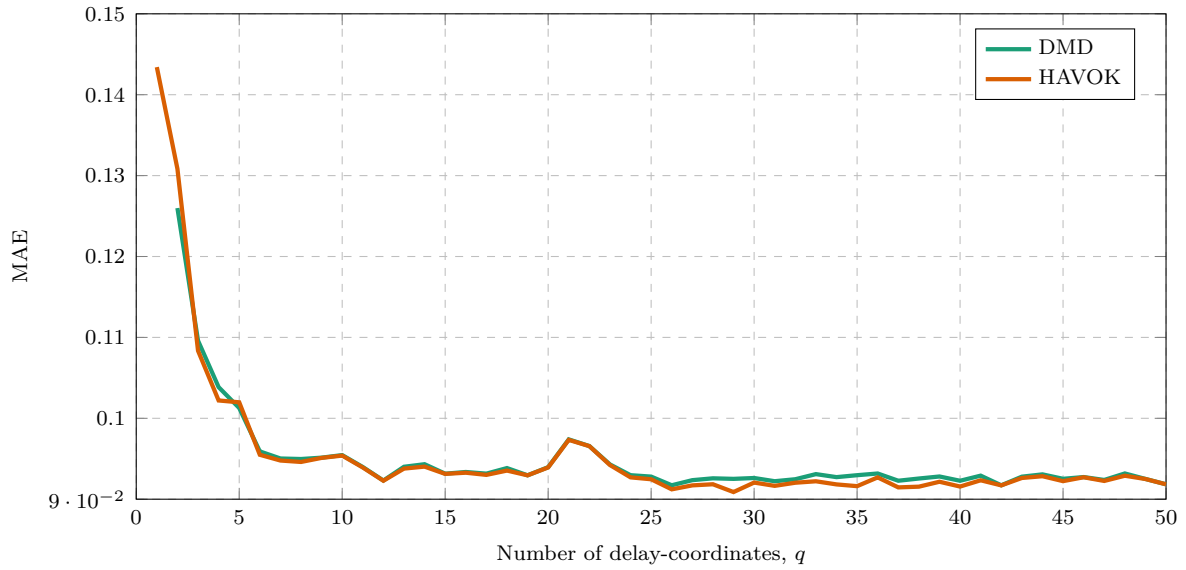
Add Information Criteria ?? AIC Brunton Kutz??

### 1.6.2. Hyperparameters

As discussed in Section 1.4 and 1.5 DMDc and HAVOK are dependent on two hyperparameters: the number of delay-coordinates,  $q$ , and the SVD truncation rank,  $p$ .

Parsimony Pareto front cite Data-Driven book

The more terms better chance to overfit, lower generalisation



**Figure 1.2:** DMD and HAVOK predictions error for different lengths of noisy training data ( $m = 0.2$  kg,  $l = 0.5$  m,  $T_s = 0.03$  s,  $T_{train} = 60$  s.)

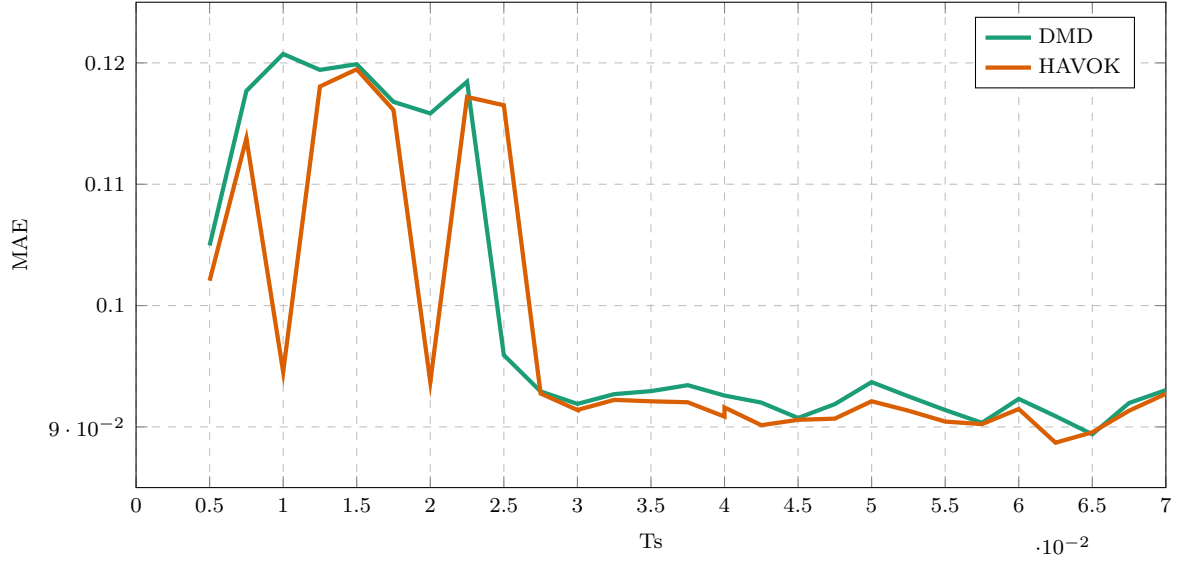
Fixed size of data Fixed sample time Fixed pendulum params Talk about the "front" Also about singular values For each of the experiments shown in this chapter, a hyperparameters selected tuned to produced

### 1.6.3. Sample time

best hyperparameters. Fixed size of data. Fixed pendulum params.

### 1.6.4. Choice of payload variable in the state vector

As discussed in Section 1.2, the equations of motion of a floating pendulum in continuous-time are dependent on  $\dot{\theta}$  and  $V_N$ , but are not dependent on  $\theta$ . Therefore it is expected that  $\mathbf{x} = [V_N \ \dot{\theta}]^T$  is used as the state vector for system identification. However, if  $\dot{\theta}$  is not included in the state vector of a discrete model, it can still be represented with numerical



**Figuur 1.3:** DMD and HAVOK predictions error for different lengths of noisy training data ( $m = 0.2 \text{ kg}$ ,  $l = 0.5 \text{ m}$ ,  $T_s = 0.03 \text{ s}$ ,  $T_{train} = 60 \text{ s}$ .)

differentiation like the backward Euler form,

$$\dot{\theta}_k = \left(\frac{1}{T_s}\right) \cdot \theta_k - \left(\frac{1}{T_s}\right) \cdot \theta_{k-1}. \quad (1.19)$$

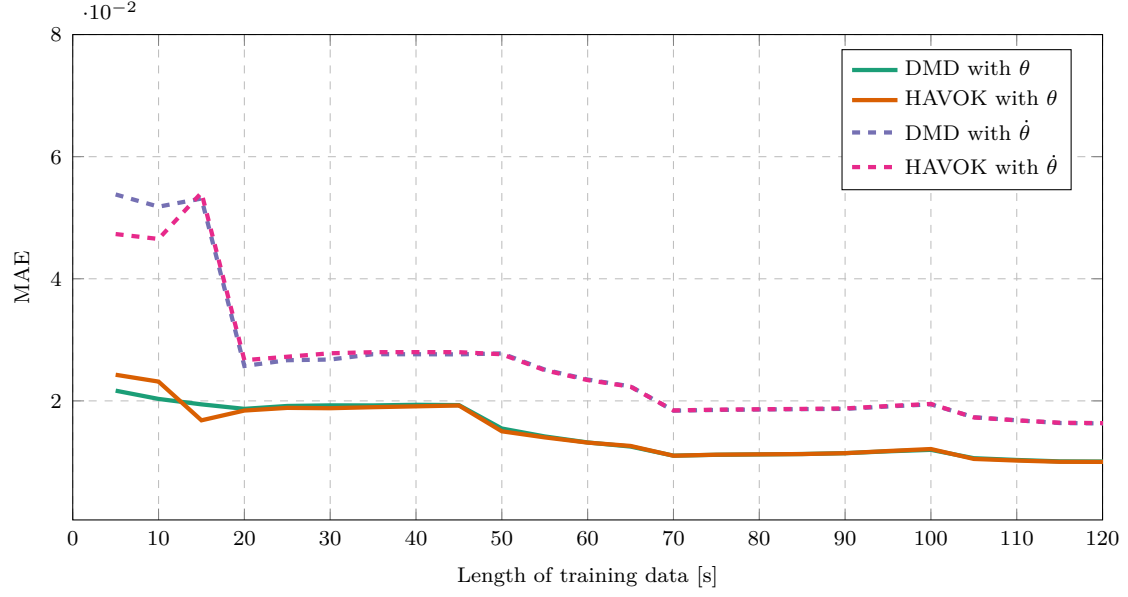
Therefore the original state vector can also be replaced by,  $\mathbf{x} = [V_N \ \theta]^T$  in system identification.

Based on the floating pendulum equations, it is expected that a model derived with  $\dot{\theta}$  data will better approximate the actual dynamics than one using  $\theta$ . This is because  $\dot{\theta}$  contains more direct information about the dynamics compared to  $\theta$ . A model using  $\theta$  needs to "learn" numerical differentiation and the effect of  $\dot{\theta}$  on the other variables. A model using  $\dot{\theta}$  only needs to consider its relationship with other variables.

For each length of training data, the hyperparameter combination producing the lowest prediction error was determined and used. From this plot it is clear that models with  $\theta$  produce more accurate predictions than those with  $\dot{\theta}$ .

### 1.6.5. Noise

Measurement noise is [Find reference for measurement noise definition](#) This is bad for system identification because the output signals no longer represent the actual process hides the actual dynamics of the system under The IMU, barometer, magnetometer and GPS sensors on the practical quadrotor are used for state estimation and all experience measurement noise. The EKF performs sensor fusion and smooths out most of the measurement noise to provide a state estimate that is less noisy than raw sensor values.



**Figure 1.4:** Prediction MAE for models using angle or angular rate measurements ( $m = 0.2$  kg,  $l = 0.5$  m,  $T_s = 0.03$  s).

The potentiometer and ADC which measure the payload angle on the quadrotor also has quite a lot of measurement noise. However, this signal is not smoothed by an onboard EKF. Figure ?? shows the noisy payload angle measurement for a practical pendulum test while the quadrotor is held stationary. For models using  $\theta$  in the state vector instead of  $\dot{\theta}$ , this noisy signal can be smoothed with [matlab smoother](#). Figure ?? compares the noisy payload angle measurement to the smoothed signal and actual payload angle for a simulated flight. This is applied as band-limited white-noise and the noise power was iteratively adjusted to match that of the practical payload measurements.

However, since there is no direct measurement of  $\dot{\theta}$ , numerical differentiation is performed on the noisy  $\theta$  measurement to estimate  $\dot{\theta}$ . This amplifies the noise and results in inaccurate  $\dot{\theta}$  signal. Total variation differentiation is implemented to estimate  $\dot{\theta}$  from the noisy measurements more accurately. [] Figure ?? shows

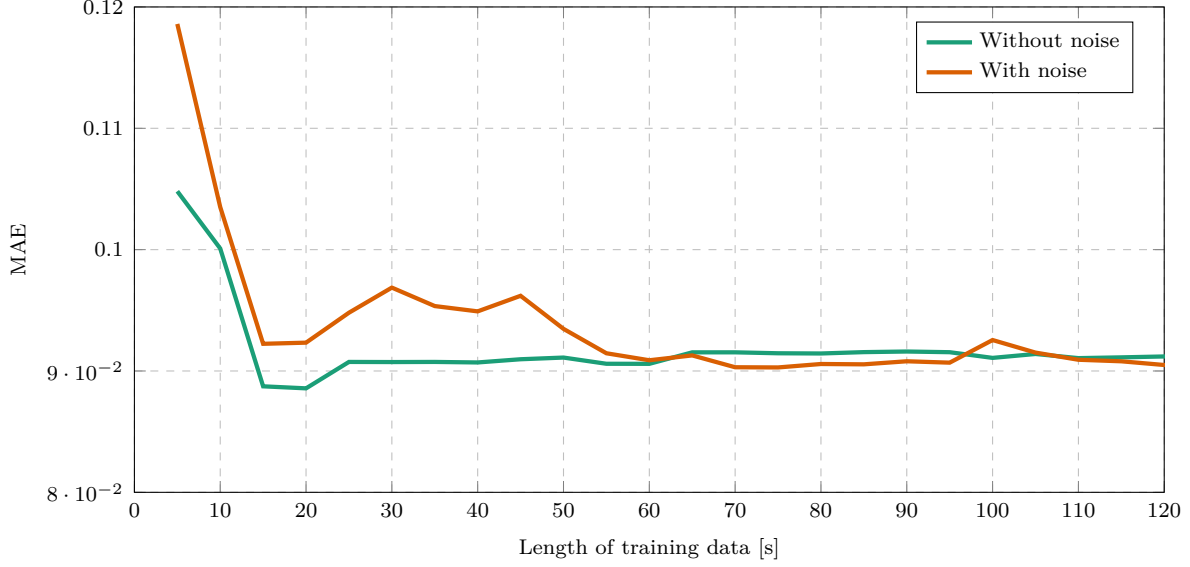
Noise also affects model prediction accuracy and the length of training data required for adequate predictions.

HAVOK performs better than DMD. This slight difference in prediction performance has a negligible effect on control.

Input data needs to be adjusted.

### 1.6.6. Size of training data

The length of training data used for system identification affects the quality of the model produced. In Figure 1.4 it is clear that prediction error decreases as the amount of training data increases. As more training data is used in the regression problem, the determined



**Figure 1.5:** HAVOK prediction error for different lengths of training data with and without noise ( $m = 0.2 \text{ kg}$ ,  $l = 0.5 \text{ m}$ ,  $T_s = 0.03 \text{ s}$ ).

model better approximates the actual dynamics because a large range of the dynamics is "seen" by the algorithm.

Models produced from data lengths as short as 5 s predict the movement of state variables surprisingly well. Note how the general shape of the prediction represents the training data, even though it contains a lot more high frequency oscillations.

Define MAE diff with equation ??.

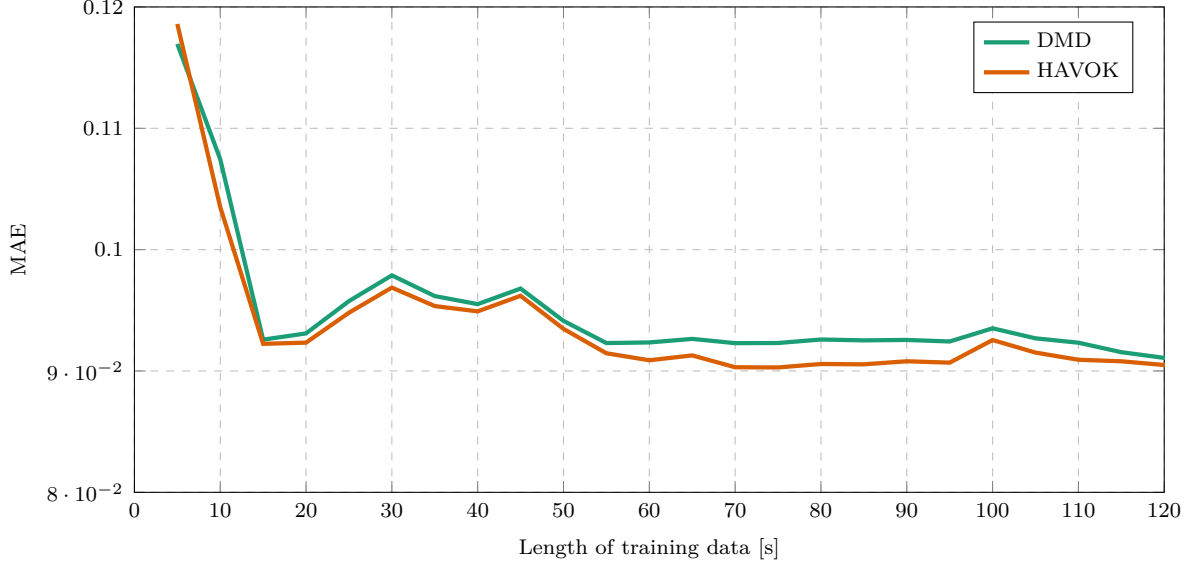
The models produced from HAVOK appear to produce slightly better prediction errors, however this small difference has a negligible effect on control performance.

the prediction error does not significantly improve with more training data. In practice less training data is desirable because less flight time will be wasted on training a model before the quadrotor can fly with a updated controller. Less training data also corresponds to lower memory usage on quadrotor hardware. Such a slight improvement in prediction error also has a negligible effect on control performance and is therefore not worth the increased data requirement.

### 1.6.7. System parameters

The payload acting as a single floating pendulum, as described in Section 1.2, has two system parameters,  $m_p$  and  $l$ . For the practical quadrotor considered, the payload mass is limited to:

$$0.01 \leq m_p \leq 0.4 \text{ kg}. \quad (1.20)$$



**Figure 1.6:** DMD and HAVOK prediction error for different lengths of noisy training data ( $m = 0.2$  kg,  $l = 0.5$  m,  $T_s = 0.03$  s).

When no external payload is attached, the connection device attached to the end of the cable is  $m_p = 0.01$  kg. On the other limit,  $m_p = 0.4$  kg is determined to be the maximum payload mass the quadrotor can carry safely based on the maximum thrust of the motors.

The cable length is limited to:

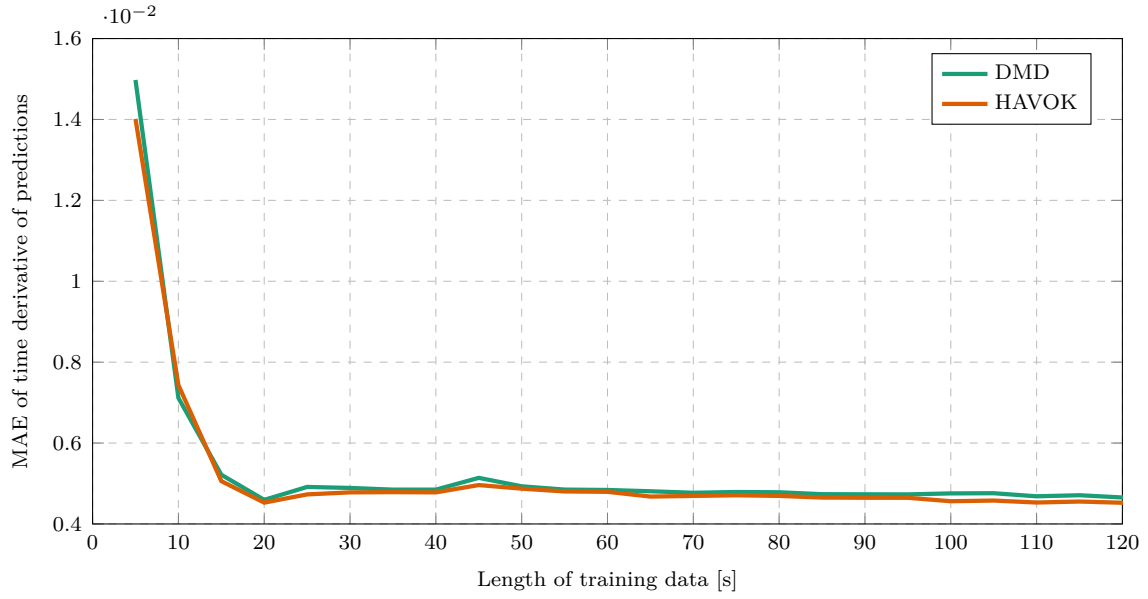
$$0.025 \leq l \leq 2 \text{ m.} \quad (1.21)$$

A cable length shorter than 0.025 m is quite impractical and may rather be attached as a rigid payload. There are very few practical applications that may require a shorter cable length. It is also unsafe to fly with a shorter cable length, since the payload may collide with the quadrotor during an uncontrolled swing. A longer cable guards against a payload and vehicle collision, because more energy needs to be transferred to the payload to reach the height of the vehicle. The maximum cable length is selected as  $l = 2$  m by intuition since a cable much longer than this may not be practically useful for a drone delivery flight with the considered quadrotor.

Plot MAE vs  $l$  Plot MAE vs  $m$  See if you can rather plot MAE vs  $l$ , with contours of  $m$

See how it affects  $T_s$  plot MAE vs  $T_s$  with contours of  $l$

Best hyperparameters. Fixed size of data. Fixed sample time.



**Figure 1.7:** DMD and HAVOK error of time derivative of predictions for different lengths of noisy training data ( $m = 0.2$  kg,  $l = 0.5$  m,  $T_s = 0.03$  s).

### 1.6.8. Dynamic payload

Some payloads attached to the cable may not satisfy the assumptions made in Section 1.2. For example, if a long payload is attached to the cable, the CoM of the payload will be quite a distance below the attachment of the cable. This creates a double pendulum model which has different dynamics than a single pendulum.

[insert picture a practical quad with long payload](#)

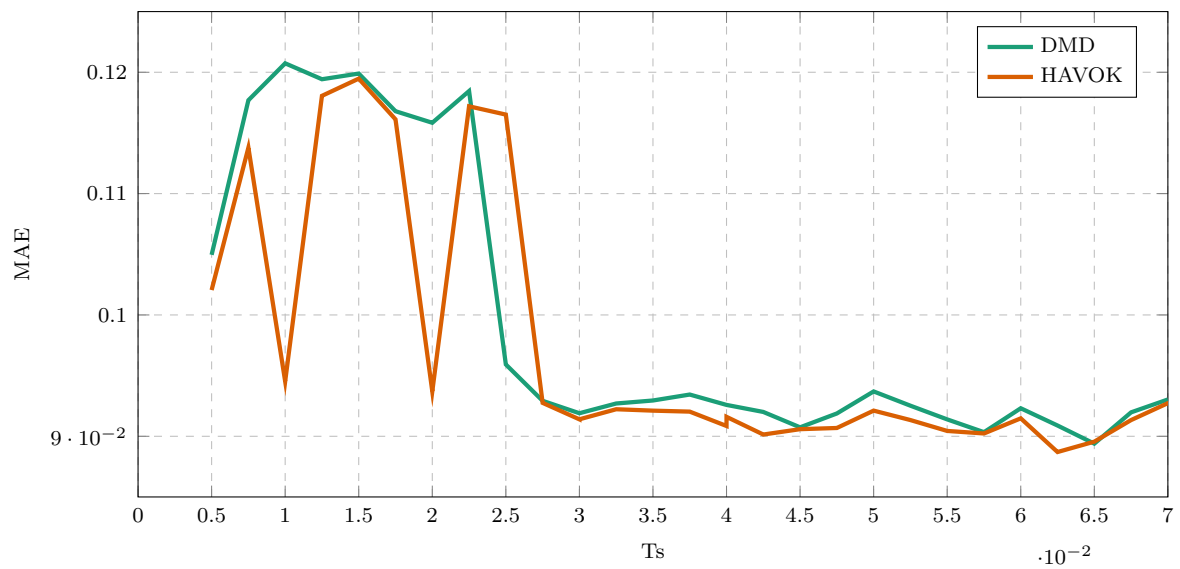
Another payload case that will cause inaccuracies in the parameter estimation technique is if a payload is attached rigidly to the quadrotor while it also has a suspended payload. The payload mass estimation is based on the assumption that the quadrotor mass is known. However if a mass is rigidly attached to the vehicle, the effective quadrotor mass is changed and the RLS payload mass estimation is no longer accurate.

These payload produce significantly different dynamics than predicted by the a priori model. Since the system identification by parameter estimation depends heavily on the pre-determined mathematical model,

For each of these payload cases, a different parameter estimation based techniques would need to be designed for effective control. This is undesirable for practical drone deliveries, especially when the type of payload is not known well in advance or changes regularly. A data-driven technique provides a more general solution since it accommodates a larger range of payload types and does not require a prior modelling information.

plot hyperparameters MAE. Not how much more delays are required





**Figuur 1.8:** DMD and HAVOK predictions error for different lengths of noisy training data ( $m = 0.2 \text{ kg}$ ,  $l = 0.5 \text{ m}$ ,  $T_s = 0.03 \text{ s}$ ,  $T_{train} = 60 \text{ s}$ .)

# Bibliografie

- [1] A. P. Erasmus and H. W. Jordaan, “Stabilization of a Rotary Wing Unmanned Aerial Vehicle with an Unknown Suspended Payload,” no. March, 2020.
- [2] J. F. Slabber and H. W. Jordaan, “Vision-Based Control of an Unknown Suspended Payload with a Multirotor Unmanned Aerial Vehicle,” Ph.D. dissertation, 2020.
- [3] J. H. Tu, C. W. Rowley, D. M. Luchtenburg, S. L. Brunton, and J. N. Kutz, “On dynamic mode decomposition: Theory and applications,” *Journal of Computational Dynamics*, vol. 1, no. 2, pp. 391–421, dec 2014. [Online]. Available: <https://www.aims sciences.org/article/doi/10.3934/jcd.2014.1.391>
- [4] J. L. Proctor, S. L. Brunton, and J. N. Kutz, “Dynamic mode decomposition with control,” *SIAM Journal on Applied Dynamical Systems*, vol. 15, no. 1, pp. 142–161, 2016.
- [5] M. Korda and I. Mezić, “Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control,” *Automatica*, vol. 93, pp. 149–160, nov 2018. [Online]. Available: <http://arxiv.org/abs/1611.03537><http://dx.doi.org/10.1016/j.automatica.2018.03.046>
- [6] H. Arbabi, M. Korda, and I. Mezic, “A Data-Driven Koopman Model Predictive Control Framework for Nonlinear Partial Differential Equations,” *Proceedings of the IEEE Conference on Decision and Control*, vol. 2018-Decem, pp. 6409–6414, 2018.
- [7] C. M. Chen and K. H. Wang, “State-space model conversion of a system with state delay,” *Proceedings of the National Science Council, Republic of China, Part A: Physical Science and Engineering*, vol. 23, no. 6, pp. 782–788, 1999.
- [8] B. R. Noack, W. Stankiewicz, M. Morzyński, and P. J. Schmid, “Recursive dynamic mode decomposition of transient and post-transient wake flows,” *Journal of Fluid Mechanics*, vol. 809, pp. 843–872, 2016.
- [9] S. L. Brunton, B. W. Brunton, J. L. Proctor, E. Kaiser, and J. Nathan Kutz, “Chaos as an intermittently forced linear system,” *Nature Communications*, vol. 8, no. 1, dec 2017.

# **Bylae A**

## **Project Planning Schedule**

This is an appendix.

# **Bylae B**

## **Outcomes Compliance**

This is another appendix.