Data-Driven Filtering for Large-Scale Adaptive Optics

Paulo Cerqueira

Master of Science Thesis

Delft University of Technology

Delft Center for Systems and Control

Data-Driven Filtering for Large-Scale Adaptive Optics

Master of Science Thesis

For the degree of Master of Science in Systems and Control at Delft University of Technology

Paulo Cerqueira

March 21, 2020

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of Technology

Delft University of Technology
Department of
Delft Center for Systems and Control (dcsc)

The undersigned hereby certify that they have read and recommend to the Faculty of Mechanical, Maritime and Materials Engineering (3mE) for acceptance of the thesis entitled

Data-Driven Filtering for Large-Scale Adaptive Optics

by

Paulo Cerqueira

in partial fulfillment of the requirements for the degree of

Master of Science Systems and Control

Dated: March 21, 2020

Supervisor(s): prof.dr.ir. Michel Verhaegen

ir. Pieter Piscaer

Reader(s): dr. Manon Kok

Abstract

The visualization of objects within or beyond a turbulent medium is hampered by the aberrations the medium induces in the wavefront. The sharpness of an image is maximal when the incoming wavefront is flat, with aberrated wavefronts yielding distorted images of limited utility. In the particular case of astronomy, the flat wavefront of the light from a distant target is aberrated as it moves through the atmosphere, before reaching our telescopes. The field of Adaptive Optics (AO) is dedicated to the correction of these aberrations, with the goal of enabling the imaging of objects through turbulent media with as much detail as possible.

Due to a delay inherent to the AO control loop, each control input is used to correct an aberration slightly ahead in time, which means that, for the correction to be effective, the controller must be equipped with predictive capabilities. Prediction demands a model, and fortunately, the dynamics of atmospheric turbulence can be effectively modelled using a linear system. This makes the Kalman filter, the statistically optimal state estimator for linear systems, a natural choice for prediction. However, the steady-state Kalman filter requires a solution to the computationally intensive Discrete-time Algebraic Riccati Equation (DARE), which precludes its application to large-scale systems. Furthermore, the Kalman filter assumes a precise model with particular properties is available, which often is not the case in practise.

The advent of extremely large telescopes demands prediction algorithms that can handle likewise extremely large system dimensions, which means that application of the steady-state Kalman filter, in its conventional form, is unfeasible. This thesis proposes a data-driven approach to Kalman filtering that exploits the intuitive sparsity pattern of the matrices involved in the modelling of an AO system. The developed algorithm replaces the DARE and, with knowledge of the system matrices, estimates the Kalman gain from measurement data, with the exploitation of sparsity providing a substantial drop in complexity when compared to the DARE, and with its data-driven nature inherently compensating, to a certain extent, for modelling errors.

This thesis further proposes a two-stage approach to reduce the burden of the online prediction operation. While the measurement and state-transition matrices of the AO system are sparse, the Kalman gain is generally dense, consequentially slowing down prediction, which should accompany the sampling period of the loop. Our proposal splits prediction into a sparse stage for prediction of local structures in the wavefront phase, and a low-dimensional dense stage for prediction of the remaining low-frequency aberrations. The two-stage predictor is inherently suboptimal, but allows for quicker prediction and identification for extremely large systems. Via tuning, the user strikes a balance between performance and execution time.

Table of Contents

Acknowledgements **xi**

1 Introduction **1**

2 Basics of Adaptive Optics **5**

2-1 The adaptive optics system 5

2-1-1 Shack-Hartmann sensor 6

2-1-2 Deformable mirror 8

2-2 Modelling 9

2-2-1 The dynamics of turbulence 9

2-2-2 AR-1 wavefront modelling 9

2-2-3 State-space and closed-loop descriptions 12

2-3 Sparse AR model identification 13

2-3-1 Wavefront phase model 14

2-3-2 Slope model 15

2-3-3 Setting distance constraints 16

3 Data-Driven Kalman Filtering **19**

3-1 Introduction 19

3-2 Single AR fitting 22

3-3 Multiple AR fitting 26

3-4 Concluding remarks 29

3-4-1 Reconstruction 30

4 Methods for Large-Scale Adaptive Optics **31**
4-1 Identification of the Kalman gain for large-scale AO 31
4-1-1 Removal of the mean 31
4-1-2 Sparse identification 33
4-1-3 Setting the distance constraint for the gain 35
4-2 Two-stage prediction 36
4-2-1 Formulation 37
4-2-2 Description of the coarse system 38
4-2-3 Summary 44

5 Results and Tuning **47**
5-1 Tuning the data-driven Kalman filtering algorithms 51
5-1-1 AR model order 51
5-1-2 Data batch length 55
5-1-3 Markov parameter distance-constraint 57
5-2 Prediction performance 58
5-2-1 Single-layer turbulence 59
5-2-2 Multi-layer turbulence 60
5-2-3 Increasing wind speed 60
5-2-4 Sparse and two-stage 62
5-3 Execution time 63
5-3-1 Identification 64
5-3-2 Prediction 65
5-4 Extremely large systems 65
5-5 Two-stage overview 67
5-6 Summary 68

6 Conclusions and Recommendations **71**
6-1 Recommendations 71

A On the modelling of turbulence **73**
A-1 Lagged AR-1 turbulence modelling 73
A-2 Iterative data-driven model refinement 77
A-3 Data-driven prediction for mismatched models 78

B On MARK **81**
B-1 Pre-filtered data 81
B-2 Weighted formulation 82

Bibliography **85**

Glossary **89**
List of Acronyms 89
List of Symbols 89

List of Figures

2-1 Schematic representation of an AO system 6
2-2 Shack-Hartmann sensor geometry and notation 7
2-3 Example G matrix 8
2-4 Example A for a wind speed of 0.25 pixels per time-step 14
2-5 Example distance-constraint in terms of lenslets 17
2-6 Exaggerated non-zero pattern of an AR-2 model of the slopes for a small-scale system 17

4-1 Identified Kalman gain without the mean removed 33
4-2 Identified Kalman gain with the mean removed 33
4-3 Example distance-constraint of the gain 35
4-4 Exaggerated non-zero pattern of the sparse gain for a small-scale system 36
4-5 Example sparse prediction vs. incoming wavefront phase 36
4-6 Example fine and corresponding coarse arrays 40

5-1 Example A for a wind speed of 0.5 pixels per time-step 50
5-2 Example A for a wind speed of 1 pixel per time-step 50
5-3 Data-driven Kalman filtering performance vs. AR order s (noiseless model) . . . 52
5-4 Data-driven Kalman filtering performance vs. AR order s (model SNR = 30 dB) 52
5-5 Data-driven Kalman filtering performance vs. AR order s (model SNR = 15 dB) 52
5-6 Performance of DARE-based and data-driven Kalman filtering vs. AR order and model quality 53
5-7 Data-driven Kalman filtering performance vs. AR order s (sub-pixel movement) . 54
5-8 Performance of DARE-based and data-driven Kalman filtering vs. AR order and model quality (sub-pixel movement) 54
5-9 Two-stage prediction performance vs. z and L_c $(s = 2)$ 55

5-10 Two-stage prediction performance vs. z and L_c $(s = 3)$ 55
5-11 Data-driven Kalman filtering performance vs. data batch length and distance-constraint of the Markov parameters 56
5-12 Data-driven Kalman filtering performance vs. data batch length and measurement SNR 57
5-13 Data-driven Kalman filtering performance vs. Markov parameter distance-constraint $(0.25$ lenslets per time-step$)$ 57
5-14 Data-driven Kalman filtering performance vs. Markov parameter distance-constraint $(0.75$ lenslets per time-step$)$ 57
5-15 Data-driven Kalman filtering performance vs. Markov parameter distance-constraint $(1.5$ lenslets per time-step$)$ 58
5-16 Data-driven Kalman filtering performance vs. Markov parameter distance-constraint $(2$ lenslets per time-step$)$ 58
5-17 Predictor performances: single-layer turbulence, SNR $= 10$ dB, noiseless model . 59
5-18 Predictor performances: single-layer turbulence, SNR $= 10$ dB, noisy model . . . 59
5-19 Predictor performances: single-layer turbulence, SNR $= 5$ dB, noiseless model . . 59
5-20 Predictor performances: single-layer turbulence, SNR $= 5$ dB, noisy model 59
5-21 Predictor performances: multi-layer turbulence, SNR $= 10$ dB, noiseless model . 60
5-22 Predictor performances vs. wind speed (measurement SNR = 10 dB) 61
5-23 Predictor performances vs. wind speed (measurement SNR = 5 dB) 61
5-24 Predictor performances vs. wind speed (multi-layer turbulence) 61
5-25 Predictor performance: sparse vs. two-stage prediction $(36 \times 36,$ noiseless model$)$ 62
5-26 Predictor performance: sparse vs. two-stage prediction $(36 \times 36,$ noisy model$)$ . . 63
5-27 Predictor performance: sparse vs. two-stage prediction $(120 \times 120,$ noiseless model$)$ 63
5-28 Execution time comparison between MATLAB's `idare`, and MARK and Juang for multiple values of s. 64
5-29 Execution time comparison between MARK and Juang for multiple values of s . 64
5-30 Prediction time comparison between MVM, full and sparse time-update, and the two-stage algorithm 65
5-31 Prediction time comparison between the sparse time-update and the two-stage algorithm 65
5-32 Two-stage performance vs. z and L_c $(120 \times 120, s = 2)$ 67
5-33 Two-stage performance vs. z and L_c $(120 \times 120, s = 3)$ 67
5-34 Two-stage performance vs. z and L_c $(150 \times 150, s = 2)$ 68
5-35 Two-stage performance vs. z and L_c $(150 \times 150,$ noisy model, $s = 1)$ 68

A-1 Iterative model refinement cycle 77
A-2 Iteratively refined model quality vs. iteration 77
A-3 Initial $\widehat{A}$ before iterative refinement 78
A-4 Final $\widehat{A}$ after iterative refinement 78
A-5 Side-by-side comparison between the final $\widehat{A}$ after refinement and the true A . . 78

A-6 Predictor performances: two-layered turbulence modelled with a single-layer model 79

A-7 Predictor performances: two-layered turbulence modelled with a single-layer model (DARE ignored) 79

B-1 Covariance of the unweighted MARK 84

B-2 Covariance of the weighted MARK 84

List of Tables

3-1 Time complexities per step of Juang and Chen's algorithm 26
3-2 Time complexities per step of MARK 29

4-1 Time complexities per step of two-stage prediction 38

5-1 Identification and prediction times for very large systems 66
5-2 Prediction performance for very large systems 66

A-1 Normalized Mean Squared Error (NMSE) comparison between the standard AR-1 and a 2-step lagged counterpart 76

Acknowledgements

I want to thank my supervisors prof.dr.ir. Michel Verhaegen and ir. Pieter Piscaer for the discussions and ideas that guided me throughout the process. Pieter's availability and expertise, whose thesis set the foundation for part of my own, proved especially invaluable.

I'd like to extend a particularly special set of thanks to my good friends Satyajith, Gabrielė, and Marta for keeping the company that helped me keep my sanity throughout. I'm not sure what would've been of me without you.

Delft, University of Technology
March 21, 2020

Paulo Cerqueira

Chapter 1

Introduction

Atmospheric turbulence proves itself an obstacle to the imaging of celestial objects. Originally flat wavefronts are aberrated as light travels through the atmosphere, resulting in an undesirable loss of detail that can not be circumvented via increases in the telescope diameter alone [1]. Adaptive Optics (AO) systems employ a deformable mirror to counteract the distortions of the incoming wavefront before it reaches the camera; a sensor acquires the necessary information for a controller to decide which shape the mirror should take.

The AO control loop has a one-step delay, which means that each control input is used to correct an aberration slightly ahead in time. Successful control must therefore account for the dynamics of turbulence throughout the delay, effectively predicting the shape of the wavefront one time-step ahead, demanding a model. The turbulence is herein modelled with an AR-1 model, and the AO loop is modelled with a state-space model whose outputs are the slopes measured by a Shack-Hartmann sensor, and states are each pixel of the wavefront phase measured at the corners of the lenslets of the sensor:

$$\begin{aligned}\phi_{k+1} &= A\phi_k + w_k \\ y_k &= G\phi_k + v_k\end{aligned}$$

where $\phi_k \in \mathbb{R}^n$ is the vectorized wavefront phase, $y_k \in \mathbb{R}^m$ comprises the slope measurements, and $w_k \sim \mathcal{N}(0, Q)$ and $v_k \sim \mathcal{N}(0, R)$ are, respectively, uncorrelated white process and measurement noises. In the context of AO, A and G are highly sparse, and consequentially, so are many other matrices that arise in equations that involve them.

The state-space model structure makes the Kalman filter an immediate choice for prediction, as it provides minimum-variance unbiased estimates of the state. The design of a steady-state Kalman filter requires a solution to the Discrete-time Algebraic Riccati Equation (DARE), which because the process and measurement noises are assumed uncorrelated, takes the form

$$P = APA^{\mathrm{T}} + Q - APG^{\mathrm{T}}(GPG^{\mathrm{T}} + R)^{-1}GPA^{\mathrm{T}} \tag{1-1}$$

where P is the steady-state state-prediction-error covariance matrix. Finding a solution to (1-1) takes $\mathrm{O}(n^3)$ time, which prevents its application to systems with extreme amounts of states. The Kalman filter additionally requires a precise linear model with the aforementioned noise properties to be optimal. Although effective [2, 3, 4], the Auto-Regressive (AR)-1 model carries inherent imperfections in case the turbulence is multi-layered or its movement is sub-pixel, unless special precautions, which may not always be applicable, are taken; furthermore, finding the matrices A, Q, and R for the model is a problem in itself, leading to additional possible modelling errors.

The dimensions n and m of an AO system whose sensor has a width of L lenslets are given by $(L+1)^2$ and $2L^2$, which means that not only do the dimensions grow quickly with the array size, but that complexities such as $\mathrm{O}(n^3)$ are effectively equivalent to $\mathrm{O}(L^6)$. Future extremely large telescopes [5, 6, 7] will thus pose a problem, and as the number of sensors reach the tens of thousands [2], more efficient approaches than the DARE are required for Kalman filtering. The introduction of Kalman filtering to these extremely large telescopes in approximate form is addressed in several works, among which [4, 8], with the former tackling the burden of the DARE, and the latter attacking both the DARE and the online prediction operation, whose time conventionally scales as $\mathrm{O}(nm)$. The sparse data-driven approach to Kalman filtering presented in this thesis tackles both the issues of computational burden and erroneous modelling. The DARE is replaced by a far lighter set of least-squares problems upon which sparsity is imposed to massively reduce complexity. Indeed, the aforementioned sparsity of the system matrices results in similarly sparse Markov parameters, the identification of which is one of the main steps of the evaluated data-driven Kalman filtering procedures; by identifying only the known non-zeros, the scale of the problem is drastically reduced, allowing obtention of the Kalman gain for extremely large systems, without the DARE. Additionally. the data-driven nature of the algorithms, which implicitly identify the process and measurement noise covariances, covers for modelling errors by accounting for them as extra noise [9].

As suggested above, there are works dedicated as well to the reduction of the complexity of the online reconstruction or prediction operations [8, 10, 11]. In Kalman filtering:

$$\hat{\phi}_{k+1} = A\hat{\phi}_k + K(y_k - G\hat{\phi}_k) \tag{1-2}$$

whereas A and G are sparse, K is normally dense, and the benefits of sparsity of A and G for prediction are entirely overshadowed by the $\mathrm{O}(nm)$ complexity of the matrix-vector product between K and $e_k = (y_k - G\hat{\phi}_k)$. A two-stage prediction algorithm is further proposed in this thesis to reduce the complexity of the prediction operation. This algorithm splits prediction into a sparse stage in the original large array, which employs a forcefully sparse gain in (1-2) to predict local structures in the wavefront phase. Because the Kalman gain is dense, the wavefront will not be entirely predicted; in particular, the low-frequency components will be missing. These are then predicted on with a low-dimensional system (a coarse stage), on which dense operations can be made with little computational burden. Two-stage prediction is inherently suboptimal, and while it will yield more error than a Kalman filter, it allows for quicker prediction and identification in extremely large systems, and reduces the memory requirements for storage of the gain. The tuning parameters decide how much performance is exchanged for time.

The thesis further describes a novel algorithm for data-driven Kalman filtering and several miscellaneous insights into the AR-1 modelling of turbulence. The thesis is organized as follows:

- Chapter 2 briefly presents the basics of AO necessary for the description of our techniques for identification and prediction. AR-1 modelling of the turbulence is described therein, along with several sources of error in that model structure. The final section of the chapter introduces sparsity to the modelling of turbulence, as described in [2].
- Chapter 3 overviews the field of data-driven Kalman filtering and elaborates upon the relevant algorithms, including a novel proposal.
- Chapter 4 combines the previous two chapters to introduce sparse data-driven Kalman filtering to AO. The two-stage algorithm is subsequently introduced and elaborated upon.
- Chapter 5 presents an analysis of the performance of the previously described algorithms on simulator-generated data. This chapter should both motivate the use of the algorithms and provide guidelines for their tuning.
- Chapter 6 concludes the main matter of the thesis with a summary of the contributions and proposals for further research.

The appendix provides additional off-topic insights and research proposals.

Chapter 2

Basics of Adaptive Optics

This chapter is dedicated to a bare-bones description of the Adaptive Optics (AO) system and turbulence modelling techniques for its control, in the context of astronomy. The content of the chapter encompasses only the essentials for the understanding the remainder of the thesis; for the underlying theory and details on the assumptions, the reader is referred to [12, 13, 14, 15, 16].

2-1 The adaptive optics system

The AO system, an example of which is depicted in Figure 2-1, comprises a wavefront sensor, a wavefront corrector, and a feedback controller [1]. The wavefront corrector is commonly a deformable mirror, onto which the incoming light shines before travelling to the wavefront sensor and the camera. The controller decides the shape the deformable mirror should take so as to nullify the wavefront aberrations before light is sent to the camera. Ideally, the mirror would take the inverse shape of the incoming wavefront, but inherent limitations in the system and its components mean the results will be inevitably suboptimal, and a residual wavefront will nevertheless reach the sensor and camera. The sensor then sends this information to the controller, which will use it to determine the shape of the deformable mirror for the next time-step.

Naturally, a one-step delay is implied in the circuit. The light received by the sensor, and therefore the information sent to the controller, pertain to the time-step previous to that when the deformable mirror applies the correction. This means that the controller must be equipped with predictive capability: given the current residual wavefront, the controller must predict which shape the incoming wavefront will take one time-step ahead, so that the deformable mirror can correct it. The lack of an exact model or noiseless measurements are among the reasons for which the correction can not be expected to be perfect. Section 2-2 ahead describes modelling techniques that, along with a Kalman filter, serve to predict the shape of the future wavefront.

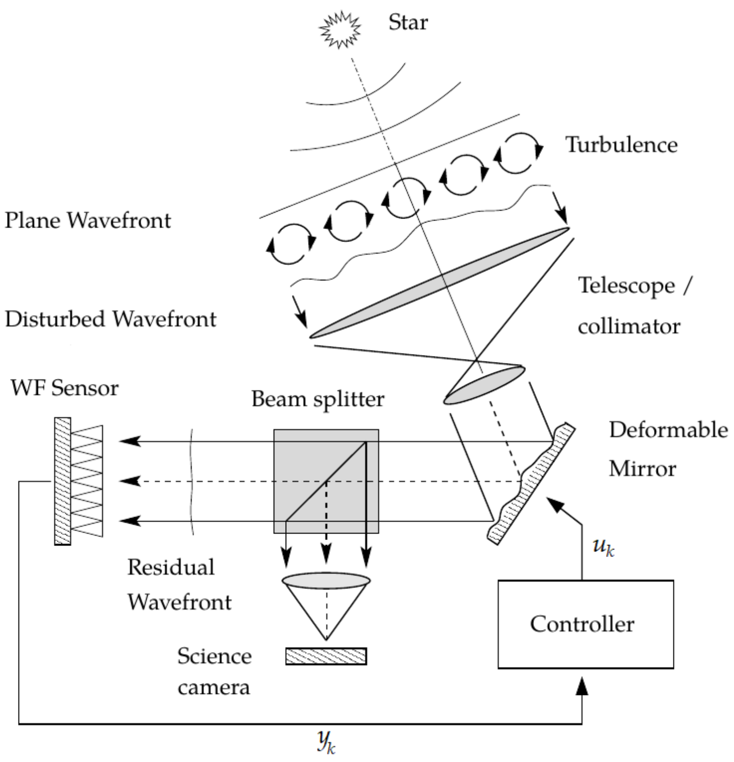

Figure 2-1: Schematic representation of an AO system. Image from [1], slightly altered to match notation.

2-1-1 Shack-Hartmann sensor

The wavefront sensor here considered is the Shack-Hartmann sensor, arranged with Fried geometry. The sensor consists of an $L \times L$ array of square lenslets, each focusing incoming light into a bright spot over a light sensor. The lenslets are assumed small enough such that the wavefront aberration can, locally over each lenslet, be approximated as a tip and/or a tilt (tip-tilt). The x- and y-displacements of the centroid of the bright spot with respect to the center of the light sensor are proportional, respectively, to the x- and y-slopes of the local tip-tilt. The measurement of these slopes is used to reconstruct the turbulent phase. Let i and j indicate the x- and y-positions, respectively, of a certain lenslet or wavefront sample point (pixel). Under the assumption of local tip-tilt, the slopes $y_\mathrm{x}^{(i,j)}$ and $y_\mathrm{y}^{(i,j)}$ measured by lenslet (i,j) are given by [1]:

$$\begin{aligned}\begin{bmatrix} y_\mathrm{x}^{(i,j)} \\ y_\mathrm{y}^{(i,j)} \end{bmatrix} &= \frac{1}{2}\begin{bmatrix} \varphi_{(i+1,j)} + \varphi_{(i+1,j+1)} - \varphi_{(i,j)} - \varphi_{(i,j+1)} \\ \varphi_{(i,j+1)} + \varphi_{(i+1,j+1)} - \varphi_{(i,j)} - \varphi_{(i+1,j)} \end{bmatrix} \\ &= \frac{1}{2}\begin{bmatrix} -1 & -1 & 1 & 1 \\ -1 & 1 & -1 & 1 \end{bmatrix}\begin{bmatrix} \varphi_{(i,j)} \\ \varphi_{(i,j+1)} \\ \varphi_{(i+1,j)} \\ \varphi_{(i+1,j+1)} \end{bmatrix}\end{aligned} \tag{2-1}$$

where $\varphi_{(i,j)}$ to $\varphi_{(i+1,j+1)}$ denote the values of the phase at the corners of the lenslet. With the slopes from all L^2 lenslets stacked into vector y, and the values of the wavefront phase at

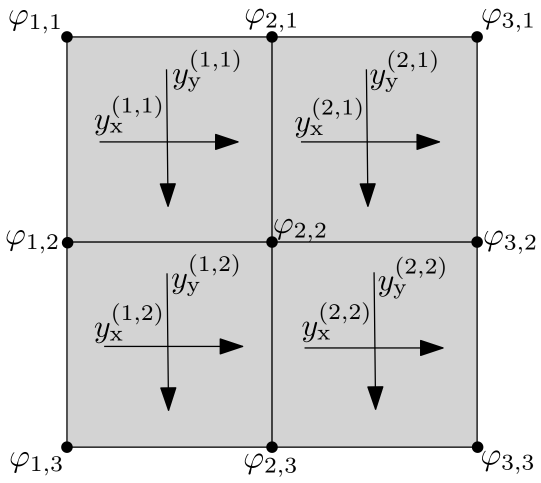

Figure 2-2: Shack-Hartmann sensor geometry and notation, exemplified with a 2×2 lenslet array. The black dots represent the pixels at which the phase is sampled, and the gray squares represent the lenslets.

all $(L+1)^2$ pixels stacked into vector ϕ, one arrives at the relationship:

$$\underbrace{\begin{bmatrix} y_{\mathrm{x}}^{(1,1)} \\ y_{\mathrm{y}}^{(1,1)} \\ \hdashline y_{\mathrm{x}}^{(1,2)} \\ y_{\mathrm{y}}^{(1,2)} \\ \hdashline \vdots \\ \hdashline y_{\mathrm{x}}^{(L,L)} \\ y_{\mathrm{y}}^{(L,L)} \end{bmatrix}}_{y} = G \underbrace{\begin{bmatrix} \varphi_{(1,1)} \\ \varphi_{(1,2)} \\ \vdots \\ \varphi_{(L+1,L)} \\ \varphi_{(L+1,L+1)} \end{bmatrix}}_{\phi}$$

where $G \in \mathbb{R}^{2L^2 \times (L+1)^2}$ is the measurement matrix, built from (2-1). An example G is shown in Figure 2-3. Note that G does not have full column rank: two modes, piston and waffle, are not detected by the sensor. The piston mode corresponds to an offset in the wavefront; from (2-1), considering an arbitrary offset a:

$$\begin{bmatrix} y_{\mathrm{x}}^{(i,j)} \\ y_{\mathrm{y}}^{(i,j)} \end{bmatrix} = \frac{1}{2} \begin{bmatrix} -1 & -1 & 1 & 1 \\ -1 & 1 & -1 & 1 \end{bmatrix} \begin{bmatrix} a \\ a \\ a \\ a \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

In the absence of other aberrations, an offset wavefront is still flat, and the quality of the image is unaffected, making piston mode irrelevant. The waffle mode corresponds to a checkerboard pattern in the wavefront:

$$\begin{bmatrix} y_{\mathrm{x}}^{(i,j)} \\ y_{\mathrm{y}}^{(i,j)} \end{bmatrix} = \frac{1}{2} \begin{bmatrix} -1 & -1 & 1 & 1 \\ -1 & 1 & -1 & 1 \end{bmatrix} \begin{bmatrix} a \\ -a \\ -a \\ a \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Although undetectable in a single time-step of measurements, unlike piston mode, the waffle mode does affect image quality. It is, however, of little significance, given the relatively low power of high-frequency modes [2]. Throughout the remainder of the thesis, the subscript k in y_k or ϕ_k and whatever numerical value it takes indicates the time-step (note that the subscripts x and y beneath y still specify the axis). Considering additive measurement noise v_k, one finally arrives at:

$$y_k = G\phi_k + v_k \tag{2-2}$$

In the experimental chapter (Chapter 5), computation of the slopes for simulation is done by simply applying (2-2) to the simulated wavefront phase screens, with added Gaussian white measurement noise of diagonal covariance. This enables control of the Signal-to-Noise Ratio (SNR) of the measurement noise directly, albeit neglecting the approximate nature of (2-1), as a real wavefront will not be exactly tip-tilt over each lenslet.

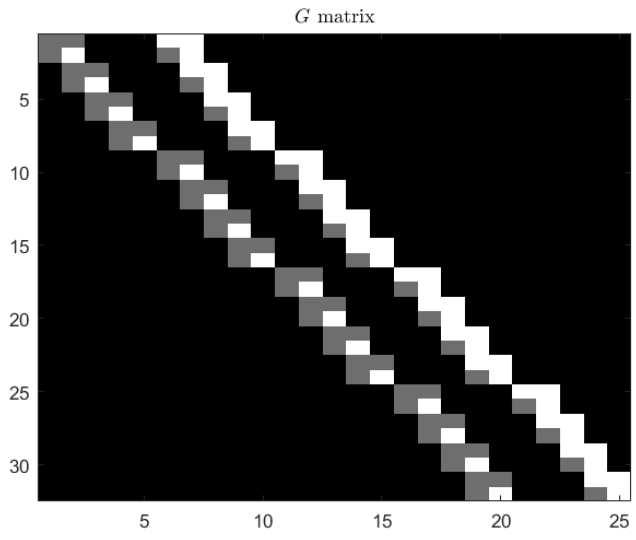

Figure 2-3: Example G matrix. White entries equal 0.5, gray entries equal -0.5, and black entries are 0.

2-1-2 Deformable mirror

The deformable mirror, used for correction of the aberrations of the incoming wavefront, consists of a malleable, or otherwise deformable, reflective surface and a system of actuators that set its shape. Given an input u_k, the shape ϕ^{m}_{k+1} of the mirror at the next time-step can be given by:

$$\phi^{\mathrm{m}}_{k+1} = Hu_k \tag{2-3}$$

where the columns of H represent the influence function of each input, sampled at the location of each actuator. The range of H defines a basis for the set of possible shapes the mirror can take [2].

2-2 Modelling

As mentioned in the beginning of the section, there is an inevitable one time-step delay between the computation of the control input and the correction by the deformable mirror. This makes one-step-ahead prediction of the wavefront phase vital for control, as the aberrated wavefront will have changed in-between time-steps. The modelling of the turbulence dynamics, on which prediction relies, is the subject of this section.

2-2-1 The dynamics of turbulence

The evolution of the turbulent phase at a fixed point in space is the result of punctual changes in the phase screen, and of its "mean" motion as it is carried by an air stream. According to Taylor's frozen flow hypothesis [16], when the rate of change of the phase screen is insignificant in comparison to the velocity of the air stream, i.e the wind speed, the evolution of the phase at the fixed point is well justified by the movement of a *frozen* phase screen, drifting over the aperture. Put simply, future wavefront phases are given by mere spatial shifts of past phases. Given a wind speed of ν_{x} along the x-axis and of ν_{y} along the y-axis, the phase at point (x, y) and time $t + \tau$ is given by [2]:

$$\varphi(x, y, t+\tau) = \varphi(x - \nu_{\mathrm{x}}\tau,\, y - \nu_{\mathrm{y}}\tau,\, t) \tag{2-4}$$

When modelling these dynamics with an AR-1 model, frozen flow gives rise to distinct structures in the transition matrix A. In particular, when the wavefront moves an integer amount of pixels each time-step, A represents a shift in the direction of the movement, except at the entries corresponding to points on the border of the pixel array where new infomation enters (see Figure 5-2). In any other case (and again, except at the array borders), A becomes a weighted average between shifts, a structure henceforth referred to as "shift-like", and seen in Figures 2-4 and 5-1.

2-2-2 AR-1 wavefront modelling

This thesis adopts the AR-1 structure to model the turbulence dynamics, a popular model due to its simplicity and efficacy [2, 3, 4]. Befitting the goal of Kalman filtering, this model structure is particularly apt for the definition of a state-space model whose outputs are the slopes measured by the Shack-Hartmann sensor and whose state is the wavefront phase screen. The AR-1 turbulence model is given by:

$$\phi_{k+1} = A\phi_k + w_k \tag{2-5}$$

where ϕ_k is the vectorized wavefront phase screen, and w_k is white process noise, uncorrelated with ϕ_k. It is assumed that $\mathrm{E}[\phi_k] = \mathrm{E}[w_k] = 0$. While it is common for authors to preset A, often as diagonal [4, 17, 18], the data-driven approach of [2] is adopted herein. When the system is in steady-state, the state-transition matrix A is given by:

$$A = C_{\phi,1}C_{\phi,0}^{-1} \tag{2-6}$$

If wavefront phase data up to time-step N is gathered into "future"- and "past"-data matrices defined as:

$$\Phi_{1,N} = \begin{bmatrix} \phi_2 & \cdots & \phi_N \end{bmatrix} \qquad\qquad \Phi_{0,N} = \begin{bmatrix} \phi_1 & \cdots & \phi_{N-1} \end{bmatrix}$$

then (2-6), with estimated covariances due to the finiteness of data, is equivalent to

$$\boxed{\widehat{A} = \Phi_{1,N} \left(\Phi_{0,N}\right)^{\dagger}}$$

Put differently, $\widehat{A}$ is the solution to

$$\boxed{\widehat{A} = \underset{A}{\operatorname{argmin}} \sum_{k=1}^{N} \left\| \phi_{k+1} - A\phi_k \right\|_2^2 = \underset{A}{\operatorname{argmin}} \left\| \Phi_{1,N} - A\Phi_{0,N} \right\|_{\mathrm{F}}^2} \tag{2-7}$$

However, the assumptions behind (2-5) do not always hold, and in such cases, a model so identified might be found wanting. This is the case for multi-layer turbulence, possibly warranting a change of state variable, and for sub-pixel movements per time-step, in which case a possible modelling alternative is passingly proposed in Appendix A-1.

Modelling error for multi-layer turbulence

For multi-layer turbulence, the process noise will span the entire array, while neccessarily breaking one of the previous assumptions: w_k *will* not be white. One implication is that such a model is subpar for application of a Kalman filter. Consider that the dynamics of layer i are

$$\phi_{k+1}^{(i)} = A^{(i)}\phi_k^{(i)} + w_k^{(i)}$$

where $w^{(i)}$ is assumed to be indeed white and uncorrelated with $\phi_k^{(i)}$ and all other $\phi_k^{(j)}$. Then, for example, the dynamics of two-layer turbulence are given by

$$\phi_{k+1}^{(1)} + \phi_{k+1}^{(2)} = A^{(1)}\phi_k^{(1)} + A^{(2)}\phi_k^{(2)} + w_k^{(1)} + w_k^{(2)} \tag{2-8}$$

Imagine now that the turbulence is modelled as

$$\underbrace{\phi_{k+1}^{(1)} + \phi_{k+1}^{(2)}}_{\phi_{k+1}} = A\,\underbrace{\left(\phi_k^{(1)} + \phi_k^{(2)}\right)}_{\phi_k} + w_k \tag{2-9}$$

then, substituting (2-8) into the left-hand-side of (2-9):

$$\begin{aligned} w_k &= A^{(1)}\phi_k^{(1)} + A^{(2)}\phi_k^{(2)} - A\Big(\phi_k^{(1)} + \phi_k^{(2)}\Big) + w_k^{(1)} + w_k^{(2)} \\ &= \underbrace{\left(A^{(1)} - A\right)}_{\neq 0}\phi_k^{(1)} + \underbrace{\left(A^{(2)} - A\right)}_{\neq 0}\phi_k^{(2)} + w_k^{(1)} + w_k^{(2)} \end{aligned} \tag{2-10}$$

Finally, with A taken from (2-6) and noting that $C_{\phi,0} = C_{\phi,0}^{(1)} + C_{\phi,0}^{(2)}$, and $C_{\phi,1} = C_{\phi,1}^{(1)} + C_{\phi,1}^{(2)}$, then,

$$\begin{aligned} \mathrm{E}[w_k \phi_k^{\mathrm{T}}] &= (A^{(1)} - A)C_{\phi,0}^{(1)} + (A^{(2)} - A)C_{\phi,0}^{(2)} \\ &= C_{\phi,1}^{(1)} + C_{\phi,1}^{(2)} - C_{\phi,1} = 0 \end{aligned}$$

and,

$$\mathrm{E}[w_{k+j} w_k^{\mathrm{T}}] = (A^{(1)} - A)C_{\phi,j}^{(1)}(A^{(1)} - A)^{\mathrm{T}} + (A^{(2)} - A)C_{\phi,j}^{(2)}(A^{(2)} - A)^{\mathrm{T}} \neq 0$$

hence breaking the key assumption of whiteness of the process noise in (2-5). Kalman filters, whose design assumes white process noise, designed for these models will be suboptimal, which can be verified in their colored prediction-error sequences. This simple example can easily be generalized to an arbitrary number of layers.

Remark 2.1. Note that w_k and ϕ_k remain uncorrelated even if the phases at each layer are cross-correlated, just as long as each $w^{(i)}$ is truly uncorrelated with $\phi_k^{(i)}$.

Rather, for AR-1 modelling in the multi-layer case, each layer should be separated within the state. For example, for q layers of turbulence, the following model

$$\begin{aligned} \begin{bmatrix} \phi_{k+1}^{(1)} \\ \vdots \\ \phi_{k+1}^{(q)} \end{bmatrix} &= \begin{bmatrix} A^{(1)} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & A^{(q)} \end{bmatrix} \begin{bmatrix} \phi_k^{(1)} \\ \vdots \\ \phi_k^{(q)} \end{bmatrix} + \begin{bmatrix} w_k^{(1)} \\ \vdots \\ w_k^{(q)} \end{bmatrix} \\ y_k &= \begin{bmatrix} G & \cdots & G \end{bmatrix} \begin{bmatrix} \phi_k^{(1)} \\ \vdots \\ \phi_k^{(q)} \end{bmatrix} \end{aligned} \tag{2-11}$$

would indeed be such that the process noise is white and remains uncorrelated with the state as long as each $w_k^{(i)}$ is uncorrelated with all $\phi_k^{(j)}$. However, identification of the dynamics of this model would require *individual* knowledge of each of the layers $\phi_k^{(1)}$ to $\phi_k^{(q)}$. In Chapter 5, the erroneous model in (2-5) will be used for multi-layer turbulence, implying an underlying assumption that the only available wavefront phase data has all the layers added up, which is the case if the wavefront phases are drawn from the slopes.

Modelling error for sub-pixel movement

Consider single-layer turbulence. Let the wavefront phase move one pixel per time-step in a hypothetical dense grid, and indicate variables in this dense grid by a superscript (d):

$$\phi_{k+1}^{(\mathrm{d})} = A^{(\mathrm{d})}\phi_k^{(\mathrm{d})} + w_k^{(\mathrm{d})}$$

where $w_k^{(\mathrm{d})}$ is assumed white and uncorrelated with $\phi_k^{(\mathrm{d})}$. Consider a coarser "true" grid with which the wavefront is sampled. In this true grid, the wavefront moves a fraction of a pixel each time-step. Define Z as the selection matrix of ones and zeros such that

$$\phi_k = Z\phi_k^{(\mathrm{d})}$$

where ϕ_k represents the wavefront phase in the true grid. Then,

$$\phi_{k+1} = ZA^{(\mathrm{d})}\phi_k^{(\mathrm{d})} + Zw_k^{(\mathrm{d})} \tag{2-12}$$

Say, again, that the turbulence is modelled as

$$\phi_{k+1} = A\phi_k + w_k \tag{2-13}$$

Comparing (2-12) and (2-13), yields:

$$w_k = \underbrace{\left(ZA^{(\mathrm{d})} - AZ\right)}_{\neq 0}\phi_k^{(\mathrm{d})} + Zw_k^{(\mathrm{d})} \tag{2-14}$$

From (2-6), one has that

$$\begin{aligned} A &= ZC_{\phi,1}^{(\mathrm{d})}Z^{\mathrm{T}}(ZC_{\phi,0}^{(\mathrm{d})}Z^{\mathrm{T}})^{-1} \\ &= ZA^{(\mathrm{d})}C_{\phi,0}^{(\mathrm{d})}Z^{\mathrm{T}}(ZC_{\phi,0}^{(\mathrm{d})}Z^{\mathrm{T}})^{-1} \end{aligned} \tag{2-15}$$

Finally, again assuming $w_k^{(\mathrm{d})}$ is white and uncorrelated with $\phi_k^{(\mathrm{d})}$,

$$\begin{aligned} \mathrm{E}[w_k\phi_k^{\mathrm{T}}] &= (ZA^{(\mathrm{d})} - AZ)\,\mathrm{E}[\phi_k^{(\mathrm{d})}\phi_k^{\mathrm{T}}] \\ &= (ZA^{(\mathrm{d})} - AZ)C_{\phi,0}^{(\mathrm{d})}Z^{\mathrm{T}} \\ &= ZA^{(\mathrm{d})}C_{\phi,0}^{(\mathrm{d})}Z^{\mathrm{T}} - ZA^{(\mathrm{d})}C_{\phi,0}^{(\mathrm{d})}Z^{\mathrm{T}} = 0 \end{aligned}$$

and,

$$\mathrm{E}[w_{k+j}w_k^{\mathrm{T}}] = (ZA^{(\mathrm{d})} - AZ)\,C_{\phi,j}^{(\mathrm{d})}\,(ZA^{(\mathrm{d})} - AZ)^{\mathrm{T}} \neq 0$$

Meaning that w_k is colored. Regardless, this model is used in Chapter 5. In Appendix A-1, we propose the use of lagged AR-1 models as a potential solution for slower cases: imagine, for example, that the wavefront moves 0.5 pixels per time-step; this is equivalent to 1 pixel every two time-steps, therefore making the dynamics of a model of lag 2 represent whole-pixel movement that can be accurately modelled without (or with minimally) colored process noise.

2-2-3 State-space and closed-loop descriptions

The AO system, as per Figure 2-1, is a closed-loop control system whose sensor measures the slopes of the *residual* wavefront, and feeds them to the controller, which then shapes the deformable mirror accordingly. As such, in the closed AO loop, (2-2) becomes instead:

$$y_k = G\varepsilon_k + v_k \tag{2-16}$$

where $\varepsilon_k = \phi_k - \phi_k^{\mathrm{m}}$ is the residual wavefront, resulting from incomplete correction of the incoming aberration by the deformable mirror. Combining (2-3), (2-5), and (2-16) yields the closed-loop system:

$$\begin{aligned} \varepsilon_{k+1} &= A\varepsilon_k - Hu_k + AHu_{k-1} + w_k \\ y_k &= G\varepsilon_k + v_k \end{aligned} \tag{2-17}$$

Notice how the stochastic part of the state update equation of (2-17), which is the target of the Kalman filter (see Remark 3.1), is the same as that of the incoming turbulence, as in (2-5). Indeed, minimization of the 2-norm of a prediction of ε_{k+1} is equivalent to the minimization of the 2-norm of a prediction of $\phi_{k+1} - Hu_k$. Because the goal of this thesis pertains to prediction, then rather than using (2-17), we will ignore the deformable mirror, and instead focus only on the turbulence itself, with the model:

$$\begin{aligned} \phi_{k+1} &= A\phi_k + w_k \\ y_k &= G\phi_k + v_k \end{aligned} \tag{2-18}$$

knowing that a Kalman filter designed for (2-18) is equivalent to one designed for (2-17), minus the input term. Let $n = (L+1)^2$ denote the number of states and $m = 2L^2$ denote the number of outputs throughout the remainder of the thesis; then, $\phi_k \in \mathbb{R}^n$, and $y_k \in \mathbb{R}^m$. Both of these scale as $\mathrm{O}(L^2)$, and $\mathrm{O}(n)$ is equivalent to $\mathrm{O}(m)$; hence, in Chapter 5, when scaling and complexities are mentioned, these are given in terms of the outputs only.

2-3 Sparse AR model identification

Recall that, under the frozen flow hypothesis, the temporal evolution of the phase is explained by the drift of a frozen pattern over the aperture [16]. Say the wind moves with a speed of $\nu = \sqrt{\nu_\mathrm{x}^2 + \nu_\mathrm{y}^2}$ m/s and the sampling period is τ. Then, (2-4), rewritten here:

$$\varphi(x, y, t+\tau) = \varphi(x - \nu_\mathrm{x}\tau,\ y - \nu_\mathrm{y}\tau,\ t)$$

establishes that, even without knowledge of the wind direction, as long as the wind speed is over-estimated as ν, it is known as well that information travels only within a radius of $r = \tau\nu$ meters around a certain point of origin in between time-steps [2]. Indeed, (2-4) gives rise to the sparse "shift-like" structure of the state transition matrix A of the AR-1 turbulence model, seen in Figure 2-4.

Moreover, not only is the AR-1 model for the phase sparse, but so is an AR model for the slopes. Naturally, however, when considering the measured slopes, one has to also account for the measurement noise, which is smoothed out with data from previous time-steps, generally demanding orders over 1, and with data from adjacent measurements, forcing a radius larger than $r = \tau\nu$. Let the slopes be modelled with an AR model of order s:

$$y_k = \mathcal{A}_1 y_{k-1} + \mathcal{A}_2 y_{k-2} + \ldots + \mathcal{A}_s y_{k-s}$$

Given a base distance-constraint of r meters imposed upon $\mathcal{A}_1$, the other coefficients $\mathcal{A}_i$ should have a radius of $r_i = i \cdot r$ meters, reflecting motion across i time-steps. If one considers the wind speed in meters per second, the sparsity pattern naturally depends as well on the inter-lenslet or (inter-pixel) distances. As established above, only an over-estimate of the wind-speed is necessary, and the inter-lenslet distances are available. However, in truth, the sparsity pattern, known *á priori*, depends solely on the over-estimate of the wind speed if it is given in lenslets or pixels per time-step (both of which should be almost the same, and will not be distinguished). Let the wind speed given as such be denoted by ω. Then, Δk time-steps apart, there is only need to consider inter-lenslet (resp. inter-pixel) interactions for lenslets (resp. pixels) separated by less than $\omega\Delta k$ lenslets (resp. pixels).

Exploitation of this sparsity pattern in identification procedures massively reduces the amount of parameters to identify, decreasing the necessary amount of data, execution time, and storage requirements, hence enabling application to extremely large-scale systems. Identification of such sparse models is dealt with in this section.

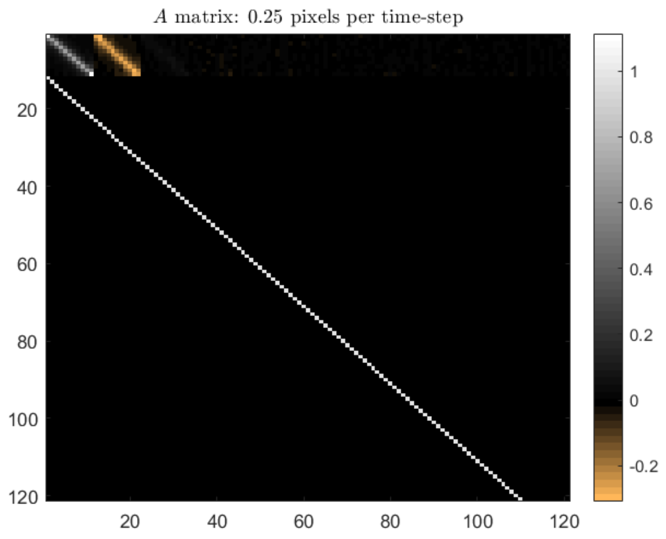

Figure 2-4: Example A, identified as full, for a wind speed of 0.25 pixels per time-step.

2-3-1 Wavefront phase model

Return to the AR-1 turbulence model (2-5):

$$\phi_{k+1} = A\phi_k + w_k$$

Frozen flow is reflected in the shift-like structure of A, with the process noise accounting for the entrance of new data at the edges. Arranging the wavefront phase data, available from time-step 0 to N, into the following matrices:

$$\Phi_{1,N} = \begin{bmatrix} \phi_1 & \cdots & \phi_N \end{bmatrix} \qquad \Phi_{0,N} = \begin{bmatrix} \phi_0 & \cdots & \phi_{N-1} \end{bmatrix}$$

then, a simple procedure for retrieving an estimate of A is given in (2-7).

$$\widehat{A} = \underset{A}{\operatorname{argmin}} \left\| \Phi_{1,N} - A\Phi_{0,N} \right\|_{\mathrm{F}}^2$$

This procedure identifies a full matrix with a time complexity of $\mathrm{O}(Nn^2 + n^3)$ (assuming the RQ-factorization is used), but as argued above, given an over-estimate of the wind speed of ω pixels per time-step, it is known that entries of A that establish relationships between pixels separated by more than ω pixels should be zero.

Now, the least-squares problem in (2-7) consists, in fact, of n independent least-squares problems corresponding to each row of $\widehat{A}$. Let idx denote the non-zero indices of the j^{th} row of A (and $\widehat{A}$), and the superscript $(\star, \star)$ indicate the row and column indices of a matrix in

MATLAB notation. With the sparsity pattern established, a set of n least-squares problems, one for **only** the non-zeros of each row of $\widehat{A}$ can be solved:

$$\boxed{\widehat{A}^{(j,idx)} = \underset{A^{(j,idx)}}{\operatorname{argmin}} \left\| \Phi_{1,N}^{(j,:)} - A^{(j,idx)} \Phi_{0,N}^{(idx,:)} \right\|_2^2} \tag{2-19}$$

Each row must be solved for independently because the regressor $\Phi_{0,N}^{(idx,:)}$ is different for each row, given the different sets of non-zero indices idx. For an average number q of non-zeros per row of A, the time complexity of (2-19) is given by:

$$n \text{ times} \begin{cases} \mathrm{O}(Nq^2) & \text{for the RQ-factorization} \\ \mathrm{O}(Nq) & \text{to form } Y_{s,N}^{(j,:)} Q^{\mathrm{T}} \\ \mathrm{O}(q^2) & \text{for the back-substitutions} \end{cases}$$

for a total of $\mathrm{O}(Nnq^2)$ operations. Note also that since the least-squares problems for each of the rows are all independent, these can be solved in parallel.

With (2-19), one has obtained an estimate of A with sparsity imposed. The matrix A is expected to be very highly sparse (see Figure 2-4), and so (2-19), compared to (2-7), brings higher estimate quality, due to the reduced amount parameters to identify, and consequently the data required; shortened prediction times; and reduced time and memory complexities.

2-3-2 Slope model

The output of asymptotically stable models of the form (2-18) can be represented by an AR model of adequate order. Consider the time-update of the Kalman filter designed for (2-18):

$$\begin{aligned} \hat{\phi}_{k+1} &= (A - KG)\hat{\phi}_k + Ky_k \\ y_k &= G\hat{\phi}_k + e_k \end{aligned} \tag{2-20}$$

The output (slopes) can be written as

$$\begin{aligned} y_k = GKy_{k-1} + G(A - KG)Ky_{k-1} + G(A - KG)^2 Ky_{k-2} + \ldots + \\ + G(A - KG)^k \hat{\phi}_0 + e_k \end{aligned} \tag{2-21}$$

Given asymptotic stability of the Kalman filter, then for some s, for all $j \geq s$, it holds that $(A - KG)^j \approx 0$. Cutting off all such terms from (2-21) yields a finite-order AR model for the outputs:

$$y_k = \mathcal{A}_1 y_{k-1} + \mathcal{A}_2 y_{k-2} + \ldots + \mathcal{A}_s y_{k-s} \tag{2-22}$$

where $\mathcal{A}_i = G(A - KG)^{i-1}K$ are the Markov parameters of the observer-form Kalman filter 2-20. Using the same reasoning as for the wavefront phase, one can establish a similar sparsity pattern, with a caveat: the slope measurements are affected by noise, and a successful model of form (2-22) will smoothen the noise at each lenslet by using information not only from previous time-steps, but also from nearby lenslets. This use of information from adjacent lenslets must be accommodated by a choice of distance-constraint that exceeds the wind speed (in lenslets per time-step). Arrange the data into matrices

$$Y_{i,j,N} = \begin{bmatrix} y_i & y_{i+1} & y_{i+2} & \cdots & y_{i+N-1} \\ y_{i+1} & y_{i+2} & & \cdots & y_{i+N} \\ y_{i+2} & & & & \\ \vdots & & & & \vdots \\ y_{i+j-1} & y_{i+j} & & \cdots & y_{i+j+N-2} \end{bmatrix}$$
$$Y_{i,N} = \begin{bmatrix} y_i & y_{i+1} & y_{i+2} & \cdots & y_{i+N-1} \end{bmatrix}$$

and define

$$\mathcal{A} = \begin{bmatrix} \mathcal{A}_s & \mathcal{A}_{s-1} & \cdots & \mathcal{A}_1 \end{bmatrix}$$

Normally, without exploiting sparsity, from (2-22), $\mathcal{A}$ would be obtained from the following least-squares problem:

$$\widehat{\mathcal{A}} = \underset{\mathcal{A}}{\operatorname{argmin}} \left\| Y_{s,N} - \mathcal{A} Y_{0,s,N} \right\|_{\mathrm{F}}^2 = Y_{s,N} Y_{0,s,N}^{\dagger} \tag{2-23}$$

where the superscript $\dagger$ indicates the appropriate pseudo-inverse. Problem (2-23) takes $\mathrm{O}(Ns^2m^2+s^2m^3)$ time (assuming the RQ-factorization is used) [19]. Similarly to the previous subsection, (2-23) consists of m independent least-squares problem, each corresponding to a row of $\widehat{\mathcal{A}}$. Rather than solving for a full $\mathcal{A}$, then, introduce the sparsity pattern [2]:

$$\boxed{\widehat{\mathcal{A}}^{(j,idx)} = \underset{\mathcal{A}^{(j,idx)}}{\operatorname{argmin}} \left\| Y_{s,N}^{(j,:)} - \mathcal{A}^{(j,idx)} Y_{0,s,N}^{(idx,:)} \right\|_2^2} \tag{2-24}$$

Given the very high sparsity of $\mathcal{A}$, (2-24) is much faster than (2-23) for large systems, while requiring far less data as well. Indeed, for an average of q non-zero elements per row, the computational complexity of an implementation of (2-24) is given by

$$m \text{ times} \begin{cases} \mathrm{O}(Nq^2) & \text{for the RQ-factorization} \\ \mathrm{O}(Nq) & \text{to form } Y_{s,N}^{(j,:)} Q^{\mathrm{T}} \\ \mathrm{O}(q^2) & \text{for the back-substitutions} \end{cases}$$

For a total of $\mathrm{O}(Nmq^2)$ operations, scaling linearly with m for a fixed average number q of non-zeros. The m independent problems in (2-23) or (2-24) can all be solved in parallel as well, further dividing the complexity per processor. The AR modelling of the slopes arises naturally in prediction-error data-driven Kalman filtering, and this sparse implementation is key to enable the implementation of data-driven filtering to large-scale AO. This sparse implementation increases the estimate quality due to the reduced necessary amount of data, and reduces both the time and memory complexity of the operations.

2-3-3 Setting distance constraints

The base distance-constraint for A or the Markov parameters is straightforward: it is given lenslets (for the Markov parameters) in pixels (in the case of A), and a distance of zero forces a lenslet (resp. pixel) to influence only itself, while a distance of ω sets a radius of influence of ω lenslets (resp. pixels). Note that fractional distances are possible (for example, diagonally adjacent lenslets are separated by a distance of $\sqrt{2}$ lenslets). Figure 2-5 shows an example distance-constraint applied to a lenslet; the application to pixels is analogous.

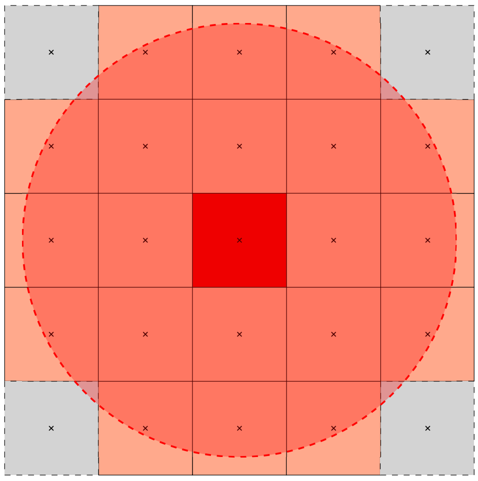

Figure 2-5: Example distance-constraint in terms of lenslets. The distance is here selected as slightly above 2. The bright red square represents the lenslet whose interactions are to be constrained, the red circle depicts the distance around the lenslet, and the pale orange squares represent the lenslets that fulfill the distance-constraint. The crosses mark the center of each lenslet.

The discrete nature of lenslets and pixels also implies that meaningful distance-constraints are discrete as well; that is, for example, all distance-constraints within the interval $[\sqrt{2}, 2[$ result in the *exact same* sparsity pattern, as do any within $[2, \sqrt{5}[$. When mentioning the chosen distance-constraints in Chapter 5, then, peculiar numbers will be avoided: for example, rather than saying that a constraint of $\sqrt{2}$ was chosen, 1.5 will be stated instead.

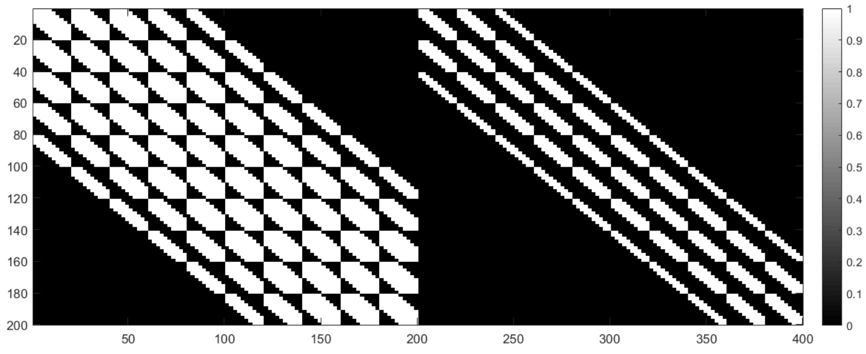

Figure 2-6: Exaggerated non-zero pattern of an AR-2 model of the slopes for a small-scale system. White entries are non-zeros, black entries are zero. The second (leftmost) Markov parameter has a constraint of double the radius of that of the first (rightmost) Markov parameter imposed. A third Markov parameter would triple it, and so forth.

Chapter 3

Data-Driven Kalman Filtering

Underlying the modelling of Adaptive Optics (AO) systems for control and estimation purposes are multiple assumptions regarding the atmospheric turbulence, beyond the frozen flow assumption; parameters such as the wind speed and direction, the number of turbulence layers, the covariance of the measurement noise, among others, must be set to determine the model for which a Kalman filter is designed. Data-driven identification of the Kalman gain is thus motivated:

- Determines the steady-state Kalman gain from data, implicitly determining the stochastic parameters, and additionally accounting for possible modelling errors by covering them as fictitious process noise.
- Allows direct determination of the steady-state Kalman gain, avoiding the computationally intensive Discrete-time Algebraic Riccati Equation (DARE), while providing a setting for the exploitation of sparsity of the system matrices for great computational benefits.
- With a dedicated implementation, enables automatic adaptation to changing turbulence statistics.

This self-contained chapter pertains to both a short survey of existing literature on algorithms for identification of the Kalman gain with special focus on the most relevant algorithm, and to the introduction of a novel approach. These two algorithms will be described and derived in a general setting, setting aside the specificity of the AO context, which will be reintroduced in Chapter 4 along with the changes it incurs. The derivations of the algorithms herein will serve to justify some alterations made for AO.

3-1 Introduction

The literature regarding data-driven Kalman filtering (usually referred to as adaptive Kalman filtering) has traditionally divided existing algorithms into four categories: Bayesian estimation, maximum likelihood methods, correlation methods, and covariance matching [20, 21].

This distinction remains useful, but does not cover all algorithms, namely prediction-error methods, a fifth category adopted in this thesis.

Most algorithms in the field of data-driven Kalman filtering estimate the process and measurement noise covariances, rather than the Kalman gain; the gain is then obtained through the cumbersome DARE, whose execution time scales with the cube of the state dimension. Naturally, then, in the context of large-scale AO systems, algorithms that estimate the Kalman gain directly, skipping the DARE, are preferred.

The first two approaches, Bayesian [22, 23] and maximum likelihood [24, 25, 26] use probabilistic arguments to derive cost functions that are then minimized using computationally expensive optimization algorithms, hampering application to large-scale systems.

Correlation methods, among which [20, 27, 28, 29], draw information from the lagged autocovariances of the prediction errors of a suboptimal initial filter or the unfiltered outputs themselves. The main contenders among these algorithms then solve a set of linear equations for the process and measurement noise covariances Q and R in a least-squares setting, enabled by the use of vectorization [27, 29]. Although suitable for application in small or moderately sized systems, the use of vectorization in the process leads to execution times and storage requirements scaling with the fourth power of the output dimension, which precludes use of the algorithms for large-scale systems. In the same category, Bélanger and Carew's iterative algorithm [28], which estimates the Kalman gain directly without vectorization, has proved to be somewhat unreliable in its convergence [30].

Covariance matching methods [31, 32, 33], under very rough abstractions, update the noise covariance matrices in such a way that the resulting estimates *may* converge to the true covariances. No guarantees of convergence are given, and the estimators are highly biased. In particular, the results of the literature survey leading up to this thesis [30] show that the popular algorithm by Myers and Tapley [31] is, for the reasons above, suitable only for small corrections to an already good initial guess.

Prediction-error methods prove the most promising for large-scale application. These set a cost function $J_N(K)$ of the form

$$J_N(K) = \frac{1}{N}\sum_{k=1}^{N}\left\|y_k - \hat{y}_{k|k-1}(K)\right\|_2^2 = \frac{1}{N}\sum_{k=1}^{N}\|e_k(K)\|_2^2 \tag{3-1}$$

which can be minimized for an estimate of the Kalman gain in a least-squares setting, without vectorization, scaling with the cube of the output dimension rather than the fourth power as with correlation methods, while also providing a good setting for exploitation of the sparsity of system matrices, when applicable. These estimators also boast smaller covariances than those obtained from correlation methods [30]. Two methods in this category will be the focus of the remainder this chapter: the algorithm developed by Juang and Chen [34], and a novel proposal. Further improvements to the computational complexity are detailed in Chapter 4, given the sparsity of the system matrices in AO.

Preliminary definitions

Throughout this chapter, consider the system

$$\begin{aligned} x_{k+1} &= Ax_k + w_k \\ y_k &= Cx_k + v_k \end{aligned} \tag{3-2}$$

where $x_k \in \mathbb{R}^n$ and $y_k \in \mathbb{R}^m$. The system is assumed observable, and w_k and v_k are uncorrelated white Gaussian process and measurement noises, with covariances Q and R, respectively; that is:

$$\begin{bmatrix} w_k \\ v_k \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} Q & 0 \\ 0 & R \end{bmatrix} \right)$$

The time-update (prediction) of the corresponding steady-state Kalman filter, in both observer and innovation form, is given by [35]

$$\hat{x}_{k+1} = (A - KC)\hat{x}_k + Ky_k \tag{3-3}$$

$$\hat{x}_{k+1} = A\hat{x}_k + Ke_k \tag{3-4}$$

$$y_k = C\hat{x}_k + e_k \tag{3-5}$$

The time-update of the Kalman filter is referred to as "Kalman predictor" in this thesis. Further define the matrices

$$Y_{i,j,N} = \begin{bmatrix} y_i & y_{i+1} & y_{i+2} & \cdots & y_{i+N-1} \\ y_{i+1} & y_{i+2} & & \cdots & y_{i+N} \\ y_{i+2} & & & & \\ \vdots & & & & \vdots \\ y_{i+j-1} & y_{i+j} & & \cdots & y_{i+j+N-2} \end{bmatrix}$$

$$Y_{i,N} = \begin{bmatrix} y_i & y_{i+1} & y_{i+2} & & \cdots & y_{i+N-1} \end{bmatrix}$$

Matrix $Y_{i,j,N} \in \mathbb{R}^{jm \times N}$ is an $N-$column block-Hankel matrix with j block-rows containing output information, and $Y_{i,N} \in \mathbb{R}^{m \times N}$ is equivalent to $Y_{i,1,N}$. Other variables, $\hat{x}_k$ and e_k, will similarly have matrices so constructed, following the same definition and notation; in particular, $\hat{X}_{i,N} \in \mathbb{R}^{n \times N} = [\hat{x}_i \quad \hat{x}_{i+1} \quad \cdots \quad \hat{x}_{i+N-1}]$ is an $N-$column block-row-vector comprising entries from the optimal state prediction sequence. Matrix $\mathcal{O}_j \in \mathbb{R}^{jm \times n}$ is an extended observability matrix up to power $j-1$ of A:

$$\mathcal{O}_j = \begin{bmatrix} C^{\mathrm{T}} & (CA)^{\mathrm{T}} & \cdots & (CA^{j-1})^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$$

Remark 3.1. When the system in question includes an input term, the time-update of its Kalman filter is given by

$$\hat{x}_{k+1} = A\hat{x}_k + Bu_k + Ke_k$$

and the stochastic part can be isolated to yield (3-2). The state prediction $\hat{x}_k$ can be separated into its stochastic part $\hat{x}_k^{(\mathrm{s})}$ and its deterministic part $x_k^{(\mathrm{d})}$

$$\begin{aligned}\hat{x}_k &= x_k^{(\mathrm{d})} + \hat{x}_k^{(\mathrm{s})} \\ x_{k+1}^{(\mathrm{d})} &= A x_k^{(\mathrm{d})} + B u_k \\ \hat{x}_{k+1}^{(\mathrm{s})} &= A \hat{x}_k^{(\mathrm{s})} + K e_k \\ y_k^{(\mathrm{s})} &= y_k - C x_k^{(\mathrm{d})}\end{aligned} \tag{3-6}$$

where $y_k^{(\mathrm{s})}$ is then the y_k considered in (3-2). The input-less formulation is the one used throughout this thesis: recalling Subsection 2-2-3, $\hat{\varepsilon}_k$ would take the place of $\hat{x}_k$ in (3-6), and $\hat{\phi}_k$ would be $\hat{x}_k^{(\mathrm{s})}$.

3-2 Single AR fitting

This section describes the data-driven Kalman filtering algorithm developed by Juang and Chen [34].

Assume data is available from time-step 0 to N_d, for a total data batch length of $N_d + 1$, and take system (3-2) and the time-update of its Kalman filter in observer form (3-3); the resulting predicted state sequence follows

$$\begin{aligned}\hat{x}_1 &= (A - KC)\hat{x}_0 + K y_0 \\ \hat{x}_2 &= (A - KC)^2 \hat{x}_0 + (A - KC) K y_0 + K y_1 \\ &\vdots \\ \hat{x}_k &= (A - KC)^k \hat{x}_0 + \sum_{i=0}^{k-1} (A - KC)^{k-1-i} K y_i\end{aligned} \tag{3-7}$$

Since the error dynamics of the Kalman filter are asymptotically stable, it is possible to assume that s is chosen such that, for any $j \geq s$, then $(A - KC)^j \approx 0$; then, all terms including $(A - KC)^j$ can be neglected from the extension of (3-7) to time indices $k \geq s$, yielding

$$\hat{x}_k \approx \sum_{i=0}^{s-1} (A - KC)^{s-1-i} K y_{k-s+i} \tag{3-8}$$

which can be written as

$$\hat{x}_k \approx \underbrace{\begin{bmatrix} (A - KC)^{s-1} K & \cdots & (A - KC) K & K \end{bmatrix}}_{\mathcal{L}} \begin{bmatrix} y_{k-s} \\ \vdots \\ y_{k-2} \\ y_{k-1} \end{bmatrix} \tag{3-9}$$

Then, introducing (3-9) into (3-5) for all time-steps for which data is available yields

$$\underbrace{\begin{bmatrix} y_s & \cdots & y_{s+N-1} \end{bmatrix}}_{Y_{s,N}} \approx C\mathcal{L} \underbrace{\begin{bmatrix} y_0 & y_1 & y_2 & \cdots & y_{N-1} \\ y_1 & y_2 & & \cdots & y_N \\ y_2 & & & & \\ \vdots & & & & \vdots \\ y_{s-1} & y_s & & \cdots & y_{s+N-2} \end{bmatrix}}_{Y_{0,s,N}} + \underbrace{\begin{bmatrix} e_s & \cdots & e_{s+N-1} \end{bmatrix}}_{E_{s,N}} \tag{3-10}$$

where $N = N_d - s + 1$, so that $y_{s+N-1} = y_{N_d}$. An estimate $\widehat{C\mathcal{L}}$ of $C\mathcal{L}$ is extracted from (3-10) by solving a least-squares problem:

$$\boxed{\widehat{C\mathcal{L}} = \underset{C\mathcal{L}}{\operatorname{argmin}} \|Y_{s,N} - (C\mathcal{L})Y_{0,s,N}\|_F^2 = Y_{s,N}Y_{0,s,N}^{\dagger}} \tag{3-11}$$

where the superscript † denotes the appropriate pseudo-inverse (right pseudo-inverse, in this case), and it is assumed that $Y_{0,s,N}$ has full row rank.

Remark 3.2. The formulation here presented differs slightly from that of Juang and Chen in the original article in that instead of equation (3-10), Juang and Chen take Y_{1,N_d} in the left-hand-side and zero pad the block-Hankel matrix of the right-hand-side [34]:

$$Y_{1,N_d} \approx C\mathcal{L} \begin{bmatrix} 0 & \cdots & 0 & y_0 & y_1 & \cdots & y_{N_d-s} \\ \vdots & \iddots & \iddots & \iddots & & \cdots & y_{N_d-s+1} \\ 0 & y_0 & y_1 & & & & \vdots \\ y_0 & y_1 & y_2 & & & \cdots & y_{N_d-1} \end{bmatrix} + E_{1,N_d}$$

Naturally, the formulation in (3-10) presents a more general case, without the assumption that the output signals prior to the first time-step are all zero.

With $\widehat{C\mathcal{L}}$ obtained, the Markov parameters of the filter in observer-form ((3-3)) have been estimated up to power $s-1$. That is,

$$C\mathcal{L} = \begin{bmatrix} C(A-KC)^{s-1}K & \cdots & C(A-KC)K & CK \end{bmatrix} \tag{3-12}$$

From these, one now proceeds to obtain the first p Markov parameters of the innovation-form in (3-4), with $p \leq s$, such that the extended observability matrix up to power $p-1$ has full column rank. For simplicity, with $j \in \{1, \ldots, p\}$, define the coefficients

$$\begin{aligned} \alpha_j &= C(A-KC)^{j-1}K \\ \beta_j &= CA^{j-1}K \end{aligned}$$

The system Markov parameters β_j can be computed recursively by the following equation:

$$\boxed{\beta_j = \alpha_j + \sum_{i=1}^{j-1} \beta_{j-i}\alpha_i} \tag{3-13}$$

So the estimate $\widehat{CL}$ is used to retrieve estimates $\hat{\alpha}_j$ of the observer-form Markov parameters as per (3-12), and using (3-13) one obtains estimates $\hat{\beta}_j$ of the innovation-form parameters up to a desired $j = p$. These are stacked into an estimate $\widehat{\mathcal{O}_p K}$

$$\widehat{\mathcal{O}_p K} = \begin{bmatrix} \hat{\beta}_1 \\ \vdots \\ \hat{\beta}_p \end{bmatrix} = \begin{bmatrix} \widehat{CK} \\ \vdots \\ \widehat{CA^{p-1}K} \end{bmatrix} \tag{3-14}$$

which is used in a second least-squares problem to retrieve $\widehat{K}$:

$$\boxed{\widehat{K} = \underset{K}{\operatorname{argmin}} \left\| \widehat{\mathcal{O}_p K} - \mathcal{O}_p K \right\|_{\mathrm{F}}^2 = \mathcal{O}_p^{\dagger} \widehat{\mathcal{O}_p K}} \tag{3-15}$$

where $\mathcal{O}_p$ should have full column rank.

Theorem 3.1. Consider the asymptotically stable observable system (3-2), its Kalman filter (3-3)-(3-5), and the estimators given by (3-12) and (3-15). Assume steady-state operation and ergodicity of the signals e_k, $\hat{x}_k$, and y_k, and full row rank of $Y_{0,s,N}$. Then

$$\lim_{\substack{N\to\infty \\ s\to\infty}} \widehat{C\mathcal{L}} = C\mathcal{L} \qquad \text{and} \qquad \lim_{\substack{N\to\infty \\ s\to\infty}} \widehat{K} = K$$

Furthermore, both estimators are asymptotically unbiased in s.

Proof. Consider the exact version of (3-10):

$$Y_{s,N} = C\mathcal{L}Y_{0,s,N} + C(A-KC)^s \hat{X}_{0,N} + E_{s,N} \tag{3-16}$$

Introducing (3-16) into the formula for the estimator $\widehat{C\mathcal{L}}$ yields:

$$\widehat{C\mathcal{L}} = Y_{s,N}Y_{0,s,N}^{\dagger} = C\mathcal{L} + C(A-KC)^s \hat{X}_{0,N} Y_{0,s,N}^{\dagger} + E_{s,N}Y_{0,s,N}^{\dagger}$$

Among the properties of the innovation sequence is that it is uncorrelated with past outputs:

$$\mathrm{E}[E_{s,N}Y_{0,s,N}^{\mathrm{T}}] = 0$$

Hence,

$$\mathrm{E}[\widehat{C\mathcal{L}}] = C\mathcal{L} + \mathrm{E}[C(A-KC)^s \hat{X}_{0,N} Y_{0,s,N}^{\dagger}]$$

and, given asymptotic stability of the system, $(A - KG)^s$ vanishes in the limit of s, leading to:

$$\lim_{s\to\infty} \mathrm{E}[\widehat{C\mathcal{L}}] = C\mathcal{L}$$

That is, $\widehat{C\mathcal{L}}$ is asymptotically unbiased in s. With this established, since (3-13) is exact, it follows that the same conclusion applies to $\widehat{\mathcal{O}_p K}$. Then,

$$\lim_{s\to\infty} \mathrm{E}[\widehat{K}] = \lim_{s\to\infty} \mathrm{E}[\mathcal{O}_p^{\dagger}\widehat{\mathcal{O}_p K}] = \lim_{s\to\infty} \mathrm{E}[\mathcal{O}_p^{\dagger}\mathcal{O}_p K] = K$$

Ergodicity of the random variables involved implies that as N tends to infinity, the product $E_{s,N}Y_{0,s,N}^{\dagger}$ equals its expected value, concluding the proof. □

Remark 3.3. In truth, the data equation is true as well when considering an unbiased suboptimal filter given by:

$$
\begin{aligned}
\hat{x}^*_{k+1} &= (A - \reflectbox{K}C)\hat{x}^*_k + \reflectbox{K}y_k \\
\hat{x}^*_{k+1} &= A\hat{x}^*_k + \reflectbox{K}e^*_k \\
y_k &= C\hat{x}^*_k + e^*_k
\end{aligned}
$$

where $\reflectbox{$K$}$ denotes a suboptimal gain. In this case, the data equation reads

$$
Y_{s,N} = C\reflectbox{$\mathcal{L}$}Y_{0,s,N} + \reflectbox{E}_{s,N}
$$

where,

$$
\reflectbox{$\mathcal{L}$} = \begin{bmatrix}(A - \reflectbox{K}C)^{s-1}\reflectbox{K} & \cdots & (A - \reflectbox{K}C)\reflectbox{K} & \reflectbox{K}\end{bmatrix}
$$

and $\reflectbox{$E$}_{s,N}$ denotes a suboptimal counterpart to $E_{s,N}$. In the suboptimal case, it is no longer true that the prediction errors are uncorrelated with past outputs; that is: $\mathrm{E}[\reflectbox{$E$}_{s,N}Y^{\mathrm{T}}_{0,s,N}] \neq 0$. Following the steps of the proof of Theorem 3.1, it can be shown that it is also true that,

$$
\lim_{s\to\infty} \mathrm{E}[\widehat{C\mathcal{L}}] = C\reflectbox{$\mathcal{L}$} + \mathrm{E}[\reflectbox{E}_{s,N}Y^{\dagger}_{0,s,N}]
$$

from which follows that,

$$
C\mathcal{L} = C\reflectbox{$\mathcal{L}$} + \mathrm{E}[\reflectbox{E}_{s,N}Y^{\dagger}_{0,s,N}]
$$

The procedure can be summarized as follows:

1) Gather the data into $Y_{s,N}$ and $Y_{0,s,N}$ as in (3-10), with s such that $(A - KC)^s \approx 0$

2) Solve the least-squares problem in (3-12) for an estimate $\widehat{C\mathcal{L}}$ of $C\mathcal{L}$

3) Compute estimates of the innovation-form Markov parameters using (3-13) and construct $\widehat{\mathcal{O}_p K}$ according to (3-14)

4) Solve the final least-squares problem in (3-15) for an estimate $\widehat{K}$ of K

Time complexity

The scaling of the computational complexities of each step of Juang and Chen's algorithm with respect to the system dimensions, measured in floating-point operations (flops), are given in Table 3-1.

Step	Flop scaling
Solution of the first least-squares problem[1] (3-11)	$\mathrm{O}(Ns^2m^2 + s^2m^3)$
Computation of the p Markov parameters as per (3-13)	$\mathrm{O}(p^2m^3)$
Solution of the second least-squares problem[1] (3-15)	$\mathrm{O}(pmn^2 + mn^2)$
Total	$\mathrm{O}(Ns^2m^2 + (s^2+p^2)m^3 + pmn^2)$

Table 3-1: Time complexities per step of Juang and Chen's algorithm

Remark 3.4. Note that, for the second least-squares problem, the QR decomposition of $\mathcal{O}_p$ need only be computed once, ideally offline, for all further use, thereby reducing the complexity of this step for all subsequent runs (say, in online application with new data batches). Because the regressor $Y_{0,s,N}$ of the first least-squares problem is built with data, its decomposition must be always be done.

3-3 Multiple AR fitting

The novel algorithm described in this subsection, henceforth named MARK, standing for "Multiple Auto-Regressive" with a "K" for "Kalman", was motivated by the introduction of knowledge of A and C in the equations from which subspace identification algorithms were derived. It is closely related to the previous algorithm, and its derivation follows the same steps up to equation (3-9). The state prediction of the optimal filter at time k is given by

$$\hat{x}_k = \underbrace{\begin{bmatrix}(A-KC)^{s-1}K & \cdots & (A-KC)K & K\end{bmatrix}}_{\mathcal{L}} \begin{bmatrix} y_{k-s} \\ \vdots \\ y_{k-2} \\ y_{k-1} \end{bmatrix} + (A-KC)^s \hat{x}_0$$

Notice again that the last block-column of $\mathcal{L}$ is K. From the innovation form of the Kalman filter in (3-4), and (3-5), one obtains for an arbitrary time index k and lag j:

$$y_{k+j} = CA^j\hat{x}_k + \sum_{i=0}^{j-1} CA^{j-1-i}Ke_{k+i} \tag{3-17}$$

It is now assumed that s is chosen such that, for $k \geq s$, then $(A-KC)^k \approx 0$; this assumption is possible given asymptotic stability of the system. Then, from (3-9) and (3-17), it follows that:

$$Y_{s,p,N} = \mathcal{O}_p\hat{X}_{s,N} + \mathcal{S}_pE_{s,p,N} \tag{3-18}$$

$$\hat{X}_{s,N} = \mathcal{L}Y_{0,s,N} + \underbrace{(A-KC)^s\hat{X}_{0,N}}_{\approx 0} \tag{3-19}$$

[1]Assuming the least-squares problems are solved using the QR decomposition. The flops are given in [19]

where

$$\mathcal{S}_p = \begin{bmatrix} I & & & & \\ CK & I & & & \\ CAK & CK & I & & \\ \vdots & & \ddots & \ddots & \\ CA^{p-2}K & & \dots & CK & I \end{bmatrix}$$

Introduction of (3-19) into (3-18) yields the data equation:

$$Y_{s,p,N} = \mathcal{O}_p\mathcal{L}Y_{0,s,N} + \underbrace{\mathcal{O}_p(A-KC)^s\hat{X}_{0,N}}_{\approx 0} + \mathcal{S}_pE_{s,p,N} \tag{3-20}$$

Discarding the negligible term, the data equation reads:

$$Y_{s,p,N} \approx \mathcal{O}_p\mathcal{L}Y_{0,s,N} + \mathcal{S}_pE_{s,p,N} \tag{3-21}$$

The following least-squares problem is thus proposed to estimate the Kalman gain:

$$\boxed{\widehat{\mathcal{L}} = \underset{\mathcal{L}}{\operatorname{argmin}} \left\|Y_{s,p,N} - \mathcal{O}_p\mathcal{L}Y_{0,s,N}\right\|_{\mathrm{F}}^2 = \mathcal{O}_p^\dagger Y_{s,p,N}Y_{0,s,N}^\dagger} \tag{3-22}$$

from which an estimate $\hat{K}$ of K is drawn from the last block-column of $\widehat{\mathcal{L}}$. Problem (3-22) is equivalent to finding the solution to two least-squares problems, one row-wise with $Y_{0,s,N}$ as the regressor matrix, and one column-wise with $\mathcal{O}_p$ as the regressor matrix:

$$\widehat{\mathcal{O}_p\mathcal{L}} = \underset{\mathcal{O}_p\mathcal{L}}{\operatorname{argmin}} \left\|Y_{s,p,N} - (\mathcal{O}_p\mathcal{L})Y_{0,s,N}\right\|_{\mathrm{F}}^2 \tag{3-23}$$

$$\widehat{\mathcal{L}} = \underset{\mathcal{L}}{\operatorname{argmin}} \left\|\widehat{\mathcal{O}_p\mathcal{L}} - \mathcal{O}_p\mathcal{L}\right\|_{\mathrm{F}}^2 \tag{3-24}$$

Theorem 3.2. Consider the asymptotically stable observable system (3-2), its Kalman filter (3-3)-(3-5), and the estimators given by (3-23) and (3-24). Assume steady-state operation and ergodicity of the signals e_k, $\hat{x}_k$ and y_k, and full row rank of $Y_{0,s,N}$. Then

$$\lim_{\substack{N\to\infty \\ s\to\infty}} \widehat{\mathcal{O}_p\mathcal{L}} = \mathcal{O}_p\mathcal{L} \qquad \text{and} \qquad \lim_{\substack{N\to\infty \\ s\to\infty}} \widehat{\mathcal{L}} = \mathcal{L}$$

with both estimators being asymptotically unbiased in s.

Proof. Recall equation (3-20)

$$Y_{s,p,N} = \mathcal{O}_p\mathcal{L}Y_{0,s,N} + \mathcal{O}_p(A-KC)^s\hat{X}_{0,N} + \mathcal{S}_pE_{s,p,N}$$

Introducing (3-20) into the formula for $\widehat{\mathcal{O}_p\mathcal{L}}$ results in

$$\widehat{\mathcal{O}_p\mathcal{L}} = Y_{s,p,N}Y_{0,s,N}^\dagger = \mathcal{O}_p\mathcal{L} + \mathcal{O}_p(A-KC)^s\hat{X}_{0,N}Y_{0,s,N}^\dagger + E_{s,p,N}Y_{0,s,N}^\dagger$$

Since the innovation sequence is uncorrelated with past outputs,

$$\mathrm{E}[E_{s,p,N}Y_{0,s,N}^{\mathrm{T}}] = 0$$

then,

$$\mathrm{E}[\widehat{\mathcal{O}_p\mathcal{L}}] = \mathcal{O}_p\mathcal{L} + \mathrm{E}[\,\mathcal{O}_p(A-KC)^s\hat{X}_{0,N}Y_{0,s,N}^{\dagger}\,]$$

Because the system is asymptotically stable, in the limit as s tends to infinity $(A-KG)^s$ vanishes, yielding

$$\lim_{s\to\infty}\mathrm{E}[\widehat{\mathcal{O}_p\mathcal{L}}] = \mathcal{O}_p\mathcal{L}$$

Then,

$$\lim_{s\to\infty}\mathrm{E}[\widehat{\mathcal{L}}] = \lim_{s\to\infty}\mathrm{E}[\mathcal{O}_p^{\dagger}\widehat{\mathcal{O}_p\mathcal{L}}] = \lim_{s\to\infty}\mathrm{E}[\mathcal{O}_p^{\dagger}\mathcal{O}_p\mathcal{L}] = \mathcal{L}$$

Ergodicity of the random signals involved implies that, as N tends to infinity, the product $E_{s,p,N}Y_{0,s,N}^{\dagger}$ equals its expected value, concluding the proof. □

A remark analogous to Remark 3.3 can be made for this algorithm. It is also possible to obtain an estimate of K by solving directly for only the last block-column of $\widehat{\mathcal{L}}$ in (3-24). In MATLAB notation:

$$\boxed{\begin{aligned}\widehat{\mathcal{O}_pK} &= \widehat{\mathcal{O}_p\mathcal{L}}(\,:\,,\mathrm{end}-m+1:\mathrm{end}) \\ \widehat{K} &= \underset{K}{\operatorname{argmin}} \left\| \widehat{\mathcal{O}_pK} - \mathcal{O}_pK \right\|_{\mathrm{F}}^2\end{aligned}} \qquad \begin{aligned}(3\text{-}25)\\(3\text{-}26)\end{aligned}$$

Meaning that once the Auto-Regressive (AR) models are fit to data with (3-23), only their first Markov parameter is further relevant for the process. Our implementation of choice has the problem split into (3-23) and (3-26), reducing the number of parameters to estimate, and thus the computational burden.

The procedure can be summarized as follows:

1) Gather the data into $Y_{s,p,N}$ and $Y_{0,s,N}$ as in (3-20), with s such that $(A-KC)^s \approx 0$

2) Solve the least-squares problems in (3-22) for an estimate $\widehat{\mathcal{L}}$ of $\mathcal{L}$

3) Extract the last m columns to obtain an estimate $\widehat{K}$ of K

Time complexity

The scaling of the computational complexities of each step of MARK with respect to the system dimensions are given in Table 3-2, considering the problem is split into (3-23) and (3-26) so that only the last block-column $\widehat{\mathcal{O}_pK}$ of $\widehat{\mathcal{O}_p\mathcal{L}}$ is used to retrieve $\widehat{K}$.

Step	Flop scaling
Solution of the first least-squares problem[1] (3-23)	$\mathrm{O}(Ns^2m^2 + ps^2m^3)$
Solution of the second least-squares problem[1] (3-26)	$\mathrm{O}(pmn^2 + mn^2)$
Total	$\mathrm{O}(Ns^2m^2 + ps^2m^3 + pmn^2)$

Table 3-2: Time complexities per step of MARK

Remark 3.4 is applicable to MARK as well.

3-4 Concluding remarks

From equations (3-12) and (3-23), it is visible that the initial step of both algorithms is to fit AR models of order s to data; Juang and Chen's algorithm fits a single AR model, whereas MARK fits a series of AR models with increasing lag up to p. The former algorithm then combines the estimated Markov parameters of the observer-form Kalman filter to obtain estimates of p of the innovation-form Markov parameters, whence $\widehat{K}$ is extracted. In turn, MARK takes the first Markov parameters of each of the p AR models to build the system Markov parameters.

Given the goal of application to extremely large systems, Juang and Chen's algorithm benefits from fitting for only one AR model, which keeps its execution time significantly lower than that of MARK. In the context of this thesis, MARK's main merit is the additional possible choice of model order s, $s = 1$: whereas Juang requires at least two observer-form Markov parameters to compute the minimum of two innovation-form parameters, MARK can, instead, identify two models of order $s = 1$ to retrieve the two innovation-form Markov parameters from which $\widehat{K}$ is extracted.

This additional choice of s, even if of limited utility, proves the most robust to modelling errors, and should be useful for prediction when A is expected to poorly represent the system dynamics.

[1]Assuming the least-squares problems are solved using the QR decomposition. The flops are given in [19]

3-4-1 Reconstruction

The Kalman filter is usually formulated with two distinct steps: the measurement-update, and the time-update, given respectively by [35]:

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_{\mathrm{r}}(y_k + C\hat{x}_{k|k-1}) \tag{3-27}$$

$$\hat{x}_{k+1|k} = A\hat{x}_{k|k-1} + K(y_k + C\hat{x}_{k|k-1}) \tag{3-28}$$

where $\hat{x}_{k|k}$ and $\hat{x}_{k+1|k}$ denote the state estimates for time k and $k+1$, respectively, using data up to time k. The latter, $\hat{x}_{k+1|k}$, was referred to as just $\hat{x}_{k+1}$ up until this point; the notation $\hat{x}_{k+1|k}$ is used only in this section, where the distinction between $\hat{x}_{k+1|k}$ and $\hat{x}_{k|k}$ is relevant.

In [35], the resulting state estimates of each update are referred to as "filtered state" for the measurement update, and "one-step-ahead predicted state" for the time-update. In this thesis, the filtered state is referred to as "reconstructed state" instead, keeping in line with the use of "wavefront reconstruction" in the adaptive optics context; gain K_{r} is here referred to as "reconstructor gain", and K as either "predictor gain" (when standing in opposition to the reconstructor gain) or just "Kalman gain". Similarly, (3-27) is labelled "Kalman reconstructor".

When the process and measurement noises are uncorrelated, $K = AK_{\mathrm{r}}$ (and $\hat{x}_{k+1|k} = A\hat{x}_{k|k}$), and estimation of K_{r} requires only a minor change to the second least-squares problem of both algorithms: take (3-15) and (3-26) and replace the regressor $\mathcal{O}_p$ with $\mathcal{O}_pA$, yielding:

$$\boxed{\widehat{K_{\mathrm{r}}} = \underset{K_{\mathrm{r}}}{\operatorname{argmin}} \left\| \widehat{\mathcal{O}_pK} - (\mathcal{O}_pA)K_{\mathrm{r}} \right\|_{\mathrm{F}}^2} \tag{3-29}$$

Chapter 4

Methods for Large-Scale Adaptive Optics

This chapter is focused on the reduction of the complexity of data-driven and prediction procedures for their application to Adaptive Optics (AO). The data-driven Kalman filtering procedures from Chapter are brought 3 to AO by introducing the sparsity patterns established in Chapter 2. A novel algorithm is then proposed to reduce the burden of the prediction operation, whose execution time normally suffers from the use of a full Kalman gain.

Resume use of notation and the AO state-space system as described in Chapter 2:

$$\begin{aligned}\phi_{k+1} &= A\phi_k + w_k\\ y_k &= G\phi_k + v_k\end{aligned}$$

where $\phi_k \in \mathbb{R}^n$, and $y_k \in \mathbb{R}^m$, with $n = (L+1)^2$, $m = 2L^2$, and L being the width of the array, measured in lenslets. Notice that the measurement matrix is now denoted by G again, rather than C as in chapter 3.

4-1 Identification of the Kalman gain for large-scale AO

Several adaptations are necessary when bringing data-driven filtering to large-scale AO. First, one of the assumptions made in the derivation of the algorithms of the previous chapter does not hold in the AO context, and has to be worked around. Second, the extremely large dimensions of the systems involved make direct application of the algorithms unfeasible; it is paramount, then, to exploit the sparsity of the matrices to reduce computational complexity.

4-1-1 Removal of the mean

In chapter 3, the terms involving $(A-KC)^s$ were discarded as negligible as it is assumed that s is such that $(A-KC)^s \approx 0$. This then yielded the important equations given in (3-10) and (3-21) that led to the development of their corresponding algorithms.

In the AO context, this is not true for any small s. The error dynamics of the Kalman filter for the AR-1 turbulence model **will** have an eigenvalue close to 1 corresponding to a constant mode (an offset in the wavefront phase), which illustrates how, although possible (even if irrelevant), it takes long before the Kalman filter can accurately predict the mean of the phase.

Nevertheless, the approximations in equations (3-10) and (3-21) still hold because although $(A - KG)^s$ is not almost zero, $\mathcal{O}_p(A - KG)^s$ is negligible for low p (hence why p is kept at its minimum of 2 in Chapter 5). Indeed, when left-multiplying a constant vector by G, the result is zero, and given the shift-like structure of A, then left-multiplication by GA is expected to be almost zero as well. So when every mode but the constant one has decayed, then $\mathcal{O}_p(A - KG)^s \approx 0$ holds satisfactorily.

A problem arises when solving for $\widehat{K}$ (or $\widehat{\mathcal{L}}$), however: the system is observable, so $\mathcal{O}_p$ has full column rank, and is cut by its pseudo-inverse when solving (3-15), (3-24), or (3-26). In terms of $\widehat{\mathcal{L}}$, substituting (3-20) into (3-24):

$$\begin{aligned}\lim_{N\to\infty} \widehat{\mathcal{L}} &= \mathcal{L} + \cancelto{I}{\mathcal{O}_p^\dagger \mathcal{O}_p}(A - KG)^s \hat{X}_{0,N} Y_{0,s,N}^\dagger \\ &= \mathcal{L} + \underbrace{(A - KG)^s \hat{X}_{0,N} Y_{0,s,N}^\dagger}\end{aligned}$$

Assuming all modes but the offset have decayed from $(A - KG)^s$, the underbraced term will be a column-wise-constant distortion of $\widehat{\mathcal{L}}$. Now, $\widehat{K}$ is retrieved from the last block-column of $\widehat{\mathcal{L}}$, and will share these distortions. Let the distorted gain $\widehat{K}$ be given by K plus the column-wise-constant distortions:

$$\widehat{K} = K + \underbrace{\begin{bmatrix} | & | & & | \\ d_1 & d_2 & \cdots & d_m \\ | & | & & | \end{bmatrix}}_{D}$$

then, the predicted wavefront is given by

$$\hat{\phi}_{k+1} = A\hat{\phi}_k + Ke_k + De_k$$

Because wavefront phase offsets do not affect image quality, the mean of $\hat{\phi}_k$ can (and should) be removed every time-step. Removal of the mean is equivalent to projecting out the column-wise means, since the phase is a column vector, which ends up removing the distortions. Say this projection matrix is Π; then:

$$\begin{aligned}\Pi\hat{\phi}_{k+1} &= \Pi A\hat{\phi}_k + \Pi K e_k + \cancelto{0}{\Pi D} e_k \\ &= \Pi A\hat{\phi}_k + \Pi K e_k\end{aligned}$$

Meaning that not only does the distortion of $\widehat{K}$ by D not affect image quality, but also that it is trivial to remove from the predictions. Nonetheless, it is advised to remove the column means from the gain $\widehat{K}$ itself as well, otherwise it is rendered unrecognizable, as shown in Figure 4-1.

Remark 4.1. When the Kalman gain is identified, the constant mode of the error dynamics might actually be unstable, rather than just having an eigenvalue close to, but smaller than 1, especially if the column-means of the gain were preemptively removed. Although irrelevant in theory, instability of the constant mode will lead to numerical problems unless the means of the predicted wavefront phases are removed.

Remark 4.2. Subsection 4-1-2 describes identification of a sparse, distance-constrained gain. This sparse gain does not fit for the distortions, and removal of its column means is not required; in fact, doing so would destroy its sparsity, much like with A, and should not be done.

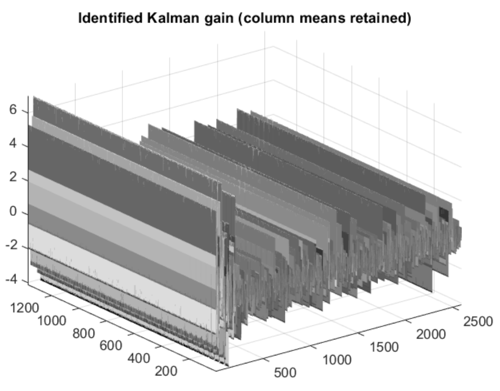

Figure 4-1: Identified Kalman gain without the column means removed.

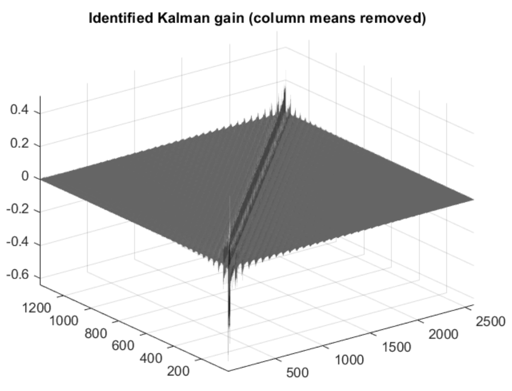

Figure 4-2: Identified Kalman gain with the column means removed.

4-1-2 Sparse identification

Recall the first least-squares problems (3-11) and (3-23) of the algorithms described in Chapter 3. In their application to AO, both of these problems consist of fitting AR models to the slope data, which means that the sparsity patterns established in Subsection 2-3-2 are applicable. Begin with Juang and Chen's algorithm, and introduce AO notation into (3-11):

$$\widehat{G\mathcal{L}} = \underset{G\mathcal{L}}{\operatorname{argmin}} \|Y_{s,N} - (G\mathcal{L})Y_{0,s,N}\|_{\text{F}}^2$$

After establishing a sparsity pattern via distance-constraints as per Section 2-3-2, then rather than identifying the full $G\mathcal{L}$, one can identify only the known non-zeros by separating the problem into its m independent rows, and solving instead

$$\boxed{\widehat{G\mathcal{L}}^{(j,idx)} = \underset{(G\mathcal{L})^{(j,idx)}}{\operatorname{argmin}} \left\| Y_{s,N}^{(j,:)} - (G\mathcal{L})^{(j,idx)} Y_{0,s,N}^{(idx,:)} \right\|_2^2} \tag{4-1}$$

for each row j (out of m rows), where idx denotes the non-zero indices of the j^{th} row of $G\mathcal{L}$, and the superscript $(\star,\star)$ indicates the row and column indices of a matrix in MATLAB notation. Analogously, consider MARK's (3-23):

$$\widehat{\mathcal{O}_p\mathcal{L}} = \underset{\mathcal{O}_p\mathcal{L}}{\operatorname{argmin}} \left\| Y_{s,p,N} - (\mathcal{O}_p\mathcal{L})Y_{0,s,N} \right\|_{\text{F}}^2$$

which involves fitting p AR models of increasing lag, indicated by a superscript (i):

$$\mathcal{O}_p\mathcal{L} = \begin{bmatrix} - & \mathcal{A}^{(1)} & - \\ - & \mathcal{A}^{(2)} & - \\ & \vdots & \\ - & \mathcal{A}^{(p)} & - \end{bmatrix} = \left[\begin{array}{ccc} \mathcal{A}_s^{(1)} & \dots & \mathcal{A}_1^{(1)} \\ \hdashline \mathcal{A}_s^{(2)} & \dots & \mathcal{A}_1^{(2)} \\ \hdashline & \vdots & \\ \hdashline \mathcal{A}_s^{(p)} & \dots & \mathcal{A}_1^{(p)} \end{array}\right]$$

Now say a radius of r is chosen as the base distance-constraint; remember that each Markov parameter $\mathcal{A}_i^{(1)} = G(A - KG)^{i-1}K$ has a distance-constraint of $r_i^{(1)} = i \cdot r$ imposed upon it to accommodate the time-step delay between the terms it relates. Then, following the same reasoning, the distance-constraint of Markov parameter $\mathcal{A}_i^{(j)}$ should be $r_i^{(j)} = i \cdot j \cdot r$. Problem (3-23) thus becomes:

$$\boxed{\widehat{\mathcal{O}_p\mathcal{L}}^{(j,idx)} = \underset{(\mathcal{O}_p\mathcal{L})^{(j,idx)}}{\operatorname{argmin}} \left\| Y_{s,N}^{(j,:)} - (\mathcal{O}_p\mathcal{L})^{(j,idx)} Y_{0,s,N}^{(idx,:)} \right\|_2^2} \tag{4-2}$$

for each row j out of a total of pm rows.

Sparse Kalman gain

As for the second least-squares problem given in (3-15) and (3-26):

$$\widehat{K} = \underset{K}{\operatorname{argmin}} \left\| \widehat{\mathcal{O}_p K} - \mathcal{O}_p K \right\|_{\text{F}}^2$$

Its regressand $\widehat{\mathcal{O}_p K}$ is taken from the solution to (4-1), and is hence sparse, and the regressor is the highly sparse extended observability matrix $\mathcal{O}_p$ known beforehand. The high sparsity of these two matrices is such that this least-squares problem can be solved quickly using dedicated sparse QR solvers. MATLAB's backslash operator applies one such solver [36]. Figure 5-29 suggests the time is roughly $\text{O}(m)$ in practise.

Nonetheless, for the two-stage algorithm described in the next section, consider an alternative gain $\overline{K}$ with a sparsity pattern such that each pixel is influenced only by measurement points within an area around itself. Then following the previous steps, the least-squares problem can be split into its m independent columns and solved only for the non-zeros:

$$\overline{K}^{(idx,j)} = \underset{K^{(idx,j)}}{\operatorname{argmin}} \left\| \widehat{\mathcal{O}_p K}^{(:,j)} - \mathcal{O}_p^{(:,idx)} K^{(idx,j)} \right\|_2^2 \tag{4-3}$$

However, the Kalman gain is usually not sparse, and this sparse distance-constrained gain will fail to predict the the modes of the wavefront that span larger distances. The next section will detail a two-stage prediction procedure that employs this sparse gain as a first prediction step, together with a second step for prediction of the low frequencies of the wavefront.

Sparse identification of a gain has merits with respect to storage: a full Kalman gain has nm entries, which, for extremely large systems, will far exceed Nm (where N is the data batch length), since the sparsity of the Markov parameters makes the required amount of data depend on the number of non-zeros, which is fixed for a given wind speed in lenslets per time-step, rather than the system dimensions.

Remark 4.3 *(Regularized least-squares).* Unlike the full Kalman gain, it is normally not necessary to remove the mean of the sparse gain, as (4-3) does not fit for the column-wise constant distortions unless the distance-constraint is too large for the gain to truly be sparse in the first place. It is also undesirable to remove the column means, as doing so destroys the sparsity of the gain.

In any case, it is worth mentioning that, rather than subtracting the mean, it is possible to truly ensure the distortions do not affect the gain, without destroying sparsity, via regularized least-squares, at the expense of efficiency:

$$\overline{K}^{(idx,j)} = \underset{K^{(idx,j)}}{\text{argmin}} \left\| \widehat{\mathcal{O}_p K}^{(:,j)} - \mathcal{O}_p^{(:,idx)} K^{(idx,j)} \right\|_2^2 + \lambda \left\| \Pi^{(:,idx)} K^{(idx,j)} \right\|_2^2$$

where $\Pi = [1 \;\; 1 \;\; \cdots \;\; 1]$. The means of the non-zero entries of each column of $\overline{K}$ are thus penalized in the cost function with weight λ, which, kept small, retains performance.

4-1-3 Setting the distance constraint for the gain

Because the gain relates pixels and lenslets, and the former outnumber the latter, its distance-constraint is set differently: a constraint of zero forces a pixel to be influenced only by the four lenslets immediately around it, with increased distances expanding the square outward. Figure 4-3 displays the acceptable interactions for the distance-constraints of 0 and 1.

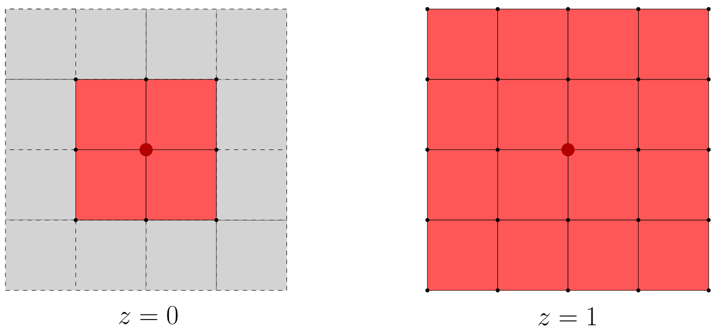

Figure 4-3: Example distance-constraint of the gain. The central dark red dot represents the pixel whose interactions are being constrained, the red squares represent the lenslets that fulfill a distance-constraint of z.

Note that having a zero distance include the immediately adjacent four lenslets is a mere design choice, and the user can choose a different convention. In fact, the area set with the constraint need not be square, but was chosen was as such for simplicity. Nonetheless, this is the convention with which the results of Chapter 5 were obtained.

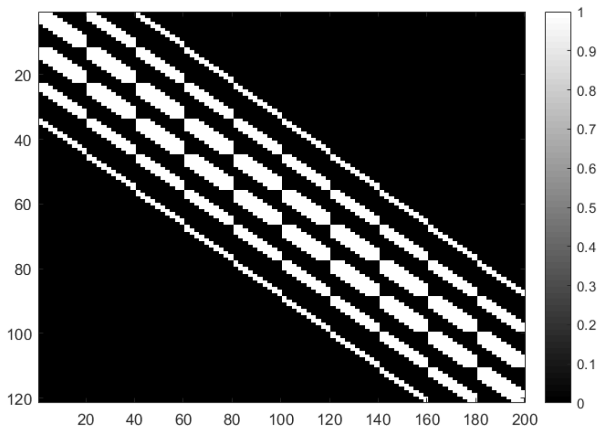

Figure 4-4: Exaggerated non-zero pattern of a sparse gain for a small-scale system. White entries are non-zeros, black entries are zero.

4-2 Two-stage prediction

The previous section outlined a technique for the identification of a sparse gain with computational benefits for the online prediction operation. However, although some authors argue that the spatially decaying structure of the Kalman gain enables sole consideration of the relationships between pixels and lenslets at a certain maximal distance [37, 38], generally, this distance will be too large for the resulting gain to be sparse; if picked too small, the deterioration of the results will be significant. Indeed, when prediction should rely heavily on the measurements rather than the model, a sparse distance-constrained gain fails to predict some modes of the wavefront phase that span larger distances. Figure 4-5 displays a particularly egregious example, showing a case in which an artificially sparse gain is able to predict local structures in the phase, but fails to account for the overarching shape.

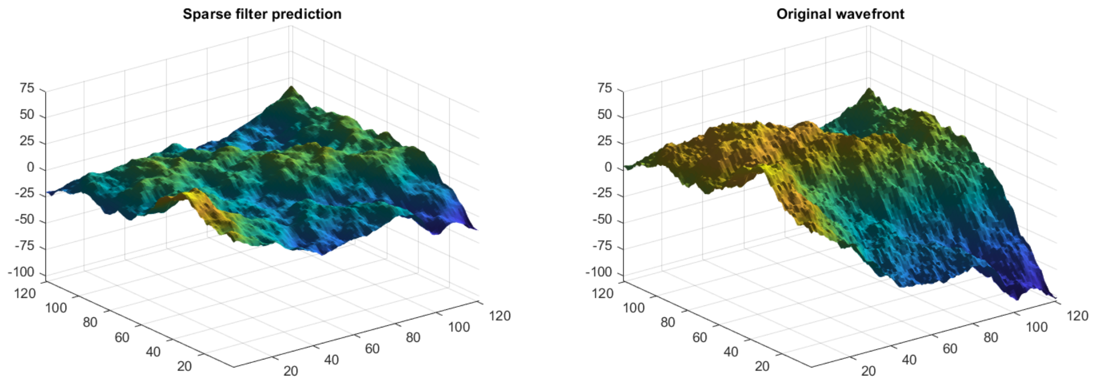

Figure 4-5: Example prediction with a sparse gain and a noisy model (left) and phase of the incoming wavefront (right). High sparsity was imposed to obtain a clear display of the nature of the error induced by distance-constrained sparsity.

Albeit to a lesser extent than shown, this will be the general case. Reliance on the measurements is reduced by increasing model quality, which might be lacking for multiple reasons, among which imperfect measurements, and those listed in Section 2-2-2. With this in mind, a two-stage approach to the prediction problem is proposed in this section:

- The high-frequency, local components of the wavefront are predicted with a sparse gain. This predictor is applied in the original fine array. Depending on the sparsity and the array width, most of the prediction should happen in this step.
- The remaining low-frequency components are predicted on a low-dimensional coarse array. These are then interpolated and added to the high-frequency predictions to yield the predicted wavefront phase.

The goal is to perform only fast sparse operations in the original large array, with further corrections done on a low-dimensional coarse array that allows operations on full matrices without significant computational burden.

However, while the two-stage predictor will use data-driven procedures that seek to attain prediction errors with the properties of the (optimal) innovation sequence, it should be noted that the obtained predictor will not be a Kalman predictor, as the separation into stages and use of interpolation will necessarily preclude optimality. Regardless, while not theoretically optimal, the two-stage approach may still yield better results in practice than the "theoretically optimal" Kalman filter, given the possibility of erroneous modelling and finiteness of data.

4-2-1 Formulation

Denote by $\overline{K}$ the distance-constrained fine gain. The first filter predicts all that is possible with $\overline{K}$:

$$\hat{\phi}^{(\mathrm{f})}_{k+1} = A\hat{\phi}^{(\mathrm{f})}_k + \overline{K}(y_k - G\hat{\phi}^{(\mathrm{f})}_k) \tag{4-4}$$

where the superscript (f) stands for "fine". The error dynamics of (4-4) are given by

$$\begin{aligned} \xi_k &= \phi_k - \hat{\phi}^{(\mathrm{f})}_k \\ \xi_{k+1} &= (A - \overline{K}G)\xi_k + \overline{K}v_k - w_k \\ e^{(\mathrm{f})}_k &= G\xi_k + v_k \end{aligned} \tag{4-5}$$

with $e^{(\mathrm{f})}_k$ denoting the prediction-error sequence of the fine filter. Consider now a second filter, with gain J, that seeks to predict ξ_{k+1}:

$$\hat{\xi}_{k+1} = (A - \overline{K}G)\hat{\xi}_k + J(e^{(\mathrm{f})}_k - G\hat{\xi}_k) \tag{4-6}$$

The idea underlying the proposed two-stage filter is that ξ_k contains only low frequencies that can be reliably predicted with a coarser array (and the *very*-high-frequencies that can not be predicted at all). Let M_c and M_i denote sparse combination and linear interpolation matrices, respectively; the combination matrix M_c is elaborated upon below. Then design the coarse second predictor:

$$\hat{\phi}^{(\mathrm{c})}_{k+1} = (A - \overline{K}G)'\hat{\phi}^{(\mathrm{c})}_k + K'(M_\mathrm{c}[y_k - G\hat{\phi}^{(\mathrm{f})}_k] - G'\hat{\phi}^{(\mathrm{c})}_k) \tag{4-7}$$

where the primes denote coarse counterparts to the matrices to which they are applied, except K', which merely denotes the gain applied to the coarse predictor. How to obtain the matrices for the coarse array (and M_c) is described ahead. The fine and coarse predictors, (4-4) and (4-7), are finally combined to yield the two-stage predictor:

$$\boxed{\begin{cases} \hat{\phi}^{(\mathrm{f})}_{k+1} &= A\hat{\phi}^{(\mathrm{f})}_k + \overline{K}(s_k - G\hat{\phi}^{(\mathrm{f})}_k) \\ \hat{\phi}^{(\mathrm{c})}_{k+1} &= (A - \overline{K}G)'\hat{\phi}^{(\mathrm{c})}_k + K'\left(M_\mathrm{c}[y_k - G\hat{\phi}^{(\mathrm{f})}_k] - G'\hat{\phi}^{(\mathrm{c})}_k\right) \\ \hat{\phi}_{k+1} &= \hat{\phi}^{(\mathrm{f})}_{k+1} + M_\mathrm{i}[\hat{\phi}^{(\mathrm{c})}_{k+1}] \end{cases}} \tag{4-8}$$

Remark 4.4. If $\overline{K}$ equals the optimal gain K, then the optimal J in (4-7) is zero, so if the first predictor is already optimal, adding a second one can not improve the results further.

Time complexity

Let n_c and m_c denote the number of states and number of outputs of the coarse system, respectively. Because both n and m scale as $\mathrm{O}(L^2)$, and n_c and m_c as $\mathrm{O}(L_\mathrm{c}^2)$, $\mathrm{O}(n)$ is equivalent to $\mathrm{O}(m)$, and so is $\mathrm{O}(n_\mathrm{c})$ to $\mathrm{O}(m_\mathrm{c})$, so the complexities will therefore be given in terms of just m and m_c.

For a fixed wind speed in lenslets per time-step, distance-constraint z, and coarse array width L_c, the non-zeros of A, $\overline{K}$, M_c, and M_i scale linearly with their numbers of rows (or columns), and each has at least one non-zero per row. Hence, terms that scale with the non-zeros of any of these sparse matrices will be said to scale as $\mathrm{O}(m)$.

The time complexity of (4-8) is given in Table 4-1:

Step	Flop scaling
Fine prediction (sparse) (4-4)	$\mathrm{O}(m)$
Coarse prediction (full) (4-7)	$\mathrm{O}(m + m_\mathrm{c}^2)$
Final combination	$\mathrm{O}(m)$
Total	$\mathrm{O}(m + m_\mathrm{c}^2)$

Table 4-1: Time complexities per step of two-stage prediction, assuming fixed tuning parameters.

If z and L_c are not kept fixed, the terms $\mathrm{nnz}(\overline{K})$, $\mathrm{nnz}(M_\mathrm{c})$, and $\mathrm{nnz}(M_\mathrm{i})$ appear in the complexities, or alternatively, m times the average non-zeros per column or row.

4-2-2 Description of the coarse system

Note that (4-7) and (4-8) refer to a particular formulation, in which the coarse stage predicts the state-prediction-error of the fine stage. Rather, here the coarse description is given for a general system whose fine *counterpart* has its measurement matrix given by G, and whose state dynamics are irrelevant. Define y' and ϕ' as the slopes and state, respectively, corresponding to the coarse system such that

$$y' = G'\phi' \tag{4-9}$$

The description herein is directly applicable to (4-7) nonetheless, with the slopes then being $M_\mathrm{c}(y_k - G\hat{\phi}_k^{(f)})$, and the state $\hat{\phi}_k^{(c)}$. In that case, the fine *counterpart* to the coarse system is (4-6), and not (4-4) (which is instead the fine *stage* of the two-stage algorithm).

The coarse array is built of units, the coarse equivalent to lenslets, which encompass multiple lenslets from the fine array. Let the superscript $\boxed{i}$ indicate lenslet i of the coarse array, and (j) indicate lenslet j of the fine array within a given coarse unit, counted in column-major order. As a simple example, consider a 2×2 array of lenslets, used as the fine array of the two-stage predictor (see Figure 4-6). In this example, the coarse array consists of a single lenslet, obtained by combining the slope measurements of the fine array. The slopes of the fine array are given by:

$$\begin{gathered}\begin{cases} 2y_\mathrm{x}^{(1)} &= \varphi_5 + \varphi_4 - \varphi_2 - \varphi_1 \\ 2y_\mathrm{y}^{(1)} &= \varphi_5 + \varphi_2 - \varphi_4 - \varphi_1 \end{cases} \\ \vdots \\ \begin{cases} 2y_\mathrm{x}^{(4)} &= \varphi_9 + \varphi_8 - \varphi_6 - \varphi_5 \\ 2y_\mathrm{y}^{(4)} &= \varphi_9 + \varphi_6 - \varphi_8 - \varphi_5 \end{cases}\end{gathered} \tag{4-10}$$

where the superscript number indicates the corresponding lenslet, and the subscript indicates the axis along which the slope is computed. The slopes of the coarse array are determined such that they fill a role equivalent to the original slopes, applied instead to the coarse array. That is,

$$\begin{cases} 2y_\mathrm{x}^{\boxed{1}} &= \varphi_9 + \varphi_7 - \varphi_3 - \varphi_1 \\ 2y_\mathrm{y}^{\boxed{1}} &= \varphi_9 + \varphi_3 - \varphi_7 - \varphi_1 \end{cases} \tag{4-11}$$

This concept can be generalized to fine coarse arrays of different sizes as long as the coarse units are square and the width of the fine array can be divided by that of the coarse array. Keep this example in mind, as its use will resume ahead, but consider as well consider arbitrary widths fulfilling both of these conditions: let L denote the width of the fine array, L_c that of the coarse array, and L_u that of the coarse units.

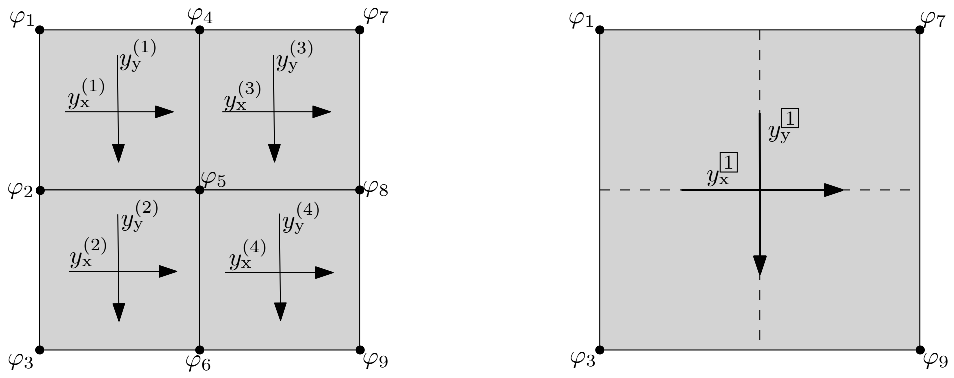

Figure 4-6: Example fine (left) and corresponding coarse (right) arrays. Because the example has only one coarse unit, the per-unit notation matches that of the entire fine array. Note that in the notation of this section, each coarse unit would look like the left image, indices included (so with the indices beginning from 1 and ending at $(L_u + 1)^2$ for the pixels, and L_u^2 for the lenslets).

Determining the combination

The slopes $y_x^{\boxed{i}}$ and $y_y^{\boxed{i}}$ corresponding to the coarse unit i are obtained by combining the fine slopes of that same unit:

$$\begin{aligned} y_x^{\boxed{i}} &= \underbrace{\begin{bmatrix} \alpha_x^{(1)} & \alpha_y^{(1)} & \cdots & \alpha_x^{(L_u^2)} & \alpha_y^{(L_u^2)} \end{bmatrix}}_{\mathcal{C}_x} \begin{bmatrix} y_x^{(1)} \\ y_y^{(1)} \\ \vdots \\ y_x^{(L_u^2)} \\ y_y^{(L_u^2)} \end{bmatrix} \\ y_y^{\boxed{i}} &= \underbrace{\begin{bmatrix} \beta_x^{(1)} & \beta_y^{(1)} & \cdots & \beta_x^{(L_u^2)} & \beta_y^{(L_u^2)} \end{bmatrix}}_{\mathcal{C}_y} \begin{bmatrix} y_x^{(1)} \\ y_y^{(1)} \\ \vdots \\ y_x^{(L_u^2)} \\ y_y^{(L_u^2)} \end{bmatrix} \end{aligned} \tag{4-12}$$

where $\mathcal{C}_x$ and $\mathcal{C}_y$ comprise coefficients to be set such that (4-11) is verified, knowing (4-10). These depend solely on the width of the coarse unit. Define the vector Φ containing the wavefront phases within an arbitrary coarse unit:

$$\Phi = \begin{bmatrix} \varphi_1 & \varphi_2 & \cdots & \varphi_{(L_u+1)^2} \end{bmatrix}^T$$

Then, each slope from (4-10) or (4-11) is given by multiplying a vector of ones and zeros to $\frac{1}{2}\Phi$:

$$\begin{cases} y_x^{(j)} &= \frac{1}{2} v_x^{(j)} \Phi \\ y_y^{(j)} &= \frac{1}{2} v_y^{(j)} \Phi \end{cases} \qquad \begin{cases} y_x^{\boxed{i}} &= \frac{1}{2} v_x' \Phi \\ y_y^{\boxed{i}} &= \frac{1}{2} v_y' \Phi \end{cases} \tag{4-13}$$

Returning to the example above, with a coarse unit width of 2, it should be visible that, for instance,

$$\begin{cases} y_{\mathrm{x}}^{(1)} &= \frac{1}{2}\begin{bmatrix}-1 & -1 & 0 & 1 & 1 & 0 & 0 & 0 & 0\end{bmatrix}\Phi \\ y_{\mathrm{y}}^{(1)} &= \frac{1}{2}\begin{bmatrix}-1 & 1 & 0 & -1 & 1 & 0 & 0 & 0 & 0\end{bmatrix}\Phi \\ &\vdots \\ y_{\mathrm{x}}^{(4)} &= \frac{1}{2}\begin{bmatrix}0 & 0 & 0 & 0 & -1 & -1 & 0 & 1 & 1\end{bmatrix}\Phi \\ y_{\mathrm{y}}^{(4)} &= \frac{1}{2}\begin{bmatrix}0 & 0 & 0 & 0 & -1 & 1 & 0 & -1 & 1\end{bmatrix}\Phi \end{cases}$$

and

$$\begin{cases} y_{\mathrm{x}}^{\boxed{1}} &= \frac{1}{2}\begin{bmatrix}-1 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 1\end{bmatrix}\Phi \\ y_{\mathrm{y}}^{\boxed{1}} &= \frac{1}{2}\begin{bmatrix}-1 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 1\end{bmatrix}\Phi \end{cases} \tag{4-14}$$

The vectors of ones and zeros corresponding to the fine slopes within the (arbitrary) unit define a basis for the space of possible combinations. Say these vectors, labelled $v_{\mathrm{x}}^{(j)}$ and $v_{\mathrm{y}}^{(j)}$ are stacked into V_{x} and V_{y}:

$$V_{\mathrm{x}} = \begin{bmatrix} — & v_{\mathrm{x}}^{(1)} & — \\ — & v_{\mathrm{x}}^{(2)} & — \\ & \vdots & \\ — & v_{\mathrm{x}}^{(L_{\mathrm{u}}^2)} & — \end{bmatrix} \qquad V_{\mathrm{y}} = \begin{bmatrix} — & v_{\mathrm{y}}^{(1)} & — \\ — & v_{\mathrm{y}}^{(2)} & — \\ & \vdots & \\ — & v_{\mathrm{y}}^{(L_{\mathrm{u}}^2)} & — \end{bmatrix}$$

Let v'_{x} and v'_{y} the coarse counterpart to these vectors, as in (4-14); that is, $y_{\mathrm{x}}^{\boxed{i}} = \frac{1}{2}v'_{\mathrm{x}}\Phi$ and $y_{\mathrm{y}}^{\boxed{i}} = \frac{1}{2}v'_{\mathrm{y}}\Phi$. Then, from (4-12), for *any* Φ:

$$\begin{aligned} v'_{\mathrm{x}}\Phi &= \mathcal{C}_{\mathrm{x}}V_{\mathrm{x}}\Phi \\ v'_{\mathrm{y}}\Phi &= \mathcal{C}_{\mathrm{y}}V_{\mathrm{y}}\Phi \end{aligned} \tag{4-15}$$

Because (4-15) must hold for all Φ, it is true only when

$$\begin{cases} v'_{\mathrm{x}} &= \mathcal{C}_{\mathrm{x}}V_{\mathrm{x}} \\ v'_{\mathrm{y}} &= \mathcal{C}_{\mathrm{y}}V_{\mathrm{y}} \end{cases} \tag{4-16}$$

which is possible only when the units are be square. In such a case, least-squares solutions are exact, but not unique, so one can use the truncated SVDs of $\mathcal{A}$ and $\mathcal{B}$ to obtain the unique minimum-norm solution to each of the equations in (4-16). Let the superscript † indicate the pseudo-inverse computed from the truncated SVD:

$$\boxed{\begin{aligned} \mathcal{C}_{\mathrm{x}} &= v'_{\mathrm{x}}V_{\mathrm{x}}^{\dagger} \\ \mathcal{C}_{\mathrm{y}} &= v'_{\mathrm{y}}V_{\mathrm{y}}^{\dagger} \end{aligned}} \tag{4-17}$$

Again, <u>as long as the coarse units are square</u>, (4-17) is such that (4-16) is exact, with error zero. Finally, define the (sparse) combination matrix M_{c} that converts the fine slopes into coarse slopes:

$$\underbrace{\begin{bmatrix} y^{\boxed{1}} \\ y^{\boxed{2}} \\ \vdots \\ y^{\boxed{L_\mathrm{c}^2}} \end{bmatrix}}_{y'} = M_\mathrm{c} y$$

where

$$y^{\boxed{i}} = \begin{bmatrix} y_\mathrm{x}^{\boxed{i}} \\ y_\mathrm{y}^{\boxed{i}} \end{bmatrix}$$

To build this matrix, determine the entries of the original slopes to which each $y_k^{(i)}$ corresponds, for all coarse units, and assign M_c the entries of $\mathcal{C}_\mathrm{x}$ and $\mathcal{C}_\mathrm{y}$ accordingly, analogously to how G is built. Let $idx^{\boxed{i}}$ be a vector containing the indices of all $y_\mathrm{x}^{(j)}$ and $y_\mathrm{y}^{(j)}$ corresponding to unit i within the vector y of original slopes; then, in MATLAB notation:

$$\boxed{M_\mathrm{c}(\ 2i-1 : 2i\ ,\ idx^{\boxed{i}}\ ) = \begin{bmatrix} \mathcal{C}_\mathrm{x} \\ \mathcal{C}_\mathrm{y} \end{bmatrix}} \tag{4-18}$$

with $2i-1$ and $2i$ ($i \geq 1$) being the row indices of the entries of M_c corresponding to $y_\mathrm{x}^{\boxed{i}}$ and $y_\mathrm{y}^{\boxed{i}}$, respectively, i.e. the indices of $y_\mathrm{x}^{\boxed{i}}$ and $y_\mathrm{y}^{\boxed{i}}$ in y'.

Remark 4.5. Keep in mind that "MATLAB notation", rather unfortunately, implies that the spots of the row- and column-indices in $(\star, \star)$ are reversed with respect to the normal x- and y-axis notation. In MATLAB notation, the first entry of $(\star, \star)$ is the row index (y-axis) and the second is the column index (x-axis).

Obtaining the system matrices

A benefit of this particular combination is that G' is built exactly as G is, as described in Subsection 2-1-1, but considering the dimensions of the coarse system instead.

As for the state-transition matrix, let Z be the selection matrix of ones and zeros such that

$$\phi' = Z\phi \tag{4-19}$$

Consider a dataset containing wavefront phase data ϕ_k, available for identification. The state-transition matrix can be computed as per Subsections 2-2-2 and 2-3-1. In the two-stage algorithm, the coarse state is given by

$$\phi_k^{(\mathrm{c})} = Z \underbrace{\left(\phi_k - \phi_k^{(\mathrm{f})}\right)}_{\xi_k}$$

which takes the spot of ϕ' in (4-9) and (4-19). Chapter 5 briefly describes a way to obtain filtered wavefront data from slopes, which is therein used to obtain a noisy model. To obtain $(A - \overline{K}G)'$, one follows the steps:

1) Identify gain $\overline{K}$ in (4-4) from data using the sparse procedure in (4-3).

2) Apply the sparse predictor to the the wavefront phase dataset, using (4-4):

$$\hat{\phi}^{(\mathrm{f})}_{k+1} = A\hat{\phi}^{(\mathrm{f})}_k + \overline{K}(y_k - G\hat{\phi}^{(\mathrm{f})}_k)$$

beginning from an arbitrary initial condition; a simple choice is all zeros.

3) Estimate the coarse wavefront-prediction-errors as

$$\phi^{(\mathrm{c})}_k = Z\xi_k = Z\phi_k - Z\hat{\phi}^{(\mathrm{f})}_k$$

where ϕ_k is wavefront data from an available identification dataset. Rather than explicit left-multiplication by Z, it is faster (and theoretically equivalent) to merely sample the variable to which it is multiplied at the pixels of the coarse array. This wavefront data **must** correspond to the slope data used in step 2! That is, this y_k must be the slopes measured from ϕ_k, including the measurement noise.

4) Form "future"- and "past"-data matrices $\Phi^{(\mathrm{c})}_{1,N}$ and $\Phi^{(\mathrm{c})}_{0,N}$:

$$\Phi^{(\mathrm{c})}_{1,N} = \begin{bmatrix}\phi^{(\mathrm{c})}_1 & \cdots & \phi^{(\mathrm{c})}_N\end{bmatrix} \qquad \Phi^{(\mathrm{c})}_{0,N} = \begin{bmatrix}\phi^{(\mathrm{c})}_0 & \cdots & \phi^{(\mathrm{c})}_{N-1}\end{bmatrix}$$

5) Identify $(A - \overline{K}G)'$ via least-squares (see subsection 2-2-2):

$$\widehat{(A - \overline{K}G)'} = \Phi^{(\mathrm{c})}_{1,N}\left(\Phi^{(\mathrm{c})}_{0,N}\right)^{\dagger} \tag{4-20}$$

or employ the sparse procedure from subsection 2-3-1.

It is common for authors to directly preset A from preliminary assumptions [3, 4], eliminating the need for a wavefront dataset for its identification. Even in such a case, wavefront data is necessary for identification of $(A - \overline{K}G)'$ (see Remark 4.6), which can be obtained from the slopes (say, by pseudo-inverting G, G', or using more robust methods).

Remark 4.6. The state transition matrix $(A-\overline{K}G)'$ as seen in (4-8), albeit being the coarse counterpart to $(A - \overline{K}G)$, should not just be a selection of the entries of $(A - \overline{K}G)$ shared with the coarse grid. Defining $(A - \overline{K}G)'$ as shown, one arrives at the relationship:

$$(A - \overline{K}G)' = Z(A - \overline{K}G)\overline{P}Z^{\mathrm{T}}(Z\overline{P}Z^{\mathrm{T}})^{-1}$$

where $\overline{P}$ is the state error covariance of the sparse predictor.

Identifying the coarse gain

Identification of the coarse gain K' requires measurement data for the data-driven identification procedures. Equation (4-5) establishes that the output of the error dynamics of the fine filter is its slope-prediction-errors, given by:

$$e^{(\mathrm{f})}_k = (y_k - G\hat{\phi}^{(\mathrm{f})}_k)$$

The measurements for the coarse filter, then, are these combined into their coarse counterparts:

$$y_k^{(\mathrm{c})} = M_{\mathrm{c}} e_k^{(\mathrm{f})} = M_{\mathrm{c}}\big(y_k - G\hat{\phi}_k^{(\mathrm{f})}\big)$$

With these measurements computed, and G' and $(A - \overline{K}G)'$ obtained previously, the user now has all that is necessary to apply either MARK or Juang to obtain K':

$$K' = \begin{cases} \texttt{MARK}\Big((A - \overline{K}G)',\, G',\, y_\star^{(\mathrm{c})}\Big) \\ \texttt{Juang}\Big((A - \overline{K}G)',\, G',\, y_\star^{(\mathrm{c})}\Big) \end{cases} \tag{4-21}$$

4-2-3 Summary

The two-stage prediction algorithm is summarized into the following steps:

1) Decide the tuning parameters: $s, p, s_{\mathrm{c}}, p_{\mathrm{c}}, L_{\mathrm{c}}$, and all the distance-constraints (Markov parameters and gain). The subscript c indicates the coarse-stage counterpart of a parameter.

2) Build G, and G', M_{c}, and M_{i} for the coarse stage. Acquire A, preferably from wavefront phase data, and using the procedure from Subsection 2-3-1. Form the extended observability matrices for both the fine and coarse stages.

3) Identify the sparse gain $\overline{K}$ in (4-4) from slope data and A using the sparse procedure in (4-3).

4) Apply the filter with gain $\overline{K}$ to the slope data, following (4-4), and compute the wavefront prediction errors $\phi_k^{(\mathrm{c})} = Z\phi_k - Z\hat{\phi}_k^{(\mathrm{f})}$.

5) Identify $(A - \overline{K}G)'$ from the wavefront prediction errors as per (4-20).

6) Compute the slope prediction errors and combine them to yield the measurements for the coarse filter: $y_k^{(\mathrm{c})} = M_{\mathrm{c}}\big(y_k - G\hat{\phi}_k^{(\mathrm{f})}\big)$.

7) Identify the coarse gain as per (4-21) using $y_\star^{(\mathrm{c})}$ and $(A - \overline{K}G)'$.

8) Apply (4-8) online for prediction.

Remark 4.7. The construction of the coarse system here described considers only square units. Rectangular unit shapes can not be made to follow (4-11), as (4-16) will not hold without an error term, thus demanding different construction. That is,

$$\begin{bmatrix} v_{\mathrm{x}}' \\ v_{\mathrm{y}}' \end{bmatrix} \notin \text{ row space}\left(\begin{bmatrix} V_{\mathrm{x}} \\ V_{\mathrm{y}} \end{bmatrix}\right)$$

One obvious shortcoming, then, is that this description of the two-stage algorithm requires the width of the original array to be divisible by that of the coarse array, limiting the possible choices (especially if the width turns out to be a prime). To apply it to circular arrays and accommodate the square units, a possible approach (not evaluated in this thesis) is to artificially set the slopes outside the aperture to zero when combining them with (4-11); another possibility is to define a measure of error between the interpolated and original wavefront (in the fine grid), and set these slopes to values that minimize the measure within their corresponding coarse unit.

Chapter 5

Results and Tuning

This chapter is dedicated to an analysis of the algorithms developed in this thesis. These are the sparse data-driven procedure for identification of the Kalman gain detailed in Section 4-1, and the data-driven two-stage predictor detailed in Section 4-2. These algorithms were created with the aim of lessening the computational complexity of their conventional counterparts (respectively, data-driven Kalman filtering and the standard Kalman filter time-update) with as little detriment to performance as possible. The chapter explores the following points:

- Tuning the data-driven Kalman filtering algorithms: how the model order, sparsity pattern of the Markov parameters, and the amount of data affect the quality of the identified gain.
- Prediction performance: comparison of how each among standard and data-driven Kalman filtering, two-stage prediction, and MVM fare for varying model qualities and measurement noise variances.
- Execution times: analysis of the run times of each algorithm for increasing system dimensions.

A short summary of the main results concludes the chapter. As mentioned above, both the Matrix-Vector-Multiplication (MVM) approach and the standard Kalman filter that uses the Discrete-time Algebraic Riccati Equation (DARE) will serve as baselines for comparison. The former approach, MVM, is described in its own subsection below.

For the analyses that follow, the setting is the following:

- The adopted sensor geometry is the Fried geometry, and the arrays are all $L \times L$ square arrays of lenslets. That is, their width, given in number of lenslets, is L. Whenever "array width" is mentioned, it is given in lenslets.
- The measure of performance is the mean 2-norm (or sample variance, as the mean is removed) of the error, normalized by the mean 2-norm of the incoming wavefront, henceforth referred to as Normalized Mean Squared Error (NMSE). With mean-removed data from N time-steps, it is given by:

  $$\frac{\sum_{k=1}^{N} \left\| \widehat{\phi}_k - \phi_k \right\|_2^2}{\sum_{k=1}^{N} \left\| \phi_k \right\|_2^2} \tag{5-1}$$

  It is important to remove the mean at each time-step when evaluating performance, as offsets in the wavefront do not affect image quality, yet, if not removed, affect the mean squared error.
- Regarding the quality of the wavefront data available for model identification, both ideal and noisy cases will be considered. The ideal case assumes the identification dataset includes true (noiseless) wavefront phases, to identify the model, and slope data with measurement noise, to identify the gains. The noisy dataset is described below. The models used in the identification and prediction algorithms are obtained as in Subsection 2-3, with a penalty on the squared Frobenius norm of A when its absence leads to instability.
- The cases of single-layer and three-layer turbulence are considered. In both cases, the system is modelled using the procedure from Subsection 2-3-1, which incurs errors in the three-layer case, as described in Subection 2-2-2. The idea behind erroneous modelling of the three-layer case is to illustrate data-driven compensation for modelling errors.
- The turbulence parameters are set to standard values of $r_0 = 0.1$ and $L_0 = 25$ m. The algorithm itself is concerned only with the wind speed given in lenslets per time-step ω, which is given by

  $$\omega = \frac{L\nu}{Df_{\text{tel}}}$$

  where L is the width of the array, ν is the wind speed in meters per second, D is the diameter of the telescope in meters, and f_{tel} is its sampling frequency. For example, for $L = 80$, $\omega = 15$ m/s, $D = 8$ m, and $f_{\text{tel}} = 500$ Hz, the wind speed is 0.2 lenslets per time-step.
- The "default" system considered in this chapter consists of a 36×36 array of lenslets, subject to single-layer turbulence with a wind speed of 0.25 lenslets per time-step, with slope measurements with a Signal-to-Noise Ratio (SNR) of 10 dB. Each section may change any of these parameters, in which case, *only* the changed parameters are specified, unless reiteration is particularly relevant.

- When performance is addressed, Monte-Carlo simulations are performed with different datasets with the same underlying turbulence parameters, and different measurement noise sequences drawn from the same Gaussian distribution. The noise at each measurement point (lenslet) is additive, white, its expected value is zero, and its covariance matrix is diagonal. For the comparison of execution times, the different Monte-Carlo simulations are mere repeats of the same operation, usually for the same dataset; longer operations will involve fewer Monte-Carlo simulations.
- Unless specified otherwise, Juang and Chen's algorithm (Section 3-2), henceforth referred to as just "Juang", is used whenever the chosen Auto-Regressive (AR) order s is larger than 1, and MARK (Section 3-3) will be used whenever the order is set to 1. The gain for the coarse system of the two-stage algorithm is always identified with MARK with AR order 3, however. The number p of innovation-form Markov parameters is set to 2 in all cases.
- The data-driven gain identification procedure of Subsection 4-1-2 employs sparsity in either just the Markov parameters or both the Markov parameters and the identified gain itself. Whenever MARK or Juang are labelled "full", it refers to the former case, in which the gain is full, while Markov parameter sparsity is indeed exploited. The Markov parameters are only considered full when explicitly stated to be so.
- Simulation data is obtained with the OOMAO simulator [39]. We altered the code of the simulator to correct a bug that led to incorrect wind speeds whenever a wind speed over 1 lenslet per time-step was selected. Additionally, to avoid having interpolations affect the wavefronts, **all** datasets are from turbulent atmospheres simulated in a 4× denser grid such that the wind speeds are all integer lenslets per time-step. These were then downsampled to yield the stated wind speeds. For example, a wind speed of 0.25 lenslets per time-step is equivalent to 1 lenslet per time-step in a grid 4× as dense.

The Matrix-Vector-Multiplication (MVM) method

The MVM method evaluated here consists of pseudo-inverting the measurement matrix G using its truncated SVD, since it lacks full column rank, right-multiplying it by the latest measurement, and left-multiplying it by A:

$$\hat{\phi}_{k+1} = AG^{\dagger}y_k$$

The computation of the pseudo-inverse of G and its left-multiplication by A are performed offline. This is equivalent to wavefront *reconstruction* using $G^{\dagger}$, followed by a one-time-step progression using A:

$$\left.\begin{aligned} \hat{\phi}_{k|k} &= G^{\dagger}y_k \\ \hat{\phi}_{k+1|k} &= A\hat{\phi}_{k|k} \end{aligned}\right\} \quad \hat{\phi}_{k+1|k} = \hat{\phi}_{k+1} = AG^{\dagger}y_k$$

Note that the reconstructions $\hat{\phi}_{k|k}$ suffer principally from integration of the measurement noise:

$$G^{\dagger}y_k = G^{\dagger}G\phi_k + \underbrace{G^{\dagger}v_k}$$

where $G^{\dagger}G$ is **approximately** equal to the identity matrix. Implications of the main source of error of these wavefront reconstructions being the measurement noise present in the underbraced term, are that not only are MVM predictions generally better than the reconstructions, but, in particularly noisy cases, poorer models may actually yield better predictions. Consider single-layer turbulence: for fractional-pixel movements per time-step, A will consist approximately of a weighted average between two shifts of the wavefront (see Figures 5-1 and 5-2), which will smoothen out the measurement noise.

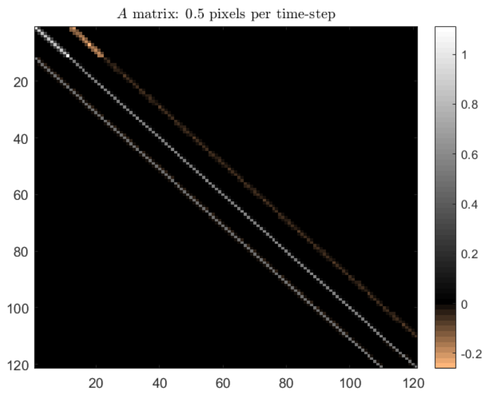

Figure 5-1: Example A for a wind speed of 0.5 pixels per time-step.

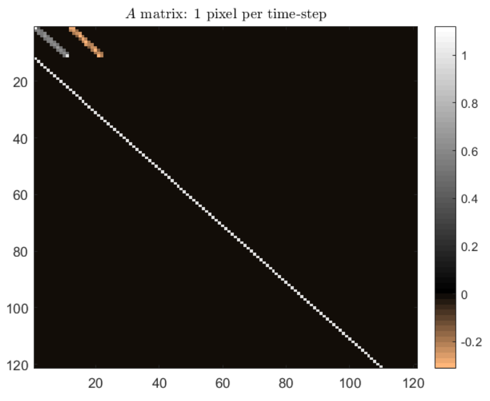

Figure 5-2: Example A for a wind speed of 1 pixel per time-step.

Consequences of this will be visible in plots that follow: in Figure 5-23, MVM predictions are worse as the wind speed approaches an integer value of pixels per time-step. In contrast, Kalman filtering performance is less susceptible to measurement noise, and will instead show worse results for worse models, with performance being maximal at integer-pixel movements per time-step.

Remark 5.1. The MVM approach can be formulated as resorting to regularized least-squares [4, 40]. However, it is also usually formulated without the product by an A as designed in Subsections 2-2-2 or 2-3-1. Multiplication by A was observed to not only generally yield a far better result, but also to eliminate the benefit of regularization.

Remark 5.2. The use of lagged AR-1 wavefront models is briefly explored in Appendix A-1. It is expected that lagged models improve Kalman filtering performance in certain conditions, yet drop MVM performance due to reduction (or elimination) of the averaging effect of A.

The noisy model identification dataset

The following is applicable to all subsections except for the one pertaining to how performance varies with the chosen AR order. Note that the procedure outlined here is meant solely to obtain a realistic noisy model on which to test the algorithms, and not to establish a standard for obtaining A.

For the noisy cases, the wavefront phase data is obtained by first pseudo-inverting G (using its truncated SVD) and multiplying it with the noisy slope data to obtain preliminary wavefront estimates to get an initial A. To improve the wavefront estimates, this preliminary A is used to identify a Kalman predictor, which retrieves better estimates of the wavefronts, using which A is re-identified.

Remark 5.3. Unless slope data for each individual layer is available, this approach for obtaining A is not applicable to multi-layer turbulence modelled with the augmented system given in (2-11). This because pseudo-inverting $[G \;\; G]$ and multiplying it to the slopes yields an estimate of the sum of the phase at all layers, rather than that of each individual layer, thus bringing us back to the erroneous model given in (2-5).

Remark 5.4. Rather than using a Kalman predictor, one could conceive of obtaining improved wavefront estimates using the reconstructor given in (3-27) instead. However, when obtaining the reconstructor gain from (3-29), it is assumed that the process and measurement noises are uncorrelated. Because the initial model is computed with wavefront estimates obtained from noisy measurements, the process and measurement noises are correlated.

5-1 Tuning the data-driven Kalman filtering algorithms

An analysis of the performance of the data-driven Kalman filtering algorithms as a function of its tuning parameters is given in this section. The performance is measured by the NMSE of the predictor built with the identified gain.

5-1-1 AR model order

The order s chosen for the AR model of the slopes, defined in Chapter 3, is of vital importance to the performance of the data-driven Kalman gain identification algorithms. Recall that the slopes are the output of the asymptotically stable state-space model given in (2-18), which means that they can be described by a high-order AR model. However, in practice, for identification of the Kalman gain, erroneous state-space modelling and the finite amount of available data limit the order one should choose.

Additionally, this is a parameter whose influence in the execution time is critical: as argued in Subsection 4-1-2, each unit increase in the AR order s introduces a new matricial Markov parameter to identify, each with more non-zeros than the previous. Even for small orders, increases in the model order add considerable overhead, with, for example, $s = 3$ generally involving more than double the non-zeros of $s = 2$. It is paramount, then, to evaluate how low s can be kept while maintaining performance. Note also that setting $s = 1$ is only possible with the MARK algorithm.

Only in this section, the noisy wavefront dataset for identification of A is obtained by adding Gaussian white noise to the ideal dataset with varying SNR to show how the model quality changes the optimal AR order. The SNR of the model identification dataset (comprising wavefront data, not slopes) is indicated by "model SNR" in the plots, and should not be confused with the SNR of the measurement data (which is used to identify the Kalman gain, rather than A).

To ensure correctness of the noiseless model, the turbulence wind speed is set to 1 lenslet per time-step (approximately 1 pixel per time-step). Figures 5-3 to 5-5 show performance for

AR orders s up to 5 and varying amounts of noise in the model identification dataset. The dashed blue lines ease comparison between MARK with order $s = 1$, and Juang with order $s = 2$, since the latter does not allow $s = 1$.

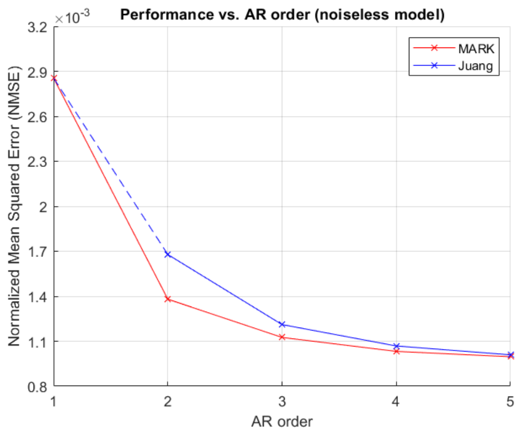

Figure 5-3: Data-driven Kalman filtering performance as a function of the AR order s (noiseless model).

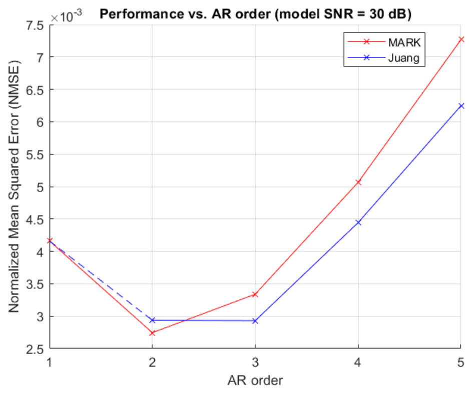

Figure 5-4: Data-driven Kalman filtering performance as a function of the AR order s (model SNR = 30 dB). "Model SNR" refers to the SNR of the phase data from which the model is identified.

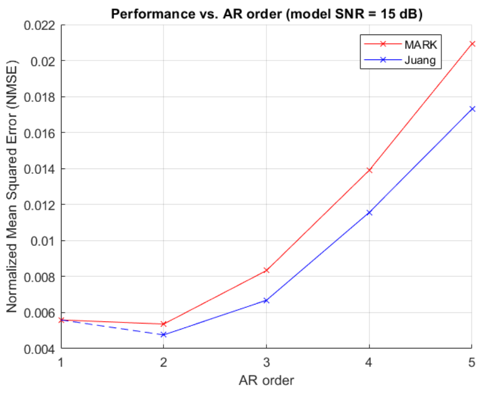

Figure 5-5: Data-driven Kalman filtering performance as a function of the AR order s (model SNR = 15 dB). "Model SNR" refers to the SNR of the phase data from which the model is identified.

Given an ideal model, increasing the order is expected to improve the results. As the model quality lowers, higher orders worsen in performance more quickly than lower orders: for both the noisy cases of Figures 5-4 and 5-5, $s = 2$ boasted the best results. Figure 5-6 below displays all results on the same axes, along with the results of the DARE.

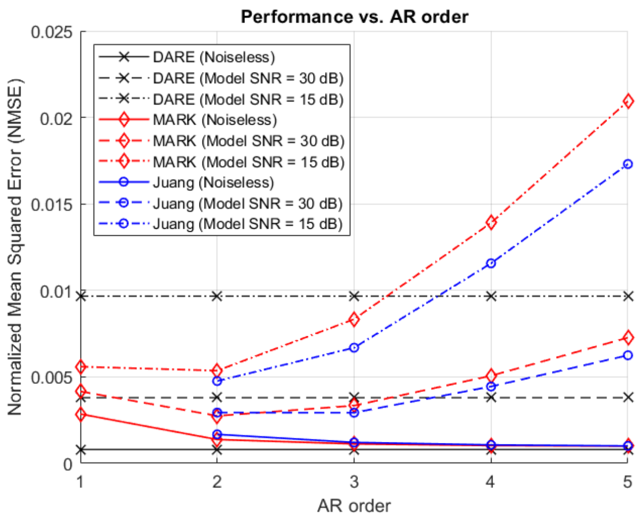

Figure 5-6: Performance of standard and data-driven Kalman filtering for varying AR order and model quality. "Model SNR" refers to the SNR of the phase data from which the model is identified.

The higher the model order, the less robust it is shown to be to erroneous modelling, with each choice of order seeing its performance deteriorate more quickly than smaller orders. Notice that gains identified with adequate orders outperformed the DARE ones for these noisy models, illustrating compensation for modelling errors.

Moreover, the noisier the measurements, assuming an ideal model, the higher the AR order must be for the performance to match that of the DARE. Indeed, it should be expected that noisier measurements demand larger orders to adequately smoothen out the noise with previous information. Subsection 5-2-3 illustrates this further. Less intuitively, more measurement noise may incur additional robustness to modelling errors; when the measurements are noisier, the predictions rely **more** on the measurements for further smoothing, reducing the weight given to the model, hence making performance less susceptible to noise in the model. When the measurement noise is low, the predictor expects to have an accurate prediction of the new data that entered at the borders shortly after it does, trusting the model afterwards.

The choice of AR order s will balance on three principal factors, then: the execution time, which rises substrantially with s, driving the user to keep it as low as possible; the user's confidence on the model, which allows for larger orders to be chosen for better performance; and the measurement noise, which normally incurs necessity for larger orders. Orders $s = 1$ and $s = 2$ are also particularly useful to improve A from a bad preliminary estimate, as done in the noisy model cases.

Remark 5.5. Figures 5-3 to 5-6 establish as well that while MARK might perform marginally better for ideal and low-noise models, Juang is more robust to modelling errors.

AR order for fractional wind speeds

In Section 2-2-2, it was established the presented AR-1 modelling technique is expected to perform worse when the turbulent wavefront moves a fraction of a pixel each time-step; the further from the integers, the worse. As stated earlier, higher values of s behave poorly when the model is inadequate, meaning that for fractional pixels (or, approximately, lenslets) per time-step, even when the model is noiseless, a behavior akin to that shown in Figures 5-4 and 5-5 is expected. Figures 5-7 and 5-8 show the results for a wind speed of 0.25 lenslets per time-step, up to $s = 4$.

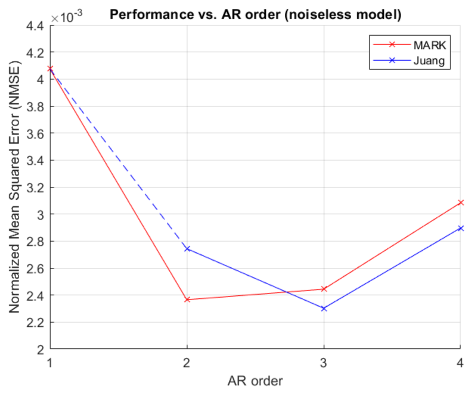

Figure 5-7: Data-driven Kalman filtering performance as a function of the AR order s (sub-pixel movement).

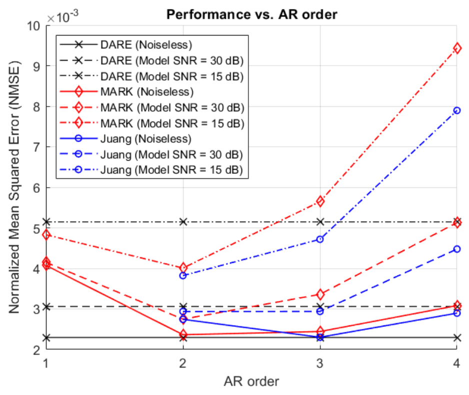

Figure 5-8: Performance of standard and data-driven Kalman filtering for varying AR order and model quality (sub-pixel movement).

It is indeed evident that when the wind speed, in lenslets per time-step, is fractional, arbitrarily increasing s does not lead to better performance due to the inherent AR-1 modelling error. The same applies to the multi-layer case, if modelled as (2-5), and for sparse gains when the Kalman gain is dense instead.

AR order for sparse gains and two-stage prediction

When identifying a sparse gain for the two-stage procedure, the imposition of sparsity when the Kalman gain is not sparse is effectively analogous to erroneous modelling in that s should not be increased arbitrarily, lest the performance deteriorate. Figures 5-9 and 5-10 show the NMSE for prediction with both the identified Kalman filter and with the two-stage predictor for $s = 2$ and $s = 3$. The system here considered is single-layer turbulence sampled by a 120×120 array, and the performance is shown as a function of the distance-constraint z of the gain and the coarse width L_c.

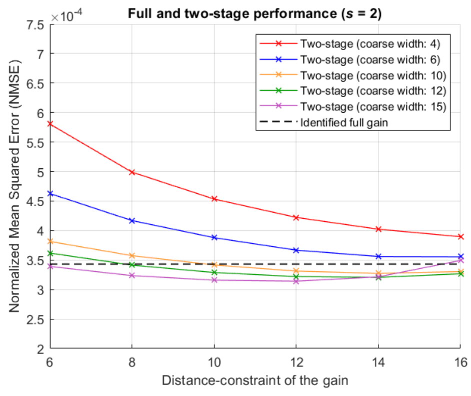

Figure 5-9: Two-stage prediction performance as a function of z and L_c ($s = 2$).

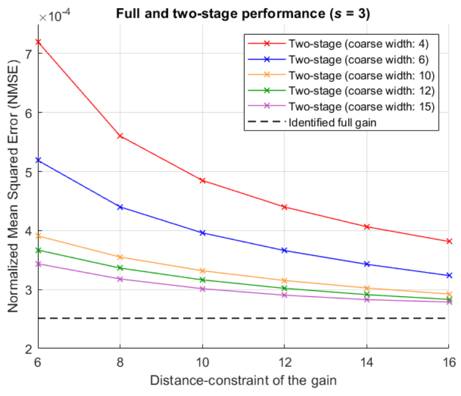

Figure 5-10: Two-stage prediction performance as a function of z and L_c ($s = 3$).

While the increase in s improved the results for the full filter, it worsened them for most of the differently-tuned two-stage predictors, improving only those where both the coarse width and the distance-constraint of the gain were set high. Nonetheless, near-best performance can be attained for $s = 2$, with some error over the best attainable performance with the full filter. This increase in error will have to be considered against the improvement in prediction time (and storage) that defines the merits of the two-stage approach.

5-1-2 Data batch length

The assumed sparsity of the Markov parameters and the consequential drastic decrease of the number of variables to identify for their retrieval leads not only to the benefits in execution time shown above, but also to a reduction of the necessary amount of data for identification of the Kalman gain. Additionally, showing that the necessary amount of data is determined by the non-zeros per row, rather than the number of outputs m, is fundamental to argue that the scalings $\mathrm{O}(Nmw^2)$ and $\mathrm{O}(Npmw^2)$ indeed represent linearity with respect to m (see Subsection 4-1-2).

The system parameters considered herein are the default values described in the chapter introduction. The SNR of the slope data is 10 dB. Figure 5-11 illustrates the performance of Juang with $s = 2$ for growing amounts of data and distance-constraints of 1.5, 2.5, 3.5, and 4.5 lenslets.

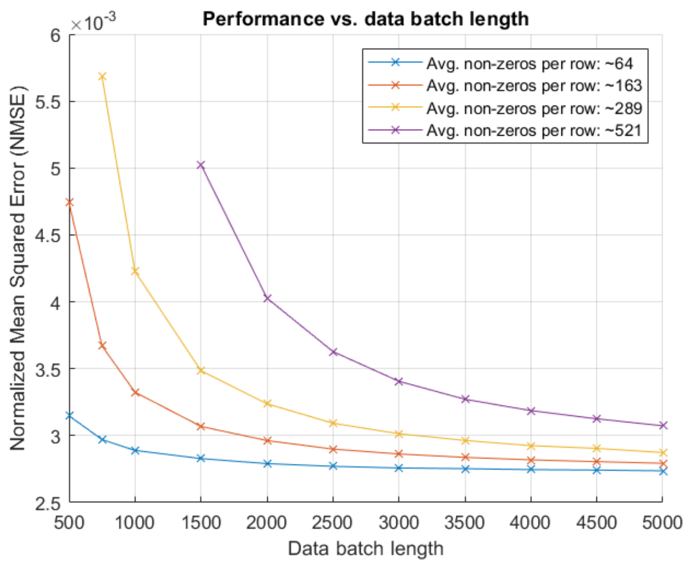

Figure 5-11: Data-driven Kalman filtering performance for varying data batch length and distance-constraint of the Markov parameters. The legend indicates the average non-zeros per row for the distance-constraints of 1.5, 2.5, 3.5, and 4.5. Some points are missing due to instability of the filters when the amount of data is too low for the amount of non-zeros.

The increasing flatness as the distance-constraint is reduced illustrates the dependency of the required amount of data on the average non-zeros per row. It should be noted that attempts to identify gains **without** exploiting sparsity of the Markov parameters up to a data batch length of $N = 5000$ always yielded unstable filters.

The dependence of the amount of data on the number of non-zeros rather than the system dimensions means that "tall" datasets, with more measurements than time-steps of data, are entirely suitable for data-driven identification of the Kalman gain, whereas any covariances computed (say, for the DARE) would never have full rank. That the identification algorithms work for the large systems of Subsection 5-4 supports this point.

Measurement noise also affects the necessary amount of data: the less the noise, the flatter the curve, and the quicker the best performance is attained. Figure 5-12 shows, for a fixed Markov parameter distance-constraint of 1.5, the NMSE as a function of the data batch length for varying measurement SNR. Because, naturally, larger measurement noises incur more error, even for hypothetical infinite datasets, the curves had their minima subtracted so that their flatness can be directly compared.

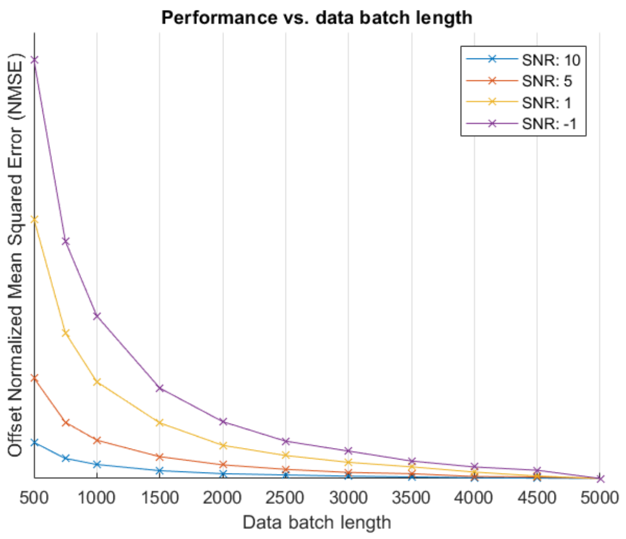

Figure 5-12: Data-driven Kalman filtering performance for varying data batch length and measurement SNR. The curves had their minima subtracted so that the flatness can be easily compared, hence the omission of the y-axis scale.

5-1-3 Markov parameter distance-constraint

The distance-constraint of the Markov parameters (again, measured in lenslets) is another paramount parameter to evaluate. The principal merit of the methodology for identification of the Kalman gain presented in this thesis is its applicability to large-scale AO systems, which is a result of the sparsity of the Markov parameters, and therefore reliant on the chosen distance-constraint. Figures 5-13 to 5-16 show how performance depends on the constraint for different wind speeds.

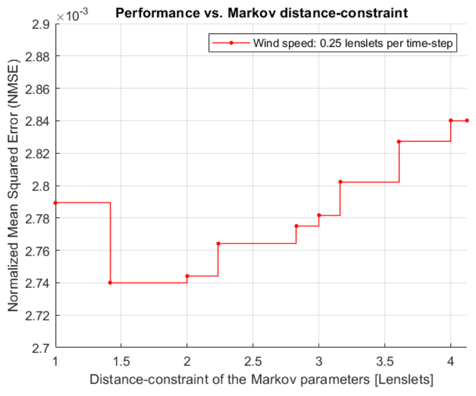

Figure 5-13: Data-driven Kalman filtering performance vs. Markov parameter distance-constraint (0.25 lenslets per time-step). Notice the tight range of the vertical axis: because of the low speed and the constraint beginning at 1 lenslet, performance varied little.

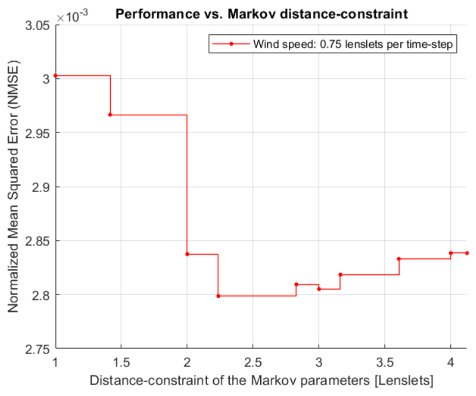

Figure 5-14: Data-driven Kalman filtering performance vs. Markov parameter distance-constraint (0.75 lenslets per time-step).

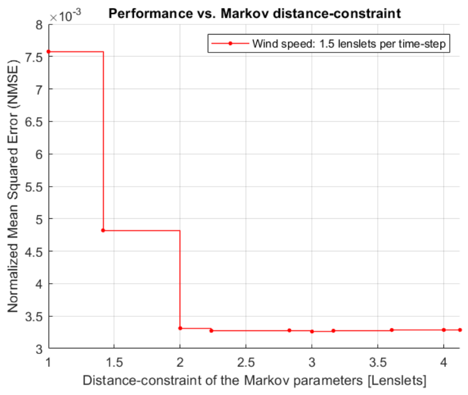

Figure 5-15: Data-driven Kalman filtering performance vs. Markov parameter distance-constraint (1.5 lenslets per time-step). Minimal error was achieved for a constraint of 3.

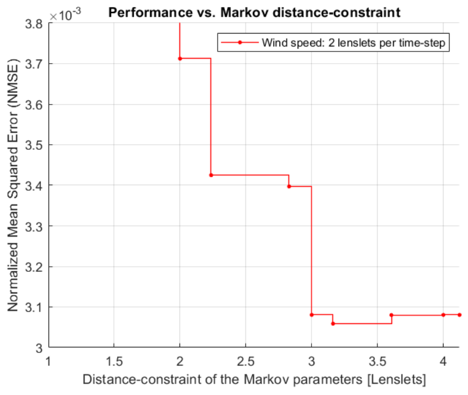

Figure 5-16: Data-driven Kalman filtering performance vs. Markov parameter distance-constraint (2 lenslets per time-step). The distance-constraint of 1 resulted in an unstable gain, and far over the limits of the plot, 1.5 resulted in a NMSE of 1.48.

The distance-constraint of the Markov parameters should be just large enough to account for wind speed and to accommodate the spatial smoothing of the noise done by the Kalman filter, which, as evident in the figures, is such that the constraint *should* exceed the wind speed in lenslets per time-step. Under-estimations severely worsen performance. Over-estimations above the optimal constraint add unnecessary parameters to identify, slightly negatively impacting performance.

Remark 5.6. Notice the slight jitter visible in Figures 5-14 and 5-15. In both examples, the error increases from constraint $\sqrt{5} \approx 2.24$ to $\sqrt{8} \approx 2.83$, and decreases further at constraint 3. This because the simulated turbulence moved horizontally, and increasing the constraint from 2.24 to 3 adds lenslets horizontally (and vertically), along the direction of the movement, while increasing it to $\sqrt{8} \approx 2.83$ adds lenslets diagonally instead. These latter ones proved unnecessary, and thus merely added more parameters to identify, decreasing performance.

5-2 Prediction performance

This section explores and compares the performance of the different prediction algorithms (Kalman filter, two-stage predictor, MVM), and of the two discussed methods for obtaining the Kalman gain (DARE and data-driven). Two-stage performance is presented as a function of the distance-constraint z of the sparse gain and the width L_c of the coarse system.

All considered systems consist of single-layer turbulence sampled by 36×36 lenslet arrays. The wind speed is 0.25 lenslets per time-step, and the Markov parameter distance-constraint is fixed at 1.5 in all cases. Results are shown for measurement SNRs of 10 and 5 dB.

5-2-1 Single-layer turbulence

Figures 5-17 and 5-18 illustrate performance for a noiseless and noisy 36×36 model, respectively, and a measurement SNR of 10 dB. The AR order is set to $s = 3$ in both cases.

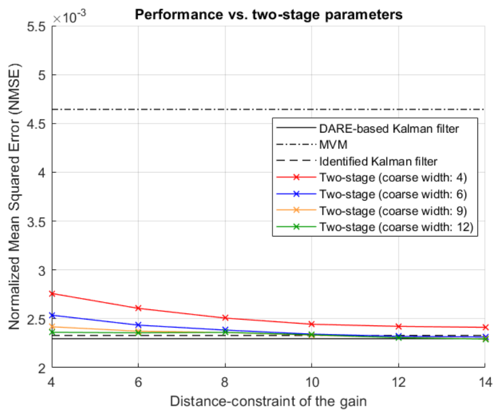

Figure 5-17: Predictor performances: single-layer turbulence, SNR = 10 dB, noiseless model

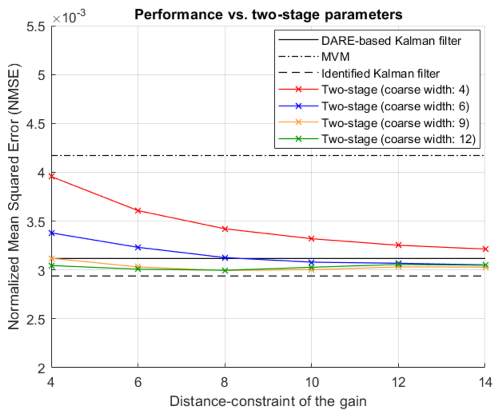

Figure 5-18: Predictor performances: single-layer turbulence, SNR = 10 dB, noisy model

The performances for a measurement SNR of 5 dB are shown in figures 5-19 (noiseless model) and 5-20 (noisy model). The orders are $s = 3$ in both cases.

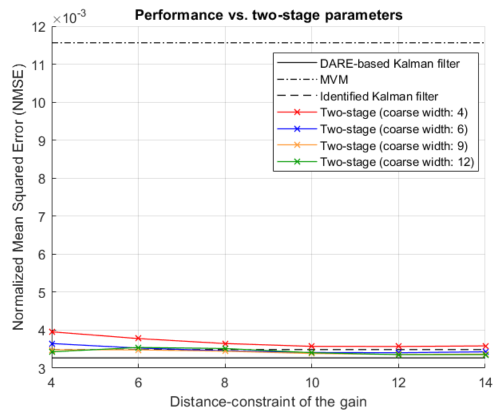

Figure 5-19: Predictor performances: single-layer turbulence, SNR = 5 dB, noiseless model.

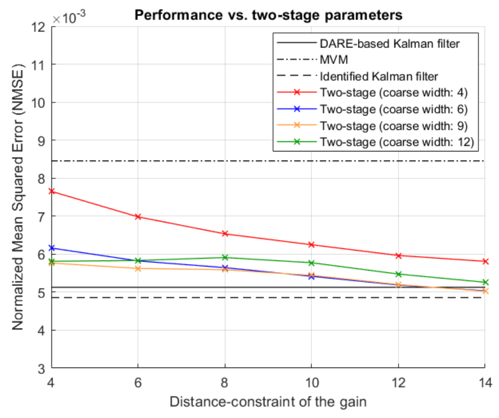

Figure 5-20: Predictor performances: single-layer turbulence, SNR = 5 dB, noisy model

The identified Kalman gain is able to match the DARE gain in performance in the noiselessly modelled cases, while improving upon it when the model is noisy. The two-stage predictor is seen to approximate the performance of the identified Kalman filter as the width L_c of the coarse array is increased and the distance-constraint z of the gain is relaxed. In all of the cases, the basic MVM performed the worst.

5-2-2 Multi-layer turbulence

AR-1 models for the dynamics of three-layer turbulence, designed as per Subsections 2-2-2 and 2-3-1, are inadequate for Kalman filter design due to their colored process noise. Figure 5-21 demonstrates how data-driven Kalman filtering compensates for this shortcoming, with $s = 3$.

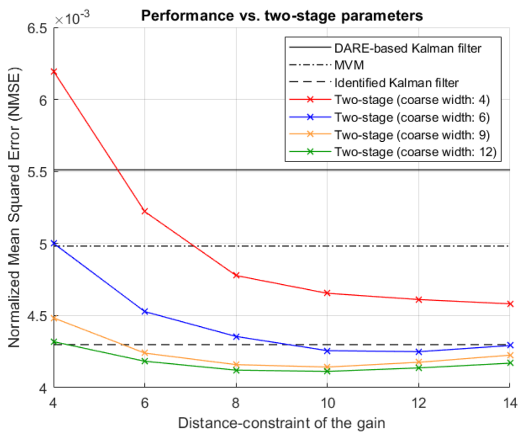

Figure 5-21: Predictor performances: multi-layer turbulence, SNR $= 10$ dB, noiseless model

The identified Kalman filter and two-stage predictor now both boast better performance than the DARE-based Kalman filter and MVM. An implication is that, even if the separation into stages and subsequent interpolation are done at the expense of performance, the data-driven compensation for modelling errors is applicable as well to the two-stage predictor. In fact, the two-stage algorithm behaved better than the identified Kalman filter for some of the parameter choices.

Remark 5.7. Albeit not the case in Figure 5-21, given the inherent modelling error in multi-layer cases (see Subsection 2-2-2), the DARE gains identified for noisy multi-layer models may be better than those identified for noiseless models. This because noise in the model identification dataset results in a flattened A matrix and an enlarged process noise covariance Q, thus reducing the importance of the model in the Kalman filter in favor of the measurements; since the model is nonetheless inherently wrong, performance might see a boost. In any case, a similar effect can be achieved by artificially reducing the measurement noise covariance matrix R, increasing the weight of the measurements.

5-2-3 Increasing wind speed

This subsection deals with how performance tends to vary with increasing wind speed. Take note that although adequate parameters were sought, the tuning might not be optimal at all wind speeds (namely in Figure 5-23) because a balance was struck between performance and time, and high AR orders are cumbersome for the identification procedure.

For the two-stage algorithm, the distance constraint of the gain and the coarse width were both kept at 6. The results are shown in Figures 5-22 and 5-23.

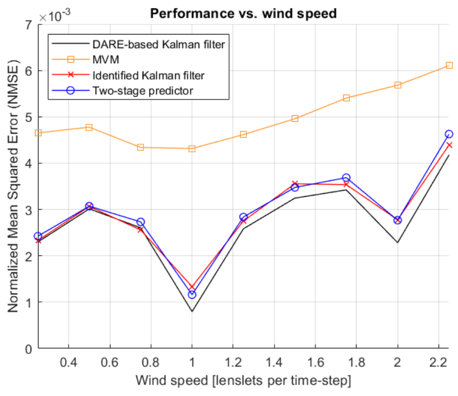

Figure 5-22: Predictor performances for varying wind speed (measurement SNR = 10 dB).

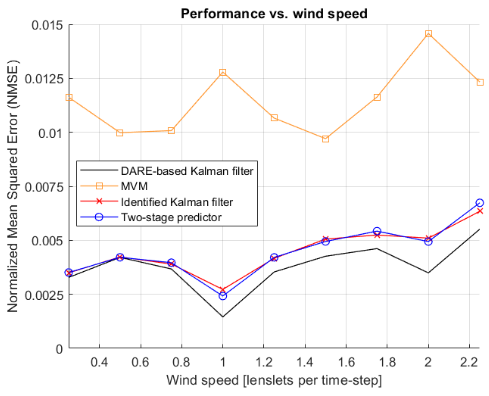

Figure 5-23: Predictor performances for varying wind speed (measurement SNR = 5 dB).

An increasing trend in the error is visible for all algorithms, with the identified Kalman filter and the two-stage algorithm consistently displaying less error than MVM, and performance similar, albeit worse, to that of the DARE-based filter. Naturally, for worse measurement noise (lower measurement SNR), higher orders are required to keep up with the DARE, which is reflected in the larger discrepancy seen in Figure 5-23.

Remark 5.8. Notice in Figure 5-23 how the quality of MVM increases as the wind speed becomes more distant from integer lenslets-per-time-step. As explained in the chapter introduction, the main reason for the lack of quality of MVM predictions is the integration of the measurement noise, and the state-transition matrices $\widehat{A}$ at these speeds are averaging some of this noise out.

Remark 5.9. In the ideal single-layer case, whenever the two-stage algorithm outperforms prediction with the full identified gain, expect that performance-oriented tuning would improve the results further, particularly for the full case, until it is better than two-stage. Two-stage prediction happens to be more robust to under-selected AR orders.

Figure 5-24 shows performance varying with wind speed for the mismodelled multi-layer case.

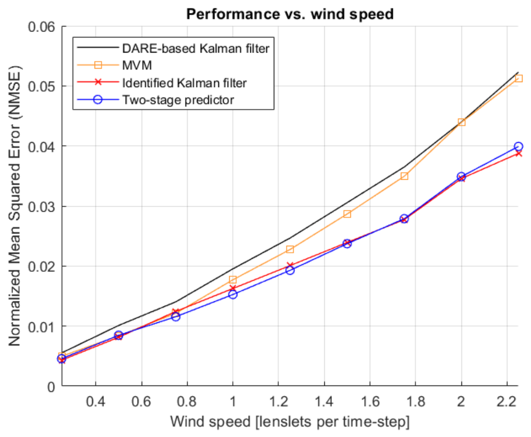

Figure 5-24: Predictor performances for varying wind speed (multi-layer turbulence).

The data-driven procedures now consistently boast better performance than the DARE and behave better than MVM as speed increases. Note that MVM outdid two-stage by a slight margin for a wind speed of 0.5 lenslets per time-step, which is when the modelling error due to fractional pixel-wind-speeds is at its worst.

5-2-4 Sparse and two-stage

Two-stage prediction consists of a sparse stage in the original grid, for local prediction, and a coarse one for the prediction of the overarching shape of the wavefront. This subsection discusses the necessity of both stages by the sparse stage alone to the two-stage algorithm. The coarse stage alone, with linear interpolation, naturally leads to very poor results.

Depending on the quality of the model and on the distance-constraint in comparison to the array width, the sparse stage might suffice for prediction. When the model is of a high quality, the Kalman filter relies upon it more heavily than otherwise, reducing the importance of the measurements. Because the weight put upon the measurements, as seen in (2-20), is the Kalman gain itself, reduced reliance on the measurements is reflected in a smaller, sparser gain. This means that, for small arrays, similarly small distance-constraints suffice for near-optimal prediction when the model is reliable; for instance, in the noiselessly modelled 36×36 case of Figures 5-17 and 5-19, the sparse gain with a constraint of 6 performs most of the prediction, with the coarse stage improving it only slightly:

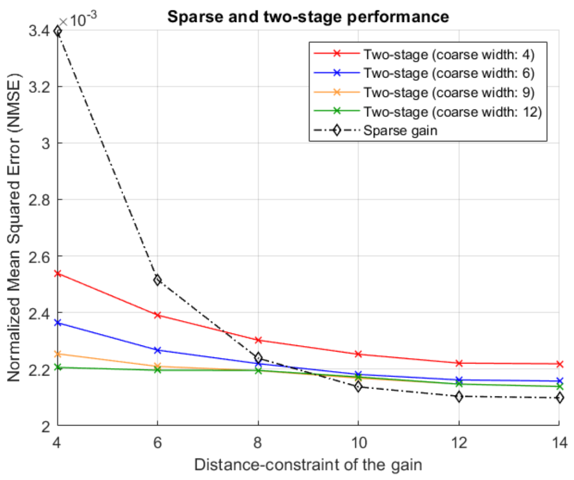

Figure 5-25: Performance comparison between sparse and two-stage prediction (36×36, noiseless model).

The utility of the coarse stage becomes evident once faulty models are introduced, as the measurements will be given increased importance, rendering the sparse stage insufficient for correction. The utility of the coarse stage is demonstrated in Figure 5-26 for a noisy model; the same is applicable to erroneous models of the natures described in Subsection 2-2-2.

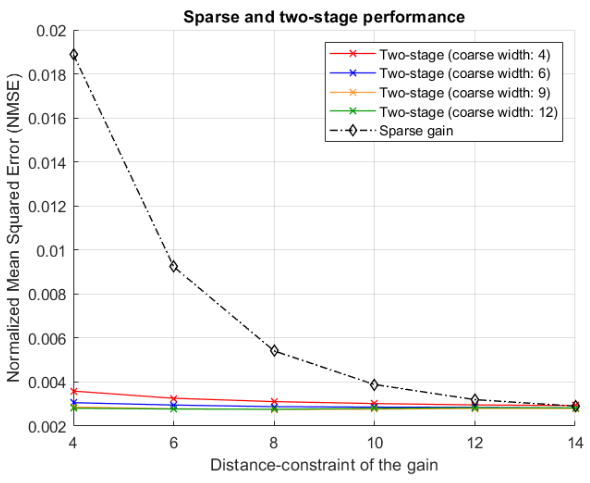

Figure 5-26: Performance comparison between sparse and two-stage prediction (36×36, noisy model).

Furthermore, most noticeably for very large system dimensions, the coarse stage allows the constraint to be kept tight, resulting in a remarkably high sparsity of the gain. Figure 5-27 shows results for a noiselessly modelled 120×120 system:

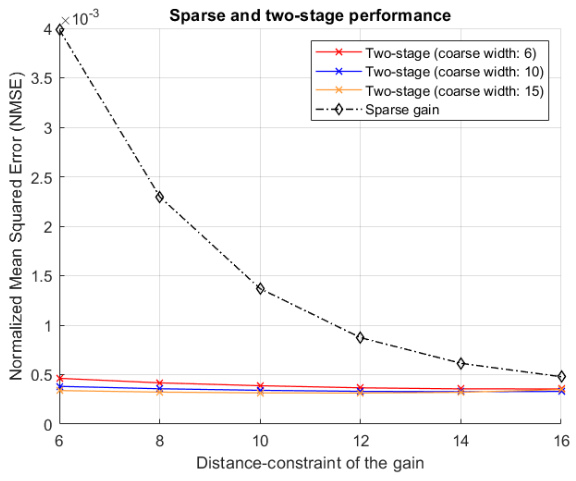

Figure 5-27: Performance comparison between sparse and two-stage prediction (z, noiseless model).

Notice how a coarse stage of width $L_c = 10$, which is less than the square root of the original array width, reduces the error from using only the sparse stage down to almost the best attainable performance, even for tight distance-constraints.

5-3 Execution time

The increases in running time of the different algorithms with the system dimensions are compared in this section. The main takeaway should not be the absolute times, but rather how quickly these rise for each algorithm as the system becomes larger. Nonetheless, the algorithms were implemented in MATLAB version 9.6.0.1150989 (R2019a), and executed on a system with an Intel Core i7-8750H CPU at 2.20 GHz, 24 GB of RAM, running Windows 10.

5-3-1 Identification

Figures 5-28 and 5-29 compare execution times for the DARE (implemented in MATLAB's `idare` function), and full and sparse identification with both MARK ($s = 1$) and Juang ($s = 2$ and $s = 3$), with the base Markov parameter distance-constraint set to 1.5.

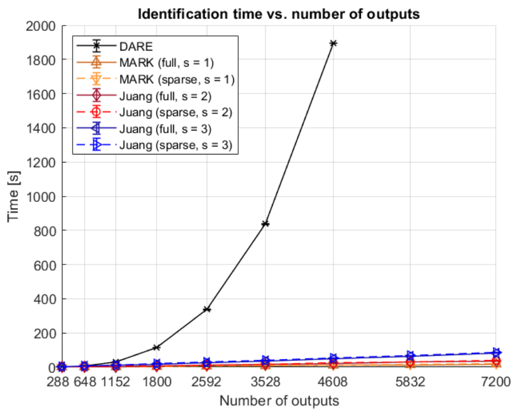

Figure 5-28: Execution time comparison between MATLAB's `idare`, and MARK and Juang for multiple values of s. The error bars have a length of two standard deviations.

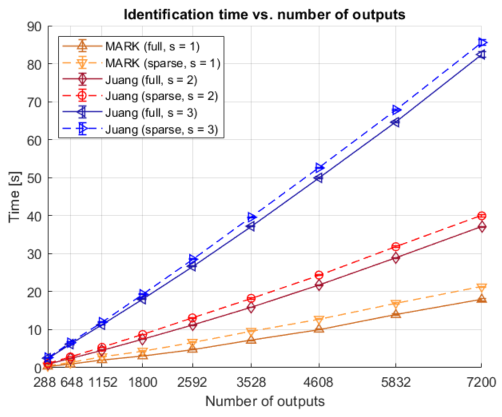

Figure 5-29: Execution time comparison between MARK and Juang for multiple values of s. The error bars have a length of two standard deviations.

The DARE sees its execution time quickly rise far above that of the identification procedures. Notice that for the considered dimensions, the sparse gains take slightly longer to identify than full gains: the high sparsity of the observability matrix is already such that the least-squares problem for the gain can be solved quickly without enforcing sparsity in the gain itself, so the overhead due to the repeated QR-factorizations necessary for algorithm (4-3) dominates.

As for the difference between MARK and Juang when both use their minimal s and p, MARK with parameters $s = 1$ and $p = 2$ solves least-squares problems with matrices of the same size as those for Juang with parameters $s = 2$ and $p = 2$, so the absence of the intermediate conversion from observer-form to innovation-form Markov parameters (step 3 of Juang as per the summary of Section 3-2) could be furnishing a, nonetheless insignificant, reduction in execution time. Most importantly, a trade-off is made between more QR-factorizations (MARK) or QR factorizations of bigger matrices (Juang); because the time to perform these QR-factorizations scales with the square of the non-zeros per row, Juang's increased non-zeros per row offset MARK's increased amount of QR-factorizations.

Remark 5.10. For extremely large systems, as shown in Section 5-4 for arrays of width equal to and over 90 lenslets, it is no longer the case that full identification is faster than sparse.

5-3-2 Prediction

As far as prediction goes (as per (3-3) or (3-4)), once the Kalman gain is obtained, it is irrelevant whether the DARE or an identification procedure were used to obtain it, as the operation is the same. As such, this subsection will only distinguish full and sparse gains, the two-stage operation, and MVM. Figure 5-30 compares the times taken for the *online* prediction operation with the full gain and a sparse one, MVM, and the two-stage procedure. Figure 5-31 isolates the sparse gain and the two-stage procedure.

The state-transition matrix A is sparse, and the distance-constraint for the sparse gain and the width of the coarse system are both kept constant at 6 throughout. The removal of the means done every iteration is included in the displayed times. Particularly, in the two-stage procedure, the means of all components are removed; that is, $\hat{\phi}_k$, $\hat{\phi}_k^{(\mathrm{f})}$, and $\hat{\phi}_k^{(\mathrm{c})}$ all have their means removed.

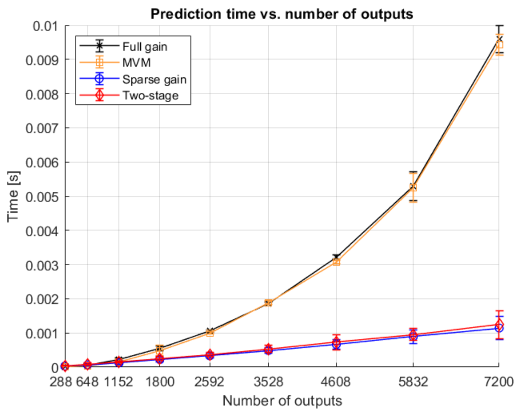

Figure 5-30: Prediction time comparison between MVM, full and sparse time-update, and the two-stage algorithm. The error bars have a length of two standard deviations.

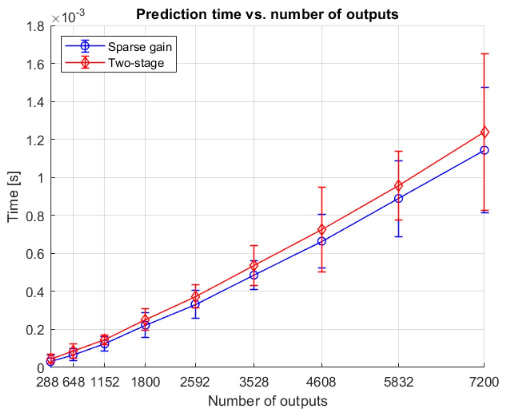

Figure 5-31: Prediction time comparison between the sparse time-update and the two-stage algorithm. The error bars have a length of two standard deviations.

The times taken for full Kalman filtering and MVM rise significantly faster with the width than those for sparse and two-stage filtering, as expected with their m^2 scaling. Figure 5-31 shows that the coarse system operations of two-stage prediction add little burden to sparse prediction for fixed distance-constraint and coarse system width.

5-4 Extremely large systems

Tables 5-1 and 5-2 show the algorithms at work for far larger dimensions than previously. The wind speed is 0.25 lenslets per time-step, the measurement SNR is 10 dB, and the Markov parameter distance-constraint is set to 1.5 lenslets. The order s is set to $s = 2$. The wavefront and slope datasets consist of only 5000 time-steps of data, while the numbers of states range from 8281 to 32761, and the numbers of outputs from 16200 to 64800; this means that all the datasets are "tall", and that any covariances estimated with these datasets would only be positive semi-definite (and not definite).

Note that, although z was kept constant at 6 here, both z and L_c should be tuned in an attempt to avoid either becoming too large, striking a balance between them.

Dimensions	L_c	Time for `pinv`(G) [s]	Ident. time [s]		Prediction time [s]			
			Full	Sparse	MVM	Full	Sparse	Two-stage
90×90	10	204.6	210.5	169.7	0.043	0.043	0.0031	0.0033
120×120	10	1067.5	492.6	333.6	0.14	0.13	0.0053	0.0057
150×150	10	4085.5	—	679.3	0.32	0.32	0.0084	0.0089
	15							0.0092
180×180	10	—	—	995.3	0.67	0.69	0.012	0.016
	15							0.014
	20							0.014

Table 5-1: Identification and prediction times for very large systems. Distance-constraint z of the gain was fixed at 6. Missing results (marked by a dash) are due to either insufficient memory to handle identification of the full K, or the large time taken to pseudo-invert G. Gray cells indicate hypothetical results, where the operations were done with artificial matrices.

Dimensions	L_c	Normalized Mean Squared Error (NMSE)			
		MVM	Full	Sparse	Two-stage
90×90	10	1.22×10^{-3}	4.82×10^{-4}	1.93×10^{-3}	4.78×10^{-4}
120×120	10	8.98×10^{-4}	3.43×10^{-4}	4.00×10^{-3}	3.62×10^{-4}
150×150	10	1.11×10^{-3}	—	6.21×10^{-2}	5.19×10^{-4}
	15				4.54×10^{-4}
180×180	10	—	—	2.65×10^{-2}	5.65×10^{-4}
	15				4.83×10^{-4}
	20				4.50×10^{-4}

Table 5-2: Prediction performance for very large systems. Distance-constraint of the gain was fixed at 6. Distance-constraint z of the gain was fixed at 6. Missing results (marked by a dash) are due to either insufficient memory to handle identification of the full K, or time-consuming pseudo-inversion of G.

Two-stage prediction shows execution times up to almost two orders of magnitude smaller than those taken by MVM or the full time-update. As for identification, it is already faster than pseudo-inverting G via SVD (using MATLAB's `pinv`) for arrays of a width of 120 lenslets; in particular, for these large systems, identification of the sparse gain is quicker than that of the full gain. Setting the sparsity patterns and the matrices required for the two-stage operation, i.e. $(A - \overline{K}G)', K', M_c, M_i$ among others, takes comparably negligible time, but note that low coarse widths may reduce the sparsity of M_c and M_i, possibly increasing prediction time (see the two-stage times for the 180×180 array).

Two-stage prediction proves itself far better than just the sparse time-update, with Table 5-2 showing reductions in the NMSE of up to almost two orders of magnitude. It is also consistently better than simple MVM while incurring much smaller prediction times. Subsection 5-5 further elaborates these points.

5-5 Two-stage overview

The previous subsections have established that the two-stage algorithm allows the user to customize a balance between error and execution time. The figures herein display surface plots of the two-stage NMSE as a function of the distance-constraint z of the gain, and the coarse width L_c with execution time generally increasing with both z and L_c.

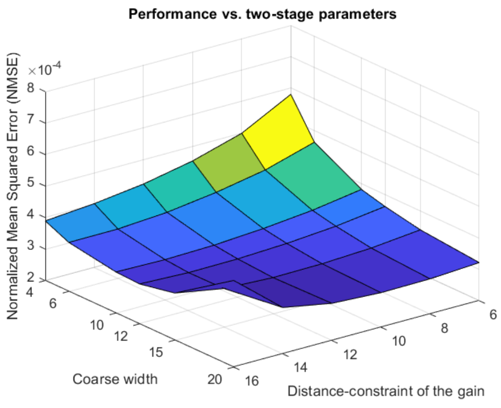

Figure 5-32: Two-stage performance for varying distance-constraints z and coarse widths $(120 \times 120, s = 2)$.

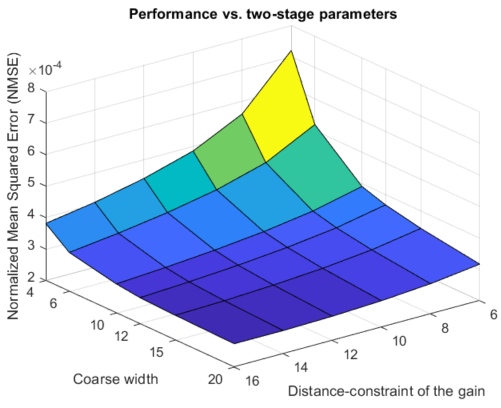

Figure 5-33: Two-stage performance for varying distance-constraints z and coarse widths L_c $(120 \times 120, s = 3)$.

In combination with Figures 5-9 and 5-10, it is visible that, even for very large systems, the two-stage performance is near-maximal for small z and L_c. In the 120×120 example, $z = 10, L_\mathrm{c} = 10$, both of which are smaller than $\sqrt{120} \approx 11$, provide performance close to that of the identified Kalman gain, and significantly better than MVM (which had an NMSE of 8.98×10^{-4} not shown in the plots).

While increasing the order s improves the results at larger values of z and L_c, it does so at the expense of performance for lower values. The two-stage algorithm always implies a trade-off between time and performance, and the general undesirability of large coarse widths or lax distance-constraints due to the resulting larger execution times means that lower values of s should be favored, which also leads to faster identification.

It is nevertheless hard to tell how free the distance-constraint z of the gain and the width L_c of the coarse system really are, as the coarse width must divide the original width, limiting the choices; if, say, the hypothetical "best" choice of coarse width were $L_\mathrm{c} = \sqrt{L}$, such a claim could only be evaluated if the studied original system widths were squares of integers, which very quickly grow out of hand. Tables 5-1 and 5-2 do nonetheless show that, employing the two-stage procedure, it is possible attain great identification and prediction times for very large systems, even if it is not conclusive that, for satisfactory performance, these truly do scale linearly with the system dimensions.

Figures 5-34 and 5-35 extend the analysis to a 150×150 array, showing similarly good performance for early values of z and L_c.

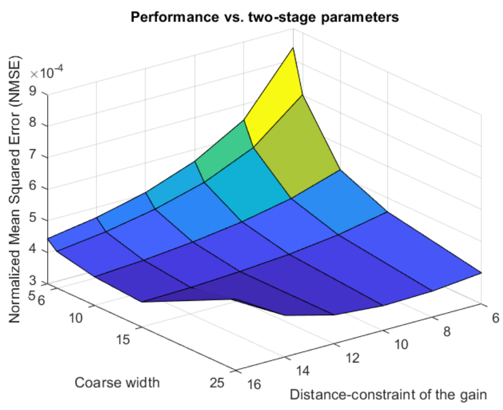

Figure 5-34: Two-stage performance for varying distance-constraints z and coarse widths L_c $(150 \times 150, s = 2)$.

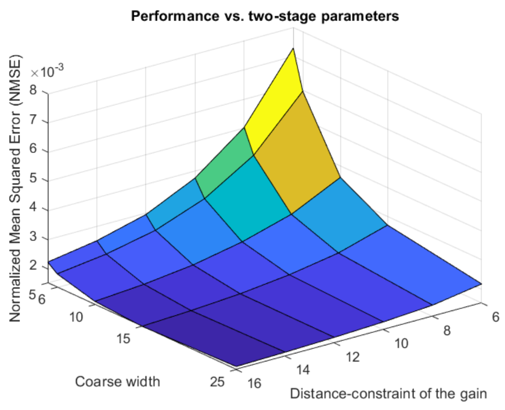

Figure 5-35: Two-stage performance for varying distance-constraints z and coarse widths L_c $(150 \times 150$, noisy model, $s = 1)$.

5-6 Summary

This chapter has provided examples and remarks to not only guide the tuning of the data-driven Kalman filtering and two-stage algorithms, but also to motivate their use. To conclude the chapter, the summary herein recapitulates the main conclusions drawn.

Tuning The order s of the AR model of the slopes should be chosen by weighing the following factors:

- Model quality, which if expected to be low should be accompanied by a likewise lower choice of s. Ideal models allow s to be arbitrarily increased for performance that asymptotically tends towards that of the DARE-based Kalman filter. Identification of the sparse constrained gain is behaves akin to a lower-quality model in that increases of s will eventually lead to worse performance.
- Execution time, which increases quickly with s, and should drive the user to keep s low.
- Measurement noise demands larger orders s, as AR model will attempt to smoothen out the noise using information from previous time-steps.

The required amount of data for identification is shown to be independent of the system dimensions, and dependent on the non-zeros per row (or column) of the Markov parameters instead. This means that datasets with less time-steps of data than measurement points are suitable for identification of the Kalman gain, whereas the consequential lack of full rank of any covariances computed with these datasets could be problematic for the DARE.

The base distance-constraint of the Markov parameters, used to enforce their sparsity pattern, can be decided based on the wind speed given in lenslets per time-step. The lower the wind speed, the tighter the constraint can be kept. It must be taken into account that the AR model will smoothen out measurement noise using information from adjacent measurements, so this constraint should exceed the wind speed (in lenslets per time-step).

The examples provided throughout the chapter establish functional selections of these tuning parameters for particular cases, which can serve as a starting point for application in different scenarios.

Performance and time The data-driven Kalman filter was demonstrated to be able to match its DARE-based counterpart in terms of NMSE, and the identification procedure was shown to scale far better than the DARE with the size of the system. However, matching the performance of the DARE should not be a strict goal, as depending on the characteristics of the turbulence, it may imply sacrificing speed; for example, if the measurements are very noisy, the DARE is matched only for high orders s, which will consequently substantially slow down identification. Instead, it should be kept in mind that one of the principal merits of this data-driven approach is its speed, and the choice of s should reflect this, even if it carries a slight cost in performance. In case A is erroneously designed, the data-driven filter proved itself capable of some compensation for the error, yielding better performance than the DARE-based Kalman filter.

As for the two-stage predictor, it always implies a sacrifice in performance in exchange for better scaling of the prediction operation. Its data-driven nature still carries the benefits of data-driven Kalman filtering, namely the and added robustness to modelling error with respect to the DARE-based filter, and its Kalman-filter-inspired structure still provides diminished susceptibility to measurement noise when compared to MVM. The main tuning parameters that determine error and time, the distance-constraint imposed upon the sparse gain and the width of the coarse array, can be kept low to attain short prediction times, all the while boasting competitive performance. As these parameters are increased, the NMSE tends towards that of the identified Kalman filter, with additional error as the inherently sub-optimal coarse stage becomes redundant. Furthermore, the two-stage predictor was shown to scale well for arrays of dimensions up to 180×180.

Note that the order s for identification of the Kalman gain of the coarse system was preset to $s = 3$ and left unchanged for simplicity of the analysis. This is, nonetheless, an additional tuning parameter, along with p for both the fine and coarse gain, but the latter is recommended to be kept at $p = 2$, as justified in Section 4-1.

Chapter 6

Conclusions and Recommendations

In Chapter 3, a novel algorithm for data-driven Kalman filtering was proposed. The algorithm is based on the subspace-identification equations, and stands beside Juang and Chen's algorithm [34] as a prediction-error method for direct identification of the Kalman gain (as opposed to identification of the noise covariances). In the context of Adaptive Optics (AO), this novel proposal provides an estimator that is especially robust to modelling errors.

Data-driven Kalman filtering algorithms were brought to AO via exploitation of sparsity in Chapter 4. Both of the evaluated data-driven Kalman filtering algorithms involve fitting an Auto-Regressive (AR) model to measurement data; the matricial coefficients of the model are intuitively sparse, a property that when exploited massively drops complexity. These sparse identification methods compete with the DARE, boasting a time complexity of $\mathrm{O}(m)$, rather than $\mathrm{O}(m^3)$, and allowing use of "tall" datasets with far less time-steps of data than measurements. Their data-driven nature also relaxes the assumption of known statistics and compensates, to a certain extent, for modelling errors.

Finally, a two-stage approach to prediction was proposed to reduce the time complexity of the online prediction operation to $\mathrm{O}(m)$, down from the conventional $\mathrm{O}(m^2)$. Prediction was split into a sparse stage for prediction of local structures in the wavefront phase, and a low-dimensional stage to correct for the remaining low-frequency components. This algorithm exchanges some of the performance of the conventional Kalman filter for significantly improved prediction times. Adequate tuning should allow the user to minimally sacrifice Kalman filter performance while achieving significant improvement of the online execution times.

6-1 Recommendations

- The two-stage predictor is still missing a formulation for circular arrays, in which case the square coarse units will necessarily include points outside the aperture. As proposed in Chapter 4, hypothetical measurements at points outside the aperture can, for example, be considered to be zero, or alternatively, one could artificially set them such that they minimize a measure of error between the interpolated and original wavefronts within each coarse unit. These proposals remain to be formalized and evaluated.

- The interpolation from the coarse grid to the original fine one used in the implementation presented herein was simple linear interpolation. More realistic methods of interpolation, in that they better replicate the shape of a wavefront, should be explored, possibly reducing the performance difference between the two-stage predictor and a Kalman filter, or allowing even lower choices of width of the coarse grid.
- Alternative descriptions of the coarse system could be explored to extend the applicability of the two-stage predictor. Subsection 4-2-2 formulates the coarse system such that its units are square, forcing the width of the coarse array to divide that of the original fine array. It would be relevant to find descriptions that allow more general unit shapes to increase the freedom of choice of both the coarse and original array widths. As it stands, the original width can not, for example, be a prime.
- Unfortunately, this thesis lacks a comparison to other methods dedicated to the reduction of the computational burden of the Kalman filter. Examples are [4, 8]. Such an evaluation is particularly important for the two-stage predictor, whose trade-off between performance and time should be compared against those of state-of-the-art methods.
- While the complexity of the sparse data-driven identification algorithms is low, these require a QR-factorization to be done every iteration, which adds a lot of overhead. The sparsity patterns involved are particular in that the non-zero pattern of a given row (resp. column) is a shift of that of a previous row (resp. column), with or without a few added non-zeros. This makes re-utilization of previous QR-factorizations, explored in works such as [41], an interesting topic to explore to reduce the execution time in practise.
- Data-driven Kalman filtering, in combination with identification of the A matrix, should permit online adaptation to changing turbulence statistics via re-identification every few time-steps. Although not in the scope of the thesis, this is a research topic of relevance.

Several additional possible off-topic research proposals are presented in the Appendix. These range from modelling alternatives to potential improvements to MARK.

Appendix A

On the modelling of turbulence

A-1 Lagged AR-1 turbulence modelling

It was mentioned in Subsection 2-2-2 that fractional-pixel movement per time-step yields AR-1 turbulence models that are suboptimal for application of a Kalman filter. As the difference between the pixel-wind-speed and the nearest integer increases, the error becomes more noticeable (see Figures 5-22 and 5-23). Consider then increasing the lag of the model, making A represent several time-steps' worth of movement, such that this movement is closer to a whole-pixel shift. For example, for a wind speed of 0.5 pixels per time-step, lagging the model one time-step further results in an A matrix that represents a shift of 1 pixel, which is accurately modelled. Consider that the standard AR-1 model as in (2-5) has lag 1. Then, take the example of an AR-1 model with lag 2:

$$\phi_{k+1} = A\phi_{k-1} + w_{k-1} \tag{A-1}$$

which is described by the following state-space model:

$$\begin{aligned} \begin{bmatrix} \phi_k \\ \phi_{k+1} \end{bmatrix} &= \begin{bmatrix} 0 & I \\ A & 0 \end{bmatrix} \begin{bmatrix} \phi_{k-1} \\ \phi_k \end{bmatrix} + \begin{bmatrix} 0 \\ w_{k-1} \end{bmatrix} \\ y_k &= \begin{bmatrix} 0 & G \end{bmatrix} \begin{bmatrix} \phi_{k-1} \\ \phi_k \end{bmatrix} + v_k \end{aligned} \tag{A-2}$$

and whose Kalman filter is given by:

$$\begin{aligned} \begin{bmatrix} \hat{\phi}_{k|k} \\ \hat{\phi}_{k+1|k} \end{bmatrix} &= \begin{bmatrix} 0 & I \\ A & 0 \end{bmatrix} \begin{bmatrix} \hat{\phi}_{k-1|k-1} \\ \hat{\phi}_{k|k-1} \end{bmatrix} + Ky_k \\ y_k &= \begin{bmatrix} 0 & G \end{bmatrix} \begin{bmatrix} \hat{\phi}_{k-1|k-1} \\ \phi_{k|k-1} \end{bmatrix} + e_k \end{aligned} \tag{A-3}$$

Say the state prediction error covariance P of the Kalman filter is divided into four blocks:

$$P = \begin{bmatrix} P_1 & P_3 \\ P_3^{\mathrm{T}} & P_2 \end{bmatrix}$$

The process and measurement noises of this model are uncorrelated, and hence the Kalman predictor gain can be developed into:

$$\begin{aligned} K &= \begin{bmatrix} 0 & I \\ A & 0 \end{bmatrix} K_{\mathrm{r}} \\ &= \begin{bmatrix} 0 & I \\ A & 0 \end{bmatrix} P \begin{bmatrix} 0 \\ G^{\mathrm{T}} \end{bmatrix} \left(\begin{bmatrix} 0 & G \end{bmatrix} P \begin{bmatrix} 0 \\ G^{\mathrm{T}} \end{bmatrix} + R \right)^{-1} \\ &= \begin{bmatrix} P_2 G^{\mathrm{T}} (G P_2 G^{\mathrm{T}} + R)^{-1} \\ A P_3 G^{\mathrm{T}} (G P_2 G^{\mathrm{T}} + R)^{-1} \end{bmatrix} \end{aligned}$$

While the Discrete-time Algebraic Riccati Equation (DARE) is

$$\begin{aligned} \begin{bmatrix} P_1 & P_3 \\ P_3^{\mathrm{T}} & P_2 \end{bmatrix} = \begin{bmatrix} 0 & I \\ A & 0 \end{bmatrix} \begin{bmatrix} P_1 & P_3 \\ P_3^{\mathrm{T}} & P_2 \end{bmatrix} \begin{bmatrix} 0 & A^{\mathrm{T}} \\ I & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & Q \end{bmatrix} \\ - \begin{bmatrix} P_2 G^{\mathrm{T}} (G P_2 G^{\mathrm{T}} + R)^{-1} G P_2 & P_2 G^{\mathrm{T}} (G P_2 G^{\mathrm{T}} + R)^{-1} G P_3 A^{\mathrm{T}} \\ A P_3 G^{\mathrm{T}} (G P_2 G^{\mathrm{T}} + R)^{-1} G P_2 & A P_3 G^{\mathrm{T}} (G P_2 G^{\mathrm{T}} + R)^{-1} G P_3 A^{\mathrm{T}} \end{bmatrix} \end{aligned}$$

or, instead,

$$\begin{aligned} \begin{bmatrix} P_1 & P_3 \\ P_3^{\mathrm{T}} & P_2 \end{bmatrix} = \begin{bmatrix} P_2 & P_3^{\mathrm{T}} A^{\mathrm{T}} \\ A P_3 & A P_1 A^{\mathrm{T}} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & Q \end{bmatrix} \\ - \begin{bmatrix} P_2 G^{\mathrm{T}} (G P_2 G^{\mathrm{T}} + R)^{-1} G P_2 & P_2 G^{\mathrm{T}} (G P_2 G^{\mathrm{T}} + R)^{-1} G P_3 A^{\mathrm{T}} \\ A P_3 G^{\mathrm{T}} (G P_2 G^{\mathrm{T}} + R)^{-1} G P_2 & A P_3 G^{\mathrm{T}} (G P_2 G^{\mathrm{T}} + R)^{-1} G P_3 A^{\mathrm{T}} \end{bmatrix} \end{aligned} \tag{A-4}$$

The equations for the supra-diagonal blocks are solved for $P_3 = 0$, thus:

$$\begin{aligned} P &= \begin{bmatrix} P_1 & 0 \\ 0 & P_2 \end{bmatrix} \\ K &= \begin{bmatrix} P_2 G^{\mathrm{T}} (G P_2 G^{\mathrm{T}} + R)^{-1} \\ 0 \end{bmatrix} \end{aligned} \tag{A-5}$$

Note now that

$$\begin{aligned} P_1 &= \mathrm{E}[(\phi_k - \phi_{k|k})(\phi_k - \phi_{k|k})^{\mathrm{T}}] \\ P_2 &= \mathrm{E}[(\phi_{k+1} - \phi_{k+1|k})(\phi_{k+1} - \phi_{k+1|k})^{\mathrm{T}}] \end{aligned}$$

and that, from (A-4),

$$\begin{aligned} P_1 &= P_2 - P_2 G^{\mathrm{T}} (G P_2 G^{\mathrm{T}} + R)^{-1} G P_2 \\ P_2 &= A P_1 A^{\mathrm{T}} + Q \end{aligned}$$

A quick comparison with the standard Kalman filter equations (with uncorrelated process and measurement noises) shows that P is thus just the state reconstruction and prediction error covariances arranged block-diagonally, and that the top block-row of K is the reconstructor gain of a standard Kalman filter for a lag 1 system with A as its state-transition matrix and Q as its process noise covariance. Indeed,

$$P_2 = AP_2A^{\mathrm{T}} + Q - AP_2G^{\mathrm{T}}(GP_2G^{\mathrm{T}} + R)^{-1}GP_2A^{\mathrm{T}}$$

Now define

$$\begin{aligned} K_{\mathrm{r}}^{(\mathrm{s})} &= P_2G^{\mathrm{T}}(GP_2G^{\mathrm{T}} + R)^{-1} \\ K^{(\mathrm{s})} &= AK_{\mathrm{r}}^{(\mathrm{s})} = AP_2G^{\mathrm{T}}(GP_2G^{\mathrm{T}} + R)^{-1} \end{aligned}$$

where the superscript (s) stands for "standard". Then, substituting it into (A-3), skipping the reconstruction step, yields:

$$\hat{\phi}_{k+1|k} = A\hat{\phi}_{k-1|k-2} + AK_{\mathrm{r}}^{(\mathrm{s})}y_{k-1} \tag{A-6}$$

or, resuming use of the usual notation and substituting $AK_{\mathrm{r}}^{(\mathrm{s})}$ for $K^{(\mathrm{s})}$:

$$\boxed{\hat{\phi}_{k+1} = A\hat{\phi}_{k-1} + K^{(\mathrm{s})}y_{k-1}} \tag{A-7}$$

where $K^{(\mathrm{s})}$ is simply the predictor gain designed for matrices (A, G, Q, R). That is, using the MATLAB function `idare` (and ignoring the output for P):

$$K^{(\mathrm{s})} = \left[\,\texttt{idare}(A^{\mathrm{T}}, G^{\mathrm{T}}, Q, R)\,\right]^{\mathrm{T}}$$

Remark A.1. Notice that in (A-3) or (A-6), the wavefront prediction does not actually use information up to time-step k. With (A-6) established, it should be noted that, to be more rigorous, $\hat{\phi}_{k+1|k}$ should actually be referred to as $\hat{\phi}_{k+1|k-1}$ instead.

As for downsides of the lagged model, notice that model (A-1) and its Kalman filter (A-3) effectively split the data between even and odd time-steps. That is, according to the model, the phase and slopes at even time-steps depend only on those from previous even time-steps, and the same applies to odd time-steps. This means that, in truth, two Kalman filters are in alternate operation, and their transient states are handled separately, doubling the time it takes to reach steady-state. This is reflected in an extended observability matrix of (A-2), which only has full rank if it includes the third power of the state-transition matrix. Using the $\mathcal{O}_p$ notation from Chapters 3 and 4:

$$\mathcal{O}_4 = \left[\begin{array}{cc} 0 & G \\ GA & 0 \\ \hdashline 0 & GA \\ GA^2 & 0 \end{array}\right]$$

meaning that it takes four time-steps of future data, as opposed to the usual two (see that $\mathcal{O}_2$ has been used throughout the thesis, rather than $\mathcal{O}_4$), to reconstruct a certain state. Moreover, lagging the model implies working in a higher-speed regime, which, as seen in

Figures 5-22 to 5-24, tends to increase the error; higher-speed regimes also require more lax distance-constraints, increasing execution times. It might thus not be advisable to lag the model when the wind speed substantially exceeds 0.5 lenslets per time-step.

Working with a lagged AR-1 model further incurs trivial changes in the data-driven Kalman filtering algorithms if the Kalman filter is written in form (A-7), rather than (A-3); the latter model could be used as-is, but the user would end up fitting for a lot of known zeros, or otherwise complicating the problem further to apply redundant sparsity patterns. Instead, organize the data into two matrices, one for even time-steps, and one for odd ones:

$$\mathcal{Y}_{i,j,N} \equiv \begin{bmatrix} y_i & y_{i+2} & y_{i+4} & \cdots & y_{i+2(N-1)} \\ y_{i+2} & y_{i+4} & & \cdots & y_{i+2N} \\ y_{i+4} & & & & \\ \vdots & & & & \vdots \\ y_{i+2(j-1)} & y_{i+2j} & & \cdots & y_{i+2(j+N-2)} \end{bmatrix}$$

$$\mathcal{Y}_{i,N} \equiv \begin{bmatrix} y_i & y_{i+2} & y_{i+4} & \cdots & y_{i+2N} & y_{i+2(N-1)} \end{bmatrix}$$

where if i is even then $\mathcal{Y}_{i,j,N}$ comprises data from even time-steps, and if i is odd, it does so for the odd time-steps. If measurements are taken from time-step 0 onward, the first least-squares problem from Juang and Chen's algorithm can be written as

$$\widehat{G\mathcal{L}} = \underset{G\mathcal{L}}{\operatorname{argmin}} \left\| \begin{bmatrix} \mathcal{Y}_{2s,N} & \mathcal{Y}_{2s+1,N} \end{bmatrix} - (G\mathcal{L}) \begin{bmatrix} \mathcal{Y}_{0,s,N} & \mathcal{Y}_{1,s,N} \end{bmatrix} \right\|_{\mathrm{F}}^2$$

and that from MARK is analogous. Once an estimate of $G\mathcal{L}$ (Juang) or $\mathcal{L}$ (MARK) are obtained, $\widehat{K}$ is retrieved as usual.

Wind Speed	Standard model (1-step lag)			2-step Lagged model		
	DARE	Ident. Full	Two-stage	DARE	Ident. Full	Two-stage
0.5	0.0030	0.0031	0.0031	0.00079	0.0012	0.0015
1.5	0.0032	0.0036	0.0035	0.0046	0.0052	0.0055

Table A-1: NMSE comparison between the standard (lag 1) AR-1 model and a 2-step lagged counterpart. The array is 36×36 and the wind speed is given in lenslets per time-step. The values within the cells are the NMSE.

For a wind-speed of 0.5 lenslets per time-step, one sees a substantial reduction in error when the lagged model is applied, near a third of the NMSE of the non-lagged model. In fact, the results now match those for a wind speed of 1, seen in Figure 5-22, as the model is effectively the same. For 1.5 lenslets per time-step, the increase in error due to the larger wind speed offset any gain from lagging the model, resulting in worse performance.

Wind speeds below 0.5 should be lagged over two steps. For example, a wind-speed of 0.25 should be approached with a model of lag 4. Nonetheless note that it may not be necessary to equal the wind speed to an integer, but rather merely bring it closer to the integers instead (in fact, modelling quality depends on the pixel-, not lenslet-movement, so although these are approximately the same, this is the case in Table A-1 already).

A-2 Iterative data-driven model refinement

The "noisy model" case of Chapter 5 relied on first obtaining a poor estimate of A and then using a single iteration of data-driven filtering to compensate for the modelling error and obtain a better $\widehat{A}$. One can wonder how far this can be taken, i.e. if $\widehat{A}$ can be seen to converge to a final best estimate as additional iterations of data-driven filtering are executed.

The process is as follows: begin from initial noisy wavefront phase data, obtained, for example, from pseudo-inversion of G and multiplication with noisy measurements; then, obtain an estimate A from the noisy phase data using the procedures in Chapter 2; with this poor first A and the measurements, identify a Kalman gain and apply it to the measurements to obtain *better* phase data; use this filtered phase data to estimate a new A, and repeat:

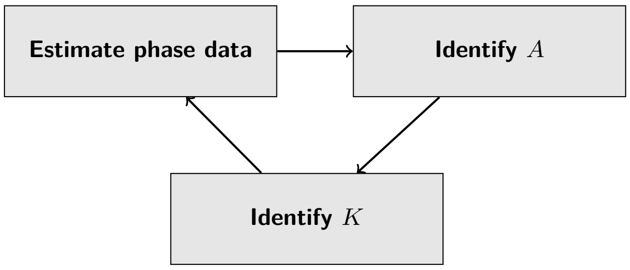

Figure A-1: Iterative model refinement cycle: identify A from phase data, identify K with A and the measurements, improve estimates of the phase with K, repeat.

The use of data-driven Kalman filtering should compensate somewhat for the erroneous noisy modelling, and the goal is for it to continuously yield better estimates of A until convergence is observed over the training data. Unfortunately, as it stands, convergence is observed only for low speeds, and divergence is seen otherwise. Being outside the scope of this thesis, this is a topic for further research. Preliminary experiments in an especially noisy scenario with a wind speed of 0.25 lenslets per time-step yielded:

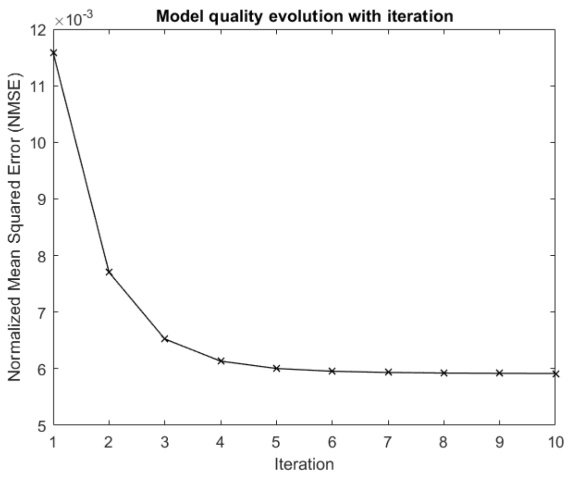

Figure A-2: Evolution of the NMSE of the identified Kalman filter for progressively improved A matrices through iterative refinement.

whereas divergence was observed for speeds of 0.5 and over, after a few iterations. The initial and final A matrices are given below, along with a side-by-side comparison between the final and true matrices:

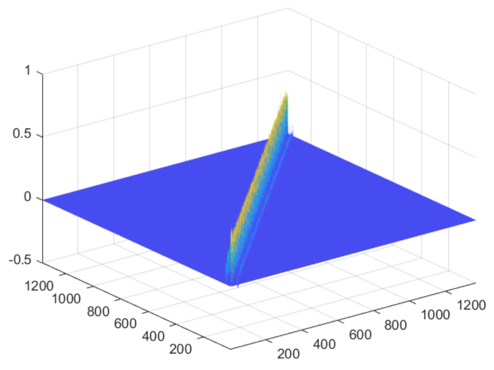

Figure A-3: Initial $\widehat{A}$ before iterative refinement.

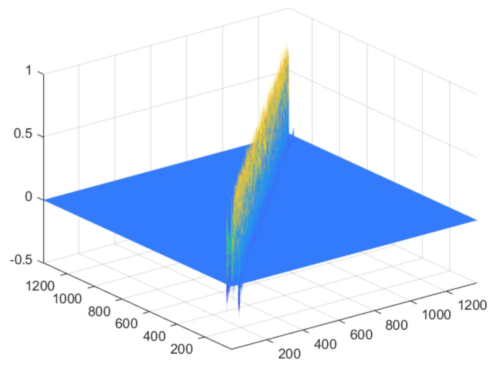

Figure A-4: Final $\widehat{A}$ after iterative refinement.

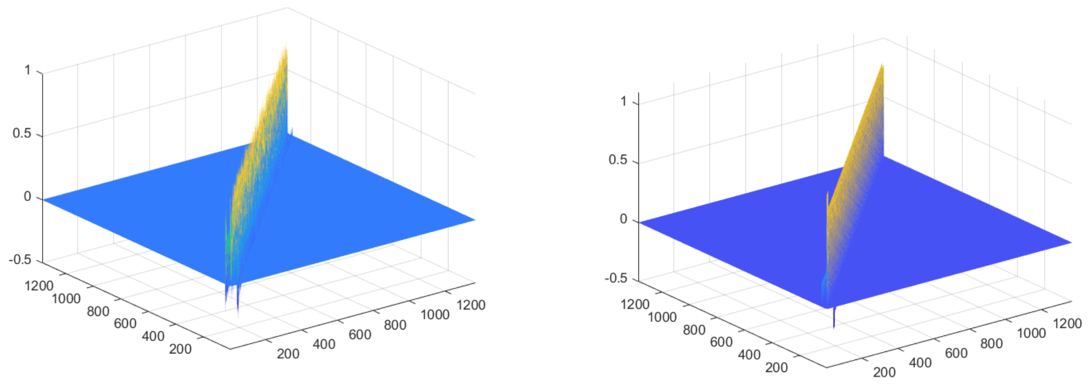

Figure A-5: Side-by-side comparison between the final $\widehat{A}$ after refinement and the true A.

A-3 Data-driven prediction for mismatched models

One of the points repeated throughout the thesis was that data-driven filtering covers for modelling errors. This is explored further here: the turbulence is two-layered, but the model accounts only for the dominant bottom layer. The DARE gain is designed also with the statistics of the bottom layer, meaning that A and Q are both ignorant of the second, less influential, layer. Measurement data, however, is true noisy slope data, and accounts for both layers. Simulation was done in OOMAO, where "dominance" was set via the "fractional r_0" parameter: the bottom layer had it set to 0.7, whereas the top layer had it set to 0.3.

This would, for example, be the case if A is preset under wrong assumptions. Because

$(A - \overline{K}G)'$ depends on $\overline{K}$, it can not be preset and will be obtained normally from the phase data obtained pseudo-inversion of G and multiplication with the slopes. Figures A-6 and A-7 show the results.

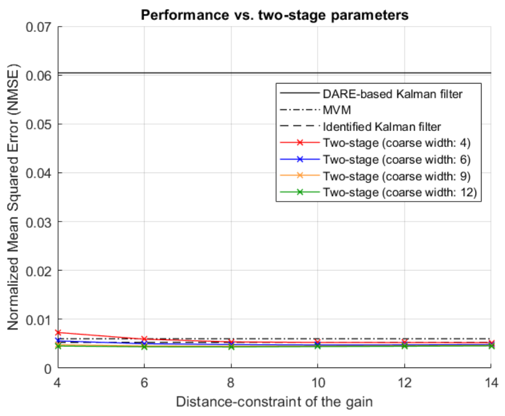

Figure A-6: Predictor performances: two-layered turbulence modelled with a single-layer model.

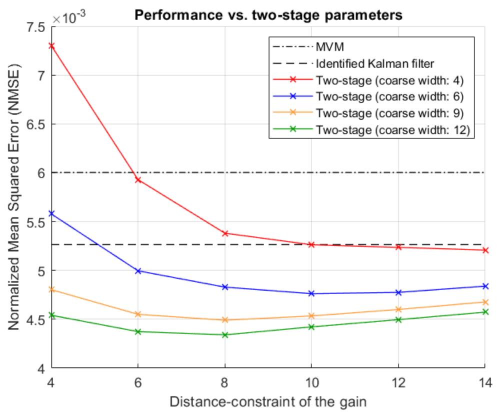

Figure A-7: Predictor performances: two-layered turbulence modelled with a single-layer model (DARE ignored).

The DARE-based gain suffered immensely from mismodelling, whereas the identified filters successfully compensated the error.

Appendix B

On MARK

B-1 Pre-filtered data

Correlation methods [20, 27, 28, 29] have been traditionally formulated under the assumption that the measurement data is pre-filtered by default. The performance of these algorithms is particularly sensitive to the choice of initial filter, and the lack of any pre-filtering will often make these algorithms yield estimators with relatively high covariance [30].

MARK has the benefit of performing better than correlation methods not only if both are directly using output data, but also when they share the initial filter. Because correlation methods are usually evaluated under the assumption that the output data is pre-filtered [42], a fair comparison would supply filtered data to MARK as well. To this end, define a suboptimal initial filter with gain J, state prediction $\hat{x}^*_k$ and prediction error e^*_k:

$$\begin{aligned} \hat{x}^*_{k+1} &= A\hat{x}^*_k + Je^*_k \\ y_k &= C\hat{x}^*_k + e^*_k \end{aligned}$$

Now define a filter-error system, whose state is the error ξ_k between the *optimal* state prediction $\hat{x}_k$ and its suboptimal counterpart $\hat{x}^*_k$:

$$\begin{aligned} \xi_k &= \hat{x}_k - \hat{x}^*_k \\ \xi_{k+1} &= (A - JC)\xi_k + (K - J)e_k \\ e^*_k &= C\xi_k + e_k \end{aligned} \tag{B-1}$$

where e_k are the (optimal) innovations. One can see this as a more general formulation that replaces the innovation-form of the optimal filter, whose state is $\xi_k = \hat{x}_k - 0$, i.e. the suboptimal filter is nonexistent. These suboptimal prediction errors e^*_k are the pre-filtered (whitened) output data to be used by MARK.

Denote the block-Hankel matrices formed with the suboptimal prediction errors by $\mathcal{E}_{i,j,N}$, following the notation of Chapter 3, and form the extended observability matrix of the suboptimal filter:

$$\mathcal{O}_p = \begin{bmatrix} C \\ C(A-JC) \\ \vdots \\ C(A-JC)^{p-1} \end{bmatrix}$$

Then $\mathcal{L}$, now defined as

$$\mathcal{L} = \begin{bmatrix} (A-KC)^{s-1}(K-J) & \cdots & (K-J) \end{bmatrix}$$

can be estimated with

$$\hat{\mathcal{L}} = \underset{\mathcal{L}}{\operatorname{argmin}} \, \|\mathcal{E}_{s,p,N} - \mathcal{O}_p \mathcal{L} \mathcal{E}_{0,s,N}\|_{\mathrm{F}}^2 \tag{B-2}$$

which is of almost the exact same form as the original problem in (3-23), requiring only the additional step of prefiltering the data, replacing $Y_{0,s,N}$ and $Y_{s,p,N}$ with $\mathcal{E}_{0,s,N}$ and $\mathcal{E}_{s,p,N}$. The gain K is now the last block-column of $\mathcal{L}$ plus the initial gain J.

B-2 Weighted formulation

It is known that the minimum-variance unbiased linear estimator of x, such that

$$y = Fx + \varepsilon$$

where $\mathrm{E}[\varepsilon] = 0$ and $\mathrm{E}[\varepsilon\varepsilon^{\mathrm{T}}] = L$, is obtained via weighted least-squares [35], rather than with the standard least-squares solution used in Chapter 3. Vectorization prior to weighting is undesirable as it turns the matrix sizes unwieldy. Instead, the original form of the problem will be maintained as much as possible. In matrix-form, the equation

$$\underbrace{\begin{bmatrix} | & & | \\ y_1 & \cdots & y_N \\ | & & | \end{bmatrix}}_{Y} = F \underbrace{\begin{bmatrix} | & & | \\ x_1 & \cdots & x_N \\ | & & | \end{bmatrix}}_{X} + \mathcal{E}$$

is equivalent to a set of independent equations, one for each column of X. The minimum variance unbiased estimator for each i^{th} column of X is the solution to a weighted least-squares problem of the form:

$$y_i = Fx_i + \varepsilon_i$$

and if $\mathrm{E}[\varepsilon_i\varepsilon_i^{\mathrm{T}}] = L$, then,

$$\hat{x}_i = \underbrace{(G^{\mathrm{T}}L^{-1}G)^{-1}G^{\mathrm{T}}L^{-1}}_{M} y_i$$

If $\mathrm{E}[\varepsilon_i\varepsilon_i^{\mathrm{T}}] = L$ is applicable to all i, then

$$\widehat{X} = \begin{bmatrix} | & & | \\ My_1 & \cdots & My_N \\ | & & | \end{bmatrix} = MY$$

If, instead, $\mathrm{E}[\varepsilon_i\varepsilon_i^{\mathrm{T}}] = L_i$, with L_i different across different i, then each x_i should be solved for independently and weighted accordingly.

Application to MARK

Although, throughout the thesis, MARK (and Juang) had the row-wise problem (3-24) solved first so that sparsity can be exploited, in general, it is indifferent whether the row- or the column-wise problem is solved first. Solving for the columns first makes the formulation of the weighted problem simpler; note however, that this weighted approach is applicable just the same to the column-wise problem if one solves the row-wise one for $\widehat{\mathcal{O}_p\mathcal{L}}$ first (using non-weighted least-squares). If MARK is solved column-first, then, recalling that $\widehat{X}_{s,N} = \mathcal{L}Y_{0,s,N}$, the first problem is to find an estimate of $\widehat{X}_{s,N}$ such that

$$Y_{s,p,N} = \mathcal{O}_p\hat{X}_{s,N} + \mathcal{S}_pE_{s,p,N}$$

Notice that each column of $\mathcal{S}_pE_{s,p,N}$ has the same covariance matrix S. Denoting the output prediction-error (or innovation) covariance by R_e, S is given by:

$$S = \mathcal{S}_p(I_p \otimes R_e)\mathcal{S}_p^{\mathrm{T}}$$

Following the reasoning above, set the weight W, found by *estimating* S with a previous estimate of K. The prediction-error sequence can be drawn from the error of the least-squares problem, and with it, R_e can be easily estimated. With S estimated as $\widehat{S}$ using $\widehat{K}$ and $\widehat{R}_e$,

$$W = \widehat{S}^{-1}$$

The optimal state prediction sequence can now be solved for in a weighted least-squares setting:

$$\widehat{X}_{s,N} = \underset{X_{s,N}}{\operatorname{argmin}} \left\| W^{\frac{1}{2}}Y_{s,p,N} - W^{\frac{1}{2}}\mathcal{O}_pX_{s,N} \right\|_{\mathrm{F}}^2$$

whose solution is merely

$$\boxed{\widehat{X}_{s,N} = (\mathcal{O}_pW\mathcal{O}_p)^{-1}\mathcal{O}_p^{\mathrm{T}}WY_{s,p,N}}$$

The matrix $\mathcal{L}$ is then obtained as usual

$$\mathcal{L} = \widehat{X}_{s,N}Y_{0,s,N}^{\dagger}$$

from which $\widehat{K}$ is retrieved, and the weighted problem can be solved again with a new weight obtained from the new $\widehat{K}$, and the problem solved iteratively. The idea is that if previously estimated gains are somewhat representative of the optimal gain, then they can be used to decide a weighting factor.

Remark B.1. Online application of the weighted procedure does not necessarily require iterations over a single data batch. The inherently repetitive nature of an online implementation allows for the weight to be recomputed as the gain is updated for the next batch.

Remark B.2. Proper conditioning of W must be ensured. Because it depends on estimated covariances $\hat{R}_e$, robust covariance estimation techniques could be explored. For large-scale problems, a popular reference for a well-conditioned covariance estimator is [43].

Consider a 1-output, 2-state system, so that the Kalman gain has two entries which can be plotted in the x- and y-axes. 200 simulations were run in both the unweighted and weighted case, and both cases had the tuning paramters set equally, and were subjected to the same process and measurement noise sequences. The weighted MARK was iterated 20 times each simulation for a final estimate of K. The results are shown in Figures B-1 and B-2.

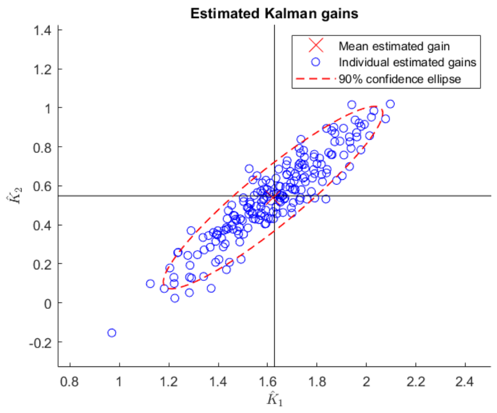

Figure B-1: Results of the unweighted MARK algorithm over 200 simulations. The large cross marks the optimal gain. Mean-squared-error: 0.09027

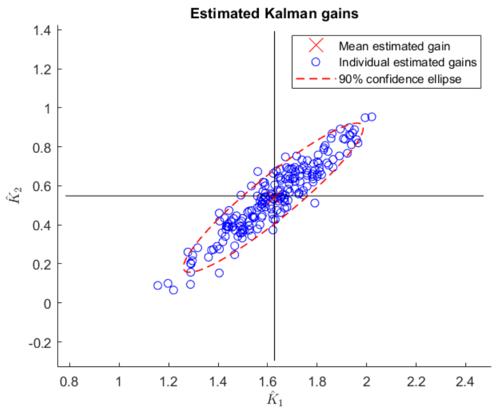

Figure B-2: Results of the weighted MARK algorithm, using 20 iterations, over 200 simulations. The large cross marks the optimal gain. Mean-squared-error: 0.06044

Abusing language, increasing p brings the unweighted problem further from the optimal weight. Over the optimal value of p, one sees deterioration of the results if the algorithm is unweighted. In contrast, the weighted version allows large increases in p without deterioration of the results, and in fact, these increases result in improved mean-squared-errors over the unweighted algorithm with its best p.

This does mean that the unweighted and weighted algorithms should not be tuned equally, although such was done in the figures for direct comparison. The weighted version should exploit large values of p as much as possible

Bibliography

[1] M. Verhaegen, P. Pozzi, O. Soloviev, G. Vdovin, and D. Wilding, "Lecture notes on control for high resolution imaging." Delft University of Technology, Apr. 2017.

[2] P. Piscaer, "Sparse VARX model identification for large-scale adaptive optics," Master's thesis, Delft University of Technology, 2016.

[3] C. Kulcsár, H.-F. Raynaud, C. Petit, J.-M. Conan, and P. V. de Lesegno, "Optimal control, observers and integrators in adaptive optics," *Opt. Express*, vol. 14, pp. 7464–7476, Aug. 2006.

[4] P. Massioni, H. Raynaud, C. Kulcsár, and J. Conan, "An approximation of the Riccati equation in large-scale systems with application to adaptive optics," *IEEE Transactions on Control Systems Technology*, vol. 23, pp. 479–487, Mar. 2015.

[5] R. Gilmozzi and J. Spyromilio, "The European Extremely Large Telescope (E-ELT)," *The Messenger*, vol. 127, p. 11, Mar. 2007.

[6] M. Johns, P. McCarthy, K. Raybould, A. Bouchez, A. Farahani, J. Filgueira, G. Jacoby, S. Shectman, and M. Sheehan, "Giant Magellan telescope: overview," in *Ground-based and Airborne Telescopes IV* (L. M. Stepp, R. Gilmozzi, and H. J. Hall, eds.), vol. 8444, pp. 526 – 541, International Society for Optics and Photonics, SPIE, 2012.

[7] L. Stepp, "Thirty Meter Telescope project update," in *Ground-based and Airborne Telescopes IV* (L. M. Stepp, R. Gilmozzi, and H. J. Hall, eds.), vol. 8444, pp. 510 – 525, International Society for Optics and Photonics, SPIE, 2012.

[8] M. Gray, C. Petit, S. Rodionov, M. Bocquet, L. Bertino, M. Ferrari, and T. Fusco, "Local ensemble transform Kalman filter, a fast non-stationary control law for adaptive optics on ELTs: theoretical aspects and first simulation results," *Opt. Express*, vol. 22, pp. 20894–20913, Aug. 2014.

[9] A. H. Jazwinski, "Limited memory optimal filtering," *Automatic Control, IEEE Transactions on*, vol. 13, pp. 558 – 563, Nov. 1968.

[10] M. Rosensteiner, “Cumulative reconstructor: fast wavefront reconstruction algorithm for Extremely Large Telescopes,” *Journal of the Optical Society of America. A, Optics, image science, and vision*, vol. 28, pp. 2132–8, Oct. 2011.

[11] C. De Visser, E. Brunner, and M. Verhaegen, “On distributed wavefront reconstruction for large-scale adaptive optics systems,” *Journal of the Optical Society of America A*, vol. 33, p. 817, May. 2016.

[12] A. N. Kolmogorov, V. Levin, J. C. R. Hunt, O. M. Phillips, and D. Williams, “The local structure of turbulence in incompressible viscous fluid for very large Reynolds numbers,” *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences*, vol. 434, no. 1890, pp. 9–13, 1991.

[13] A. N. Kolmogorov, V. Levin, J. C. R. Hunt, O. M. Phillips, and D. Williams, “Dissipation of energy in the locally isotropic turbulence,” *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences*, vol. 434, no. 1890, pp. 15–17, 1991.

[14] J. Schmidt, *Numerical simulation of optical wave propagation: With examples in MATLAB.* SPIE, Jan. 2010.

[15] R. Conan, “Mean-square residual error of a wavefront after propagation through atmospheric turbulence and after correction with Zernike polynomials,” *J. Opt. Soc. Am. A*, vol. 25, pp. 526–536, Feb. 2008.

[16] G. I. Taylor, “The spectrum of turbulence,” *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, vol. 164, no. 919, pp. 476–490, 1938.

[17] C. Correia, J.-M. Conan, C. Kulcsar, H.-F. Raynaud, and C. Petit, “Adapting optimal LQG methods to ELT-sized AO systems,” *Proceedings of Adaptive Optics for Extremely Large Telescopes (AO4ELT)*, Jan. 2010.

[18] D. P. Looze, “Minimum variance control structure for adaptive optics systems,” *J. Opt. Soc. Am. A*, vol. 23, pp. 603–612, Mar. 2006.

[19] D. Q. Lee, “Numerically efficient methods for solving least squares problems,” 2012.

[20] R. Mehra, “On the identification of variances and adaptive Kalman filtering,” *IEEE Transactions on Automatic Control*, vol. 15, pp. 175–184, Apr. 1970.

[21] J. Duník, O. Straka, O. Kost, and J. Havlík, “Noise covariance matrices in state-space models: A survey and comparison of estimation methods - Part I,” *International Journal of Adaptive Control and Signal Processing*, vol. 31, no. 11, pp. 1505–1543, 2017.

[22] R. Mehra, “Approaches to adaptive filtering,” *Automatic Control, IEEE Transactions on*, vol. AC-17, pp. 693 – 698, Nov. 1972.

[23] D. Magill, “Optimal adaptive estimation of sampled stochastic processes,” *IEEE Transactions on Automatic Control*, vol. 10, pp. 434–439, Oct. 1965.

[24] R. Kashyap, “Maximum likelihood identification of stochastic linear systems,” *Automatic Control, IEEE Transactions on*, vol. 15, pp. 25 – 34, Mar. 1970.

[25] R. H. Shumway and D. S. Stoffer, "An approach to time series smoothing and forecasting using the EM algorithm," *Journal of Time Series Analysis*, vol. 3, no. 4, pp. 253–264, 1982.

[26] N. Gupta and R. Mehra, "Computational aspects of maximum likelihood estimation and reduction in sensitivity function calculations," *IEEE Transactions on Automatic Control*, vol. 19, pp. 774–783, Dec. 1974.

[27] P. R. Bélanger, "Estimation of noise covariance matrices for a linear time-varying stochastic process," *Automatica*, vol. 10, no. 3, pp. 267 – 275, 1974.

[28] B. Carew and P. Bélanger, "Identification of optimum filter steady-state gain for systems with unknown noise covariances," *IEEE Transactions on Automatic Control*, vol. 18, pp. 582–587, Dec. 1973.

[29] B. J. Odelson, M. R. Rajamani, and J. B. Rawlings, "A new autocovariance least-squares method for estimating noise covariances," *Automatica*, vol. 42, no. 2, pp. 303 – 308, 2006.

[30] P. Cerqueira, "Adaptive Kalman filtering." Literature survey, Nov. 2019.

[31] K. Myers and B. Tapley, "Adaptive sequential estimation with unknown noise statistics," *IEEE Transactions on Automatic Control*, vol. 21, pp. 520–523, Aug. 1976.

[32] J. Wang, "Stochastic modeling for real-time kinematic GPS/GLONASS positioning," *Navigation*, vol. 46, no. 4, pp. 297–305, 1999.

[33] W. C. Louv, "Adaptive filtering," *Technometrics*, vol. 26, no. 4, pp. 399–409, 1984.

[34] J. Juang, C. Chen, and M. Phan, "Estimation of Kalman filter gain from output residuals," *Journal of Guidance Control and Dynamics*, vol. 16, pp. 903–908, Nov. 1993.

[35] M. Verhaegen and V. Verdult, *Filtering and System Identification: A Least Squares Approach.* Cambridge University Press, 2007.

[36] T. A. Davis, "Algorithm 915, SuiteSparseQR: Multifrontal Multithreaded Rank-Revealing Sparse QR Factorization," *ACM Trans. Math. Softw.*, vol. 38, Dec. 2011.

[37] P. Massioni, C. Kulcsár, H.-F. Raynaud, and J.-M. Conan, "Fast computation of an optimal controller for large-scale adaptive optics," *J. Opt. Soc. Am. A*, vol. 28, pp. 2298–2309, Nov. 2011.

[38] B. Bamieh, F. Paganini, and M. A. Dahleh, "Distributed control of spatially invariant systems," *IEEE Transactions on Automatic Control*, vol. 47, pp. 1091–1107, Jul. 2002.

[39] R. Conan and C. Correia, "Object-oriented Matlab adaptive optics toolbox," in *Adaptive Optics Systems IV* (E. Marchetti, L. M. Close, and J.-P. Véran, eds.), vol. 9148, pp. 2066 – 2082, International Society for Optics and Photonics, SPIE, 2014.

[40] C. R. Vogel and Q. Yang, "Fast optimal wavefront reconstruction for multi-conjugate adaptive optics using the Fourier domain preconditioned conjugate gradient algorithm," *Opt. Express*, vol. 14, pp. 7487–7498, Aug 2006.

[41] S. Hammarling and C. Lucas, “Updating the QR factorization and the least squares problem,” Jan. 2008.

[42] J. Duník, O. Straka, O. Kost, and J. Havlík, “Noise covariance matrices in state-space models: A survey and comparison of estimation methods—part i,” *International Journal of Adaptive Control and Signal Processing*, vol. 31, no. 11, pp. 1505–1543, 2017.

[43] O. Ledoit and M. Wolf, “A well-conditioned estimator for large-dimensional covariance matrices,” *Journal of Multivariate Analysis*, vol. 88, no. 2, pp. 365–411, 2004.

Glossary

List of Acronyms

AO	Adaptive Optics
AR	Auto-Regressive
DARE	Discrete-time Algebraic Riccati Equation
MVM	Matrix-Vector-Multiplication
SVD	Singular Value Decomposition
SNR	Signal-to-Noise Ratio
NMSE	Normalized Mean Squared Error