JOHANNES KEPLER
UNIVERSITÄT LINZ | JKU

TNF

Technisch-Naturwissenschaftliche
Fakultät

Wavelet Methods in Adaptive Optics

DISSERTATION

zur Erlangung des akademischen Grades

Doktor der Technischen Wissenschaften

im Doktoratsstudium der

Technischen Wissenschaften

Eingereicht von:
Dipl.-Ing. Mykhaylo Yudytskiy, Bakk.techn.

Angefertigt am:
Institut für Industriemathematik

Beurteilung:
Prof. Dr. Ronny Ramlau (Betreuung)
Dr. Brent Ellerbroek

Linz, April 2014

Abstract

The next generation ground-based telescopes rely heavily on adaptive optics (AO) for overcoming the limitations of atmospheric turbulence. AO is a hardware-based technology in which the wavefront distortions arising from the atmospheric turbulence are corrected in real-time. Atmospheric tomography, or the reconstruction of the refractive index fluctuations in the atmosphere, is a major mathematical and computational challenge in AO. In this severely ill-posed problem, a stable and fast reconstruction algorithm that can take into account many real-life phenomena of telescope imaging is needed. For the future extremely large telescopes, such as the European Extremely Large Telescope (E-ELT), the algorithm must additionally process a large amount of data in the millisecond time frame, a substantial hurdle for standard reconstruction techniques even with heavy parallelization.

The goal of this work was to develop a new algorithm for AO with an emphasis on atmospheric tomography that is as effective as current techniques while, at the same time, greatly reducing the computing load on the computer that performs the calculations. In this thesis we present the novel algorithm, which we call the Finite Element-Wavelet Hybrid Algorithm (FEWHA) for atmospheric tomography, that fulfills this goal.

The FEWHA is a conjugate gradient based approach in which the Bayesian MAP estimate of the turbulence layers of the atmosphere is discretized using a finite element and a wavelet basis simultaneously. This dual-domain strategy induces a very efficient "matrix-free" representation of the underlying operators. The method utilizes the locality properties of compactly supported orthonormal wavelets, both in spatial and frequency domains.

The convergence of this iterative scheme is accelerated by designing an efficient preconditioner and by utilizing multi-scale techniques. Altogether, the computational complexity of the FEWHA scales linearly with the dimension of the problem; the method converges in a few iterations, is highly parallelizable and has a small memory footprint. Moreover, our algorithm is versatile as it can be applied to different AO systems.

We verified the performance of our method in terms of quality and speed for four AO systems in the E-ELT configuration using numerical simulations. In terms of quality, the method is comparable to the state of the art algorithms, the MVM and the FrIM, and in some configurations the FEWHA outperforms these benchmark approaches. The speed capabilities of the method were demonstrated on off-the-shelf computing systems. On these hardware platforms the FEWHA was from 5 to 50 times faster than the MVM, depending on the AO system. The highlight case for our method was the multi conjugate adaptive optics (MCAO) system where the algorithm delivered superior qualitative performance to the MVM and close to real-time computational performance on the non-dedicated hardware.

Zusammenfassung

Die erdgebundenen Teleskope der nächsten Generation beruhen besonders auf der adaptiven Optik (AO), welche die Überwindung der Einschränkungen durch die atmosphärischen Turbulenzen ermöglicht. AO ist eine hardware-basierte Technologie in der die Wellenfrontstörungen, die aus den atmosphärischen Turbulenzen entstehen, in Echtzeit korrigiert werden. Atmosphärische Tomographie, oder die Rekonstruktion der Brechungsindexschwankungen in der Atmosphäre, stellt eine große mathematische und rechnerische Herausforderung in AO dar. Für dieses extrem schlecht gestellte Problem wird ein schneller und stabiler Rekonstruktionsalgorithmus, der mehrere reale Phänomene der Bildgebung des Teleskops berücksichtigt, benötigt. Für die extrem großen Teleskope der Zukunft, wie das European Extremely Large Telescope (E-ELT), muss der Algorithmus noch zusätzlich eine große Menge von Daten in einem Millisekundenbereich verarbeiten können. Dies stellt ein wesentliches Hindernis für die Standardmethoden, auch unter massiver Parallelisierung, dar.

Das Ziel dieser Arbeit war die Entwicklung einer neuen Methode für AO mit dem Fokus auf die atmosphärische Tomographie, welche gleich effektiv wie die herkömmlichen Verfahren ist, jedoch weit geringere Computerleistungen benötigt. In dieser Dissertation präsentieren wir die neue Methode, die wir als der Finite Element-Wavelet Hybrid Algorithm (FEWHA) für die atmosphärische Tomographie bezeichnen, die dieses Ziel erfüllt.

Der FEWHA ist eine Methode, die auf dem Verfahren der konjugierten Gradienten basiert ist, in dem der Bayessches MAP Schätzer der Turbulenzschichten der Atmosphäre gleichzeitig in der Finite Elemente Basis und der Wavelet Basis diskretisiert ist. Diese "dual-domain" Strategie führt zu einer sehr effizienten matrizenfreien Darstellung der zugrundeliegenden Operatoren. Das Verfahren nutzt die Lokalisierungseigenschaften im Raum- und Frequenzbereich der orthonormalen Wavelets mit einem kompakten Träger.

Das Konvergenzverhalten dieses iterativen Schema ist beschleunigt durch den Entwurf eines effizienten Vorkonditionierers und durch die Anwendung der Multiskalen-Methoden. Insgesamt verhält sich die Komplexität von FEWHA linear mit der Dimension des Problems, das Verfahren konvergiert in wenigen Iterationen und hat eine effiziente Parallelisierung sowie wenig Speicherbedarf. Außerdem ist unser Algorithmus vielseitig, da er auf verschiedene AO Systeme anwendbar ist.

Numerische Tests bzgl. der Qualität und Geschwindigkeit wurden für vier AO Systeme in der E-ELT Konfiguration durchgeführt. Das Verfahren ist vergleichbar mit den Algorithmen auf dem neuesten Stand der Technik: MVM und FrIM, bzgl. der Qualität, und in manchen Konfigurationen dieser Benchmarkmethoden überlegen. Die Rechengeschwindigkeit der Methode wurde auf serienmäßig produzierten Rechner demonstriert. Auf diesen Maschinen ist der FEWHA 5 bis 50 mal schneller als die MVM für

die verschiedenen AO Systeme. Der herausragende Beispiel unserer Methode war das multi conjugate adaptive optics (MCAO) System, wo der Algorithmus eine überragende Leistung bzgl. der Qualität in nahezu Echtzeit auf den serienmäßig produzierten Rechner geliefert hat.

Acknowledgements

First and foremost, I would like to express my deep gratitude to Prof. Ronny Ramlau, director of the Johann Radon Institute for Computational and Applied Mathematics (RICAM) and my supervisor. I am greatly indebted to him for giving me this unique opportunity to be involved with the project "Mathematical Algorithms for E-ELT Adaptive Optics." I generously appreciate his guidance and also the freedom and trust he offered me to conduct my research.

It was an honor to have Dr. Brent Ellerbroek, Department Head of Instrumentation for the Thirty Meter Telescope, as the second referee of this dissertation. I am grateful for the time he invested in reading the thesis and for his guiding questions over the duration of the project.

I would like to express my most sincere thanks to Dr. Tapio Helin, Dr. Valeriya Naumova, Dr. Andreas Obereder, DI Daniela Saxenhuber and Dr. Kirk Soodhalter. Over the past several years, these people, who I have come to know as friends, have not only helped me expand my knowledge in mathematics, but also grow on the personal level.

It has been a great pleasure working with Dr. Matthias Rosensteiner, MSc Iuliia Shatokhina and everyone else from the present and former Austrian Adaptive Optics Team.

I wish to express my thanks to Dr. Clémentine Béchet, Dr. Enrico Fedrigo, Dr. Miska Le Louarn and Prof. Curtis R. Vogel for the fruitful discussions and their help regarding the captivating field of adaptive optics.

My deep gratitude goes to my friends for their moral support in the duration of the project.

Finally, I would like to express my kindest thanks to my parents, Peter and Lena, and to my sister Jenia, without the care and love of whom I would not be where I am right now.

This work has been carried out at RICAM, Linz, 2010-2014 in the framework of the project "Mathematical Algorithms for E-ELT Adaptive Optics." The project was funded by the Austrian Federal Ministry of Science and Research. The research was partially supported by the Austrian Science Fund (FWF): W1214-N15, project DK8.

Linz, April 2014 Mykhaylo Yudytskiy

Contents

Chapter 1

Introduction

1.1 Challenges of the future ground-based telescopes

Ever since the early civilizations people have been curious about the celestial objects they saw in the sky. During the course of human history, this curiosity developed into a powerful natural science called astronomy. The need of astronomers to learn more about the universe has led to the development of sophisticated optical imaging tools to study the furthest depths of space.

At the present day, astronomers are equipped with high-tech ground-based telescopes to observe electromagnetic radiation, such as visible light, emitted from the celestial objects. Diffraction defines a fundamental limit of an optical imaging system. Consequently, to improve the resolution of the image larger telescopes have been built with an ever increasing mirror diameter.

This fact lead to the development of most advanced *very large telescope* projects such as the Very Large Telescope (VLT) and the Gran Telescopio Canarias with a primary mirror of 8.2 and 10.4 meters across, respectively. The primary mirrors of the next generation telescopes, called *extremely large telescopes*, are planned to greatly surpass that size.

The European Extremely Large Telescope (E-ELT) is the future 40 meter-class telescope of the European Southern Observatory (ESO). The E-ELT and other extremely large telescopes will push the astronomical research frontier and will, among other goals, increase the chance of finding Earth-like planets. To a large extent, the feasibility of the present and future telescopes is made possible with the recent developments in other science disciplines.

Amongst many challenges behind operating present and future telescopes is image distortion due to the atmospheric turbulences. The fluctuating refractive index of air in the Earth's atmosphere induces aberrations in the wavefront that degrade the imaging quality of the telescope. In fact, this deficiency becomes more pronounced as the telescope diameter increases. Even for the very large telescopes of today, atmospheric turbulence is the major limiting factor far beyond the diffraction limit. It is a great challenge to find ways to achieve diffraction-limited imaging for the future ground-based telescopes.

Powerful adaptive optics (AO) techniques have been developed in the last few decades to remedy this problem. Adaptive optics refers to real-time compensation for the distortions in the wavefronts of incoming light due to the atmospheric turbulence. Application

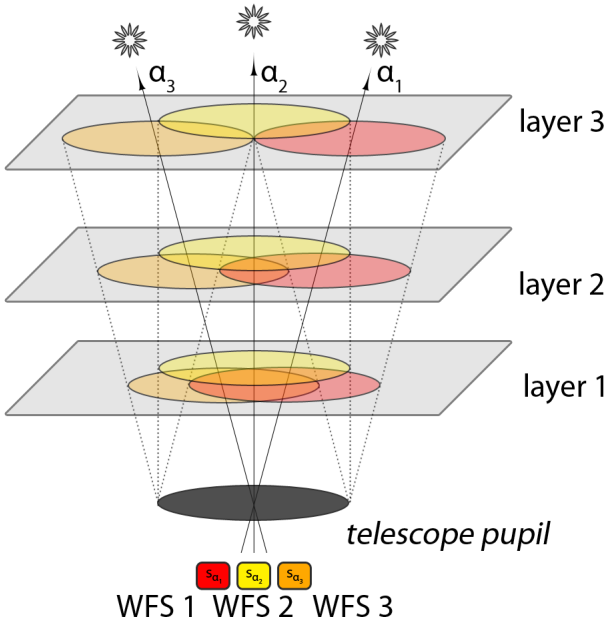

Figure 1.1: In atmospheric tomography, turbulence layers are reconstructed using wavefront sensor (WFS) measurements in the directions of several guide stars.

of AO involves a number of engineering challenges, but the resulting benefits of such techniques have been overwhelming. Adaptive optics correction is an essential part of current and future telescopes.

Principal components of an AO system are a wavefront sensor (WFS) that measures the wavefront distortions, a bright guide star (GS) used as a reference point for the measurements, a deformable mirror (DM) that provides the wavefront correction and a computer that determines the shape of the DM from the obtained WFS measurements. Due to the rapid changes in the atmosphere wavefront compensation must be performed in the millisecond time frame.

In the classical AO, a single guide star, i.e., a single light source is observed. The AO system corrects the cumulative effect of the turbulence towards directions close to the guide star with a DM. Classical AO is also referred to as single conjugate adaptive optics (SCAO).

The future AO modalities, such as the multi object adaptive optics (MOAO) or the multi conjugate adaptive optics (MCAO), extend this idea by using several guide stars and multiple deformable mirrors. First, data obtained from the guide stars is used to solve a reconstruction of the turbulence profile in the atmosphere. This problem is called atmospheric tomography, see Figure 1.1.

Second, having multiple deformable mirrors and the three dimensional reconstruction of the turbulence enable the system to correct for a much wider field of view than in the classical implementation. Thus, several objects of interest can be observed simultaneously.

The dimension of data that the AO computer is expected to handle for extremely large telescopes is significantly larger than for the present telescopes. To bring such a complex system into the realm of feasible implementations, the software and hardware of

the AO computer must be able to handle a substantial amount of data in the constrained time frame.

The work in this thesis was largely carried out within a project established towards developing fast mathematical algorithms and software for the AO systems of the E-ELT. The goal was to find a new and different approach that is as effective as current techniques while, at the same time, greatly reducing the computing load on the computer that performs the calculations.

1.2 State of the art

The mathematical problem behind atmospheric tomography is severely ill-posed and is very closely connected to limited angle tomography [11]. The mathematical challenges and future prospects for inverse problems field in AO were comprehensively reviewed by Ellerbroek and Vogel in [15]. With the arrival of next generation implementations of adaptive optics, the severely ill-posed atmospheric tomography problem becomes the crux of mathematical AO research.

The physics of turbulence is an extensive field of study and much is understood about how the turbulence in the atmosphere is formed. Statistical models for turbulence are frequently utilized in the adaptive optics literature by postulating the tomography step as a Bayesian inference problem. In addition, this makes it possible to take into account the statistical nature of the measurement noise. The maximum a posteriori (MAP) estimate is the standard point estimate used to describe the corresponding Gaussian posteriori distribution.

In literature, both iterative and non-iterative solution methods have been proposed for the MAP estimator. The computational cost of the non-iterative matrix-vector multiply (MVM) methods, which are currently the only working methods for an MCAO system [43], scales with $\mathcal{O}(n^2)$, where n is the dimension of the problem. With the launch of the next generation extremely large telescopes the dimension of the problem increases rapidly. Even with heavy parallelization, the MVM and other non-iterative methods have a high computational cost and the research in recent years has been inclined towards iterative methods. Most importantly, iterative methods benefit from fast system update. However, more work is needed to develop a fast iterative method which is robust under the real-life effects and bad imaging conditions.

Let us shortly describe the existing iterative methods. The iterative solvers of the MAP estimate are typically based on conjugated gradient (CG) methods, see [13] and references therein. Effort has been put into developing an efficient preconditioner for the problem. Multigrid preconditioners and Fourier domain preconditioners have been proposed [26, 24, 77, 75], which led to the MG-PCG and the FD-PCG methods with a computational complexity that scales with $\mathcal{O}(n^{3/2})$ and $\mathcal{O}(n \log n)$, respectively.

Especially for iterative methods it is of value to be able to represent the operators in a sparse form. Often, a sparse approximation of the inverse covariance of the prior is required. In the approach introduced by Ellerbroek [12] the turbulence power law is modified in order to achieve a sparse approximation by biharmonics. Later, a very promising CG based method called the Fractal Iterative Method (FrIM) has been developed by Tallon and others in [70, 66], where an efficient sparse factorization of the covariance term

yielded a method with $\mathcal{O}(n)$ complexity.

Outside of the Bayesian framework the atmospheric tomography problem has been approached by an algorithm based on the Kaczmarz iteration by Ramlau and Rosensteiner in [54, 60]. The authors obtain a fast "matrix-free" solver by decoupling the wavefront reconstruction and the tomography problem in two separate steps. Wavefront reconstruction for the Kaczmarz method is performed with, e.g., the CuReD algorithm [58] and the whole method scales globally with $\mathcal{O}(n)$ operations.

1.3 Overview of our method

In this work we present the Finite Element-Wavelet Hybrid Algorithm (FEWHA) for the atmospheric tomography problem. The FEWHA is a novel wavelet-based iterative method, which solves the MAP estimate using a CG-based algorithm. The method was introduced in [33, 78, 79], where significant contributions including major ideas and numerical implementations were done by the author of this thesis.

One of the key features of our method is to utilize the fast decay properties of compactly supported orthonormal wavelets in the frequency domain. This allows us to approximate the inverse covariance of the prior with a completely diagonal representation, see [33]. A frequency-dependent preconditioning technique based on this approximation was introduced in [78] for our method to reduce the number of iterations.

In addition to the frequency-dependent preconditioner, we further improve convergence and stabilize the reconstructor by utilizing the multi-scale structure of wavelets. Namely, we solve a sub-problem defined for the coarse wavelet scales of the ground layer to improve the initial guess of the CG iteration. We call this technique the ground layer multi-scale (GLMS) method.

Numerical efficiency of the method is directly related to the sparsity of the underlying operators. To achieve a computationally efficient representation, we apply a dual-domain discretization strategy, in which we discretize the problem with the piecewise bilinear basis of finite elements and the wavelet basis, simultaneously. A fast transition between the two bases is possible with the discrete wavelet transform (DWT). The DWT is a method of linear complexity and is parallelizable. The GLMS method and the dual-domain discretization strategy are documented in [79].

By combining the dual-domain discretization strategy, the frequency-dependent preconditioner and the GLMS method, the FEWHA scales globally as $\mathcal{O}(n)$. Moreover, in the implementation of our algorithm we are able to formulate most of the components of the method as "matrix-free" operators. This simultaneously improves the computational performance and reduces the required storage of the method in memory. Furthermore, our method is highly parallelizable.

These features make our method especially suitable for a modern computing architecture, such as a multi-core CPU. In numerical simulations, we evaluate the reconstruction time of our method and demonstrate fast performance of the reconstructor even on non-dedicated off-the-shelf hardware. In these tests, the method is compared to the MVM in terms of speed.

In terms of reconstruction quality, the performance of the FEWHA is validated on OCTOPUS [40], the end-to-end simulation tool of the ESO. We show the advantages of

the method in low flux imaging conditions, i.e., when only a low number of photons can be measured, and with respect to some practical phenomena that are well-known to limit the reconstruction quality. Further, we compare our algorithm to the state of the art methods, the MVM and the FrIM, and show that the wavelet reconstructor performs at the level of or surpasses the quality attained by these benchmark methods. Additionally, we point out the stability of our method and show that it is easy to configure.

Let us mention that wavelets have been applied in single conjugate adaptive optics in [31, 30]. To the best of our knowledge wavelets have not been considered before in the context of atmospheric tomography prior to [33]. In the field of inverse problems wavelets are applied widely, see, e.g., [64, 37]. For an extensive introduction to wavelets we refer to [10].

This work is organized as follows. The thesis is split into two essential parts. In Chapters 2-4 we discuss the results that are currently available in literature and in Chapters 5-9 we present new research based on our work in [33, 78, 79].

- In Chapter 2, we briefly introduce adaptive optics and its components. We present the typical mathematical models of telescope imaging and of the atmospheric turbulence. We also give a brief overview of the physical components of the adaptive optics system and present a few AO system configurations, which are handled in this work.

- In Chapter 3, we present an excerpt of the standard mathematical tools that are necessary to define the problem and to state our solution. We shortly cover the fields of deterministic and statistical inverse problems, wavelets and methods for solving systems of linear equations.

- In Chapter 4, we formulate the atmospheric tomography problem, the focus of our thesis. We also give a small overview of the state of the art methods to solve this problem.

- In Chapter 5, we state our solution method for the atmospheric tomography problem, the Finite Element-Wavelet Hybrid Algorithm. We show that the turbulence statistics of the atmosphere have a sparse representation in the wavelet domain. Moreover, we present the dual-domain discretization strategy, in which components of the method are discretized in the finite element basis and the wavelet basis simultaneously.

- We solve the atmospheric tomography problem using a preconditioned conjugate gradient (PCG) iteration. In Chapter 6, we define the frequency-dependent preconditioner for our problem and formulate the GLMS method, a multi-scale method to improve the initial guess of the PCG algorithm.

- Our reconstruction algorithm for different AO systems is introduced in Chapter 7. The reconstruction algorithm incorporates our solution method to the atmospheric tomography, presented in Chapters 5 and 6, together with mirror fitting and temporal control.

- In Chapter 8, the performance of our method in terms of quality is presented. The tests are evaluated using numerical simulations. Here we compare our algorithm

to other state of the art methods in various adaptive optics system configurations. We also investigate the influence of parameters on the performance of our method.

- In Chapter 9, the performance of our method is validated in terms of speed. Here, we discuss an efficient "matrix-free" formulation of our method and talk about parallelization. We then derive computational cost estimates and demonstrate the numerical performance by evaluating the computing time for our reconstructor.
- In Chapter 10, we state our conclusions and outlook.

Chapter 2

Adaptive optics

2.1 Telescope imaging

Several limiting factors affect the resolution of an optical imaging system. Diffraction always defines a fundamental physical limit to the resolution, due to the finite dimension of the instrument's aperture. In ground-based telescope imaging, the quality of the image is further reduced by the effects of atmospheric turbulence. When observing a distant light source, such as a star or a galaxy, the wavefront of light is distorted when it passes through the Earth's atmosphere.

In isoplanatic Fourier optics model [28], the relation between an observed object O and the resulting image I is described by a convolution with the so called *point spread function* (PSF),

$$I(x) = \int_{\mathbb{R}^2} PSF_{\tilde{\varphi}}(x - x')O(x')\mathrm{d}x' = (O * PSF_{\tilde{\varphi}})(x). \tag{2.1}$$

Here, $PSF_{\tilde{\varphi}}$ is the PSF and $x = (x_1, x_2) \in \mathbb{R}^2$ are the coordinates in the focal plane, see Figure 2.1.

The PSF depends on the *aperture function*, which for simplicity is the characteristic function on a circular aperture of the telescope $\Omega \subset \mathbb{R}^2$, and the *phase* $\tilde{\varphi}$, and is defined by

$$\begin{aligned} PSF_{\tilde{\varphi}} : \mathbb{R}^2 &\to \mathbb{R}, \\ x &\mapsto |\mathcal{F}(\chi_\Omega e^{i\tilde{\varphi}})(x)|^2. \end{aligned} \tag{2.2}$$

The phase $\tilde{\varphi}$ is measured in radians and $\mathcal{F}$ is the Fourier transform. For a function $f : \mathbb{R}^d \to \mathbb{R}$ and $x \in \mathbb{R}^d$, where $d = 2$ or 3, we use the following definition of the transform,

$$\mathcal{F}(f)(x) := \int_{\mathbb{R}^d} f(y)e^{-2\pi i x \cdot y}\mathrm{d}y. \tag{2.3}$$

The aperture function and the phase are functions in the aperture plane, see Figure 2.1. Whereas the aperture function is fixed by the geometry of the telescope, the phase of light $\tilde{\varphi}$ fluctuates over time.

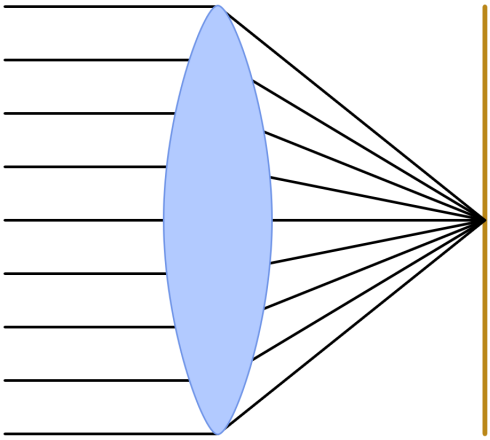

Figure 2.1: The lens is in the aperture plane (a). The light of a plane wave is gathered in one point on the focal plane (b).

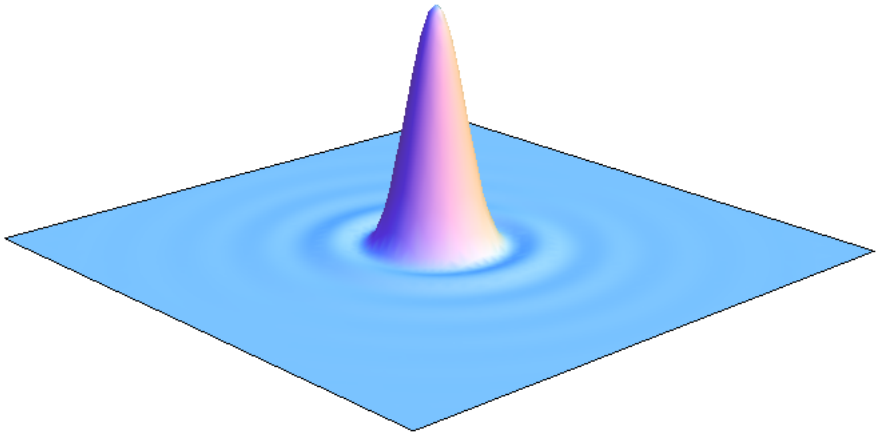

Figure 2.2: The image of a point source in a diffraction limited telescope is given by an Airy function.

The theoretical limit of an optical imaging system is known as the *diffraction limit*. For this ideal case consider incoming plane waves for which $\tilde{\varphi}(y_1, y_2) = c$ at the aperture for some constant $c \in \mathbb{R}$. Thus, the image of the observed object is perturbed only by diffraction and the PSF is given by

$$PSF_0(x) = |\mathcal{F}(\chi_\Omega)(x)|^2. \tag{2.4}$$

The function PSF_0 is also known as the *Airy function*, see Figure 2.2. Asymptotically, as the diameter of the telescope goes to infinity, PSF_0 tends towards the delta distribution and the image resolution improves.

Another example that illustrates the effect of the phase on the PSF are the *tip-tilt aberrations*. Let $\tilde{\varphi}(y_1, y_2) = 2\pi(t_1 y_1 + t_2 y_2)$ be given with fixed parameters $t_1, t_2 \in \mathbb{R}$, which are related to the tip and the tilt slopes of the incoming wavefront, respectively.

Then the PSF is given by

$$\begin{aligned} PSF_{\tilde{\varphi}}(x_1, x_2) &= \left| \int_{\mathbb{R}} \int_{\mathbb{R}} \chi_\Omega(y_1, y_2) e^{2\pi i(t_1 y_1 + t_2 y_2)} e^{-2\pi i(x_1 y_1 + x_2 y_2)} \mathrm{d}y_1 \mathrm{d}y_2 \right|^2 \\ &= \left| \int_{\mathbb{R}} \int_{\mathbb{R}} \chi_\Omega(y_1, y_2) e^{-2\pi i((x_1 - t_1) y_1 + (x_2 - t_2) y_2)} \mathrm{d}y_1 \mathrm{d}y_2 \right|^2 \\ &= PSF_0(x_1 - t_1, x_2 - t_2). \end{aligned} \tag{2.5}$$

In other words, tip-tilt aberrations shift the diffraction limited PSF according to the parameters (t_1, t_2). This results in a shift of the focus of the image, according to (2.1).

An important measure of imaging quality in astronomy is the *Strehl ratio*. The Strehl ratio relates the PSF to the peak diffraction limited PSF in the origin by

$$S_{\tilde{\varphi}} := \frac{PSF_{\tilde{\varphi}}(0,0)}{PSF_0(0,0)}. \tag{2.6}$$

Higher Strehl ratio implies a better resolution of the image. Since the diffraction limited optical system sets the theoretical limit of performance, the Strehl ratio is bounded from above by 1.

Towards directions close to the zenith the ratio can be estimated by the Maréchal approximation [55, 74] according to

$$S_{\tilde{\varphi}} \approx \exp(-\frac{1}{|\Omega|} \|\tilde{\varphi} - \bar{\varphi}\|^2_{L^2(\Omega)}), \tag{2.7}$$

where

$$\bar{\varphi} = \frac{1}{|\Omega|} \int_\Omega \tilde{\varphi}(y) \mathrm{d}y \tag{2.8}$$

is the average phase at the aperture and $|\Omega| := \int_\Omega \mathrm{d}y$ is the area of the aperture Ω. This approximation relates the L^2-error of the phase with the Strehl ratio at the aperture.

The phase $\tilde{\varphi}$ is related to *wavefront aberrations*, which we denote by φ, according to

$$\tilde{\varphi}(y) = \frac{2\pi}{\lambda} \varphi(y), \tag{2.9}$$

where λ is the wavelength on the incoming light and $y \in \mathbb{R}^2$ is a point in the aperture plane. The wavefront aberrations, also referred to as *the optical path distance*, are measured in meters; the unit of the phase is radians. In this work, the quantity of interest are the wavefront aberrations φ.

The main source for wavefront aberrations is the atmospheric turbulence, discussed in the next section. One possible way to avoid these distortions is to use a space telescope, which is positioned above the atmosphere of the Earth. In ground-based telescope imaging, the quality of the image can be improved using an adaptive optics (AO) system.

Utilizing hardware based techniques in real time, a classical AO system aims at reducing the wavefront aberrations of light. In doing so, the $PSF_{\tilde{\varphi}}$ tends closer to the diffraction limited case. Thus, in mathematical terms, the AO system has an effect of "improving" the convolution kernel in equation (2.1) to increase the resolution of the image.

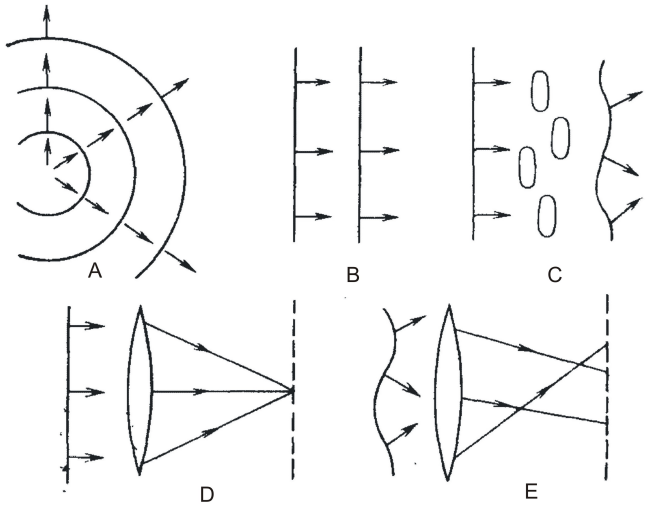

Figure 2.3: Wavefront distortions: A - a spherical wavefront coming from a close point light source; B - a plane wavefront coming from a distant point light source without atmospheric distortions; C - the plane wavefront distorted by the refractive index inhomogeneity; D - the plane undistorted wavefront focuses into a point; E - the plane distorted wavefront does not focus into a point. Source: [68].

2.2 Turbulence in the atmosphere

The main source of distortions for the wavefront of light is the *atmospheric turbulence*. Atmospheric turbulence emerges from irregular mixing of cold and hot air, affected by the sun and the wind. Due to these irregularities, the refractive index of air is inhomogeneous throughout the atmosphere. This causes the wavefront which propagates as plane waves to arrive distorted at the pupil of the ground-based telescope after passing through the Earth's atmosphere.

The effects of atmospheric turbulence on a propagating wavefront are illustrated in Figure 2.3. Figures 2.3 A and B show a wavefront of light that propagates from a close point source and a distant point source, such as a star or a galaxy, respectively. Inhomogeneities of the air refractive index distort the plane wavefront, see Figure 2.3 C. The focus of a plane wavefront and a distorted wavefront are illustrated in Figures 2.3 D and E, respectively. The distortions of the wavefront due to the turbulence in the atmosphere result in blurring or twinkling of the image.

Typically, the effects of atmospheric turbulence are unpredictable and thus are modeled by random processes. A random process describes a sequence of random variables over time or space. In this work we neglect the dependence on time and focus only on the space variable to describe a random process.

The index of refraction of the atmosphere $n(x)$ is modeled as the sum of a mean index of refraction n_0, which for air is $n_0 = 1$, and a randomly fluctuating term $n_1(x)$,

$$n(x) = n_0 + n_1(x), \tag{2.10}$$

see, e.g., [56]. The fluctuating term is described via a random process that depends on a three-dimensional space variable $x = (x_1, x_2, h) \in \mathbb{R}^2 \times [0, \infty)$, where h is the altitude.

The fundamental model of turbulence in the atmosphere is due to Kolmogorov. It was suggested by Kolmogorov that the statistical behavior of the atmosphere is modeled by an isotropic stationary random process [38, 56], i.e., the statistics of the process are the same at any spatial point and the correlation of the process is described via the radial distance $|\Delta x|$ between any two points x and $x + \Delta x$. Here, $|x| := \sqrt{x_1^2 + x_2^2 + h^2}$ is the Euclidean norm.

The tools used to describe a random process are the *structure function*, which describes the expected difference of values of a random process at any two points and the *covariance function*, which measures the spatial covariance of a random process. For a stationary process, both of the these functions depend only on the separation Δx, rather than a specific point x.

The structure function of a stationary process f is defined by

$$D_f(\Delta x) := \mathbb{E}\left((f(x + \Delta x) - f(x))^2\right), \tag{2.11}$$

and the covariance function is given by

$$C_f(\Delta x) := \mathbb{E}\left((f(x) - \mathbb{E}(f))(f(x + \Delta x) - \mathbb{E}(f))\right). \tag{2.12}$$

Here, $\mathbb{E}$ denotes the expected value. The behavior of the covariance function in the Fourier domain,

$$\Phi_f(\kappa) := \mathcal{F}(C_f)(\kappa). \tag{2.13}$$

is known as the *power spectral density* (PSD) or *power spectrum*. Here $\kappa = (\kappa_1, \kappa_2, \kappa_3)$ is the frequency in the Fourier domain, associated to the three-dimensional spatial variable x.

It was stated by Kolmogorov [38, 19], that within a certain range of separations, the structure function of the refractive index of the atmosphere at altitude h is of the form

$$D_n(\Delta x) = C_n^2(h)|\Delta x|^{2/3}, \tag{2.14}$$

for $\ell_0 < |\Delta x| < L_0$. The quantity $C_n^2(h)$, called the *refractive index structure constant*, measures the optical strength of the turbulence at the altitude h. The constant depends on the weather conditions and is usually derived empirically. The term *turbulence profile* refers to the dependence of $C_n^2(h)$ on the altitude h.

The bounds ℓ_0 and L_0 are known as the *inner scale* and the *outer scale*, respectively. They correspond to the size of the smallest and the largest eddies in the turbulence. Typical values for ℓ_0 are a few millimeters; values for L_0 vary between 1 and 100 meters, see e.g., [28].

Further, the PSD of the refractive index n is predicted by the Kolmogorov theory [34],

$$\Phi_n(\kappa) = 0.033 C_n^2(h)|\kappa|^{-11/3}, \tag{2.15}$$

for $2\pi L_0^{-1} < |\kappa| < 2\pi \ell_0^{-1}$. The predicted behavior of the PSD is given only within a so called *inertial range*, which is defined by the inner and outer scale. For very small frequencies $|\kappa|$, i.e., very large eddies, the Kolmogorov model breaks down due to the singularity at the origin.

Typically the von Karman spectrum has been used [34] to avoid this singularity, defined by

$$\Phi_n(\kappa) = \frac{0.033 C_n^2(h)}{(|\kappa|^2 + \kappa_0^2)^{11/6}} \exp\left(-\frac{|\kappa|^2}{\kappa_m^2}\right), \tag{2.16}$$

where $\kappa_0 = 2\pi L_0^{-1}$ and $\kappa_m = 5.92\ell_0^{-1}$. Note that for the large frequencies, $|\kappa| > \kappa_m$, the PSD exhibits a fast decay towards 0, which corresponds to the observation that turbulent air motion ceases at scales smaller than ℓ_0. In some formulations, the exponent term is omitted, as it only affects the high frequencies, where the value is small.

Using geometrical optics and neglecting scattering of light, the phase in the direction of zenith and the aberrations of the incoming wavefront are related to the index of refraction according to

$$\tilde{\varphi}(x_1, x_2) = \frac{2\pi}{\lambda}\varphi(x_1, x_2) = \frac{2\pi}{\lambda}\int_0^H n_1(x_1, x_2, h)\mathrm{d}h \tag{2.17}$$

in radians, where λ is the wavelength, see [61, 69]. Here, the spatial coordinates of the wavefront (x_1, x_2) are given at the aperture of the telescope. Further, $H > 0$ is the finite height of the atmosphere.

Statistics of the phase are derived from the statistics of the refractive index, see [56]. The structure function of the phase is given by

$$D_{\tilde{\varphi}}(\Delta x) = 6.88\left(\frac{|\Delta x|}{r_0}\right)^{5/3}, \tag{2.18}$$

where r_0 is the *Fried parameter*, defined via

$$r_0 = 0.185\left(\frac{\lambda^2}{\int_0^H C_n^2(h)\mathrm{d}h}\right)^{3/5}. \tag{2.19}$$

The Fried parameter gives a qualitative assessment of the seeing conditions, relating a particular wavelength and the total air mass. If no wavelength is specified, the wavelength of 500 nanometers is assumed. Typically, the Fried parameter is in the range of 10 to 20 centimeters for a visible region of the spectrum.

The PSD of the phase for the von Karman spectrum is given via

$$\Phi_{\tilde{\varphi}}(\kappa) = \frac{0.023 r_0^{-5/3}}{(|\kappa|^2 + \kappa_0^2)^{11/6}}. \tag{2.20}$$

In this form, the exponent term, corresponding to the high frequencies, is omitted.

In many applications it is sufficient to consider a layered model of the atmosphere, see, e.g., [56]. The model states that the turbulence is concentrated on $L \in \mathbb{N}$ layers of the atmosphere at distinct altitudes $0 \leq h_1 < ... < h_L < H$. Moreover, the atmosphere is sliced into L vertical blocks $[h_\ell, h_\ell + \Delta h_\ell]$ with $h_{\ell+1} = h_\ell + \Delta h_\ell$ for $1 \leq \ell < L$ and $H = h_L + \Delta h_L$, where the refractive index structure constant $C_n^2(h)$ is approximately constant; we denote it by $C_n^2(h_\ell)$.

A *turbulence layer* is defined by the wavefront aberrations component on the corresponding block,

$$\phi_\ell(x_1, x_2) = \int_{h_\ell}^{h_\ell+\Delta h_\ell} n_1(x_1, x_2, h)\mathrm{d}h. \tag{2.21}$$

Thus, the propagation of the incoming wavefront in the direction of zenith (2.17) reduces to a finite sum,

$$\varphi(x_1, x_2) = \sum_{\ell=1}^{L} \phi_\ell(x_1, x_2), \tag{2.22}$$

and the turbulence layer has its statistics described by the PSD of the component of the phase at layer ℓ,

$$\Phi_{(2\pi/\lambda)\phi_\ell}(\kappa) = \frac{0.023 r_0^{-5/3} C_n^2(h_\ell)}{(|\kappa|^2 + \kappa_0^2)^{11/6}}. \tag{2.23}$$

2.3 Classical AO

The goal of a classical adaptive optics (AO) system is to correct the wavefront aberrations caused by the turbulent atmosphere. The AO system consists of four primary components: a deformable mirror (DM), a guide star (GS), a wavefront sensor (WFS) and a computer. In this section a brief overview of the interaction between these components is presented. These components are described in more detail in the following sections.

Wavefront correction in AO is performed using a *deformable mirror*. A DM is a segmented mirror that can be controlled using a computer. Parts of a DM move in the micrometer range; the shape of the DM can be updated in under a millisecond. In AO, the mirror is deformed in such a way that the reflection of the distorted wavefront is as close as possible to a plane wave, see Figure 2.4. Rapid changes of the wavefront due to the fluctuations of the refractive index requires the DM shape to be updated in the millisecond time frame.

A *guide star* is a bright source of light which is used as a reference for the AO system. It could be a bright star in the sky, in which case it is referred to as a *natural guide star* (NGS). Typically, the scientific object to be observed is too faint to be used as a guide star. If no bright stars can be found in the vicinity of the scientific object, an artificial GS can be generated using a powerful laser beam; in this case it is referred to as a *laser guide star* (LGS). A laser beam that is shot into the air scatters in the sodium layer of the atmosphere, above the turbulent layers, to produce a bright beacon of light. Apart from the added operational cost, an LGS has other issues related to it; these issues are discussed in the Section 2.6.2.

A *wavefront sensor* indirectly measures the distortions in the phase of light coming from a guide star. The measurements produced by the WFS are sent to a computer, which determines the new shape of the DM. See Figure 2.5 for a schematic representation of a full AO system. In the telescope the light is reflected by one or several deformable mirrors. Using a beam splitter, part of the light is sent to the scientific instrument, such as a camera, whereas part of the light is sent to a WFS. The WFS measures the aberrations in the phase of light and a computer, running a mathematical algorithm, determines how to update the DM to reduce the distortion in the wavefront.

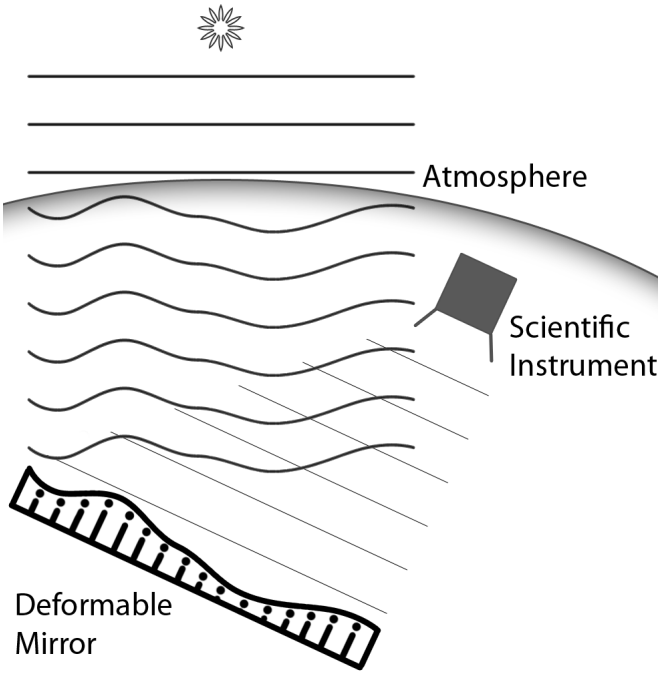

Figure 2.4: Perturbed phase of light is reflected from a deformable mirror as a plane wave.

2.4 Deformable mirror

A deformable mirror typically consists of a thin surface to reflect light and a set of actuators that drive the mirror, see Figure 2.4. There are several types of driving actuators used in AO systems, such as, electromechanical, electromagnetic and piezoelectric, see, e.g., [74, 71].

There are also several kinds of DM designs. For instance, in segmented DM a thin faceplate is bonded to each actuator separately. A continuous DM uses a thin reflective facesheet stretched over all actuators. By pushing one actuator on a continuous DM, a neighboring actuator is also affected.

In this work we assume the simple model of a bilinear DM. To a certain approximation, such a model is sufficient, especially for an AO systems that operates in closed loop, see Section 2.7. The shape of a bilinear DM is described using a piecewise continuous bilinear function a, which is defined below.

Let $\Omega := [-D/2, D/2]^2$ be the domain on which the DM is defined and let D be the diameter of the telescope aperture. The domain is centered at the origin and is symmetric in both axes.

The number of *actuators*, or nodal points of the piecewise bilinear function, is denoted by n_a^2. We assume they are arranged on a rectangular grid of points with spacing $d := D/(n_a - 1)$. Note that not all of these actuators need to be active; due to the circular shape of the telescope some actuators have no effect on the wavefront correction.

The actuator grid is given by a set of points,

$$\{(x_i, x_j) : 0 \leq i, j < n_a\}, \tag{2.24}$$

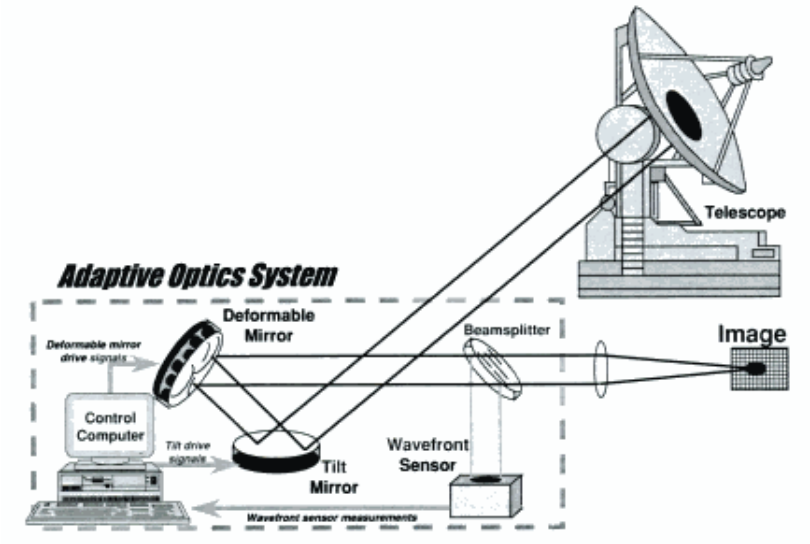

Figure 2.5: An AO system. Source: [74].

where

$$x_i := -D/2 + di, \tag{2.25}$$

for $i = 0, ..., n_a - 1$.

We define square sub-domains of Ω by

$$\Omega_{ij} := [x_i, x_{i+1}] \times [x_j, x_{j+1}] \tag{2.26}$$

for $0 \leq i, j < n_a - 1$. Observe that the sub-domains cover Ω, i.e.,

$$\Omega = \bigcup_{0 \leq i,j < n_a - 1} \Omega_{ij}. \tag{2.27}$$

To each subdomain Ω_{ij} we associate a bilinear function defined on $[0, 1]^2$,

$$\begin{aligned} b_{ij}(x, y) &= c_0 + c_1 x + c_2 y + c_3 xy \\ &= a_{ij}(1 - x - y + xy) + a_{i,j+1}(x - xy) + a_{i+1,j}(y - xy) + a_{i+1,j+1}xy. \end{aligned} \tag{2.28}$$

Here, $c_0, ..., c_3$ are some constants and $a_{ij}, a_{i,j+1}, a_{i+1,j}, a_{i+1,j+1}$ are nodal values associated to the corner points of the sub-domain. A linear mapping between Ω_{ij} and $[0, 1]^2$ is given by

$$\begin{aligned} t_{ij} : \Omega_{ij} &\to [0, 1]^2, \\ (x, y) &\mapsto (\frac{x - x_i}{d}, \frac{y - x_j}{d}). \end{aligned} \tag{2.29}$$

Using all of the above, we define a continuous piecewise bilinear function on Ω by

$$\begin{aligned} a : \Omega &\to \mathbb{R}, \\ (x, y) &\mapsto (b_{ij} \circ t_{ij})(x, y) \quad \text{for } (x, y) \in \Omega_{ij}. \end{aligned} \tag{2.30}$$

Observe that $a(x_i, x_j) = a_{ij}$, i.e., a_{ij} are the nodal values of a, and that a is indeed continuous due to the linearity in each of the coordinates.

We refer to the set (2.24) as the *actuator positions* and to the values a_{ij} as the *actuator commands*. Since the function is uniquely defined via its nodal values we often interchange between the nodal values and the DM shape, when referring to a.

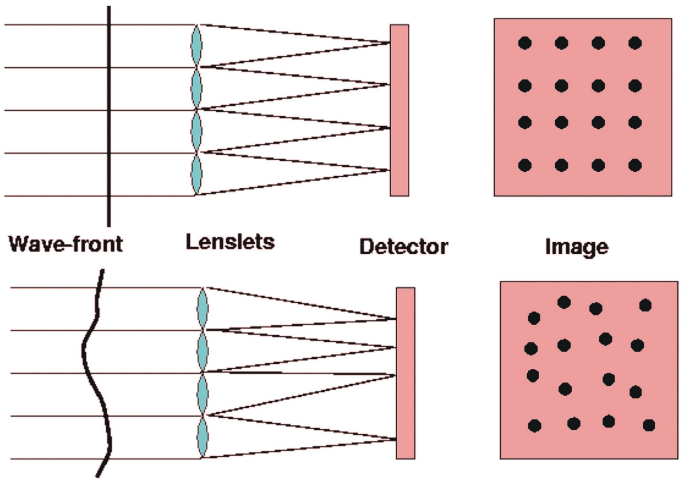

Figure 2.6: A Shack-Hartmann WFS. A plane wave corresponds to a regular grid of the focal points (top). Deformations of the wavefront are observed as shifts of the focal points on the detector plane (bottom). Source: [71].

2.5 Wavefront sensor

A WFS indirectly measures the wavefront aberrations. Various sensors are used in AO, each with their own functionality, advantages and a mathematical model that relates the wavefront to measurements. Some of the sensors used in AO are: the Shack-Hartmann wavefront sensor (SH-WFS), the curvature WFS and the pyramid WFS [74].

In this work we focus on the SH-WFS. The Shack-Hartmann sensor utilizes the relation (2.5), which states that a linear aberration in the wavefront results in a shift of the focal point of the image. By measuring the shift in the x- and y-directions, the slope of the wavefront can be inferred.

Following this relation, a SH-WFS utilizes an array of little lenses, each focused on a CCD detector plane. The vertical and the horizontal shifts of the focal points determine the average slope of the wavefront over the area of the lens, also known as a *subaperture*. The Shack-Hartmann sensor is illustrated in Figure 2.6.

To define the mathematical model of the SH-WFS we first introduce the subaperture grid. Let $\Omega := [-D/2, D/2]^2$ be the domain on which the wavefront is defined. Again, D is the diameter of the telescope. Subapertures are modeled by square sub-domains of Ω.

Let the number of subapertures be denoted by n_s^2. Just as for the actuators, not all subapertures are active, i.e., due to the shape of the telescope pupil not all subapertures are illuminated. Typically, the *active subapertures* are arranged in a circular shape with a central obstruction, see Figure 2.7. The area of a subaperture is d^2, where $d := D/n_s$.

The subaperture grid is defined in a similar way to the actuator grid. Let

$$x_i := -D/2 + di \tag{2.31}$$

for $i = 0, ..., n_s$ and let

$$\{(x_i, x_j) : 0 \leq i, j \leq n_s\} \tag{2.32}$$

be a grid of points with equidistant spacing. We define a subaperture as an open square

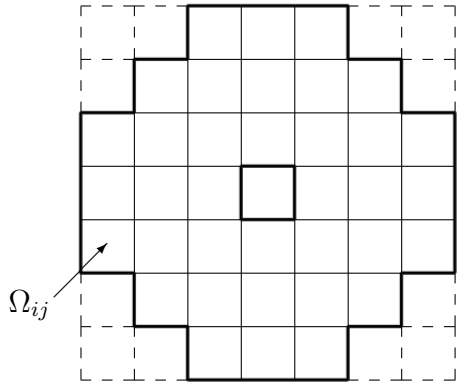

Figure 2.7: A typical arrangement of active subapertures due to the shape of the telescope's circular aperture with central obstruction.

sub-domain of Ω by

$$\Omega_{ij} := (x_i, x_{i+1}) \times (x_j, x_{j+1}), \tag{2.33}$$

for $0 \leq i, j < n_s$. Observe that the closure of the sub-domains cover Ω, i.e.,

$$\Omega = \bigcup_{0 \leq i,j < n_s} \overline{\Omega}_{ij}. \tag{2.34}$$

Further, we associate an index set

$$\mathcal{M} \subseteq \{(i,j) : 0 \leq i, j < n_s\} \tag{2.35}$$

to those indices that correspond to active subapertures Ω_{ij}. Such an index set is often referred to as the *mask*. We refer to the domain of all active subapertures of a SH-WFS by

$$\Omega_{\mathcal{M}} := \bigcup_{(i,j) \in \mathcal{M}} \overline{\Omega}_{ij}. \tag{2.36}$$

The Shack-Hartmann measurements in a subaperture Ω_{ij} are modeled by the average slopes of the wavefront aberration φ in Ω_{ij}, i.e.,

$$\begin{aligned} s_{ij}^x &= \frac{1}{d^2} \int_{\Omega_{ij}} \frac{\partial \varphi}{\partial x}(x,y) \mathrm{d}(x,y), \\ s_{ij}^y &= \frac{1}{d^2} \int_{\Omega_{ij}} \frac{\partial \varphi}{\partial y}(x,y) \mathrm{d}(x,y), \end{aligned} \tag{2.37}$$

where d^2 is the area of Ω_{ij} and $(s_{ij}^x, s_{ij}^y) \in \mathbb{R}^2$ are the average x- and y-slopes in Ω_{ij}.

If the incoming wavefront aberration is described by a continuous piecewise bilinear function φ with nodal values φ_{ij} at the points $\{(x_i, x_j) : 0 \leq i, j \leq n_s\}$, equations in

(2.37) reduce to

$$s_{ij}^x = \frac{(\varphi_{i,j+1} - \varphi_{ij}) + (\varphi_{i+1,j+1} - \varphi_{i+1,j})}{2}, \qquad (2.38)$$
$$s_{ij}^y = \frac{(\varphi_{i+1,j} - \varphi_{ij}) + (\varphi_{i+1,j+1} - \varphi_{i,j+1})}{2}.$$

In this work we refer to the SH-WFS measurement vector by $s = (s^x, s^y)$, which is a concatenation of two vectors, s^x and s^y. The vectors s^x and s^y are a concatenation of values s_{ij}^x and s_{ij}^y, respectively, for $(i,j) \in \mathcal{M}$, i.e., from the active subapertures Ω_{ij}. Those subapertures where no measurements are available are excluded from s. Hence, the dimension of s is $2|\mathcal{M}| \leq 2n_s^2$, where $|\mathcal{M}|$ is the number of elements in the set $\mathcal{M}$.

To relations (2.37) we associate a *SH-WFS operator* which we denote by $\Gamma = (\Gamma^x, \Gamma^y)$, where Γ^x and Γ^y determine the x- and y-slopes in the subapertures. The SH-WFS operator maps wavefronts to SH-WFS measurements, i.e.,

$$s = \begin{pmatrix} s^x \\ s^y \end{pmatrix} = \begin{pmatrix} \Gamma^x \varphi \\ \Gamma^y \varphi \end{pmatrix} = \Gamma\varphi. \qquad (2.39)$$

A Shack-Hartmann operator applied to a continuous piecewise bilinear phase φ is given by (2.38). We reuse the notation when we apply the Shack-Hartmann operator to the vector of nodal values φ_{ij}; in this case Γ is a matrix.

In practice, the CCD detector senses individual photons over a certain duration of time, referred to as a *time frame*. The Shack-Hartmann wavefront sensor suffers from measurement noise. The typical source of noise of a SH-WFS includes the photon noise and read out noise.

The photon noise relates to the number of photons that are sensed by the CCD in a single subaperture during a time frame. If the light beacon is relatively faint, too few photons are detected by the sensor and the position of the focal point in the subaperture is inaccurate.

The photon noise is modeled by a poisson process, where individual photons are distributed over the CCD array according to the number of photons detected in a time frame. For a large number of photons the photon noise in the measurement slopes can be approximated by a Gaussian random variable. The photon noise is different for an NGS and an LGS, see Section 2.6.

The read out noise refers to the errors in reading the photons by the CCD plane. The read out noise is measured in the number of erroneous electrons per pixel.

2.6 Guide stars

2.6.1 Natural guide star

An NGS is a bright star that serves as a reference point for the WFS to detect wavefront aberrations. The star is modeled as a point source at infinity. Assuming geometric propagation of light, the wavefront at the aperture in a direction close to the zenith is distorted according to

$$\varphi_\theta(x) \approx \int_0^H n_1(x + \theta\xi)\mathrm{d}\xi \qquad (2.40)$$

where $x = (x_1, x_2, 0)$ is a point on the aperture, $\theta = (\theta_1, \theta_2, 1)$ is a direction vector of the guide star close to the zenith, n_1 are the random fluctuations of the refractive index in the atmosphere and H is the height of the atmosphere.

Assuming the layered atmospheric model, wavefront aberrations are given by the finite sum

$$\varphi_\theta(x) = (P_\theta^{\mathrm{NGS}}\boldsymbol{\phi})(x) := \sum_{\ell=1}^{L} \phi_\ell(x + \theta h_\ell), \tag{2.41}$$

where ϕ_ℓ is the turbulence layer at altitude h_ℓ for $\ell = 1, ..., L$. We denote the concatenation of all turbulence layers by a vector

$$\boldsymbol{\phi} = (\phi_1, ..., \phi_L) \tag{2.42}$$

and the geometric propagation operator in the direction of the NGS by P_θ^{NGS}.

In the most simple case, the photon noise from the NGS that affects the SH-WFS measurements is modeled by a Gaussian random variable of dimension $2|\mathcal{M}|$ with zero mean, where $\mathcal{M}$ is the mask, see (2.35).

The noise is identically independently distributed in each subaperture; moreover, the x- and y-measurement noise is uncorrelated. Hence, the covariance matrix of the random variable is given by

$$C_\eta = \sigma^2 I, \tag{2.43}$$

where σ^2 is the noise variance of a single measurement. We use the model

$$\sigma^2 = \frac{1}{n_{\mathrm{photons}}}, \tag{2.44}$$

where n_{photons} is the number of photons per subaperture and frame that the sensor detects. In the section on numerics, we show that this model is sufficient for our simulations. More accurate models for the noise are available, see, e.g., [55].

2.6.2 Laser guide star

The laser guide star is generated artificially when a powerful laser beam is shot into the atmosphere. The laser beam scatters in the sodium layer and the backscattered light is sensed by the wavefront sensor. There are three important effects that need to be taken into account with an LGS: cone effect, spot elongation and tip-tilt indetermination. These effects are explained below.

Cone effect

To model light propagation through the atmosphere, the LGS is considered to be a fixed point at a finite altitude H. Due to the finite altitude, the light detected by the telescope passes through a cone-like volume in the atmosphere, see Figure 2.8. This is referred to as the *cone effect* of the LGS.

In a layered turbulence model, the incoming wavefront aberration in the direction of an LGS is given by

$$\varphi_\theta(x) = (P_\theta^{\mathrm{LGS}}\boldsymbol{\phi})(x) := \sum_{\ell=1}^{L} \phi_\ell\left(\left(1 - \frac{h_\ell}{H}\right)x + \theta h_\ell\right), \tag{2.45}$$

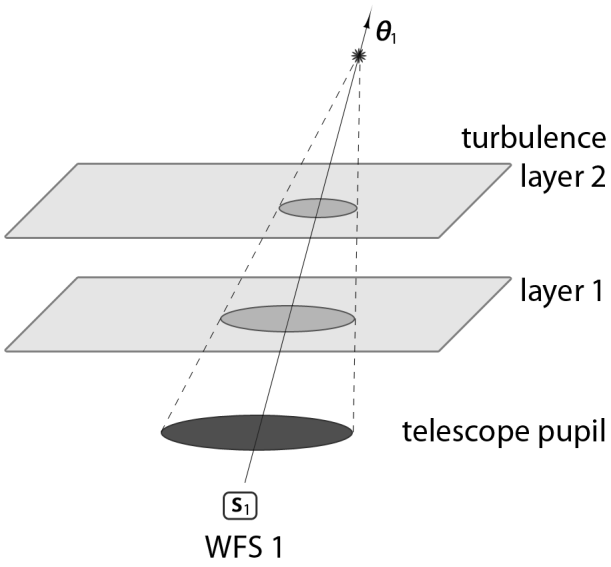

Figure 2.8: The cone effect. Laser guide star light source is fixed above the turbulence layers at some finite altitude. Light traveling from the LGS to the telescope pupil passes through smaller areas at higher turbulence layers.

where P_θ^{LGS} is the geometric propagation operator in the direction of the LGS. As before $x = (x_1, x_2, 0)$ is a point on the aperture, $\theta = (\theta_1, \theta_2, 1)$ is a direction vector of the LGS close to the zenith and $\boldsymbol{\phi}$ is the vector of turbulence layers. The scaling factor $(1 - h_\ell/H)$ reduces the area observed by the telescope at the altitude h_ℓ.

Spot elongation

The LGS photon noise model is more complex than that for the NGS. The sodium layer "thickness" has to be taken into account. As the sodium layer has a certain vertical width, scattering of the laser beam happens not in a single point, but rather in a vertical stripe in the atmosphere. The stripe is observed as an elongated spot by the CCD detector of the WFS, see Figure 2.9. This effect of the LGS is called *spot elongation*. In this work we follow the spot elongation model taken in [2].

The vertical density profile of the laser beam scatter in the sodium layer is modeled by a Gaussian random variable with mean H and a full width at half maximum FWHM parameter. The FWHM parameter defines the distance between the two points at which the Gaussian function attains half of its maximum value. It relates to the standard deviation σ of the random variable via

$$\mathrm{FWHM} = 2\sqrt{2\ln 2}\sigma. \tag{2.46}$$

The typical values for the sodium layer are, e.g., $H = 90$ km and the FWHM $= 11.4$ km.

In [2], the photon noise is modeled by a two-dimensional Gaussian random variable where the x- and y-components are correlated according to the geometric position of the subaperture with respect to the vertical density profile in the sodium layer.

Let the midpoints of the subapertures Ω_{ij} be given by the set of points $\{(\bar{x}_i, \bar{x}_j) : 0 \leq$

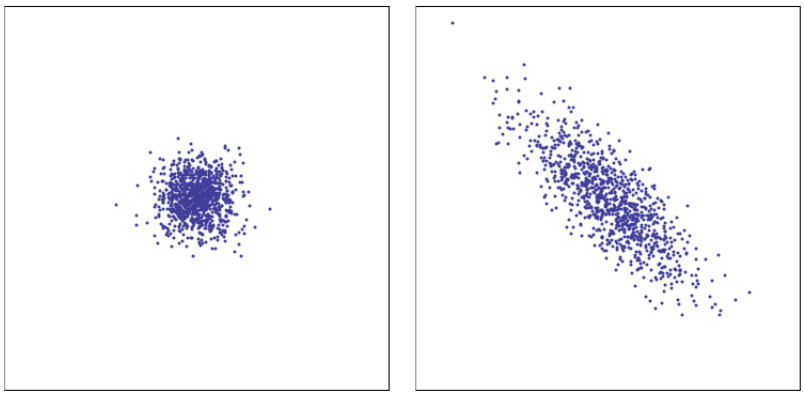

Figure 2.9: Graphic representation of the photon distribution from an NGS (left) and the spot elongation generated by an LGS (right) on the detector.

$i, j < n_s\}$, where

$$\bar{x}_i = \frac{x_i + x_{i+1}}{2}, \tag{2.47}$$

for $0 \leq i < n_s$ and x_i are given by (2.31).

Further, let $(x_1^{\mathrm{LL}}, x_2^{\mathrm{LL}}) \in \mathbb{R}^2$ denote the *laser launch position*, i.e., the position in the pupil plane of the telescope, from which the laser is shot into the atmosphere.

Then the elongation vector in subaperture Ω_{ij} is given by

$$\beta_{ij} = (\beta_{ij,1}, \beta_{ij,2}) = \frac{\mathrm{FWHM}}{H^2} \left((\bar{x}_i, \bar{x}_j) - (x_1^{\mathrm{LL}}, x_2^{\mathrm{LL}}) \right). \tag{2.48}$$

The spot elongated noise covariance in subaperture Ω_{ij} is given by

$$C_{ij} = \sigma^2 \left(I + \frac{1}{f^2} \begin{pmatrix} \beta_{ij,1}^2 & \beta_{ij,1}\beta_{ij,2} \\ \beta_{ij,1}\beta_{ij,2} & \beta_{ij,2}^2 \end{pmatrix} \right), \tag{2.49}$$

where σ^2 is as in (2.44), I is the identity matrix, $(\beta_{ij,1}, \beta_{ij,2})$ is the elongation vector (2.48) and f is the full width at half maximum of the non-elongated spot.

To cope with other sources of noise, such as the read out noise, that are not included in this model, a scalar parameter α_η is introduced,

$$C_{ij} = \sigma^2 \left(I + \frac{\alpha_\eta^2}{f^2} \begin{pmatrix} \beta_{ij,1}^2 & \beta_{ij,1}\beta_{ij,2} \\ \beta_{ij,1}\beta_{ij,2} & \beta_{ij,2}^2 \end{pmatrix} \right). \tag{2.50}$$

By changing the value of α_η, it is possible to fine-tune the noise model (2.49). Observe that for $\alpha_\eta = 0$ the model reduces to the NGS noise model (2.43), and for $\alpha_\eta = 1$, to the full LGS model (2.49).

Thus, the noise model for a SH-WFS associated to an LGS is given by a Gaussian random variable with zero mean and a block-diagonal covariance matrix

$$C_\eta = \mathrm{diag}(C_{ij}), \tag{2.51}$$

with $0 \leq i, j < n_s$ associated to active subapertures Ω_{ij}.

In the reconstruction algorithm, the inverse of the noise covariance matrix is of importance. The inverse is given as a block diagonal matrix

$$C_\eta^{-1} = \mathrm{diag}(C_{ij}^{-1}), \tag{2.52}$$

for $0 \leq i, j < n_s$ associated to active subapertures Ω_{ij} of the inverted two-by-two blocks,

$$C_{ij}^{-1} = \frac{1}{\sigma^2} \frac{f^2}{f^2 + \alpha_\eta^2(\beta_{ij,1}^2 + \beta_{ij,2}^2)} \left(I + \frac{\alpha_\eta^2}{f^2} \begin{pmatrix} \beta_{ij,2}^2 & -\beta_{ij,1}\beta_{ij,2} \\ -\beta_{ij,1}\beta_{ij,2} & \beta_{ij,1}^2 \end{pmatrix} \right). \tag{2.53}$$

Tip-tilt indetermination

The third complication associated with an LGS is the inability for the WFS to detect the low order tip-tilt aberrations of the wavefront correctly. This is referred to as *tip-tilt indetermination* or *tip-tilt uncertainty*. This LGS deficiency can be understood in the following way.

The backscattered light from an LGS passes through the same turbulence layers as the laser beam projected from the ground onto the sodium layer, see Figure 2.10. This leads to an uncertainty in the position of the light source, which translates to an uncertainty in the absolute position of all Shack-Hartmann spots on the detector [1]. Hence, the low-order aberrations of the LGS are untrustworthy. Nevertheless, the relative motion of the Shack-Hartmann spots within the subapertures of the sensor, and therefore the high-order information, is retained.

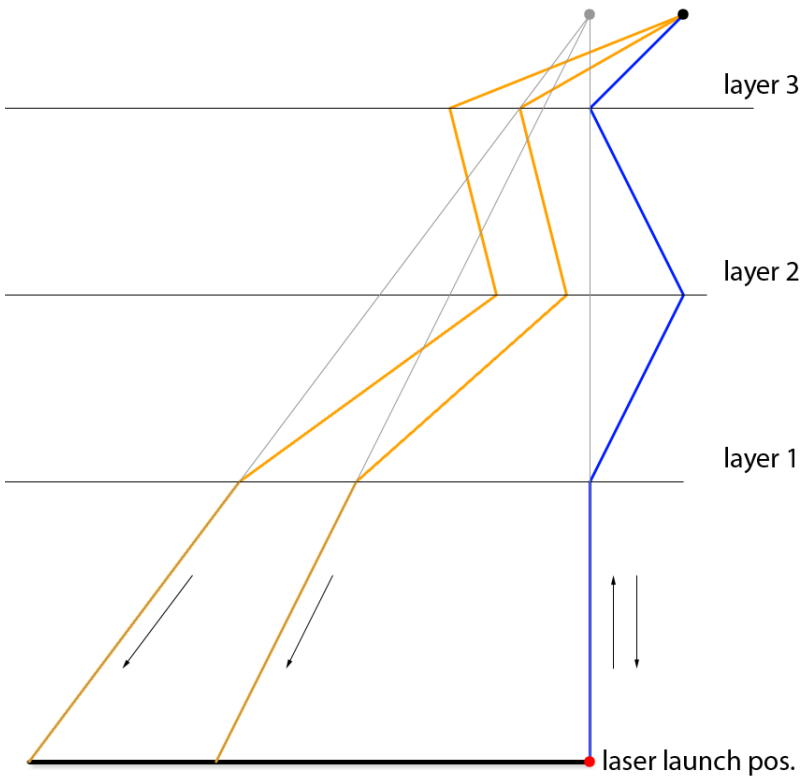

Figure 2.10: Tip-tilt indetermination of the LGS.

For the measurements, tip-tilt uncertainty may be interpreted as an incorrect value of the average slopes over the telescope pupil. Let $s = (s^x, s^y) = (s^x_{ij}, s^y_{ij})_{(i,j)\in\mathcal{M}}$ be the vector of SH-WFS measurements in active subapertures and $\mathcal{M}$ be the mask, see (2.35). The average slope over the telescope pupil $\Omega_{\mathcal{M}}$, see (2.36), is related to the measurements via

$$\begin{aligned}\bar{s}^x &:= \frac{1}{|\mathcal{M}|}\sum_{(i,j)\in\mathcal{M}} s^x_{ij} = \frac{1}{d^2|\mathcal{M}|}\sum_{(i,j)\in\mathcal{M}}\int_{\Omega_{ij}}\frac{\partial\varphi}{\partial x}(x,y)\mathrm{d}(x,y) \qquad (2.54)\\ &= \frac{1}{|\Omega_{\mathcal{M}}|}\int_{\Omega_{\mathcal{M}}}\frac{\partial\varphi}{\partial x}(x,y)\mathrm{d}(x,y),\\ \bar{s}^y &:= \frac{1}{|\mathcal{M}|}\sum_{(i,j)\in\mathcal{M}} s^y_{ij} = \frac{1}{|\Omega_{\mathcal{M}}|}\int_{\Omega_{\mathcal{M}}}\frac{\partial\varphi}{\partial y}(x,y)\mathrm{d}(x,y), \qquad (2.55)\end{aligned}$$

where $\bar{s}^x$ and $\bar{s}^y$ approximate the tip and the tilt, respectively.

For a good wavefront correction it is vital to determine the tip-tilt aberrations. Therefore, typically an LGS is coupled with an NGS, which is used to sense the low order tip-tilt components. However, a much fainter star and a lower resolution sensor can be used. In Figure 2.11 we illustrate such a low order *tip-tilt sensor* (TTS), which uses 2×2 subapertures.

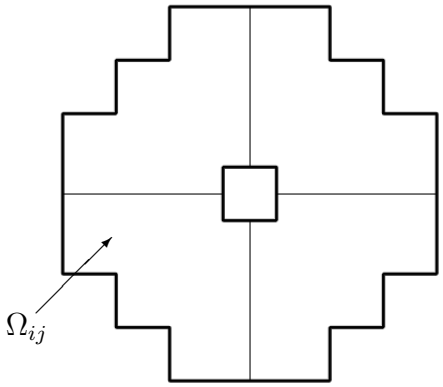

Figure 2.11: Domain of subapertures corresponding to a 2×2 tip-tilt sensor.

There are several ways how an LGS and an NGS can be combined in the wavefront reconstruction. The two sensors can be associated to two mirrors, one that corrects for the tip-tilt aberrations and another that corrects for the higher order modes, as in Figure 2.5. Alternatively, the wavefront reconstruction can be performed by coupling the two problems related to the NGS and the LGS sensors. In this work we use the coupled approach.

2.7 Delay and control

There is an inherent delay in the AO system between the time that the measurements are obtained and the time the DM correction is applied. Due to the rapid evolution of the

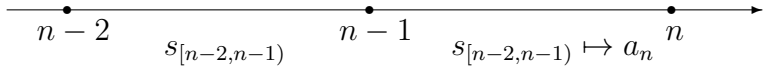

Figure 2.12: Time model of the AO system with a two-step delay. Measurements are obtained in the interval $[n-2, n-1)$. Reconstruction takes place in the interval $[n-1, n)$. Deformable mirror shapes are set at time steps $n-2, n-1$ and n.

atmosphere, a control scheme must be utilized to predict the shape of the DM update, based on the collected measurements and previous DM shape(s).

In the following we present an AO system that exhibits a two-step delay between the time when the measurements are obtained and the time when the mirror updates are applied. The model is illustrated in Figure 2.12. Here, $n \geq 2$ denotes units of time and the intervals $[n-2, n-1)$ and $[n-1, n)$ are time frames. In the model, the DM shapes a_{n-2}, a_{n-1} and a_n are applied at time steps $n-2$, $n-1$ and n, respectively.

Let us denote the average wavefront aberrations over the time frame $[n-2, n-1)$ by $\varphi_{[n-2,n-1)}$. The interval $[n-2, n-1)$ is allocated for obtaining the slope measurements $s_{[n-2,n-1)}$. During this time, the WFS collects photons corresponding to $\varphi_{[n-2,n-1)}$.

The following frame $[n-1, n)$ is allocated for the numerical algorithm that determines the mirror shapes based on the collected measurements. At time step n, the mirrors are updated after the calculation is completed. The new mirror shapes a_n are determined from the reconstruction based on the measurements $s_{[n-2,n-1)}$ and from the previous mirror shapes.

There are two regimes of operations of the AO system: closed loop and open loop. In an *open loop* regime, the measurements are obtained directly from the wavefronts,

$$s_{[n-2,n-1)} = \Gamma \varphi_{[n-2,n-1)} + \eta, \tag{2.56}$$

where Γ is the Shack-Hartmann operator, see (2.37), and η denotes the measurement noise.

A more stable regime of operation of the AO system is the *closed loop*. Here, the deformable mirrors correct the wavefront before the measurements are obtained,

$$s_{[n-2,n-1)} = \Gamma(\varphi_{[n-2,n-1)} - a_{n-2}) + \eta. \tag{2.57}$$

In this case, the measurements $s_{[n-2,n-1)}$ correspond to the residuals of the corrected wavefront post DM correction.

2.8 AO Systems

In this work we handle four AO systems: single conjugate adaptive optics (SCAO), laser tomography adaptive optics (LTAO), multi object adaptive optics (MOAO) and multi conjugate adaptive optics (MCAO). We briefly describe each system below.

2.8.1 SCAO

A classical adaptive optics system, also known as *single conjugate adaptive optics* (SCAO), is used when the object of interest, which could be a star or a galaxy, is located near a

bright NGS. In a SCAO system, a single WFS and a single DM are used for wavefront correction.

To operate SCAO, the wavefront is reconstructed from the measurement data and the mirror shape is chosen according to the shape of the reconstructed wavefront. The SCAO system can be operated in open and closed loops; in this work we concentrate on a closed loop SCAO system.

The mirror correction is good for those directions that are close to the direction of the NGS, see the wavefront propagation equation (2.41). The further away the object of interest is from the NGS, the worse is the AO correction.

2.8.2 LTAO

In the absence of a bright star in the vicinity of the object of interest, a laser guide star can be used. The LGS is combined with at least one NGS to correct for the low order modes, which are not available using the LGS alone, see Section 2.6.2. In the general setting, a combination of several LGSs and NGSs is possible.

A *laser tomography adaptive optics* (LTAO) system is an AO system that utilizes G_{LGS} laser guide stars and G_{NGS} natural guide stars in combination, to provide a correction using a single mirror towards a direction within the *field of view* (FoV), the observable region of the instrument.

The correction is performed in two steps. In the first step, L turbulence layers are reconstructed from measurements of $G = G_{\text{LGS}} + G_{\text{NGS}}$ wavefront sensors. This step is known as *atmospheric tomography*, see Figure 1.1. In the second step, the DM is chosen by the shape of the wavefront projected through the reconstructed layers in the direction of interest.

There are crucial differences in performing atmospheric tomography, as opposed to wavefront reconstruction. Apart from solving a more complex system with larger dimensions, the intrinsic properties of the underlying problem make it unstable, in terms of the dependence of the solution on the data, and therefore harder to handle. The focus of the thesis is to provide an efficient, stable and fast solution method for atmospheric tomography.

The LTAO system can be operated in open and closed loops; in this work we concentrate on a closed loop system. An LTAO system is illustrated in Figure 2.13.

2.8.3 MOAO

Based on the same principles as LTAO, by combining several guide stars and atmospheric tomography it is possible to correct for multiple directions of interest simultaneously. This can be achieved by using several mirrors. The mirrors can be arranged in parallel or in sequence.

In *multi object adaptive optics* (MOAO), a combination of LGSs and NGSs is used to reconstruct the turbulence layers in the atmospheric tomography step, as for LTAO. However, instead of using a single mirror, multiple DMs are used in parallel to correct for several directions of interest within the field of view.

The MOAO system that we consider operates in open loop. Thus, the measurements are obtained independently from the mirror corrections.

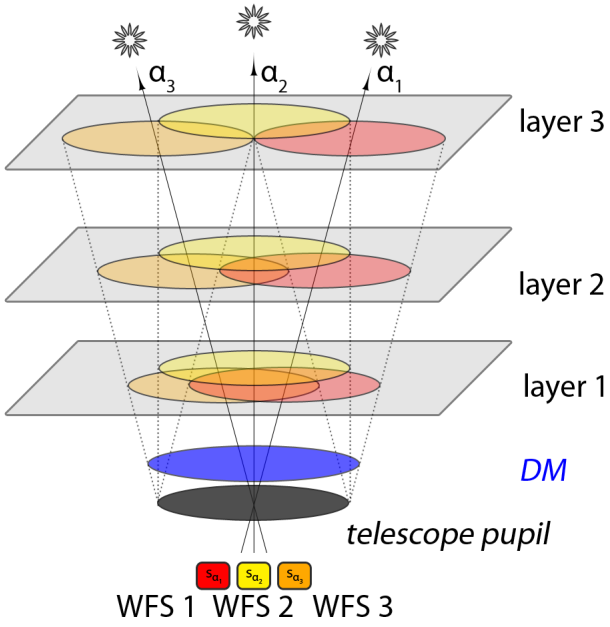

Figure 2.13: An LTAO System. Several guide stars are used as reference points to obtain the WFS measurements. The DM correction is determined via the cumulative wavefront through the reconstructed atmosphere towards the direction of interest.

2.8.4 MCAO

Another possibility to correct for multiple directions simultaneously is to use several mirrors in sequence. A *multi conjugate adaptive optics* (MCAO) system utilizes multiple DMs that are conjugated to different altitudes. The goal of MCAO is to provide a uniform correction within the field of view. We illustrate an MCAO system in Figure 2.14.

The MCAO reconstruction is split into two steps. In the first step, L turbulence layers are computed from WFS measurements in G guide star directions by solving the atmospheric tomography problem. In the second step, the shapes of M mirrors conjugated to different altitudes are fitted to the reconstructed layers to optimize the quality over the FoV in a *fitting step*.

The standard approach for mirror fitting, see e.g., [15], is to minimize the cost functional

$$\int_{\mathrm{FoV}} \int_{\Omega_{\mathcal{M}}} \|(P_\theta^{\mathrm{NGS}}\boldsymbol{\phi})(x) - (\bar{P}_\theta^{\mathrm{NGS}}\mathbf{a})(x)\|^2 \mathrm{d}x\mathrm{d}\theta, \tag{2.58}$$

with respect to a vector of mirror shapes $\mathbf{a} = (a_1, ..., a_M)$. Here, $\boldsymbol{\phi} = (\phi_1, ..., \phi_L)$ is a vector of reconstructed layers, P_θ^{NGS} and $\bar{P}_\theta^{\mathrm{NGS}}$ are projection operators (2.41) through the reconstructed layers and the mirrors, respectively, $\Omega_{\mathcal{M}} \subseteq \mathbb{R}^2$ is the pupil of the telescope and $\mathrm{FoV} \subseteq \mathbb{R}^2 \times \{1\}$ is a set of directions in the field of view.

Due to the sequential positioning of the mirrors, it is possible to operate the MCAO system in closed loop as well as in open loop. In this work we concentrate on a closed loop MCAO system, where the WFS determines the slopes of the wavefront after each mirror correction has been applied.

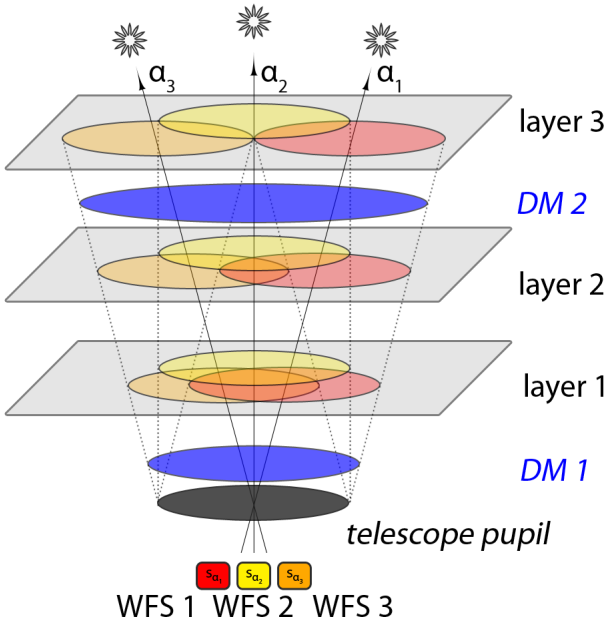

Figure 2.14: An MCAO system. The mirrors are conjugated to several altitudes to optimize the correction within the field of view.

Chapter 3

Mathematical preliminaries

3.1 Inverse problems

In the following sections we discuss the mathematical preliminaries required to introduce and solve the problem. In this section we present a general introduction to inverse problems. The classical regularization theory is briefly discussed in Section 3.2. Both of these sections follow closely [17]. In Section 3.3 we go into more detail on the Bayesian approach to inverse problems; our problem is formulated in the Bayesian framework. This section is based on [36]. Next, in Section 3.4 we discuss wavelets, which we use for discretization. More information on wavelets can be found in [10, 42]. We then recap some algorithms and techniques for solving a discretized linear system of equations in Section 3.5. Finally, in Section 3.6 we discuss how an iterative method for solving linear systems of equations can be accelerated using a technique called preconditioning. The discussion of the iterative methods in the last two sections is to a large extent based on [62].

Let $\mathcal{X}$ and $\mathcal{Y}$ be two Hilbert spaces and $T \in L(\mathcal{X}, \mathcal{Y})$ be a bounded linear operator between them. The problem of finding an $x \in \mathcal{X}$ for a given $y \in \mathcal{Y}$ that satisfies the equation

$$Tx = y \tag{3.1}$$

is called *well-posed*, according to the definition by Hadamard [29], if all of the following three conditions are satisfied:

$$\mathcal{R}(T) = \mathcal{Y} \quad \text{for all data } y, \text{ there exists a solution } x, \tag{3.2}$$

$$\mathcal{N}(T) = \{0\} \quad \text{for all data } y, \text{ the solution } x \text{ is unique}, \tag{3.3}$$

$$T^{-1} \in L(\mathcal{Y}, \mathcal{X}) \quad \text{every solution } x \text{ depends continuously on the data } y. \tag{3.4}$$

If the operator T does not satisfy at least one of the three conditions above, then the problem (3.1) is called an *ill-posed problem.*

The violation of the first condition (3.2) means that for some data $y \in \mathcal{Y}$ no solution $x \in \mathcal{X}$ exists. In this case, it is typical to seek an x, such that Tx lies closest to y in $\mathcal{Y}$. Such a solution x is called a *least-squares* solution of the problem (3.1).

The violation of the second condition (3.3) means that if $x \in \mathcal{X}$ is a solution for $y \in \mathcal{Y}$, then $x + \mathcal{N}(T)$ is a solution set of (3.1), i.e., there are infinite number of solutions.

Uniqueness of the solution can be enforced by selecting an x with the smallest norm in $\mathcal{X}$. A least-squares solution x of minimal norm is called the *best-approximate* solution.

These two concepts are summarized in the following definition.

Definition 3.1 ([17, Definition 2.1])**.** *Let $T \in L(\mathcal{X}, \mathcal{Y})$. An $x \in \mathcal{X}$ is called a* least-squares *solution of $Tx = y$ if*

$$\|Tx - y\| = \inf_{z \in \mathcal{X}} \|Tz - y\|. \tag{3.5}$$

Further, $x \in \mathcal{X}$ is called a best-approximate *solution of $Tx = y$ if x is a least-squares solution of $Tx = y$ and*

$$\|x\| = \inf\{\|z\| : z \text{ is least-squares solution of } Tx = y\}. \tag{3.6}$$

The operator that maps $y \in \mathcal{Y}$ to the best-approximate solution is called the *Moore-Penrose (generalized) inverse*. It is defined by a restriction of the operator T to make it invertible, combined with a linear extension on the domain.

Definition 3.2 ([17, Definition 2.2])**.** *For $T \in L(\mathcal{X}, \mathcal{Y})$, let $\widetilde{T}$ be introduced as its restriction,*

$$\widetilde{T} := T|_{\mathcal{N}(T)^\perp} : \mathcal{N}(T)^\perp \to \mathcal{R}(T). \tag{3.7}$$

Then, the Moore-Penrose (generalized) inverse *$T^\dagger$ of T is defined as the unique linear extension of $\widetilde{T}^{-1}$ to*

$$\mathcal{D}(T^\dagger) := \mathcal{R}(T) + \mathcal{R}(T)^\perp \tag{3.8}$$

with

$$\mathcal{N}(T^\dagger) = \mathcal{R}(T)^\perp. \tag{3.9}$$

Observe that $T^\dagger$ is well defined. The inverse $\widetilde{T}^{-1}$ exists, as $\mathcal{N}(\widetilde{T}) = \{0\}$ and $\mathcal{R}(\widetilde{T}) = \mathcal{R}(T)$. For any $y \in \mathcal{D}(T^\dagger)$, there is a unique representation $y = y_1 + y_2$, such that $y_1 \in \mathcal{R}(T)$ and $y_2 \in \mathcal{R}(T)^\perp$. Due to the requirement that $T^\dagger$ is linear and (3.9), we have $T^\dagger y = \widetilde{T}^{-1} y_1$.

The Moore-Penrose inverse has the following properties.

Proposition 3.3 ([17, Proposition 2.4, Theorem 2.5, Theorem 2.6])**.**

1. *$T^\dagger$ is bounded (continuous) if and only if $\mathcal{R}(T)$ is closed.*
2. *Let $y \in \mathcal{D}(T^\dagger)$. Then $Tx = y$ has a unique best-approximate solution which is given by $x^\dagger := T^\dagger y$. The set of all least-squares solutions is $x^\dagger + \mathcal{N}(T)$.*
3. *Let $y \in \mathcal{D}(T^\dagger)$. Then $x \in \mathcal{X}$ is a least-squares solution of $Tx = y$ if and only if the normal equation*

$$T^*Tx = T^*y \tag{3.10}$$

holds, where $T^* : \mathcal{Y} \to \mathcal{X}$ is the adjoint of T.

The first property relates the generalized inverse to the third condition of Hadamard (3.4) and is discussed in the following remark. The second property relates the concepts of the least-squares and the best-approximate solution to the generalized inverse. The third property states that a Moore-Penrose inverse can be computed by finding a solution to the normal equation (3.10) of minimal norm.

Remark 3.4. It may be concluded from (3.8) and Proposition 3.3.1 that if $\mathcal{R}(T)$ is not closed, then $\mathcal{D}(T^\dagger)$ is a proper subset of $\mathcal{Y} = \overline{\mathcal{R}(T)} \oplus \mathcal{R}(T)^\perp$ and $T^\dagger$ is unbounded. If this is the case, there exists $y \in \mathcal{Y}$ such that $y \notin \mathcal{D}(T^\dagger)$ and, consequently, no corresponding solution exists. Moreover, for $y \in \mathcal{D}(T^\dagger)$, the solution $x^\dagger$ will not depend continuously on y. Thus, there is no guarantee that for a slight perturbation $y^\delta \in \mathcal{D}(T^\dagger)$ of y, $T^\dagger y^\delta$ is in the neighborhood of $T^\dagger y$ in $\mathcal{X}$.

Compact linear operators can be additionally characterized via their singular systems. Let T be a compact linear operator. The self-adjoint operator $T^*T : \mathcal{X} \to \mathcal{X}$ has an *eigensystem* (σ_n^2, v_n), with non-zero *eigenvalues* $\sigma_n^2 \in \mathbb{R}$ and *eigenvectors* v_n with the property

$$T^*Tv_n = \sigma_n^2 v_n. \tag{3.11}$$

The eigenvectors $\{v_n\}$ form an orthonormal basis of $\overline{\mathcal{R}(T^*T)} = \overline{\mathcal{R}(T^*)} \subseteq \mathcal{X}$, see, e.g., [16]. All eigenvalues of the operator T^*T are positive, $\sigma_n^2 = \|Tv_n\|^2 > 0$, and we write them down in decreasing order with multiplicity.

The eigensystem allows the operator to be diagonalized as follows,

$$T^*Tx = \sum_{n=0}^{\infty} \sigma_n^2 \langle x, v_n \rangle v_n, \tag{3.12}$$

for all $x \in \mathcal{X}$.

If the operator T is not self-adjoint, no eigensystem needs to exist. However, as a substitute, a *singular system* (σ_n, v_n, u_n) of the compact linear operator T can be defined as follows. For the eigensystem (σ_n^2, v_n) of T^*T, let us define vectors

$$u_n := \frac{1}{\sigma_n} T v_n. \tag{3.13}$$

One can show [16] that the vectors $\{u_n\}$ form an orthonormal basis of $\overline{\mathcal{R}(TT^*)} = \overline{\mathcal{R}(T)} \subseteq \mathcal{Y}$ and that the following relations hold,

$$Tv_n = \sigma_n u_n, \tag{3.14}$$

$$T^*u_n = \sigma_n v_n, \tag{3.15}$$

$$Tx = \sum_{n=0}^{\infty} \sigma_n \langle x, v_n \rangle u_n, \tag{3.16}$$

$$T^*y = \sum_{n=0}^{\infty} \sigma_n \langle y, u_n \rangle v_n, \tag{3.17}$$

for all $x \in \mathcal{X}$ and $y \in \mathcal{Y}$. The values $\sigma_n > 0$ are called the *singular values* of T and are written down in a decreasing order with multiplicity. The vectors u_n, v_n are called the *singular vectors* of T and equations (3.16), (3.17) are called the *singular value expansion* of the operator.

If there are infinitely many singular values of a compact operator, the only accumulation point is zero, i.e.,

$$\lim_{n \to \infty} \sigma_n = 0, \tag{3.18}$$

see [17, 16].

For compact operators, unboundedness of the generalized inverse is related to the dimension of its range. The generalized inverse of a compact operator is unbounded if the dimension of its range is infinite [17, Proposition 2.7]. In this case, the problem (3.1) is ill-posed.

Moreover, solvability of equation (3.1) with a compact operator T is characterized via its singular value decomposition, which is reflected in the following theorem.

Theorem 3.5 ([17, Theorem 2.8])**.** *Let (σ_n, v_n, u_n) be a singular system of the compact operator $T : \mathcal{X} \to \mathcal{Y}$ and $y \in \mathcal{Y}$. Then it holds that*

1. *$y \in \mathcal{D}(T^\dagger)$ if and only if*
$$\sum_{n=0}^{\infty} \frac{|\langle y, u_n \rangle|^2}{\sigma_n^2} < \infty, \tag{3.19}$$
2. *for any $y \in \mathcal{D}(T^\dagger)$,*
$$T^\dagger y = \sum_{n=0}^{\infty} \frac{\langle y, u_n \rangle}{\sigma_n} v_n. \tag{3.20}$$

The first condition of the theorem is known as the *Picard criterion*. The interpretation of it is that the decomposition of a specific $y \in \mathcal{Y}$ with respect to the singular vectors u_n of T must decay faster towards zero than the singular values of the operator, see (3.18), i.e., the sequence $\langle y, u_n \rangle / \sigma_n$ is in ℓ^2, if y is an admissible right-hand side. In that case, the second statement gives the exact representation of the solution in terms of the singular system of T.

3.2 Regularization in the deterministic setting

In many real world applications, the mathematical model that relates an unknown quantity x to a measurement y exhibits the behavior of an ill-posed problem. In this case, the measurement operator T violates one of the three conditions of Hadamard. If the measurement model can be characterized by an operator with a closed range, then the uniqueness and existence of a solution are obtainable via the Moore-Penrose generalized inverse.

For many of the practical problems, such as X-ray tomography, backwards heat equation or signal and image processing, the measurement operator does not have a closed range. Moreover, measurements are usually contaminated with noise, i.e., the exact data y are not available, instead only an approximation, which we denote by $y^\delta \in \mathcal{Y}$, is given.

This situation is outlined in Remark 3.4. For one, the domain of the generalized inverse is a proper subset of $\mathcal{Y}$. Hence, the perturbed y^δ need not belong to the domain of $T^\dagger$ and in that case the generalized inverse cannot be applied. Moreover, even if y^δ is in the domain of $T^\dagger$, the unboundedness of $T^\dagger$ implies that $T^\dagger y^\delta$ is in general a poor approximation of $T^\dagger y$.

In this case, the unstable relation of the noisy measurement y^δ to the best-approximate solution $x^\dagger = T^\dagger y$ of the exact data y can be improved by a process called *regularization*. Regularization introduces a class of stable problems that approximate the original

problem (3.1). The advantage of regularization is a stable relation of the perturbed measurement to the solution.

There are several approaches on how (3.1) can be regularized. In this section, the deterministic approach is discussed. In the subsequent section we present the statistical approach to inverse problems.

In the deterministic setting, the assumption is that instead of the *exact data* y, a noisy approximation y^δ with

$$\|y - y^\delta\| \leq \delta \tag{3.21}$$

is available. Here, y^δ are referred to as the *noisy data* and $\delta > 0$ is the *noise level* that characterizes the discrepancy between the noisy and the exact data.

In the deterministic setting, a *regularization method* consists of a pair (R_α, α), where $\{R_\alpha\}$ is a family of continuous *regularization operators* that are meant to approximate $T^\dagger$ and α is a *regularization parameter*. A regularization method (R_α, α) is referred to as a convergent regularization method if $R_\alpha y^\delta$ tends to $x^\dagger$, when the noise level δ in (3.21) goes to zero and the parameter α is chosen appropriately.

This property is summarized in the following definition.

Definition 3.6 ([17, Definition 3.1])**.** *For Hilbert spaces $\mathcal{X}, \mathcal{Y}$, let $T \in L(\mathcal{X}, \mathcal{Y})$ and $\alpha_0 \in (0, +\infty]$. For every $\alpha \in (0, \alpha_0)$, let*

$$R_\alpha : \mathcal{Y} \to \mathcal{X} \tag{3.22}$$

be a continuous (not necessarily linear) operator. The family $\{R_\alpha\}$ is called a regularization *or a* regularization operator *(for $T^\dagger$), if, for all $y \in \mathcal{D}(T^\dagger)$, there exists a* parameter choice rule $\alpha = \alpha(\delta, y^\delta)$ *such that*

$$\lim_{\delta \to 0} \sup\{\|R_{\alpha(\delta, y^\delta)} y^\delta - T^\dagger y\| : y^\delta \in \mathcal{Y}, \|y - y^\delta\| \leq \delta\} = 0 \tag{3.23}$$

holds. Here,

$$\alpha : \mathbb{R}^+ \times \mathcal{Y} \to (0, \alpha_0) \tag{3.24}$$

is such that

$$\lim_{\delta \to 0} \sup\{\alpha(\delta, y^\delta) : y^\delta \in \mathcal{Y}, \|y - y^\delta\| \leq \delta\} = 0. \tag{3.25}$$

For a specific $y \in \mathcal{D}(T^\dagger)$, a pair (R_α, α) is called a (convergent) regularization method *(for solving $Tx = y$) if (3.23) and (3.25) hold.*

The convergence (3.23) is assured for any collection of noisy data y^δ satisfying the noise level condition (3.21) and thus it is a "worst-case" concept of convergence with respect to the right-hand side.

A prominent example of a regularization operator is Tikhonov regularization. The Tikhonov regularization operator is defined via

$$R_\alpha := (T^*T + \alpha I)^{-1} T^*. \tag{3.26}$$

Thus, the regularized solution $x_\alpha^\delta = R_\alpha y^\delta$ is found as the solution of the linear equation

$$(T^*T + \alpha I)x = T^* y^\delta. \tag{3.27}$$

Tikhonov regularization can also be formulated as a minimization problem of the so-called *Tikhonov functional*,

$$x_\alpha^\delta = \operatorname*{argmin}_{x \in \mathcal{X}} \|Tx - y^\delta\|^2 + \alpha \|x\|^2. \tag{3.28}$$

The functional is strictly convex and tends to infinity as $\|x\|$ goes to infinity for all $\alpha > 0$, so the minimizer is unique.

For a compact operator, Tikhonov regularization has a representation in terms of the singular system (σ_n, v_n, u_n),

$$x_\alpha^\delta = \sum_{n=0}^{\infty} \frac{\sigma_n}{\sigma_n^2 + \alpha} \langle y^\delta, u_n \rangle v_n. \tag{3.29}$$

Observe that whereas (3.20) can be unbounded as σ_n decay to zero, see (3.18), this singularity is removed in (3.29) for any positive α.

The role of the regularization parameter α is especially apparent, when the regularization operator is stated in the form of the functional (3.28). The parameter α determines the tradeoff between stability and accuracy of the regularized solution. By choosing a small α, the weight of the minimizer is concentrated on the fitting term $\|Tx - y^\delta\|^2$. In this case, the problem is more accurate, but less stable, as the original problem is unstable. By choosing a large α, the stability of the solution is enforced, through the minimization of the regularization term $\|x\|^2$. However, as α increases, the regularized problem deviates further from the original problem. Finding the right choice of α to balance accuracy and stability of the solution is one of the key tasks in inverse problems.

A condition on the choice of α for Tikhonov regularization to be a convergent regularization method is stated in the following theorem.

Theorem 3.7 ([17, Theorem 5.2])**.** *Let x_α^δ be defined by (3.28), $y \in \mathcal{R}(T)$, $\|y - y^\delta\| \leq \delta$. If $\alpha = \alpha(\delta)$ is such that*

$$\lim_{\delta \to 0} \alpha(\delta) = 0 \quad \text{and} \quad \lim_{\delta \to 0} \frac{\delta^2}{\alpha(\delta)} = 0, \tag{3.30}$$

then

$$\lim_{\delta \to 0} x_{\alpha(\delta)}^\delta = T^\dagger y. \tag{3.31}$$

The theorem states that if α, chosen according to the noise level δ, tends at a reasonable rate towards zero as the noise level decreases, the minimizer of the Tikhonov functional tends towards the exact solution.

The parameter choice rules are split into a-priori and a-posteriori rules, where the former use only the knowledge of the noise level δ and the latter rely in addition on the noisy data y^δ. We refer to [17] and the references therein for more information on parameter choice rules.

Another very important aspect of a regularization method is the choice of the solution space $\mathcal{X}$. Different properties of the solution can be attained by selecting a different $\mathcal{X}$. For instance, a Sobolev space $\mathcal{X} = H^s$ with $s > 0$ enforces smoothness of the solution.

Regularization can also be generalized to Banach spaces. For instance, the space $\mathcal{X} = L^p$ with $1 \leq p < 2$ promotes sparsity of the solution. In this regard, the regularized

problem is typically formulated via the Tikhonov functional (3.28), which allows a more flexible way of stating the problem. In this setting the cost functional is expressed with a modified penalty term,

$$x_\alpha^\delta = \operatorname*{argmin}_{x \in L^p} \|Tx - y^\delta\|^2 + \alpha \|x\|_{L^p}^p. \tag{3.32}$$

The standard theory discussed in this section can no longer be applied directly. For more information on regularization methods in Banach spaces we refer to [63] and the references therein.

3.3 Bayesian inference

In the statistical approach to inverse problems, equation (3.1) is formulated in a probabilistic setting. Instead of computing a specific solution, the solution to the problem in the Bayesian approach appears in the form of a probability distribution. To derive this posterior distribution, the approach combines the prior information about the solution, the measurement noise and the observed measurement. Consequently, it is possible to query the posterior distribution for a specific solution, like in the deterministic case, or to obtain other information about the solution, such as confidence intervals where the solution is located with a given probability.

Unlike the deterministic approach, in which the regularization method is represented as a family of well-posed problems, the statistical approach regularizes the problem inherently by taking the uncertainty of the solution and the measurement into its formulation.

Bayesian inverse problems in literature are often formulated as finite dimensional problems. The main reason is the availability of density functions and the straightforward formalism related to them. From the asymptotical point of view the theory of Bayesian methods in the infinite dimensions is not fully established. Below we assume to have a discretized formulation of the inverse problem, i.e., the quantities are finite dimensional.

3.3.1 Probability distributions

We begin by introducing the notation of probability distributions. Let $(\Omega, \mathcal{A}, P)$ be a *probability space*, where Ω is a *sample space*, $\mathcal{A}$ is the *set of events*, i.e., a σ-algebra over a space Ω and P is a *probability measure*. Two events $A, B \in \mathcal{A}$ are independent if $P(A \cap B) = P(A)P(B)$.

Let us denote by $\mathcal{B}$ the Borel σ-algebra over $\mathbb{R}^n$, which is the smallest σ-algebra containing the open sets of $\mathbb{R}^n$. The Borel σ-algebra is equipped with the Lebesgue measure m that assigns a volume to a set, $m(B) = \int_B \mathrm{d}x$ for $B \in \mathcal{B}$.

Let X be a *random variable* with values in $\mathbb{R}^n$, i.e., X is a measurable mapping

$$X : \Omega \to \mathbb{R}^n, \tag{3.33}$$

such that $X^{-1}(B) \in \mathcal{A}$ for every open set $B \subseteq \mathbb{R}^n$. In our notation we use capital letters to denote the random variables and lower-case letters to denote their realizations, e.g., $X(\omega) = x$. Moreover, we use a shorthand notation $X \in \mathbb{R}^n$ when referring to the mapping $X : \Omega \to \mathbb{R}^n$.

A random variable induces a probability measure on the Borel σ-algebra via

$$\mu_X(B) := P(\{X \in B\}) = P(\{\omega \in \Omega : X(\omega) \in B\}), \tag{3.34}$$

for $B \in \mathcal{B}$, called the *probability distribution* of X.

The random variable X is called *absolutely continuous* if its probability distribution is absolutely continuous with respect to the Lebesgue measure over $\mathbb{R}^n$, i.e., if $m(B) = 0$ implies $\mu_X(B) = 0$ for all $B \in \mathcal{B}$.

According to the Radon-Nikodym theorem, an absolutely continuous random variable X defines a *probability density* π_X via

$$P(\{X \in B\}) = \mu_X(B) = \int_B \pi_X(x)\mathrm{d}x \tag{3.35}$$

for $B \in \mathcal{B}$. The function π_X is also called the Radon-Nikodym derivative of μ_X with respect to m.

Now let $X \in \mathbb{R}^n, Y \in \mathbb{R}^m$ be two random variables over the same probability space. Their *joint probability distribution* is defined as

$$\mu_{XY}(A, B) = P(\{X \in A\} \cap \{Y \in B\}), \tag{3.36}$$

for any two open sets $A \in \mathbb{R}^n$, $B \in \mathbb{R}^m$. Observe, the two random variables are independent if

$$\mu_{XY}(A, B) = \mu_X(A)\mu_Y(B), \tag{3.37}$$

for all A, B.

Now assume that X and Y are absolutely continuous. Then as above, their *joint probability density* $\pi_{XY}(x, y)$ is given by

$$\mu_{XY}(A, B) = \int_{A\times B} \pi_{XY}(x, y)\mathrm{d}x\mathrm{d}y. \tag{3.38}$$

Moreover, it holds that

$$\mu_X(A) = \mu_{XY}(A, \mathbb{R}^m) = \int_{A\times\mathbb{R}^m} \pi_{XY}(x, y)\mathrm{d}x\mathrm{d}y = \int_A \pi_X(x)\mathrm{d}x, \tag{3.39}$$

where the *marginal probability density* is given by

$$\pi_X(x) = \int_{\mathbb{R}^m} \pi_{XY}(x, y)dy. \tag{3.40}$$

Finally, the *conditional measure of A conditioned on B* is defined by

$$\mu_X(A \mid B) = \frac{\mu_{XY}(A, B)}{\mu_Y(B)}, \tag{3.41}$$

for $\mu_Y(B) > 0$ and the *conditional probability density* by

$$\pi_X(x \mid y) = \frac{\pi_{XY}(x, y)}{\pi_Y(y)} \tag{3.42}$$

for $\pi_Y(y) > 0$.

The *Bayes formula* states that the following relation holds,

$$\pi_{XY}(x, y) = \pi_X(x \mid y)\pi_Y(y) = \pi_Y(y \mid x)\pi_X(x). \tag{3.43}$$

The Bayes formula allows to solve the direct and the inverse problems to (3.1) in Bayesian framework.

3.3.2 The Bayesian framework

Using the notation above it is possible to address the inverse problem (3.1) in the Bayesian setting. In the Bayesian approach to inverse problems, the problem (3.1) is formulated in random variables. We assume that noise is additive. Then, the problem (3.1) can be formulated in terms of random variables as

$$Y = TX + E, \tag{3.44}$$

where the random variable $X \in \mathbb{R}^n$ is the *unknown* and $Y, E \in \mathbb{R}^m$ are the random variables that correspond to the *measurement* and the *noise*, respectively. We assume that these random variables are absolutely continuous. Furthermore, we assume that the noise is independent from the unknown. The measurement is assumed to be a realization of Y. In the following we denote the realization by y^δ.

Any prior information on the solution is represented as a *prior* probability density $\pi_{\text{pr}}(x)$. In the deterministic case this relates to the choice of the solution space $\mathcal{X}$ and the knowledge about the source conditions, i.e., prior information about the solution, which is available independently of the observed measurement or noise.

If the joint probability density $\pi_{XY}(x, y)$ of X and Y is known, then the marginal probability density of X is

$$\pi_{\text{pr}}(x) = \int_{\mathbb{R}^m} \pi_{XY}(x, y)\mathrm{d}y. \tag{3.45}$$

The direct problem is described via the conditional probability density of Y,

$$\pi_Y(y^\delta \mid x) = \frac{\pi_{XY}(x, y^\delta)}{\pi_{\text{pr}}(x)} \tag{3.46}$$

for $\pi_{\text{pr}}(x) > 0$. This is known as the *likelihood function*. It describes the likelihood of a measurement y^δ if a certain $X = x$ is observed.

The inverse problem is characterized by the conditional probability density of X,

$$\pi_X(x \mid y^\delta) = \frac{\pi_{XY}(x, y^\delta)}{\pi_Y(y^\delta)}, \tag{3.47}$$

for $\pi_Y(y^\delta) > 0$. This is also called the *posterior* distribution of X and we often omit y^δ by denoting $\pi_{\text{post}}(x) = \pi_X(x \mid y^\delta)$. The posterior distributions describes the behavior of the solution, under the condition that a certain measurement y^δ has taken place. In the Bayesian approach to inverse problems, the solution of the inverse problem is the posterior distribution $\pi_{\text{post}}(x)$.

The posterior distribution of X can be expressed using the prior distribution and the likelihood function by applying the Bayes formula, which is formulated in the following theorem.

Theorem 3.8 ([36, Theorem 3.1])**.** *Let $X \in \mathbb{R}^n$ be a random variable with a known prior probability density $\pi_{\text{pr}}(x)$ and y^δ be a realization of the random variable $Y \in \mathbb{R}^m$, such that $\pi_Y(y^\delta) > 0$. Then the posterior probability distribution of X for the observed data y^δ is given by*

$$\pi_{\text{post}}(x) = \pi_X(x \mid y^\delta) = \frac{\pi_{\text{pr}}(x)\pi_Y(y^\delta \mid x)}{\pi_Y(y^\delta)}. \tag{3.48}$$

Using the posterior distribution it is possible to calculate *point estimates*. An example of a point estimate is the *maximum a-posteriori* (MAP) estimate. The MAP estimate answers the question: "if a measurement y^δ has been observed, what is the most probable occurrence of the unknown X?" It is defined via the optimization problem

$$x_{\text{MAP}} = \operatorname*{argmax}_{x \in \mathbb{R}^n} \pi_X(x \mid y^\delta). \tag{3.49}$$

Another point estimate is the *conditional mean* (CM). The CM estimate determines the expected value of X for an observed instance of a measurement y^δ,

$$x_{\text{CM}} = \mathbb{E}(x \mid y^\delta) = \int_{\mathbb{R}^n} x\pi_X(x \mid y^\delta)\mathrm{d}x. \tag{3.50}$$

Computing the CM estimate is an integration problem.

Apart from point estimates, the posterior distribution allows one to calculate *interval estimates*. For instance, a *Bayesian credibility set* answers the question "in what interval are the values of the unknown with a given probability p and observed data y^δ?" This problem is formulated as

$$\int_D \pi_X(x \mid y^\delta)\mathrm{d}x = p, \qquad \pi_X(x \mid y^\delta) = c \text{ for all } x \in \partial D, \tag{3.51}$$

for the unknown set D, given probability $p \in (0, 1)$, c is some constant value and ∂D is the boundary of D. In this work, we will only concentrate on the MAP estimate.

3.3.3 Additive noise and Gaussian densities

So far, we have not used the assumptions on the additive noise. Using the relation (3.44) and the assumption that E and X are mutually independent, the likelihood function can be reformulated in terms of the noise probability density function,

$$\begin{aligned}\int_B \pi_Y(y \mid x)\mathrm{d}y = \mu_Y(B \mid x) &= \frac{\mu_{XY}(x, B)}{\mu_X(x)} = \frac{P(\{X = x\} \cap \{Y \in B\})}{P(\{X = x\})} \\ &= \frac{P(\{X = x\} \cap \{E \in B - Tx\})}{P(\{X = x\})} = \mu_E(B - Tx) \\ &= \int_B \pi_E(y - Tx)\mathrm{d}y,\end{aligned} \tag{3.52}$$

for all open sets $B \subseteq \mathbb{R}^m$ and a fixed $x \in \mathbb{R}^n$. The relation between the probability density functions is thus,

$$\pi_Y(y \mid x) = \pi_E(y - Tx). \tag{3.53}$$

Therefore, according to the Bayes theorem, the posterior distribution with additive noise is given by

$$\pi_{\text{post}}(x) = \pi_X(x \mid y^\delta) = \frac{\pi_{\text{pr}}(x)\pi_E(y^\delta - Tx)}{\pi_Y(y^\delta)}. \tag{3.54}$$

Observe that to compute the MAP estimate, it is enough to know the prior and the measurement noise probability density functions, since $\pi_Y(y^\delta)$ is a scaling factor and does not influence the optimization problem,

$$x_{\text{MAP}} = \operatorname*{argmax}_{x \in \mathbb{R}^n} \pi_X(x \mid y^\delta) = \operatorname*{argmax}_{x \in \mathbb{R}^n} \pi_{\text{pr}}(x)\pi_E(y^\delta - Tx). \tag{3.55}$$

For a Gaussian random variable $X \in \mathbb{R}^n$ with mean x_0 and a symmetric positive definite covariance matrix C_X, the probability density function is of an explicit form,

$$\pi_X(x) = \frac{1}{(2\pi)^{n/2}(\det C_X)^{1/2}} \exp\left(-\frac{1}{2}\|x - x_0\|^2_{C_X^{-1}}\right). \tag{3.56}$$

Here, a symmetric positive definite matrix C induces a scalar product and a norm,

$$(x, y)_C := (Cx, y), \qquad \|x\|^2_C := (x, x)_C. \tag{3.57}$$

If both the prior and the noise random variables are modeled by Gaussian random variables with mean x_0, e_0 and covariance matrices C_X, C_E of the prior and the noise, respectively, then the optimization functional in (3.55) is proportional to

$$\exp\left(-\frac{1}{2}\left(\|x - x_0\|^2_{C_X^{-1}} + \|y^\delta - Tx - e_0\|^2_{C_E^{-1}}\right)\right). \tag{3.58}$$

Thus, the maximum of (3.55) is attained when the negative argument of the exponent is minimized, i.e.,

$$x_{\text{MAP}} = \operatorname*{argmin}_{x \in \mathbb{R}^n} \left(\|x - x_0\|^2_{C_X^{-1}} + \|y^\delta - Tx - e_0\|^2_{C_E^{-1}}\right). \tag{3.59}$$

The solution to the minimization problem is given by the solution of the linear system of equations

$$(T^* C_E^{-1} T + C_X^{-1})x = T^* C_E^{-1}(y^\delta - e_0) + C_X^{-1} x_0. \tag{3.60}$$

In the special case, where the prior and the noise are Gaussian random variables with zero mean and the noise components are independent and identically distributed with variance σ_E^2, the minimization problem (3.59) reduces to

$$x_{\text{MAP}} = \operatorname*{argmin}_{x \in \mathbb{R}^n} \left(\sigma_E^2 \|x\|^2_{C_X^{-1}} + \|y^\delta - Tx\|^2\right). \tag{3.61}$$

In this special case, the MAP estimates coincides with Tikhonov functional (3.28) where the regularization parameter is chosen according to the information on the noise and the choice of the solution space depends on the prior distribution. It can be observed that both the choice of the solution space and the regularization parameter have an interdependent relation on the regularized solution.

3.4 Wavelets

We will use wavelet bases for the discretization of the AO related operators. In the section we briefly introduce the necessary components of wavelets, which are used in our work. We start with the wavelet basis of the multiresolution analysis of $L^2(\mathbb{R})$, then we discuss the wavelet decomposition and the discrete wavelet transform. We then turn to wavelets on a bounded domain and finally conclude with wavelets in two dimensions.

3.4.1 The multiresolution analysis

We introduce wavelets via the definition of a multiresolution analysis (MRA) of $L^2(\mathbb{R})$.

Definition 3.9 ([10])**.** *A* multiresolution analysis *(MRA) of $L^2(\mathbb{R})$ consists of a nested set of approximation spaces of $L^2(\mathbb{R})$,*

$$\{0\} \subset ... \subset V_{-1} \subset V_0 \subset V_1 \subset ... \subset L^2(\mathbb{R}), \tag{3.62}$$

with the following properties,

$$\overline{\bigcup_{j\in\mathbb{Z}} V_j} = L^2(\mathbb{R}), \tag{3.63}$$

$$\bigcap_{j\in\mathbb{Z}} V_j = \{0\}, \tag{3.64}$$

$$f \in V_j \iff f(2^{-j}\cdot) \in V_0, \tag{3.65}$$

$$f \in V_0 \implies f(\cdot - k) \in V_0, \tag{3.66}$$

for all $j, k \in \mathbb{Z}$, and there exists a function $\varphi \in V_0$, such that the set

$$\{\varphi(\cdot - k) : k \in \mathbb{Z}\} \tag{3.67}$$

forms an orthonormal basis of V_0.

Whereas the properties (3.62)-(3.64) are fulfilled by many spaces, properties (3.65)-(3.67) are the defining characteristics of an MRA.

The functions

$$\varphi_{jk}(x) := 2^{j/2}\varphi(2^j x - k), \tag{3.68}$$

where j is the *scale index* and k is the *shift index* of φ are referred to as the *scaling functions*. The spaces V_j are spanned by the scaling functions at scale $j \in \mathbb{Z}$,

$$V_j = \text{span}\{\varphi_{jk} : k \in \mathbb{Z}\}. \tag{3.69}$$

It can be shown [10] that if the conditions (3.62)-(3.67) are satisfied, then there exists a *wavelet function* ψ, such that the span of

$$\psi_{jk}(x) := 2^{j/2}\psi(2^j x - k) \tag{3.70}$$

forms an orthonormal basis of $L^2(\mathbb{R})$ for $j, k \in \mathbb{Z}$. Moreover, the space

$$W_j := \text{span}\{\psi_{jk} : k \in \mathbb{Z}\} \tag{3.71}$$

is the orthogonal complement of V_j in V_{j+1} for each $j \in \mathbb{Z}$, i.e.,

$$V_{j+1} = V_j \oplus W_j. \tag{3.72}$$

Observe that the spaces W_j are orthogonal to each other,

$$W_j \perp W_\ell, \qquad j \neq \ell. \tag{3.73}$$

One way to construct such function ψ is via the so called low-pass and high-pass filters, which we explain below. Since $\varphi \in V_0 \subset V_1$, the function φ can be represented via the basis of V_1,

$$\varphi(x) = \sum_{k \in \mathbb{Z}} \langle \varphi, \varphi_{1k} \rangle \varphi_{1k}(x). \tag{3.74}$$

The coefficients

$$l_k := \langle \varphi, \varphi_{1k} \rangle, \tag{3.75}$$

for $k \in \mathbb{Z}$, are referred to as the *low-pass filter* coefficients. Here $\langle \cdot, \cdot \rangle$ is the $L^2(\mathbb{R})$ scalar product. If the support of φ is finite, then the sequence (l_k) is also finite.

We now turn our discussion to the Daubechies N wavelets [9, 6], a special family of wavelets that have a finite support,

$$\overline{\operatorname{supp} \varphi} = [0, 2N-1], \tag{3.76}$$

and a fast decay in the frequency domain. Asymptotically, with an increasing order N, the support of the wavelet widens and a faster decay in the frequency domain is attained, see [10].

The low-pass filter of the Daubechies N wavelets are of dimension $2N$. We denote the vector of low-pass filter coefficients by

$$l = (l_0, ..., l_{2N-1}). \tag{3.77}$$

Moreover, for the Daubechies N wavelets, the *high-pass filter* coefficients are related to the low-pass filter coefficients according to

$$h_k := (-1)^k l_{2N-k-1}. \tag{3.78}$$

The high-pass filter is also of a finite length, $2N$; we denote the associated vector by

$$h = (h_0, ..., h_{2N-1}) = (l_{2N-1}, ..., -l_0). \tag{3.79}$$

Since $\psi \in W_0 \subset V_1$, the wavelet function ψ has a decomposition in V_1, similar to (3.74),

$$\psi(x) = \sum_{k \in \mathbb{Z}} \langle \psi, \varphi_{1k} \rangle \varphi_{1k}(x), \tag{3.80}$$

It can be shown [10] that a wavelet function ψ can be constructed via the high-pass filter coefficients,

$$h_k = \langle \psi, \varphi_{1k} \rangle. \tag{3.81}$$

Daubechies wavelet are not available analytically, except in the case of Haar wavelets, i.e., $N = 1$. Instead, they are defined via their filter coefficients. Using the filter coefficients l and h, a numerical approximation of the wavelets can be constructed. However, in our later discussion we show that the scaling and the wavelet functions are never needed explicitly.

3.4.2 Wavelet decomposition

The MRA of $L^2(\mathbb{R})$ generates different ways to decompose the function space,

$$L^2(\mathbb{R}) = \bigoplus_{j\in\mathbb{Z}} W_j = V_{j_0} \oplus \left(\bigoplus_{j\geq j_0} W_j \right), \tag{3.82}$$

where j_0 is an arbitrary scale of the approximation space.

Thus, a function f in $L^2(\mathbb{R})$ has a decomposition with respect to the wavelet basis,

$$f(x) = \sum_{j,k\in\mathbb{Z}} \langle f, \psi_{jk}\rangle \psi_{jk}. \tag{3.83}$$

We refer to

$$d_{jk} := \langle f, \psi_{jk}\rangle \tag{3.84}$$

as the *detail coefficients* of the decomposition. Alternatively, f can be decomposed with respect to the scaling and the wavelet functions,

$$f(x) = \sum_{k\in\mathbb{Z}} \langle f, \varphi_{j_0 k}\rangle \varphi_{j_0 k} + \sum_{j\geq j_0} \sum_{k\in\mathbb{Z}} \langle f, \psi_{jk}\rangle \psi_{jk}, \tag{3.85}$$

We refer to

$$a_{jk} := \langle f, \varphi_{jk}\rangle \tag{3.86}$$

as the *approximation coefficients*.

The sequence $(d_{jk})_{j,k\in\mathbb{Z}}$ and $(a_{j_0,k}, d_{jk})_{j\geq j_0, k\in\mathbb{Z}}$ are in ℓ^2. The coefficient representation of f is norm-preserving,

$$\|f\|_{L^2} = ||(d_{jk})_{j,k\in\mathbb{Z}}||_{\ell^2} = ||(a_{j_0,k}, d_{jk})_{j\geq j_0, k\in\mathbb{Z}}||_{\ell^2}. \tag{3.87}$$

Here, $\|\cdot\|_{L^2}$ is the $L^2(\mathbb{R})$ norm and $\|\cdot\|_{\ell^2}$ is the ℓ^2 norm.

3.4.3 Discrete wavelet transform

We are now able to derive a very important tool to work with wavelets, the *discrete wavelet transform* (DWT). The DWT relates the approximation coefficients at a fine scale to the approximation coefficients at a coarse scale and the detail coefficients at the intermediate scales via a convolution of the coefficients with the low-pass and high-pass filters.

Let j_0 and J be two scale indices, such that $j_0 < J$. Without loss of generality, let $j_0 = 0$. Then, applying (3.72) recursively, we obtain

$$V_J = V_0 \oplus W_0 \oplus W_1 \oplus ... \oplus W_{J-1}, \tag{3.88}$$

i.e., an approximation of $L^2(\mathbb{R})$ at a fine scale J can be represented by a coarse scale approximation of $L^2(\mathbb{R})$ at scale $j = 0$ and the details at scales $j = 0, 1, ..., J-1$.

The DWT is a mapping from the approximation coefficients at the fine scale $(a_{Jk})_{k\in\mathbb{Z}}$ to the coefficients $(a_{0k}, d_{jk})_{0\leq j<J, k\in\mathbb{Z}}$, which we refer to as the *wavelet coefficients*. This mapping is also referred to as the *wavelet decomposition*. The transform is performed by

recursively applying a convolution operation at each scale $0 \leq j < J$ to obtain a_{jk}, d_{jk} from $a_{j+1,k}$, which we discuss below.

The reverse operation, i.e., the reconstruction of the approximation coefficients, at a fine scale from the wavelet coefficients is referred to as the *wavelet reconstruction*. An *inverse discrete wavelet transform* performs the wavelet reconstruction and can, as the DWT, be defined via a recurrence of convolution operations.

To define these recursive relations, we first make the following observation. Since $V_j \subset V_{j+1}$ and $W_j \subset V_{j+1}$ holds for all j, relations (3.74), (3.80) can be generalized to

$$\varphi_{jn} = \sum_{k\in\mathbb{Z}} l_{k-2n}\varphi_{j+1,k}, \tag{3.89}$$

$$\psi_{jn} = \sum_{k\in\mathbb{Z}} h_{k-2n}\varphi_{j+1,k}, \tag{3.90}$$

for any shift $n \in \mathbb{Z}$ and scale $j \in \mathbb{Z}$. Indeed, due to (3.68), (3.74) and by performing an index shift,

$$\begin{aligned}\varphi_{jn}(x) &= 2^{j/2}\varphi(2^j x - n) = 2^{(j+1)/2}\sum_{k\in\mathbb{Z}} l_k\varphi(2^{j+1}x - 2n - k) \\ &= 2^{(j+1)/2}\sum_{k\in\mathbb{Z}} l_{k-2n}\varphi(2^{j+1}x - k) = \sum_{k\in\mathbb{Z}} l_{k-2n}\varphi_{j+1,k}(x)\end{aligned} \tag{3.91}$$

holds, which implies (3.89); in a similar way (3.90) can be shown.

A function $f \in V_{j+1}$ has a decomposition in V_{j+1} and in $V_j \oplus W_j$,

$$f = \sum_{k\in\mathbb{Z}} a_{j+1,k}\varphi_{j+1,k} = \sum_{k\in\mathbb{Z}} a_{jk}\varphi_{jk} + \sum_{k\in\mathbb{Z}} d_{jk}\psi_{jk}. \tag{3.92}$$

Using the relations (3.89) and (3.90), the convolution operation for the DWT and the inverse DWT are derived. The approximation and the detail coefficients of f at scale j can be computed by a convolution of the approximation coefficients at scale $j+1$ with the low-pass and the high-pass filters,

$$a_{jn} = \sum_{k\in\mathbb{Z}} l_{k-2n}a_{j+1,k} \quad \text{and} \quad d_{jn} = \sum_{k\in\mathbb{Z}} h_{k-2n}a_{j+1,k}. \tag{3.93}$$

Conversely, the approximation coefficients at scale $j+1$ are synthesized via

$$a_{j+1,n} = \sum_{k\in\mathbb{Z}} l_{n-2k}a_{jk} + \sum_{k\in\mathbb{Z}} h_{n-2k}d_{jk}. \tag{3.94}$$

The decomposition equations (3.93) are shown by computing the scalar product of (3.89) or (3.90) with f, respectively. The first equation holds due to

$$a_{jn} = \langle\varphi_{jn}, f\rangle = \langle\sum_{k\in\mathbb{Z}} l_{k-2n}\varphi_{j+1,k}, \sum_{m\in\mathbb{Z}} a_{j+1,m}\varphi_{j+1,m}\rangle = \sum_{k\in\mathbb{Z}} l_{k-2n}a_{j+1,k}, \tag{3.95}$$

and the second can be shown in a similar way. By performing a coefficient comparison, the reconstruction equation (3.94) holds due to (3.89) and (3.90),

$$\begin{aligned}\sum_{k\in\mathbb{Z}} a_{j+1,k}\varphi_{j+1,k} &= \sum_{n\in\mathbb{Z}} (a_{jn}\varphi_{jn} + d_{jn}\psi_{jn}) \\ &= \sum_{n\in\mathbb{Z}} \left(a_{jn} \sum_{k\in\mathbb{Z}} l_{k-2n}\varphi_{j+1,k} + d_{jn} \sum_{k\in\mathbb{Z}} h_{k-2n}\varphi_{j+1,k} \right) \\ &= \sum_{k\in\mathbb{Z}} \left(\sum_{n\in\mathbb{Z}} l_{k-2n}a_{jn} + h_{k-2n}d_{jn} \right) \varphi_{j+1,k}.\end{aligned} \tag{3.96}$$

This verifies the governing relations of the DWT, equations (3.93) and (3.94).

3.4.4 Wavelets on a bounded domain

In general, wavelet bases of $L^2(\mathbb{R})$ do not automatically form an orthonormal basis on an interval. There are several possibilities to introduce the orthonormal wavelet bases on an interval, for instance by using periodic wavelets, folded wavelets, or boundary wavelets [42].

Periodic wavelets refer to extending the wavelets periodically around the boundary of the interval. *Folded wavelets* refer to symmetrically extending the function to the outside of the boundary. *Boundary wavelets* are a special kind of scaling and wavelet functions, which are adapted to form an orthonormal basis at the boundary.

In this work, we will only concentrate on the periodic wavelets. For convenience, let us focus on the interval $[0, 1]$. Then, the periodic scaling and wavelet functions are defined as follows,

$$\varphi_{jk}^{\text{per}}(x) := \sum_{t\in\mathbb{Z}} \varphi_{jk}(x+t), \tag{3.97}$$

$$\psi_{jk}^{\text{per}}(x) := \sum_{t\in\mathbb{Z}} \psi_{jk}(x+t). \tag{3.98}$$

The shifts "wrap" the function periodically around the boundary. These periodic wavelets form a basis of $L^2([0,1])$ [42].

To simplify the notation, from this point on we will omit the distinction between the periodic and non-periodic wavelets when referring to the wavelets on an interval.

Notice that for periodic wavelets, the approximation and the detail spaces V_j, W_j are now finite dimensional, due to a finite number of shifts. In fact, there are a total of 2^j shifts at each scale j. For instance, at scale $j = 0$, V_0 consists of only one function, as all shifts of the scaling function coincide with φ_{00},

$$\varphi_{0k}(x) = \sum_{t\in\mathbb{Z}} \varphi(x+t-k) = \sum_{t\in\mathbb{Z}} \varphi(x+t) = \varphi_{00}(x), \tag{3.99}$$

for any $k \in \mathbb{Z}$.

Let f be a function in the approximation space V_J. Then f has the decomposition with a finite number of terms,

$$f = \sum_{k=0}^{2^J-1} a_{Jk}\varphi_{Jk} = a_{00}\varphi_{00} + \sum_{j=0}^{J-1} \sum_{k=0}^{2^j-1} d_{jk}\psi_{jk}. \tag{3.100}$$

Observe that both decompositions consist of $2^J = 1 + \sum_{j=0}^{J-1} 2^j$ coefficients.

As the function decomposition is finite, so is the number of convolution operations (3.93) and (3.94) that constitute the direct and the inverse wavelet transform, respectively.

For simplicity, we write down the convolution operations in the matrix form. At each scale $j = 0, ..., J-1$, we define a matrix W_j of dimension $2^{j+1} \times 2^{j+1}$, by

$$(W_j)_{i,(2i+p \bmod 2^{j+1})} = \sum_{\substack{0 \le q < 2N \\ (q \bmod 2^{j+1}) = p}} l_q, \tag{3.101}$$

$$(W_j)_{2^j+i,(2i+p \bmod 2^{j+1})} = \sum_{\substack{0 \le q < 2N \\ (q \bmod 2^{j+1}) = p}} h_q,$$

for $0 \le i < 2^j$ and $0 \le p < 2N$, where $2N$ is the filter length of the low-pass filter (3.75) and the high pass filter (3.78). The matrix W_j corresponds to the discrete wavelet transform at one scale.

The matrix is split into two parts. The first 2^j rows contain the low-pass filters l, repeated in each row, with a column-shift of two. The periodic extension on the boundary is implemented by wrapping the coefficients around. The last 2^j rows contain the high-pass filters h arranged in the same way. For a schematic representation of W_j, see Figure 3.1.

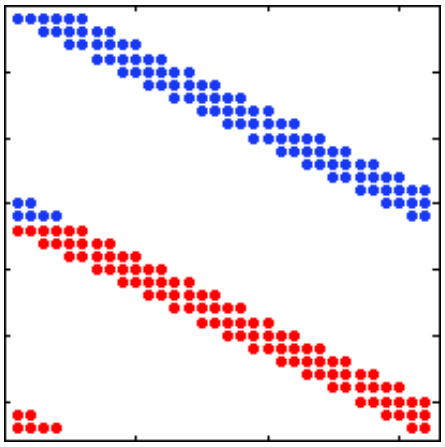

Figure 3.1: Schematic representation of a DWT matrix W_4 for the Daubechies 3 wavelets. Non-zero entries of the matrix are marked. Daubechies 3 filters l and h have six coefficients. Low-pass filter (blue) are repeated in rows 0 to $2^4 - 1$, high-pass filters (red) are repeated in rows 2^4 to $2^5 - 1$. Periodic extension of the wavelets on the boundary is represented by "wrapping" the filter coefficients.

Let us denote the approximation coefficients at the scale j as the column-vector $a_j = (a_{j0}, ..., a_{j,2^j-1})$ and the vector of detail coefficients as the column-vector $d_j = (d_{j0}, ..., d_{j,2^j-1})$.

Then, the wavelet transform at one scale is a sequence of convolution operations (3.93) that transform the approximation coefficients at the fine scale a_{j+1} to the approximation coefficients a_j and the detail coefficients d_j at the coarse scale. In matrix notation, the

wavelet transform corresponds to a matrix multiplication,

$$W_j a_{j+1} = \begin{pmatrix} a_j \\ d_j \end{pmatrix}, \tag{3.102}$$

In the same way, the inverse wavelet transform at one scale is a sequence of convolution operations (3.94) that synthesize the approximation coefficients at the fine scale a_{j+1} using the approximation coefficients a_j and the detail coefficients d_j at the coarse scale. In the matrix notation, the inverse wavelet transform corresponds to a matrix multiplication by the transpose,

$$W_j^T \begin{pmatrix} a_j \\ d_j \end{pmatrix} = a_{j+1}. \tag{3.103}$$

Notice, the wavelet transform at one scale W_j is an orthogonal matrix, i.e., $W_j^T W_j = W_j W_j^T = I$, where I is the identity matrix.

Thus, the direct DWT, which is a mapping of a_J to $(a_0, d_0, ..., d_{J-1})$, is a sequence of recursive multiplications by W_{J-1}, ..., W_0. Correspondingly, the inverse DWT is mapping of $(a_0, d_0, ..., d_{J-1})$ to a_J via recursive multiplications by $W_0^T, ..., W_{J-1}^T$.

Finally, we state how the DWT can be applied to a piecewise linear continuous function to transform its nodal values to wavelet coefficients. Let $x_k = \delta k$, for $0 \leq k < 2^J$, be a set of 2^J points with equidistant spacing $\delta > 0$. Further, let $f : [0, \delta(2^J - 1)] \to \mathbb{R}$ be a piecewise linear function over the nodes x_k, which is linear on the intervals $[x_{k-1}, x_k]$, for $0 < k < 2^J$.

Then, the approximation coefficients at the finest scale are approximated from the nodal values of f, according to

$$a_{Jk} \approx \delta^{1/2} f(\delta k), \tag{3.104}$$

for $0 \leq k < 2^J$. It can be shown [42] that the scaling factor $\delta^{1/2}$ enforces a norm-preserving representation,

$$\|f\|_{L^2} \approx ||a_J||_{\ell^2} = ||c||_{\ell^2}, \tag{3.105}$$

where $c = (a_{00}, d_{00}, d_{10}, ..., d_{J-1,2^{J-1}-1})$ is the vector of wavelet coefficients.

3.4.5 Wavelets in two dimensions

One way to generalize wavelets to two dimension is by using tensor products. The MRA of $L^2(\mathbb{R}^2)$ is generated by the approximation spaces

$$\mathbf{V}_j = V_j \otimes V_j, \tag{3.106}$$

which are formed by tensor products of V_j in both the x and the y variables. In two dimensions there are three types of detail spaces,

$$\begin{aligned} \mathbf{W}_j^1 &= V_j \otimes W_j, \\ \mathbf{W}_j^2 &= W_j \otimes V_j, \\ \mathbf{W}_j^3 &= W_j \otimes W_j, \end{aligned} \tag{3.107}$$

which are are referred to as the *horizontal*, *vertical* and *diagonal detail spaces*. Similar to (3.82) in one dimension, the representation of $L^2(\mathbb{R}^2)$ is given by

$$L^2(\mathbb{R}^2) = \bigoplus_{j\in\mathbb{Z}} (\mathbf{W}_j^1 \oplus \mathbf{W}_j^2 \oplus \mathbf{W}_j^3) = \mathbf{V}_{j_0} \oplus \left(\bigoplus_{j\geq j_0} (\mathbf{W}_j^1 \oplus \mathbf{W}_j^2 \oplus \mathbf{W}_j^3) \right), \tag{3.108}$$

for any index $j_0 \in \mathbb{Z}$.

The scaling functions that span $\mathbf{V}_j$ are given by

$$\varphi_{jk}(x,y) = \varphi_{jk_1}(x)\varphi_{jk_2}(y), \tag{3.109}$$

and the three types of wavelet functions by

$$\begin{aligned} \psi_{jk}^1(x,y) &= \varphi_{jk_1}(x)\psi_{jk_2}(y), \\ \psi_{jk}^2(x,y) &= \psi_{jk_1}(x)\varphi_{jk_2}(y), \\ \psi_{jk}^3(x,y) &= \psi_{jk_1}(x)\psi_{jk_2}(y), \end{aligned} \tag{3.110}$$

for the two-dimensional shift indices $k = (k_1, k_2) \in \mathbb{Z}^2$.

We use the periodic wavelets on an interval in the x and y variables to obtain the periodic wavelets on a square domain. Thus, on a $[0,1]^2$ square, there are a finite number of shifts $0 \leq k_1, k_2 < 2^j$ at scale j, with a total of 2^{2j} scaling or wavelet functions.

The approximation coefficients of $f \in L^2(\mathbb{R}^2)$ at scale j are given by a matrix a_j of dimension $2^j \times 2^j$,

$$(a_j)_{k_1k_2} := a_{jk} := \langle f, \varphi_{jk} \rangle, \tag{3.111}$$

whereas the three types of detail coefficients of f at scale j are denoted by matrices d_j^t of $2^j \times 2^j$ elements,

$$(d_j^t)_{k_1k_2} := d_{jk}^t := \langle f, \psi_{jk}^t \rangle \tag{3.112}$$

for $t = 1, 2, 3$. Here, $k = (k_1, k_2)$ with $0 \leq k_1, k_2 < 2^j$.

The representation of $f \in \mathbf{V}_J$ at some fine scale J in two dimensions is given by

$$f = \sum_{k_1,k_2=0}^{2^J-1} a_{Jk}\varphi_{Jk} = a_{00}\varphi_{00} + \sum_{j=0}^{J-1} \sum_{t=1}^{3} \sum_{k_1,k_2=0}^{2^j-1} d_{jk}^t \psi_{jk}^t. \tag{3.113}$$

The wavelet transform at one scale j in two dimensions corresponds to two multiplications by the wavelet transform matrix W_j,

$$(W_j(W_j a_{j+1})^T)^T = W_j a_{j+1} W_j^T = \begin{pmatrix} a_j & d_j^1 \\ d_j^2 & d_j^3 \end{pmatrix}. \tag{3.114}$$

The wavelet transform matrix is given by (3.101) as in the one-dimensional case. In a similar way, one level of the inverse DWT in two dimensions is realized through the transposed multiplication,

$$\left(W_j^T \left(W_j^T \begin{pmatrix} a_j & d_j^1 \\ d_j^2 & d_j^3 \end{pmatrix} \right)^T \right)^T = W_j^T \begin{pmatrix} a_j & d_j^1 \\ d_j^2 & d_j^3 \end{pmatrix} W_j = a_{j+1}. \tag{3.115}$$

As in one dimension, the DWT corresponds to applying (3.114) recursively for $j = J-1, ..., 0$; the inverse DWT corresponds to applying (3.115) recursively for $j = 0, ..., J-1$.

The DWT can be applied to a continuous piecewise bilinear function to transform its nodal values to wavelet coefficients. Let

$$\{(x_{k_1}, x_{k_2}) : 0 \leq k_1, k_2, < 2^J\} \tag{3.116}$$

be a grid of 2^{2J} equidistant points in $\mathbb{R}^2$, where $x_{k_1} = \delta k_1$ for $0 \leq k_1 < 2^J$ and $\delta > 0$ is the spacing constant. A continuous piecewise bilinear function

$$f : [0, \delta(2^J - 1)]^2 \to \mathbb{R} \tag{3.117}$$

over the grid points (3.116) can be approximated via the approximation coefficients at scale $j = J$ according to

$$a_{Jk} \approx \delta f(\delta k_1, \delta k_2), \tag{3.118}$$

for $k = (k_1, k_2)$, where $0 \leq k_1, k_2 < 2^J$. The scaling factor δ enforces a norm-preserving representation [42],

$$\|f\|_{L^2}^2 \approx ||a_J||_{\mathrm{F}}^2 = ||c||_{\ell^2}^2, \tag{3.119}$$

where $||\cdot||_{\mathrm{F}}$ is the Frobenius norm and

$$c = (a_{00}, d_{00}^1, d_{00}^2, d_{00}^3, d_{10}^1, ..., d_{J-1,(2^{J-1}-1,2^{J-1}-1)}^3) \tag{3.120}$$

is the vector of wavelet coefficients, see (3.113).

In the vector form, we index the wavelet coefficients according to a global wavelet index $\lambda = 0, ..., 2^{2J} - 1$ where,

$$\lambda = \lambda(j, k, t) = 2^{2j}t + 2^j k_2 + k_1. \tag{3.121}$$

Thus, $c_{\lambda(0,(0,0),0)} = c_0 = a_{00}$ and $c_{\lambda(j,k,t)} = d_{jk}^t$ for $0 \leq j < J$, $k = (k_1, k_2)$, $0 \leq k_1, k_2 < 2^j$ and $t = 1, 2, 3$. Observe that λ is a bijective mapping, i.e., an index λ defines the unique set of parameters j, k, t.

3.5 Methods for solving linear systems

Discretization of many problems leads to a linear system of equations

$$Ax = b, \tag{3.122}$$

with a square matrix A of dimensions $n \times n$, where b is a known vector and x is the unknown vector, both of dimension n. A discretized system of equations can be solved on a computer. The goal of this section is to recap some of the efficient techniques to solve equation (3.122).

Solution methods are often compared to each other in terms of their *computational complexity*, which is measured in the number of operations needed to compute a solution. The computational complexity is typically stated asymptotically in the order of n, for large values of n. Problems that are time sensitive require a fast method to solve (3.122).

In this sense, a method with *linear complexity*, i.e., that requires $\mathcal{O}(n)$ operations to obtain a solution is preferred over a method with *quadratic complexity*, i.e., $\mathcal{O}(n^2)$.

Anther important factor of a method is the required storage in memory. The memory of a physical computing system is finite. Hence, for very large systems (3.122), solution methods that require less storage in memory have an advantage.

Another typical requirement of the solution techniques for solving (3.122), especially for large n, is parallelization. We refer to a method as *parallelizable* if the whole method or the major parts of the method can be performed simultaneously, without interdependence of the intermediate results. A parallelizable method has the advantage of reducing the computing time if the components of the method are executed on different computing systems in parallel.

We now make some classifications of the matrix A. A matrix is called *sparse* if many of its entries are zero. Conversely, the matrix which is not sparse is called *full*. A full matrix generally needs n^2 units of memory for storage and $2n^2 - n$ *floating point operations* (FLOP) to multiply it with a vector. In contrast, storage and application of a sparse matrix is in the order of $\text{nnz}(A)$, where $\text{nnz}(A) \in \mathbb{N}$ is the number of non-zero elements of A.

In many applications, A may be available as a linear function $A : \mathbb{R}^n \to \mathbb{R}^n$ that corresponds to the explicit matrix $A \in \mathbb{R}^{n\times n}$. Sometimes, numerical advantage can be gained by storing and applying the matrix A as a function instead of a full (or a sparse) matrix.

For instance, the matrix $A = vv^T$, where v is some vector in $\mathbb{R}^n$, requires n^2 units of memory to store the full matrix and $2n^2 - n$ FLOP to apply it. On the contrary, if only the vector v is stored in memory, the memory consumption is just n and the application of A to a vector x requires $3n - 1$ FLOP by sequentially multiplying x with v^T and then with v. In this work we will utilize special decompositions of A, which will lead to *"matrix-free"* methods for solving linear system of equations (3.122).

In the following we will focus on matrices A that are *symmetric positive definite* (SPD). A square matrix A is *symmetric* if $A = A^T$. A symmetric matrix is *positive definite* if $(Ax, x) > 0$ for all $x \in \mathbb{R}^n, x \neq 0$. Here, $(\cdot, \cdot)$ denotes the Euclidean scalar product.

3.5.1 Matrix multiplication

The most straightforward approach to solving (3.122) is by directly computing the inverse of A. If A is invertible and A^{-1} is available, the equation (3.122) is solved by multiplication,

$$x = A^{-1}b. \tag{3.123}$$

Such a method is referred to as a *matrix vector multiplication* (MVM) method.

Often, even if A is a sparse matrix or has a favorable matrix-free representation, its inverse can be full and the multiplication (3.123) requires $\mathcal{O}(n^2)$ operations. Storage of a full inverse needs n^2 units of memory; computing the inverse is in the order of $\mathcal{O}(n^3)$. Thus, for large n, such a method can be computationally very demanding or even unfeasible.

The positive feature of the MVM is that it is fully parallelizable. Due to

$$x_i = A_i^{-1}b, \tag{3.124}$$

for $0 \leq i < n$, where A_i^{-1} is the i-th row of A^{-1}, the multiplication step in (3.123) can be performed on separate systems for each component of the solution x_i without any intermediate communication.

3.5.2 Matrix factorization

Another direct method for solving (3.122) is based on the decomposition of the matrix in triangular blocks,

$$A = LU, \tag{3.125}$$

where L is a lower-triangular and U is an upper triangular matrix. This method is referred to as the *LU-factorization* and it is related to the *Gaussian elimination* algorithm. The solution is found by solving the two systems subsequently,

$$Lc = b, \tag{3.126}$$

$$Ux = c, \tag{3.127}$$

where $c \in \mathbb{R}^n$ is an auxiliary vector. Each of the two equations is solved by backward substitution.

For SPD matrices, this decomposition is referred to as *Cholesky factorization*, where the decomposition

$$A = LL^T \tag{3.128}$$

consists of the lower-triangular block L and its transpose.

For a full matrix, the memory consumption of the method is now only $n^2/2$, but the computation of the solution is still of quadratic complexity, i.e., $\mathcal{O}(n^2)$; the decomposition of the two factors requires $\mathcal{O}(n^3)$ operations. Moreover, generally, the two backward substitution operations (3.126) and (3.127) are not parallelizable.

There are other direct methods for computing the solution of (3.122), based on other factorizations of A.

For instance, an SPD matrix A has an *eigenvalue decomposition*,

$$A = U\Lambda U^T, \tag{3.129}$$

where U is an orthogonal matrix of *eigenvectors* and $\Lambda = \mathrm{diag}(\lambda_i)$ is a diagonal matrix of real *eigenvalues* λ_i of A. The inverse of A is therefore given by

$$A^{-1} = U\Lambda^{-1}U^T. \tag{3.130}$$

Typically, U is a full matrix. The storage and the computational cost of applying (3.130) is in the same order as the Cholesky factorization.

3.5.3 Conjugate gradient method

A viable alternative to direct solvers, such as the MVM or the Cholesky factorization, are iterative solvers. Instead of inverting or factorizing A, iterative methods use the forward operator recursively to obtain the solution. The methods can be efficient for those problems (3.122), where A is sparse or has an inexpensive representation as a function. The downside of iterative methods is that several iterations need to be performed to

obtain the solution. Note that generally no pre-computation or factorization is needed for iterative methods.

If A has a favorable representation, where, for instance, the storage and the application of A is of linear complexity, and the number of iterations until convergence is limited, then the iterative method has a computational complexity of $\mathcal{O}(n)$.

Many iterative methods are also parallelizable. A necessary requirement for parallelization is that the application of A is also parallelizable. If A is a matrix, full or sparse, this can be guaranteed. However, this does not need to hold for those A that are applied as a linear function.

The most prominent iterative method for (3.122) where A is SPD is the *conjugate gradient* (CG) algorithm. The method is summarized in Algorithm 3.1.

Algorithm 3.1: The CG algorithm

Input : residual $r_0 = b - Ax_0$, initial guess x_0
Output: solution x

for *i=1, ...* **do**
  $\rho_{i-1} = (r_{i-1}, r_{i-1})$
  if *i=1* **then**
    $p_i = r_{i-1}$
  else
    $p_i = r_{i-1} + (\rho_{i-1}/\rho_{i-2})p_{i-1}$
  $q_i = Ap_i$
  $\alpha_i = \rho_{i-1}/(p_i, q_i)$
  $x_i = x_{i-1} + \alpha_i p_i$
  $r_i = r_{i-1} - \alpha_i p_i$
Set $x = x_i$

The vector x_0 is called the *initial guess* and it is the starting value of the iteration. The vector r_i is the *residual* vector; it relates the discrepancy between the iterative solution at iteration i and the right-hand side vector, $r_i = b - Ax_i$. The vector p_i is called the *search direction*.

The CG iterates satisfy the following properties:

$$(r_i, r_j) = 0 \tag{3.131}$$

for $i, j \geq 0$, $i \neq j$, i.e., the residuals are orthogonal to each other, and

$$(Ap_i, p_j) = 0 \tag{3.132}$$

for $i, j \geq 1$, $i \neq j$, i.e., the search directions are A-orthogonal or *conjugate*.

Theoretically, the CG algorithm requires at most n iterations until convergence, see [62] or (3.137) below. However, performing n iterations is unfavorable, since then the computational advantage of the iterative method is lost. For instance, even if the cost of applying A scales with $\mathcal{O}(n)$, repeating the iteration n times leads to an iterative method of quadratic complexity. In many situations the iterations may be terminated

earlier if a stopping criterion is fulfilled. Many techniques are developed to decrease the number of iterations, some of which we discuss below.

A possible stopping criterion for the CG method is to terminate the iteration when the relative residual error is below a certain tolerance $\tau \in (0, 1)$,

$$\|r_i\| < \tau \|r_0\|. \tag{3.133}$$

These norms are available without any additional computation, as $\|r_i\| = \sqrt{\rho_i}$.

For time sensitive problems, the number of iterations may be predetermined by the limitations of the computing system. In this case, the algorithm is terminated at iteration n_{iter}, where the number of iterations is determined by the numerical cost per iteration and the total number of floating point operations that the system allows within the limited time frame. However, in this case convergence is not guaranteed.

Convergence is influenced by the eigenvalue decomposition of A. Recall the definition of the norm induced by an SPD matrix (3.57). The convergence result of the CG algorithm, see, e.g., [62], states that

$$\|x_* - x_i\|_A \leq 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^i \|x_* - x_0\|_A, \tag{3.134}$$

where we denote the *exact solution* by x_* and

$$\kappa = \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)} \tag{3.135}$$

is the *condition number* of A. Here, $\lambda_{\max}(A)$ and $\lambda_{\min}(A)$ are the largest and the smallest eigenvalues of A from the eigenvalue decomposition (3.129). This is a loose upper bound and an improved estimate is stated below.

Due to the nature of the algorithm, the iterates x_i have a non-linear relation with the initial guess. In fact, there exists a polynomial $q_i \in P_i$ with $q_i(0) = 1$, where P_i is the space of polynomials of degree at most i, that fulfill the following relations,

$$x_* - x_i = q_i(A)(x_* - x_0) \quad \text{and} \quad r_i = q_i(A) r_0. \tag{3.136}$$

It can be shown (see, e.g., [65]) that

$$\|x_* - x_i\|_A^2 = \min_{q \in P_i, q(0)=1} \sum_{j=0}^{n-1} \frac{(u_j, r_0)^2}{\lambda_j} q(\lambda_j)^2, \tag{3.137}$$

where u_j are the eigenvectors of A, which are the column vectors of U in (3.129) and λ_j are the eigenvalues. Thus, the convergence of the CG algorithm is controlled by two factors: the eigenvalue distribution of the operator and the decomposition of the residual vector in terms of the eigenvectors.

There is a relation of the CG algorithm to inverse problems. The algorithm can be applied directly to the infinite dimensional normal equation (3.10) in Hilbert spaces where the operator T has a closed range, see [49]. For the operators appearing in the context of ill-posed problems, which do not have a closed range, convergence was affirmed for $y \in \mathcal{D}(T^\dagger)$, see [27, 3, 41].

It has been shown that the iterates x_i do not in general converge towards a stable solution for $y \notin \mathcal{D}(T^\dagger)$, see, e.g., [45, 32]. This is in part motivated by the continuous decay of the eigenvalues of T^*T towards zero, see (3.18) and compare (3.19) with (3.137). However, it has been shown that the CG algorithm with a proper stopping rule is a regularization method [46, 32], where the number of iterations is the regularization parameter. For ill-posed problems, this behavior for the infinite dimensional operators can be observed in the discretized linear system of equations too.

Alternatively, the CG method can also be applied to the regularized problem, such as (3.27). In this case, the operator in the equation is SPD and so the CG algorithm exhibits the behavior as when it is applied to a well-posed equation.

3.6 Preconditioning

Convergence behavior of an iterative method is influenced by the eigenvalue distribution of the operator, as can be observed for the CG algorithm from estimates (3.134) and (3.137). The eigenvalue distribution can be modified by a technique called *preconditioning*. A preconditioned system of equations is a transformed system which has the same solution, but where the operator has a more favorable distribution of the eigenvalues in terms of convergence of the iterative solver.

In *left preconditioning*, for example, equation (3.122) is multiplied from the left with the inverse of an invertible matrix M,

$$M^{-1}Ax = M^{-1}b. \tag{3.138}$$

Such a matrix M should on one hand approximate A, but on the other hand have an inverse which is easier to compute than the inverse of A. The two extreme cases are: $M = I$, which is very simple to invert, but (3.138) reduces to solving the original equation (3.122), and $M = A$, which simplifies the equation (3.138) to be solved in one step, but applying the preconditioner is as complicated as solving the original problem (3.122). Finding the right preconditioner that balances these two requirements, is generally not straightforward and typically problem-dependent, i.e., a preconditioner must be tailored to the underlying problem.

The preconditioner can also be applied from the right, in which case it is called a *right preconditioner*,

$$AM^{-1}y = b, \tag{3.139}$$

with

$$M^{-1}y = x. \tag{3.140}$$

Right-preconditioning corresponds to a variable transformation. After the solution y to (3.139) is found, the solution to the original problem is obtained by applying the preconditioner to y in (3.140).

To preserve the symmetry of the problem, a *mixed preconditioner*, which is applied from the left and right, can be used. For example, an SPD matrix M can be factorized into

$$M = M^{1/2}M^{1/2}, \tag{3.141}$$

where $M^{1/2} = U\Lambda^{1/2}U^T$ with Λ and U from the eigenvalue decomposition (3.129) of M. A mixed preconditioner is applied from both left and right sides,

$$M^{-1/2}AM^{-1/2}y = M^{-1/2}b, \tag{3.142}$$

with

$$M^{-1/2}y = x. \tag{3.143}$$

The matrix $M^{-1/2}AM^{-1/2}$ is symmetric and positive definite and thus the CG method can be applied to (3.142). The CG algorithm for equation (3.142) is called the *preconditioned conjugate gradient* (PCG) algorithm. The method is summarized in Algorithm 3.2.

Algorithm 3.2: The PCG algorithm

Input : residual $r_0 = b - Ax_0$, initial guess x_0
Output: solution x

1 **for** *i=1, ...* **do**
2 $z_{i-1} = M^{-1}r_{i-1}$
3 $\rho_{i-1} = (r_{i-1}, z_{i-1})$
4 **if** *i=1* **then**
5 $p_i = z_{i-1}$
6 **else**
7 $p_i = z_{i-1} + (\rho_{i-1}/\rho_{i-2})p_{i-1}$
8 $q_i = Ap_i$
9 $\alpha_i = \rho_{i-1}/(p_i, q_i)$
10 $x_i = x_{i-1} + \alpha_i p_i$
11 $r_i = r_{i-1} - \alpha_i p_i$
12 Set $x = x_i$

In step 2 of Algorithm 3.2, the preconditioner is applied as a full matrix. It can be shown [62] that this coincides with applying the CG algorithm to the mixed preconditioner problem (3.142). Thus, the factorization matrices $M^{1/2}$ or $M^{-1/2}$ need not be computed explicitly.

The algorithm also allows the use of a more generalized preconditioner. A preconditioner can be any linear function $M^{-1} : \mathbb{R}^n \to \mathbb{R}^n$, which acts as the corresponding matrix $M^{-1} \in \mathbb{R}^{n\times n}$. This allows, for instance, the use of the multigrid method as a preconditioner [4].

The estimates (3.134) and (3.137) provide insight into how a good preconditioner changes the convergence behavior of the problem.

The estimate (3.134) suggests the use of a preconditioner that reduces the condition number of the operator. If the condition number of $M^{-1/2}AM^{-1/2}$ is much smaller than the condition number of A, then based on the estimate (3.134) one can expect a faster convergence of the algorithm.

In fact, if M is *spectrally equivalent* to A, i.e., there are two constants $\gamma_1, \gamma_2 > 0$ independent of n such that

$$\gamma_1(Mx, x) \leq (Ax, x) \leq \gamma_2(Mx, x) \tag{3.144}$$

holds for any $x \in \mathbb{R}^n$, then the equivalent relation

$$\gamma_1(x,x) \leq (M^{-1/2}AM^{-1/2}x, x) \leq \gamma_2(x,x) \tag{3.145}$$

implies

$$\gamma_1 \leq \lambda_{\min}(M^{-1/2}AM^{-1/2}) = \min_{x\in\mathbb{R}^n, x\neq 0} \frac{(M^{-1/2}AM^{-1/2}x, x)}{(x,x)} \tag{3.146}$$

and

$$\lambda_{\max}(M^{-1/2}AM^{-1/2}) = \max_{x\in\mathbb{R}^n, x\neq 0} \frac{(M^{-1/2}AM^{-1/2}x, x)}{(x,x)} \leq \gamma_2, \tag{3.147}$$

which yields the bound on the condition number

$$\kappa = \frac{\lambda_{\max}(M^{-1/2}AM^{-1/2})}{\lambda_{\min}(M^{-1/2}AM^{-1/2})} \leq \frac{\gamma_2}{\gamma_1}. \tag{3.148}$$

For tight bounds γ_1, γ_2 independent of the matrix size, the condition number and therefore the convergence rate, of the preconditioned system is bounded independent of the matrix size.

From the estimate (3.137) it can be deduced that clustering of the eigenvalues is of importance. By eigenvalue clustering we refer to eigenvalues which are are clustered around a certain value. In this sense, if the value of the polynomial q in (3.137) is small at a λ_j, then by continuity of the polynomial, the value is also small at all eigenvalues clustered in the neighborhood of λ_j.

One of the basic choices for a preconditioner is the *Jacobi preconditioner*, which is a diagonal matrix with the diagonal components of A,

$$M = \operatorname{diag}(A). \tag{3.149}$$

Although it is in general not spectrally equivalent to A, the Jacobi preconditioner has the effect of centering all the Gerschgorin discs, in which the eigenvalues are contained, around the value 1. Being a diagonal matrix, it is easy to invert.

A matrix $A \in \mathbb{R}^{n\times n}$ has n *Gerschgorin discs*, where the i-th disc is a closed disc centered around a_{ii} with radius $\sum_{j\neq i} |a_{ij}|$. The Gerschgorin circle theorem states that all the eigenvalues are contained within at least one of the Gerschgorin discs. As the diagonal values of the matrix $M^{-1/2}AM^{-1/2}$ with the Jacobi preconditioner (3.149) are all 1, the Gerschgorin discs are all centered around 1. However, the Jacobi preconditioner provides no information about the radii of these discs.

Chapter 4

Atmospheric tomography

4.1 Problem formulation

Atmospheric tomography, or the reconstruction of the refractive index of the turbulent atmosphere, is the fundamental problem of many next generation AO systems, such as MCAO, LTAO or MOAO. We refer to Chapter 2 for more details on the AO systems and their components.

Assuming the layered model of the atmosphere, see Section 2.2, where $\boldsymbol{\phi} = (\phi_1, ..., \phi_L)$ are L turbulence layers at altitudes $0 \leq h_1 < ... < h_L$, the task in the atmospheric tomography problem is to reconstruct the turbulence layers from WFS measurements, see Figure 1.1.

In this work we assume that all measurements are obtained from Shack-Hartmann wavefront sensors. We assume there are G guide stars in directions $\theta_1, ..., \theta_G$, of which the first G_{LGS} are assigned to LGSs and the other $G_{\text{NGS}} = G - G_{\text{LGS}}$ to NGSs. Further, we assume that in the tomography problem, at least one NGS is present to correct for the tip-tilt uncertainty.

The turbulence layers are related to wavefront sensors measurements according to

$$s_g = \Gamma_g P_g^{\text{LGS}} \boldsymbol{\phi} \qquad \text{and} \qquad s_{g'} = \Gamma_{g'} P_{g'}^{\text{NGS}} \boldsymbol{\phi} \tag{4.1}$$

for $1 \leq g \leq G_{\text{LGS}}$ and $G_{\text{LGS}} < g' \leq G$; we explain the components of the equations below. To simplify the notation, we write the sub-index g in place of the direction vector θ_g. Here, P_g^{LGS} is a geometric propagation operator towards a laser guide star (2.45), $P_{g'}^{\text{NGS}}$ is a geometric propagation operator towards a natural guide star (2.41), and Γ_g is the SH-WFS operator (2.37). Note that in principle, the geometry of the SH-WFS may be different for each guide star direction, hence we associate a sub-index to the measurement operator. The elements of the vectors s_g are the Shack-Hartmann measurements.

We denote the *atmospheric tomography* problem by

$$\mathbf{s} = (s_g)_{g=1}^{G} = \mathbf{A}\boldsymbol{\phi}, \tag{4.2}$$

i.e., a concatenation of all equations in (4.1) in all directions.

Due to the small angle of separation, the underlying mathematical problem in (4.2) is extremely ill-posed [11] and regularization is necessary to obtain a stable solution from noisy measurements. To this end, the problem is often formulated in the Bayesian framework, where the statistics of turbulence and noise can be utilized in the model.

We consider $\mathbf{S}$ and $\mathbf{\Phi}$ as random variables corresponding to the vectors of measurements and of turbulence layers, respectively. Further, we assume the presence of additive noise, and denote the noise random variable by $\boldsymbol{\eta}$. Then in the random setting equation (4.2) is formulated as

$$\mathbf{S} = \mathbf{A}\mathbf{\Phi} + \boldsymbol{\eta}. \tag{4.3}$$

The prior and the noise random variables are both modeled as Gaussian with zero mean and covariance matrices $\mathbf{C}_{\mathbf{\Phi}}$ and $\mathbf{C}_{\boldsymbol{\eta}}$, respectively.

Each layer is statistically independent from the others, hence the prior has a block-diagonal structure with respect to layers,

$$\mathbf{C}_{\mathbf{\Phi}} = \operatorname{diag}(C_1, ..., C_L), \tag{4.4}$$

where C_ℓ is the covariance matrix associated to layer $\ell = 1, ..., L$. We assume the von Karman turbulence model (2.16). Therefore, C_ℓ is formally given by

$$C_\ell = \mathcal{F}^{-1}\Phi_\ell\mathcal{F}, \tag{4.5}$$

where $\mathcal{F}$ is the Fourier transform and Φ_ℓ is the spectral density of the turbulence layer in the optical path distance, defined by

$$\Phi_\ell(\kappa) := \frac{0.023 r_0^{-5/3}\lambda^2 C_n^2(h_\ell)}{4\pi^2(|\kappa|^2 + \kappa_0^2)^{11/6}}, \tag{4.6}$$

which is derived from the PSD of the phase component of layer ℓ, see (2.23).

The measurement noise is independently distributed in each wavefront sensor. Thus, also the noise covariance has a block-diagonal structure with respect to guide star directions,

$$\mathbf{C}_{\boldsymbol{\eta}} = \operatorname{diag}(\widetilde{C}_1, ..., \widetilde{C}_{G_{\mathrm{LGS}}}, \widetilde{C}_{G_{\mathrm{LGS}}+1}, ..., \widetilde{C}_G). \tag{4.7}$$

Here, $\widetilde{C}_g$ for $1 \le g \le G_{\mathrm{LGS}}$ is the LGS noise covariance given by (2.51) and $\widetilde{C}_{g'}$ for $G_{\mathrm{LGS}} < g' \le G$ is the NGS noise covariance given by (2.43).

The optimal solution in this setting [20] is the maximum a-posteriori (MAP) estimate, which is obtained by solving the minimization problem

$$\boldsymbol{\phi}_{\mathrm{MAP}} = \arg\min_{\boldsymbol{\phi}} \|\mathbf{C}_{\mathbf{\Phi}}^{-1/2}\boldsymbol{\phi}\|^2 + \|\mathbf{C}_{\boldsymbol{\eta}}^{-1/2}(\mathbf{s} - \mathbf{A}\boldsymbol{\phi})\|^2, \tag{4.8}$$

see (3.59). Equivalently, the MAP estimate is obtained as the solution of the linear system of equations,

$$(\mathbf{A}^*\mathbf{C}_{\boldsymbol{\eta}}^{-1}\mathbf{A} + \mathbf{C}_{\mathbf{\Phi}}^{-1})\boldsymbol{\phi} = \mathbf{A}^*\mathbf{C}_{\boldsymbol{\eta}}^{-1}\mathbf{s}, \tag{4.9}$$

see (3.60), where $\mathbf{A}^*$ is the adjoint of the tomography operator $\mathbf{A}$.

The inverse of $\mathbf{C}_{\boldsymbol{\eta}}$ has a sparse representation, as each of the diagonal blocks is either a diagonal matrix in the case of an NGS or a block-diagonal matrix with 2×2 blocks (2.53) in the LGS case. On the other hand, sparsity of $\mathbf{C}_{\mathbf{\Phi}}^{-1}$ or the forward operator $\mathbf{A}$ depends greatly on the discretization of the problem.

4.2 Standard approach: the MVM

The standard approach to solve equation (4.9) is to explicitly pre-compute the inverse of the discretized left-hand side matrix and then to multiply the pre-computed matrix with the data.

A typical choice for discretization is the basis of Zernike polynomials, see, e.g., [20]. The matrix

$$\mathbf{B} = (\mathbf{A}^* \mathbf{C}_{\boldsymbol{\eta}}^{-1} \mathbf{A} + \mathbf{C}_{\boldsymbol{\Phi}}^{-1})^{-1} \mathbf{A}^* \mathbf{C}_{\boldsymbol{\eta}}^{-1} \tag{4.10}$$

is computed explicitly. The numerical cost of directly inverting the left-hand side matrix of (4.9) scales with $\mathcal{O}(n^3)$, where the matrix is of dimensions $n \times n$. The inverse must be computed once for a specific geometry of the problem (directions θ_g and layer altitudes h_ℓ), certain noise levels (entries of the matrix $\mathbf{C}_{\boldsymbol{\eta}}$) and the turbulence parameters of the layers (r_0 and $C_n^2(h_\ell)$ in the matrix $\mathbf{C}_{\boldsymbol{\Phi}}$). Thus, whenever any of these parameters are modified, matrix $\mathbf{B}$ must be recomputed.

The MAP estimate is then determined by a *matrix-vector multiplication* (MVM),

$$\boldsymbol{\phi}_{\text{MAP}} = \mathbf{B}\mathbf{s}. \tag{4.11}$$

The matrix representation of $\mathbf{B}$ is full. Hence, numerically the method scales with $\mathcal{O}(n^2)$ operations. Still, the method is highly efficient with respect to parallelization, as the matrix-vector multiplication can be performed in parallel, without any intermediate data communication.

Typically, a mirror fitting operator $\mathbf{F}$ is combined with the atmospheric reconstruction. In this case an MVM method is a direct mapping from WFS measurements onto the mirror shapes,

$$\mathbf{a} = \mathbf{F}\mathbf{B}\mathbf{s}, \tag{4.12}$$

with a single pre-computed matrix $\mathbf{FB}$, where $\mathbf{a}$ is a vector of mirror shapes. The dimensions of the matrix, and therefore the computational performance, depends on the dimensions of the system: the number of active subapertures of all sensors and the number of active actuators of all mirrors.

Different discretization strategies of equation (4.9) are possible, hence many variations of the MVM method exist.

4.3 Iterative methods

The dimension of the problem (4.9) increases with the telescope size. Even with heavy parallelization, an MVM method becomes computationally very demanding in the millisecond time frame. Recently, research tended more towards iterative solution strategies. This led to the development of methods that scale with lower than quadratic computational complexity.

Two factors are necessary for the iterative methods to achieve better numerical performance: the left-hand side operator in equation (4.9) must have an efficient representation and the number of iterations must be small. The regularized problem of atmospheric tomography is symmetric and positive definite. This allows the most prominent iterative method for such kind of problems, the conjugate gradient (CG) algorithm, to be used.

One of the best ways to reduce the number of iterations in this scheme is by preconditioning. Several preconditioned conjugate gradient (PCG) methods have been introduced for atmospheric tomography.

The multigrid preconditioners (MG-PCG) have been proposed in [26, 24, 76], where the complexity of the problem was of the order $\mathcal{O}(n^{3/2})$. Later, in [77, 75, 25, 23] the Fourier-domain preconditioning techniques (FD-PCG) led to methods which scale according to $\mathcal{O}(n \log n)$.

Especially for iterative methods it is of value to be able to represent the operators in a sparse form. In this regard the Fourier basis is very useful and Fourier-transform based reconstructors have been proposed in [73, 72, 21]. In other typical bases, the forward operator and the inverse covariance of the noise are fast to apply. However, the inverse covariance of the prior is often a full matrix and thus a sparse approximation is required. In the approach introduced by Ellerbroek [12] the turbulence power law is modified in order to achieve a sparse approximation by biharmonics.

A promising method, called the Fractal Iterative Method (FrIM), has been developed by Tallon and others in [67, 70, 66, 5], where an efficient factorization of the inverse prior matrix yields a method scaling with $\mathcal{O}(n)$ operations. Iterative methods have been developed also outside the Bayesian framework. An algorithm based on the Kaczmarz iteration was proposed in [54, 59, 60] and scales with $\mathcal{O}(n)$ even without preconditioning.

The remainder of this section is a brief overview of the FD-PCG, the FrIM and the Kaczmarz methods.

4.3.1 FD-PCG

Most of the operators in the MAP estimate equation (4.9) have an efficient representation in the Fourier domain. The projection and the Shack-Hartmann operators, as well as the turbulence covariance (4.5), have a diagonal matrix representation in the Fourier space. However, due to the finite size of the pupil, the mask does not.

The Fourier-domain PCG method is based on this sparse representation of the operators. A preconditioner is formulated in the Fourier domain, by approximating the left-hand side operator of (4.9) via

$$\mathbf{A}^* \mathbf{C}_{\boldsymbol{\eta}}^{-1} \mathbf{A} + \mathbf{C}_{\boldsymbol{\Phi}}^{-1} \approx \boldsymbol{\mathcal{F}}^{-1} (\widehat{\mathbf{A}}^* \mathbf{C}_{\boldsymbol{\eta}}^{-1} \widehat{\mathbf{A}} + \widehat{\mathbf{C}}_{\boldsymbol{\Phi}}^{-1}) \boldsymbol{\mathcal{F}}, \tag{4.13}$$

where $\boldsymbol{\mathcal{F}}$ is the block-diagonal Fourier transform on each layer. Here, $\widehat{\mathbf{C}}_{\boldsymbol{\Phi}}^{-1}$ has a diagonal representation, see (4.5) and (4.6), and $\widehat{\mathbf{A}}$ is a sparse approximation of $\mathbf{A}$ in the Fourier space, where the mask is approximated by a sparse matrix.

The FD-PCG method can be formulated in one of two ways. In the first, the problem (4.9) is discretized in the bilinear domain and the sparse approximation of the inverse turbulence covariance by Ellerbroek [12] is utilized. The problem is solved using the PCG algorithm with the sparse preconditioner (4.13). The Fourier transform is applied to invoke the preconditioner.

In the second configuration, the whole problem (4.9) is formulated in the Fourier domain, where all operators have a sparse representation, except for the mask. To apply the mask, the inverse Fourier transform is performed. The same Fourier domain preconditioner without the Fourier transform operations (4.13) is used.

In both configurations, the direct and the inverse Fourier transform is applied once per a PCG iteration. Due to the scalability of the Fourier transform, the method scales globally as $\mathcal{O}(n \log n)$ in both configurations.

4.3.2 FrIM

The Fractal Iterative Method is based on an efficient approximation of the turbulence covariance matrix (4.5),

$$C_\ell \approx K K^T, \tag{4.14}$$

where K is a matrix with a hierarchical structure,

$$K = K_1 K_2 \dots K_p, \tag{4.15}$$

and p is the number of scales. The approximation matrices K_i, for $1 \leq i \leq p$, are based on a modification of a mid-point algorithm [39] relating the phase to the structure function, e.g., (2.18). Matrices K_i are sparse and the decomposition (4.15) allows for K, its inverse or its transposed to be applied with a linear computational complexity, i.e., with $\mathcal{O}(n)$ operations.

In FrIM, the MAP estimate (4.9) is reformulated via a variable substitution,

$$(\mathbf{K}^T \mathbf{A}^* \mathbf{C}_{\boldsymbol{\eta}}^{-1} \mathbf{A} \mathbf{K} + \mathbf{I}) \mathbf{x} = \mathbf{K}^T \mathbf{A}^* \mathbf{C}_{\boldsymbol{\eta}}^{-1} \mathbf{s}, \tag{4.16}$$

where $\boldsymbol{\phi} = \mathbf{K}\mathbf{x}$ and $\mathbf{K}\mathbf{K}^T$ is the block-diagonal approximation of $\mathbf{C}_{\boldsymbol{\Phi}}$, see (4.4). The equation is solved using the PCG iteration with an efficient diagonal preconditioner. The method scales globally as $\mathcal{O}(n)$.

4.3.3 Kaczmarz

The Kaczmarz method differs from the others, as it is not based on the Bayesian formulation of the problem. In the method, the atmospheric tomography problem (4.2) is split into two sub-problems. The first sub-problem is wavefront reconstruction,

$$s_g = \Gamma_g \varphi_g, \tag{4.17}$$

for $1 \leq g \leq G$. In this step, a fast solver for wavefront reconstruction is applied, see, e.g., [57, 58, 53], to obtain the wavefront in each direction.

The second sub-problem is atmospheric tomography for the wavefront data,

$$\varphi_g = P_g^{\mathrm{LGS}} \boldsymbol{\phi} \qquad \text{and} \qquad \varphi_{g'} = P_{g'}^{\mathrm{NGS}} \boldsymbol{\phi}, \tag{4.18}$$

for $1 \leq g \leq G_{\mathrm{LGS}} < g' < G$. This problem is solved using the Kaczmarz iteration [35], which is a typical approach for tomography problems, see, e.g., [45]. In this approach, the Bayesian formalism is avoided.

Splitting the atmospheric tomography problem provides computational advantage, which is realized by a reduction in complexity of each sub-problem. Each sub-problem can therefore be solved quickly, in $\mathcal{O}(n)$ operations.

However, the inclusion of the noise statistics, although possible to some extent, provides to be difficult in this formulation, as opposed to the Bayesian methods. Thus, in some configurations, the method delivers results of lower quality.

Chapter 5

The wavelet method for atmospheric tomography

5.1 The wavelet method

In the following we present our method for solving the atmospheric tomography problem (4.2), which we call the Finite Element-Wavelet Hybrid Algorithm (FEWHA). The FEWHA is an iterative algorithm to compute the MAP estimate (4.8). The main principle behind our method is to use compactly supported orthonormal wavelets to represent the turbulence layers of the atmosphere.

There are certain fundamental advantages of using wavelets. A compactly supported orthonormal wavelet basis exhibits favorable properties in the frequency and the spatial domains. The properties in the frequency domain allow a completely diagonal approximation of the penalty term in (4.8), which we present in Section 5.2. The fitting term in the wavelet basis, however, is not as sparse; this is illustrated in Section 5.3.

To achieve a sparse representation, we discretize the fitting term using a continuous piecewise bilinear basis of finite elements. This dual-domain discretization approach in which the turbulence layers are discretized using a finite element and a wavelet basis simultaneously is presented in Section 5.4. We use the discrete wavelet transform (DWT) to transition between the two discretization domains. Due to the favorable properties of wavelets in the spatial domain, the DWT scales as $\mathcal{O}(n)$, moreover it is parallelizable.

The discretized equation is solved using the PCG algorithm. To reduce the number of iterations, we construct a frequency-dependent preconditioner. In Section 6.2, we show how it can be formulated for the wavelet method as a diagonal matrix. A diagonal preconditioner is especially efficient in terms of the computational cost. Moreover, we further improve convergence by utilizing the multi-scale structure of wavelets, see Section 6.3.

5.2 Turbulence statistics in the wavelet domain

The wavelet representation of the turbulence layers leads to a completely diagonal approximation of the penalty term in the MAP estimate (4.8). This key contribution of the method is discussed in this section.

Wavelets and their approximative properties of Kolmogorov turbulence have been

studied via the structure function before in [48]. However, the typical minimum variance solution methods in adaptive optics utilize Bayesian inference where the inverse covariance of the turbulence model is needed. This is not always easily available from structure function approximations. We are motivated by finding an efficient numerical approximation to the inverse covariance in the wavelet scheme.

We first settle on the notation for the wavelet discretization of the turbulence layers. Let $\phi : \mathbb{R}^2 \to \mathbb{R}$ be a turbulence layer at altitude $h \geq 0$. We consider the MRA of $L^2(\mathbb{R}^2)$, i.e., there exist scaling and wavelet functions φ_{jk} and ψ^t_{jk} with $j \in \mathbb{Z}$, $k = (k_1, k_2) \in \mathbb{Z}^2$ and $t = 1, 2, 3$ given by (3.109) and (3.110), respectively. The wavelet decomposition of a layer ϕ in $L^2(\mathbb{R}^2)$ is given by

$$\phi(x, y) = \sum_{j\in\mathbb{Z}} \sum_{k\in\mathbb{Z}} \sum_{t=1,2,3} \langle \phi, \psi^t_{jk} \rangle \psi^t_{jk}(x, y), \tag{5.1}$$

for $(x, y) \in \mathbb{R}^2$.

We assume that the spectral density of the turbulence field satisfies the von Karman model (4.6). Hence, the turbulence covariance operator of a layer ϕ at altitude h formally satisfies (for details see [33])

$$C_\Phi = c(h)\mathcal{F}^{-1} m \mathcal{F}, \tag{5.2}$$

where the spectral density m multiplies each frequency component with the term

$$m(\kappa) = (\kappa_0^2 + |\kappa|^2)^{-11/6} \tag{5.3}$$

in the frequency domain, $\mathcal{F}$ is the Fourier transform, and the altitude dependent constant $c(h)$ is given via

$$c(h) = \frac{0.023 r_0^{-5/3} \lambda^2 C_n^2(h)}{4\pi^2}, \tag{5.4}$$

see (4.6).

A stochastic process with spectral density m does not have a straightforward random variable representation in Sobolev spaces since its realizations are not compactly supported and do not decay at infinity. Here, we avoid these technical considerations and assume that the prior induces the widely used penalty term [12, 15]

$$\begin{aligned} \|C_\Phi^{-1/2} f\|^2_{L^2} &= \frac{1}{c(h)} \|(\kappa_0^2 + |\kappa|^2)^{11/12} \mathcal{F} f\|^2_{L^2} \\ &\simeq \frac{1}{c(h)} \left(\kappa_0^{11/3} \|f\|^2_{L^2} + \|(-\Delta)^{11/12} f\|^2_{L^2} \right), \end{aligned} \tag{5.5}$$

for any $f \in C_0^\infty(\mathbb{R}^2)$. Here and in what follows, we write $p \simeq q$ if the two pseudo-norms p and q are equivalent.

Assume for the moment that the wavelets studied here are 2-regular, i.e., have 2 vanishing moments and are 2 times continuously differentiable. Then the last term in (5.5) is equivalent to the expression

$$\|(-\Delta)^{11/12} f\|^2_{L^2} \simeq \sum_{j\in\mathbb{Z}} \sum_{k\in\mathbb{Z}^2} \sum_{t=1,2,3} 2^{2(11/6)j} |\langle f, \psi^t_{jk} \rangle|^2, \tag{5.6}$$

where j is the scale index of the wavelet. The equivalence above is known as the Bernstein-Jackson inequalities [44].

We now focus on the discretized turbulence layer. In the discrete setting, the turbulence layer is represented with only a finite number of wavelet scales, $j = 0, ..., J-1$, where $J \in \mathbb{N}$ is the total number of wavelet scales.

Further, we are only interested in reconstructing the turbulence layer on a bounded region, i.e., the region of interest. The shape of this region of interest is determined by the setting of the tomography problem, e.g., by the layer altitude or the star asterism. For our purposes, we define the wavelet decomposition on a square domain in $\mathbb{R}^2$ that is larger than the region of interest. We denote the square domain by

$$\Omega_\phi := [0, \delta(2^J - 1)]^2 - \xi, \tag{5.7}$$

where J is the number of wavelet scales in the discretization, $\delta > 0$ is a scaling factor and $\xi = (\xi^x, \xi^y) \in \mathbb{R}^2$ is a shift of the corner point of the domain away from the origin.

Thus, Ω_ϕ is an arbitrary square domain, parametrized by the choice of J, δ and ξ. The values of these parameters are chosen such that the region of interest is contained within Ω_ϕ. In particular, the domain Ω_ϕ must contain the regions observed by the telescope in all guide star directions, see Section 5.3.

Typically, the region of interest is a proper subset of Ω_ϕ. The reconstruction on the complement of this region in Ω_ϕ is allowed to be arbitrarily poor, as it does not influence the quality of the AO correction.

We consider the periodically extended wavelets on Ω_ϕ. Hence, the wavelet expansion of ϕ takes the form

$$\begin{aligned}\phi(x,y) &= \langle \phi, \varphi_{00} \rangle \varphi_{00}(x,y) + \sum_{j=0}^{J-1} \sum_{k_1=0}^{2^j-1} \sum_{k_2=0}^{2^j-1} \sum_{t=1}^{3} \langle \phi, \psi_{jk}^t \rangle \psi_{jk}^t(x,y) \\ &= a_{00}\varphi_{00} + \sum_{j=0}^{J-1} \sum_{k_1=0}^{2^j-1} \sum_{k_2=0}^{2^j-1} \sum_{t=1}^{3} d_{jk}^t \psi_{jk}^t(x,y),\end{aligned} \tag{5.8}$$

where a_{00} is the approximation coefficient, d_{jk}^t are the detail coefficients, $k = (k_1, k_2)$ and $(x, y) \in \Omega_\phi$.

In the discrete setting, the function f in (5.5) is represented with a finite number of wavelet coefficients. Together with (5.6), this leads to an equivalent representation for the regularizing term in (5.5) by a diagonal matrix D that satisfies

$$D_{\lambda\lambda} = \frac{1}{c(h)} \left(\kappa_0^{11/3} + 2^{(11/3)j} \right), \tag{5.9}$$

for the global wavelet index $\lambda = 0, ..., 2^{2J} - 1$, see (3.121), and the corresponding wavelet scale index $j = 0, ..., J-1$. Note that in practice the term $\kappa_0^{11/3}$ becomes negligible.

This concept is extended to L turbulence layers $\boldsymbol{\phi} = (\phi_1, ..., \phi_L)$ at altitudes $0 \le h_1 < ... < h_L$ of the full problem. At each layer $\ell = 1, ..., L$, we denote the square domain (5.7) by

$$\Omega_\ell := [0, \delta_\ell(2^{J_\ell} - 1)]^2 - \xi_\ell, \tag{5.10}$$

which is determined, up to the shift of the coordinates ξ_ℓ, via the number of wavelet scales J_ℓ and the scaling parameter δ_ℓ. To each layer we associate a wavelet decomposition (5.8) and refer to the vector of wavelet coefficients by c_ℓ at layer ℓ, see (3.120).

The diagonal matrix corresponding to the inverse turbulence covariance in the wavelet domain (5.9) at layer ℓ is denoted by D_ℓ. The values and the dimensions of the matrix are determined via the turbulence strength constant $c(h_\ell)$, see (5.4), at altitude h_ℓ and the number of wavelet scales J_ℓ.

Similar to (4.4) we define a block-diagonal matrix for the full problem,

$$\mathbf{D} = \operatorname{diag}(D_1, ..., D_L). \tag{5.11}$$

The matrix $\mathbf{D}$ is a block-diagonal matrix of diagonal matrices, i.e., a diagonal matrix itself.

Finally, by approximating the prior covariance of the discretized problem at each layer we get

$$\|\mathbf{C}_{\boldsymbol{\Phi}}^{-1/2}\boldsymbol{\phi}\|^2_{(L^2)^L} = \sum_{\ell=1}^{L} \|C_\ell^{-1/2}\phi_\ell\|^2_{L^2} \simeq \sum_{\ell=1}^{L}(D_\ell c_\ell, c_\ell)_2 = (\mathbf{D}\mathbf{c}, \mathbf{c})_2 \tag{5.12}$$

for the vector of wavelet coefficients $\mathbf{c} = (c_\ell)_{\ell=1}^L$ of the full problem.

We point out that similar approximation techniques can be applied to more general classes of power laws. This is valuable since experiments suggest that deviations from the Kolmogorov model may appear [47]. More generalized random processes have been considered, e.g., in [50].

In the numerical simulations we utilize the Daubechies N wavelet basis. They are an orthogonal wavelet family with compact support [10]. For a sufficiently large N, the Daubechies N wavelets are 2-regular. In order to keep the basis functions well-localized in the spatial domain, we have chosen to use $N = 3$.

It is well-known that the Daubechies 3 wavelets belong to the Hölder space $C^{1+\delta}(\mathbb{R}^2)$ with $\delta \approx 0.0878$ [10]. Even though they are not 2-regular, they form a Riesz basis in $H^2(\mathbb{R}^2)$ [8]. Based on our numerical tests we believe that this is sufficient in practice.

The choice of periodic wavelets on a bounded domain is motivated by implementational considerations. The effect of artifacts appearing due to the non-periodicity of the turbulence layer is mitigated by sufficiently extending the domain Ω_ℓ outside of the region of interest.

5.3 Atmospheric tomography in the wavelet domain

In this section we derive the wavelet discretization of the fitting term in equation (4.8), i.e., we discretize the tomography operator $\mathbf{A}$ in the wavelet basis. Although this discretization is asymptotically sparse, in practice the operator does not have a numerically efficient representation. In the next Section 5.4 we use the continuous piecewise bilinear basis of finite elements to discretize $\mathbf{A}$ to obtain a computationally efficient algorithm.

5.3.1 Discretization of the atmospheric tomography operator in wavelet domain

The atmospheric tomography operator $\mathbf{A}$ is discretized in the wavelet domain by computing the interaction of the operator between a wavelet function on a single layer and the corresponding Shack-Hartmann measurement. The computation of this operator can be reduced to one dimensional evaluations of one dimensional wavelet functions, which we discuss below.

First, observe that $\mathbf{A}$ in (4.2) has an equivalent block component representation,

$$\mathbf{A} = \begin{bmatrix} \Gamma_{11}^{\text{LGS}} & \cdots & \Gamma_{1L}^{\text{LGS}} \\ \vdots & & \vdots \\ \Gamma_{G1}^{\text{NGS}} & \cdots & \Gamma_{GL}^{\text{NGS}} \end{bmatrix}, \tag{5.13}$$

where $\Gamma_{g\ell}^{\text{LGS}}$ and $\Gamma_{g'\ell}^{\text{NGS}}$ are Shack-Hartmann operators in an LGS and an NGS directions $g = 1, ..., G_{\text{LGS}}$ and $g' = G_{\text{LGS}} + 1, ..., G$, respectively, defined directly on the layer $\ell = 1, ..., L$.

This representation of $\mathbf{A}$ follows from the fact that the projection and the Shack-Hartmann operators in equations (4.1) are interchangeable,

$$\Gamma_g P_g^{\text{LGS}} \boldsymbol{\phi} = \Gamma_{g1}^{\text{LGS}} \phi_1 + ... + \Gamma_{gL}^{\text{LGS}} \phi_L, \tag{5.14}$$

$$\Gamma_{g'} P_{g'}^{\text{NGS}} \boldsymbol{\phi} = \Gamma_{g'1}^{\text{NGS}} \phi_1 + ... + \Gamma_{g'L}^{\text{NGS}} \phi_L, \tag{5.15}$$

for $1 \leq g \leq G_{\text{LGS}} < g' \leq G$. Here, projection appears in the form of the summation of the SH-WFS measurements and the cone effect is taken into account by scaling the domain on which the operator $\Gamma_{g\ell}^{\text{LGS}}$ is applied. We discuss the operators $\Gamma_{g\ell}^{\text{LGS}}$ and $\Gamma_{g'\ell}^{\text{NGS}}$ below.

Let us denote the domain observed by the WFS in direction $\theta_g = (\theta_g^x, \theta_g^y, 1)$ of the LGS g at layer ℓ by

$$\Omega_{g\ell}^{\text{LGS}} := \left(1 - \frac{h_\ell}{H}\right) \Omega + (\theta_g^x, \theta_g^y) h_\ell \subseteq \mathbb{R}^2 \tag{5.16}$$

and the domain observed by the WFS in direction $\theta_{g'} = (\theta_{g'}^x, \theta_{g'}^y, 1)$ of the NGS g' at layer ℓ by

$$\Omega_{g'\ell}^{\text{NGS}} := \Omega + (\theta_{g'}^x, \theta_{g'}^y) h_\ell \subseteq \mathbb{R}^2. \tag{5.17}$$

Here, $\Omega \subseteq \mathbb{R}^2$ is the union of subapertures at the telescope pupil (2.34) and H is the finite altitude of the LGS. Note how the LGS domain is scaled according to the cone effect. These domains are obtained from the definition of the corresponding projection operators (2.45) and (2.41).

The turbulence layer ϕ_ℓ, which is defined on Ω_ℓ, see (5.10), is observed by all WFSs. Hence, the layer domain Ω_ℓ must fulfill the condition

$$\left(\bigcup_{g=1}^{G_{\text{LGS}}} \Omega_{g\ell}^{\text{LGS}} \right) \cup \left(\bigcup_{g'=G_{\text{LGS}}+1}^{G} \Omega_{g'\ell}^{\text{NGS}} \right) \subseteq \Omega_\ell, \tag{5.18}$$

i.e., all observed regions must be contained within the domain of the turbulence layer.

The operators $\Gamma_{g\ell}^{\mathrm{LGS}}$ and $\Gamma_{g'\ell}^{\mathrm{NGS}}$ are defined as Shack-Hartmann operators (2.37) on the projected domains $\Omega_{g\ell}^{\mathrm{LGS}}$ and $\Omega_{g'\ell}^{\mathrm{NGS}}$, respectively. Let us examine how the matrix $\Gamma_{g\ell}^{\mathrm{LGS}}$ is computed; $\Gamma_{g'\ell}^{\mathrm{NGS}}$ associated to an NGS is defined analogously.

The LGS operator consists of a concatenation of two components,

$$\Gamma_{g\ell}^{\mathrm{LGS}} := ((\Gamma_{g\ell}^{\mathrm{LGS}})^x, (\Gamma_{g\ell}^{\mathrm{LGS}})^y), \tag{5.19}$$

associated to the x- and the y-slopes. Each component has a matrix representation,

$$\begin{aligned}(\Gamma_{g\ell}^{\mathrm{LGS}})_{\gamma\lambda}^{x} &:= ((\Gamma_{g\ell}^{\mathrm{LGS}})^x \psi_\lambda)_\gamma, \\ (\Gamma_{g\ell}^{\mathrm{LGS}})_{\gamma\lambda}^{y} &:= ((\Gamma_{g\ell}^{\mathrm{LGS}})^y \psi_\lambda)_\gamma,\end{aligned} \tag{5.20}$$

where ψ_λ is a wavelet on layer ℓ with a global index $\lambda = 0, ..., 2^{2J_\ell} - 1$ and $\gamma \in \mathcal{M}$ is a global sub-aperture index, where $\mathcal{M}$ is the mask, see (2.35). We focus only on the x-component, as the y-component is defined similarly.

The scaling and wavelet functions (3.109) and (3.110), respectively, have a multiplicative representation $f(x, y) = f_1(x)f_2(y)$. For a continuously differentiable function f of that form, the value of the average x-slope in subaperture $(\Omega_{g\ell}^{\mathrm{LGS}})_{ij}$ is given by

$$\begin{aligned}((\Gamma_{g\ell}^{\mathrm{LGS}})^x f)_{ij} &= d^{-2} \int\limits_{(\Omega_{g\ell}^{\mathrm{LGS}})_{ij}} \frac{\partial f}{\partial x}(x, y)\mathrm{d}(x, y) \\ &= d^{-2} \int\limits_{x_i}^{x_{i+1}} \int\limits_{y_i}^{y_{i+1}} \frac{\partial f}{\partial x}(x, y)\mathrm{d}x\mathrm{d}y \\ &= d^{-2} \left(f_1(x_{i+1}) - f_1(x_i)\right) \int\limits_{y_i}^{y_{i+1}} f_2(y)\mathrm{d}y,\end{aligned} \tag{5.21}$$

where the domain of the projected subaperture with index $(i, j) \in \mathcal{M}$ is given by $(\Omega_{g\ell}^{\mathrm{LGS}})_{ij} = (x_i, x_{i+1}) \times (y_i, y_{i+1})$ and its area by $d^2 = (x_{i+1} - x_i)(y_{i+1} - y_i)$. Note that these values may be different for each guide star direction g and layer ℓ.

Hence, the assembly of the matrices $\Gamma_{g\ell}^{\mathrm{LGS}}$ and $\Gamma_{g'\ell}^{\mathrm{NGS}}$ can be reduced to computing the differences and one dimensional integrals of the one dimensional scaling and wavelet functions over the edges of the projected subapertures.

This gives rise to the matrix form of the operator $\mathbf{A}$ in the wavelet domain, which we denote by $\widetilde{\mathbf{A}}$. By combining this representation with the results from the previous section, we obtain the discretization of equation (4.8) in the wavelet domain,

$$(\widetilde{\mathbf{A}}^T \mathbf{C}_{\boldsymbol{\eta}}^{-1} \widetilde{\mathbf{A}} + \alpha \mathbf{D})\mathbf{c} = \widetilde{\mathbf{A}}^T \mathbf{C}_{\boldsymbol{\eta}}^{-1} \mathbf{s}, \tag{5.22}$$

where the components of the equation are defined as follows. The matrix $\widetilde{\mathbf{A}}^T$ is the transpose of $\widetilde{\mathbf{A}}$, $\mathbf{C}_{\boldsymbol{\eta}}^{-1}$ is the inverse of the measurement noise covariance matrix (4.7) and $\mathbf{D}$ is the approximation of the inverse covariance of the turbulence layers in the wavelet domain (5.11). Further, we introduce a scalar parameter α to tune the method. Parameter α is relevant since our approximation of the inverse covariance of the prior (5.12) is an equivalence instead of an equality. The vector $\mathbf{c} = (c_\ell)_{\ell=1}^{L}$ is the concatenation of all

wavelet coefficients of all turbulence layers and the vector $\mathbf{s} = (s_g)_{g=1}^{G}$ is the concatenation of all SH-WFS measurements from all guide star directions.

Note that, as discussed in Section 2.5, the circular shape and central obstruction of the telescope pupil are handled by eliminating those equations from the linear system (4.1), for which no measurement data are available, i.e., using only the measurements from the active subapertures.

In Section 9.1.2 we discuss the sparsity of the matrix $\widetilde{\mathbf{A}}$ and confirm that it does not have a numerically efficient representation.

5.3.2 Handling the tip-tilt uncertainty

The laser guide star measures the incorrect tip-tilt, see Section 2.6.2 for more details. The global tip and tilt of the wavefront are modeled by the average slope over the active subapertures. There are several tip-tilt correction methods that can be applied, such as a split tomography approach [23] or a noise-weighted approach [12, 77].

We use a more straightforward approach, in which we remove the incorrect tip-tilt component in the LGS measurements by modifying equation (4.1) for the LGS directions $g = 1, ..., G_{\text{LGS}}$ to

$$(I - T)s_g = (I - T)\Gamma_g P_g^{\text{LGS}} \boldsymbol{\phi}, \tag{5.23}$$

in which the incorrect component of the tip-tilt is removed from both sides of the equation. The operator T in this setting is an orthonormal projection into the tip and tilt measurement space and I is the identity matrix.

That is, T is defined as a mapping onto the space spanned by the two tip and tilt measurement vectors of dimension $2|\mathcal{M}|$,

$$t^x = \begin{pmatrix} e \\ 0 \end{pmatrix} \qquad \text{and} \qquad t^y = \begin{pmatrix} 0 \\ e \end{pmatrix}, \tag{5.24}$$

which are defined via a vector of ones $e = (1, ..., 1)^T$ and a zero vector, both of dimension $|\mathcal{M}|$. Here, $\mathcal{M}$ is the mask of the SH-WFS associated with the LGS, see (2.35). The projection T is thus given by

$$T = \frac{1}{|\mathcal{M}|} \begin{bmatrix} t^x & t^y \end{bmatrix} \begin{bmatrix} t^x & t^y \end{bmatrix}^T. \tag{5.25}$$

Other way of stating (5.23) is to say that we use

$$\widehat{C}_g^{-1} = (I - T)\widetilde{C}_g^{-1}(I - T) \tag{5.26}$$

as the inverse covariance matrix instead of $\widetilde{C}_g^{-1}$ in (4.7) for $1 \leq g \leq G_{\text{LGS}}$, coupled with the original equations (4.1).

The equation that takes the incorrect tip-tilt of the LGS into account is stated as

$$(\widetilde{\mathbf{A}}^T \widehat{\mathbf{C}}_{\boldsymbol{\eta}}^{-1} \widetilde{\mathbf{A}} + \alpha \mathbf{D})\mathbf{c} = \widetilde{\mathbf{A}}^T \widehat{\mathbf{C}}_{\boldsymbol{\eta}}^{-1} \mathbf{s}, \tag{5.27}$$

with the noise covariance matrix given by

$$\widehat{\mathbf{C}}_{\boldsymbol{\eta}} = \text{diag}(\widehat{C}_1, ..., \widehat{C}_{G_{\text{LGS}}}, \widetilde{C}_{G_{\text{LGS}}+1}, ..., \widetilde{C}_G), \tag{5.28}$$

and other components defined as for (5.22). Here, $\widehat{C}_g$ for $1 \le g \le G_{\text{LGS}}$ is the LGS noise covariance as above and $\widetilde{C}_{g'}$ for $G_{\text{LGS}} < g' \le G$ is the NGS noise covariance given by (2.43).

We conclude this section with a short remark regarding the matrix form of a Shack-Hartmann operator $\Gamma^{\text{NGS}}_{g'\ell}$ associated to a tip-tilt sensor. A tip-tilt sensor is a low order WFS, see Figure 2.11.

Using the identity,

$$\int\limits_{(\Omega^{\text{NGS}}_{g'\ell})_{ij}} \nabla f(x,y)\mathrm{d}(x,y) = \int\limits_{\partial(\Omega^{\text{NGS}}_{g'\ell})_{ij}} f(x,y)n(x,y)\mathrm{d}s(x,y), \tag{5.29}$$

we observe that the computation of the matrix form of the operator requires only the evaluations of the wavelets along the boundary of the domain. Here, $\nabla f = (\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y})$ is the gradient of f, $n = (n_x, n_y)$ is the outer normal vector on the boundary, $\partial(\Omega^{\text{NGS}}_{g'\ell})_{ij}$ is the boundary of the domain and the integral on the right is a surface integral.

This type of boundary evaluation for a low order tip-tilt sensor is usually numerically cheaper to evaluate than (5.21).

5.4 Dual domain discretization

Asymptotically, the wavelet discretization of the tomography operator $\mathbf{A}$ has a sparse representation. As the wavelet scale index increases, the support of the wavelet function shrinks. Hence, the operator tends to be more sparse if more wavelet scales are used. However, in practice the number of scales is limited and the operator is not sufficiently sparse for an efficient numerical representation.

The operator has a more efficient representation in a finite element basis. Due to the local support of these basis functions, the interaction between the elements and the corresponding Shack-Hartmann measurements is small. Hence, sparsity of the representation is promoted. Further efficiency in the finite element domain is obtained by separating the projection and the Shack-Hartmann steps into two sequential operations. These concepts are utilized in many iterative methods for atmospheric tomography [12, 75].

The block representation of the operator $\mathbf{A}$ in (4.1) is of the form

$$\mathbf{A} = \begin{bmatrix} \Gamma_1 & & \\ & \ddots & \\ & & \Gamma_G \end{bmatrix} \begin{bmatrix} P^{\text{LGS}}_{11} & \cdots & P^{\text{LGS}}_{1L} \\ \vdots & & \vdots \\ P^{\text{NGS}}_{G1} & \cdots & P^{\text{NGS}}_{GL} \end{bmatrix}. \tag{5.30}$$

Here, Γ_g is the Shack-Hartmann operator at the telescope aperture (2.37) and the operators $P^{\text{LGS}}_{g\ell}$ and $P^{\text{NGS}}_{g'\ell}$, for $1 \le g \le G_{\text{LGS}} < g' \le G$ and $1 \le \ell \le L$, are the block-components of the projection operators P^{LGS}_{g} and $P^{\text{NGS}}_{g'}$ in (2.45) and (2.41), respectively. The projection operators have a block-component representation,

$$\begin{aligned} P^{\text{LGS}}_{g} &= \begin{bmatrix} P^{\text{LGS}}_{g1} & \cdots & P^{\text{LGS}}_{gL} \end{bmatrix}, \\ P^{\text{NGS}}_{g'} &= \begin{bmatrix} P^{\text{NGS}}_{g'1} & \cdots & P^{\text{NGS}}_{g'L} \end{bmatrix}, \end{aligned} \tag{5.31}$$

where each block acts on a turbulence layer ϕ_ℓ of the vector $\boldsymbol{\phi} = (\phi_1, ..., \phi_L)$. We explain these block-components below.

Projection blocks

Let Ω be the domain of subapertures at the telescope pupil (2.34), Ω_ℓ be the domain (5.10) on which the turbulence layer ϕ_ℓ is defined, and $\Omega_{g\ell}^{\mathrm{LGS}}$ and $\Omega_{g\ell}^{\mathrm{NGS}}$ be the observed domains (5.16) and (5.17) in direction g on turbulence layer ℓ, respectively.

The operator $P_{g\ell}^{\mathrm{LGS}}$ associated to an LGS direction acts as a restriction of the turbulence layer ϕ_ℓ from Ω_ℓ to $\Omega_{g\ell}^{\mathrm{LGS}}$, followed by a shift and rescaling (due to the cone effect) of the function to Ω.

Similarly, the operator $P_{g'\ell}^{\mathrm{NGS}}$ associated to an NGS direction is a restriction of the turbulence layer ϕ_ℓ from Ω_ℓ to $\Omega_{g'\ell}^{\mathrm{NGS}}$, followed by a shift of the function to Ω.

Dual-domain discretization

In the finite element discretization we utilize continuous piecewise bilinear functions to represent the incoming wavefronts and layers. The domains on which these two kinds of functions are discretized are defined below.

To define the piecewise bilinear wavefront functions, we use a square mesh with equidistant spacing on Ω. The mesh consists of $(n_s+1)^2$ corner points of the subapertures, see equations (2.31)-(2.34). Continuous piecewise bilinear functions on this grid were introduced in Section 2.5. We denote the piecewise bilinear wavefront functions by φ_g and the associated vector of the function values at the grid points by the same variable $\varphi_g = (\varphi_{g,ij})_{i,j=0}^{n_s}$. The Shack-Hartmann operator Γ_g for a continuous piecewise bilinear wavefront function is defined in (2.38).

To define the piecewise bilinear layer functions, we use a square mesh with equidistant spacing on Ω_ℓ. The mesh consists of 2^{2J_ℓ} points arranged on a square grid with equidistant spacing, given by

$$\{(x_i, y_j) : 0 \le i, j < 2^{J_\ell}\}, \tag{5.32}$$

with

$$x_i := \delta_\ell i - \xi_\ell^x, \tag{5.33}$$
$$y_j := \delta_\ell j - \xi_\ell^y, \tag{5.34}$$

where $\xi_\ell = (\xi_\ell^x, \xi_\ell^y) \in \mathbb{R}^2$ is the shift of coordinates in (5.10) and δ_ℓ is the scaling constant at layer ℓ. We choose to use this grid of 2^{2J_ℓ} points in order to define the wavelet transform on the layer.

The grid induces square finite elements $[x_i, x_{i+1}] \times [y_j, y_{j+1}]$ for $0 \le i, j < 2^{J_\ell}$. On each element we introduce a bilinear function which is uniquely defined by its four values at the corner points, $\phi_{\ell,ij}, \phi_{\ell,i+1,j}, \phi_{\ell,i,j+1}$ and $\phi_{\ell,i+1,j+1}$. In this way we obtain a continuous piecewise bilinear function on Ω_ℓ, which is uniquely defined via its values at the grid points, $(\phi_{\ell,ij})_{i,j=0}^{2^{J_\ell}-1}$. In the following use the same notation to denote the piecewise bilinear layer function ϕ_ℓ and the vector of its values at the grid points $\phi_\ell = (\phi_{\ell,ij})_{i,j=0}^{2^{J_\ell}-1}$.

The link between the finite element discretization of the layer and the wavelet discretization (5.8) is established via the discrete wavelet transform (DWT). The relation between the approximation coefficients and the nodal values of the function is given by

$$a_{J,ij} = \delta_\ell \phi_\ell(\delta_\ell i - \xi_\ell^x, \delta_\ell j - \xi_\ell^y) = \delta_\ell \phi_{\ell,ij}, \tag{5.35}$$

for $0 \leq i, j < 2^{J_\ell}$. Thus, the wavelet coefficients and the nodal values of the continuous piecewise bilinear functions on layer ℓ are related via the DWT,

$$c_\ell = \delta_\ell W \phi_\ell, \tag{5.36}$$

where W is the DWT on layer ℓ. Conversely, the inverse DWT maps wavelet coefficients back to the nodal values,

$$\phi_\ell = \delta_\ell^{-1} W^{-1} c_\ell. \tag{5.37}$$

Observe that the transpose of (5.37) coincides with (5.36), up to the scaling constant,

$$(\delta_\ell^{-1} W^{-1})^T = \delta_\ell^{-1} W. \tag{5.38}$$

Bilinear interpolation

Grid points of domains $\Omega_{g\ell}^{\text{LGS}}$ and $\Omega_{g'\ell}^{\text{NGS}}$ are defined by the projection of the grid points of Ω to the layer, according to (5.16) and (5.17), respectively. We illustrate such a projected grid in Figure 5.1.

In general, the grid of the projected domain will not align with the layer grid Ω_ℓ. Hence, the component of the incoming wavefront at layer ℓ in direction g is found by bilinear interpolation of the layer nodal values ϕ_ℓ on Ω_ℓ onto the grid defined by $\Omega_{g\ell}^{\text{LGS}}$ or $\Omega_{g\ell}^{\text{NGS}}$.

Thus, in the discrete setting $P_{g\ell}^{\text{LGS}}$ and $P_{g'\ell}^{\text{NGS}}$ in (5.30) are bilinear interpolation operators. Below, we define bilinear interpolation as a concatenation of two linear interpolation operators.

Let use denote by

$$I(x; a, b, f_a, f_b) := \frac{(f_b - f_a)}{(b-a)}(x-a) + f_a \tag{5.39}$$

a linear interpolation operator at $x \in [a, b]$ of a linear function with boundary values f_a and f_b at a and b, respectively.

Further, let us fix a layer $\ell = 1, ..., L$, a direction $g = 1, ..., G$ and a point (x_i, x_j) on the mesh grid of Ω, with $0 \leq i, j \leq n_s$. The projected point on layer ℓ in an LGS direction g is given by a grid point of $\Omega_{g\ell}^{\text{LGS}}$,

$$(x, y) = \left(1 - \frac{h_\ell}{H}\right)(x_i, x_j) + (\theta_g^x, \theta_g^y) h_\ell. \tag{5.40}$$

Due to the condition (5.18), there exist indices $0 \leq p, q < 2^{J_\ell}$, such that

$$(x, y) \in [x_p, x_{p+1}] \times [y_q, y_{q+1}]. \tag{5.41}$$

Then, the bilinear interpolation of ϕ_ℓ onto the point (x, y) is given by the interpolation in the y-direction

$$(P_{g\ell}^{\text{LGS}} \phi_\ell)_{ij} := I(y; y_q, y_{q+1}, t_1, t_2), \tag{5.42}$$

where the intermediate values t_1, t_2 are given by the interpolation in the x-direction,

$$\begin{aligned} t_1 &= I(x; x_p, x_{p+1}, \phi_{\ell,p,q}, \phi_{\ell,p+1,q}), \\ t_2 &= I(x; x_p, x_{p+1}, \phi_{\ell,p,q+1}, \phi_{\ell,p+1,q+1}). \end{aligned} \tag{5.43}$$

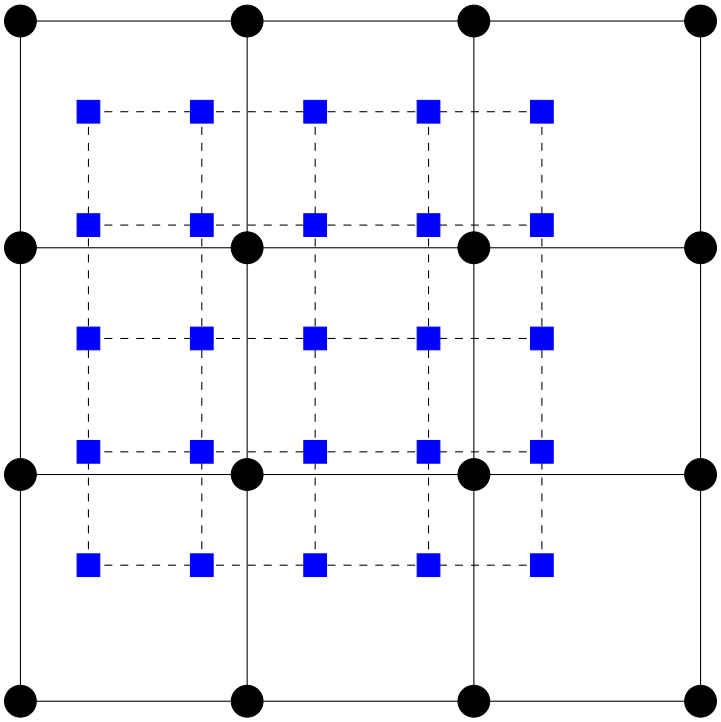

Figure 5.1: Layer grid Ω_ℓ (black circles) depicted with $2^{2J_\ell} = 2^4$ points and equidistant grid spacing δ_ℓ. Projected subaperture grid $\Omega_{g\ell}^{\mathrm{LGS}}$ with $n_s^2 = 16$ subapertures and $(n_s + 1)^2 = 25$ corner points on layer ℓ in direction g (blue squares).

See Figure 5.2 for a schematic representation. Alternatively, the same value is obtained by first interpolating in the y-direction and then in the x-direction.

For an NGS, the grid point of $\Omega_{g\ell}^{\mathrm{LGS}}$ is given by

$$(x, y) = (x_i, x_j) + (\theta_g^x, \theta_g^y)h_\ell, \tag{5.44}$$

for $0 \leq i, j \leq n_s$ and the bilinear interpolation operation is defined analogously.

Thus, the incoming wavefront at a subaperture corner point with index (i, j), $0 \leq i, j \leq n_s$, towards the LGS g or NGS g' is given by the sum of interpolation operations through all layers,

$$\varphi_{g,ij} = \sum_{\ell=1}^{L} (P_{g\ell}^{\mathrm{LGS}} \phi_\ell)_{ij} \tag{5.45}$$

or

$$\varphi_{g',ij} = \sum_{\ell=1}^{L} (P_{g'\ell}^{\mathrm{NGS}} \phi_\ell)_{ij}, \tag{5.46}$$

respectively. This kind of propagation in the piecewise bilinear domain is also called *ray-tracing*.

Observe that if layer $\ell = 1$, i.e., the ground layer, is located at the altitude of $h_1 = 0$ meters, the interpolation step can be avoided by aligning the layer grid with the wavefront grid. At the altitude of $h_1 = 0$ meters, all projected domains coincide with the wavefront domain $\Omega = \Omega_{g1}^{\mathrm{LGS}} = \Omega_{g'1}^{\mathrm{NGS}}$ for $1 \leq g \leq G_{\mathrm{LGS}} < g' \leq G$. By aligning the layer grid with the wavefront grid, the wavefront values at the subaperture corner points are given by

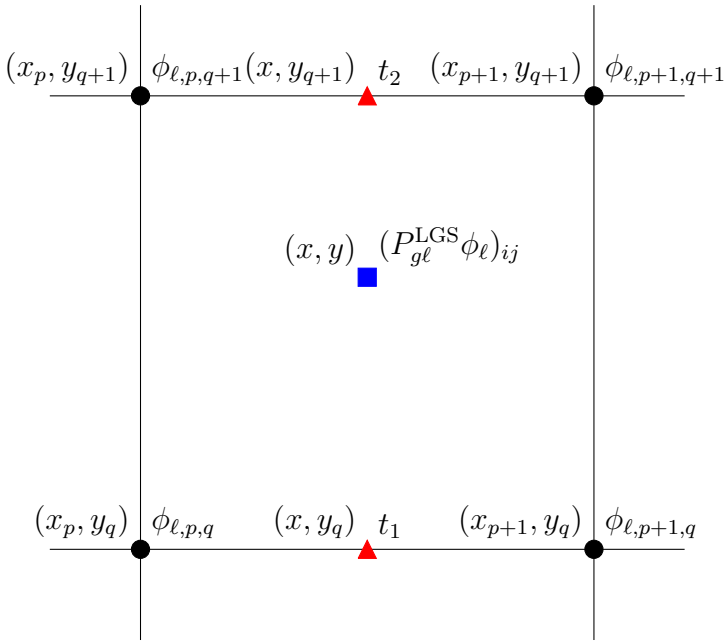

Figure 5.2: Bilinear interpolation of ϕ_ℓ onto the point (x, y). Coordinates (left of the point) and function values (right of the point) are labeled. Values of ϕ_ℓ are available at black circles, temporary values are computed at red triangles and the final interpolated value is given at the blue square.

the grid values of the layer $\varphi_{g,ij} = \phi_{1,ij}$ for $0 \leq i, j \leq n_s$ and $1 \leq g \leq G$. Note that it is possible that the layer function is discretized with more points than the wavefront function, i.e., $(n_s + 1)^2 \leq 2^{2J_1}$. In this case the layer values $\phi_{1,ij}$ for $n_s < i < 2^{J_1}$ or $n_s < j < 2^{J_1}$ are outside of the region of interest.

Dual-domain problem formulation

Altogether, we obtain a finite element discretization of the operator $\mathbf{A}$, which we denote by $\widehat{\mathbf{A}}$. The operator maps the vector of piecewise bilinear nodal points of all layers to SH-WFS measurements of all sensors.

Further, we define the mapping between the finite element and the wavelet domains at all layers by $\mathbf{W}$,

$$\mathbf{W} = \mathrm{diag}(\delta_1 W, ..., \delta_L W), \tag{5.47}$$

where W is the DWT. The inverse mapping is given by a concatenation of inverse DWT operations at each layer,

$$\mathbf{W}^{-1} = \mathrm{diag}(\delta_1^{-1} W^{-1}, ..., \delta_L^{-1} W^{-1}). \tag{5.48}$$

For convenience we write $\mathbf{W}^{-T} = (\mathbf{W}^{-1})^T = \mathrm{diag}(\delta_1^{-1} W, ..., \delta_L^{-1} W)$.

Thus, the dual domain discretization of (4.9) is formulated via

$$(\mathbf{W}^{-T}\widehat{\mathbf{A}}^T\widehat{\mathbf{C}}_{\boldsymbol{\eta}}^{-1}\widehat{\mathbf{A}}\mathbf{W}^{-1} + \alpha\mathbf{D})\mathbf{c} = \mathbf{W}^{-T}\widehat{\mathbf{A}}^T\widehat{\mathbf{C}}_{\boldsymbol{\eta}}^{-1}\mathbf{s}, \tag{5.49}$$

where the operator $\widehat{\mathbf{A}}$ is discretized in the bilinear domain and $\widehat{\mathbf{A}}^T$ is its transpose, $\mathbf{D}$ is the diagonal approximation of the inverse covariance of the prior in the wavelet domain,

and the operators $\mathbf{W}^{-1}$ and $\mathbf{W}^{-T}$ are mappings between the two domains. We use the inverse noise covariance $\widehat{\mathbf{C}}_{\boldsymbol{\eta}}^{-1}$ in (5.28), which takes tip-tilt uncertainty of the LGS into account.

For convenience, we denote the left-hand side operator by

$$\mathbf{M} := \mathbf{W}^{-T}\widehat{\mathbf{A}}^T\widehat{\mathbf{C}}_{\boldsymbol{\eta}}^{-1}\widehat{\mathbf{A}}\mathbf{W}^{-1} + \alpha\mathbf{D} \tag{5.50}$$

and the right-hand side vector by

$$\mathbf{b} := \mathbf{W}^{-T}\widehat{\mathbf{A}}^T\widehat{\mathbf{C}}_{\boldsymbol{\eta}}^{-1}\mathbf{s}. \tag{5.51}$$

Observe that the matrix $\mathbf{M}$ is symmetric and positive definite: let $\mathbf{c}$ be a non-zero vector, then

$$(\mathbf{M}\mathbf{c}, \mathbf{c}) = \|\widehat{\mathbf{A}}\mathbf{W}^{-1}\mathbf{c}\|^2_{\widehat{\mathbf{C}}_{\boldsymbol{\eta}}^{-1}} + \alpha(\mathbf{D}\mathbf{c}, \mathbf{c}) > 0, \tag{5.52}$$

since the first term of the sum is non-negative and the second term is strictly positive for all $\alpha > 0$.

In the numerics section, we show that this representation is sparse and efficient. Moreover, the computational complexity of applying $\mathbf{M}$ or computing $\mathbf{b}$ scale linearly with the dimension of the vectors, which in combination with other techniques leads to a method that scales globally as $\mathcal{O}(n)$.

Chapter 6

Acceleration techniques

6.1 Improving convergence

We reconstruct the turbulence layers of the atmosphere by solving equation (5.49) in the wavelet domain using the PCG Algorithm 3.2. A successful iterative method must satisfy the following two criteria: the operator(s) in the algorithm must have a numerically cheap representation and the number of iterations to achieve a solution of sufficient quality must be low.

In Chapter 5 we derived a sparse representation of the forward operator $\mathbf{M}$, which needs to be applied once per iteration of the algorithm. To improve convergence of the method in terms of the number of iterations, several techniques are applied.

First, we use a good initial guess for the atmosphere. The turbulent atmosphere evolves rapidly, however a solution from a previous time-step, see Section 2.7, is a good initial guess for the problem at the current time step. This technique, called *warm restart*, is widely used for iterative methods in AO [15].

Next, we are able to improve the initial guess from the warm restart by solving a small sub-problem, which is based on the multi-scale structure of wavelets. We describe this approach in Section 6.3.

Finally, we improve convergence by the use of an efficient preconditioner. The preconditioner is based on the heuristic considerations about the temporal evolution of the atmosphere. We modify a Jacobi preconditioner for the problem (5.49) to obtain a different weighting of the low and high frequency regimes. Apart from reducing the number of iterations, the benefit of using such an approach is an increased robustness and stability of the overall method. Moreover, for our method this preconditioner can be formulated as a diagonal matrix, which is very efficient regarding the computational cost. We describe this frequency-dependent preconditioner in Section 6.2.

6.2 Frequency-dependent preconditioner

Using the short-hand notation, equation (5.49) is given by

$$\mathbf{M}\mathbf{c} = \mathbf{b}. \tag{6.1}$$

The corresponding Jacobi preconditioner to this problem is given by the diagonal of $\mathbf{M}$,

$$\mathbf{\Lambda} := \mathrm{diag}(\mathbf{M}) = \mathrm{diag}(\mathbf{W}^{-T}\widehat{\mathbf{A}}^T\widehat{\mathbf{C}}_{\boldsymbol{\eta}}^{-1}\widehat{\mathbf{A}}\mathbf{W}^{-1}) + \alpha\mathbf{D}. \tag{6.2}$$

Note that $\mathbf{D}$ is a diagonal matrix. The Jacobi preconditioner can be seen as a mixed preconditioner, applied from the left and the right, with factorization (3.141).

Our choice of the frequency-dependent preconditioner is obtained by modifying $\mathbf{\Lambda}$ according to

$$\mathbf{\Lambda}_\tau := \mathrm{diag}(\mathbf{W}^{-T}\widehat{\mathbf{A}}^T\widehat{\mathbf{C}}_{\boldsymbol{\eta}}^{-1}\widehat{\mathbf{A}}\mathbf{W}^{-1}) + \alpha\,\mathrm{max}(\mathbf{D}, \tau\mathbf{I}), \tag{6.3}$$

where τ is a non-negative scalar parameter and $\mathbf{I}$ is the identity. Above, the maximum is taken component-wise.

Low and high wavelet scales correspond to low and high frequency regimes of the layers, respectively. Since the values of $\mathbf{D}$ increase fast with respect to the scales, by increasing the parameter τ we are able to affect the level of damping for the lower frequencies in the preconditioner. In practice, the standard Jacobi preconditioner will damp the high scales too much with respect to the low scales. The preconditioner above introduces a way to balance the standard Jacobi preconditioner with the non-preconditioned CG in a scale-dependent way.

Note that for $\tau = 0$ one arrives at the standard Jacobi preconditioner, whereas for a very large choice of $\tau \gg 1$, the identity-term dominates (6.3) and the preconditioner is approximately $\tau\mathbf{I}$. A scaled identity preconditioner has the same effect as the identity preconditioner, hence the algorithm reduces to the standard CG in this case. We demonstrate this property numerically in Section 8.5.2.

Above, τ is a global thresholding parameter that damps all scales at all layers simultaneously. Instead of using a global thresholding parameter for all layers, we can also control the thresholding on each layer separately according to the wavelet scale. We define a set of preconditioners

$$\mathbf{\Lambda}_j := \mathrm{diag}(\mathbf{W}^{-T}\widehat{\mathbf{A}}^T\widehat{\mathbf{C}}_{\boldsymbol{\eta}}^{-1}\widehat{\mathbf{A}}\mathbf{W}^{-1}) + \alpha\mathbf{D}_j, \tag{6.4}$$

where

$$\mathbf{D}_j := \mathrm{diag}(\mathrm{max}(\tau_{1,j}I, D_1), ..., \mathrm{max}(\tau_{L,j}I, D_L)), \tag{6.5}$$

and the parameters $\tau_{\ell,j}$ are given by the value of D_ℓ at scale j, i.e., by

$$\tau_{\ell,j} := \frac{1}{c(h_\ell)}\left(\kappa_0^{11/3} + 2^{(11/3)j}\right), \tag{6.6}$$

see (5.9). Above, the maximum is taken component-wise and I is the identity. Note that unlike in (6.3) where τ takes values in a continuous interval, in this representation we have a discrete set of parameter choices $\tau_{\ell,j}$, which are controlled by the thresholding scale index $j = 0, ..., J_\ell - 1$ on each layer.

In Figure 6.1 we plot the diagonal values of the preconditioner $\mathbf{\Lambda}_j$ for $j = J_\ell - 3$, $J_\ell - 2$ and $J_\ell - 1$, where $J_1 = 7$ and $J_2 = J_3 = 6$ scales in a $L = 3$ layer turbulence model. The effects of the choice of thresholding scale j on the reconstruction are illustrated in Section 8.4.5.

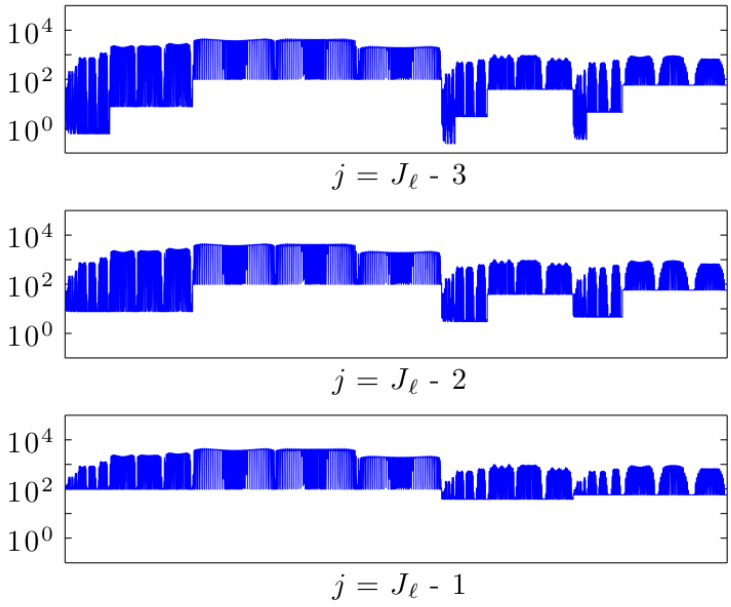

Figure 6.1: The effect of thresholding at scale j to the preconditioner is illustrated. The values on the diagonal of $\mathbf{\Lambda}_j$ are plotted for three thresholding scales $j = J_\ell - 3$, $J_\ell - 2$ and $J_\ell - 1$. The y-axis corresponds to the values on a logarithmic scale. The x-axis stands for the global index of the wavelet coefficients λ. These are ordered from left to right starting from the ground layer up to the layer with highest altitude; for each layer the values are plotted from left to right starting with the coarsest scale.

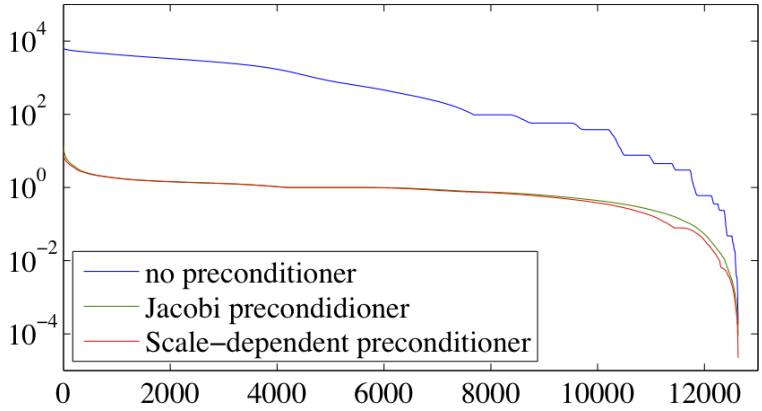

Figure 6.2: Logarithmic plot of the eigenvalue decomposition of $\mathbf{M}$ (blue curve), $\mathbf{\Lambda}^{-1/2}\mathbf{M}\mathbf{\Lambda}^{-1/2}$ (green curve) and $\mathbf{\Lambda}_j^{-1/2}\mathbf{M}\mathbf{\Lambda}_j^{-1/2}$ with $j = J_\ell - 2$ (red curve). The Jacobi and the scale dependent preconditioner enforce clustering of eigenvalues around 1. Eigenvalues decay towards 0.

In Figure 6.2 we plot the eigenvalue decomposition of the unpreconditioned left-hand side operator $\mathbf{M}$, of the Jacobi preconditioned operator $\mathbf{\Lambda}^{-1/2}\mathbf{M}\mathbf{\Lambda}^{-1/2}$ and of the scale-dependent preconditioned operator $\mathbf{\Lambda}_j^{-1/2}\mathbf{M}\mathbf{\Lambda}_j^{-1/2}$ with $j = J_\ell - 2$ of the $L = 3$ layer turbulence model as above.

The characteristic effect of an ill-posed problem is observed in the decay of the eigenvalues towards 0, see Section 3.1. The eigenvalues of the Jacobi and the scale-dependent preconditioner cluster around one, see Section 3.6.

6.3 Multiscale methods

Convergence of any iterative scheme can be improved by finding a better initial value. Iterative methods in AO benefit from the possibility of the warm restart technique, i.e., using the solution generated in the previous loop iteration. In our algorithm we implement an additional step, where the initial guess from the warm restart is improved at the coarse wavelet scales.

Due to the multi-scale structure of wavelets the problem (5.49) can be posed at the coarse scales by truncating the wavelet decomposition of the unknown. To keep the computational cost low, we restrict this intermediate problem to the ground layer $\ell = 1$. This is motivated by the empirical observation that the highest optical turbulence strength is located near the ground.

We use the notation from Section 5.2 and denote by J_ℓ the number of wavelet scales on layer $\ell = 1, ..., L$. The sub-problem is defined on $\tilde{J}_1$ coarse scales of the ground layer, where $\tilde{J}_1 \leq J_1$.

In the wavelet domain, the restriction from $\sum_{\ell=1}^{L} 2^{2J_\ell}$ wavelet coefficients of the full problem to $2^{2\tilde{J}_1}$ wavelet coefficients of the coarse-scale sub-problem is given by a restriction matrix,

$$\mathbf{R} := \left[\begin{array}{cc} I & 0 \end{array} \right], \tag{6.7}$$

that consists of an identity block and zero block. Correspondingly, the prolongation matrix is given by $\mathbf{R}^T$. The effect of the prolongation matrix is the padding of the coarse scale coefficients at high scales and other layers with zeros. Using the restriction matrix, we define the left-hand side matrix that corresponds to the coarse wavelet scales of the ground layer by

$$\widetilde{M} := \mathbf{R}\mathbf{M}\mathbf{R}^T, \tag{6.8}$$

where $\mathbf{M}$ is the left-hand side operator (5.50). The matrix $\widetilde{M}$ is of dimension $2^{2\tilde{J}_1} \times 2^{2\tilde{J}_1}$. Observe that the matrix for the sub-problem inherits the properties of the full matrix: it is symmetric and positive definite.

Below we describe how the sub-problem is integrated in the solution method. Let us denote the initial guess of the full problem that comes from the warm restart by $\mathbf{c}^{(0)}$. Due to the linearity of $\mathbf{M}$, applying the PCG algorithm (see Algorithm 3.2) to (6.1) with an initial guess $\mathbf{c}^{(0)}$ is equivalent to applying the PCG algorithm to the residual equation

$$\mathbf{M}\mathbf{v} = \mathbf{r}^{(0)}, \tag{6.9}$$

where $\mathbf{r}^{(0)} = \mathbf{b} - \mathbf{M}\mathbf{c}^{(0)}$ is the residual and $\mathbf{v}$ is an unknown correction with a zero initial guess. Consequently, the solution is updated according to

$$\mathbf{c}^{(1)} = \mathbf{c}^{(0)} + \mathbf{v}. \tag{6.10}$$

The purpose of the multi-scale method is to update the initial guess and the corresponding residual by solving the problem on the coarse scales prior to applying the PCG algorithm. To this end, first, the residual vector of the full problem associated to the initial guess is restricted to the lower scales,

$$\tilde{r} = \mathbf{R}\mathbf{r}^{(0)}. \tag{6.11}$$

Then, the problem

$$\widetilde{M}\tilde{v} = \tilde{r} \tag{6.12}$$

is solved at the coarse scales of the ground layer for a correction vector $\tilde{v}$. Finally, the correction vector is prolongated to the full vector size,

$$\mathbf{v} = \mathbf{R}^T\tilde{v}, \tag{6.13}$$

and the initial guess for the PCG algorithm is updated via (6.10) to obtain an improved initial guess $\mathbf{c}^{(1)}$. Additionally, the corresponding residual must be recomputed,

$$\mathbf{r}^{(1)} = \mathbf{b} - \mathbf{M}\mathbf{c}^{(1)} = \mathbf{r}^{(0)} - \mathbf{M}\mathbf{v}. \tag{6.14}$$

The resulting method, which we call the *ground layer multi-scale* (GLMS) method, is summarized in the following algorithm.

Algorithm 6.1: The GLMS algorithm

Input : vector $\mathbf{c}^{(0)}$, residual $\mathbf{r}^{(0)} = \mathbf{b} - \mathbf{M}\mathbf{c}^{(0)}$
Output: vector $\mathbf{c}^{(1)}$, residual $\mathbf{r}^{(1)} = \mathbf{b} - \mathbf{M}\mathbf{c}^{(1)}$

1 $\tilde{r} = \mathbf{R}\mathbf{r}^{(0)}$ `/* restrict residual */`
2 $\tilde{v} = \widetilde{M}^{-1}\tilde{r}$ `/* solve sub-problem */`
3 $\mathbf{v} = \mathbf{R}^T\tilde{v}$ `/* prolongate correction */`
4 $\mathbf{c}^{(1)} = \mathbf{c}^{(0)} + \mathbf{v}$ `/* update initial guess */`
5 $\mathbf{r}^{(1)} = \mathbf{r}^{(0)} - \mathbf{M}\mathbf{v}$ `/* update residual */`

If the dimensions of the matrix $\widetilde{M}$ are sufficiently small, it is computationally reasonable to directly invert it. In this case, step 2 is a matrix-vector multiplication. Note that the update vector $\mathbf{v}$ is of special structure,

$$\mathbf{v} = \begin{bmatrix} \tilde{v} \\ 0 \end{bmatrix}, \tag{6.15}$$

hence, the operation $\mathbf{Mv}$ is typically cheaper than applying $\mathbf{M}$ to an arbitrary vector.

By examining the updated residual, it can be observed that the components corresponding to the coarse scales of the ground layer are determined explicitly with the GLMS method. Let

$$\mathbf{R}' := \begin{bmatrix} 0 & I \end{bmatrix}, \tag{6.16}$$

be the restriction of the wavelet coefficients of the full problem to the $\sum_{\ell=1}^{L} 2^{2J_\ell} - 2^{2\tilde{J}_1}$ coefficients that are not in the coarse wavelet scales of the ground layer. Further, let us denote the blocks of $\mathbf{M}$ by

$$\mathbf{M} = \begin{bmatrix} \widetilde{M} & M_{12} \\ M_{12}^T & M_{22} \end{bmatrix}, \tag{6.17}$$

where $M_{12} = \mathbf{R}\mathbf{M}(\mathbf{R}')^T$ and $M_{22} = \mathbf{R}'\mathbf{M}(\mathbf{R}')^T$. Then,

$$\begin{aligned} \mathbf{r}^{(1)} = \mathbf{b} - \mathbf{M}\mathbf{c}^{(1)} &= \mathbf{r}^{(0)} - \mathbf{M}\mathbf{v} = \\ &= \mathbf{r}^{(0)} - \begin{bmatrix} \widetilde{M} & M_{12} \\ M_{12}^T & M_{22} \end{bmatrix} \begin{bmatrix} \tilde{v} \\ 0 \end{bmatrix} \\ &= \mathbf{r}^{(0)} - \begin{bmatrix} \tilde{r} \\ M_{12}^T \tilde{v} \end{bmatrix} \\ &= \begin{bmatrix} 0 \\ \mathbf{R}' r^{(0)} - M_{12}^T \tilde{v} \end{bmatrix}. \end{aligned} \tag{6.18}$$

Hence, the coarse scale components of the ground layer are computed explicitly. Moreover, computing the residual (step 5 of Algorithm 6.1) is mathematically equivalent to the matrix-vector multiplication $M_{12}^T \tilde{v}$, where the matrix M_{12} is of dimension $2^{2\tilde{J}_1} \times \sum_{\ell=1}^{L} 2^{2J_\ell}$. Whether this explicit computation is numerically cheaper than applying the full operator $\mathbf{M}$ to $\mathbf{v}$ depends on the total number of coefficients of the full problem and on the number of scales used in the GLMS method, $\tilde{J}_1$.

The GLMS method has certain similarities to a multi-grid method for partial differential equations, see, e.g., [4]. Like a multi-grid method, the problem is defined on several grids. However, whereas the goal of a classical multi-grid method is to eliminate the high-frequency error components using smoothing iterations, such as the Jacobi or the Gauss-Seidel iterations, on each grid, the goal of our multi-scale adaptation is to improve the low-frequency solution before PCG is applied to the full problem. This is a well-known strategy for ill-posed problems solved by the CG algorithm on multiple grids, see, e.g., [37]. A multi-grid method MG-PCG has been studied in terms of AO in [26, 24, 76].

The multi-scale approach that we use is by no means the only possible way to define a sub-problem for equation (5.49). For instance, the sub-problem can be defined on the coarse scales of all layers. Also, one can consider the sub-problem by restricting the equation to only one layer, e.g., the ground layer. In the rest of this section we discuss the possible variations of the ground layer multi scale method. Numerical tests with these methods are left for future work.

6.3.1 Multi-scale method on all layers

As a generalization to the ground layer multi-scale method, the sub-problem can be defined for coarse scales of all layers, i.e., for $\tilde{J}_\ell \leq J_\ell$ coarse scales at all layers $\ell = 1, ..., L$. The GLMS method is a special case of this approach with $\tilde{J}_\ell = 0$ for all $\ell > 1$.

In the generalized approach, the restriction matrix $\mathbf{R}$ has a block-diagonal representation,

$$\mathbf{R} = \begin{bmatrix} R_1 & & \\ & \ddots & \\ & & R_L \end{bmatrix} \tag{6.19}$$

with R_ℓ of the form (6.7) and of dimension $2^{2\tilde{J}_\ell} \times 2^{2J_\ell}$ for $\ell = 1, ..., L$.

As a direct consequence, the matrix $\widetilde{M}$ increases in dimension, and hence step 2 of Algorithm 6.1 becomes computationally more expensive. Additionally, the update vector $\mathbf{v}$ loses the special structure (6.15). Therefore, the multiplication $\mathbf{Mv}$ is not numerically cheaper than applying $\mathbf{M}$ to a general vector in step 5 of the algorithm.

6.3.2 Ground layer PCG

GLPCG method

Another generalization of the method goes in the direction of defining the sub-problem on less layers, rather than less wavelet scales of the wavelet decomposition. Specifically, we concentrate on the sub-problem for the ground layer $\ell = 1$, which is the layer with the highest turbulence strength.

We consider the sub-problem for all wavelet scales of the ground layer, $\tilde{J}_1 = J_1$. The restriction operator $\mathbf{R}$ is of the form (6.7) and of dimension $2^{2J_1} \times \sum_{\ell=1}^{L} 2^{2J_\ell}$.

Thus, the atmospheric tomography reduces to equations only for the ground layer, i.e., the concatenation of the prolongation operation and the wavelet transform reduces to

$$\mathbf{W}^{-1}\mathbf{R}^T = \begin{bmatrix} \delta_1^{-1}W^{-1} & & & \\ & \delta_2^{-1}W^{-1} & & \\ & & \ddots & \\ & & & \delta_L^{-1}W^{-1} \end{bmatrix} \begin{bmatrix} I \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \begin{bmatrix} \delta_1^{-1}W^{-1} \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \tag{6.20}$$

and therefore, the tomography operation takes the form

$$\mathbf{A}\mathbf{W}^{-1}\mathbf{R}^T = \begin{bmatrix} \Gamma_1 & & \\ & \ddots & \\ & & \Gamma_G \end{bmatrix} \begin{bmatrix} \delta_1^{-1}P_{11}^{\mathrm{LGS}}W^{-1} \\ \vdots \\ \delta_1^{-1}P_{G1}^{\mathrm{NGS}}W^{-1} \end{bmatrix} = \begin{bmatrix} \delta_1^{-1}\Gamma_1 P_{11}^{\mathrm{LGS}}W^{-1} \\ \vdots \\ \delta_1^{-1}\Gamma_G P_{G1}^{\mathrm{NGS}}W^{-1} \end{bmatrix}. \tag{6.21}$$

Computationally, this operation is very efficient if the altitude of the ground layer is $h_1 = 0$ meters. In this case, all projection operators are identical,

$$P := P_{11}^{\mathrm{LGS}} = ... = P_{G1}^{\mathrm{NGS}}, \tag{6.22}$$

see equations for LGS and NGS projections (2.45) and (2.41), respectively.

Hence, the left-hand side operator of the sub-problem is of the form,

$$\widetilde{M} = \delta_1^{-2} W P^T (\Gamma_1^T \widehat{C}_1 \Gamma_1 + ... + \Gamma_G^T \widetilde{C}_G \Gamma_G) P W^{-1} + \alpha D_1. \tag{6.23}$$

By further assuming that the SH-WFSs are of the same geometry for all guide star directions, i.e.,

$$\Gamma := \Gamma_1 = ... = \Gamma_G, \tag{6.24}$$

the operator reduces to

$$\widetilde{M} = \delta_1^{-2} W P^T \Gamma^T \widehat{C}_\eta \Gamma P W^{-1} + \alpha D_1, \tag{6.25}$$

where the operator $\widehat{C}_\eta$ consists of two pre-computed sparse matrices related to the LGS and NGS components and tip-tilt removal steps for the LGS components,

$$\begin{aligned} \widehat{C}_\eta &:= \widehat{C}_1 + ... + \widetilde{C}_G \\ &= (I - T)(\widetilde{C}_1 + ... + \widetilde{C}_{G_{\mathrm{LGS}}})(I - T) + (\widetilde{C}_{G_{\mathrm{LGS}}+1} + ... + \widetilde{C}_G). \end{aligned} \tag{6.26}$$

Finally, in the discrete setting, by aligning the wavefront grid with the ground layer grid, the projection operators (interpolation operators) vanish and one is left with the operator

$$\widetilde{M} = \delta_1^{-2} W \Gamma^T \widehat{C}_\eta \Gamma W^{-1} + \alpha D_1, \tag{6.27}$$

where δ_1 is the discretization spacing on the ground layer, Γ is the Shack-Hartmann operator, W is the wavelet transform and D_1 is the regularization matrix (5.9) for the ground layer.

The matrix representation of $\widetilde{M}$ is not particularly sparse and the inverse of the matrix is full. However, in the operator form, where each component is applied in sequence, the operator can be applied with linear complexity in terms of number of operations.

Thus, in this approach, the GLMS Algorithm 6.1 can be applied with the step 2 replaced by a few iterations of the PCG algorithm with the left-hand side operator (6.27). Note that the PCG iterations of the ground layer problem are significantly cheaper numerically than the iterations of the full tomography problem (6.1). The special structure of the correction vector $\mathbf{v}$, see (6.15), is retained. Thus, step 5 of the algorithm does not change in numerical complexity from the GLMS method.

GLPCG with GLMS

The GLMS and the ground layer PCG implementation can be combined in two subsequent steps, where the initial guess is first improved by solving a small problem directly on the coarse scales, and then further improved with a few PCG iterations on all scales of the ground layer. Below we describe these two steps.

Let us denote by $\mathbf{R}_{\mathrm{CS}}$ and $\mathbf{R}_{\mathrm{AS}}$ the restriction operator associated to the coarse scales of the ground layer and the restriction operator associated to all scales of the ground layer, respectively. Further, we denote the sub-problem left-hand side operators restricted to the coarse scales and all scales of the ground layer by $\widetilde{M}_{\mathrm{CS}} = \mathbf{R}_{\mathrm{CS}} \mathbf{M} \mathbf{R}_{\mathrm{CS}}^T$ and by $\widetilde{M}_{\mathrm{AS}} = \mathbf{R}_{\mathrm{AS}} \mathbf{M} \mathbf{R}_{\mathrm{AS}}^T$, i.e., operators (6.8) and (6.27), respectively.

Observe that the operator combination $\mathbf{R}_{\mathrm{AS}} \mathbf{R}_{\mathrm{CS}}^T$ is a prolongation mapping from the coarse scale coefficients of the ground layer to all coefficients of the ground layer; the

operator is of dimension $2^{2J_1} \times 2^{2\tilde{J}_1}$, where $\tilde{J}_1$ is the number of coarse scales and J_1 is the number of all scales of layer $\ell = 1$.

We denote the coarse scale update vector by $\tilde{v}_{\rm CS}$ and the prolongated coarse scale update vector by $\mathbf{v}_{\rm CS} = \mathbf{R}_{\rm CS}^T \tilde{v}_{\rm CS}$. Applying the GLMS Algorithm 6.1, the coarse scale update is given by

$$\mathbf{v}_{\rm CS} = \mathbf{R}_{\rm CS}^T \widetilde{M}_{\rm CS}^{-1} \mathbf{R}_{\rm CS} \mathbf{r}^{(0)} \tag{6.28}$$

and the corresponding updated residual vector by

$$\mathbf{r}_{\rm CS} = \mathbf{r}^{(0)} - \mathbf{M}\mathbf{v}_{\rm CS}. \tag{6.29}$$

Hence, the residual equation for all scales of the ground layer is

$$\widetilde{M}_{\rm AS} \tilde{v}_{\rm AS} = \tilde{r}_{\rm AS}, \tag{6.30}$$

where $\tilde{r}_{\rm AS}$ is the updated residual, given by

$$\begin{aligned} \tilde{r}_{\rm AS} &= \mathbf{R}_{\rm AS}\mathbf{r}_{\rm CS} = \mathbf{R}_{\rm AS}\mathbf{r}^{(0)} - \mathbf{R}_{\rm AS}\mathbf{M}\mathbf{v}_{\rm CS} = \\ &= \mathbf{R}_{\rm AS}\mathbf{r}^{(0)} - \mathbf{R}_{\rm AS}\mathbf{M}\mathbf{R}_{\rm AS}^T\mathbf{R}_{\rm AS}\mathbf{v}_{\rm CS} \\ &= \mathbf{R}_{\rm AS}\mathbf{r}^{(0)} - \widetilde{M}_{\rm AS}(\mathbf{R}_{\rm AS}\mathbf{R}_{\rm CS}^T)\tilde{v}_{\rm CS}, \end{aligned} \tag{6.31}$$

i.e., the update of the residual involves invoking $\widetilde{M}_{\rm AS}$ once. We apply the PCG algorithm to the residual equation (6.30) and prolongate the update vector

$$\mathbf{v}_{\rm AS} = \mathbf{R}_{\rm AS}^T \tilde{v}_{\rm AS}. \tag{6.32}$$

Thus, the updated initial guess is given by

$$\mathbf{c}^{(1)} = \mathbf{c}^{(0)} + \mathbf{v}_{\rm CS} + \mathbf{v}_{\rm AS} \tag{6.33}$$

and the corresponding residual by

$$\mathbf{r}^{(1)} = \mathbf{b} - \mathbf{M}\mathbf{c}^{(1)} = \mathbf{r}^{(0)} - \mathbf{M}(\mathbf{v}_{\rm CS} + \mathbf{v}_{\rm AS}) \tag{6.34}$$

The GLMS the ground layer PCG steps are combined in the following algorithm.

Algorithm 6.2: The GLMS-GLPCG algorithm

Input : vector $\mathbf{c}^{(0)}$, residual $\mathbf{r}^{(0)} = \mathbf{b} - \mathbf{M}\mathbf{c}^{(0)}$
Output: vector $\mathbf{c}^{(1)}$, residual $\mathbf{r}^{(1)} = \mathbf{b} - \mathbf{M}\mathbf{c}^{(1)}$

1 $\tilde{r}_{\rm CS} = \mathbf{R}_{\rm CS}\mathbf{r}^{(0)}$ /* restrict residual */
2 $\tilde{v}_{\rm CS} = \widetilde{M}_{\rm CS}^{-1}\tilde{r}_{\rm CS}$ /* solve coarse scale sub-problem */
3 $\tilde{v}_{\rm AS}^{(0)} = (\mathbf{R}_{\rm AS}\mathbf{R}_{\rm CS}^T)\tilde{v}_{\rm CS}$ /* prolongate correction */
4 $\tilde{r}_{\rm AS} = \mathbf{R}_{\rm AS}\mathbf{r}^{(0)} - \widetilde{M}_{\rm AS}\tilde{v}_{\rm AS}^{(0)}$ /* update residual */
5 $\tilde{v}_{\rm AS}^{(1)} = \texttt{PCG}(\widetilde{M}_{\rm AS}, \tilde{r}_{\rm AS})$ /* apply Algorithm 3.2 to (6.30) */
6 $\tilde{v}_{\rm AS} = \tilde{v}_{\rm AS}^{(0)} + \tilde{v}_{\rm AS}^{(1)}$ /* combine corrections */
7 $\mathbf{v} = \mathbf{R}_{\rm AS}^T\tilde{v}_{\rm AS}$ /* prolongate correction */
8 $\mathbf{c}^{(1)} = \mathbf{c}^{(0)} + \mathbf{v}$ /* update initial guess */
9 $\mathbf{r}^{(1)} = \mathbf{r}^{(0)} - \mathbf{M}\mathbf{v}$ /* update residual */

The computationally demanding steps of the algorithm are 2, 4, 5 and 9. The cost of step 2 depends on the dimensions of the coarse scale matrix $\widetilde{M}_{\mathrm{CS}}^{-1}$. In steps 4 and 5, the operator $\widetilde{M}_{\mathrm{AS}}$ is applied.

Finally, the full operator $\mathbf{M}$ is invoked once in step 9, which is the predominantly expensive step of this algorithm. Here, the special structure of the update vector $\mathbf{v}$, see (6.15), can be used. Numerical tests of this method are left for future work.

Chapter 7

Numerics: the algorithm

7.1 Algorithm for AO systems

The FEWHA can be applied to various AO systems. In this section we outline the general structure of the algorithm for the following AO systems: SCAO, LTAO, MOAO and MCAO. These systems are introduced in Sections 2.8.1, 2.8.2, 2.8.3 and 2.8.4, respectively. In the following sections of this chapter we address the differences in the main components of the algorithm for each AO system.

Regime of operation of the AO system plays a vital role in the reconstruction algorithm. As discussed in Section 2.7, an AO system can operate either in open or closed loop. In open loop the measurements from the WFSs correspond to slopes of the incoming wavefronts, whereas in closed loop the WFS measurements originate from residual wavefronts post DM correction.

Let us summarize the four AO systems. The SCAO system utilizes $M = 1$ ground DM and $G = 1$ NGS. No tomographic reconstruction is required for the SCAO system. However our method can still be applied to this wavefront correction problem by reconstructing $L = 1$ layer at the ground, i.e., at the altitude of $h_1 = 0$ meters. The SCAO system runs in closed loop.

An LTAO system utilizes $M = 1$ ground DM and a combination of several LGSs and NGSs in G directions. The ground DM corrects towards a direction within the field of view, where no guide star is available. Hence, the LTAO system requires a tomographic reconstruction of the atmosphere to determine the wavefront aberrations in the direction of interest. Thus, in our algorithm for LTAO we first solve the atmospheric tomography problem on L layers, and then determine the mirror shape. The LTAO system runs in closed loop, where the residual data is obtained from the single DM in all G directions.

An MOAO system is similar to LTAO, in that the correction happens at the ground and G LGSs and NGSs are used in combination to obtain the measurement data. However, the MOAO system uses M mirrors to correct for multiple objects within the field of view, simultaneously. The MOAO system is operated in open loop.

Finally, an MCAO system utilizes M DMs and G WFSs towards LGSs and NGSs with the goal of obtaining a uniform correction over the field of view. These DMs are conjugated to several altitudes. The wavefront is corrected by all mirrors before the measurements are made, thus the MCAO system operates in closed loop.

In Section 5.2, we discussed how turbulence statistics are used as a regularization

term for the atmospheric tomography problem. The statistical information of the turbulence corresponds to full incoming wavefronts, not to the residual wavefronts after DM correction. Therefore, in order to apply our algorithm to the closed loop data, we modify the measurements with a method called the pseudo-open loop control (POLC) before applying our reconstruction algorithm. This problem does not arise with open loop data, hence the reconstructor may be applied directly to the data.

The straightforward idea of POLC is to approximate open loop measurements by combing the mirror shapes with the residual data. The POLC was introduced to AO in [14] and further studied in [22, 52]. It has proven to be stable and robust against large levels of system errors [52]. We rely on a modified POLC, where an integrator is used in the control scheme (see e.g., [52]).

The general structure of our method is presented in the Algorithm 7.1. In this general form, the algorithm can be applied to various AO systems (SCAO, LTAO, MOAO or MCAO). The two-step delay, see Section 2.7, is taken into account.

Algorithm 7.1: Reconstruction algorithm

Input : $\mathbf{s}$ (measurement vector)
$\mathbf{c}^{(0)}$ (wavelet coefficients vector)
$\mathbf{r}^{(0)}$ (residual coefficients vector)
$\mathbf{b}^{(0)}$ (right-hand side vector)
$\mathbf{a}^{(0)}$ (current DM shapes)
if *loop = closed* **then**
$\mathbf{a}^{(-1)}$ (previous DM shapes)

Output: $\mathbf{a}^{(1)}$ (next DM shapes)

if *loop = closed* **then**
  $\mathbf{s}^{\mathrm{ol}} = \mathbf{s} + \boldsymbol{\Gamma}\mathbf{a}^{(-1)}$ `/* compute pseudo-open loop data */`
else if *loop = open* **then**
  $\mathbf{s}^{\mathrm{ol}} = \mathbf{s}$
$\mathbf{b}^{(1)} = \mathbf{W}^{-T}\widehat{\mathbf{A}}^{T}\widehat{\mathbf{C}}_{\boldsymbol{\eta}}^{-1}\mathbf{s}^{\mathrm{ol}}$ `/* compute new RHS */`
$\mathbf{r}^{(0,1)} = \mathbf{b}^{(1)} - \mathbf{M}\mathbf{c}^{(0)} = (\mathbf{b}^{(1)} - \mathbf{b}^{(0)}) + \mathbf{r}^{(0)}$ `/* update residual */`
$(\mathbf{c}^{(1)}, \mathbf{r}^{(1)}) = \mathtt{tom}(\mathbf{c}^{(0)}, \mathbf{r}^{(0,1)})$ `/* tomography step, Alg. 7.2 */`
$\mathbf{a} = \mathbf{F}\mathbf{W}^{-1}\mathbf{c}^{(1)}$ `/* fitting step, Section 7.4 */`
if *loop = closed* **then**
  $\mathbf{a}^{(1)} = \mathbf{a}^{(0)} + \mathrm{gain} \cdot (\mathbf{a} - \mathbf{a}^{(-1)})$ `/* closed loop control */`
else if *loop = open* **then**
  $\mathbf{a}^{(1)} = (1 - \mathrm{gain}) \cdot \mathbf{a}^{(0)} + \mathrm{gain} \cdot \mathbf{a}$ `/* open loop control */`

We explain the components of the algorithm below. The input variables consist of the measurement data $\mathbf{s} = (s_g)_{g=1}^{G}$, whether from open or closed loop, the wavelet coefficients $\mathbf{c} = (c_\ell)_{\ell=1}^{L}$ containing the solution to the atmospheric tomography from the previous loop iteration, the associated right-hand side vector $\mathbf{b}^{(0)}$, see (5.51), and the corresponding residual vector $\mathbf{r}^{(0)} = \mathbf{b}^{(0)} - \mathbf{M}\mathbf{c}^{(0)}$. Additionally, we use the current DM shapes $\mathbf{a}^{(0)}$ and in the case of a closed loop system with a two-step delay, the previous DM shapes $\mathbf{a}^{(-1)}$. The output of the method are the new mirror shapes $\mathbf{a}^{(1)}$.

If the AO system runs in closed loop, the open loop-like measurements are computed in step 2 of the algorithm. Due to the two-step delay, see Section 2.7, the measurements $\mathbf{s}$ are obtained with the DM shapes from the previous step, $\mathbf{a}^{(-1)}$. In POLC, the open loop-like measurements are generated via the sum of the actual residual measurements coming from the sensor and the simulated Shack-Hartmann measurements through the DM(s). Depending on the AO system, the operator $\mathbf{\Gamma}$ has a different structure, which is presented for each AO system in Section 7.2.

Consequently, in step 10 of the algorithm, the artificially added DM shapes $\mathbf{a}^{(-1)}$ are again subtracted from the computed mirror shapes $\mathbf{a}$, such that the quantity $(\mathbf{a} - \mathbf{a}^{(-1)})$ corresponds to the reconstruction from the closed loop measurements.

In step 5 of the algorithm, the right-hand side vector $\mathbf{b}^{(1)}$ is computed for new data. We compute the new residual associated to $\mathbf{b}^{(1)}$ in step 6 of the algorithm. Observe how invoking $\mathbf{M}$ can be avoided in this step by keeping track of $\mathbf{r}^{(0)}$ and $\mathbf{b}^{(0)}$.

In step 7 and 8, the system-specific atmospheric reconstruction and the fitting step, respectively, are performed. In the atmospheric tomography step, the PCG algorithm is applied to equation (5.49) in the wavelet domain. The turbulence reconstruction may or may not be supplemented by the ground layer multi-scale (GLMS) method to improve the initial guess. We discuss the atmospheric tomography algorithm in Section 7.3. In the fitting step, the mirror shapes are derived directly from the reconstructed turbulence layers, or in the case of the MCAO an extra problem may need to be solved with the CG algorithm. We perform the fitting step in the piecewise bilinear domain, where $\mathbf{F}$ denotes the fitting operator. We describe this step in Section 7.4.

Finally, in step 10 and 12, the close and open loop control, respectively, is applied. The new DM shape is determined as a linear combination of the current and the reconstructed DM shapes. The weight between these two quantities is controlled by a scalar parameter called the gain, which has a value between zero and one. With a gain control, stability of the reconstruction is improved. More complex control schemes, such as Kalman filtering [51] or other predictive schemes [7] are possible.

The warm restart technique is implemented by keeping track of the wavelet coefficients, the residual coefficients and the right-hand side vectors.

7.2 Pseudo-open loop data

Pseudo-open loop data is generated differently for each of the AO systems in closed loop. In this section we give an overview of the different forms of the operator $\mathbf{\Gamma}$ in step 2 of Algorithm 7.1.

For the SCAO system in closed loop, one mirror is used to correct for measurements in one direction. Hence, the operator to produce the pseudo-open loop data is a single Shack-Hartmann operator,

$$\mathbf{\Gamma} = \Gamma. \tag{7.1}$$

In the case of an LTAO system, a single mirror at the ground is used to correct for all measurements in G directions. Here, the operator is a mapping of the form,

$$\mathbf{\Gamma} = \begin{bmatrix} \Gamma_1 \\ \vdots \\ \Gamma_G \end{bmatrix}. \tag{7.2}$$

In the case that all the sensors have the same geometry, i.e.,

$$\Gamma := \Gamma_1 = ... = \Gamma_G, \tag{7.3}$$

the Shack-Hartmann operator can be applied once and the measurements are copied to each direction,

$$\mathbf{\Gamma} = \begin{bmatrix} I \\ \vdots \\ I \end{bmatrix} \Gamma, \tag{7.4}$$

where I is the identity operator.

Finally, for an MCAO system with M deformable mirrors conjugated to altitudes $0 \leq \bar{h}_1 < ... < \bar{h}_M$, the operator that produces SH-WFS measurements through the DMs has a similar structure as the atmospheric tomography operator (5.30),

$$\mathbf{\Gamma} = \begin{bmatrix} \Gamma_1 & & \\ & \ddots & \\ & & \Gamma_G \end{bmatrix} \begin{bmatrix} \bar{P}_{11}^{\mathrm{LGS}} & \cdots & \bar{P}_{1M}^{\mathrm{LGS}} \\ \vdots & & \vdots \\ \bar{P}_{G1}^{\mathrm{NGS}} & \cdots & \bar{P}_{GM}^{\mathrm{NGS}} \end{bmatrix}. \tag{7.5}$$

Here, the first operator mimics the propagation of the wavefront through the mirrors, while the second operator simulates the WFSs. The blocks $\bar{P}_{gm}^{\mathrm{LGS}}$ and $\bar{P}_{g'm}^{\mathrm{NGS}}$ are projection operators defined similar to those for the atmospheric tomography operator (5.30) through the deformable mirrors $m = 1, ..., M$ towards the LGS $g = 1, ..., G_{\mathrm{LGS}}$ and the NGS $g' = G_{\mathrm{LGS}} + 1, ..., G$, respectively. The laser guide star projection takes the cone-effect into account.

The MOAO system runs in open loop, so the measurements correspond to full open loop wavefronts.

7.3 Atmospheric tomography algorithm

The atmospheric tomography algorithm, or wavefront reconstruction in the case of a SCAO system, involves solving the equation (5.49). Primarily, the problem is solved by applying a few iterations of the PCG Algorithm 3.2 with the operator $\mathbf{M}$, see (5.50), and a frequency-dependent preconditioner $\mathbf{\Lambda}_\tau$ or $\mathbf{\Lambda}_j$, see (6.3) or (6.4), respectively. Note that the dimension of the problem varies according to the number of layers L that the algorithm reconstructs.

The PCG is started with an initial guess provided from the warm restart. Prior to the PCG iteration, the initial guess can be improved by solving a coarse scale ground layer approximation of the problem via the GLMS Algorithm 6.1.

The algorithm for the atmospheric tomography is summarized in the following.

Algorithm 7.2: Atmospheric tomography

Input : $\mathbf{c}^{(0)}$ (wavelet coefficients vector)
$\mathbf{r}^{(0)}$ (residual coefficients vector)
Output: $\mathbf{c}^{(1)}$ (wavelet coefficients vector)
$\mathbf{r}^{(1)}$ (residual coefficients vector)

if *GLMS* **then**
  $(\mathbf{c}, \mathbf{r}) = \texttt{GLMS}(\mathbf{c}^{(0)}, \mathbf{r}^{(0)})$ `/* Algorithm 6.1 */`
else
  $(\mathbf{c}, \mathbf{r}) = (\mathbf{c}^{(0)}, \mathbf{r}^{(0)})$
$(\mathbf{c}^{(1)}, \mathbf{r}^{(1)}) = \texttt{PCG}(\mathbf{c}, \mathbf{r})$ `/* Algorithm 3.2 */`

Output of the method are the wavelet coefficients and the residual coefficients, both are kept to warm start the algorithm in the following iteration of the loop. The mirror shapes are determined from the reconstructed wavelet coefficients of the turbulence layers in the fitting step.

It is important to mention that we may reduce the dimension of the problem by eliminating those wavelets that are explicitly in the nullspace of the observation operator $\widehat{\mathbf{A}}\mathbf{W}^{-1}$, which maps wavelet coefficients to SH-WFS measurements. For instance, the wavelet φ_{00} on a bounded domain in the decomposition (5.8) is a constant function; hence, the Shack-Hartmann measurement of this function at any layer is zero. Similarly, wavelets at higher scales that have support outside of the union of the observed domains (5.18) also lie in the nullspace of the operator.

Naturally, due to the regularization term and the nature of the CG algorithm to project the solution away from the nullspace components, the wavelets in the nullspace will not be reconstructed by the algorithm. Nevertheless, by eliminating the nullspace wavelets from the problem, the dimension of the system can be reduced.

The wavelets that are fully in the nullspace of the observation operator are indexed by λ,

$$\mathbf{e}_\lambda \in \mathcal{N}(\widehat{\mathbf{A}}\mathbf{W}^{-1}), \tag{7.6}$$

where $\mathcal{N}(\cdot)$ is the nullspace of an operator and $\mathbf{e}_\lambda$ is a vector of dimension $\sum_{\ell=1}^{L} 2^{2J_\ell}$ with one at the λ-th component and zero elsewhere.

We compute the nullspace elements, i.e., such $\mathbf{e}_\lambda$ that $\widehat{\mathbf{A}}\mathbf{W}^{-1}\mathbf{e}_\lambda = 0$ holds, by verifying this condition for the norm,

$$\|\widehat{\mathbf{A}}\mathbf{W}^{-1}\mathbf{e}_\lambda\|^2_{\widehat{\mathbf{C}}_{\boldsymbol{\eta}}^{-1}} = 0. \tag{7.7}$$

Observe that the quantity

$$\|\widehat{\mathbf{A}}\mathbf{W}^{-1}\mathbf{e}_\lambda\|^2_{\widehat{\mathbf{C}}_{\boldsymbol{\eta}}^{-1}} = (\mathbf{W}^{-T}\widehat{\mathbf{A}}^T\widehat{\mathbf{C}}_{\boldsymbol{\eta}}^{-1}\widehat{\mathbf{A}}\mathbf{W}^{-1}\mathbf{e}_\lambda, \mathbf{e}_\lambda) \tag{7.8}$$

is the λ-th diagonal entry of the operator $\mathbf{W}^{-T}\widehat{\mathbf{A}}^T\widehat{\mathbf{C}}_{\boldsymbol{\eta}}^{-1}\widehat{\mathbf{A}}\mathbf{W}^{-1}$, which are available from pre-computing the frequency-dependent preconditioner. Thus, the explicit nullspace elements are given without any additional computational cost.

We point out that there are other elements in the nullspace that do not have an explicit form.

7.4 Fitting step

In the fitting step, the mirror shapes are fit to the reconstructed atmosphere. In principle, this is different for each AO system. However, the general form consists of first transforming the layers from the wavelet domain to the bilinear domain and then determining the mirror shapes, which are modeled as continuous piecewise bilinear functions, using projection operators in the bilinear domain.

We denote this step by

$$\mathbf{a} = \mathbf{F}\mathbf{W}^{-1}\mathbf{c}, \tag{7.9}$$

where $\mathbf{a} = (a_m)_{m=1}^M$ are the mirror shapes, $\mathbf{c} = (c_\ell)_{\ell=1}^L$ are the wavelet coefficients from the atmospheric tomography step, $\mathbf{W}^{-1}$ is the inverse wavelet transform and $\mathbf{F}$ is the fitting operator in bilinear domain, which is different for each AO system.

For the SCAO system, the reconstructed layer is located at the altitude of the DM, at the ground. The grid points of the reconstructed domain are aligned with the mirror nodal values. Hence, the mirror shape is obtained directly by applying the inverse DWT,

$$a_1 = \delta_1^{-1} W^{-1} c_1. \tag{7.10}$$

Here, the fitting operator $\mathbf{F}$ is the identity.

For the LTAO system, the mirror is optimized towards a certain direction of interest $\bar{\theta}_1$. The mirror shape is derived by first computing the inverse DWT to transform the layers from the wavelet domain to the bilinear domain and then by projecting through the reconstructed layers towards $\bar{\theta}_1$,

$$a_1 = \begin{bmatrix} P_{11}^{\mathrm{NGS}} & \cdots & P_{1L}^{\mathrm{NGS}} \end{bmatrix} \begin{bmatrix} \delta_1^{-1}W^{-1} & & \\ & \ddots & \\ & & \delta_L^{-1}W^{-1} \end{bmatrix} \begin{bmatrix} c_1 \\ \vdots \\ c_L \end{bmatrix}. \tag{7.11}$$

Here, the operator $P_{1\ell}^{\mathrm{NGS}}$ is a projection operator (bilinear interpolation) on layer $\ell = 1, ..., L$ towards the direction of interest, $\bar{\theta}_1$. The fitting operator is given by the projection $\mathbf{F} = (P_{1\ell}^{\mathrm{NGS}})_{\ell=1}^L$.

For the MOAO system, the mirror fitting is similar to that of LTAO, except that M mirrors are fit simultaneously towards directions of interest $\bar{\theta}_1, ..., \bar{\theta}_M$,

$$\begin{bmatrix} a_1 \\ \vdots \\ a_M \end{bmatrix} = \begin{bmatrix} P_{11}^{\mathrm{NGS}} & \cdots & P_{1L}^{\mathrm{NGS}} \\ \vdots & & \vdots \\ P_{M1}^{\mathrm{NGS}} & \cdots & P_{ML}^{\mathrm{NGS}} \end{bmatrix} \begin{bmatrix} \delta_1^{-1}W^{-1} & & \\ & \ddots & \\ & & \delta_L^{-1}W^{-1} \end{bmatrix} \begin{bmatrix} c_1 \\ \vdots \\ c_L \end{bmatrix}. \tag{7.12}$$

As above, $P_{m\ell}^{\mathrm{NGS}}$ is a projection operator on layer $\ell = 1, ..., L$ towards the direction of interest $\bar{\theta}_m$, for $m = 1, ..., M$. The fitting operator is again the projection $\mathbf{F} = (P_{m\ell}^{\mathrm{NGS}})_{g=1,\ell=1}^{G,L}$.

Finally, for the MCAO system, the fitting operation is a bit more complex, since the fitting requires aligning M deformable mirrors at various altitudes $0 \leq \bar{h}_1 < ... < \bar{h}_M$ to obtain a good correction over the field of view. There are two standard approaches how this can be done [20]. If the reconstructed layers are at the altitudes of the mirrors, the mirror shapes are given directly by the shapes of the layers. Otherwise, an auxiliary linear system of equations needs to be solved.

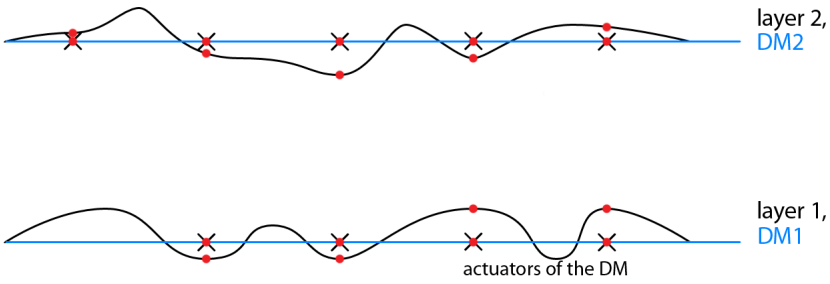

Figure 7.1: MCAO fitting step: mirror shapes are chosen according to the reconstructed layers.

In the first case, we reconstruct $L = M$ layers at altitudes specified by the mirrors, $h_1 = \bar{h}_1, ..., h_L = \bar{h}_M$. The mirror shapes are determined directly from the shapes of the turbulence layers,

$$\begin{bmatrix} a_1 \\ \vdots \\ a_M \end{bmatrix} = \begin{bmatrix} \delta_1^{-1} W^{-1} & & \\ & \ddots & \\ & & \delta_L^{-1} W^{-1} \end{bmatrix} \begin{bmatrix} c_1 \\ \vdots \\ c_L \end{bmatrix}. \tag{7.13}$$

Note that no interpolation onto the mirrors is required if the layer mesh grid is aligned with the mesh grid of the mirrors. We illustrate this configuration in Figure 7.1. In this case the fitting operator $\mathbf{F}$ is the identity operator.

In the second case, we reconstruct more layers than DMs, i.e., $L > M$ and consequently minimize the cost functional

$$\int_{\text{FoV}} \int_{\Omega_{\mathcal{M}}} \|(P_\theta^{\text{NGS}} \boldsymbol{\phi})(x) - (\bar{P}_\theta^{\text{NGS}} \mathbf{a})(x)\|^2 \mathrm{d}x \mathrm{d}\theta, \tag{7.14}$$

see (2.58). We discretize the integral over the field of view by choosing $\mathcal{I} \in \mathbb{N}$ discrete directions of interest, $\bar{\theta}_1, ..., \bar{\theta}_{\mathcal{I}}$, such that these directions cover the field of view. For a bilinear representation of the mirrors and the wavelet representation of the layers, this leads to a least squares equation, where the minimization functional is given by

$$\left\| \begin{bmatrix} \bar{P}_{11}^{\text{NGS}} & \cdots & \bar{P}_{1M}^{\text{NGS}} \\ \vdots & & \vdots \\ \bar{P}_{\mathcal{I}1}^{\text{NGS}} & \cdots & \bar{P}_{\mathcal{I}M}^{\text{NGS}} \end{bmatrix} \begin{bmatrix} a_1 \\ \vdots \\ a_M \end{bmatrix} - \begin{bmatrix} P_{11}^{\text{NGS}} & \cdots & P_{1L}^{\text{NGS}} \\ \vdots & & \vdots \\ P_{\mathcal{I}1}^{\text{NGS}} & \cdots & P_{\mathcal{I}L}^{\text{NGS}} \end{bmatrix} \begin{bmatrix} \delta_1^{-1} W^{-1} c_1 \\ \vdots \\ \delta_L^{-1} W^{-1} c_L \end{bmatrix} \right\| \tag{7.15}$$

over $\mathcal{I}$ discrete directions. Here the operator $\bar{P}_{im}^{\text{NGS}}$ is a projection (bilinear interpolation) through the DM m in direction $\bar{\theta}_i$ and the operator $P_{i\ell}^{\text{NGS}}$ is a projection through the layer ℓ in direction $\bar{\theta}_i$, for $1 \leq i \leq \mathcal{I}$, $1 \leq m \leq M$ and $1 \leq \ell \leq L$.

We denote minimization problem above using a short-hand notation by

$$\min_{\mathbf{a}} \|\bar{\mathbf{P}}\mathbf{a} - \mathbf{P}\mathbf{W}^{-1}\mathbf{c}\|. \tag{7.16}$$

The minimizer of the problem is found by solving the normal equation,

$$\bar{\mathbf{P}}^T \bar{\mathbf{P}} \mathbf{a} = \bar{\mathbf{P}}^T \mathbf{P} \mathbf{W}^{-1} \mathbf{c}. \tag{7.17}$$

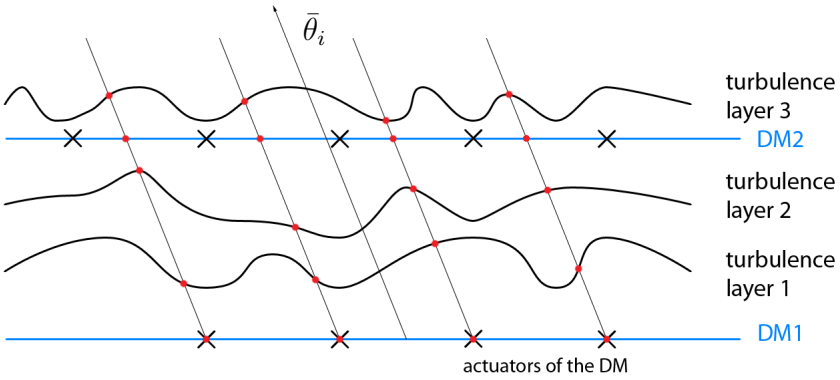

Figure 7.2: MCAO fitting step: fitting mirrors to reconstructed turbulence layers in a direction $\bar{\theta}_i$.

Here, the fitting operator is given by $\mathbf{F} = (\bar{\mathbf{P}}^T\bar{\mathbf{P}})^{-1}\bar{\mathbf{P}}^T\mathbf{P}$.

Instead of directly inverting the matrix, we keep the sparsity of the problem and solve the equation iteratively. We introduce a small penalty term for numerical stability and solve the equation

$$(\bar{\mathbf{P}}^T\bar{\mathbf{P}} + \alpha\mathbf{I})\mathbf{a} = \bar{\mathbf{P}}^T\mathbf{P}\mathbf{W}^{-1}\mathbf{c}, \tag{7.18}$$

with a very small α using the CG Algorithm 3.1. The CG algorithm is warm-started by keeping track of the mirror shapes $\mathbf{a}^{(0)}$ in Algorithm 7.1.

Note that this equation is solved completely in the piecewise bilinear domain. Alternatively, a multi-grid approach for this problem is possible, in which equation (7.18) is discretized on multiple grids with different grid resolution. Subsequently, the CG iteration is applied on the coarse grid first and the prolongated solution is used as the starting iterate for the CG method at a finer scale. This procedure is repeated until the final scale is reached. Numerical tests of this multi-grid adaptation are left for future work.

We point out that the minimization problem (7.16) is just one possible way how the cost function (2.58) can be discretized. Other discretization techniques might lead to a more favorable result or a numerically cheaper problem to solve. This investigation is also left for future work.

Chapter 8

Numerics: quality

8.1 Simulation configuration

To validate the performance of the FEWHA in terms of quality we test our method using numerical simulations. In this section we present the simulation environment settings and discuss the parameters of the algorithm.

Then, in the following sections we document the performance of the method for the SCAO, LTAO, MOAO and MCAO systems. For each system we compare our method with the corresponding benchmark method. Further, we examine the effects of certain features and parameters of the wavelet method on the quality. We refer to Chapter 2 for more information on the AO system components.

8.1.1 Simulation environment

For numerical simulations, we used the early design of the European Extremely Large Telescope. The telescope gathers light through a circular pupil with diameter of 42 m, of which roughly 28 percent are obstructed. At the time of writing, the proposed size of the primary mirror of the E-ELT is 39.3 m.

We demonstrate the performance of our method on OCTOPUS [40], the official end-to-end simulation tool of the ESO. The tool generates nine layers that obey the von Karman statistics, see Table 8.1. The Fried parameter is $r_0 = 0.129$ m. In the simulation, these layers are shifted according to the wind direction and speed. At discrete time frames, the wavefronts are projected onto the WFSs and the Shack-Hartmann measurements are computed from the photon distribution in the pixel detector plane of the sensor. The distribution of the photons on the pixel plane is governed by a Poisson process. The amount of noise is determined by the number of photons per subaperture of the sensor and frame. Additionally, read out noise can be modeled by introducing a failure rate in the number of measured electrons per pixel of the detector and frame.

The LGS measurements are simulated with cone effect, false tip-tilt component and spot elongation. The sodium layer at which the laser beam scatters is modeled as a Gaussian random variable at the mean altitude of $H = 90$ km and a FWHM parameter of 11.4 km. The laser beams are launched from four positions on the sides of the telescope, at the corner points of the $[-16.26 \text{ m}, 16.26 \text{ m}]^2$ square.

The AO system obeys a two-step delay between the time that the measurements are

Layer	Altitude	Strength C_n^2	Speed
1	47 m	0.5224	15 m/s
2	140 m	0.0260	13 m/s
3	281 m	0.0444	13 m/s
4	562 m	0.1160	9 m/s
5	1125 m	0.0989	9 m/s
6	2250 m	0.0295	15 m/s
7	4500 m	0.0598	25 m/s
8	9000 m	0.0430	40 m/s
9	18000 m	0.0600	21 m/s

Table 8.1: Configuration for the simulated 9 layers atmosphere.

Parameter	Value
Telescope diameter	42 m
Central obstruction	$\sim 28\%$
Fried parameter r_0	0.129 m
Na-layer height H	90 km
Na-layer thickness FWHM	11.4 km
Delay	2 frames
Simulated duration	1 s
Evaluation criterion	LE Strehl
Evaluation wavelength	K band (2200 nm)

Table 8.2: Simulation settings.

collected and the time that the DM updates are applied. In the test cases below we simulate one second of evolving atmosphere. As a criterion, the long exposure (LE) Strehl ratio is used in the K band (for a wavelength of 2200 nm).

We summarize these general simulation settings in Table 8.2. The system-specific settings for SCAO, LTAO, MOAO and MCAO are outlined in Table 8.3, 8.4 (for LTAO and MOAO) and 8.5, respectively. These settings are explained below.

The time unit corresponds to the duration of one time frame. For the SCAO system it is 1 ms, whereas for the other systems it is 2 ms. For the simulated 1 s of atmosphere, this corresponds to 1000 loop iterations in the SCAO case and 500 loop iterations for the other systems.

Each system utilizes G_{LGS} LGSs and $G_{\text{NGS}} = G - G_{\text{LGS}}$ NGSs, i.e., G guide stars in total. A Shack-Hartmann wavefront sensor is assigned to each guide star. Two types of sensors are used: high resolution sensors for full wavefront correction and low resolution sensors for tip-tilt aberrations. The high resolution sensors utilize $n_s \times n_s = 84 \times 84$ subapertures, of which roughly 72 % are active. Hence, for a telescope diameter of 42 m, the subaperture detects wavefront aberrations over a 0.5 m $\times$ 0.5 m square area. The low resolution tip-tilt sensors have $n_s \times n_s = 2 \times 2$ or 1×1 subapertures, which are used to detect the low order wavefront aberrations. The geometry of the area over which the tip-tilt sensor detects light is illustrated in Figure 2.11. All four simulated AO

Parameter	Value
Time unit (sampling frequency)	1 ms (1000 Hz)
Number of loop iterations	1000
Number of NGSs	1
NGS WFS subapertures	84×84, 4976 active
NGS direction	zenith
NGS wavelength	700 nm
NGS flux	1 - 10^4 photons / subap. / frame
NGS read out noise	0 e^- / pixel / frame
Number of LGSs	0
Number of probe-stars	1
Probe-star direction	zenith
Number of DMs	1
DM actuators	85×85, 5402 active
DM altitude(s)	0 m
DM actuator spacing	0.5 m

Table 8.3: SCAO system configuration.

Parameter	LTAO Value	MOAO Value
Time unit (sampling frequency)	2 ms (500 Hz)	
Number of loop iterations	500	
Number of LGSs	6	
LGS WFS subapertures	84×84, 5040 active	
LGS direction	circle of 7.5 arcmin diameter	
LGS wavelength	589 nm	
LGS flux	20 - 500	5 - 500 photons / subap. / frame
LGS read out noise	3	0 e^- / pixel / frame
Number of NGSs	3	
NGS WFS subapertures	84×84, 5040 active	
NGS direction	circle of 10 arcmin diameter	
NGS wavelength	500 nm	
NGS flux	300	5 - 500 photons / subap. / frame
NGS read out noise	3	0 e^- / pixel / frame
Number of probe-stars	1	
Probe-star direction	zenith	
Number of DMs	1	
DM actuators	85×85, 5402 active	
DM altitude(s)	0 m	
DM actuator spacing	0.5 m	

Table 8.4: LTAO and MOAO system configurations.

Parameter	Value
Time unit (sampling frequency)	2 ms (500 Hz)
Number of loop iterations	500

Number of LGSs	6
LGS WFS subapertures	84×84, 5040 active
LGS direction	circle of 2 arcmin diameter
LGS wavelength	589 nm
LGS flux	20 - 500 photons / subap. / frame
LGS read out noise	3 e^- / pixel / frame

Number of NGSs	3
NGS WFS subapertures	one 2×2, two 1×1
NGS direction	circle of 8/3 arcmin diameter
NGS wavelength	1650 nm
NGS flux	500 photons / subap. / frame
NGS read out noise	5 e^- / pixel / frame

Number of probe-stars	25
Probe-star direction	5×5 grid, over 2 arcmin square, see Fig. 8.1
Number of DMs	3
DM actuators	85×85, 5402 active 47×47, 1669 active 53×53, 2225 active
DM altitude(s)	0 km, 4 km, 12.7 km
DM actuator spacing	0.5 m, 1 m, 1 m

Table 8.5: MCAO system configurations. The star asterism and the evaluation grid for MCAO are illustrated in Figure 8.1.

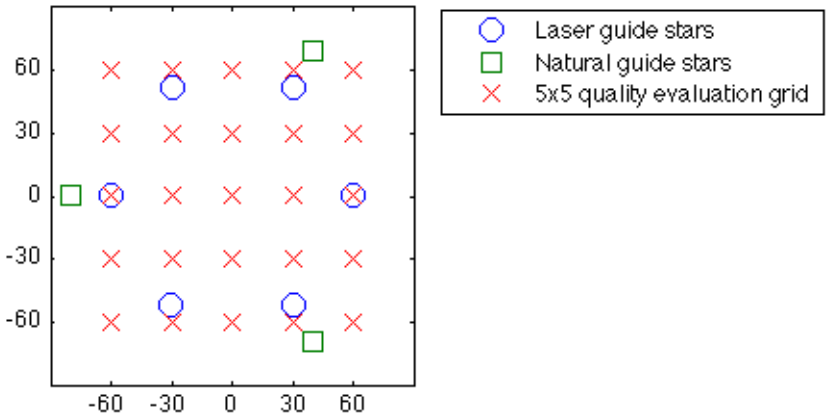

Figure 8.1: MCAO star asterism of the six LGSs and three NGSs, as well as the 5×5 quality evaluation grid over the field of view (in arcsec).

systems use high order sensors; the high resolution LGS sensors of the MCAO system are supplemented with tip-tilt sensors.

The wavelength λ at which the WFS operates appears in the scaling factor between the phase and the wavefront aberrations, see (2.9). The flux and the read out noise parameters determine the amount of noise that affects the sensor.

A probe-star is the direction in which the Strehl ratio is evaluated. The incoming wavefront towards a probe-star passes one or multiple DMs. The AO system utilizes M DMs at altitudes $\bar{h}_m$. Each DM is modeled as a continuous piecewise bilinear function defined on a grid of $n_a \times n_a$ points. The nodal values of these functions are given by the actuator positions. Due to the circular shape of the telescope with central obstruction, not all of the actuator are active.

In SCAO, the guide star and the probe-star directions coincide. The guide stars of the LTAO and MOAO systems are located at a wide angle of separation, positioned in a circle of 7.5 and 10 arcmin diameter for the LGS and NGS, respectively; the quality is evaluated for a correction towards the center, i.e., the zenith direction.

The MCAO LGS and NGS are located at a more narrow separation angle of 2 and 8/3 arcmin diameter, respectively. However, these NGS stars are fainter than those for LTAO and MOAO simulations. The quality for MCAO is measured in 25 probe-star directions. The LGS and NGS star asterism along with the 25 probe-star directions of the MCAO simulations are illustrated in Figure 8.1.

8.1.2 Method parameters

The wavelet method can be configured using several parameters. These parameters are summarized in Table 8.6 and explained below.

For each parameter in Table 8.6 we state its range of values and a typical value that is used in the algorithm. Moreover, we classify a parameter as variable, fixed or trade-off, if it must be tuned for different noise levels, if it is chosen once for any noise level or if it relates to a trade-off between quality and speed of the method, respectively.

We call a variable parameter sensitive if a small deviation from the optimal value has a big effect on the performance of the method. We mark a parameter as an off-line parameter if the Jacobi preconditioner or the GLMS matrix must be recomputed when the parameter is modified. Otherwise, updating the parameter can be performed at run-time and it is considered as an on-line parameter. Note that updating the preconditioner or recomputing the GLMS matrix is computationally cheap for the wavelet method.

The first parameter to be set for the atmospheric tomography reconstructors (for LTAO, MOAO and MCAO) is the number of reconstructed layers, L. For SCAO this number is fixed to one, the ground layer. For each layer $\ell = 1, ..., L$ we set its altitude h_ℓ and optical strength $C_n^2(h_\ell)$.

Although the number of layers can be considered as a trade-off parameter and h_ℓ and $C_n^2(h_\ell)$ as variable parameters, for our purposes they are fixed. A sensitivity study of the method to varying L, h_ℓ and $C_n^2(h_\ell)$ is future work.

Additionally, we set the layer discretization parameters J_ℓ and δ_ℓ. The number of wavelet scales J_ℓ induces a grid of $2^{J_\ell} \times 2^{J_\ell}$ points with equidistant spacing on layer ℓ, where the spacing between the points is controlled by δ_ℓ.

The discretization parameters have an effect on the quality and the computational cost

Parameter	Range of values	Typical value	Variable / fixed / trade-off	Sensitive	Update
Num. of layers L	$\mathbb{N}$	see Sec. 8.1.3	fixed	–	off-line
Altitude h_ℓ	$[0, \infty)$ m	see Sec. 8.1.3	fixed	–	off-line
Strength $C_n^2(h_\ell)$	(0,1]	see Sec. 8.1.3	fixed	–	on-line
Scales J_ℓ	$\mathbb{N}$	see Sec. 8.1.3	trade-off	–	off-line
Spacing δ_ℓ	$(0, \infty)$ m	see Sec. 8.1.3	trade-off	–	off-line
Regularization α	$[0, \infty)$	1	variable	very	on-line
Elongation tuning α_η	[0,1]	0.8	variable	yes	off-line
Gain	$(0, 1]$	0.4	variable	no	on-line
Num. of GLMS scales	$0, ..., J_1$	4	trade-off	–	off-line
Num. of PCG iterations n_{iter}	$\mathbb{N}$	4	trade-off	–	on-line
Preconditioner thresholding	$[0, \infty)$ or $0, ..., J_\ell - 1$	10^7 or $J_\ell - 2$	fixed	–	on-line
Num. of optimization directions $\mathcal{I}^{(1)}$	$\mathbb{N}$	25	trade-off		on-line
Optimization directions $\bar{\theta}_i^{(1)}$	angles over FoV	probe-star directions	trade-off	–	on-line
Num. of CG iterations for fitting equation$^{(1)}$	$\mathbb{N}$	4	trade-off	–	on-line

$^{(1)}$ – only for MCAO with more reconstructed layers than DMs.

Table 8.6: Parameters of the wavelet method.

of the method. We discuss the effect of discretization on the quality of the reconstruction in terms of an example in Section 8.4.6 and present the computational cost estimates with respect to the grid size in Section 9.2.

Next, the regularization parameter α, see (5.49), the elongation tuning parameter α_η, see (2.53), and the gain, see Algorithm 7.1 are the variable parameter of the method. By tuning these parameters, the qualitative performance of the method can be improved for individual (photon) noise levels. We comment on the sensitivity of the method to the changes in these parameters in Section 8.4.2.

Further, the number of GLMS scales and PCG iterations can be adjusted, to a certain extent, to improve the quality of the method. By increasing the number of scales or iterations, the computational cost of the algorithm also increases. Hence, these are trade-off parameters between quality and speed of the method. We investigate the qualitative performance of the method with respect to these parameters in Sections 8.4.3 and 8.4.4. Moreover, we present the computational cost estimates in terms of these parameters in Section 9.2 and measure the speed of the algorithm for different number of iterations in Section 9.3.

Finally, the preconditioner thresholding parameter can be adjusted either by the global value τ, see (6.3), or by the number of scales j, see (6.4). Alternatively, if no preconditioner is used (or a very large τ is chosen), the method reverts to a CG iteration. The thresholding parameter needs to be set once for a specific AO system configuration and is fixed with respect to changing (photon) noise levels. We observe the effect of these parameters on our method in Sections 8.4.5 and 8.5.2.

The last three parameters in Table 8.6 are used only for MCAO if we reconstruct more layers than DMs. By choosing a different number of optimization directions and different optimization geometries for the fitting problem, different results in quality and speed of the method are obtained. We compare several optimization geometries in terms of quality in Section 8.5.3 and present the computational cost estimates in Section 9.2. The number of CG iterations for the fitting equation is also a trade-off parameter between quality and speed.

8.1.3 System-specific method parameters

The reconstruction layers are configured differently for each AO system. We present the system-specific layer settings below.

For SCAO, where we reconstruct $L = 1$ layer, the altitude is set to $h_1 = 0$ m and the optical strength to $C_n^2(h_1) = 1$. The discretization grid is set according to the DM specifications, see Table 8.3: we use a spacing of $\delta_1 = 0.5$ m and $J_1 = 7$ wavelet scales, such that the layer grid of $2^{J_1} \times 2^{J_1} = 128 \times 128$ points overlaps the DM grid of 85×85 actuators. The layer grid is aligned with the actuator grid.

In the LTAO and MOAO tests we reconstruct the $L = 9$ simulated layers and assume their position and optical strength to be known, see Table 8.1. The exception is the altitude of layer $\ell = 1$, which is set to 0 m for the reasons of computational efficiency. Reconstruction layer settings for LTAO and MOAO are summarized in Table 8.7. We use two configurations for layer discretization: a quality-oriented setup and a speed-oriented one.

In the quality-oriented setup all layers are discretized with the resolution of 0.5 m.

			Quality setup			Speed setup		
Layer ℓ	Altitude h_ℓ	Strength $C_n^2(h_\ell)$	Scales J_ℓ	Grid points	Spacing δ_ℓ	Scales J_ℓ	Grid points	Spacing δ_ℓ
1	0 m	0.5224	7	128×128	0.5 m	7	128×128	0.5 m
2	140 m	0.0260	7	128×128	0.5 m	7	128×128	0.5 m
3	281 m	0.0444	7	128×128	0.5 m	7	128×128	0.5 m
4	562 m	0.1160	7	128×128	0.5 m	7	128×128	0.5 m
5	1125 m	0.0989	7	128×128	0.5 m	7	128×128	0.5 m
6	2250 m	0.0295	7	128×128	0.5 m	7	128×128	0.5 m
7	4500 m	0.0598	7	128×128	0.5 m	7	128×128	0.5 m
8	9000 m	0.0430	8	256×256	0.5 m	7	128×128	1 m
9	18000 m	0.0600	8	256×256	0.5 m	7	128×128	1 m

Table 8.7: LTAO / MOAO configuration with $L = 9$ reconstructed layers. Quality setup uses a finer layer discretization for two upper-most layers; speed setup uses the same number of grid points at all layers.

Layer ℓ	Altitude h_ℓ	Strength $C_n^2(h_\ell)$	Scales J_ℓ	Grid points	Spacing δ_ℓ
1	0 m	0.75	7	128×128	0.5 m
2	4000 m	0.15	6	64×64	1 m
3	12700 m	0.10	6	64×64	1 m

Table 8.8: MCAO 3-layer configuration where $L = 3$ layers are reconstructed at DMs.

Due to the wide angle of separation of the guide stars, the two upper-most layers are discretized with more grid points than the rest. The choice of 0.5 m spacing is motived by the DM spacing and WFS subaperture size, which in both cases is 0.5 m.

In the speed-oriented setup we use a grid of 128×128 points for the ground layer that overlaps the DM grid. Consequently, the same number of points is used to discretize each layer at higher altitudes, which is advantageous in terms of parallelization of the algorithm. This results in the two upper-most layers represented at a lower resolution of 1 m than layers 1 through 7. In Section 8.4.6 we demonstrate that the performance loss in quality of the speed-oriented approach is small compared to the quality-oriented setup.

For MCAO, the method can be run in one of two configurations. In the first configuration we reconstruct layers at the altitudes of the DMs. We set the altitude, the number of grid points and the spacing parameters of the three layers according to the mirror specifications in Table 8.5. Moreover, the layer grids are aligned with the actuator grids of the mirrors. The C_n^2 profile is chosen heuristically, such that the ground layer carries most of the optical strength; this is motivated by empirical observations. These settings are outlined in Table 8.8. We refer to this configuration as the 3-layer approach.

In the second configuration for MCAO we reconstruct the whole simulated atmosphere, see Table 8.1. As for LTAO and MOAO, we set the altitude of the ground layer

Layer ℓ	Altitude h_ℓ	Strength $C_n^2(h_\ell)$	Scales J_ℓ	Grid points	Spacing δ_ℓ
1	0 m	0.5224	7	128×128	0.5 m
2	140 m	0.0260	7	128×128	0.5 m
3	281 m	0.0444	7	128×128	0.5 m
4	562 m	0.1160	7	128×128	0.5 m
5	1125 m	0.0989	7	128×128	0.5 m
6	2250 m	0.0295	7	128×128	0.5 m
7	4500 m	0.0598	7	128×128	0.5 m
8	9000 m	0.0430	7	128×128	0.5 m
9	18000 m	0.0600	7	128×128	0.5 m

Table 8.9: MCAO 9-layer configuration with $L = 9$ reconstructed layers.

to 0 m. Due to the narrow angle of separation of the guide stars, all layers are discretized with a resolution of 0.5 m and a discretization grid of 128×128 points. These settings are summarized in Table 8.9. In this configuration, the mirrors are determined by solving the fitting equation, where we optimize the reconstruction towards a certain set of directions. We refer to this configuration as the 9-layer approach.

The 3-layer approach coincides with the 9-layer approach if all guide stars are NGS, i.e., the cone effect is not present, and the optimization directions are chosen as the directions of the guide stars, see [54]. Note that the first configuration uses no prior knowledge on the atmosphere, whereas the second one does.

8.2 SCAO results

8.2.1 Performance against the benchmark method

The benchmark in quality for SCAO is the MVM tuned by the ESO. The reference results were provided by the ESO. In this section we demonstrate the performance of our method against the benchmark.

The wavelet method is configured as follows. We reconstruct $L = 1$ ground layer at altitude $h_1 = 0$ m with optical strength of $C_n^2(h_1) = 1$ and discretization parameters $\delta_1 = 0.5$ m, $J_1 = 7$. The variable parameters of the method, i.e., regularization α and gain parameters, are summarized in Table 8.10 for different flux levels.

Parameter	Flux						
	10^4	1000	100	50	10	5	1
α	20	10	1	1	1	1	1
Gain	0.4	0.4	0.3	0.3	0.2	0.2	0.15

Table 8.10: Parameter settings for SCAO for different flux, given in number of photons per subaperture and frame.

For these tests we used the standard Jacobi preconditioner $\mathbf{\Lambda}$, see (6.2), and did not

use the GLMS method. The number of PCG iterations is 8.

The LE Strehl in K band after 1000 iterations is plotted versus detected NGS photon flux in Figure Figure 8.2, with $0e^-$ read out noise per pixel.

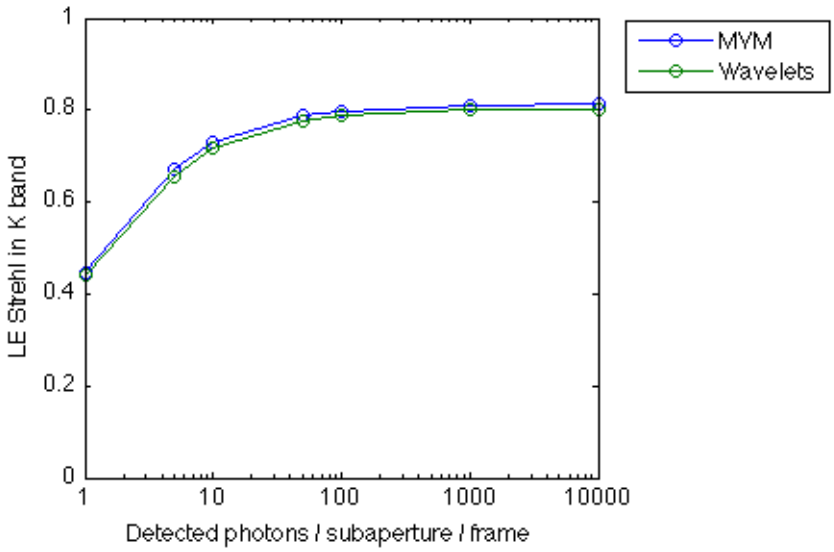

Figure 8.2: SCAO K band Strehl vs NGS flux.

The tests show that both methods produce comparable results.

8.3 LTAO results

8.3.1 Performance against the benchmark method

The benchmark in quality for LTAO is the FrIM tuned by the ESO. The reference results were provided by the ESO using an unpreconditioned version of the algorithm. In this configuration, FrIM uses up to 100 CG iterations. A preconditioned FrIM needs much less iteration until convergence.

In this section we demonstrate the performance of our method against the benchmark. We investigate the performance of the wavelet method in two configurations: with 4 and 10 PCG iterations.

The wavelet method is configured as follows. We reconstruct $L = 9$ layers of the atmosphere, where the altitude, optical strength and discretization parameters are given by the "quality setup" configuration in Table 8.7. The variable parameters of the method, i.e., regularization α, elongation tuning α_η and gain parameters, are summarized in Table 8.11 for different flux levels.

The number of GLMS scales used in all tests is $\tilde{J}_1 = 4$. We use the global thresholding preconditioner $\mathbf{\Lambda}_\tau$, see (6.3), with the thresholding parameter given by $\tau = 10^6 \min(\mathrm{diag}(\mathbf{D}))$.

The LE Strehl in K band after 500 iterations for LTAO versus subaperture photon flux is plotted in Figure 8.3. These flux values are for the LGS detector only. For the

Parameter	Flux						
	500	400	300	200	100	50	20
α	24	18	18	18	18	18	18
α_η	0.2	0.4	0.6	0.8	0.8	0.8	0.8
Gain	0.4	0.4	0.4	0.4	0.4	0.4	0.4

Table 8.11: Parameter settings for LTAO for different flux, given in number of photons per subaperture and frame.

NGS there are 300 photons/subaperture. The detector read out noise in all cases is 3 electrons/pixel for both detectors.

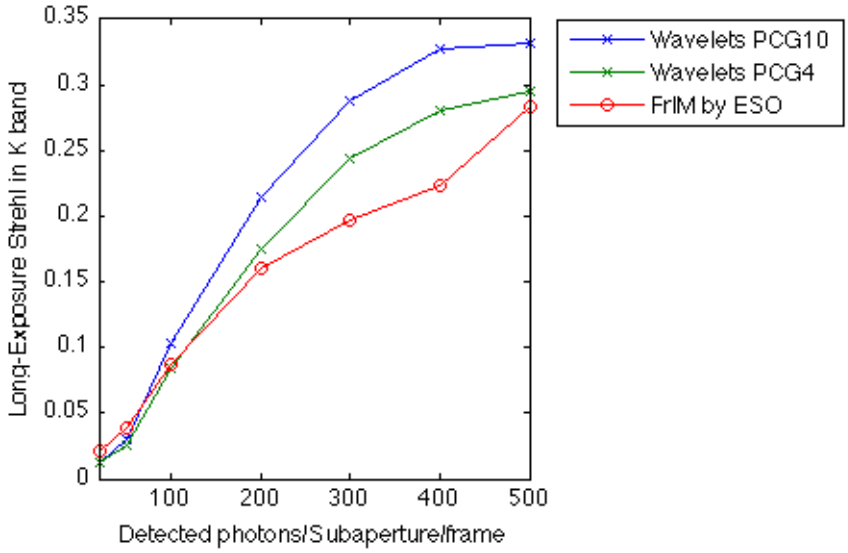

Figure 8.3: LTAO K band Strehl vs LGS flux.

Whereas the configuration of the wavelet method with 4 PCG iterations meets the quality requirements of the reference method, the configuration with 10 PCG iterations shows a considerable improvement over the reference results.

8.4 MOAO results

8.4.1 Performance against the benchmark method

As for LTAO, the benchmark in quality for MOAO is the FrIM tuned by the ESO. The reference results were provided by the ESO using an unpreconditioned version of the algorithm. In this configuration, FrIM uses up to 100 CG iterations. A preconditioned FrIM needs much less iteration until convergence.

In this section we demonstrate the performance of our method against the benchmark. We investigate the performance of the wavelet method in two configurations: with 8 PCG iterations in a speed-oriented setup and with 10 PCG iterations in a quality-oriented setup.

The wavelet method is configured as follows. We reconstruct $L = 9$ layers of the atmosphere, where the altitude, optical strength and discretization parameters are given by the "speed setup" configuration in Table 8.7 for the speed-oriented approach and by the "quality setup" configuration for the quality-oriented approach. The variable parameters of the method, i.e., regularization α, elongation tuning α_η and gain parameters, are summarized for both configurations in Table 8.12 for different flux levels.

Parameter	Flux				
	500	100	20	10	5
α	24	4	1	1	1
α_η	0.2	0.4	0.8	0.8	0.8
Gain	0.5	0.4	0.2	0.2	0.2

Table 8.12: Parameter settings for MOAO for different flux, given in number of photons per subaperture and frame.

The number of GLMS scales used in all tests is $\tilde{J}_1 = 4$. We use the scale controlled preconditioner $\mathbf{\Lambda}_j$, see (6.4), with the scale threshold parameter set to $j = J_\ell - 2$ for the speed-oriented approach and we use the global thresholding preconditioner $\mathbf{\Lambda}_\tau$, see (6.3), with the thresholding parameter given by $\tau = 10^6 \min(\mathrm{diag}(\mathbf{D}))$ for the quality-oriented approach.

The LE Strehl in K band after 500 iterations for MOAO versus subaperture photon flux is plotted in Figure 8.4. These flux values are for both the LGS and NGS. The detector read out noise in all cases is 0 electrons/pixel for both detectors.

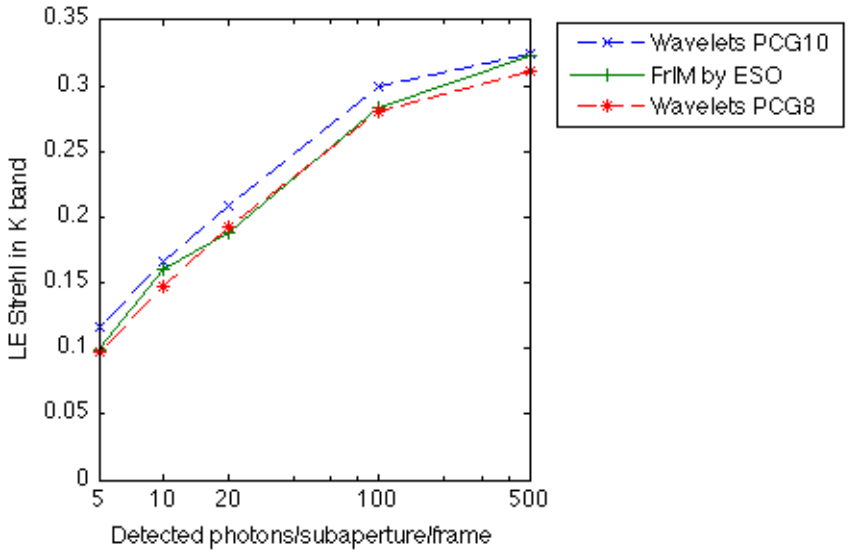

Figure 8.4: MOAO K band Strehl vs LGS flux.

Whereas the speed-oriented configuration of the wavelet method with 8 PCG iterations meets the quality requirements of the reference method, the quality-oriented configuration with 10 PCG iterations shows an improvement over the reference results.

8.4.2 Sensitivity of the method to variable parameters

We investigate the parameter sensitivity of the method for the 100 photon case only. Default parameters for the following tests are as in the quality-oriented approach above: the layers are set according to the "quality setup" configuration in Table 8.7, we use 10 PCG iterations, the number of GLMS scales used in all tests is $\tilde{J}_1 = 4$, a global thresholding preconditioner $\mathbf{\Lambda}_\tau$, see (6.3), is used with the thresholding parameter given by $\tau = 10^6 \min(\mathrm{diag}(\mathbf{D}))$.

The variable parameters in these tests are: regularization α, elongation tuning α_η and the gain. The default settings for these parameters are chosen as in Table 8.12, i.e., $\alpha = 4$, $\alpha_\eta = 0.4$ and gain is 0.4.

The metric is as before, the LE Strehl in K band after 500 iterations of the loop. The noise level is fixed to 100 photons per subaperture and frame. The reference method in the test is again the FrIM tuned by the ESO. Note that also the reference case parameter-dependent. We compare the performance of our method with the FrIM result for the 100 photon case.

In Figure 8.5 we illustrate the sensitivity of the method to a change in the regularization parameter α, see (5.49). We variate parameter α between 0.25 and 64 while keeping the other parameters constant. As can be observed, the method is very sensitive to the regularization parameter. Although for values $\alpha \in [2, 8]$ the method still outperforms the reference case, the wrong choice of the regularization parameter has a significant negative effect on the result. Note that α is an on-line parameter, i.e., it can updated on the fly without any significant computing costs.

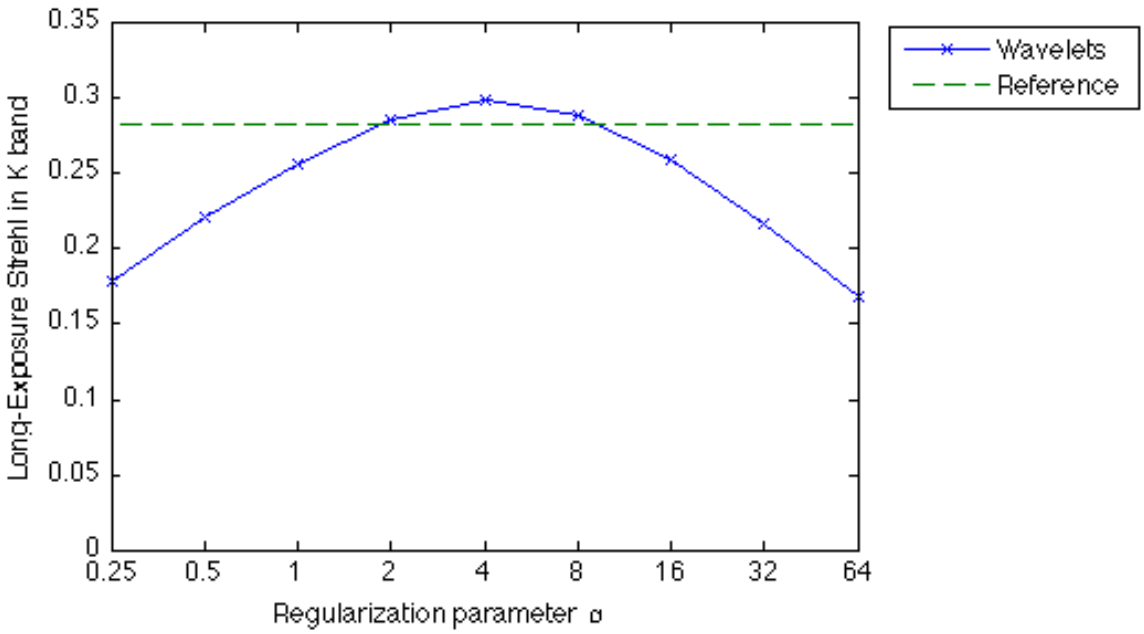

Figure 8.5: Performance of the algorithm with different regularization parameters α for the 100 photon case.

The test shows that the wavelet approximation of the penalty term (5.12) holds up to a certain constant. Moreover, to get a sufficient result, the relation between the penalty

term and the fitting term in the system (5.49) must be correctly balanced. All higher values of α than the optimal over-regularize the problem and the performance of the algorithm degrades. In contrast, all lower values of α under-regularize the problem and the performance of the algorithm becomes unstable.

In Figure 8.6 we illustrate the sensitivity of the method to changes in the spot elongation tuning parameter α_η, see (2.53). We variate parameter α_η between 0 (full NGS noise model) and 1 (full LGS noise model) while keeping the other parameters constant. As can be observed, the method is moderately sensitive to the choice of the elongation tuning parameter. The method outperforms the reference in the range of values between 0.2 and 0.6. Moreover, increasing the parameter up to the value 1 does not decrease the performance significantly. Only at value $\alpha_\eta = 0$, i.e., the full non-elongated noise model, the method suffers dramatically in performance. This test shows that the spot elongation noise must be taken into account to obtain reasonable reconstruction from the LGS sensors.

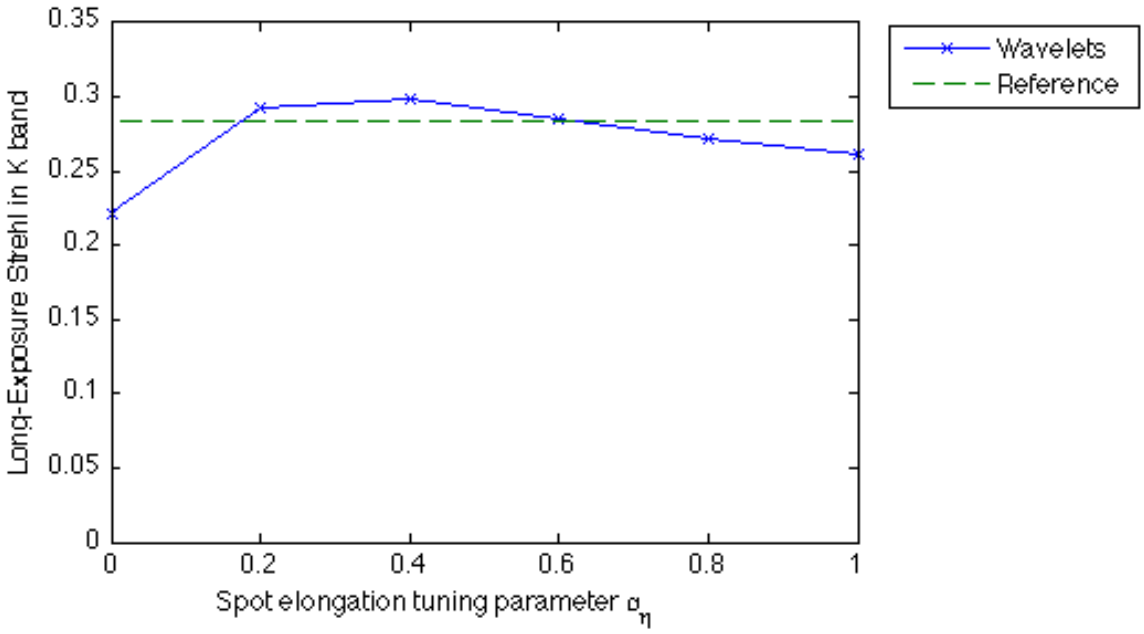

Figure 8.6: Performance of the algorithm with different elongation tuning parameters α_η for the 100 photon case. The case $\alpha_\eta = 0$ corresponds to the full NGS noise model; the case $\alpha_\eta = 1$ to the full LGS noise model.

In Figure 8.7 we illustrate the sensitivity of the method to change in the gain, see Algorithm 7.1. We variate the gain parameter between 0.1 and 1 while keeping the other parameters constant. As can be observed, the method is not sensitive to gain. The method outperforms the reference in the range of values between 0.3 and 0.8. Moreover, the performance of the method is relatively constant for all choices of gain between 0.2 and 1. Only for a very low gain value of 0.1 the method suffers dramatically in performance. Note that the gain is an on-line parameter, i.e., it can updated on the fly without any computing costs.

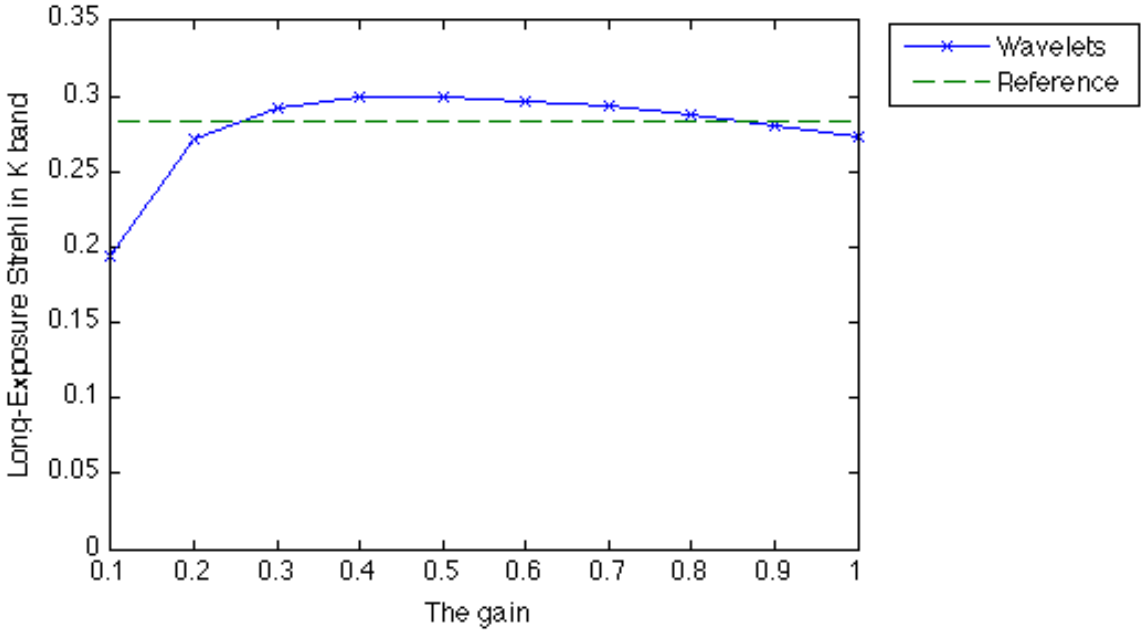

Figure 8.7: Performance of the algorithm with different gain for the 100 photon case.

8.4.3 Convergence of the method

In this test we demonstrate the convergence properties of the method in terms of the number of PCG iterations. As above we concentrate on the 100 photon case only. Default parameters for the test are as in the speed-oriented approach above: the layers are set according to the "speed setup" configuration in Table 8.7, the variable parameters are chosen as in Table 8.12, i.e., $\alpha = 4$, $\alpha_\eta = 0.4$ and gain is 0.4, the number of GLMS scales used in all tests is $\tilde{J}_1 = 4$, a scale controlled preconditioner $\mathbf{\Lambda}_j$, see (6.4), is used with the thresholding scale set to $j = J_\ell - 2$.

The variable parameter in this test is the number of PCG iterations. The metric is as before, the LE Strehl in K band after 500 iterations of the loop. The noise level is fixed to 100 photons per subaperture and frame.

In Figure 8.8 we plot the LE Strehl of our method using a different number of PCG iterations. We variate the number of iterations between 1 and 10, while keeping the other parameters constant. In the 100 photon case, the method reaches a high performance with 7 iterations. Increasing the number of iterations above 8 brings only a minor improvement in the result. Generally, for higher noise levels (lower flux) the method reaches the optimal performance with less iterations.

8.4.4 Performance of GLMS

In this example we investigate the performance of the reconstructor with different number of scales in the GLMS method. As above we concentrate on the 100 photon case only. Default parameters for the test are as in the speed-oriented approach above: the layers are set according to the "speed setup" configuration in Table 8.7, the variable parameters are chosen as in Table 8.12, i.e., $\alpha = 4$, $\alpha_\eta = 0.4$ and gain is 0.4, a scale controlled

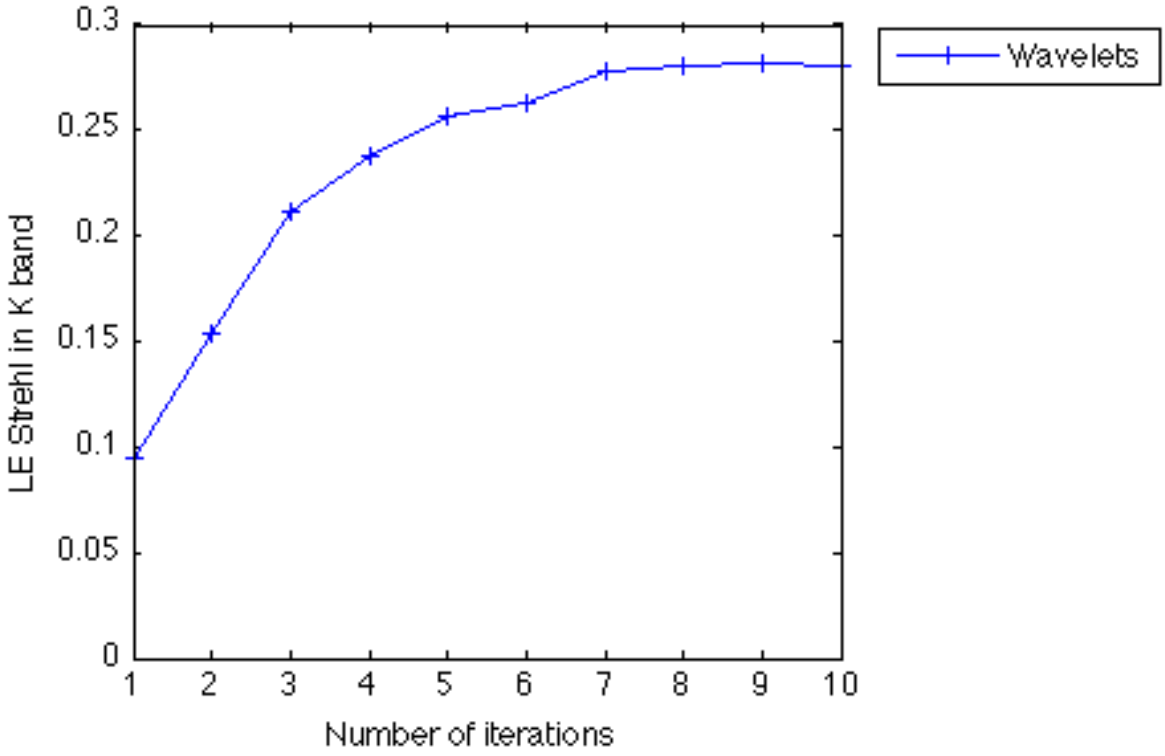

Figure 8.8: Performance of the algorithm with different number of iterations for the 100 photon case.

preconditioner $\mathbf{\Lambda}_j$, see (6.4), is used with the thresholding scale set to $j = J_\ell - 2$.

The variable parameter in this test are the number of GLMS scales $\tilde{J}_1$ and the number of PCG iterations. The metric is as before, the LE Strehl in K band after 500 iterations of the loop. The noise level is fixed to 100 photons per subaperture and frame.

In Figure 8.9 the performance of the method is shown for different number of GLMS scales. The number of GLMS scales $\tilde{J}_1$ varies between 3 and 7, the number of iterations for each scale choice varies between 3 and 10. Generally, using more scales in the GLMS implies a better qualitative performance of the method. For instance, a similar result with $\tilde{J}_1 = 4$ scales and 8 iterations is attained with 5 iterations and $\tilde{J}_1 = 6$ scales. However, with more scales the memory consumption of the overall method increases, see Section 9.2, which might be a drawback on the computational performance of the method. The actual trade-off between computational cost and the memory depends on the architecture of the computational system. Note that no significant improvement is gained by increasing the number of scales from 6 to 7.

8.4.5 Influence of the preconditioner scales

The choice of the thresholding scale j of the preconditioner $\mathbf{\Lambda}_j$, see (6.4), has a significant influence on the behavior of the overall method. When the value is set too high, the method shows a slow, but stable convergence. When the value is set too low, an instability during first few iterations can be observed. At the optimal level, the method shows a fast

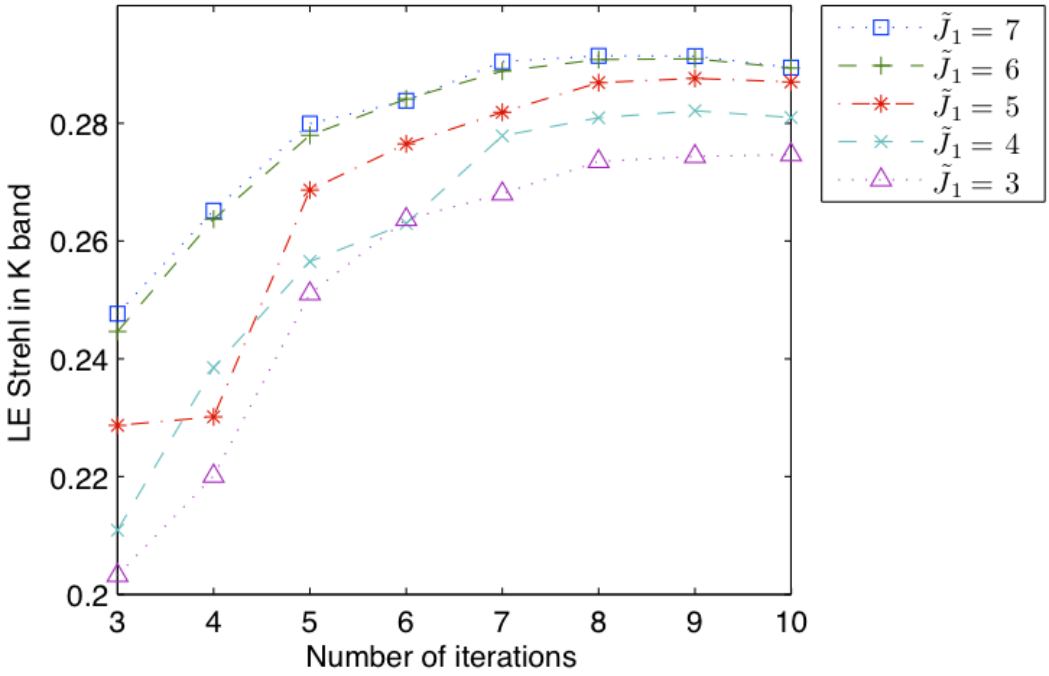

Figure 8.9: Performance of the algorithm with different number of GLMS scales.

and stable convergence.

In this example we investigate the convergence of the algorithm with respect to the thresholding scales j. As above we concentrate on the 100 photon case only. Default parameters for the tests are as in the speed-oriented approach above: the layers are set according to the "speed setup" configuration in Table 8.7, the variable parameters are chosen as in Table 8.12, i.e., $\alpha = 4$, $\alpha_\eta = 0.4$ and gain is 0.4, the number of GLMS scales used in all tests is $\tilde{J}_1 = 4$.

The variable parameter in this test are the thresholding scale j of the preconditioner $\mathbf{\Lambda}_j$ and the number of PCG iterations. The metric is as before, the LE Strehl in K band after 500 iterations of the loop. The noise level is fixed to 100 photons per subaperture and frame.

In Figure 8.10 the performance of the method is shown for different thresholding scales. The scales j vary between $J_\ell - 3$ and $J_\ell - 1$, i.e., between 4 and 6, the number of iterations for each scale choice varies between 1 and 10. The optimal choice for the thresholding scale is $j = J_\ell - 2$. With $j = J_\ell - 1$ the performance is stable, but convergence is slow. Using $j = J_\ell - 3$, the performance of the method is unstable.

Similar behavior is observed for the MCAO reconstructor with the global thresholding preconditioner in Section 8.5.2.

8.4.6 Influence of the discretization grid

In this example we compare the difference in performance of the algorithm between the two layer discretization configurations: "quality" and "speed" in Table 8.7. The difference between the two configurations is in the discretization of the two upper-most layers. In

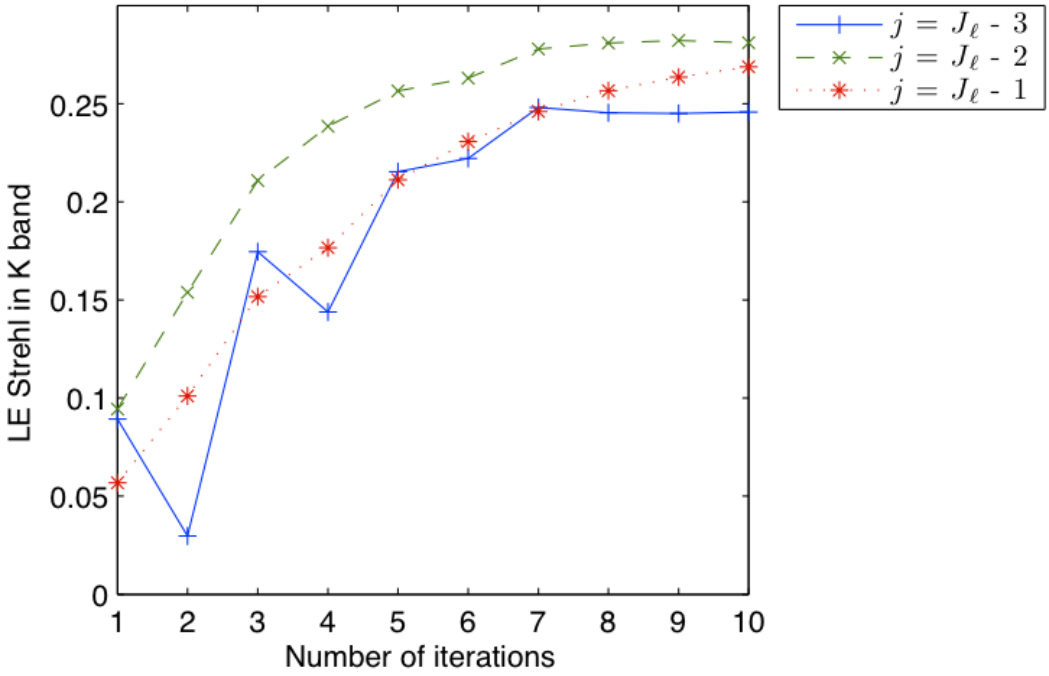

Figure 8.10: Performance of the algorithm with different thresholding scales j of $\mathbf{\Lambda}_j$.

the speed-oriented configuration, the two upper-most layers are discretized on a 128×128 grid of points with a lower resolution of $\delta_\ell = 1$ m, whereas in the quality-oriented setup the resolution is kept constant for all layers at $\delta = 0.5$ m, but the two upper-most layers are discretized over a larger grid of 256×256 points.

In Section 8.4.1 we already compared the two configurations, but with a different number of iterations and choice of the preconditioner. In this section we compare the two discretization configurations head-to-head with all other parameters kept constant.

For demonstrative purposes, we concentrate on the 500, 100 and 20 photon case. The wavelet method is configured as above. The variable parameters of the method, i.e., regularization α, elongation tuning α_η and gain parameters, are summarized in Table 8.12 for different flux levels. The number of GLMS scales used in all tests is $\tilde{J}_1 = 4$. We use the global thresholding preconditioner $\mathbf{\Lambda}_\tau$, see (6.3), with the thresholding parameter given by $\tau = 10^6 \min(\mathrm{diag}(\mathbf{D}))$. The number of PCG iterations is 10 for all test cases.

The variable parameter in this test is the layer discretization configuration: we either set the layers according to the "quality setup" or the "speed setup" configuration in Table 8.7. The metric is as before, the LE Strehl in K band after 500 iterations of the loop. The noise level is varied between 500, 100 and 20 photons per subaperture and frame.

In Table 8.13 the performance of the method is shown for the different discretization configurations, as well as for the reference method, which is FrIM tuned by the ESO.

By reducing the resolution of the highest layers, a small loss in performance can be observed. In this example we see that the performance of the lower resolution configuration is still above the reference case in two out of three cases.

Method	Setup	Flux		
		500	100	20
FrIM by ESO	–	0.323	0.283	0.189
Wavelets	quality	0.324	0.299	0.209
Wavelets	speed	0.319	0.287	0.198

Table 8.13: Performance of the algorithm in "quality" and "speed" setup for different flux, given in number of photons per subaperture and frame.

8.5 MCAO results

8.5.1 Performance against the benchmark method

The benchmark in quality for MCAO is the MVM tuned by the ESO. The reference results were provided by the ESO. The reference method reconstructs three layers at the altitudes of the DMs.

In this section we demonstrate the performance of our method against the benchmark. We investigate the performance of the wavelet method in two configurations: where layers are reconstructed at the DMs and the approach where the full atmosphere is reconstructed, followed by the fitting step. We refer to the two configurations as the 3-layer and the 9-layer approach, respectively.

The wavelet method is configured as follows. In the 3-layer approach we reconstruct $L = 3$ layers of the atmosphere directly at the deformable mirrors. In this configuration, the altitude, optical strength and discretization parameters are given in Table 8.8. In the 9-layer approach we reconstruct $L = 9$ layers for which the layer parameters are given in Table 8.9. The variable parameters for both configurations, i.e., regularization α, elongation tuning α_η and gain parameters, are summarized in Table 8.14 for different flux levels.

Approach	Parameter	Flux					
		500	400	300	200	100	50
3-layer	α	4	4	4	4	4	4
	α_η	0.2	0.4	0.6	0.8	0.8	0.8
	Gain	0.4	0.4	0.4	0.4	0.4	0.4
9-layer	α	16	8	8	4	4	4
	α_η	0.2	0.4	0.6	0.8	0.8	0.8
	Gain	0.4	0.4	0.4	0.4	0.4	0.4

Table 8.14: Parameter settings for MCAO for different flux, given in number of photons per subaperture and frame.

The GLMS algorithm is not used for these tests, i.e., $\tilde{J}_1 = 0$. We use the global thresholding preconditioner $\mathbf{\Lambda}_\tau$, see (6.3), with the thresholding parameter given by $\tau = 10^7 \min(\mathrm{diag}(\mathbf{D}))$. The number of PCG iterations in both configurations is 4; in the 9-layer approach we solve the fitting equation using an unpreconditioned CG algorithm with 4 iterations. For the fitting step of the 9-layer approach we optimize the mirror shapes towards the 25 directions of the probe-stars, see Figure 8.1.

We plot the center LE Strehl (evaluated at the zenith), as well as the average LE Strehl over 25 probe-star directions, see Figure 8.1, in K band after 500 iterations for MCAO versus subaperture photon flux in Figure 8.11.

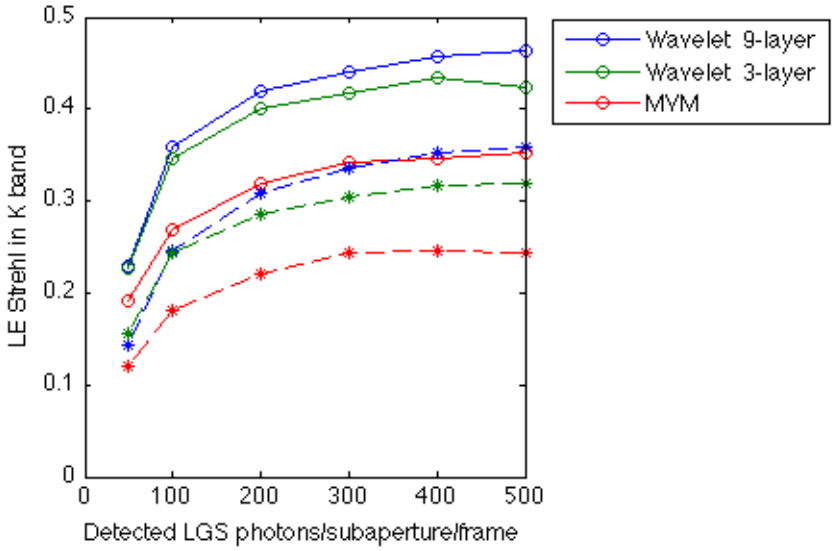

Figure 8.11: MOAO LE K band center Strehl (solid) and average over the FoV (dashed) vs LGS flux.

The wavelet method outperforms the reference method in both configurations by a significant margin. From the two approaches, the method where the full atmosphere is reconstructed shows a better qualitative performance. However, it can be observed that the difference in quality between the two approaches is not very large, whereas the computational cost differs immensely, see Section 9.2. The performance of the method near the center, where most overlap of the measured domains occurs is typically higher than quality further away from the center.

There are several possible reasons why the wavelet method performs better than the MVM. For one, tuning of the wavelet algorithm is easier than tuning the MVM: being an iterative method, the wavelet algorithm requires little pre-computation when changing a parameter. In contrast, the MVM needs to be recomputed every time that a parameter (except for the gain) is modified. Another reason might be the numerical stability of the PCG method against the multiplication by an inverse matrix. Finally, the approximation of the layers in the wavelet basis might be better than the approximation used in the MVM, which is typically discretized with the Zernike polynomials.

8.5.2 Influence of the preconditioner thresholding parameter

Similar to the choice of scale j for the scale controlled preconditioner $\mathbf{\Lambda}_j$, see Section 8.4.5, the global thresholding parameter τ of the preconditioner $\mathbf{\Lambda}_\tau$, see (6.3), has a significant influence on the behavior of the overall method.

In this example we investigate the convergence of the algorithm with respect to the thresholding parameter τ. We concentrate on the 100 photon case for the 9-layer approach only. Default parameters for the tests are as above: the layers are set according to the

configuration in Table 8.9, the variable parameters are chosen as in Table 8.14, i.e., $\alpha = 4$, $\alpha_\eta = 0.8$ and gain is 0.4, no GLMS scales are used. We use the same optimization directions for the fitting step as the probe-star directions; to solve the fitting equation we use 4 iterations of the CG method.

The variable parameter in this test are the thresholding parameter τ of the preconditioner $\mathbf{\Lambda}_\tau$ and the number of PCG iterations. We control the parameter τ by scaling it with the smallest value of the diagonal of $\mathbf{D}$ via a parameter α_J, i.e., $\tau = \alpha_J \min(\text{diag}(\mathbf{D}))$. The metric is the average LE Strehl in K band after 500 iterations over the 25 probe-star directions. The noise level is fixed to 100 photons per subaperture and frame.

In Figure 8.12 the performance of the method is shown for different choices of parameter α_J and PCG iterations. For α_J we use the values $\alpha_J = 0$ (Jacobi preconditioner), $10^4, 10^5, 10^6, 10^7, 10^8$ and 10^9; additionally we plot the CG result where no preconditioner is used. The number of iterations varies between 4, 6 and 10.

From the plot we observe that the performance of the standard Jacobi preconditioner ($\alpha_J = 0$) is rather unstable. Quality begins to improve when a larger thresholding parameter is used, with an optimal value at 10^7 for PCG with four and six iterations, and at 10^8 for PCG with ten iterations. By increasing the parameter α_J beyond its optimal value, we notice that the performance of the algorithm tends towards the non-preconditioned CG result, which is the right-most value in the plot in Figure 8.12. Note that the higher number of CG iterations leads to a higher result in quality. The balancing effect that τ has on the behavior of the method, whether closer to the Jacobi PCG or an unpreconditioned CG method, is observed.

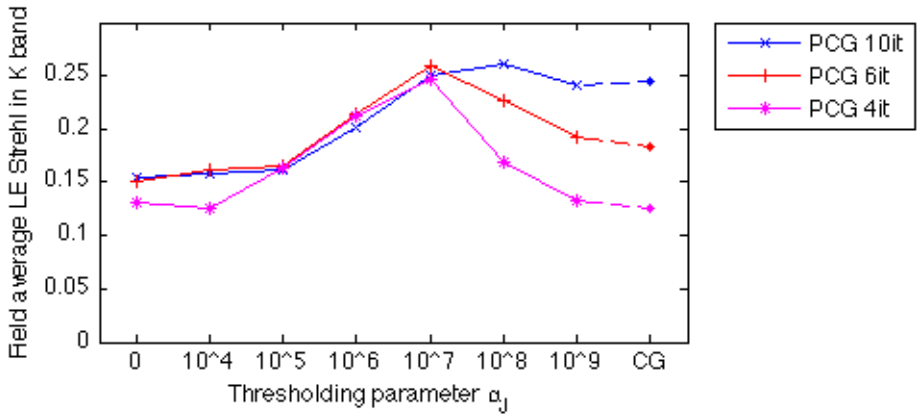

Figure 8.12: Sensitivity of PCG with respect to thresholding parameter τ, where $\tau = \alpha_J \min(\text{diag}(\mathbf{D}))$. Standard CG results with the same number of iterations are displayed at the far right side of each graph.

8.5.3 Choice of optimization directions

The choice of optimization directions impacts the quality and the computational performance of the method. In this section we illustrate how different choices of the optimization directions over the field of view affect the quality of the reconstruction. For this test we consider a so called "high flux" example - where the number of detected photons is very large and the noise error does not factor into the reconstruction.

The number of detected photons per subaperture and frame is 10000 and 50000 for LGS and NGS sensors, respectively. The read noise is $3e^-$ and $5e^-$ per pixel and frame for each detector, respectively. Moreover, the LGS detectors do not suffer from spot elongation (i.e., the noise for LGS is modeled as for the NGS).

The method parameter are chosen as follows: We focus on the 9-layer approach, i.e., we reconstruct $L = 9$ layers given in Table 8.9. The variable parameters for these tests are given by the regularization $\alpha = 20$ and the gain of 0.4; spot elongation tuning α_η is not used, since spot elongation is not taken into account. Neither GLMS method nor preconditioning are used for these tests; the atmospheric reconstruction problem is solved with 20 CG iterations. The subsequent fitting problem is solved with 4 iterations of the CG method.

The results from these tests are compared to the the results from the 3-layer approach and the MVM for the high flux case. The 3-layer approach is configured to reconstruct $L = 3$ layers defined in Table 8.8. Moreover, the parameter $\alpha = 20$ and gain of 0.4 are used. The atmospheric tomography problem is solved with 20 CG iterations; no preconditioning or the multi-scale approach is active for this test.

We examine how the different arrangements of the optimization directions $\bar{\theta}_1, ..., \bar{\theta}_{\mathcal{I}}$ effect the reconstruction quality. Note that the first sub-problem, the atmospheric tomography, is solved with the same method in the same configuration for all of the tests. However, due to the closed loop regime, different mirror shapes lead to different input data for the method over the duration of a simulation.

The performance metric for these tests is the average LE Strehl over separation from the zenith after 500 iterations of the loop. The average LE Strehl over separation is computed as the average LE Strehl value over those directions in the 5×5 grid that have the same radial distance from the zenith.

In Figure 8.13 we plot the different grids of optimization directions. The 5×5 grid refers to the configuration where the directions of interest are aligned with the probe-star directions. We examine two 3×3 grids: one that is placed in between of the 5×5 grid (green triangle) and another rotated by 45 degrees, where the optimization directions coincide with 9 probe-star directions (red rhombus), see Figure 8.13, left. We also test more alternative configurations, where we use either four or five optimization directions, see Figure 8.13, right.

The LE Strehl over separation for the 5×5 and 3×3 configurations is plotted in Figure 8.14; for the other configurations in Figure 8.15. Each plot is supplemented with the reference results as well as the results of the 3-layer configuration.

From the plots in Figure 8.14 we observe that the results produced with the 3×3 optimization grids are close to those obtained with the full 5×5 grid. In fact, both results outperform the 3 layer and the MVM tests. The result from the points chosen in between of the full grid (green triangles) is close to the 5×5 result near the zenith, but the performance decays for points that are further away from center. The results from the rotated grid (red rhombus) is comparable with the 5×5 grid at those points that are within a radius of 1 arcmin, but the performance drops for the points outside of this radius.

The results in Figure 8.15, that are associated to the grids in Figure 8.13, right, are less successful than those in Figure 8.14. The performance spikes for the configurations where the optimization directions coincide with the probe-stars located at a radius of 40

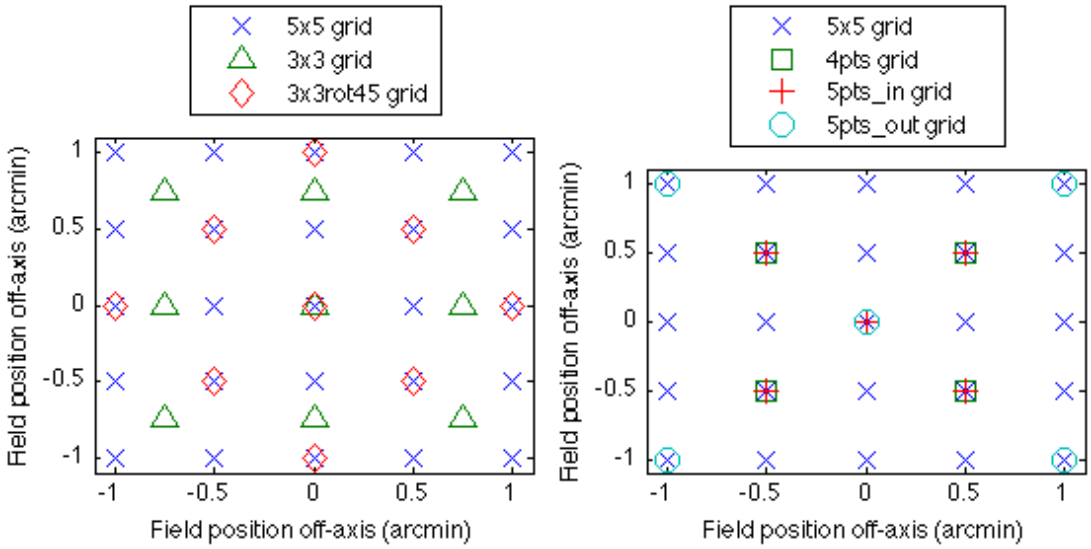

Figure 8.13: MCAO optimization directions over the field of view (2×2 arcsec).

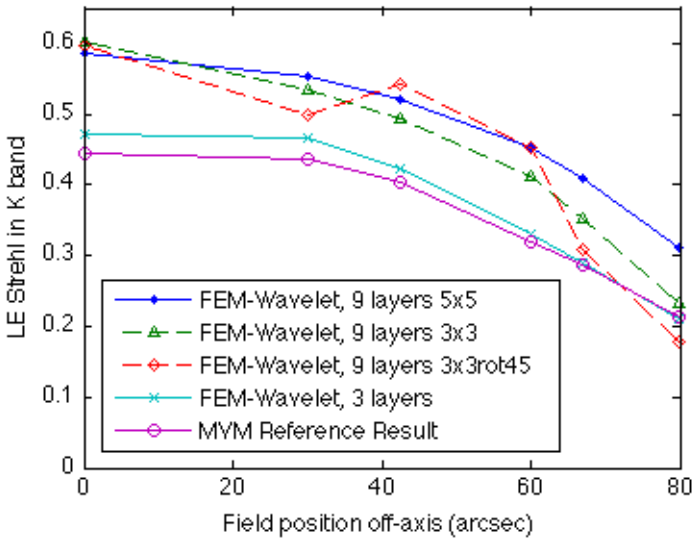

Figure 8.14: High flux LE Strehl vs. separation for optimization grids in Figure 8.13, left. The average off-axis value is plotted.

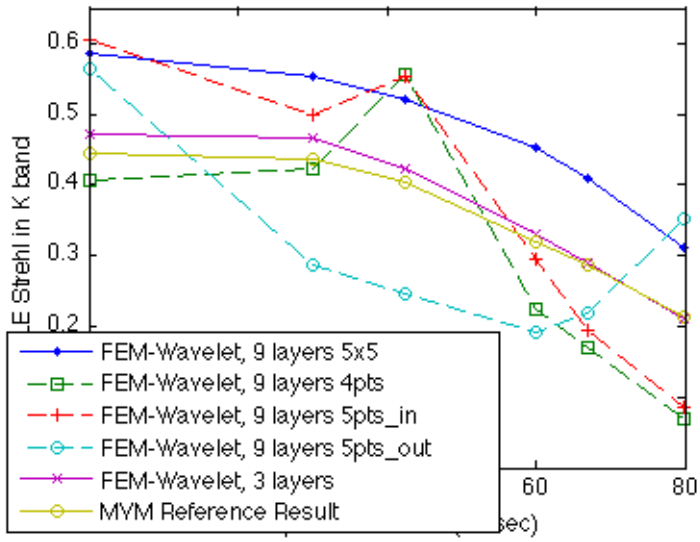

Figure 8.15: High flux LE Strehl vs. separation for optimization grids in Figure 8.13, right. The average off-axis value is plotted.

arcsec (4pts and 5pts_in); for the configuration that also optimizes at the center (5pts_in), the performance is also good near the zenith. The quality of the reconstruction of the 5pts_out configuration reflects the configuration of the optimization grid: only the outermost and the center directions show a good performance, whereas the quality in between is relatively poor.

Chapter 9

Numerics: speed

9.1 Computational efficiency

Numerical efficiency of a method depends on several factors. The number of floating point operations (FLOP) is one of them. The FLOP determines the number of additions and multiplications of real numbers.

For a modern system, however, FLOP is not the only indicator of numerical efficiency. Current CPUs use a multi-core architecture with a shared memory cache. Thus, properties such as parallelization and small memory requirements play a vital role as well. In this chapter we present the numerical performance of the wavelet method in terms of three factors: FLOP, parallelization and memory requirements.

In Sections 9.1.1 and 9.1.2 we show how by using a special representation of the operators we achieve a method that is computationally cheap and light in terms of memory consumption. Then, in Sections 9.1.3-9.1.6 we discuss the parallelization capabilities of the wavelet method.

In the following Section 9.2 we provide the computational estimates of our algorithm and in Section 9.3 we demonstrate the reconstruction speed of the method on specific computational systems.

9.1.1 Matrix-free operators

An important feature of the wavelet method is the ability to represent almost all of its components, see Algorithm 7.1, in a matrix-free way. The two direct benefits of such a representation are: a reduction of the number of floating point operations and a reduction in storage of the components.

The three main components of the wavelet method are the Shack-Hartmann operator, the bilinear interpolation and the wavelet transform. We discuss the matrix-free representation of these operators below.

Matrix-free Shack-Hartmann

In its discrete form, the Shack-Hartmann operator $\Gamma = (\Gamma^x, \Gamma^y)$, see (2.38), is a mapping between $(n_s + 1)^2$ nodal values of the wavefront discretized in the continuous piecewise

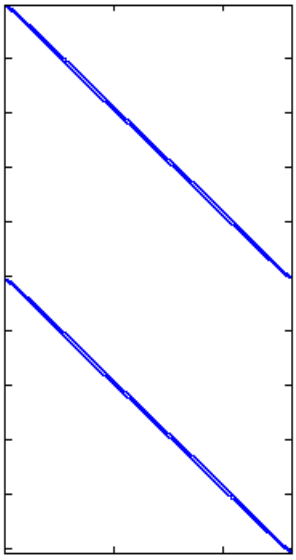

Figure 9.1: Sparse structure of the Shack-Hartmann matrix for the 84×84 sensor ($n_s = 84$). The 40320 non-zero elements of the matrix are marked.

bilinear basis and $2n_s^2$ values that correspond to the average x- and y-slopes over the the n_s^2 subapertures of a WFS.

Thus, the linear operator Γ has a matrix representation of dimension $2n_s^2 \times (n_s+1)^2$. The full matrix requires $2n_s^2(n_s+1)^2$ units of storage and the computational cost of multiplying the full matrix with a wavefront vector of dimension $(n_s+1)^2$ is $4((n_s+1)^2-1)n_s^2 = 4n_s^3(n_s+2)$. Note that this corresponds to a quadratic order in FLOP and memory consumption for the input and output vectors that scale as $\mathcal{O}(n_s^2)$.

Due to the local support of the piecewise bilinear basis, one function element, i.e., a function which has the value 1 at one of the $(n_s+1)^2$ nodes and 0 in the rest, interacts with at most four subapertures. In the same sense, the measurement in one subaperture is influenced only by the four nodal values of the wavefront function at the corner points of the subaperture. Thus, one row of the matrix Γ, which corresponds to one subaperture, has only four non-zero elements.

This sparse structure of the matrix leads to a significant reduction in storage, as the total number of non-zero elements is only $8n_s^2$. Further, only 4 multiplications and 3 additions need to be performed per row, so the FLOP reduces to $14n_s^2$. Even with the additional overhead memory requirements to store the positions of the non-zero elements, the sparse representation of the matrix induces a linear computational complexity and storage requirement. The sparse structure of Γ is illustrated in Figure 9.1.

However, it is possible to reduce the computational cost and the storage of Γ even further by utilizing the special structure of the operator. To this end we replace the matrix by a routine which performs the same action as the matrix.

To compute the x-slopes, we split the evaluation (2.38) into two steps. First, we compute the differences of the nodal values and store them in a temporary variable of dimension $(n_s+1) \times n_s$,

$$t_{ij} = \varphi_{i,j+1} - \varphi_{ij}, \tag{9.1}$$

for $0 \le i \le n_s$ and $0 \le j < n_s$. Consequently, we compute the averages,

$$s^x_{ij} = (t_{ij} + t_{i+1,j})/2, \tag{9.2}$$

for $0 \le i < n_s$ and $0 \le j < n_s$.

Analogously, the y-slopes are computed by evaluating the differences in the other direction,

$$t_{ij} = \varphi_{i+1,j} - \varphi_{ij}, \tag{9.3}$$

for $0 \le i < n_s$ and $0 \le j \le n_s$, stored in the temporary variable of dimension $n_s \times (n_s+1)$, and then computing the averages,

$$s^y_{ij} = (t_{ij} + t_{i,j+1})/2, \tag{9.4}$$

for $0 \le i < n_s$ and $0 \le j < n_s$.

The number of operations to compute the differences is $(n_s + 1)n_s$ and to compute the averages is $2n_s^2$ per direction. The total number of operations in both directions is therefore $6n_s^2 + 2n_s$. This is less operations than in the sparse matrix representation.

Moreover, we use no memory to store the operator itself. Instead, we allocate memory of dimension $(n_s + 1)n_s$ to store the temporary results. We reuse the memory for the x- and y-computations.

The comparison of the three ways to represent the Shack-Hartmann operators are presented in Table 9.1. In terms of both computational cost and memory, the matrix-free representation is the most efficient one.

Representation	FLOP	Memory		
		Operator	Temporary	Total
Full matrix	$2n_s^2(n_s + 1)^2$	$4n_s^3(n_s + 2)$	0	$4n_s^3(n_s + 2)$
Sparse matrix	$14n_s^2$	$8n_s^2$	0	$8n_s^2$
Matrix-free	$6n_s^2 + 2n_s$	0	$n_s^2 + n_s$	$n_s^2 + n_s$

Table 9.1: Computational cost and memory requirements for the Shack-Hartmann operator, represented as a full matrix, sparse matrix and a matrix-free routine. Input and output vectors are of dimension $(n_s + 1)^2$ and $2n_s^2$, respectively.

Matrix-free bilinear interpolation

A bilinear interpolation operator $P_{g\ell}$, see (5.42), is a mapping between the bilinear coefficients of the layer ℓ and the bilinear coefficients of the wavefront in direction g. The layer coefficients are defined as the nodal values of a continuous piecewise bilinear function over a grid of $2^{J_\ell} \times 2^{J_\ell}$ points, the wavefront coefficients over a grid of $(n_s + 1) \times (n_s + 1)$ points projected onto layer ℓ in direction of the guide star g according to (5.16) or (5.17) for LGS or NGS, respectively.

Bilinear interpolation is a concatenation of two linear interpolation operations, see (5.42) and (5.43). We illustrate these two steps in Figure 9.2. The layer grid of $2^{J_\ell} \times 2^{J_\ell}$ is given by black circles; the wavefront grid projected onto the layer, shifted according to the guide star direction and scaled due to the cone effect (in this illustration) is given by blue squares.

In the first step we interpolate in the x-direction: using the known function values at the black circles we perform linear interpolation to determine the function values at red triangles. In the second step we interpolate in the y-direction: using the computed values at the red triangles we perform linear interpolation to determine the function values at the blue squares.

Observe that if the projected grid spacing is smaller than the layer spacing, then the number of interpolated points in the first interpolation step (red triangles in Figure 9.2) may be smaller than $(n_s+1)^2$. We denote the dimensions of the grid for the temporary values after the first interpolation (red triangles in Figure 9.2) by $(\bar{n}_s+1)\times(n_s+1)$, where $0<\bar{n}_s\leq n_s$.

To perform linear interpolation as a routine, we store $2(n_s+1)$ values: for each nodal point index $0,...,n_s$ we store an index $i=0,...,2^{J_\ell}-1$ associated to the interval $[x_i,x_{i+1}]$ in the layer grid where the linear function is interpolated and the corresponding weight, $(x-x_i)/(x_{i+1}-x_i)$, where $x\in[x_i,x_{i+1}]$ is the coordinate of the interpolation point, see (5.39). Therefore, the storage of a bilinear interpolation operator is $4(n_s+1)$, for the x- and y-indices and weights.

Using pre-computed interpolation weights, a linear interpolation step (5.39) requires 3 operations per interpolation point. Thus, the total number of operations for a bilinear interpolation is $3(\bar{n}_s+1)(n_s+1)+3(n_s+1)^2=3(\bar{n}_s+n_s+2)(n_s+1)$ where $0<\bar{n}_s\leq n_s$ refers to the dimension of the temporary grid. The exact number of operations depend on the direction of the guide star, the altitude of the layer and whether the cone effect is present, i.e., factors that determine the value of $\bar{n}_s$. The upper bound for the number of operations is $6(n_s+1)^2$.

In addition to the storage of the operator, we also store the temporary values from the first linear interpolation (red triangles in Figure 9.2) in a temporary variable of dimension $(\bar{n}_s+1)(n_s+1)$. The temporary storage is at most $(n_s+1)^2$.

In total, the number of floating point operations for a bilinear interpolation step is $3(\bar{n}_s+n_s+2)(n_s+1)\leq 6(n_s+1)^2$ and the memory requirement for this step is $(\bar{n}_s+1)(n_s+1)+4(n_s+1)\leq(n_s+1)^2+4(n_s+1)$. The operation is of linear complexity in terms of the computational cost and storage for a grid of $(n_s+1)^2$ interpolation points. We summarize the results on the computational cost and memory requirements of this operation in Table 9.2.

Representation	FLOP	Memory		
		Operator	Temporary	Total
Full matrix	$(2^{2J_\ell+1}-1)(n_s+1)^2$	$2^{2J_\ell}(n_s+1)^2$	0	$2^{2J_\ell}(n_s+1)^2$
Matrix-free	$6(n_s+1)^2$	$4(n_s+1)$	$(n_s+1)^2$	$(n_s+1)^2+$ $4(n_s+1)$

Table 9.2: Computational cost and memory requirements for the bilinear interpolation operator, represented as a full matrix and a matrix-free routine. Input and output vectors are of dimension 2^{2J_ℓ} and $(n_s+1)^2$, respectively.

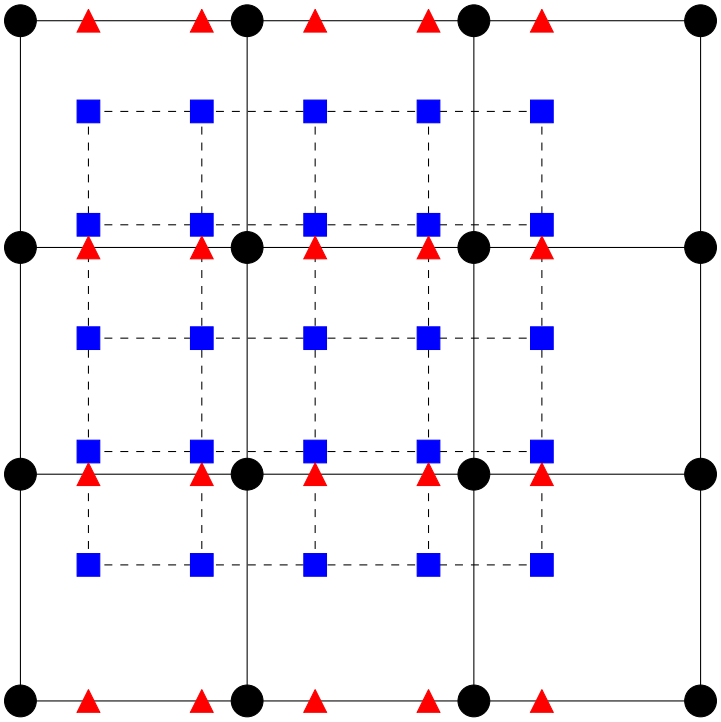

Figure 9.2: Interpolation of the layer grid (black circles) onto the projected wavefront grid (blue squares). In the first linear interpolation the intermediate values (red triangles) are computed, in the second linear interpolation step the values at the wavefront grid are determined.

Matrix-free wavelet transform

The DWT and the inverse DWT are computed via a cascading sequence of operations (3.114) and (3.115) in which the direct transform matrix W_j and the inverse transform matrix W_j^T are applied, respectively, at scales $j = 0, ..., J-1$.

Matrices W_j and W_j^T have an equivalent representation as a set of convolution operations with the wavelet filter coefficients, (3.93) and (3.94), respectively. A discrete wavelet transform is more efficient if it is represented as a convolution rather than as sparse matrices.

The computational cost of the direct or the inverse DWT at scale $j = 0, ..., J-1$ originates from applying W_j or W_j^T, respectively, twice, as a set of convolution operations with a matrix of coefficients of dimension $n_j \times n_j = 2^{j+1} \times 2^{j+1}$. Let p be the length of the wavelet filter. Then, applying W_j to one column of the coefficient matrix requires $(2p-1)n_j$ operations (p multiplications and $p-1$ additions). Applying W_j twice to the whole matrix of coefficients therefore requires $2(2p-1)n_j^2$ operations; the same cost holds for the inverse transform.

Thus the total cost of the direct or the inverse DWT on all scales is given by the sum of these operations,

$$\sum_{j=0}^{J-1} 2(2p-1)n_j^2 = \sum_{j=0}^{J-1} 2(2p-1)2^{2(j+1)} = \frac{8}{3}(2p-1)(2^{2J}-1), \tag{9.5}$$

where 2^{2J} is the total number of coefficients transformed from the bilinear to the wavelet domain, or vice versa. For our choice of Daubechies 3 wavelets with filter length $p = 6$, the number of floating point operations is $(88/3)(2^{2J}-1)$.

In terms of memory, the DWT requires the storage of the wavelet filter of length p, which is constant with respect to the grid size. For our purposes, we consider the memory for storing the filter to be negligible. Additionally, we use a temporary variable of dimension 2^{2J} to store the temporary result from applying one of the wavelet transform operators, W_j or W_j^T, for the direct and the inverse transform, respectively. In fact, the temporary variable is of dimension $2^{2(j+1)}$ at scale $j = 0, ..., J-1$, but we may reuse the memory reserved for the largest temporary variable at scale $j = J-1$ for other scales.

In summary, the direct or the inverse DWT of a variable with 2^{2J} coefficients, requires $(8/3)(2p-1)(2^{2J}-1)$ floating point operations, where p is the filter length and 2^{2J} units of memory. Both the computational cost and the memory consumption of the DWT scale linearly with the dimension of the variable. We outline the results on the computational cost and memory requirements of this operation in Table 9.3.

Representation	FLOP	Memory		
		Operator	Temporary	Total
Full matrix	$(2^{2J+1}-1)2^{2J}$	2^{4J}	0	2^{4J}
Matrix-free	$(8/3)(2p-1)(2^{2J}-1)$	0	2^{2J}	2^{2J}

Table 9.3: Computational cost and memory requirements for the discrete wavelet transform, represented as a full matrix and a matrix-free routine. Input and output vectors are both of dimension 2^{2J}.

We summarize the computational cost and the memory consumption of all operators of the whole wavelet method in Section 9.2.

9.1.2 Sequential operators

The computational performance gain of the matrix-free operators described above can only be attained when all operators are applied sequentially, instead of as a single pre-computed matrix. We illustrate this point on the atmospheric tomography operator $\mathbf{A}$.

On the one hand, we analyze the sparse matrix obtained by pre-computing the discrete operator $\widehat{\mathbf{A}}\mathbf{W}^{-1}$ in the wavelet domain, see Section 5.4. In fact, numerically this is a close approximation to the operator $\widetilde{\mathbf{A}}$ that we would obtain if we were to discretize the atmospheric tomography operator fully in the wavelet domain, as described in Section 5.3.1.

On the other hand, we compute the total cost and storage requirements of all components in the operator decomposition,

$$\begin{aligned}&\widehat{\mathbf{A}}\mathbf{W}^{-1} \\ &= \begin{bmatrix} \Gamma_1 & & \\ & \ddots & \\ & & \Gamma_G \end{bmatrix} \begin{bmatrix} P_{11}^{\mathrm{LGS}} & \cdots & P_{1L}^{\mathrm{LGS}} \\ \vdots & & \vdots \\ P_{G1}^{\mathrm{NGS}} & \cdots & P_{GL}^{\mathrm{NGS}} \end{bmatrix} \begin{bmatrix} \delta_1^{-1} W^{-1} & & \\ & \ddots & \\ & & \delta_L^{-1} W^{-1} \end{bmatrix}.\end{aligned} \tag{9.6}$$

Invoking the operator corresponds to three sequential operations: applying an inverse wavelet transform on all layers, followed by projecting the wavefronts and subsequently applying the Shack-Hartmann operator. For more details on the components of this decomposition, see Section 5.4.

For this comparison let us focus on a specific example: the MOAO "speed" configuration with $L = 9$ layers, see Table 8.7, and $G = 9$ guide stars with high resolution WFSs ($n_s^2 = 84^2$ subapertures) assigned to them, of which $G_{\mathrm{LGS}} = 6$ are LGSs. Note that in this configuration all layers $\ell = 1, ..., L$ are discretized with $2^{2J_\ell} = 128^2$ grid points.

The dimensions of this operator are $2Gn_s^2 \times 2^{2J_\ell}L = 18 \cdot 84^2 \times 9 \cdot 128^2$. In the sparse matrix form, roughly $2.49 \cdot 10^8$ elements are non-zero coefficients, which corresponds to 1.3% of all elements of the matrix. The sparse structure of this matrix is illustrated in Figure 9.3.

In the decomposed form (9.6), the number of floating point operations originates from the sum of the following operations: $(88/3)(2^{2J_\ell} - 1)L$ for the inverse DWT, $2^{2J_\ell}L$ for the scalar multiplication by δ_ℓ^{-1}, at most $6(n_s + 1)^2(L - 1)G$ for the bilinear interpolations on layers $\ell = 2, ..., L$ (ground layer grid is aligned with the wavefront grid, so no interpolation is necessary), $(n_s + 1)^2(L - 1)G$ for the accumulation operations of the wavefront components at all layers, and finally $(6n_s^2 + 2n_s)G$ for the Shack-Hartmann operators.

In this example, the number of operations is therefore at most $8.03 \cdot 10^6$. This results in an improvement against the sparse matrix formulation by a factor of 60. The actual number of operations is lower for the matrix-free representation, due to the cone effect in the laser guide stars, which makes the bilinear interpolation operations cheaper.

Moreover, the number of units of memory required to store the matrix-free components of the decomposition (9.6) is a total of: $2^{2J_\ell}L$ for the inverse DWT, at most $[(n_s + 1)^2 +$

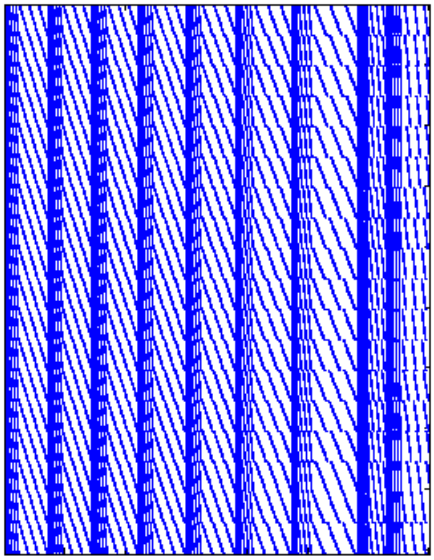

Figure 9.3: Sparse structure of the atmospheric tomography operator $\widetilde{\mathbf{A}}$ in the wavelet domain as a pre-computed matrix. The 1.3% non-zero elements of the matrix are marked.

$4(n_s + 1)](L - 1)G$ for the bilinear interpolation, $(n_s^2 + n_s)G$ for the Shack-Hartmann operator and $(n_s + 1)^2 G$ for the temporary variable to store the wavefront coefficients.

In this example, this corresponds to at most $8.21 \cdot 10^5$ units of memory. This is a reduction in memory consumption over the sparse matrix representation by a factor of 300, even if we neglect the extra memory requirements to store the positions of the non-zero elements of the sparse matrix.

9.1.3 Parallelization

Further computational efficiency of the wavelet method can be gained by parallelization. The computational cost of certain operations can be shared over several parallel processes, which reduces the execution time of the algorithm. The wavelet algorithm allows two types of parallelization: global and local.

Global parallelization is motivated by the block decomposition of the operators discussed in Section 9.1.2. This decomposition introduces a natural way to split the work between different processes, such that each block is applied in parallel. We discuss global parallelization of the algorithm in Section 9.1.4.

Local parallelization, on the other hand, refers to the parallelization of the individual blocks. It was shown in Section 9.1.1 that many operations in the matrix-free representation of the operators can be computed in parallel. Local parallelization of the routines is described in Section 9.1.5.

The two strategies are not mutually exclusive. In fact, the combination of the two types leads to a very efficient parallelization scheme of the whole algorithm. We discuss an example where global and local parallelization strategies are applied simultaneously in Section 9.1.6.

With parallel computing two important points must be taken into account: data

communication and writing conflicts. We discuss both points below.

Data communication refers to the exchange of intermediate data between running parallel processes. On a distributed memory system, data communication introduces a delay in computation, since certain time is spent on synchronizing the data between the processes. In contrast, on a shared memory system, the same physical memory is accessed by all processes, hence data communication is avoided.

A parallel algorithm which must frequently exchange a sizable amount of data, as the wavelet algorithm does (explained in Section 9.1.4), is best suited for a shared memory system or a system where data communication can be performed especially fast. An example of a shared memory system is a modern multi-core CPU with shared cache. An example of a system with fast distributed memory communication is a multi-CPU system on the same board.

Writing conflicts occur when two parallel processes try to write data to the same block of memory simultaneously. In this situation, certain errors may occur to the written data, or the speed at which the data is written may be reduced. To avoid these uncertainties, it is ideal to structure the parallel code in such a way that avoids writing conflicts, i.e., each parallel processes should write the data to a separate block of memory. Conversely, no conflicts arise when several processes are simultaneously reading data from the same block of memory.

9.1.4 Global parallelization

The wavelet algorithm, Algorithm 7.1, and the underling atmospheric tomography problem (5.49) have a block representation, where each block is an operation associated either to one layer, one guide star or both. In the global parallelization scheme, these blocks are computed on separate processes in parallel.

We illustrate how the global parallelization is applied to the left-hand side operator $\mathbf{M}$, see (5.50). The operator $\mathbf{M}$ is applied once per a PCG iteration and is the computationally heaviest part of the whole algorithm. Other parts of the algorithm consist of similar steps that appear in $\mathbf{M}$ and therefore can be parallelized analogously.

Parallelization of the operator $\mathbf{M}$ is presented in the Algorithm 9.1. We refer to Section 5.4 for the details on the structure the operator. We explain the components of the algorithm below.

The input and the output variables $\mathbf{c}$ and $\mathbf{q} = \mathbf{M}\mathbf{c}$ are vectors of wavelet coefficients. Two temporary variables are used in the algorithm, $\tilde{\mathbf{s}}$ and $\tilde{\boldsymbol{\varphi}}$, where the intermediate Shack-Hartmann measurements and wavefronts are stored, respectively.

The first operation in $\mathbf{M}$ is the inverse wavelet transform. It is computed in parallel for all layers in steps 2-3. The output of the transform is written to the variable $\mathbf{q}$. In this part of the algorithm, $\mathbf{q}$ is used as a temporary variable to store the bilinear coefficient of the layers. Note that in both domains, wavelet and bilinear, the number of coefficients of the layer variable is $\sum_{\ell=1}^{L} 2^{2J_\ell}$.

The following operation in $\mathbf{M}$ is the projection of incoming wavefronts in their respective guide star directions from layers. The projected wavefronts are computed in two steps: 6 and 8.

First, the incoming wavefront component of the ground layer is determined in step 6. No interpolation is performed at this step, since the wavefront and the ground layer

Algorithm 9.1: Apply $\mathbf{M}$

Input : $\mathbf{c} = (c_\ell)_{\ell=1}^L$ (wavelet coefficient vector)
Output : $\mathbf{q} = (q_\ell)_{\ell=1}^L = \mathbf{Mc}$ (wavelet coefficient vector)
Temporary: $\tilde{\mathbf{s}} = (\tilde{s}_g)_{g=1}^G$ (measurement vector)
$\tilde{\boldsymbol{\varphi}} = (\tilde{\varphi}_g)_{g=1}^G$ (wavefront vector)

begin in parallel $\ell = 1, ..., L$
  $q_\ell = \delta_\ell^{-1} c_\ell$ /* multiply by scaling constant */
  $q_\ell = W^{-1} q_\ell$ /* apply inverse DWT */
distribute $\mathbf{q}$ to all processes /* data communication */
begin in parallel $g = 1, ..., G$
  $\tilde{\varphi}_g = P_{g1} q_1$ /* apply projection */
  for $\ell = 2, ..., L$ **do**
    $\tilde{\varphi}_g = \tilde{\varphi}_g + P_{g\ell} q_\ell$ /* apply projection */
  $\tilde{s}_g = \Gamma_g \tilde{\varphi}_g$ /* apply SH */
  if $1 \le g \le G_{\mathrm{LGS}}$ **then**
    $\tilde{s}_g = \mathcal{M}_g \tilde{s}_g$ /* apply mask */
    $\tilde{s}_g = (I - T)\tilde{s}_g$ /* remove TT */
    $\tilde{s}_g = \widetilde{C}_g^{-1} \tilde{s}_g$ /* apply noise covariance */
    $\tilde{s}_g = (I - T)\tilde{s}_g$ /* remove TT */
    $\tilde{s}_g = \mathcal{M}_g \tilde{s}_g$ /* apply mask */
  else
    $\tilde{s}_g = \mathcal{M}_g \tilde{s}_g$ /* apply mask */
    $\tilde{s}_g = \sigma_g^{-2} \tilde{s}_g$ /* multiply by noise variance */
  $\tilde{\varphi}_g = \Gamma_g^T \tilde{s}_g$ /* apply transpose SH */
distribute $\tilde{\boldsymbol{\varphi}}$ to all processes /* data communication */
begin in parallel $\ell = 1, ..., L$
  $q_\ell = P_{1\ell}^T \tilde{\varphi}_1$ /* apply transpose projection */
  for $g = 2, ..., G$ **do**
    $q_\ell = q_\ell + P_{g\ell}^T \tilde{\varphi}_g$ /* apply transpose projection */
  $q_\ell = W q_\ell$ /* apply DWT */
  $q_\ell = \delta_\ell^{-1} q_\ell$ /* multiply by scaling constant */
  $q_\ell = q_\ell + D_\ell c_\ell$ /* add regularization term */

grids are aligned, i.e., the ground layer components of $\tilde{\varphi}_g$ are determined directly from q_1 without additional computation. Next, in step 8, for each guide star direction and layer, bilinear interpolation $P_{g\ell}$ is performed and the wavefront component is accumulated.

The wavefronts are computed in parallel for each guide star direction $g = 1, ..., G$. In doing so, the writing conflict is avoided: data is read simultaneously by all processes from the variable $\mathbf{q}$ and written to the individual wavefront $\tilde{\varphi}_g$ by each process g.

Before the projection step, the vector $\mathbf{q}$ is distributed to G processes in step 4. On a shared memory system, this step is omitted.

The following few operations associated to a guide star direction are also computed in parallel. The first is the Shack-Hartmann operator, applied in step 9. Next, depending on whether the guide star is an LGS or an NGS, the noise statistics are applied in steps 11-15 or 17-18, respectively. Finally, the transposed Shack-Hartmann operator is applied in step 19.

The resulting wavefronts $\tilde{\boldsymbol{\varphi}}$ are projected back to layers in steps 22 and 24. These and the subsequent operations are computed in parallel on L processes; this way the writing conflicts are avoided. The variable $\tilde{\boldsymbol{\varphi}}$, which is read by all processes, is synchronized on a distributed memory architecture in step 20. On a shared memory system, this step is omitted.

Observe that the transposed projection operation is different it terms of the computational cost for each layer $\ell = 1, ..., L$. For the ground layer $\ell = 1$, the computation is very cheap due to the grid alignment of the ground layer and the wavefront grid. For all altitude layer $\ell \geq 2$, the wavefronts must be interpolated back onto the layer grids. Due to the cone effect, this operation is cheaper for layers with a higher altitude for the LGS directions $1 \leq g \leq G_{\text{LGS}}$ than for the layers closer to the ground.

At last, the forward wavelet transform is performed on all layers in steps 25-26. Finally, we add the regularizing term in step 27, where the wavelet coefficients are multiplied with a diagonal matrix on each layer.

In total, the number of parallel processes in this scheme is determined by the maximum number of layers and guide star directions, i.e., by $\max(L, G)$.

Parallelization configuration presented in Algorithm 9.1 is therefore especially efficient for the LTAO and MOAO systems in the speed-oriented setup, see Section 8.1 and Table 8.7.

In this setup, the number of layers and guide star directions is 9. Therefore, the number of parallel processes is $\max(L, G) = 9$. Moreover, due to the same resolution of all sensors ($n_s^2 = 84^2$) and the same resolution of all layers ($2^{2J_\ell} = 128^2$), the work in Algorithm 9.1 is split relatively equally between each parallel process.

For the MCAO 3-layer setup, see Section 8.1, where $L = 3$ layers are reconstructed from $G = 9$ sensors, where layers and sensors have different resolution, parallelization can be improved. A more efficient parallelization is obtained by using the special structure of the problem and local parallelization of the operators, which we discuss in the following Section 9.1.5. An adapted algorithm for MCAO 3-layer reconstructor is presented in Section 9.1.6.

9.1.5 Local parallelization

In the local parallelization scheme, we compute the individual block operators in parallel, as opposed to the block parallelization of the global scheme.

In Section 9.1.1 it was observed that the matrix-free operators of the wavelet algorithm share the same basic characteristics: they consist of a sequence of repeated instructions applied to a grid of values that can be computed independently of each other.

In parallel computing, this is referred to as the single instruction, multiple data (SIMD) paradigm. We explain SIMD parallelization in terms of the three operators from Section 9.1.1: the Shack-Hartmann operator, the bilinear interpolation and the discrete wavelet transform.

Parallel Shack-Hartmann

The Shack-Hartmann operator consists of two subsequent operations: computing the differences, steps (9.1) and (9.3), followed by computing the averages, steps (9.2) and (9.4), for x- and y-directions, respectively. The operations are performed over a grid of values; the computation in each operation is performed independently for each grid value.

Thus, computing the differences, which costs 1 floating point operation, can be performed simultaneously for all $2(n_s+1)n_s$ grid values of both x- and y-directions. Subsequently, computing the averages, which costs 2 floating point operations, can be performed simultaneously for $2n_s^2$ grid values.

In total, the Shack-Hartmann operator costs 3 floating point operations on $2(n_s+1)n_s$ parallel processes. In comparison, the operator costs $6n_s^2 + 2n_s$ operations to apply on a single process.

Parallel bilinear interpolation

Bilinear interpolation also consists of two subsequent operations: a linear interpolation in the x-direction (red triangles in Figure 9.2), followed by a linear interpolation in the y-direction (blue squares in Figure 9.2). Each interpolated value is determined independently from the others.

Therefore, linear interpolation, which requires 3 floating point operations per one interpolation point, can be performed in parallel over $(\bar{n}_s+1)(n_s+1)$ points for the first operation and over $(n_s+1)^2$ points for the second operation.

Neglecting the computational advantage of the reduced grid dimensions due to the cone effect, i.e., $\bar{n}_s = n_s$, bilinear interpolation costs 6 floating point operations on $(n_s+1)^2$ parallel processes. In comparison, the operator costs at most $6(n_s+1)^2$ operations to apply as a single process.

Parallel wavelet transform

Also the discrete wavelet transform consists of two subsequent operations at each scale $j = 0, ..., J-1$. For the direct transform, the operator W_j is applied twice in sequence to matrices of dimensions $n_j \times n_j = 2^{j+1} \times 2^{j+1}$, see (3.114); for the inverse transform, the same action is performed with the operator W_j^T, see (3.115).

Each entry of the output matrix under the action of W_j or W_j^T can be computed independently from the others. The cost of computing one such entry is the cost of the convolution operation, $(2p-1)$, where p is the length of the wavelet filter. For the Daubechies 3 wavelets where $p = 6$, the required number of floating point operations is therefore 11. This computation can be performed over n_j^2 matrix entries in parallel.

Thus, the cost of applying the DWT at scale j, i.e., the cost of applying W_j or W_j^T twice, is $2(2p-1)$ floating point operations over n_j^2 parallel processes. In summary, the cost of the DWT on all J scales is at most $2(2p-1)J$ floating point operations over at most $n_{J-1}^2 = 2^{2J}$ processes, where $2^J \times 2^J$ are the dimensions of the largest matrix. In comparison, the cost of the DWT on a single process is $(8/3)(2p-1)(n_{J-1}^2 - 1)$.

Local parallelization summary

An outline of the local parallelization capabilities for the three operators is presented in Table 9.4.

Operator	FLOP single process	FLOP parallel	Parallel processes
Shack-Hartmann	$6n_s^2 + 2n_s$	3	$2(n_s+1)n_s$
Bilinear interpolation	$6(n_s+1)^2$	6	$(n_s+1)^2$
DWT at scale j	$2(2p-1)n_j^2$	$2(2p-1)$	n_j^2
DWT at all scales	$(8/3)(2p-1)(n_{J-1}^2-1)$	$2(2p-1)J$	n_{J-1}^2

Table 9.4: Local parallelization overview.

We point out that the results presented in this section show only the theoretical advantage of local parallelization. The number of available parallel processes and overhead costs limit this theoretical advantage. The actual advantage depends on the computing system architecture

A modern CPU has a finite number of cores, which is typically much lower than the required number of processes for full local parallelization of the method. To put things into perspective, full theoretical parallelization advantage of the DWT on a grid of $2^{2J_\ell} = 128^2$ points is attainable with 128^2 parallel processes, whereas the maximum number of cores of an off-the-shelf CPU is roughly 15 at the time of writing.

However, we indicate that the number of parallel processes in Table 9.4 is only an upper bound for local parallelization. Advantage is still attainable from parallelization on a number of processes that is lower than the theoretical limit in Table 9.4.

Further, parallelization always incurs overhead costs. Typically, the efficiency of parallelization depends on the amount of data to be processed. Due to the overhead costs, parallelization is more efficient for larger amounts of data, whereas for smaller amounts, parallel computing might even be slower than single process computing.

We demonstrate these effects of local parallelization on a CPU for the wavelet transform at different scales j in Section 9.3.4. There we present the benefit of local parallelization on a system with up to eight parallel processes and show that parallel computing pays off only at the higher scales j.

Finally, we point out that the SIMD structure of the operators can be especially exploited on a General Purpose GPU (GPGPU), where a stream of processes perform

the same operations on a large set of data. Tests of the algorithm on a GPGPU are left for future work.

9.1.6 Combined parallelization strategy

Depending on the AO system requirements and the computing system specifications, it is possible to combine the global and local parallelization strategies. To demonstrate this idea we focus on the MCAO 3-layer (DM on mirrors) configuration, where the system specifications are defined in Section 8.1.

In this configuration we reconstruct $L = 3$ layers, where the ground layer is discretized at a finer resolution, on a grid of 128×128 points, than the two altitude layers, discretized on grids of 64×64 points. The MCAO configuration utilizes $G = 9$ guide stars, of which $G_{\text{LGS}} = 6$ are LGS with high resolution 84×84 subaperture sensors and $G_{\text{NGS}} = G - G_{\text{LGS}} = 3$ are NGS with 2×2 or 1×1 tip-tilt sensors associated to them.

As before we focus our attention on the parallelization of the left-hand side operator $\mathbf{M}$. Although the Algorithm 9.1 can be applied to this AO system configuration directly, parallel work can be split in a more uniform way with additional local parallelization.

In Algorithm 9.2 we propose a parallelization configuration that is based on the following observations: the ground layer has more grid points, hence the wavelet transform is computationally more expensive for the ground than the altitude layers; the projection operation on the altitude layers is more expensive than on the ground layer, due to the bilinear interpolation; the tomography operator associated to the tip-tilt sensors has a very sparse pre-computed matrix representation, see below.

The tomography operator for the tip-tilt sensors, i.e., the concatenation of the projection and the Shack-Hartmann operators towards the NGS directions, is pre-computed as a single matrix,

$$\boldsymbol{\gamma} := \begin{bmatrix} \Gamma_{G_{\text{LGS}}+1} & & \\ & \ddots & \\ & & \Gamma_G \end{bmatrix} \begin{bmatrix} P^{\text{NGS}}_{G_{\text{LGS}}+1,1} & \cdots & P^{\text{NGS}}_{G_{\text{LGS}}+1,L} \\ \vdots & & \vdots \\ P^{\text{NGS}}_{G1} & \cdots & P^{\text{NGS}}_{GL} \end{bmatrix}. \tag{9.7}$$

The matrix $\boldsymbol{\gamma}$ is of dimension $12 \times \sum_{\ell=1}^{L} 2^{2J_\ell}$ for one 2×2 and two 1×1 sensor, i.e., it has a very low rank. Moreover, due to (5.29), only the boundary values of the observed domains $\Omega^{\text{NGS}}_{g'\ell}$, see (5.17), affect the measurements; hence the matrix is very sparse.

We point out the differences of this implementation of the algorithm to the general Algorithm 9.1 below. Parallelization of this algorithm is optimized towards a shared memory CPU with 7 cores.

First, we utilize 6 processes to perform the inverse DWT: 4 are assigned to the ground layer (step 4), which has a larger grid dimension, and 1 to each of the altitude layers (step 6).

In the following steps, 7 through 25, operations associated to the guide star directions are performed. Here we parallelize the algorithm on 7 processes: 1 process is associated to each of the LGSs and 1 process to all of the tip-tilt sensors. In step 9 the few tip-tilt measurements are determined by sparse matrix-vector multiplication. At the same time, the incoming wavefronts towards the LGSs are computed in steps steps 11-13.

When these computations are complete, in step 14 we synchronize all processes. At this point, the vector $\mathbf{q}$, which was used as a temporary variable to store the bilinear

Algorithm 9.2: Apply $\mathbf{M}$ for MCAO 3-layer configuration

Input : $\mathbf{c} = (c_\ell)_{\ell=1}^{L}$ (wavelet coefficient vector)
Output : $\mathbf{q} = (q_\ell)_{\ell=1}^{L} = \mathbf{Mc}$ (wavelet coefficient vector)
Temporary: $\tilde{\mathbf{s}} = (\tilde{s}_g)_{g=1}^{G}$ (measurement vector), $\tilde{\boldsymbol{\varphi}} = (\tilde{\varphi}_g)_{g=1}^{G}$ (wavefront vector), $\tilde{\mathbf{q}} = (\tilde{q}_\ell)_{\ell=2}^{L}$ (wavelet coefficient vector)

begin in parallel $\ell = 1, ..., L$
  $q_\ell = \delta_\ell^{-1} c_\ell$ /* scaling constant */
  if $\ell = 1$ **then**
    $q_\ell = W^{-1} q_\ell$ on 4 processes /* inverse DWT (ground) */
  else
    $q_\ell = W^{-1} q_\ell$ on 1 process /* inverse DWT (altitude) */

begin in parallel $g = 1, ..., G_{\mathrm{LGS}} + 1$
  if $g = G_{\mathrm{LGS}} + 1$ **then**
    $(\tilde{s}_{g'})_{g'=G_{\mathrm{LGS}}+1}^{G} = \boldsymbol{\gamma}\mathbf{q}$ /* projection, SH (NGS) */
  else
    $\tilde{\varphi}_g = P_{g1} q_1$ /* projection (LGS) */
    for $\ell = 2, ..., L$ **do**
      $\tilde{\varphi}_g = \tilde{\varphi}_g + P_{g\ell} q_\ell$ /* projection (LGS) */

begin in parallel $g = 1, ..., G_{\mathrm{LGS}} + 1$
  if $g = G_{\mathrm{LGS}} + 1$ **then**
    $(\tilde{s}_{g'})_{g'=G_{\mathrm{LGS}}+1}^{G} = (\sigma_{g'}^{-2} \tilde{s}_{g'})_{g'=G_{\mathrm{LGS}}+1}^{G}$ /* noise variance */
    $\mathbf{q} = \boldsymbol{\gamma}^T (\tilde{s}_{g'})_{g'=G_{\mathrm{LGS}}+1}^{G}$ /* transpose projection, SH (NGS) */
  else
    $\tilde{s}_g = \Gamma_g \tilde{\varphi}_g$ /* SH (LGS) */
    $\tilde{s}_g = \mathcal{M}_g \tilde{s}_g$ /* mask (LGS) */
    $\tilde{s}_g = (I - T) \tilde{s}_g$ /* remove TT (LGS) */
    $\tilde{s}_g = \widetilde{C}_g^{-1} \tilde{s}_g$ /* noise covariance (LGS) */
    $\tilde{s}_g = (I - T) \tilde{s}_g$ /* remove TT (LGS) */
    $\tilde{s}_g = \mathcal{M}_g \tilde{s}_g$ /* mask (LGS) */
    $\tilde{\varphi}_g = \Gamma_g^T \tilde{s}_g$ /* transpose SH (LGS) */

begin in parallel $\ell = 1, ..., L$
  if $\ell = 1$ **then**
    for $g = 1, ..., G_{\mathrm{LGS}}$ **do**
      $q_\ell = q_\ell + P_{g\ell}^T \tilde{\varphi}_g$ /* transpose projection (LGS) */
    $q_\ell = W q_\ell$ on 2 processes /* DWT (ground) */
  else
    begin in parallel
      for $g = 1, ..., G_{\mathrm{LGS}}/2$ **do**
        $q_\ell = q_\ell + P_{g\ell}^T \tilde{\varphi}_g$ /* transpose projection (LGS) */
      for $g = G_{\mathrm{LGS}}/2 + 1, ..., G_{\mathrm{LGS}}$ **do**
        $\tilde{q}_\ell = \tilde{q}_\ell + P_{g\ell}^T \tilde{\varphi}_g$ /* transpose projection (LGS) */
    $q_\ell = q_\ell + \tilde{q}_\ell$ /* transpose projection (LGS) */
    $q_\ell = W q_\ell$ /* DWT (altitude) */
  $q_\ell = \delta_\ell^{-1} q_\ell$ /* scaling constant */
  $q_\ell = q_\ell + D_\ell c_\ell$ /* add regularization term */

layer coefficients is free. Hence, we write to $\mathbf{q}$ the transposed tip-tilt tomography data in step 17 after applying the NGS noise statistics in step 16. In parallel, the standard operations for the LGS sensors are performed in steps 19-25.

Next, we apply the transposed projection operators for the LGS wavefronts, followed by the wavelet transform in steps 29-39. The result of the transposed projection is accumulated to the vector $\mathbf{q}$, which was initialized in step 17.

The two expensive operations here are the transposed projections at the altitude layers and the wavelet transform at the ground layer. Conversely, transposed projection at the ground layer and the wavelet transform at the altitude layers are cheap, due to the grid alignment of the ground layer with the wavefront grid and the smaller grid dimensions of the altitude layers, respectively.

We utilize 6 processes to parallelize these operations: 2 processes are assigned to each layer. At the ground, one process is used to compute the sum over all wavefronts in step 29 and then 2 processes are used to perform the wavelet transform in step 30.

At the same time, 2 processes are used per each altitude layer to compute the transposed bilinear interpolation. To avoid writing conflicts each of these 4 processes writes the interpolated result into a different block of memory, q_2, q_3 or temporary variables $\tilde{q}_2, \tilde{q}_3$, in steps 34 and 36, after which the results are combined in step 37. Consequently, the wavelet transform is computed on each of the altitude layers without parallelization.

The numerical performance of this scheme is compared to the generic Algorithm 9.1 in Section 9.3.3.

9.2 Computational estimates

In this section we present the computational estimates of the algorithm and its components. The estimates are presented in terms of three attributes: the total number of floating point operations, the number of floating point operations per one parallel process and the memory consumption. For parallelization we focus on the global strategy, see Section 9.1.4, only.

We present the computational estimates in the following hierarchical way. We start with the smallest building blocks of the method: the operator blocks in Section 9.2.1. Then we dissect the major operations of the wavelet Algorithm 7.1 in Sections 9.2.2-9.2.6. Finally, in Section 9.2.7, we demonstrate the computational estimates for one reconstructor configuration. The estimates for the other systems are derived analogously from Sections 9.2.1-9.2.6.

9.2.1 Block operator estimates

The estimates of the block operators of the wavelet method, see, e.g., Algorithm 9.1, are summarized in Table 9.5. We distinguish between the memory used to store an operator and the temporary memory used by the operator during the computation.

The computational estimates are given in terms of two factors: the number of WFS subapertures n_s^2 and the layer discretization grid dimension 2^{2J_ℓ}, where J_ℓ is the number of wavelet scales at layer ℓ. We make a few comments regarding the operator estimates below.

Operator	FLOP	Memory		
		Operator	Temporary	Total
Γ_g	$6n_s^2 + 2n_s$	0	$n_s^2 + n_s$	$n_s^2 + n_s$
Γ_g^T	$6n_s^2 + 2n_s$	0	0	0
$\mathcal{M}_g$	0	n_s	0	n_s
$I - T$	$4n_s^2$	0	0	0
$\widetilde{C}_g^{-1}$	$6n_s^2$	$3n_s^2$	0	$6n_s^2$
$\widetilde{C}_{g'}^{-1}$	$2n_s^2$	0	0	0
P_{g1}	0	0	0	0
$P_{g\ell}, \ell \geq 2$	$6(n_s+1)^2$	$4(n_s+1)$	$(n_s+1)^2$	$(n_s+1)^2+$ $4(n_s+1)$
P_{g1}^T	0	0	0	0
$P_{g\ell}^T, \ell \geq 2$	$10(n_s+1)^2$	$4(n_s+1)$	$(n_s+1)^2$	$(n_s+1)^2+$ $4(n_s+1)$
W	$(88/3)(2^{2J_\ell} - 1)$	0	2^{2J_ℓ}	2^{2J_ℓ}
W^{-1}	$(88/3)(2^{2J_\ell} - 1)$	0	2^{2J_ℓ}	2^{2J_ℓ}
D_ℓ	2^{2J_ℓ}	J_ℓ	0	J_ℓ

Table 9.5: Computational cost and memory usage estimates of the operators. In the case of asymptotically constant memory consumption with respect to the system dimensions, we write 0.

In Section 9.1.1 we derived the estimates for the Shack-Hartmann operator Γ_g. The transposed Shack-Hartmann operator Γ_g^T can also be represented as a matrix-free routine. The computational estimates for it are similar to the direct operator, but the operator requires no temporary storage.

The operator $\mathcal{M}_g$ is the mask. It assigns the value 0 to all entries of the measurement vector s_g that correspond to the inactive subapertures. Thus, no explicit computational cost is involved. The operator uses n_s units of storage; moreover the operator must be stored only once per WFS geometry.

Tip-tilt removal operator $I - T$, see Section 5.3.2, computes the average slopes over the x- and the y-measurements. The average values, corresponding to the tip and the tilt of the wavefront, are subtracted from each measurement.

The inverse noise covariance operator $\widetilde{C}_g^{-1}$ for the LGS directions $g = 1, ..., G_{\text{LGS}}$ contains the information on the correlation between the x-to-x, x-to-y, y-to-x and y-to-y measurements. Since, x-to-y correlation coincides with y-to-x (the matrix is symmetric), we need to store only three correlation vectors of dimension n_s^2. Moreover, these vectors need to be stored only once per WFS geometry and laser launch position. The matrix is applied by multiplying the data with the three vectors. For the NGS directions $g' = G_{\text{LGS}} + 1, ..., G$, only the scalar variance $\sigma_{g'}^{-2}$ of the matrix $\widetilde{C}_{g'}^{-1} = \sigma_{g'}^{-2} I$ is stored.

When the ground layer grid is aligned with the wavefront grid, the projection operator and its transposed, P_{g1} and P_{g1}^T, respectively, require no computational cost or storage. For higher altitudes, $P_{g\ell}$ is a bilinear interpolation, for which we derived an upper bound on the estimates in Section 9.1.1. The transposed projection operator $P_{g\ell}^T$ also has a matrix-free representation. The computational cost of the transposed operation

is slightly higher. The same interpolation indices and weights are stored for both direct and transposed operators. Moreover, the same temporary variable can be reused for both operators.

In Section 9.1.1 we derived the estimates for the forward and the inverse wavelet transforms, W and W^{-1}, respectively. The wavelet transform costs are given for the Daubechies 3 wavelets with a filter length of $p = 6$ elements, see (9.5).

The operator D_ℓ is the diagonal approximation of the inverse turbulence covariance, see Section 5.2. We only need to store the different weights at each scale, i.e., J_ℓ numbers.

9.2.2 Left-hand side operator estimates

Based on the estimates in Table 9.5, we are able to analyze the computational cost of the components of the wavelet method. In particular, we are able to provide the estimates for the left-hand side operator $\mathbf{M}$, see (5.50), of the equation (5.49) and its global parallelization in Algorithm 9.1.

Here and below we make a few simplifying assumptions: all guide stars use the same WFS geometry with n_s^2 subapertures and all layers are discretized over the same number of 2^{2J_ℓ} points.

The computational cost estimates of Algorithm 9.1 are presented in Table 9.6. The total cost is an estimated upper bound.

Step in Alg. 9.1	FLOP total	FLOP per process
2	$2^{2J_\ell}L$	2^{2J_ℓ}
3	$(88/3)L(2^{2J_\ell}-1)$	$(88/3)(2^{2J_\ell}-1)$
6	0	0
8	$7(L-1)G(n_s+1)^2$	$7(L-1)(n_s+1)^2$
9	$G(6n_s^2+2n_s)$	$6n_s^2+2n_s$
11-18	$(14G_{\text{LGS}}+2G_{\text{NGS}})n_s^2$	$14n_s^2$
19	$G(6n_s^2+2n_s)$	$6n_s^2+2n_s$
22-24	$[10(L-1)G+L(G-1)](n_s+1)^2$	$(11G-1)(n_s+1)^2$
25	$(88/3)L(2^{2J_\ell}-1)$	$(88/3)(2^{2J_\ell}-1)$
26	$2^{2J_\ell}L$	2^{2J_ℓ}
27	$2\cdot 2^{2J_\ell}L$	$2\cdot 2^{2J_\ell}$
Total	$(188/3)L2^{2J_\ell}+[26G+$ $17(L-1)G+L(G-1)](n_s+1)^2$	$(188/3)2^{2J_\ell}+[26+7(L-1)$ $+(11G-1)](n_s+1)^2$

Table 9.6: Computational estimates of the operator $\mathbf{M}$ in Algorithm 9.1.

The estimate shows that the total cost of applying the forward operator is of linear complexity with respect to the dimensions of the system, which are stated in terms of two factors: the number of subapertures n_s^2 and the number of layer discretization points 2^{2J_ℓ}.

Note that the number of layer discretization points depends linearly on the number of subapertures. The number of scales J_ℓ is chosen such that the layer grid overlaps the

wavefront grid, i.e., such that the condition

$$2^{2(J_\ell-1)} < (n_s+1)^2 \leq 2^{2J_\ell} \tag{9.8}$$

holds.

The corresponding memory consumption of the operator $\mathbf{M}$ is presented in Table 9.7. The memory is reused between the direct and the transposed projection operators and the forward and the inverse transforms, respectively. The total memory consumption is an estimated upper bound. This estimate shows that also the memory requirement for the operator is linear with respect to the number of layer discretization points and subapertures.

Operator / variable	Operator memory	Temporary memory
$\tilde{\mathbf{s}}$	0	$2Gn_s^2$
$\tilde{\boldsymbol{\varphi}}$	0	$G(n_s+1)^2$
Γ_g	0	$(n_s^2+n_s)G$
$\mathcal{M}_g$	n_s	0
$\widetilde{C}_g^{-1}$	$3G_{\mathrm{LLP}}n_s^2$	0
$P_{g\ell}, P_{g\ell}^T$	$4(L-1)G(n_s+1)$	$(L-1)G(n_s+1)^2$
W, W^{-1}	0	$2^{2J_\ell}L$
D_ℓ	$J_\ell L$	0
Total	$2^{2J_\ell}L + (L+6)G(n_s+1)^2 + (4L-3)G(n_s+1) + J_\ell L + n_s$	

Table 9.7: Memory estimates of the operator $\mathbf{M}$. Parameter $G_{\mathrm{LLP}} \leq G_{\mathrm{LGS}} \leq G$ is the number of laser launch positions.

9.2.3 Right-hand side operator estimates

The computation of the right-hand side of equation (5.49), which appears in step 5 of the Algorithm 7.1, is similar to applying $\mathbf{M}$. In fact, the right-hand side operator consists of steps 11-26 of Algorithm 9.1.

Following the results in Table 9.6, the upper bound on the total number of floating point operations is $(97/3)L2^{2J_\ell} + [20G + 10(L-1)G + L(G-1)](n_s+1)^2$. Moreover, the number of floating point operations per process is $(97/3)2^{2J_\ell} + [20 + (11G-1)](n_s+1)^2$.

9.2.4 GLMS estimates

The two main steps of the GLMS Algorithm 6.1 are: solving the sub-problem in step 2 and computing the residual in step 5.

The computational cost of solving the sub-problem is determined by the dimensions of the matrix $\widetilde{M}$, which is the restriction of $\mathbf{M}$ to the coarse scales of the ground layer. The dimensions of $\widetilde{M}$ are $2^{2\tilde{J}_1} \times 2^{2\tilde{J}_1}$, where $\tilde{J}_1$ is the number of coarse scales of the ground layer.

We solve the sub-problem by matrix vector multiplication. The inverse matrix is full, hence, the cost of storing the matrix is $2^{4\tilde{J}_1}$ units of memory and the cost of applying it is $2^{2\tilde{J}_1}(2^{2\tilde{J}_1+1}-1)$. Both the storage and the multiplication are quadratic with respect to the coarse scale vector dimension. Nevertheless, the number of coarse scales of the ground layer $\tilde{J}_1$ is chosen independently from the number of scales of the ground layer J_1. Therefore, this does not affect the complexity of the wavelet method with respect to the AO system dimensions.

To put things into perspective, results in Sections 8.3 and 8.4 have been obtained with $J_\ell = 7$ scales on $L = 9$ layers and $\tilde{J}_1 = 4$ coarse ground layer scales. In this configuration, the cost of applying the GLMS matrix is $130,816$ floating point operations, which is less than any vector operation of the full problem, where the vector of wavelet coefficients of all layers and scales has $147,456$ elements.

We indicate that the matrix vector multiplication is parallelizable. Moreover, the dimensions of the matrix can be reduced by filtering out those wavelets that are explicitly in the nullspace of the operator, see Section 7.3 for more details.

The cost of computing the residual outweighs the cost of solving the sub-problem, as it is closely related to the cost of invoking the operator $\mathbf{M}$. This cost can be slightly reduced by using the knowledge that the input vector has zero entries at layers $\ell \geq 2$.

In this case, the inverse DWT and the projection operations for the altitude layers are omitted, i.e., step 3 for $\ell \geq 2$ and step 8 of the Algorithm 9.1. The total cost of the operation is $[(91/3)(L+1)+2L]2^{2J_\ell} + [26G + 10(L-1)G + L(G-1)](n_s+1)^2$ and the cost per processor is at most $(188/3)2^{2J_\ell} + [26 + (11G-1)](n_s+1)^2$ operations. This is less than the cost of applying $\mathbf{M}$ to an arbitrary vector.

9.2.5 PCG estimates

One iteration of the PCG Algorithm 3.2, excluding the preconditioner and the left-hand side operator, consists of 8 vector operations.

We store the frequency-dependent preconditioner, see Section 6.2, as a vector of $2^{2J_\ell}L$ elements. The cost of multiplying this diagonal matrix with a vector is therefore $2^{2J_\ell}L$ floating point operations.

Thus, one iteration of the PCG Algorithm for the wavelet method costs $9 \cdot 2^{2J_\ell}L$ floating point operations, plus the cost of applying $\mathbf{M}$, which is given in Table 9.6. The algorithm is parallelizable in terms of the vector operations; parallelization of the left-hand side operator $\mathbf{M}$ is discussed in Sections 9.1.4-9.1.6.

This estimate shows that the cost of applying one PCG iteration is dominated by the computational cost of invoking the operator $\mathbf{M}$.

9.2.6 Fitting step estimates

The fitting step, see Section 7.4, consists of two operations: the transformation of the reconstructed layer coefficients from the wavelet domain to the bilinear domain, followed by the AO-specific fitting operation.

For SCAO and 3-layer MCAO reconstructors, only the wavelet transform is performed. For LTAO and MOAO, a subsequent projection operation determines the mirror shapes. Finally, for MCAO 9-layer, a subsequent fitting equation is solved using the CG algorithm.

The wavelet transform at all layers costs $(91/3)L2^{2J_\ell}$ floating point operations and can be performed on L layers in parallel.

The subsequent projection for LTAO and MOAO systems costs additional $7(L-1)Mn_a^2$ operations, where M is the number of DMs and n_a^2 the number of actuators. Whereas the LTAO systems utilizes $M = 1$ DM, the MOAO system running in open loop can use multiple mirrors, where the shape of each mirror can be computed in parallel at the cost of $7(L-1)n_a^2$ floating point operations.

It can be observed that the wavelet method is a more compact solution to MOAO than the MVM in terms of the computational system requirements. By separating the atmospheric reconstruction and the fitting step, one additional mirror shape is derived at a marginal additional cost, or is obtained "for free" with parallelization, using the wavelet algorithm. On the other hand, the MVM requires one full real time computing system for each additional mirror.

For the 9-layer MCAO system, the key operators are the direct and the transposed projections towards the directions of interest. The cost is determined by the number of reconstructed layers L, the number of mirrors M and the number of directions of interest $\mathcal{I}$.

Let n_a^2 be the number of actuators of the ground DM. Then, the cost to compute the right-hand side vector of equation (7.17) is $(91/3)L2^{2J_\ell} + [7(L-1)\mathcal{I} + 10(M-1)\mathcal{I} + M(\mathcal{I}-1)]n_a^2$. In parallel this cost reduces to $(91/3)2^{2J_\ell} + [7(L-1) + 11\mathcal{I}]n_a^2$ operations per process.

The cost of applying the left-hand side operator in one CG iteration is $[17(M-1)\mathcal{I} + M(\mathcal{I}-1)]n_a^2$. The corresponding parallel cost of the operator is $[7(M-1) + (11\mathcal{I}-1)]n_a^2$ per process. The number of parallel processes is determined by $\max(L, M, \mathcal{I})$.

We summarize the costs of the fitting step for these five reconstructors in Table 9.8. The number of CG iterations for the fitting sub-problem is denoted by n_{iter}.

Reconstructor	FLOP total	FLOP per process
SCAO / MCAO 3-layer	$(91/3)L2^{2J_\ell}$	$(91/3)2^{2J_\ell}$
LTAO / MOAO	$(91/3)L2^{2J_\ell} + 7(L-1)Mn_a^2$	$(91/3)2^{2J_\ell} + 7(L-1)n_a^2$
MCAO 9-layer	$(91/3)L2^{2J_\ell}+$ $[7(L-1)\mathcal{I} + 10(M-1)\mathcal{I}+$ $M(\mathcal{I}-1)]n_a^2+$ $n_{\text{iter}}[7(M-1) + (11\mathcal{I}-1) + 8M]n_a^2$	$(91/3)L2^{2J_\ell}+$ $[7(L-1) + 11\mathcal{I}]n_a^2+$ $n_{\text{iter}}[7(M-1) + (11\mathcal{I}-1) + 8]n_a^2$

Table 9.8: Computational estimates of the fitting step.

This estimate highlights the difference in the computational cost between the 3-layer and the 9-layer configurations for MCAO. Apart from the higher computational cost in the atmospheric tomography step, the 9-layer approach is more costly in the fitting step. The cost of the fitting step depends on the number of evaluation directions $\mathcal{I}$. By using more evaluation directions, the computational cost of the method increases.

9.2.7 Estimates of the full algorithm

We provide the computational estimates of the full algorithm for the LTAO reconstructor in the speed-oriented setup. The configuration of the reconstruction algorithm is given in Section 8.1 and the speed-oriented discretization of the layers is outlined in Table 8.7. Estimates for the other AO systems can be derived in a similar way by combining the results from Sections 9.2.1-9.2.6.

In the following configuration, the LTAO utilizes G guide stars. To each guide star a WFS in an identical configuration of n_s^2 subapertures is assigned. One ground DM is used with n_a^2 actuators. Furthermore, each of the L layers is discretized on a grid of 2^{2J_ℓ} points.

The solution method utilizes both the GLMS algorithm and the frequency-dependent preconditioner. The GLMS algorithm is defined for a vector of $2^{2\tilde{J}_1}$ elements, corresponding to the $\tilde{J}_1$ coarse scales of the ground layer.

Based on the estimates that we obtained in Sections 9.2.1-9.2.6, the overall cost of the wavelet Algorithm 7.1 for LTAO is given in Table 9.9. We state the total number of floating point operations and the number of operations per process for the global parallelization strategy, see Section 9.1.4.

The total computational cost of the algorithm is therefore split into three factors: the fixed computational cost, the cost per iteration and the number of iterations n_{iter}. The full algorithm is linear with respect to the dimensions of the AO system, 2^{2J_ℓ} and n_s^2.

Memory requirements of the LTAO algorithm are presented in Table 9.10. We distinguish between two types of memory: memory used to store the algorithm variables and memory used to store and evaluate the operators of the method. For more details on the algorithm variables, see the wavelet Algorithm 7.1, the GLMS Algorithm 6.1 and the PCG Algorithm 3.2.

Memory estimates to store the left-hand side operator $\mathbf{M}$ are given in Table 9.7. Other components of the algorithm, except for the GLMS sub-problem, reuse parts of the operator $\mathbf{M}$ and therefore require no additional memory.

The GLMS matrix $\widetilde{M}^{-1}$ for the sub-problem is stored as a separate variable. It requires at most $2^{4\tilde{J}_1}$ units of memory, which is a factor that grows quadratically with the dimensions of the ground layer coarse scale vector $2^{2\tilde{J}_1}$. Note that otherwise the memory requirements are linear with respect to the the dimensions of the system, 2^{2J_ℓ} and n_s^2.

In Table 9.11 we demonstrate the effect of the different number of scales on the dimensions of $\widetilde{M}^{-1}$ and the total memory consumption of the algorithm. We use the data type `float` to represent one unit of memory, which uses 4 bytes on our system. We illustrate the situation for $\tilde{J}_1 = 4, 5$ and 6 coarse scales. The actual dimensions of $\widetilde{M}^{-1}$ are smaller than their corresponding upper-bound estimate $2^{2\tilde{J}_1} \times 2^{2\tilde{J}_1}$, due to explicitly removing the wavelets in the nullspace, see Section 7.3. Observe that at higher scales more wavelets are in the nullspace of the operator, since the support of the wavelet decreases as the scale increases.

We finish the section by comparing the computational cost and the memory estimates derived for the specific LTAO system to those of the MVM. The numerical performance of the MVM depends on the dimensions of the matrix. For this LTAO configuration the dimensions are $2Gn_s^2 \times n_a^2 = 2 \cdot 9 \cdot 84^2 \times 85^2$. Due to the circular shape of the pupil and

Operation	Step in Alg. 7.1	FLOP total	FLOP per process
Pseudo-open loop data	2	$(2G+6)(n_s+1)^2$	$8(n_s+1)^2$
right-hand side	5	$(97/3)L2^{2J_\ell}+[20G+10(L-1)G+L(G-1)](n_s+1)^2$	$(97/3)2^{2J_\ell}+[20+(11G-1)](n_s+1)^2$
update residual	6	$2L2^{2J_\ell}$	$2\cdot 2^{2J_\ell}$
GLMS	in 7	$2^{2\tilde{J}_1}(2^{2\tilde{J}_1+1}-1)+[(91/3)(L+1)+2L]2^{2J_\ell}+[26G+10(L-1)G+L(G-1)](n_s+1)^2$	$2^{2\tilde{J}_1}(2^{2\tilde{J}_1+1}-1)+(188/3)2^{2J_\ell}+[26+(11G-1)](n_s+1)^2$
1 PCG iter.	in 7	$(188/3)L2^{2J_\ell}+[26G+17(L-1)G+L(G-1)](n_s+1)^2$	$(188/3)2^{2J_\ell}+[26+7(L-1)+(11G-1)](n_s+1)^2$
Fitting	8	$(91/3)L2^{2J_\ell}+7(L-1)n_a^2$	$(91/3)2^{2J_\ell}+7(L-1)n_a^2$
Control	10	$3n_a^2$	$3n_a^2$
Total		$[97L+(91/3)]2^{2J_\ell}+[(22L+28)G-2L+6](n_s+1)^2+2^{4\tilde{J}_1+1}-2^{2\tilde{J}_1}+(7L-4)n_a^2+n_{\text{iter}}[(188/3)L2^{2J_\ell}+(9(2L+1)G-L)(n_s+1)^2]$	$(382/3)2^{2J_\ell}+(22G+52)(n_s+1)^2+2^{4\tilde{J}_1+1}-2^{2\tilde{J}_1}+(7L-4)n_a^2+n_{\text{iter}}[(188/3)2^{2J_\ell}+(7L+11G+18)(n_s+1)^2]$

Table 9.9: Computational cost of the wavelet Algorithm 7.1 for LTAO.

Algorithm variables	Memory
$\mathbf{s}$	$2Gn_s^2$
$\mathbf{c},\mathbf{r},\mathbf{b}$	$3L2^{2J_\ell}$
$\mathbf{a},\mathbf{a}^{(0)},\mathbf{a}^{(-1)}$	$3n_a^2$
$\tilde{v}$ (GLMS)	$2^{2\tilde{J}_1}$
$\mathbf{p},\mathbf{q},\mathbf{z}$ (PCG)	$3L2^{2J_\ell}$
Operator memory	
$\widetilde{M}^{-1}$	$2^{4\tilde{J}_1}$
$\mathbf{M}$	$2^{2J_\ell}L+(L+6)G(n_s+1)^2+(4L-3)G(n_s+1)+J_\ell L+n_s$
Total	$7L2^{2J_\ell}+J_\ell L+2^{4\tilde{J}_1}+2^{2\tilde{J}_1}+(L+8)G(n_s+1)^2+[(4L-3)G+1](n_s+1)+3n_a^2$

Table 9.10: Memory requirements of the LTAO algorithm.

GLMS scales $\tilde{J}_1$	Dimensions $2^{2\tilde{J}_1} \times 2^{2\tilde{J}_1}$	Dimensions of $\widetilde{M}^{-1}$	Memory (Mb)
4	256×256	255×255	8.58
5	1024×1024	853×853	11.8
6	4096×4096	2434×2434	31.6

Table 9.11: Estimated memory consumption of the whole method with different number of GLMS scales. The total memory consumption of the algorithm is measured in megabytes.

the central obstruction, not all of the measurements and actuators are active, hence, the actual dimension of the MVM matrix is roughly 99900×5400. Note that the matrix dimensions are independent of the level of discretization of the layers.

Thus, the computational cost of the MVM is $99900(2 \cdot 5400 - 1) = 1.08 \cdot 10^9$ floating point operations. In comparison, the total computational cost of the wavelet algorithm is $3.00 \cdot 10^7 + 2.03 \cdot 10^7 n_{\text{iter}}$ with $\tilde{J}_1 = 4$ GLMS scales. The cost of the wavelet method with $n_{\text{iter}} = 4$ PCG iterations is $1.11 \cdot 10^8$ operations and $2.33 \cdot 10^8$ operations with $n_{\text{iter}} = 10$ iterations. This corresponds to an improvement by a factor 10 and 5 over the MVM, respectively.

The memory consumption of the MVM is $99900 \cdot 5400 = 5.39 \cdot 10^8$. Conversely, the memory consumption of the wavelet method is $2.25 \cdot 10^6$, which is by a factor 240 less than the MVM. In terms of the `float` data type, the wavelet method requires 8.58 megabytes, whereas the MVM requires 2.01 gigabytes of memory. Thus, the wavelet method has the advantage of completely fitting into the cache of the modern CPU, which is not the case for the MVM.

9.3 Computational performance

In this section we demonstrate the computational performance of the wavelet method by evaluating the reconstruction time of the full algorithm and its components. The tests are carried out on two computing systems that are specified in Section 9.3.1.

In Section 9.3.2 we present the reconstruction time of the full algorithm for LTAO and MOAO and compare it to that of the MVM. We show that the algorithm scales linearly with respect to the number of iterations. Further, we highlight the compactness of the algorithm for MOAO with multiple DMs: the method can determine the shapes of several DMs on a single computing system without it having an effect on performance. We also evaluate the time for individual components of the algorithm.

Next, in Section 9.3.3 we compare the reconstruction time for our method with the MVM in the 3-layer MCAO configuration. Here we demonstrate the benefits of using the combined parallelization strategy, see Section 9.1.6, where global and local parallelization strategies are applied simultaneously.

Finally, in Section 9.3.4 we study the benefits of local parallelization, see Section 9.1.5, in terms of the discrete wavelet transform on the CPU. Apart from parallelization, we show that computational advantage can be gained by using standard library components.

9.3.1 Hardware and test specifications

To test the speed of the algorithm and its components, the code was written in `C` and parallelization was implemented using OpenMP. The data type `float` was used to store real numbers, which uses four bytes on our systems.

To evaluate the speed of the full algorithm, we measured the time needed for one loop iteration, i.e., one instance of Algorithm 7.1. We also measured the computing time of individual components. The algorithm was executed 1000 times and the average time was taken.

For the following tests we used two hardware systems outlined in Table 9.12. The X5650 system is a multi-CPU multi-core system with a total of 12 cores (6 cores on 2 CPUs). This configuration allows to test global parallelization of the algorithm for LTAO and MOAO, see Section 9.1.4, which requires at least 9 cores.

The E5-1650 system is a single-CPU multi-core system with 6 cores. Although it has less cores than the X5650 system, the E5-1650 CPU has a faster clock speed and a newer architecture. This system is well suited for the 3-layer MCAO reconstructor with the combined parallelization strategy, see Section 9.1.6.

CPU	Num. of cores	Clock speed	L3 cache	Launch date	Recommended price
Intel Xeon X5650	2×6	2.66 GHz	2×12 Mb	Q1 2010	\$2000.00
Intel Xeon E5-1650	6	3.20 GHz	12 Mb	Q1 2012	\$583.00

Table 9.12: Hardware specifications of the computing systems.

The cache of both systems is larger than the required memory consumption of the wavelet algorithm in the demonstrated tests below. Compactness of the method in terms of memory allows the algorithm to efficiently utilize the CPU architecture of the computing systems.

In addition to the system specifications, we also list the recommended retail price of the CPUs. The cost of the complete workstations with the X5650 and the E5-1650 CPUs was €5000 and €2000, respectively. The experiments in the following sections on these off-the-shelf hardware systems exemplify the potential financial cost reduction by utilizing the wavelet reconstructor.

9.3.2 Performance for LTAO and MOAO

We demonstrate the reconstruction times of the full algorithm for LTAO and MOAO systems below. The AO system and the algorithm are configured in a similar way to the simulations in quality in Sections 8.3 and 8.4 for LTAO and MOAO, respectively.

Both AO systems utilize $G = 9$ guide stars, of which $G_{\text{LGS}} = 6$ are LGSs and $G_{\text{NGS}} = 3$ are NGSs. To each guide star a WFS in an identical configuration with $n_s^2 = 84^2$ subapertures is assigned. Four laser launch positions are used for the LGSs. One ground DM is used with $n_a^2 = 85^2$ actuators.

The algorithm reconstructs $L = 9$ turbulence layers. The layers are discretized on a grid of $2^{2J_\ell} = 128^2$ points; we use the speed-oriented setup for the following tests, see

Table 8.7. The solution method utilizes $\tilde{J}_1 = 4$ scales of the GLMS algorithm and the frequency-dependent preconditioner is used.

More details on the system configuration are found in Section 8.1. The method is parallelized on $\max(L, G) = 9$ processes according to the global parallelization approach in Section 9.1.4. For the following tests we use the X5650 CPU, see Table 9.12.

Reference case: the MVM

The reference reconstructor is the MVM. The matrix size is 5402×99900 elements for both the LTAO and the MOAO systems. The MVM implementation has been provided by the ESO. The results in speed were obtained on the X5650 system.

In Table 9.13 we present the reconstruction time of the MVM with different levels of parallelization. The reconstruction matrix consumes 2.01 gigabytes of memory, see Section 9.2.7. Thus, a large amount of data is transferred between the CPU and the memory, which is a possible explanation to why further parallelization does not improve the reconstruction speed on this architecture.

Threads	1	2	3	4	6	8	10	12
MVM	243	141	122	110	95	81	79	80

Table 9.13: Reconstruction time for the MVM in ms.

The wavelet reconstructor

In Table 9.14 we demonstrate the computational performance of the wavelet method with $n_{\text{it}} = 4$ iterations and different levels of parallelization for LTAO.

Threads	1	2	3	4	5	6	7	8	9
Wavelets	38.87	22.45	15.08	13.19	10.54	10.62	10.75	9.06	6.18

Table 9.14: Reconstruction time in ms of the wavelet method with 4 iterations for LTAO.

In Table 9.15 the computational performance of the wavelet method with $n_{\text{tier}} = 10$ iterations and different levels of parallelization is presented for MOAO.

Threads	1	2	3	4	5	6	7	8	9
Wavelets	81.57	47.06	31.78	27.51	22.16	21.92	21.90	18.62	12.65

Table 9.15: Reconstruction time in ms of the wavelet method with 10 iterations for MOAO.

The reconstruction time of the wavelet approach with different number of threads for LTAO and MOAO is plotted in Figure 9.4.

Contrary to the MVM, which requires 2.01 gigabytes, the wavelet method is relatively compact in terms of memory, with an estimated memory consumption of 8.58 megabytes, see Section 9.2.7. The memory requirements of the wavelet method is below the cache size of our computing system and the amount of communicated data is small. Thus, the

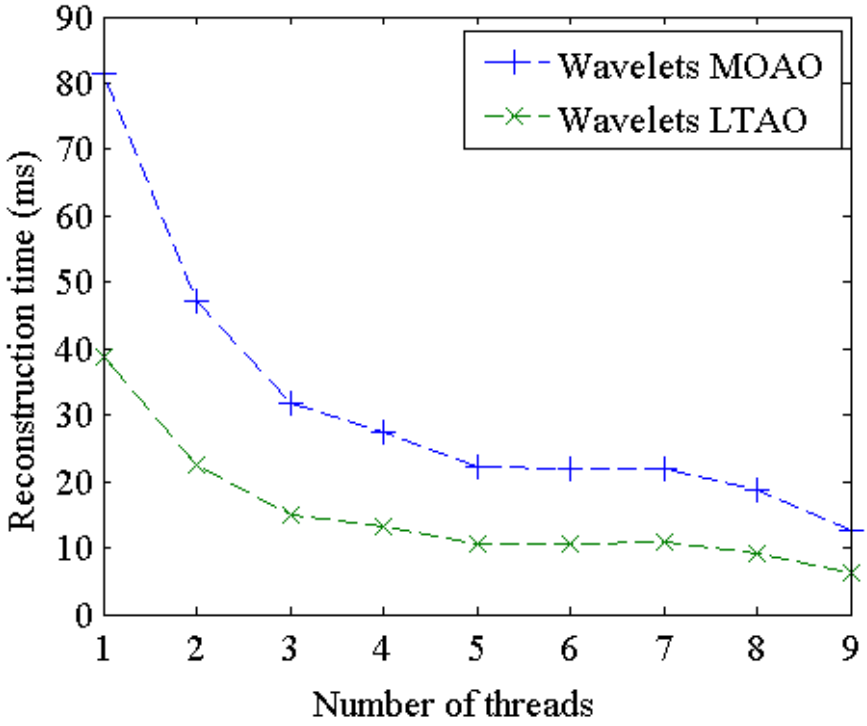

Figure 9.4: Reconstruction time in ms of the wavelet method for LTAO and MOAO using up to 9 cores for parallelization.

wavelet method can efficiently utilize the parallel architecture of the computing system. The best performance of the method is attained with full parallelization on 9 cores.

The speed-up factor of approximately 13 is gained on our system for the LTAO reconstructor with 4 iterations and of 6 for the MOAO reconstructor with 10 iterations over the MVM.

Scalability of the method with number of iterations

The wavelet algorithm for LTAO and MOAO is almost identical. The difference in the reconstruction time for the two systems is due to the different number of PCG iterations.

The wavelet method scales linearly with the number of PCG iterations. We plot the reconstruction time of the wavelet method for MOAO with different number of iterations in Figure 9.5.

One can observe that roughly 1 ms is needed per iteration on the X5650 system. Moreover, the evaluation time of the method without any PCG iterations is roughly 2 ms. This is consumed largely by the computation of the right-hand side vector, the DM fitting and the GLMS method, i.e., steps 5 and 8 in Algorithm 7.1 and step 2 in Algorithm 7.2, respectively.

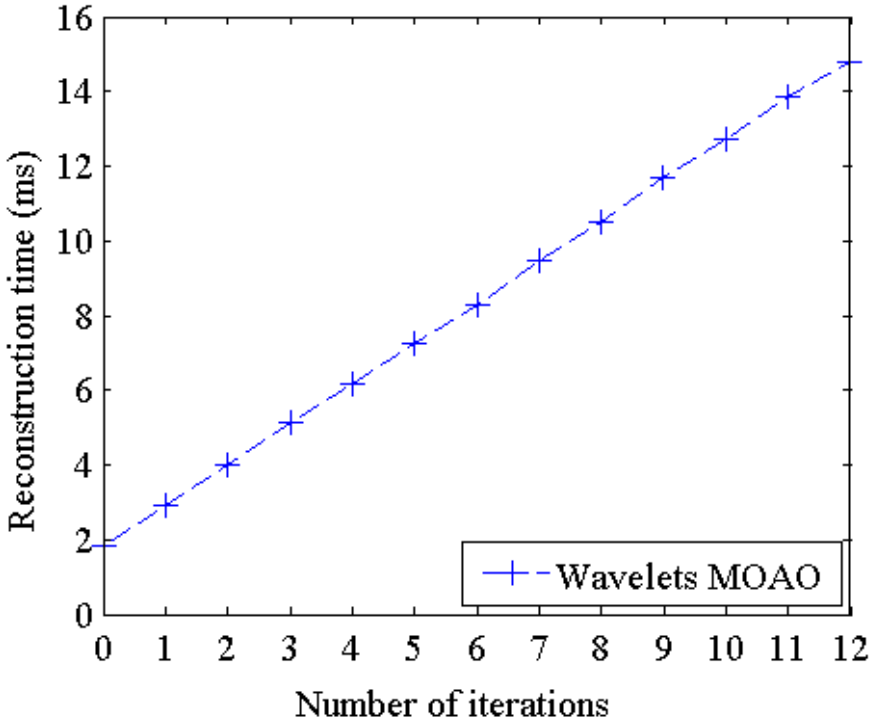

Figure 9.5: Reconstruction time in ms of the wavelet method for MOAO with different number of iterations.

Scalability of the method with number of DMs

The wavelet method for MOAO has a big advantage over the MVM when more than one DM is used. Contrary to the MVM, the wavelet method for MOAO can use the same computing system to determine shapes of multiple DMs. Atmospheric tomography is performed once for all DMs. Consequently, the fitting operator $\mathbf{F}$ in step 8 of Algorithm 7.1 can be computed in parallel with respect to the number of DMs, see equation (7.12).

We illustrate this property in Figure 9.6. In this test we compute up to 11 DM shapes in parallel. The difference in the computational performance between reconstructing one or multiple mirrors is negligible.

In comparison, an MVM reconstructor requires an additional computing system per each DM. Thus, the wavelet method is a more compact reconstructor for an MOAO system when multiple objects are observed.

Time for individual components

In Table 9.16, we present the computing time of the individual components of Algorithm 7.1 for LTAO.

Computing the pseudo-open loop data and control are both numerically cheap. On the other hand, computing the right-hand side vector, the GLMS method, the PCG method and the fitting of the DMs are relatively expensive. Each of the latter steps utilizes parts of the left-hand side operator $\mathbf{M}$, see (5.50). We state the computing times

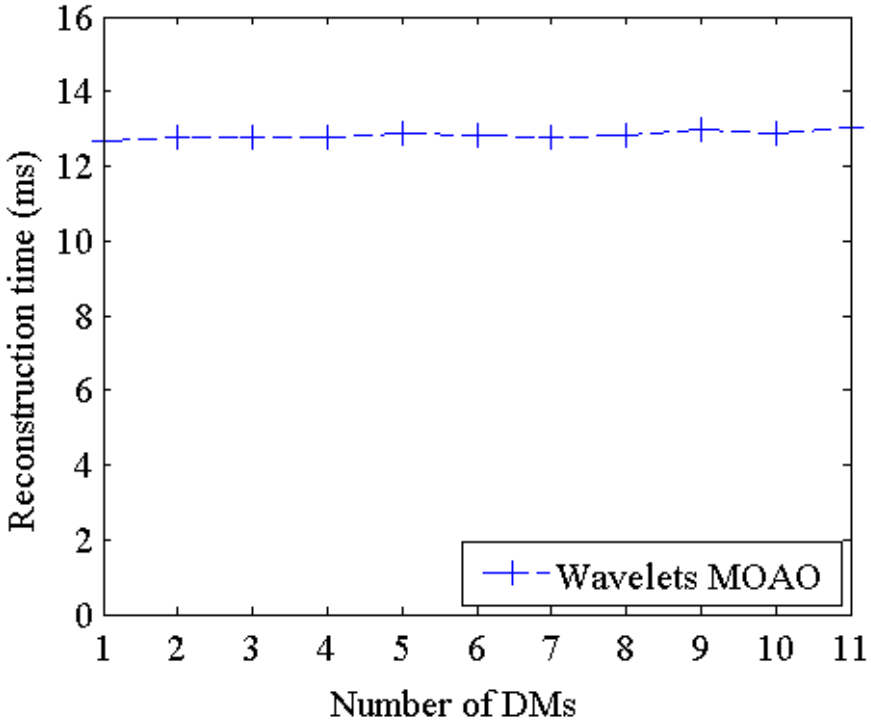

Figure 9.6: Reconstruction time in ms of the wavelet method for MOAO with different number of DMs.

Operation	Step in Alg. 7.1	Time (μs)
Pseudo-open loop data	2	39
Right-hand side	5	576
GLMS	in 7	786
1 PCG iteration	in 7	1122
Fitting	8	359
Control	10	3

Table 9.16: Computing time for individual components of the wavelet Algorithm 7.1 for LTAO.

of the components of $\mathbf{M}$ below.

As discussed in Section 9.2.5 above, the cost of one PCG iterations is predominantly determined by the cost of applying the operator $\mathbf{M}$. Invoking $\mathbf{M}$ also outweighs the cost of the GLMS method, provided that the number of coarse scales is low. Thus, both steps are to a large extent determined by the cost of applying the left-hand side operator. We point out that the special structure of the GLMS vector, which is zero at all layers except for the ground layer, influences the computing time of $\mathbf{M}$: the GLMS method is 336 μs faster to apply than one PCG iteration.

We present the execution time of $\mathbf{M}$ and its components under the global paralleliza-

tion scheme of Algorithm 9.1 in Table 9.17.

Operation	Step in Alg. 9.1	Time (μs)
$\mathbf{M}$	–	1019
$\delta_\ell^{-1} W^{-1}$	2-3	186
P_g	6-8	202
Γ_g	9	14
$\mathcal{M}_g (I-T) \widetilde{C}_g^{-1} (I-T) \mathcal{M}_g$	11-15	33
Γ_g^T	19	17
P_1^T	22-24	61
P_2^T	22-24	355
P_9^T	22-24	294
$\delta_\ell^{-1} W$	25-26	170
αD_ℓ	27	7

Table 9.17: Computing time of $\mathbf{M}$ and its components, see Algorithm 9.1 for more details. The operations are applied at layers $1 \leq \ell \leq L$ and guide star directions $1 \leq g \leq G_{\mathrm{LGS}} < g' \leq G$ in parallel, where each component is computed on one core.

Observe that one PCG iteration, which takes 1122 μs to compute, consists of the operator $\mathbf{M}$, which takes 1019 μs to evaluate, and all other operations, including preconditioning, which require 103 μs. Invoking the left-and side operator is the most expensive step of the algorithm.

The cost of the projection operation P_g in a guide star direction g is roughly the same for all directions. In GLMS, this projection operation is omitted. Thus, applying $\mathbf{M}$ to the ground layer vectors is roughly 200 μs faster.

The transposed projection operation P_ℓ^T, on the other hand, has different computational costs for different layers. At the ground layer $\ell = 1$, the wavefront and the layer grids are aligned, no interpolation is performed, and the operation is cheap. At the other layers $\ell \geq 2$, bilinear interpolation makes the operation more expensive. Due to the cone effect, the cost at higher layers is cheaper than at the lower layers. Here we compare the evaluation time for layers $\ell = 1$, $\ell = 2$ and $\ell = 9$, which are 61, 355 and 294 μs, respectively.

The direct and the transposed Shack-Hartmann and the inverse noise covariance are all relatively inexpensive operations. We indicate that due to the special diagonal structure of the matrix $\mathbf{D}$, adding the regularizing term in the wavelet domain is the cheapest operation of $\mathbf{M}$.

Thus, the expensive operations of $\mathbf{M}$ are the wavelet transform and the bilinear interpolations. These operations can be improved by local parallelization or by using standard library components, see Sections 9.3.3 and 9.3.4 below.

9.3.3 Performance for MCAO

In this section we demonstrate the reconstruction times of the full algorithm for the MCAO system. The system and the algorithm are configured in a similar way to the

simulations in quality in Section 8.5. We use the 3-layer configuration of the algorithm.

The MCAO system utilizes $G = 9$ guide stars, of which $G_{\text{LGS}} = 6$ are LGSs and $G_{\text{NGS}} = 3$ are faint NGSs used for tip-tilt correction. To each LGS a WFS in an identical configuration with $n_s^2 = 84^2$ subapertures is assigned. Four laser launch positions are used. The tip-tilt sensors have 2×2 or 1×1 subapertures. There are $M = 3$ DMs conjugated to different altitudes.

The algorithm reconstructs $L = 3$ turbulence layers at the altitudes of the DMs. The ground layer is discretized on a grid of $2^{2J_1} = 128^2$ points and the other two layers are discretized on a grid of $2^{2J_\ell} = 64^2$ points, for $\ell = 2, 3$; we use the 3-layer setup for the following tests, see Table 8.8. The solution method does not use the GLMS algorithm, but the frequency-dependent preconditioner is used with $n_{\text{it}} = 4$ PCG iterations. More details on the system configuration are found in Section 8.1.

The method is parallelized on $\max(L, G_{\text{LGS}} + 1) = 7$ processes according to the global parallelization approach in Section 9.1.4. Here, we compute the tip-tilt operations on one core.

Additionally, we utilize the combined parallelization scheme in Section 9.1.6, which uses the special features of the system. There, the operators are parallelized similarly as for $\mathbf{M}$, see Algorithm 9.2. The optimal number of cores for this scheme is 7, but 6 cores can also be used efficiently.

For the following tests we use both the X5650 and the E5-1650 systems, see Table 9.12

Reference case: the MVM

The reference reconstructor is the MVM. The matrix size is 9296×66612 elements. The MVM implementation has been provided by the ESO. The results in speed were obtained on both the X5650 and the E5-1650 systems.

In Table 9.18 we present the reconstruction time of the MVM with different levels of parallelization using the two systems. The matrix of the MVM consumes 2.31 gigabytes of memory. Thus, a large amount of data is transferred between the CPU and the memory, which is a possible explanation to why further parallelization does not improve the reconstruction speed on this architecture.

Threads	1	2	3	4	6	8	10	12
MVM on X5650	268	166	135	126	108	101	104	104
MVM on E5-1650	169	96	79	73	72	–	–	–

Table 9.18: Reconstruction time for the MVM in ms.

Observe how the newer architecture of the E5-1650 system outperforms the older X5650 system, even with less processor cores.

The wavelet reconstructor

We demonstrate the performance of the 3-layer MCAO wavelet method in two parallelization configurations, the global one, see Algorithm 9.1, and the combined one, see Algorithm 9.2, on the two computing systems, X5650 and E5-1650.

In Table 9.19 we present the computational performance of the wavelet method with different levels of parallelization. The generic global parallelization scheme allows parallelization on up to 7 cores, whereas the combined algorithm is tailored towards a 7 core system. However, it can also be executed on 6 cores if parallelization of steps 7-13 and 14-25 of Algorithm 9.2 is performed on 6 cores only.

Threads	1	2	3	4	5	6	7
Wavelets on X5650, global	7.19	5.30	3.89	3.49	3.51	3.47	3.05
Wavelets on X5650, combined	–	–	–	–	–	2.70	2.15
Wavelets on E5-1650, global	4.42	3.15	2.20	2.00	2.03	2.01	–
Wavelets on E5-1650, combined	–	–	–	–	–	1.56	–

Table 9.19: Reconstruction time in ms of the wavelet method for MCAO.

As for the MVM method, the newer E5-1650 system shows a better performance than X5650 in the same parallelization configuration. In the global parallelization scheme, most efficiency is attained already on 3 cores, which corresponds to the number of layers. Indeed, the most expensive operations are associated with layers: the inverse and the direct wavelet transforms, and the transposed projection operations, i.e., steps 3, 25 and 22-24 of Algorithm 9.1.

The advantage of the combined local and global parallelization schemes is apparent. On the X5650 system, the peak performance is attained with the combined parallelization scheme on 7 cores. Even on 6 cores, the performance is higher for the combined scheme than with the generic global parallelization on 7 cores.

The most efficient performance of all is attained on the E5-1650 system with 6 cores. Judging from the performance spike between 6 and 7 cores on the X5650 system, the performance can still be improved for the E5-1650 system if at least 7 cores are available.

Contrary to the MVM, the wavelet method is relatively compact in terms of memory. The speed-up factor of approximately 46 is gained on the E5-1650 system for the MCAO 3-layer reconstructor over the MVM.

Time for individual components

We present the execution time of $\mathbf{M}$ and its components on the E5-1650 system in Table 9.20. The operator is parallelized on 6 cores according to the combined parallelization strategy in Algorithm 9.2.

Due to the smaller problem size, better parallelization and faster computing architecture, the total evaluation time of $\mathbf{M}$ for MCAO 3-layer problem is lower than that for LTAO or MOAO. Below we point out the effects of combined parallelization.

One place where combined parallelization improves performance of the method is in the computation of the inverse wavelet transform. It takes 56 μs for 4 cores to compute the transform of the ground layer on a 128×128 point grid. In parallel, 26 μs are spent evaluating the transform of the other two layers on a grid of 64×64 points. On one core, the transform of the ground layer consumes approximately 100 μs, see Table 9.22. Thus, the difference in computing time between the ground layer and the altitude layers is reduced from approximately 75 μs to 30 μs by local parallelization of the ground layer transform.

Operation	Step in Alg. 9.2	Cores	Time (μs)
$\mathbf{M}$	–	6	289
$\delta_\ell^{-1} W^{-1}, \ell = 1$	2, 4	4	56
$\delta_\ell^{-1} W^{-1}, \ell \geq 2$	2, 6	1	26
$\boldsymbol{\gamma}$	9	1	9
P_g	11-13	1	20
$\sigma_{g'}^{-2}, \boldsymbol{\gamma}^T$	16-17	1	11
Γ_g	19	1	9
$\mathcal{M}_g (I - T) \widetilde{C}_g^{-1} (I - T) \mathcal{M}_g$	20-24	1	21
Γ_g^T	25	1	11
$P_\ell^T, \ell = 1$	29	1	16
$P_\ell^T, \ell \geq 2$	34-37	2	55
$\delta_\ell^{-1} W, \ell = 1$	30, 39	2	75
$\delta_\ell^{-1} W, \ell \geq 2$	38, 39	1	27
αD_ℓ	40	1	5

Table 9.20: Computing time of $\mathbf{M}$ and its components, see Algorithm 9.2 for more details. The operations are applied at layers $1 \leq \ell \leq L$ and guide star directions $1 \leq g \leq G_{\mathrm{LGS}}$ in parallel; the number of cores assigned to each process is stated.

The other place where combined parallelization improves the performance is in the computation of the transposed projections followed by the direct wavelet transforms. On the ground layer, the transposed projection step takes 16 μs to compute; then, the wavelet transform is performed in parallel on two cores in 75 μs. On the altitude layers, the transposed projection is computed on two cores and takes 55 μs to evaluate; then the wavelet transform on one core is applied in 27 μs. In total, the ground layer operations consume 91 μs and the other layers 82 μs, which means the work is shared fairly over the parallel processes during the computation. In the global parallelization scheme, the ground layer and the altitude layers require 115 and 135 μs to apply, respectively, and thus are roughly 45 μs slower.

9.3.4 Local parallelization and standard library components for DWT

In the section above, we have shown that the performance of the wavelet algorithm can be improved by using local parallelization. In this section we concentrate on the discrete wavelet transform and provide a case study for local parallelization of the DWT, see Section 9.1.5.

Additionally, we show that the performance of the algorithm can also be improved by using standard library components. We demonstrate this for the DWT implementation of the Intel Integrated Performance Primitives (Intel IPP).

The DWT consists of the wavelet transform operator W_j applied twice subsequently at each scale $j = J - 1, ..., 0$. At each scale, the operator W_j is of dimension $n_j \times n_j = 2^{j+1} \times 2^{j+1}$.

Practically, on a CPU it is favorable to parallelize the DWT only at the higher scales, where a large amount of data is processed. At the lower scales, the dimensions of the data are too small for parallelization to be efficient, due to the incurred overhead costs.

In the following tests, we compute the discrete wavelet transform on a grid of 128×128 and 256×256 points, i.e., for $J = 7$ and 8 wavelet scales, respectively. For some threshold scale $j_p = 0, ..., J-1$, we apply W_j in parallel on scales $j = J-1, ..., j_p$. Further computation on scales $j = j_p - 1, ..., 0$ is performed using one process.

In Table 9.21 we present the computing time of the parallel DWT on different number of processes for different threshold scales j_p on the X5650 system. Additionally, we state the execution time of the Intel IPP implementation. In Table 9.22 the same is presented for the E5-1650 system.

Grid size	Threshold scale j_p	Number of cores				Intel IPP
		1	2	4	8	
128×128	6	168	111	101	86	110
	5		100	92	73	
	4		99	108	102	
256×256	7	905	559	406	361	356
	6		494	317	254	
	5		477	313	237	
	4		475	327	252	

Table 9.21: Parallelization of the DWT on the X5650 system in μs. Parallelization is implemented on scales $j = j_p, ..., J-1$ only.

Grid size	Threshold scale j_p	Number of cores				Intel IPP
		1	2	4	6	
128×128	6	100	71	58	52	64
	5		65	49	42	
	4		67	55	53	
	3		67	57	55	
256×256	7	553	346	247	216	184
	6		302	183	146	
	5		295	170	132	
	4		299	177	140	
	3		300	181	141	

Table 9.22: Parallelization of the DWT on the E5-1650 system in μs. Parallelization is implemented on scales $j = j_p, ..., J-1$ only.

Although the E5-1650 system outperforms the X5650 system as in the tests above, the following conclusions are drawn. By increasing the number of cores, the computing time decreases. The optimal value of the threshold scale is $j_p = 5$. Thus, parallelization is efficient on grids of 64×64 points and above, but further parallelization slows down the computation. Also, the Intel IPP implementation reduces the computing costs over a single-core implementation, but it is less efficient than the parallelization approach.

Very promising results have been shown on parallelization of the DWT on the GPGPU in, e.g., [18]. The authors have shown that especially for large grid dimensions, a parallel implementation of the DWT on a GPGPU has a significant advantage over the CPU.

The results in Section 9.3.3 were obtained with the parallelization threshold parameter $j_p = 5$ for the ground layer.

Chapter 10

Conclusion and outlook

10.1 Conclusion

In this thesis we presented the novel Finite Element-Wavelet Hybrid Algorithm (FEWHA) for atmospheric tomography. We summarize the algorithm and our results regarding its quality and speed below.

The algorithm

The FEWHA is a solution method for the MAP estimate (4.8) of the atmospheric tomography problem (4.2). The method is highly parallelizable with global computational complexity of $\mathcal{O}(n)$. The FEWHA utilizes a dual-domain discretization strategy, where both the piecewise bilinear basis of finite elements and the wavelet basis are used to parametrize the turbulence layers.

This allows a sparse "matrix-free" representation of the fitting term in the piecewise bilinear basis, see Sections 5.4, 9.1.1 and 9.1.2, and a completely diagonal approximation of the penalty term in the wavelet basis, see Section 5.2. The two bases are linked via the DWT, which is an operation of linear complexity.

Our method can handle the complex deficiencies of an LGS, such as the cone effect and spot elongation, see Section 2.6.2, and the tip-tilt indetermination, see Section 5.3.2. The method shows a stable performance even in the presence of high measurement noise, which we demonstrated using numerical simulations in Sections 8.3-8.5.

In FEWHA, we solve the discretized linear system of equations with the PCG algorithm. Apart from the standard approach of warm restarting the iteration, the convergence of the PCG algorithm is improved using two new techniques designed for our method: a frequency-dependent preconditioner and the GLMS method to improve the initial guess.

The frequency-dependent preconditioner is a modification of the Jacobi preconditioner and is described in Section 6.2. We have examined the convergence properties of this preconditioner via numerical experiments in Sections 8.4.3, 8.4.5 and 8.5.2.

In the GLMS method, we solve a sub-problem on the coarse wavelet scales of the ground layer, see Section 6.3. We investigated the effect of the number of coarse scales on the qualitative performance of the algorithm by simulations in Section 8.4.4.

The wavelet reconstructor is summarized in Algorithm 7.1. It combines the FEWHA

for atmospheric tomography with a control loop and a mirror fitting operation. In closed loop, we use the pseudo-open loop control. We described how the algorithm can be applied to the various AO systems: SCAO, in which we perform a wavefront reconstruction and LTAO, MOAO and MCAO, where the full atmospheric tomography problem is solved. The different mirror fitting steps are outlined in Section 7.4.

The quality

We demonstrated the performance of our method in terms of quality on OCTOPUS, the simulation tool of the ESO. We have shown that our method performs similar to the MVM for SCAO in Section 8.2.1 and outperforms the MVM for MCAO in Section 8.5.1. Further, we have shown that our method outperforms the FrIM tuned by the ESO for LTAO and performs similar to FrIM by ESO for MOAO in Sections 8.3.1 and 8.4.1, respectively.

For MCAO, the algorithm can be configured to reconstruct the mirror shapes directly, or reconstruct the full atmosphere and then compute the optimal mirror corrections. We have shown the difference between these two operations in terms of numerical simulations in Section 8.5.1 and compared different fitting step configurations in Section 8.5.3.

We also studied parameter sensitivity of the method in Section 8.4.2 for MOAO, where we determined that the method is sensitive to the regularization tuning and the spot elongation tuning parameters, but the method is not sensitive to gain.

The speed

We analyzed the reconstruction speed of our method in terms computational estimates and evaluated the reconstruction time of the algorithm in specific configurations.

We derived the computational cost estimates for our method and showed that they are in line with the expectation of scaling as $\mathcal{O}(n)$, where n is the dimension of the system. The estimates for the components of the algorithm have been presented in Sections 9.2.1-9.2.6. Estimates for the full algorithm for LTAO were derived in Section 9.2.7.

In addition, we have shown that the "matrix-free" operators induce a very memory efficient representation of our algorithm. For instance, for LTAO, the memory consumption of our method was 240 times less than that of the MVM, see Section 9.2.7.

We proposed two parallelization strategies for our method, global and local, which can also be combined, see Sections 9.1.3-9.1.6. In global parallelization we split the work in parallel over the different layers and/or guide star directions. Global parallelization has been applied to the LTAO and MOAO reconstructors and the corresponding numerical tests were carried out in Section 9.3.2. In local parallelization we split the work in computing the individual components of the method. Numerical tests were carried out for the local parallelization of the discrete wavelet transform in Section 9.3.4. Finally, both strategies have been applied for the MCAO reconstructor and the corresponding results were documented in Section 9.3.3.

The wavelet algorithm was written in `C` and parallelization was implemented using OpenMP. The comparison of the numerical performance of our method against the MVM is summarized in Table 10.1. Here, we state the reconstruction times of the two methods and the speed-up factor of the wavelet algorithm over the MVM.

AO system	LTAO	MOAO	MCAO
Computing system	X5650	X5650	E5-1650
Computing time MVM (ms)	79	$79^{(1)}$	72
Computing time FEWHA (ms)	6.18	$12.65^{(2)}$	1.56
Speed-up factor	13	6	46

Table 10.1: Summary of the computing time. Remarks: $^{(1)}$ - the time for MVM for 1 DM; $^{(2)}$ - the time for FEWHA for multiple DMs.

The speed-up factor for MOAO over the MVM is the lowest from the three systems. However, the reconstruction algorithm is more compact than the MVM. For the MVM, an additional mirror requires an additional computing system. In comparison, the wavelet method can compute the shape of multiple mirrors simultaneously on one system.

We point out that the MCAO case highlights the full potential of the wavelet algorithm in both speed and quality. Even on a non-dedicated hardware and a non-optimal number of CPUs (6 instead of 7), the reconstruction time is close to the allotted computing time of 1 ms for a real-world application. This is in part due to the combination of local and global parallelization strategies. This experiment suggests that the wavelet method can achieve the required computing time even on the off-the-shelf hardware with more CPU cores.

In terms of quality, the wavelet method in the MCAO 3-layer configuration outperforms the MVM by a significant margin, see the plot in Figure 8.11. Moreover, we indicate that the wavelet method is also easier to configure than the MVM, since no major pre-computing is necessary, such as the matrix inversion for the MVM. Most of the parameters of the wavelet algorithm can be updated at run-time.

10.2 Outlook

We believe that the FEWHA is a very promising algorithm in the field of atmospheric tomography. The method can be developed further in the following three directions.

First, the reconstruction algorithm itself can be improved. With the aim of reducing the number of iterations, more advanced multi-scale methods than the GLMS can be investigated. Early tests show that the GLPCG algorithm, see Section 6.3.2, is a promising supplementary method to the GLMS. The algorithm makes it possible to exchange a certain number of PCG iterations of the full problem with the GLPCG iterations, which are significantly cheaper. Alternatively, the multi-scale method can be applied to more layers, as was described in Section 6.3.1.

Further, more sophisticated control methods should be looked into. For instance, a scale-dependent gain control can be applied without any significant additional computational cost for the wavelet method. An application of the wavelet framework is of interest with respect to the Kalman filter.

Second, the algorithm should be validated further against more realistic assumptions. This requires further simulations and optical bench tests. The influence of more accurate models of the DMs, misalignment of the DM with the WFS and telescope spiders, i.e., the mounting arms of the central obstruction, and other realistic phenomena on the method

should be examined.

Also, further parameter sensitivity studies can be performed. For instance, the performance of the reconstruction can be investigated with respect to the variations in turbulence layer heights and strengths. In this respect, we have already shown for the MCAO example that the 3-layer model where these parameters are not known has a similar performance to the case where these parameters are known in the 9-layer model. However, other simulations can be performed.

Finally, the third direction to improve the method is the computational implementation. Here, we would like to investigate the numerical performance of the method on other computational architectures. The algorithm should be tested on a multi-CPU multi-core system with more parallel processes, where the combined local and global parallelization schemes can be exploited even further. Of special interest is the GPGPU architecture, which fits to the SIMD parallelization paradigm of our algorithm, as was discussed in Section 9.1.5.

These and other topics are left for future work. And that's all I have to say about that.

Bibliography

[1] N. Ageorges and C. Dainty, editors. *Laser Guide Star Adaptive Optics for Astronomy*, volume 551 of *Mathematical and Physical Sciences*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2000.

[2] C. Béchet, M. Le Louarn, R. Clare, M. Tallon, I. Tallon-Bosc, and É. Thiébaut. Closed-loop ground layer adaptive optics simulations with elongated spots: impact of modeling noise correlations. In *1st AO4ELT conference - Adaptive Optics for Extremely Large Telescopes*, page 03004, 2010.

[3] H. Brakhage. On ill-posed problems and the method of conjugate gradients. In H. W. Engl and C. W. Groetsch, editors, *Inverse and Ill-Posed Problems*, pages 165–175. Academic Press, Orlando, 1987.

[4] W. L. Briggs, V. E. Henson, and S. F. McCormick. *A multigrid tutorial*. SIAM, 2nd edition, 2000.

[5] E. Brunner, C. Béchet, and M. Tallon. Optimal projection of reconstructed layers onto deformable mirrors with fractal iterative method for AO tomography. In *Proc. SPIE, Astronomical Telescopes + Instrumentation*, pages 84475I–84475I–9, 2012.

[6] A. Cohen, I. Daubechies, and J.-C. Feauveau. Biorthogonal bases of compactly supported wavelets. *Comm. Pure Appl. Math.*, 45(5):485–560, 1992.

[7] C. Correia, K. Jackson, J.-P. Véran, D. Andersen, O. Lardière, and C. Bradley. Static and predictive tomographic reconstruction for wide-field multi-object adaptive optics systems. *J. Opt. Soc. Am. A*, 31(1):101–113, Jan 2014.

[8] W. Dahmen. Multiscale analysis, approximation, and interpolation spaces. In *Approximation theory VIII, Vol. 2 (College Station, TX, 1995)*, volume 6 of *Ser. Approx. Decompos.* World Sci. Publ., River Edge, NJ, 1995.

[9] I. Daubechies. Orthonormal bases of compactly supported wavelets. *Comm. Pure Appl. Math.*, 41(7):909–996, 1988.

[10] I. Daubechies. *Ten Lectures on Wavelets*. SIAM, 1992.

[11] M. A. Davison. The ill-conditioned nature of the limited angle tomography problem. *SIAM J. Appl. Math.*, 43:428–448, 1983.

[12] B. L. Ellerbroek. Efficient computation of minimum-variance wave-front reconstructors with sparse matrix techniques. *J. Opt. Soc. Am.*, 19(9):1803–1816, 2002.

[13] B. L. Ellerbroek, L. Gilles, and C. R. Vogel. Numerical simulations of multiconjugate adaptive optics wavefront reconstruction on giant telescopes. *Applied Optics*, 42:4811–4818, 2003.

[14] B. L. Ellerbroek and C. R. Vogel. Simulations of closed-loop wavefront reconstruction for multiconjugate adaptive optics on giant telescopes. *Proc. SPIE*, 5169:206–217, 2003.

[15] B. L. Ellerbroek and C. R. Vogel. Inverse problems in astronomical adaptive optics. *Inverse Problems*, 25:063001, 2009.

[16] H. W. Engl. *Integralgleichungen*. Springer-Verlag, Wien, 1997.

[17] H. W. Engl, M. Hanke, and A. Neubauer. *Regularization of Inverse Problems*. Kluwer Academic Publishers, Dordrecht, 1996.

[18] J. Franco, G. Bernabe, J. Fernandez, and M.E. Acacio. A parallel implementation of the 2d wavelet transform using cuda. In *Parallel, Distributed and Network-based Processing, 2009 17th Euromicro International Conference on*, pages 111–118, 2009.

[19] D. L. Fried. Atmospheric turbulence optical effects: Understanding the adaptive optics implications. In D. M. Alloin and J-M Mariotti, editors, *Adaptive Optics in Astronomy*, pages 25–57. Kluwer Academic Publishers, 1994.

[20] T. Fusco, J.-M. Conan, G. Rousset, L.M. Mugnier, and V. Michau. Optimal wavefront reconstruction strategies for multi conjugate adaptive optics. *J. Opt. Soc. Am. A*, 18(10):2527–2538, 2001.

[21] D. T. Gavel. Tomography for multiconjugate adaptive optics systems using laser guide stars. In *SPIE*, volume 5490, pages 1356–1373, 2004.

[22] L. Gilles. Closed-loop stability and performance analysis of least squares and minimum-variance control algorithms for multiconjugate adaptive optics. *Appl. Opt.*, 44:993–1002, 2005.

[23] L. Gilles and B. L. Ellerbroek. Split atmospheric tomography using laser and natural guide stars. *J. Opt. Soc. Am.*, 25(10):2427–35, 2008.

[24] L. Gilles, B. L. Ellerbroek, and C. R. Vogel. Preconditioned conjugate gradient wavefront reconstructors for multiconjugate adaptive optics. *Applied Optics*, 42(26):5233–5250, 2003.

[25] L. Gilles, B. L. Ellerbroek, and C. R. Vogel. A comparison of multigrid v-cycle versus fourier domain preconditioning for laser guide star atmospheric tomography. In *Adaptive Optics: Analysis and Methods/Computational Optical Sensing and Imaging/Information Photonics/Signal Recovery and Synthesis Topical Meetings on CD-ROM*, page JTuA1. Optical Society of America, 2007.

[26] L. Gilles, C. R. Vogel, and B. L. Ellerbroek. Multigrid preconditioned conjugate-gradient method for large-scale wave-front reconstruction. *J. Opt. Soc. Am. A*, 19(6):1817–1822, 2002.

[27] S. F. Gilyazov. Iterative solution methods for inconsistent operator equations. *Moscow Univ. Comput. Math. Cybernet*, 3:78–84, 1977.

[28] J.W. Goodman. *Introduction to Fourier Optics*. Roberts & Company Publishers, 3rd edition, Dec. 2004.

[29] J. Hadamard. *Lectures on Cauchy's Problem in Linear Partial Differential Equations*. Yale University Press, New Haven, CT, 1923.

[30] P. J. Hampton, P. Agathoklis, R. Conan, and C. Bradley. Closed-loop control of a woofer–tweeter adaptive optics system using wavelet-based phase reconstruction. *J. Opt. Soc. Am. A*, 27(11):A145–A156, Nov. 2010.

[31] P. J. Hampton and C. Bradley. A new wave-front reconstruction method for adaptive optics systems using wavelets. *Selected Topics in Signal Processing, IEEE Journal of*, 2(5):781 –792, Oct. 2008.

[32] Martin Hanke. *Conjugate gradient type methods for ill-posed problems*. Pitman Research Notes in Mathematics Series. 327. Harlow: Longman Scientific & Technical., 1995.

[33] T. Helin and M. Yudytskiy. Wavelet methods in multi-conjugate adaptive optics. *Inverse Problems*, 29(8):085003, 2013.

[34] A. Ishimaru. *Wave propagation and scattering in random media*, volume 2. Academic Press, New York, 1978.

[35] S. Kaczmarz. Angenäherte Auflösung von Systemen linearer Gleichungen. *Bulletin International de l'Académie Polonaise des Sciences et des Lettres*, 35:355–357, 1937.

[36] J. Kaipio and E. Somersalo. *Statistical and Computational Inverse Problems*, volume 160 of *Applied Mathematical Sciences*. Springer Science+Business Media, Inc., 2005.

[37] E. Klann, R. Ramlau, and L. Reichel. Wavelet-based multilevel methods for linear ill-posed problems. *BIT Numerical Mathematics*, 51(3):669–694, 2011.

[38] A. N. Kolmogorov. The local structure of turbulence in incompressible viscous fluids for very large Reynolds' numbers. *Dokl. Akad. Nauk SSSR*, 30:301–305, 1941.

[39] R. G. Lane, A. Glindemann, and J. C. Dainty. Simulation of a kolmogorov phase screen. *Waves in Random Media*, 2(3):209–224, 1992.

[40] M. Le Louarn, C. Vérinaud, V. Korkiakoski, N. Hubin, and E. Marchetti. Adaptive optics simulations for the European Extremely Large Telescope. In *Proc. SPIE 6272, Advances in Adaptive Optics II*, 2006.

[41] A. K. Louis. Convergence of the conjugate gradient method for compact operators. In H. W. Engl and C. W. Groetsch, editors, *Inverse and Ill-Posed Problems*, pages 177–183. Academic Press, Orlando, 1987.

[42] S. Mallat. *A wavelet tour of signal processing*. Academic Press, San Diego, Calif., 2nd edition, 1999.

[43] E. Marchetti, N. Hubin, E. Fedrigo, et al. MAD the ESO multi-conjugate adaptive optics demonstrator. In *Proc. SPIE 4839, Adaptive Optical System Technologies II*, pages 317–328, 2003.

[44] Y. Meyer and D. H. Salinger. *Wavelets and Operators*. Cambridge Studies in Advanced Mathematics. Cambridge University Press, 1992.

[45] F. Natterer. *The Mathematics of Computerized Tomography*. Wiley, 1986.

[46] A. Nemirovskii. The regularizing properties of the adjoint gradient method in ill-posed problems. *USSR Comp. Math. Math. Phys.*, 26:7–16, 1986.

[47] T. W. Nicholls, G. D. Boreman, and J. C. Dainty. Use of a Shack–Hartmann wavefront sensor to measure deviations from a Kolmogorov phase spectrum. *Opt. Lett.*, 20(24):2460–2462, Dec 1995.

[48] G.C. Papanicolaou and K. Sølna. Wavelet based estimation of local Kolmogorov turbulence. In *Theory and applications of long-range dependence*, pages 473–505. Birkhäuser Boston, Boston, MA, 2003.

[49] W. M. Patterson. *Iterative Methods for the Solution of a Linear Operator Equation in Hilbert Space - A Survey*. Springer, Berlin, Heidelberg, New York, 1974.

[50] D.G. Pérez and L. Zunino. Generalized wavefront phase for non-Kolmogorov turbulence. *Opt. Lett.*, 33(6):572–574, Mar 2008.

[51] C. Petit, J.-M. Conan, C. Kulcsár, and H.-F. Raynaud. Linear quadratic gaussian control for adaptive optics and multiconjugate adaptive optics: experimental and numerical analysis. *J. Opt. Soc. Am. A*, 26(6):1307–1325, Jun 2009.

[52] P. Piatrou and L. Gilles. Robustness study of the pseudo open-loop controller for multiconjugate adaptive optics. *Appl. Opt.*, 44(6):1003–1010, Feb 2005.

[53] L. A. Poyneer, D. T. Gavel, and J. M. Brase. Fast wave-front reconstruction in large adaptive optics systems with use of the Fourier transform. *J. Opt. Soc. Am.*, 19(10):2100, 2002.

[54] R. Ramlau and M. Rosensteiner. An efficient solution to the atmospheric turbulence tomography problem using Kaczmarz iteration. *Inverse Problems*, 28(9):095004, 2012.

[55] F. Roddier, editor. *Adaptive Optics in Astronomy*. Cambridge University Press, Cambridge, U.K., 1999.

[56] M. C. Roggemann and B. Welsh. *Imaging through turbulence*. CRC Press laser and optical science and technology series. CRC Press, 1996.

[57] M. Rosensteiner. Cumulative reconstructor: fast wavefront reconstruction algorithm for Extremely Large Telescopes. *J. Opt. Soc. Am. A*, 28(10):2132–2138, Oct 2011.

[58] M. Rosensteiner. Wavefront reconstruction for extremely large telescopes via CuRe with domain decomposition. *J. Opt. Soc. Am. A*, 29(11):2328–2336, Nov 2012.

[59] M. Rosensteiner and R. Ramlau. Efficient iterative atmospheric tomography reconstruction from LGS and additional tip/tilt measurements. In *SPIE 8447, Adaptive Optics Systems III*, pages 84475S–84475S–6, 2012.

[60] M. Rosensteiner and R. Ramlau. The Kaczmarz algorithm for multi-conjugate adaptive optics with laser guide stars. *J. Opt. Soc. Am. A*, 30(8):1680–1686, 2013.

[61] S. M. Rytov, Y. A. Kravtsov, and V. I. Tatarski. *Principles of Statistical Radiophysics 4*. Springer–Verlag, Berlin, 1989.

[62] Y. Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, second edition, April 2003.

[63] T. Schuster, B. Kaltenbacher, B. Hofmann, and K. Kazimierski. *Regularization Methods in Banach Spaces.* de Gruyter, Berlin, 2012.

[64] S. Siltanen and J. L. Müller. *Linear and Nonlinear Inverse Problems with Practical Applications*. SIAM. SIAM, 2012.

[65] Z. Strakoš and P. Tichý. On error estimation in the conjugate gradient method and why it works in finite precision computations. *Electron. Trans. Numer. Anal.*, 13:56–80 (electronic), 2002.

[66] M. Tallon, I. Tallon-Bosc, C. Béchet, F. Momey, M. Fradin, and É. Thiébaut. Fractal iterative method for fast atmospheric tomography on extremely large telescopes. In *Proc. SPIE 7736, Adaptive Optics Systems II*, pages 77360X–77360X–10, 2010.

[67] M. Tallon, E. Thiébaut, and C. Béchet. A fractal iterative method for fast wavefront reconstruction for extremely large telescopes. In *Adaptive Optics: Analysis and Methods/Computational Optical Sensing and Imaging/Information Photonics/Signal Recovery and Synthesis Topical Meetings on CD-ROM*, page PMA2. Optical Society of America, 2007.

[68] V. Taranenko and O. Shanin. *Adaptive Optics*. Radio i Svyaz', Moscow, 1990. in Russian.

[69] V. I. Tatarski. *Wave propagation in a turbulent medium*. Dover, New York, 1961.

[70] E. Thiébaut and M. Tallon. Fast minimum variance wavefront reconstruction for extremely large telescopes. *J. Opt. Soc. Am. A*, 27:1046–1059, 2010.

[71] A. Tokovinin. Adaptive optics tutorial at CTIO. `http://www.ctio.noao.edu/~atokovin/tutorial/`, 2005.

[72] A. Tokovinin, M. Le Louarn, and M. Sarazin. Isoplanatism in a multiconjugate adaptive optics system. *J. Opt. Soc. Am. A*, 17(10):1819–1827, Oct 2000.

[73] A. Tokovinin and E. Viard. Limiting precision of tomographic phase estimation. *J. Opt. Soc. Am. A*, 18(4):873–882, Apr 2001.

[74] R. K. Tyson. *Introduction to Adaptive Optics*. SPIE Press, Bellingham, WA, 2000.

[75] 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*, 14(17):7487–7498, Aug 2006.

[76] C. R. Vogel and Q. Yang. Multigrid algorithm for least-squares wavefront reconstruction. *Applied Optics*, 45(4):705–715, 2006.

[77] Q. Yang, C. R. Vogel, and B. L. Ellerbroek. Fourier domain preconditioned conjugate gradient algorithm for atmospheric tomography. *Applied Optics*, 45(21):5281–5293, 2006.

[78] M. Yudytskiy, T. Helin, and R. Ramlau. A frequency dependent preconditioned wavelet method for atmospheric tomography. In *Third AO4ELT Conference - Adaptive Optics for Extremely Large Telescopes*, May 2013.

[79] M. Yudytskiy, T. Helin, and R. Ramlau. Finite element-wavelet hybrid algorithm for atmospheric tomography. *J. Opt. Soc. Am. A*, 31(3):550–560, Mar 2014.

Ich, Dipl.-Ing. Mykhaylo Yudytskiy, geboren am 12. Mai 1987, erkläre an Eides statt, dass ich die vorliegende Dissertation selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt bzw. die wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe. Die vorliegende Dissertation ist mit dem elektronisch übermittelten Textdokument identisch.

Linz, April 2014

(Mykhaylo Yudytskiy)

Mykhaylo Yudytskiy

Dipl.-Ing., Bakk.techn.

Geburtstag	12.05.1987
Geburtsort	Kharkov, Ukraine
Staatsangehörigkeit	Ukraine
Familienstand	ledig
Dienstadresse	RICAM Altenberger Straße 69 4040 Linz, Austria
Telefonnummer	+43 732 2468 4113
Email	mykhaylo.yudytskiy@oeaw.ac.at
Web	http://www.mishay.info/

Ausbildung

09.1994 - 01.1999	Schule in Kharkov, Ukraine
01.1999 - 12.1999	Linz International School Auhof
01.2000 - 07.2003	East Lansing Public Schools, Michigan, US
09.2003 - 08.2005	**International Baccalaureate Diploma Program** (bilingual Schulabschluss) Linz International School Auhof
10.2005 - 03.2009	**Bachelorstudium Technische Mathematik** Johannes Kepler University Linz Bachelorarbeit: *Derivative free optimization - geometry of the interpolation sets* Betreuung: A. Univ. Prof. DI Dr. Helmut Gfrerer
03.2009 - 07.2010	**Masterstudium Industriemathematik** (mit Auszeichnung) Johannes Kepler University Linz Masterarbeit: *Stable parameter identification for a second order ODE - analysis and an application from biogeochemistry* Betreuung: Univ. Prof. Dr. Ronny Ramlau, Priv. Doz. DI Dr. Stefan Kindermann

Berufliche Erfahrung

10.2010 - Heute	**Johann Radon Institute for Computational and Applied Mathematics (RICAM)** **Österreichische Akademie der Wissenschaften (ÖAW)** Mitarbeit im Projekt "Mathematical Algorithms for E-ELT Adaptive Optics" für die Europäische Südsternwarte (ESO)
10.2009 - 07.2013	**Johannes Kepler University Linz** Lehre an der Universität im Umfang von 12 Semesterkursen

Veröffentlichungen in wissenschaftlichen Fachzeitschriften

- T. Helin, M. Yudytskiy. *Wavelet methods in multi-conjugate adaptive optics.* Inverse Problems, 29(8):085003, 2013.
- M. Yudytskiy, T. Helin, R. Ramlau. *Finite element-wavelet hybrid algorithm for atmospheric tomography,* J. Opt. Soc. Am. A, 31(3):550-560, Mar 2014.

Proceedingsbeiträge

- M. Yudytskiy, T. Helin, R. Ramlau. *A frequency dependent preconditioned wavelet method for atmospheric tomography,* in Third AO4ELT Conference - Adaptive Optics for Extremely Large Telescopes, 2013.

Wissenschaftliche Berichte (Auswahl)

- M. Yudytskiy, T. Helin, M. Zhariy. *Wavelet-based wavefront reconstruction for SCAO.* Technical Report E-TRE-AAO-528-0005 Issue 2, Austrian In-Kind Contribution - AO, 2011.
- T. Helin and M. Yudytskiy. *Wavelet-based wavefront reconstruction for MCAO.* Technical Report E-TRE-AAO-528-0014 Issue 2, Austrian In-Kind Contribution - AO, 2012.
- M. Yudytskiy. *Fast wavefront reconstruction with wavelet regularization for MCAO.* Technical Report E-TRE-AAO-528-0027 Issue 2, Austrian In-Kind Contribution - AO, 2013.
- M. Rosensteiner and M. Yudytskiy. *Use Cases Reference Results - Quality, AO2.* Technical Report E-TRE-AAO-528-0033, Austrian In-Kind Contribution - AO, 2012.
- M. Yudytskiy and T. Helin. *Numerical effort of the wavelet-based wavefront reconstruction for SCAO.* Technical Report E-TRE-AAO-528-0036 Issue 2, Austrian In-Kind Contribution - AO, 2013.
- G. Auzinger, M. Rosensteiner, and M. Yudytskiy. *Tests for handling spot elongation noise using a SCAO system.* Technical Report E-TRE-AAO-528-0041 Issue 2, Austrian In-Kind Contribution - AO, 2013.
- M. Yudytskiy, T. Helin, and R. Ramlau. *A finite element-wavelet hybrid algorithm for atmospheric tomography.* Technical Report E-TRE-AAO-528-0045 Issue 2, Austrian In-Kind Contribution - AO, 2013.
- M.Yudytskiy. *Use Cases Reference Results - Quality of the Wavelet Method, AO3.* Technical Report E-TRE-AAO-528-0046 Issue 2, Austrian In-Kind Contribution - AO, 2013.
- M. Yudytskiy. *RTC for the Finite Element-Wavelet Iterative Method.* Technical Report E-TRE-AAO-528-0047 Issue 2, Austrian In-Kind Contribution - AO, 2013.
- M. Yudytskiy. *Information on the use of the wavelet reconstructor for LTAO and MOAO.* Technical Report E-TRE-AAO-528-0051, Austrian In-Kind Contribution - AO, 2013.
- M. Yudytskiy. *Analysis of the computational performance of the wavelet method.* Technical Report E-TRE-AAO-528-0052, Austrian In-Kind Contribution - AO, 2013.

Vorträge an Konferenzen und Universitäten

05.2011	Applied Inverse Problems Conference. College Station, Texas, US.
09.2011	Chemnitz Symposium on Inverse Problems. Chemnitz, Deutschland.
03.2012	Mathematics for Innovation: Large and Complex Systems. Tokyo, Japan.
05.2012	Second Annual Workshop on Inverse Problems. Sunne, Schweden.
05.2012	Inverse Problems: Modeling and Simulation. Antalya, Türkei.
10.2012	INAF - Astronomical Observatory of Padova. Padua, Italien.
11.2012	University of Helsinki. Helsinki, Finnland.
12.2012	Real Time Control for Adaptive Optics Workshop. Garching bei München.
12.2012	Inverse Days. Jyväskylä, Finnland.
05.2013	Adaptive Optics for Extremely Large Telescopes, 3rd edition. Florenz, Italien.
09.2013	IFIP TC 7 / 2013 System Modelling and Optimization. Klagenfurt.
11.2013	ESO Lunch Talks. Garching bei München.
12.2013	Inverse Days. Sodankylä, Finnland.
03.2013	AO Tomography Workshop. Edinburgh, UK.

Lehre an der Johannes Kepler Universität Linz

WS 2009/2010	Programmierung (Tutorium)
SS 2009/2010	Computersysteme (Tutorium)
SS 2011/2012/2013	Wavelets (Übung)
WS 2011	Integral equations and boundary value problems (Übung)
SS 2012	Partielle Differentialgleichungen (Übung)
WS 2012	Mathematik 3 (Übung für MechatronikerIn)
WS 2012	Übungen aus Mathematik I (für ChemikerIn)
SS 2013	Übungen aus Mathematik II (für ChemikerIn)

Linz, April 2014