SIMULATION AND ANALYSIS OF TURBULENCE PROFILING NEURAL NETWORKS FOR TOMOGRAPHIC LAYER RECONSTRUCTION AND WIDE FIELD IMAGE CORRECTION IN TELESCOPES

by

R. J. Hamilton

A Dissertation Submitted to the Faculty of the

WYANT COLLEGE OF OPTICAL SCIENCES

In Partial Fulfillment of the Requirements
For the Degree of

DOCTOR OF PHILOSOPHY

In the Graduate College

THE UNIVERSITY OF ARIZONA

2023

THE UNIVERSITY OF ARIZONA
GRADUATE COLLEGE

As members of the Dissertation Committee, we certify that we have read the dissertation prepared by **Ryan Hamilton**, titled ***Simulation and Analysis of Turbulence Profiling Neural Networks for Tomographic Layer Reconstruction and Wide Field Image Correction in Telescopes*** and recommend that it be accepted as fulfilling the dissertation requirement for the Degree of Doctor of Philosophy.

[illegible] Date: 4/21/2023
Professor Michael Hart

[illegible] Date: 4/21/2023
Professor Daewook Kim

Amit Date: 04/21/2023
Professor Amit Ashok

Final approval and acceptance of this dissertation is contingent upon the candidate's submission of the final copies of the dissertation to the Graduate College.

I hereby certify that I have read this dissertation prepared under my direction and recommend that it be accepted as fulfilling the dissertation requirement.

[illegible] Date: 4/21/2023
Professor Michael Hart
Dissertation Committee Chair
Wyant College of Optical Sciences

ACKNOWLEDGEMENTS

I would like to thank everybody, past and present, at the Wyant College of Optical Sciences for 9 years of support and knowledge. The professors, administrators, and fellow students have given me an experience I would not trade for anything.

I would also like to extend special acknowledgments to Tom Milster, who nurtured my initial interests in research and encouraged me to get a PhD; Justin Knight for helping me gain access to a wicked fast computer for all my simulations; Phil Scott for providing a huge body of research to base my work on; and Michael Hart for his guidance on this project and for bestowing me with optics knowledge and intuition that I would not have received from anywhere else.

Lastly, a special thanks to my family for their bottomless support, Lucy for her encouragement of my pursuits and the constant happiness she provides me in life outside of research, and Wally for being the best four-legged office mate I could have ask for.

Contents

List of Figures

List of Tables

ABSTRACT

Light that propagates through the atmosphere is subject to phase perturbations at layers of turbulent flow. For decades, traditional adaptive optics (AO) has used a deformable mirror (DM) to correct the phase at the system pupil. Since the correction is applied at the pupil – not at the layers of turbulence – traditional AO is only valid over a field of view of a few arcseconds for visible light. To obtain wide field image correction, the phase has to be compensated for at optical conjugates to the layers themselves. Doing so with traditional AO hardware increases system cost and complexity because multiple DMs are required. This has motivated the exploration of a software-based image correction technique called multi-object image correction (MOIC). As the cost of computational power continues to improve, MOIC has the potential to become a viable option for wide field turbulence compensation. Thus, the development of simulations and algorithms at the current moment will enable software-based MOIC for a range of applications in the future.

In this work, a simulation of a MOIC system is built. The simulation enables exploration of machine learning based turbulence profiling and offline tomographic layer reconstruction to further wide field image correction without a DM. Chapter 1 provides background on turbulence, AO, MOIC, and machine learning. Chapter 2 describes how Shack-Hartmann wavefront sensor data can be used to measure the position of turbulence in front of the optical system. In Chapter 3, a simulation environment of an imaging system looking through a dynamic atmosphere is developed. We then present the layer signal to noise ratio (SNR) in Chapter 4 and demonstrate that it is a statistically valid metric for quantifying how difficult a layer of turbulence is to find in the atmosphere. Chapter 5 then details a process for using the layer SNR and the simulation environment to condition and generate large sets of data for training turbulence profiling neural networks. In Chapter 6, multi-layer turbulence reconstruction from a known turbulence profile is explored using different decomposition models. A four layer atmosphere is then modeled, measured, reconstructed, and compensated for to demonstrate end-to-end MOIC. Chapter 7 summarizes the work and suggests future areas of research.

Chapter 1 Introduction

1.1 Background

1.1.1 Atmospheric turbulence in optical systems

Any optical system which images distant objects through volumes of the atmosphere is subject to the image blurring effects of turbulence. While the motion of atmospheric air is generally laminar, turbulent flow is always present in regions where high speed winds or different molecular compositions, temperatures, and pressures sheer against each other (USAF 1997). The turbulent motion of the air produces regions with small temperature and pressure fluctuations, resulting in non-uniform refractive index distributions. Non-uniformity of the index of refraction introduces nominal phase shifts between closely separated points in a propagating wavefront, ultimately aberrating the image captured by an optical system looking through the atmosphere.

Fundamentally, turbulent flow occurs when the force of motion in a fluid greatly exceeds the viscous forces which keep the fluid together. While there is no absolute measure of when turbulent flow will begin in a fluid, it is generally characterized by a large Reynolds number:

$$R = \frac{|\vec{v}|L_0}{\eta_v}, \tag{1.1}$$

where $\vec{v}$ is the average velocity of the fluid of size L_0 with viscosity η_v. Once the Reynolds number becomes sufficiently large, such that the fluid flow becomes turbulent, the fluid of size L_0 will swirl and cascade into smaller swirls called eddies. This cascade continues until eddies of some small size ℓ_0 are generated with a Reynolds number small enough to stop the swirling motion and dissipate the system energy into heat. Due to turbulence occurring within the inertial range $[\ell_0, L_0]$, the large starting scale length, L_0, is the outer scale of the turbulence and the small ending scale length, ℓ_0, is the inner scale (Andrews and Philips 2012).

To describe turbulent motion, statistical representations are required to capture the general trends of the chaotic swirling. While a universal description of turbulent flow for all fluids does not exist, there are models which can accurately represent the fluctuations of phase in a wavefront of light traveling through a volume of atmospheric turbulence. The statistics of the phase fluctuations are described by the structure function (Tatarski 1961)

$$D(r) = \left\langle (\phi(\rho) - \phi(\rho + r))^2 \right\rangle, \tag{1.2}$$

where $\phi(\rho)$ is the function for the turbulent phase. If we assume that the turbulence is homogeneous, isotropic, and stationary in increments over the region of interest – an acceptable assumption for fully developed turbulence in an optical aperture – we can find additional more convenient forms of the structure function. One particularly useful representation is

$$D(r) = 6.88\,(r/r_0)^{5/3}, \tag{1.3}$$

where r_0 is the Fried length (Fried 1965). The Fried length is a monochromatic constant which is related to the strength of the phase aberrations in a particular turbulent flow. In its formal definition, r_0 is the correlation length of the phase statistics in the system pupil. However, if we assume that the atmosphere is made up of a series of statistically independent thin layers of turbulence, indexed $\ell \in [1, L]$ each with Fried length $r_{0\ell}$, the total coherence length in the system pupil can be written as

$$r_0^{-5/3} = \sum_{\ell=1}^{L} r_{0\ell}^{-5/3}. \tag{1.4}$$

There are additional conveniences to using r_0 as the defining constant of turbulence. The influence of r_0 on an optical system can be observed directly by setting Eq. (1.2) equal to Eq. (1.3). When $r = r_0$ the average phase variance between the functions $\phi(\rho)$ and $\phi(\rho + r)$ is 6.88 $\approx$ 2π rad^2 – approximately one wave of pupil aberration. As r_0 decreases, the average phase variance between the two points will increase. As r_0 increases, the average phase variance will decrease. This reinforces the idea that r_0 is an indicator of the aberrating strength of the turbulence. We can extrapolate this relationship to write the phase error in an optical system with a circular entrance pupil of diameter D as

$$\sigma_s^2 = 1.03 \left(\frac{D}{r_0} \right)^{5/3}, \tag{1.5}$$

where σ_s^2 is in units of rad^2 of pupil aberration (Hardy 1998).

The immediate implication of this relationship is that it is not possible to improve the angular resolving capabilities of an optical system which images through the atmosphere by only increasing D. To obtain the increased diffraction resolution enabled by having a large entrance pupil, blurring due to atmospheric turbulence must be compensated. Since the Fried length at the ground is usually on the order of 5-20 cm for green light (Andrews and Philips 2012), it can be expected that any optical system with a pupil larger than that will be seeing limited – i.e. resolution limited by turbulence rather than diffraction. Note that this relationship also inform us that systems with an aperture smaller than the Fried length are approximately unaffected by atmospheric turbulence and therefore do no need atmospheric compensation.

The top two rows of Fig. (1.1) illustrate how the long exposure point spread function (PSF) of a telescope with $D = 2$ m is affected by turbulence with varying r_0 values. The PSF is the image of an unresolved point in the object field, such as a distant star. From the simulated examples it can be seen that the expected average turbulence values of 5-20 cm will cause the system to be seeing limited. It also demonstrates that as r_0 approaches D the blurring from the atmosphere becomes increasingly negligible. The effect on measured signal is also observable by looking at the color bars. The maximum pixel value measured by the diffraction limited image is over an order of magnitude greater than for the $r_0 = 0.15$ m. The amount of photon energy is approximately equal in each case, but stronger turbulence forces the energy over a larger area. The same process was applied to an extended scene of the ISS, the results of which are shown in the bottom two rows of Fig. (1.1).

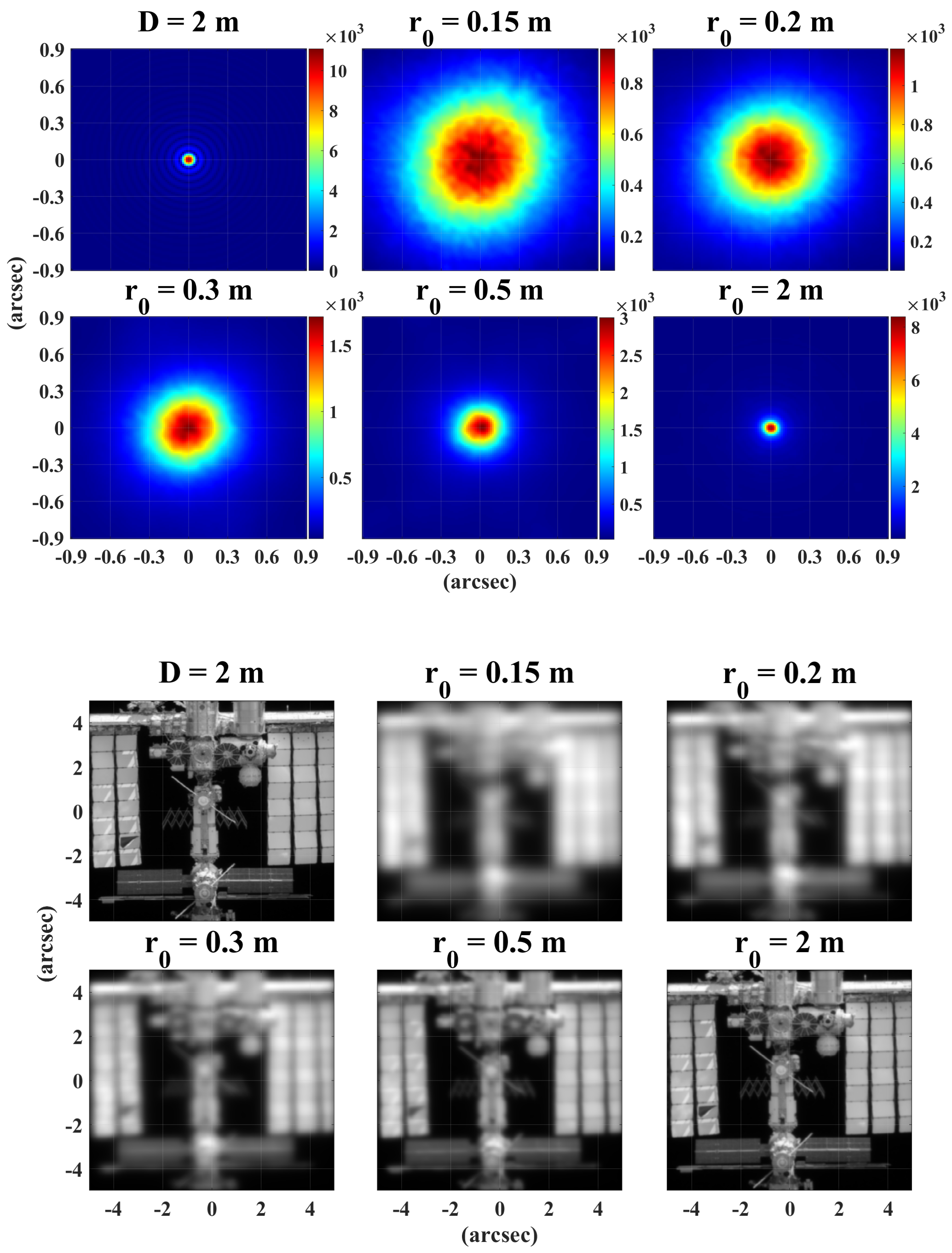


FIGURE 1.1: Simulation of images formed by a $D = 2$ m telescope for different turbulence strengths. The top two rows are for point sources, thereby demonstrating the system PSF. These images are from a 2 minute exposure and are shown on a square root scale to improve contrast. The bottom two rows are images when looking at the center module of the ISS with perfect tracking for 30 seconds. The images labeled $D = 2$ m are the diffraction limited images of the system.

1.1.2 Traditional adaptive optics

Adaptive optics (AO) is the dominant method used to compensate for turbulence in optical systems which are seeing limited. The traditional architecture for adaptive optics starts with a wavefront sensor (WFS) which measures phase in the system pupil using light collected from a bright reference object. The WFS data is then interpreted by a computer which sends commands to a deformable mirror (DM) placed in the system upstream from the final image sensor. The deformable mirror is given the inverse shape of the aberrated wavefront, dynamically removing the aberrations caused by turbulence before the final image is formed (Tyson 2011).

A limitation to traditional AO stems from the fact that the WFS measures the phase errors in the pupil exclusively along the line of sight of the reference object. In reality, the phase which is aberrating the wavefront originates at multiple layers of turbulence distributed out in front of the optical system. Therefore, the shapes sent to the DM will only provide accurate correction for the reference object line of sight. This disagreement in phase between two angularly separated objects observed by an optical system through turbulence is known as anisoplanatism and it fundamentally restricts the field of view of traditional AO. The system geometry which produces anisoplanatism is illustrated in Fig. (1.2).

Since anisoplanatism is an effect that decreases AO system performance with increasing angle from the reference object, we can perceive an angular separation to a science object which can be considered isoplanatic – i.e. unaffected by anisoplanatism. This parameter is given by

$$\theta_0 = 0.31\frac{r_{0\ell}}{h_\ell}, \tag{1.6}$$

where h_ℓ is the distance from the system aperture to the layer and θ_0 is the isoplanatic angle of the layer with Fried length $r_{0\ell}$ (Fried 1982). θ_0 gives us a direct method for estimating the wavefront error in waves for applying correction to an object at an angle θ away from the reference object:

$$\sigma_a^2 = \left(\frac{\theta}{\theta_0}\right)^{5/3}. \tag{1.7}$$

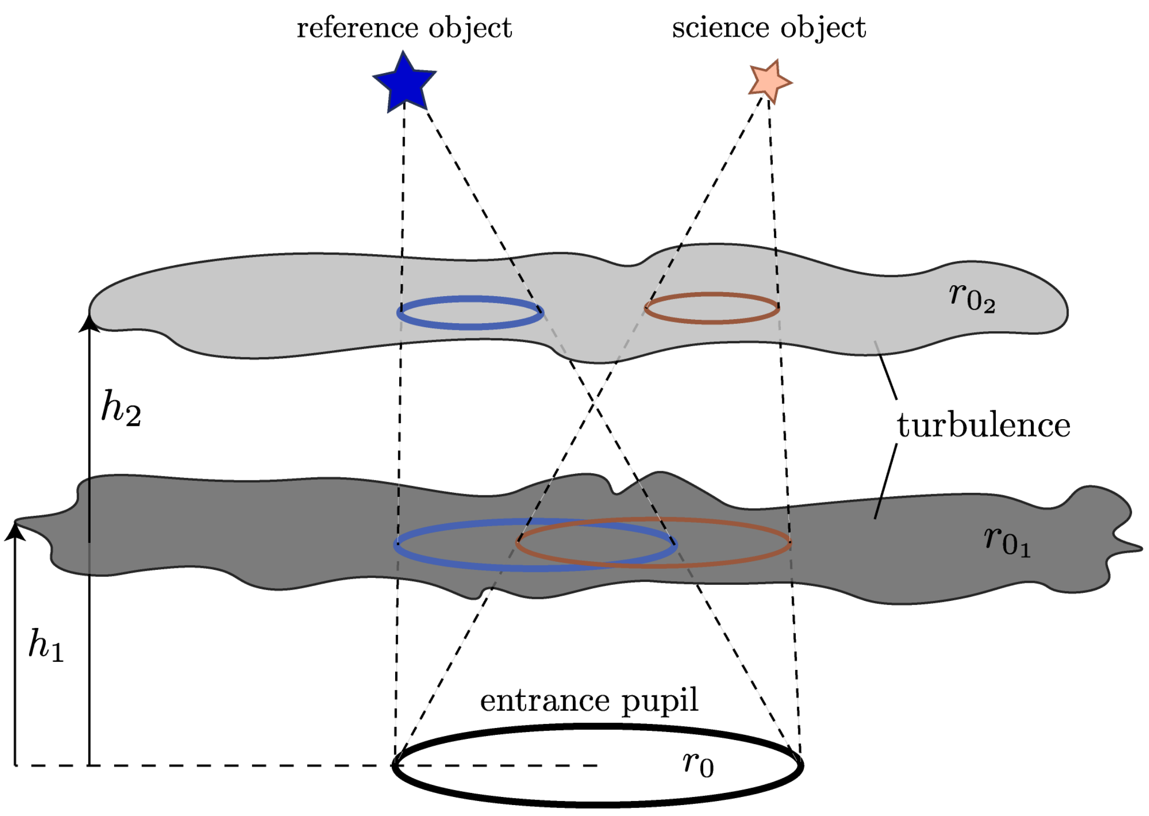


FIGURE 1.2: Illustration of how phase anisoplanatism occurs from multi-layered turbulence out in front of an optical system between a traditional AO reference object and some angularly separated science object. It can be seen here that anisoplanatism is worse from distant layers since a near layer of turbulence, such as the layer (r_{0_1}, h_1), will have more phase overlap than a distant layer of turbulence, such as layer (r_{0_2}, h_2).

By using Eq. (1.6) and Eq. (1.7), along with the average values of r_0 mentioned in Sec. (1.1.1), it can be shown that the average isoplanatic angle for green light traveling through a few kilometers of atmosphere is on the order of a few arcseconds.

The direct solution to limit anisoplanatic errors in traditional AO correction is to minimize the angular separation of the reference and science objects. Ideally, the science object is bright enough to be used as a natural reference source. Due to the sparsity of bright astronomical sources, it has become common practice to generate artificial reference sources with lasers, called laser guide stars, at telescope facilities (Wizinowich et al. 2006). This is a key component for traditional AO because the longer the WFS takes to collect sufficient signal to reconstruct the wavefront the more the atmosphere will change before a correction is applied. The time window for accurate adaptive compensation is called the coherence time. As the temporal analog to the Fried length, the coherence time can be estimated using the expression

$$\tau_0 = 0.314 \frac{r_0}{|\vec{v}|}, \tag{1.8}$$

where $\vec{v}$ is the group velocity of the turbulence in the pupil or at a specific layer. Just as with the isoplanatic angle, we can use the coherence time to estimate the pupil wavefront error in waves from correction lag:

$$\sigma_t^2 = \left(\frac{\delta t}{\tau_0}\right)^{5/3}, \tag{1.9}$$

where δt is the time between a WFS measurement and a correction made by a traditional AO system DM. Using the average values of r_0 mentioned in Sec. (1.1.1) with typical wind speeds in atmospheric turbulence (USAF 1997), we find that the coherence time of green light is on the order of $\sim 1 - 10$ ms.

Together, r_0, θ_0, and τ_0 account for the fundamental spatial, angular, and temporal scales of traditional AO, respectively. r_0 limits the size of the pupil which does not need AO correction and indicates the strength of the phase fluctuations. τ_0 puts a requirement on how quickly the measurement and correction process needs to be to provide accurate real-time image correction. θ_0 limits the field of view available to an optical system receiving correction from a traditional AO system. While r_0 and τ_0 are fundamental to each individual layer of turbulence, θ_0 is an artifact of how traditional AO measures and corrects for all of the layers of turbulence simultaneously at the pupil. By taking a more comprehensive approach to the turbulence measurement and reconstruction process, the maximum field of view of correction can be greatly improved over that of traditional AO.

1.1.3 Wide field image correction

To obtain reliable turbulence correction over a field of view wider than the isoplanatic angle, wavefront reconstruction cannot be treated in the traditional sense. The prevailing treatment is to reconstruct the phase as multiple infinitely thin layers located out in front of the optical system. While the turbulent phase actually comes from three dimensional volumes of atmosphere, it is reasonable to assume that from a fixed observation point the air volumes can be flattened into infinitely thin screens of phase which a propagating wave will accumulate upon transmission of the layer volume (Roggemann and Welsh 1996). This allows us to simplify the reconstruction process from one three dimensional problem down to a series of two dimensional reconstructions.

Assuming that wavefront sensor data can be used to tomographically reconstruct multiple layers of turbulent phase at their approximate distances, a topic to be covered in Chapter 6, the geometric relationships which led to anisoplanatism in the correction loop are remedied. Namely, the phase is no longer collapsed into the pupil along the sole line of sight of the reference object. Instead, the unique phase for each layer of turbulence along each line of sight can be accounted for since the phase at each layer is known. Correction can then be applied per layer, as is done in multi-conjugate adaptive optics (MCAO), or per line of sight, as is done in multi-object adaptive optics (MOAO).

Multi-conjugate adaptive optics

MCAO is a technique which places DMs at the optical conjugate of the reconstructed layer of turbulence. By removing each layer, the turbulence is corrected and anisoplanatism is accounted for. Mathematically this is sound, but there are practical issues with implementing this which make MCAO easier said than done.

The three primary atmospheric challenges for MCAO are that layers can be at any distance from the system, the number of layers of turbulence can change, and the distance to a given layer can drift. Since layer correction is performed with hardware, it is impossible to fully generalize the hardware to account for all possible configurations of turbulence in the atmosphere. This is commonly addressed by fixing the number of deformable mirrors and predetermining their conjugate distances to ranges where turbulence is common at the observation facility (Beckers 1988). Fixing the mirror locations provides constraints on DM size requirements and where to place them in the system. It also means that the number of reconstructed layers and their distances are already known, thereby simplifying the layer reconstruction process. Several MCAO systems have been in operation for years using the fixed DM architecture, primarily at large telescope sites. At these sites, MCAO has been proven capable of obtaining near-traditional AO performance over a field of view of 1 arcmin^2 (Rigaut and Neichel 2018). However, obtaining a high degree of correction beyond this field of view has proven difficult.

Due to the system requiring N DMs to correct N layers, and the inherent mechanical and optical alignment issues with moving the DMs during operation, system performance is dependent on how closely the real turbulence matches the built and modeled system. In the case where there are more layers of turbulence than DMs, the turbulence reconstruction algorithm will offload the additional layers to the nearest mirrors, resulting in aliasing errors. As the separation between a conjugated DM altitude and the actual layer increases, the highest spatial frequency which can be measured and corrected in that layer decreases (Johnston and Welsh 1994). To mitigate this effect requires increasing the number of DMs to obtain greater coverage. This will inevitably increases system cost and complexity due to the fact that DMs are expensive, require calibration, and will scale in size with the conjugated layer altitude.

The cost restrictions of MCAO have made it a tomographic AO solution primarily for extremely large telescopes (ELTs). Additionally, since ELTs are built at special astronomical sites with decades of documented atmospheric turbulence measurement data, it is possible to make informed decisions about how many deformable mirrors are needed as well as their conjugate altitudes. This does imply that smaller budget projects and imaging systems which move to different observation sites are less likely to find consistent performance from MCAO due to mismatches between the fixed DM conjugate distances and the real atmosphere at a given moment.

Multi-object adaptive optics

MOAO corrects phase conjugated at the pupil, just like traditional AO, but it applies a specific correction for each object line of sight. This requires more computational resources than MCAO. One particularly challenging aspect of MOAO is determining where each layer of turbulence is located. Since the DMs are not conjugated to specific layer distances, as is the case for MCAO, the layer reconstruction algorithm needs to be told where to reconstruct each layer at. The degree to which the reconstructed layers provide correction and mitigate anisoplanatism is highly sensitive to the provided information on layer location (B. Neichel et al. 2009, Costille and Fusco 2012). MOAO has been demonstrated on-sky with performance similar to traditional AO over a wider field of view (Gendron et al. 2011), but doing so has required the use of sophisticated model fitting techniques which estimate the distance to each layer of turbulence (Vidal et al. 2010a).

MOAO has two primary challenges to overcome to increase corrected field of view: hardware complexity and computational power. The original schematic for MOAO was to build one traditional AO system per object of interest (Dekany et al. 2004). This means one WFS and DM for each field. However, given enough reference objects, it is possible to use a single WFS to obtain all of the object-specific pupil phase needed for reconstruction. As for computational requirements, it is both time consuming and memory intensive to reconstruct layers of turbulence at any specific altitude, determine the field dependent phase, and then send commands to each DM within the τ_0 value of the atmosphere.

Software based multi-object image correction

A solution to the mechanical cost and complexity of MOAO is software based multi-object image correction (MOIC) (Scott 2021). Layer detection and reconstruction is identical between MOAO and MOIC, but the two techniques differ at the correction step. MOIC uses the reconstructed layers and the field-specific phase in the pupil to compute the point spread function (PSF) for each object. This anisoplanatic PSF can then be used to deconvolve the unique blur caused by the atmosphere for each object region to estimate the image before turbulence. Since this method does not provide active correction of the wavefront, it is an image correction operation rather than an adaptive optics technique. The downside here is that the image formed on the system image detector is not deblurred which decreases the signal to noise ratio (SNR) of the image. The upside of this, however, is that all of the data needed for offline correction is available and can be computed at any time.

MOIC is primarily limited by computational requirements. Given the constant improvement in the cost of computational power, MOIC is a wide field turbulence compensation process that will become less expensive and better performing in the future. Since the only hardware requirements are a wavefront sensor, multiple objects, and a computer, MOIC is compact and is free from having one or more DMs which is favorable for system cost, complexity, and size. This means that MOIC could be implemented on ELTs, small telescopes, mobile terrestrial imaging systems, and Earth-viewing satellites alike.

1.1.4 Machine learning and artificial neural networks

Machine learning (ML) is used to describe any computer algorithm or program which uses information and experience to learn a task (Mitchell 1997). From self driving vehicles, to targeted advertising on social media platforms, and to classifying catalogues of astronomical objects in sky survey images, different ML models have been developed to solve a multitude of problems in a diverse range of fields over the past few decades. ML models can look very different from one another in both their theoretical behaviors as well as their implementations in code. A throughline, however, is that the models must be trained on data to learn the behavior needed to accomplish the prescribed task. While different ML models have been shown to excel at solving different tasks, few have shown to be as capable at solve a range of different problems as artificial neural networks (ANNs).

ANNs are a ML model inspired by the biological network of neurons in animal brains. An ANN consist of an interconnected network of nodes. The first layer of nodes receives input information of a predetermined size, such as an RGB image or accelerometer data, and produces weighted responses. Each weighted response is then sent to another layer of nodes, and the process is repeated until an output layer is reached. The output layer defines the high-level interpretation of the input stimulus by the network. Thus, for a particular input stimulus there is a distinct cascade of responses throughout the network that results in a specific output result. This could be a simple output, such as a binary response indicating an accelerometer just sensed a person taking a step rather than jumping in place, or a more sophisticated output, such as a vector of labels stating that the input image contains a chair and a person but does not contain 10 other objects of interest.

To create an ANN which can produce the desired output from a particular input, the cascades of node responses have to be trained from an initial model. Training occurs with known data containing a diverse set of inputs which are representative of the problem, as well as the correct output for each input. After exposing the network to each input, the actual outputs are compared to the known correct output to compute an error metric. The node responses are tweaked based on a learning rule to reduce the error metric, and the process is repeated for all of the data. Different learning rules and error metrics can result in ANNs that are better at different tasks and are more resilient to noise.

Even though the idea of ANNs has been around for 80 years (McCulloch and Pitts 1943), the recent boom in implementation in the past decade is the result of modern computing. Open source software packages, like PyTorch and TensorFlow, contain function libraries which have generalized and automated the complicated network generation, training, and performance evaluation processes. These libraries for creating and training the networks are programmed to run quickly with CPUs and GPUs that are widely available on inexpensive computers – ultimately making ANNs a practical ML model for anybody to use.

With widely available ANN generation and training libraries, the current practical limitation in using them to solve most problems is obtaining sufficient training data. The training data has to be both properly labeled at the output and sufficiently large to be representative of the diversity of possible inputs during real world operation. As the complexity of the problem increases, more data is required to properly teach the networks to respond to intricacies in the input. Additionally, if the training set is not consistent, such as containing improperly labeled or noisy data, the learning process might not work at all - resulting in a ‘trained’ network that is actually a random or incorrect guessing machine. Therefore, implementing ANNs to solve particular problems comes down to providing a sufficient amount of accurate training data to solve a learnable problem.

1.2 Motivation

Tomographic layer reconstruction is a holistic approach for determining the phase errors in an optical system caused by turbulence in the atmosphere. By providing the full treatment of multiple layers of turbulence, there is no upper limit to the field of view over which turbulence correction can be applied. The trade-off is increased computational requirements – which have only became feasible in the recent past in the forms of MCAO and MOAO. However, these techniques rely on expensive hardware to apply correction, such as multiple DMs, which limits the practicality of obtaining correction over an arbitrarily large field of view. Software based MOIC is not bound by hardware to obtain correction over a large field of view, but remains a few generations of computation away from practical implementation. Under the expectation that the cost of computer power will reach the point where software based MOIC is feasible, we propose that now is the time to begin building up the models and algorithms to implement MOIC in on-sky systems.

Since most of the literature for atmospheric tomography has been for MCAO and MOAO systems built in the 2000s and early 2010s, the data processing techniques they use generally predate widely available machine learning techniques. As machine learning becomes increasingly more ubiquitous and open access, it is worthwhile to investigate the possibility of using neural networks as a replacement for the intricate fitting techniques used by current systems for the turbulence profiling portion of the tomographic layer reconstruction process (Vidal et al. 2010a, Scott 2021). Therefore, while building on the literature of MOIC we aim to use modern machine learning techniques where it is suitable to replace older and less accessible data interpretation algorithms.

1.3 Scope of work

The goal of this dissertation is to expand on MOIC by building on the work of Phil Scott (Scott 2021) using the software based correction scheme in Fig. (1.3). This requires developing a dynamic turbulence simulation where any number of layers of turbulence can be modeled at any distance in front of a chosen optical system. The dynamic model allows us to generate large batches of synthetic WFS data and telescope images on a desktop computer. The simulated data is then used to explore machine learning techniques for measuring layer position and velocity information. This is enabled by the development of a theory for the SNR of a single layer of turbulence in a multilayer atmosphere. The SNR is shown to be essential for simulating data which is valid for training turbulence profiling neural networks. For completeness, a modified version of Scott's turbulence reconstruction method is also presented, and the image correction process is demonstrated. The scalability of turbulence profiling neural networks and Scott's modified layer reconstruction method is discussed for different applications to advise future work on MOIC for wide field turbulence compensation.

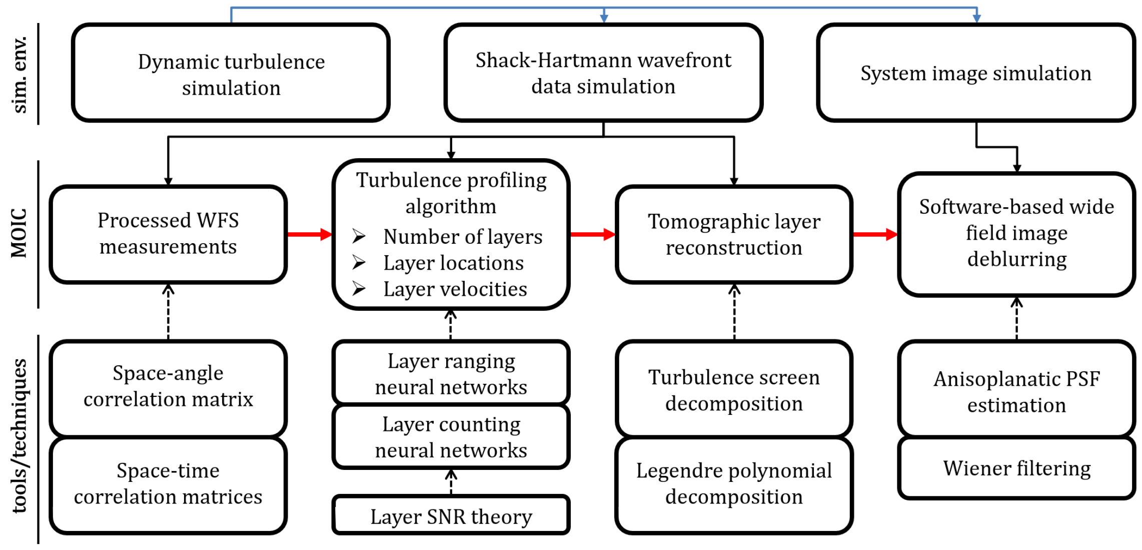

sim. env.
MOIC
tools/techniques
Dynamic turbulence simulation
Shack-Hartmann wavefront data simulation
System image simulation
Processed WFS measurements
Turbulence profiling algorithm
Number of layers
Layer locations
Layer velocities
Tomographic layer reconstruction
Software-based wide field image deblurring
Space-angle correlation matrix
Space-time correlation matrices
Layer ranging neural networks
Layer counting neural networks
Layer SNR theory
Turbulence screen decomposition
Legendre polynomial decomposition
Anisoplanatic PSF estimation
Wiener filtering

FIGURE 1.3: Software based MOIC correction scheme used. The tools and techniques which were investigated are specified for each component in the MOIC process.

Chapter 2 Turbulence profiling theory

2.1 Shack-Hartmann wavefront sensors

To reconstruct multiple layers of turbulence, the distance to each layer has to be provided to constrain the reconstruction process. While there are multiple documented techniques for finding the distance to a layer of turbulence, we focus on the methods which use data from a Shack-Hartmann wavefront sensor (SHWFS). As common and inexpensive WFS devices, layer ranging with SHWFS data is a generalizable and an accessible option for MOIC which is applicable in a variety of wide field imaging applications.

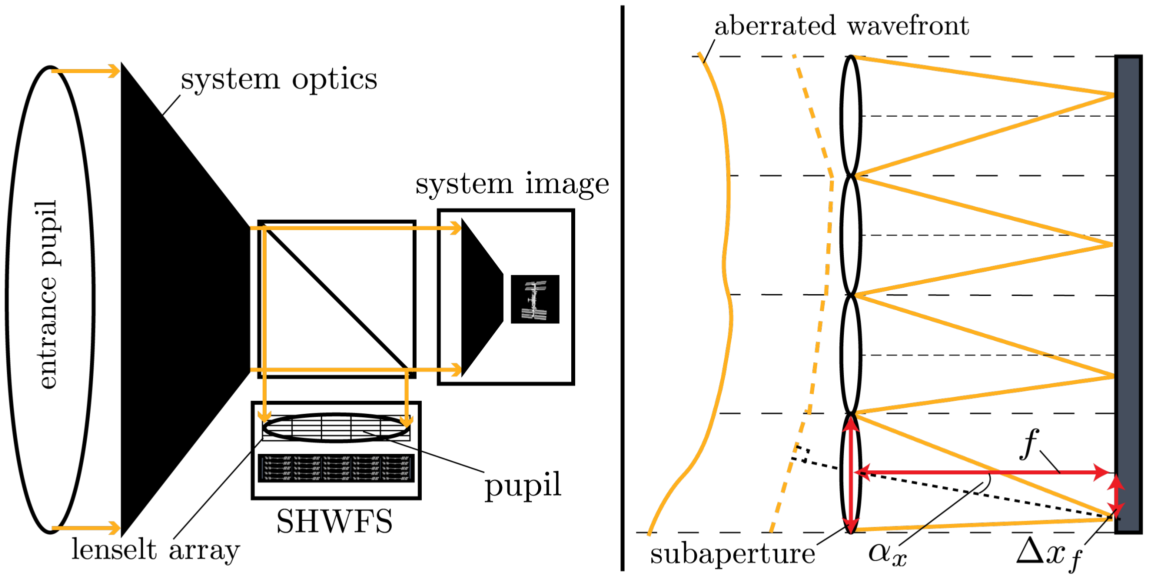


FIGURE 2.1: Left: Standard placement of a SHWFS in a system. Light going to the system image is picked off and a lenslet array is placed at a plane conjugate to the system entrance pupil. Each lenslet forms an image of the object scene. Right: Cross-section of an aberrated wavefront interacting with SHWFS lenslets. The tilt of each wavefront segment is illustrated as the dotted line between the wavefront itself and each lenslet, and the shift of the on-axis image location due to the wavefront tilt is shown.

2.1.1 Device architecture

The layout of a SHWFS is given in Fig. (2.1). A lenslet array is placed conjugate to the system entrance pupil, effectively turning the pupil into a grid of subapertures. Each subaperture forms its own system image of the scene and the aberration content in each sub-image is specific to the location of each subaperture in the system entrance pupil. To estimate the wavefront from a SHWFS we approximate the wavefront as flat over each subaperture. This requires that the side length of each subaperture, d, meets the condition

$$d \leq r_0. \tag{2.1}$$

Under this principle small shifts of the image, $(\Delta x_f, \Delta y_f)$, can be approximated as flat segments containing the local tip and tilt of the pupil phase:

$$\alpha_x = \tan^{-1}(\Delta x_f / f), \tag{2.2}$$

where f is the lenslet focal length. By determining the shift of each image, Eq. (2.2) can be used to estimate the tip and tilt of the wavefront over each subaperture.

The subaperture images are processed and the tip/tilt coefficients for each subaperture, indexed $b \in [1, B]$, are stored in a vector. For a SHWFS viewing multiple objects, the phase in each sub-aperture from objects with angular separations exceeding the isoplanatic angle will vary nominally (see Sec. (1.1.2)). To account for this, the image is segmented into subfields indexed $a \in [1, A]$, and the local image motion from each subfield if found independently. The resulting pupil slopes are calculated and grouped into an $A \times B$ matrix of subfield-subaperture tip/tilt measurements. For the remainder of this work, (a, b) will be used to specifying a specific subfield a viewed within subaperture b.

The matrix containing every subfield-subaperture slope (a, b) is denoted as the system warp map, $\mathbf{w}$. $\mathbf{w}$ contains sufficient information to find the distance to the layers of turbulence as well as to reconstruct them (Scott 2021). A cadence of warp map matrices collected at different times can also be used to enable layer velocity estimation. Since SHWFS data is used to generate **w**, which in turn enables MOIC, it is important that the fundamentals of SHWFS operation are covered here so that convincing SHWFS outputs can be generated by the system data generation block in Fig. (1.3)

2.1.2 Image motion estimation

There are many techniques for extracting image motion estimations from individual subaperture images. Two common methods are point-source centroiding and matched filtering. Both of these techniques relate the intended position of the object on the detector for the unaberrated system to the measured position of the object after turbulence has introduced tip and tilt. The differences between the two techniques lend each method to different wavefront sensing applications.

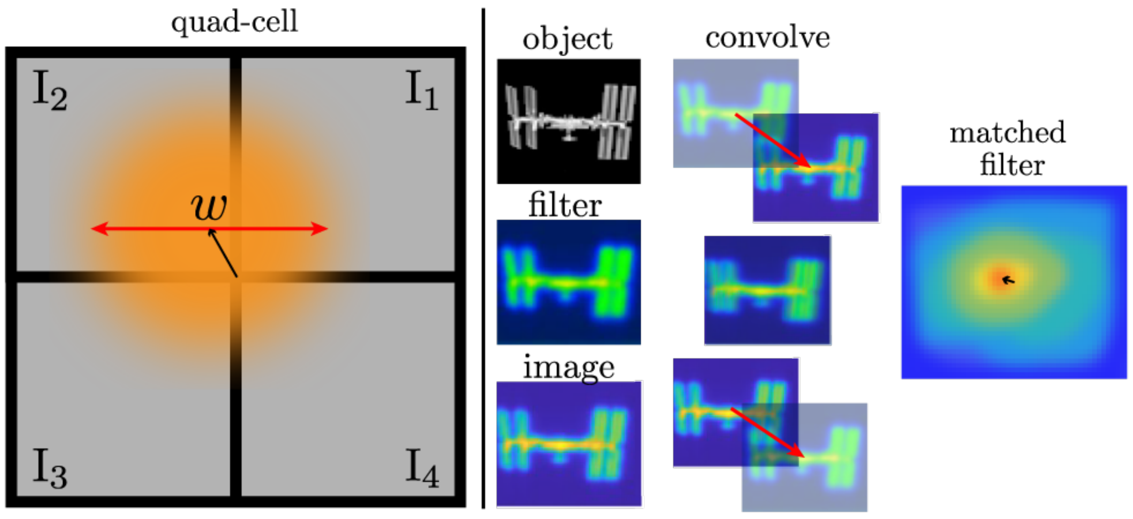


FIGURE 2.2: Left: Illustration of a quad-cell centroid measurement architecture. The image is of a point source, which is centered on the shared corner of the four pixels in the cell in the absence of turbulence. In this example, tip and tilt has shifted the center of the PSF of width w by the arrow which starts at the cell center. Right: Overview of the matched filter process for an extended object. The filter is the time averaged image of the object. The center column of images illustrates the 2D convolution process which produces the matched filter output in the right-most column. The shift in the image can be seen in the output of the 2D convolution as the arrow starting at the image center (these results are illustrative and were not calculated explicitly).

Point-source centroiding

The statistical centroids of the image $I(x,y)$ are given by the ratios (Tyson 2011)

$$x_0 = \frac{\iint_\infty xI(x,y)\,dx\,dy}{\iint_\infty I(x,y)\,dx\,dy}, \tag{2.3}$$

$$y_0 = \frac{\iint_\infty yI(x,y)\,dx\,dy}{\iint_\infty I(x,y)\,dx\,dy}. \tag{2.4}$$

The above expressions are for continuous intensity distributions over all space, making it impractical to implement on a discrete sampling generated by a detector array. However, one particularly simple case for centroiding is a point object imaged onto the shared corners of a 2×2 cell of pixels. This geometry, referred to as a bicell or quad-cell SHWFS, is illustrated on the left side of Fig. (2.2). In this configuration, the above expressions simplify and the measured intensities in each pixel of the quad cell can be used to calculate the centroids:

$$x_0 = \gamma \frac{(I_1 + I_4) - (I_2 + I_3)}{I_1 + I_2 + I_3 + I_4}, \tag{2.5}$$

$$y_0 = \gamma \frac{(I_1 + I_2) - (I_3 + I_4)}{I_1 + I_2 + I_3 + I_4}, \tag{2.6}$$

where γ is a scale factor which is a function of the width w and changes depending on shape of the spot intensity pattern. For example, a Gaussian spot with standard deviation $\sigma = w/2$ can obtain accurate scaling with $\gamma = \sigma\sqrt{\pi/2}$ (Thomas et al. 2006).

When estimating the position of a diffraction limited spot from a quad-cell, the uncertainty in the spot position can take the forms

$$\sigma_{\Delta x} = \frac{\lambda}{2w\sqrt{n_s}}, \tag{2.7}$$

$$\sigma_{\Delta x} = 1.05\frac{\sqrt{I_B}\lambda p}{wn_s}. \tag{2.8}$$

In the above expressions λ is the wavelength of the light, n_s is the number of signal photoelectrons on the quad-cell, p is the pixel size, and I_B is the background intensity in photoelectrons per steradian. Eq. (2.7) is for a quad-cell with weak background noise while Eq. (2.8) considers a strong background flux relative to the signal. Centroiding can also be carried out for a point source imaged onto a cell of pixels which is larger than 2×2. The forms of the spot position uncertainty in the case of strong and weak background change from the quad-cell case, but still depend on all the exact same parameters with the same proportionality (Hardy 1998).

Matched filtering

Since a centroiding algorithm requires prior knowledge of the intensity distribution, $I(x,y)$, it is impractical to directly centroid images of arbitrary extended scenes. This is where matched filtering can be used. Matched filtering is the process of convolving an image with a reference filter to determine how the image has shifted relative to the reference. If the filter object is somewhere in the image, the output of the convolution will have peaks where the shared object is located. Therefore, the matched filtering operation is a method which takes a complex arbitrary scene and turns it into a smoothed image with maxima and minima. When the filter is a shifted version of the image being filtered, the location of the global maximum will be a distance from the center equal to the image shift. The center of the shift can be found to sub-pixel accuracy using a parabolic interpolation of the global maximum. Eq. (2.2) can then be used to find the corresponding subfield-subaperture slope values.

The statistical properties of turbulence give us some indication as to what might be a quick and easy filter to use. Since the average tip and tilt of turbulence is zero (Tyson 2011), the long exposure of each subaperture image is representative of the object without tip and tilt. Thus, to estimate image motion from turbulence, the time-averaged scene can used as a reliable filter object.

Due to the randomness of natural scenes, matched filtering needs to be tuned to work for specific object structures. Because of this, the amount of image shift uncertainty from scene-based matched filtering is primarily a function of the statistics of the scene. Scenes which have repeated patterns can give rise to multiple peaks which must be disambiguated to determine which peak should be used for image shift estimation. Scenes with high spatial frequency content have less uncertainty due to more distinguishing features between similar shapes. Bright scenes follow the same proportionality to measured signal and noise photoelectrons as quad-cell centroiding, meaning that high signal and high spatial frequency scenes will generally have low image shift uncertainty (Poyneer 2003).

2.1.3 Assumptions for fast SHWFS data simulation

The amount of memory required to store raw SHWFS images is much larger than the size of corresponding tip and tilt coefficients for each subfield-subaperture pair. Even a quad-cell SHWFS image will occupy twice as much memory as the resulting tip/tilt vector. Additionally, the process of forming each subaperture image along each line of sight for a system with a wide field of view in simulation is extraordinarily time intensive.

To increase the amount of SHWFS data available in this work, we will forego the image formation and matched filtering process when generating batches of SHWFS slope data. Instead, we will collect the subfield-subaperture specific phases needed to form each image and extract the tip and tilt from that data directly. The cost of this assumption is that the synthetic SHWFS data will not natively contain noise terms which come from performing the matched filtering process. By assuming we are looking at bright extended scenes with sufficient high spatial frequency content, thereby decreasing the tip/tilt uncertainty in our measurements, we assume that this will not introduce intractable errors which would otherwise render our results and conclusions incorrect.

2.2 Slope detection and ranging

SLOpe Detection And Ranging (SLODAR) was originally designed to estimate the average diurnal and seasonal C_n^2 profile at astronomical sites (Wilson 2002). Knowing the average turbulence at an astronomical site means the average r_0 as a function of altitude can be known, informing engineers and astronomers on subjects such as best altitudes to conjugate MCAO DMs to (see Sec. (1.1.3) for more on this). Later work with SLODAR has demonstrated that it is a viable technique for determining the instantaneous C_n^2 profile for the purpose of tomographic layer reconstruction (Butterley et al. 2006, Farley et al. 2020, Shikhovtsev 2022). Warp map data typically used for SLODAR has also been shown to be interpretable by trained neural networks (Hamilton and Hart 2022) – a process which will be developed in detail later in this work. As a proven turbulence telemetry extraction method, we focus on it here as the technique to be used for finding layers of turbulence and their velocities from our simulated SHWFS data.

2.2.1 Slope detection and ranging toy model

SLODAR is a triangulation technique which seeks to find the phase correlation at discrete space and angle samples. The spatial sampling is provided by the separation of each pair of subapertures in the pupil. Angular sampling is provided by having multiple objects in the field. To understand how this sampling is used to determine the turbulence profile, consider the SLODAR toy model in Fig. (2.3). The toy model consists of a linear SHWFS array made of three subapertures viewing two point sources located at infinity.

We will keep track of the SHWFS measurements in the toy model using matrix notation. Each element of the warp map matrix $\mathbf{w}(t)$ is a vector $\vec{\alpha}_{a,b}(t)$ containing the tip and tilt coefficients measured in subfield a for subaperture b. The matrix is organized for each subfield a by matrix row and each subfield b by each matrix column for the measurement at time t:

$$\mathbf{w}(t) = \begin{bmatrix} \vec{\alpha}_{1,1}(t) & \vec{\alpha}_{1,2}(t) & \vec{\alpha}_{1,3}(t) \\ \vec{\alpha}_{2,1}(t) & \vec{\alpha}_{2,2}(t) & \vec{\alpha}_{2,3}(t) \end{bmatrix}.$$

2.2.2 Measuring phase patch distance

In the toy model, a single patch of phase, ϕ_p, with tip and tilt $\vec{\alpha}_p$, is located out in front of the system at a distance h. Consider the geometry at time $t = 0$ given by the top left panel of Fig. (2.3). In this

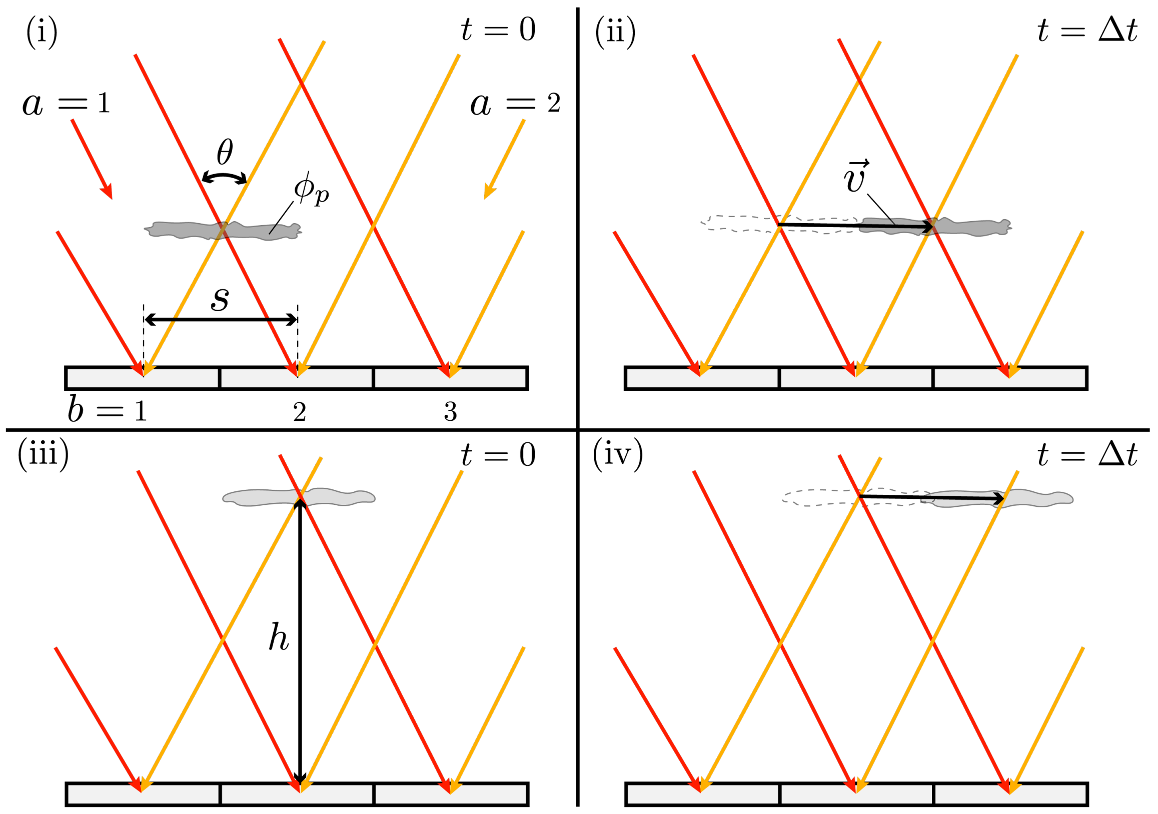


FIGURE 2.3: Toy model of SLODAR measurement geometry. The subfields are labeled by index a and subapertures are labeled by index b. The patch of phase ϕ_p is at a distance h and has velocity $\vec{v}$. Each subaperture is separated by distance s and the object separation is θ. (i): The patch of phase is seen by light from subfield-subaperture pair $(a,b;a'b') = (1,2;2,1)$. (ii): System (i) after time Δt has passed. The patch of turbulence has translated by $\vec{v}\Delta t$ and is now seen simultaneously by subfield-subaperture pair $(a,b;a'b') = (1,3;2,2)$. (iii): A single patch of phase at a further distance than in (i), with the patch being shared by $(a,b;a'b') = (1,3;2,1)$. (iv): System (iii) after time Δt has passed.

measurement the patch of phase only interacts with light from subfield $a = 1$ seen by subaperture $b = 2$, and from subfield $a = 2$ seen by subaperture $b = 1$. Assuming no other sources of tip/tilt in the space between the object field and the entrance pupil, we can write the SHWFS measurement matrix as

$$\mathbf{w}(0) = \begin{bmatrix} 0 & \vec{\alpha}_p & 0 \\ \vec{\alpha}_p & 0 & 0 \end{bmatrix}.$$

By taking the dot product of each tip/tilt vector in the above matrix with every other vector, we will find that every permutation is zero except for

$$\vec{\alpha}_{1,2} \cdot \vec{\alpha}_{2,1} = |\vec{\alpha}_p|^2.$$

This is because the two measurements are the same and will therefore have perfect correlation. To determine the distance to the patch of turbulence, we will take advantage of the fact that perfect correlation is occurring for measurement pairs $(a,b;a'b') = (1,2;2,1)$. Subapertures 2 and 1 are separated in the pupil by a distance s and subfields 1 and 2 are angularly separated by θ, which corresponds to the triangulated distance

$$h = \frac{s}{\theta} \tag{2.9}$$

as long as θ satisfies the small angle approximation.

Now consider the measurement geometry in the bottom left panel of Fig. (2.3). While the angular and spatial sampling of the SLODAR system remains unchanged, the patch of phase is now located at a higher altitude corresponding to the intersection from the subfield-subaperture pairs $(a,b;a'b') = (1,3;2,1)$. The resulting measurement matrix is

$$\mathbf{w}(0) = \begin{bmatrix} 0 & 0 & \vec{\alpha}_p \\ \vec{\alpha}_p & 0 & 0 \end{bmatrix}.$$

When correlating these measured subfield-subaperture slopes, the only non-zero dot product will be

$$\vec{\alpha}_{1,3} \cdot \vec{\alpha}_{2,1} = |\vec{\alpha}_p|^2.$$

Since the spatial separation from subaperture 3 to subaperture 1 is $2s$, the triangulated patch of phase is at a distance

$$h = \frac{2s}{\theta}. \tag{2.10}$$

2.2.3 Generalized turbulent layer ranging

Extrapolating this concept to A subfields and B subapertures, we can develop a measurement scheme for finding layers of turbulence at specific altitudes based on a designed SHWFS. By correlating the tip and tilt coefficients for every combination of $\vec{\alpha}_{a,b}$ and $\vec{\alpha}_{a',b'}$, dot products with high correlation will correspond to spatial sample vector $\Delta\vec{s}_{b,b'}$ and angular sample vector $\Delta\vec{\theta}_{a,a'}$ which triangulate a layer of turbulence present at a range h_ℓ:

$$h_\ell = \frac{|\Delta\vec{s}_{b,b'}|}{|\Delta\vec{\theta}_{a,a'}|}. \tag{2.11}$$

To keep track of the subfield and subaperture separation vectors for each dot product, an organized correlation scheme must be employed. Doing so requires simplifying the arbitrary 3D geometry of a system with A subfields and B subapertures into an orderly measurement grid. While we reserve the details of the correlation method for Section (5.2.1), here we present a measurement grid scheme and discuss its implications on altitude sampling.

The typical construction of a SHWFS lenslet array is to fill the pupil with uniform square subapertures, each with center-to-center separation equal to the subaperture side length s. To simplify the discussion going forward, we will assume that the object field is also a uniform seamless grid of squares with angular separation between adjacent fields of $\theta \leq \theta_0$. This will ensure that the angular sampling is sufficient to compensate for anisoplanatism. The subfield grid is made of rows and columns which are parallel to the rows and columns of the subaperture grid, respectively.

For a system with A subfields in a uniform square grid there are $\sqrt{A}$ subfields along each direction. Likewise for the system pupil, there are $\sqrt{B}$ subapertures along each direction in the pupil. We count the subfields and subapertures starting at the top left corner, with $a = 1$ and $b = 1$, and increase the corresponding index with increasing column number. At the end of the first row – i.e. $a = \sqrt{A}$ or $b = \sqrt{B}$ – we move back to the first column and down one row and continue counting up. This counting method is illustrated in the generalized system geometry in Fig. (2.4).

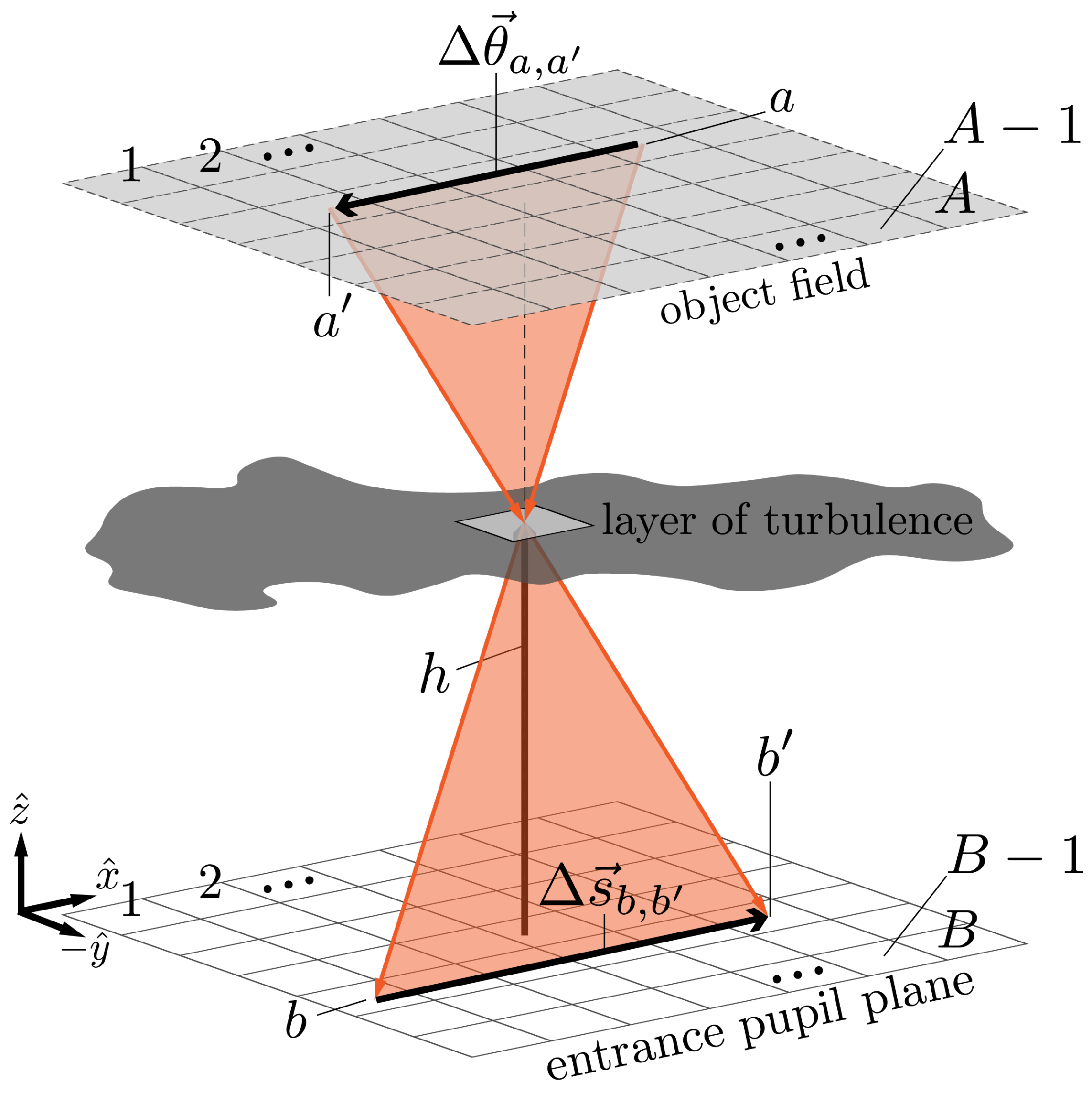


FIGURE 2.4: 3D geometry of a 2D grid of subapertures viewing a 2D grid of subfields with the subfield-subaperture pair $(\Delta\vec{\theta}_{a,a'}\Delta\vec{s}_{b,b'})$ illustrated. The subfield indices $a \in [1,A]$ count up along each row starting at the $(-x,+y)$ corner of the field. The subapertures follow an identical indexing for $b \in [1,B]$. The shown $\Delta\vec{\theta}_{a,a'}$ and $\Delta\vec{s}_{b,b'}$ are chosen to be anti-parallel to each other and parallel to the subaperture and subfield rows. This measurement geometry forms a 2D plane in the 3D system which will follow the layer ranging properties of the toy model.

We will also narrow our discussion of $\vec{\alpha}_{a,b}$ and $\vec{\alpha}_{a',b'}$ to data from subfield-subaperture combinations that have anti-parallel $\Delta\vec{\theta}_{a,a'}$ and $\Delta\vec{s}_{b,b'}$ vectors that are both either along the row or column direction. Looking at Fig. (2.4) we can see that with anti-parallel vectors $\Delta\vec{\theta}_{a,a'}$ and $\Delta\vec{s}_{b,b'}$, a 2D plane is formed in 3D space which will follow the geometric relationships developed by the toy model. Thus, by only considering these subfield-subaperture combinations the general 3D measurement geometry can be treated as a collection of 2D problems.

Since A subfields will contain $\sqrt{A}$ subfields along each direction, the maximum angular separation for any measurement using the proposed method is

$$\theta_{\mathrm{M}} = (\sqrt{A}-1)\theta. \tag{2.12}$$

Similarly, with $\sqrt{B}$ subapertures along each direction, the maximum spatial separation sampled is

$$s_{\mathrm{M}} = (\sqrt{B}-1)s. \tag{2.13}$$

We can use these relationships to write the minimum triangulated measurement distance for the system using the principles which resulted in Eq. (2.9):

$$h_{\mathrm{m}} = \frac{s}{\theta_{\mathrm{M}}}. \tag{2.14}$$

Note that due to the relationship between θ and h, the minimum measured distance corresponds to the widest angular separation sampled by the system. From h_{m}, the object separation provides uniform sampling in steps of h_{m} for each additional subaperture separation up to the maximum altitude

$$H_{\mathrm{m}} = \frac{(\sqrt{B}-1)s}{\theta_{\mathrm{M}}} = \frac{s_{\mathrm{M}}}{\theta_{\mathrm{M}}}. \tag{2.15}$$

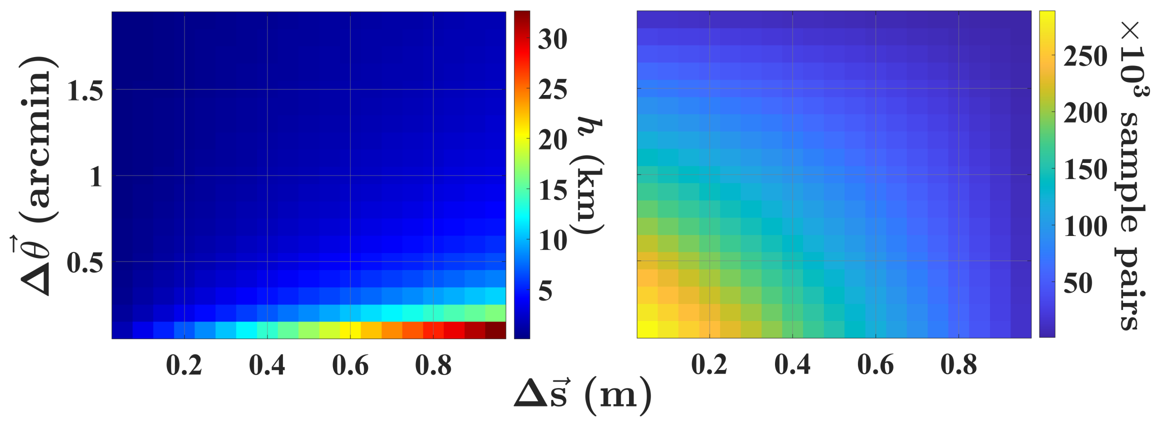


FIGURE 2.5: Visualizations of the non-uniformity in measured distance and sample averaging for SLODAR systems. This example uses $\sqrt{A} = 20$ over a $2'$ field of view and $\sqrt{B} = 20$ with $s = 0.05$ m. Left: distribution of measured h for each combination of $(\Delta\vec{s}, \Delta\vec{\theta})$. Right: number of samples available to average for each $(\Delta\vec{s}, \Delta\vec{\theta})$ given the anti-parallel subfield-subaperture requirement.

This altitude sampling pattern repeats itself with increasing distance for decreasing object separation $\left[(\sqrt{A}-2)\theta, (\sqrt{A}-3)\theta, \ldots\right]$ until the maximum sampling altitude of the entire system is reached:

$$H_{\mathrm{M}} = \frac{s_{\mathrm{M}}}{\theta}. \tag{2.16}$$

The above sampling relationships result in the non-uniform distance sampling which is inherent to SLODAR. Sampling is denser closer to the system aperture and becomes increasingly more separated as the angular separation of the objects decreases. An example of non-uniform SLODAR height sampling is shown in the left plot in Fig. (2.5).

SLODAR measurements also sample each distance a different number of times, producing non-uniform averaging. For the minimum subaperture separation, s, and the minimum subfield separation, θ, there are

$$N_{s,\theta} = 2\left[\sqrt{B}(\sqrt{B}-1)\right]\left[\sqrt{A}(\sqrt{A}-1)\right] \tag{2.17}$$

combinations of the vectors $\vec{\alpha}_{a,b}$ and $\vec{\alpha}_{a',b'}$ in each SHWFS measurement frame which meet the toy model approximation condition. The factor of 2 is due to the fact that s and θ can be either horizontal or vertical separations in the subaperture and subfield grid. At the maximum subfield separation and minimum subaperture separation there are

$$N_{s,\theta_M} = 2\left[\sqrt{B}(\sqrt{B}-1)\right]\sqrt{A} \tag{2.18}$$

combinations. Likewise, for the maximum subaperture separation and minimum subfield separation there are

$$N_{s_M,\theta} = 2\left[\sqrt{A}(\sqrt{A}-1)\right]\sqrt{B} \tag{2.19}$$

combinations. Since A and B are positive integers, we can be certain that $N_{s_M,\theta} < N_{s,\theta}$ and $N_{s,\theta_M} < N_{s,\theta}$. It is also known that the least sampled combination is for the maximum subaperture and subfield sample:

$$N_{s_M,\theta_M} = 2\sqrt{A}\sqrt{B}. \tag{2.20}$$

Because of the described sampling relationships, distances which are measured by a small number of subaperture or subfield separations are measured more times than wide subaperture and subfield separations. The implication is that distances measured by combinations of small subfield and subaperture separations will be more highly averaged and therefore less noisy. The plot on the right side of Fig. (2.5) visualizes the distribution of available samples for each $(\Delta\vec{s}, \Delta\vec{\theta})$.

2.2.4 Measuring phase patch velocity

Returning to the toy model, the patch of phase ϕ_p has a velocity $\vec{v}$. To measure the patch velocity, consider the top row of measurement geometries in Fig. (2.3). The left panel is at time $t = 0$ while the right panel is the same system at time $t = \Delta t$. Since the patch has velocity $\vec{v}$, we know that the patch will move by the vector

$$\Delta\vec{x} = \vec{v}\Delta t. \tag{2.21}$$

Δx is specifically the magnitude and direction such that the phase is now simultaneously seen by the subfield-subaperture pair $(a,b;a'b') = (1,3;2,2)$. The two measurement matrices for this system are

$$\mathbf{w}(0) = \begin{bmatrix} 0 & \vec{\alpha}_p & 0 \\ \vec{\alpha}_p & 0 & 0 \end{bmatrix}, \quad \mathbf{w}(\Delta t) = \begin{bmatrix} 0 & 0 & \vec{\alpha}_p \\ 0 & \vec{\alpha}_p & 0 \end{bmatrix}.$$

We can see that the signal of the patch in the measurement matrix at time $t = \Delta t$ is shifted by one subaperture from the time $t = 0$. Since the subaperture and subfield separation indicating the distance to the patch is unchanged, we can reason that the patch has moved by $|\Delta\vec{x}| = s$. Using this and by rearranging Eq.(2.21), we can estimate the patch speed as

$$|\vec{v}| = \frac{s}{\Delta t}.$$

The direction of the velocity vector can then be extracted using the direction from subaperture 1 to subaperture 2, and reinforced by the direction from subaperture 2 to subaperture 3.

Applying the same analysis to the higher altitude patch of phase shown in the bottom row of Fig. (2.3), we have the two measurement matrices

$$\mathbf{w}(0) = \begin{bmatrix} 0 & 0 & \vec{\alpha}_p \\ \vec{\alpha}_p & 0 & 0 \end{bmatrix}, \quad \mathbf{w}(\Delta t) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \vec{\alpha}_p & 0 \end{bmatrix}.$$

Since the layer is at the maximum altitude which the system can measure – i.e. maximum subaperture separation and minimum object separation – there is only one possible measurement where two subapertures see the patch at the same time. At the time $t = \Delta t$, the patch is only seen by subaperture 2 along subfield 2. However, we can still see that the $t = \Delta t$ measurement is the $t = 0$ measurement matrix shifted by one subaperture, meaning we could still estimate that the patch has moved by the distance s in the direction from subaperture 1 to subaperture 2 as was the case for the lower altitude patch. With fewer subfield-subaperture combinations marking this shift, the signal used to calculate the velocity will have less averaging than the low altitude patch. From this, we can expect that velocity measurements from distant layers will be noisier than near layers.

2.2.5 Generalized turbulent layer velocity estimation

Using the generalized A subfields and B subapertures under the same geometric assumptions used for generalized layer ranging, we can construct a scheme for determining the velocity of layers of turbulence from SHWFS data. By adding time information to the data the problem is 3D rather than 2D, as was the case for layer ranging, and is therefore more complicated to organize and parse. While details of the technique used in this work for interpreting velocities from SHWFS data will be presented in Chapter 5, there are some basic relationships related a generalized velocity measurement scheme from SLODAR geometry which are relevant to the toy model.

Measuring layer velocity requires locating the same patch of phase in different subapertures at different times for specific subfield separations. The velocity is then found based on the number of subapertures it has drifted across. This means that there is a maximum layer velocity which can be measured for a given SHWFS exposure time. Namely, if the time between measurement frames is δt, then the maximum measurement velocity for the system is

$$\vec{v}_{\max} = \frac{(B-1)s}{\delta t}. \tag{2.22}$$

Considering wind speeds anywhere in the atmosphere with significant air volume to generate optical aberrations will not exceed 50 m/s (Hardy 1998) and exposure times need to meet the condition $\delta t \leq \tau_0 \sim 10$ ms, this should not be a problem for systems with an entrance pupil of $D \geq 0.5$ m. However, systems smaller than this may be limited in finding the velocity of high speed layers in the atmosphere.

Unlike the maximum measurement velocity of a layer, the minimum measurable velocity can be calibrated by increasing the time delay until the condition

$$\Delta t = \frac{s}{|\vec{v}_{\min}|} \tag{2.23}$$

is met. We can assume that turbulence will not be occurring for very slow wind speeds since turbulence requires a high Reynolds number which is proportional to wind speed. This means we can reasonably expect that even unusually slow layers ($|\vec{v}_{\min}| \sim 2$ m/s) will be detectable within ~ 50 ms if we assume that subapertures will not be larger than the Fried length of strong turbulence ($r_0 \sim 0.1$ m).

2.3 Other layer profiling techniques

There are three other common techniques for extracting information about the location, dynamics, and physical properties from a layer of turbulence. While this work exclusively uses SLODAR for extracting layer telemetry from data, the other three are summarized here for completeness.

Differential image motion monitoring (DIMM) is a measurement technique which uses an array of small telescopes to extract seeing information from a single star image (Sarazin and Roddier 1990). The measurement process is similar to SLODAR in that the separation of the DIMM telescopes provides spatial sampling, but this technique does not use angular sampling and therefore cannot extract layer ranges or layer velocity. To the benefit of SLODAR, the seeing estimation process in DIMM can be used with the subapertures in a SHWFS.

Since DIMM cannot be used to estimate layer ranges and only needs a single object, it is often paired with another single object turbulence telemetry system called a multi-aperture scintillation sensor (MASS) (Kornilov et al. 2003). MASS uses single star images formed by multiple different sized entrance pupil telescopes at the same location in space. This is usually done using a wheel of different pupil sizes on a single telescope. The pupil plane for each aperture size is collected, and the scintillation index in each pupil image is investigated to estimate the distance to layers of turbulence (Tokovinin and Kornilov 2002). While MASS and DIMM together can do most of the same job as a SLODAR system with only a single object, implementing both MASS and DIMM requires a collection of multiple telescopes independent from the science imager. The additional hardware requirements generally makes MASS+DIMM a turbulence profiling technique only available to large astronomical facilities.

The most similar technique to SLODAR is SCIntillation Detection And Ranging (SCIDAR). Instead of correlating pupil slopes, SCIDAR correlates images of pupil scintillation at different conjugate planes from multiple objects and extracts the distances to layers from the result (Rocca et al. 1974). To the benefit of SCIDAR, scintillation effects increase with increased propagation distance which makes it a good technique for finding layers of turbulence tens of kilometers away (Osborn et al. 2018). The downside is that SCIDAR does not natively detect turbulence close to the pupil. It is possible to add additional optics to increase the optical distance to layers near to the ground, thereby increasing the scintillation signal of the layers, but this comes with increasing hardware complexity and facility size requirements (Fuchs et al. 1998).

There are several astronomical sites around the world that implement one or more of these techniques to obtain turbulence profile data – most notably the ESO observatory in Paranal, Chile (Osborn et al. 2018, Lombardi et al. 2008). Each technique has its own domains of operation which allow it to excel for measuring turbulence at specific distances from the system. Within the overlap regions of each technique, performance has been shown to be in good agreement. This means that whichever technique best serves the engineering and science requirements of a particular system is a fair choice for extracting turbulence telemetry data.

Chapter 3 Simulation environment

To develop turbulence profiling methods for MOIC, a realistic simulation of a SHWFS looking through the atmosphere is needed to generate synthetic data. The turbulence must be statistically valid and the process of generating the layers must be time and memory efficient. Once layers of turbulence are properly modeled, the anisoplanatic phase has to be determined for each subfield-subaperture combination. The resulting tip/tilt vectors $\vec{\alpha}_{a,b}(t)$ have to then be calculated, sorted and stored in the organized warp map matrices, $\mathbf{w}$. The layers must also be generated in such a way that the anisoplanatic phase in the system pupil can be found and used to simulate images which have been blurred by the multi-layer atmosphere.

Throughout this chapter both function notation and matrix notation are used. Functions are written as depending on space and time variables. Matrices are written in bold text and are only written as either a function of time, $\mathbf{M}(t)$, or not as a function of any specified variable. For any matrix, the value stored in a specific cell corresponding to the indices (a,b,ℓ) is called using the notation

$$\mathbf{M}(a,b,\ell). \tag{3.1}$$

If a continuous region of cells are to be sampled from a matrix at once, the notation

$$\mathbf{M}([\text{row}_{\text{start}} : \text{row}_{\text{end}}], [\text{column}_{\text{start}} : \text{column}_{\text{end}}], \text{etc...}) \tag{3.2}$$

is used. Since the simulation environment here was developed in MATLAB, we will use the same row-column indexing as MATLAb – i.e. indexing starts at $(1,1)$ at the top left corner of the matrix.

For matrices where the row-column fields represent a grid in 2D space, rows define samples along the y axis and columns define samples along the x axis. We choose to define the xy plane such that y is increasingly negative for increasing row value and x is increasingly positive for increasing column value. The matrices are implicitly over the xy plane so they are not written as being a function of specific indices. To sample these matrices over a specific region in the xy plane at the time t, we use the notation

$$\mathbf{M}([y_{\text{start}} : y_{\text{end}}], [x_{\text{start}} : x_{\text{end}}], t). \tag{3.3}$$

In the case where the sample grid does not contain the exact coordinates which are sampled, we assume that the matrix cells corresponding to the nearest neighbor are used. Since this simulation environment was developed in MATLAB, where matrices are the native format, discussions which begin in functional notation will be moved to matrix format at the earliest convenience to better state exactly how these calculations are performed while modeling.

3.1 Turbulence simulator

The turbulence simulator adopts a near-field thin sheet of phase model (Roggemann and Welsh 1996). We also assume that the layers of turbulence satisfy the frozen flow hypothesis (FFH). The FFH postulates that the turbulence structure is not changing as it moves. This is a good approximation for time scales of 10 ms, which is on the order of typical coherence times (Schöck and Spillar 2000). This allows us to model volumetric turbulence in the atmosphere as a series of translating 2D matrices of phase. Each pixel in the layer matrix designates a phase value accumulated by light rays which interact with the matrix at that coordinate.

The simplest way to model dynamic turbulence is to generate the entire region of interest for each layer, and then move the optical pupil along each matrix following the layer wind direction. However, generating layers of turbulence in a computer is memory intensive. This issue is compounded when multiple layers need to be considered simultaneously. A simulation which models 10s or 100s of seconds of data collection, which contains turbulence with wind speeds > 10 m/s, requires layers which are > 100 m along one dimension. Picking even large pixel scales for systems with a sub-arcminute field of view quickly results in multiple matrices which are $> 10,000 \times 10,000$ elements. Additionally, this problem scales with field of view, layer distance, and entrance pupil diameter, making it challenging to generate each layer entirely when initializing a wide field image simulation.

For example, consider a system with a field of view of 5', an entrance pupil diameter 1 m, and a layer of turbulence at 20 km with relatively large pixel scale 0.01 m. At the layer, the pupil has projected to an approximate width of 30 m. Assuming an average wind speed of 10 m/s at that altitude (Hardy 1998), the layer matrix could exceed $10,000 \times 10,000$ pixels for a 10 second simulation, and $100,000 \times 100,000$ pixels for a 1.5 minute simulation. While a standard desktop may be able to handle a few double precision number $10,000^2$ pixel matrices, that is not the case for $100,000^2$ pixel matrices with current technology.

Therefore it is important that any turbulence engine built for wide field image correction analysis is capable of generating any small patch from within a continuous layer of turbulence. This allows us to only generate the patch within each layer of turbulence corresponding to where the system pupil projects to. By going with this more memory efficient method it is also important that the chosen method is capable of generating the desired patch quickly because a new layer matrix has to be calculated at each simulation time step.

3.1.1 Generalized plane wave representation for turbulent phase

To generate any part of an entire layer of turbulence, we can write a complex plane wave form of the phase at a point $\vec{r}$ at time t:

$$\begin{aligned}\phi_\ell(\vec{r},t) &= \sum_n^N \tilde{\phi}_n(\vec{r},t) \\ &= \sum_n^N A_n e^{i(\vec{k}_n\cdot\vec{r}+\Psi_n(t,\theta_n))}.\end{aligned} \tag{3.4}$$

where the quantities

$$\vec{k}_n \cdot \vec{r} = k_n\left[\cos(\theta_n)x + \sin(\theta_n)y\right], \tag{3.5}$$

$$\Psi_n(t,\theta_n) = k_n t|\vec{v}_\ell|\cos\left(\tan^{-1}\left(\frac{v_y}{v_x}\right) - \theta_n\right) \tag{3.6}$$

$$= k_n t|\vec{v}_\ell|\cos(\psi_n). \tag{3.7}$$

This allows us to express a particular layer of phase, indexed ℓ, as a superposition of wave components with individual wave vectors $|\vec{k}_n| = k_n$, corresponding complex amplitudes A_n, and component orientations in the layer, θ_n. The layer is defined over all space and follows the FFH with layer velocity $\vec{v}_\ell$. Since the turbulence is itself a random process, k_n, A_n, and θ_n must all be random variables with distributions which match those of a particular turbulence model. If these distributions are properly represented, then a power law structure function of the random process $\phi_\ell(\vec{r},t)$ will satisfy the condition

$$D_{\phi_\ell}(r) = C\left(\frac{r}{r_0}\right)^p = \left\langle[\phi_\ell(\vec{r},t) - \phi_\ell(\vec{r}+\delta\vec{r},t)]^2\right\rangle \tag{3.8}$$

where $|\delta\vec{r}| = r$ and the angle brackets indicate the expectation value for each random variable as well as the ensemble time average.

To understand the distributions of the random variables that form each wave component, $\tilde{\phi}_n(\vec{r},t)$, we need to state assumptions for turbulence which will be valid for all models used in this work. These assumptions were originally postulated by Andrey Kolmogorov and are the foundational statistical description for turbulent flow. For any region of fully developed turbulence (Kolmogorov 1941):

1. the random process occurs for length scales within the inertial range $l \in [l_0, L_0]$,
2. turbulent flow forms swirling eddies that are self similar on all scales in the inertial range,
3. the statistics of the process and its variables are homogeneous, isotropic, and stationary in increments.

Assumptions 1 and 2 allow us to assume that the perturbations which make up the turbulent flow are so numerous that the complex amplitudes of each component should follow a complex Gaussian distribution. This constrains the moments of the amplitudes to

$$\langle A_n \rangle = 0 \qquad \langle A_n A^*_{n'} \rangle = \left\langle |A_n|^2 \right\rangle \delta_{n,n'}. \tag{3.9}$$

Since the real and imaginary parts of the complex amplitude are independent and identically distributed, either part can be used once the layer is formed.

Assumption 3 simplifies the dimensionality of the problem since an isotropic and homogeneous random process is invariant under rotation. For this to be true the distribution of wave component angles must be

$$\theta_n \sim U(-\pi, \pi). \tag{3.10}$$

Further applying assumption 3 to a zero mean Gaussian distributed random process, it can be shown that the structure function will be related to the 2D power spectrum, $S_{\phi_\ell}(\vec{k})$, by (Tatarski 1961)

$$D_{\phi_\ell}(r) = 2 \int\limits_{-\infty}^{\infty} S_{\phi_\ell}(\vec{k})[1 - \cos(\vec{k} \cdot \delta\vec{r})]\, d\vec{k}. \tag{3.11}$$

We will not be able to model the continuous structure function and power spectrum on a computer when we generate layers – that would require infinite sampling of the wave vector space. Instead we seek to satisfy

$$D_n(r) = 2 \iint\limits_{\Omega} S_{\phi_\ell}(\vec{k}_n)[1 - \cos(\vec{k}_n \cdot \delta\vec{r})]\, d\vec{k}_n \tag{3.12}$$

where Ω is a region around $\vec{k}_n$ with approximately constant sampled power spectrum $S_{\phi_\ell}(\vec{k}_n)$. As $L \to \infty$, the region Ω becomes differentially small and the sum of all $D_n(r)$ will converge to $D_{\phi_\ell}(r)$ point-wise. The goal is to determine what sampling of $\vec{k}_n$ and the corresponding A_n values which form each $\tilde{\phi}_n(\vec{r}, t)$ will yield sufficient convergence in the structure function for a reasonable value of L when simulated on a computer.

To sample the wave vector space efficiently, we use Charnotskii's non-overlapping sparse uniform partitioning (Charnotskii 2020). Sparse uniform sampling is based on assumption 2 – that the eddies are self similar on all scales in the inertial range. This is analogous to the length scales, and therefore the wave vectors, being distributed uniformly in log-space. The log-uniform partition sampled wave vectors are

$$K_n = K_{\min}\exp\left[\frac{n}{N}\ln\left(\frac{K_{\max}}{K_{\min}}\right)\right] \tag{3.13}$$

where

$$K_{\min} = \kappa_{\min}\frac{2\pi}{L_0} \qquad K_{\max} = \kappa_{\max}\frac{2\pi}{l_0}. \tag{3.14}$$

$\kappa_{\min}$ and $\kappa_{\max}$ are scaling factors used to tune the behavior of the turbulence to better match the desired structure functions. Each partition sample wave vector K_n is related to the wave vector in the decomposition through the geometric relationship

$$k_n = \sqrt{K_{n-1}^2 + \xi_n\left(K_n^2 - K_{n-1}^2\right)}, \tag{3.15}$$

where $\xi_n \sim U(0,1)$ random variables. This introduces uniform probability into where each k_n is in each partition, thereby limiting any bias from sampling over a log-uniform wave vector space.

With a defined partition, we guess the form of the amplitudes to be

$$A_n = (\beta_n + i\gamma_n)\sqrt{CS_{\phi_\ell}(k_n)\Omega(K_n)}. \tag{3.16}$$

$\Omega(K_n)$ is the area of the n^{th} partition and C is the structure function power law scaling constant in Eq. (3.8). β_n and γ_n are independent zero mean Gaussian random numbers with a variance of 1 to ensure Eq. (3.9) is satisfied. Since each partition is a ring from radius K_{n-1} to radius K_n, the area of each partition is

$$\Omega(K_n) = \pi(K_n^2 - K_{n-1}^2). \tag{3.17}$$

Now we can check if the chosen wave vector sampling and amplitudes satisfy the partitioned structure function. Plugging $\tilde{\phi}_n(\vec{r},t)$ into the definition of the structure function:

$$\begin{aligned} D_n(r) &= \left\langle \left[\tilde{\phi}_n(\vec{r},t) - \tilde{\phi}_n(\vec{r}+\delta\vec{r},t)\right]^2 \right\rangle_{\{A_n,\vec{k}_n,\theta_n\}} \\ &= \left\langle \left[A_n e^{i\Psi_n(t,\theta_n)} \left(e^{i\vec{k}_n\cdot\vec{r}} - e^{i\vec{k}_n\cdot(\vec{r}+\delta\vec{r})}\right)\right] \left[A_n^* e^{-i\Psi_n(\vec{v}_\ell,t)} \left(e^{-i\vec{k}_n\cdot\vec{r}} - e^{-i\vec{k}_n\cdot(\vec{r}+\delta\vec{r})}\right)\right] \right\rangle_{\{A_n,\vec{k}_n,\theta_n\}} \\ &= \left\langle \langle A_n A_n^* \rangle \left| e^{i\vec{k}_n\cdot\vec{r}} - e^{i\vec{k}_n\cdot(\vec{r}+\delta\vec{r})} \right|^2 \right\rangle_{\{\vec{k}_n\}} . \end{aligned} \tag{3.18}$$

To arrive at the desired relationship between the remaining random variables in Eq. (3.18) and the power spectrum we use the identity

$$\left| e^{i\vec{k}_n\cdot\vec{r}} - e^{i\vec{k}_n\cdot(\vec{r}+\delta\vec{r})} \right|^2 = 2\left[1 - \cos(\vec{k}_n\cdot\delta\vec{r})\right].$$

With this identity and by using the chosen form of A_n from Eq. (3.16), the partition structure function can be shown to reduce to the form (Charnotskii 2020)

$$\begin{aligned} D_n(r) &= 2\left\langle \Omega(K_n) \left\langle |\beta_n + i\gamma_n|^2 \right\rangle_{\{\beta_n,\gamma_n\}} S_{\phi_\ell}(\vec{k}_n) \left[1 - \cos(\vec{k}_n\cdot\delta\vec{r})\right] \right\rangle_{\{\vec{k}_n\}} \\ &= 2 \iint_\Omega S_{\phi_\ell}(\vec{k}_n)[1 - \cos(\vec{k}_n\cdot\delta\vec{r})]\, d\vec{k}_n \end{aligned} \tag{3.19}$$

which verifies our choices for k_n and A_n. Now all that is needed to simulate layers with Eq. (3.4) is a functional form for the power spectrum. Once the power spectrum is specified, we can use a Monte Carlo simulation to verify the structure function of the simulated layers.

3.1.2 Kolmogorov turbulence simulation and verification

The simplest form of turbulence is the Kolmogorov model. This model assumes that the only regime of interest is that of fully developed turbulence. To ensure this is the case, a vanishing inner scale and infinite outer scale can be used to force the inertial range over all scales. Under this assumption the generalized 2D power spectrum of Kolmogorov turbulence is given by (Charnotskii 2013)

$$S^{(\mathrm{kol})}(k,p) = B(p) r_0^{-p} k^{-2-p}, \tag{3.20}$$

where r_0 is the correlation length of the process and

$$B(p) = \frac{p 2^{p-2} \Gamma(1+p/2)}{\pi \Gamma(1-p/2)}. \tag{3.21}$$

For phase in atmospheric turbulence we are interested in the case where $p = 5/3$ (Fried 1965):

$$S^{(\mathrm{kol})}(k) = 3.3028 r_0^{-5/3} k^{-11/3}. \tag{3.22}$$

While the power spectrum for Kolmogorov turbulence is derived assuming $l_0 \approx 0$ and $L_0 \to \infty$, care has to be taken when modeling these regimes on a computer. The partition wave vectors in Eq. (3.14) cannot accept $l_0 = 0$ since it will result in division by zero. Instead, we have found that using a simulated layer pixel scale $\Delta x > l_0$ is an effective method for mitigating the effects of nonzero inner scale. Additionally, using large outer scales which exceed the spatial region over which the layer will be generated – e.g. $L_0 \geq 100{,}000$ m – is a sufficient approximation in most cases to $L_0 = \infty$.

Applying these concepts, two different r_0 Kolmogorov layers were simulated over a 30 m^2 region at three different time steps. The results are shown in Fig. (3.1), where we can visually observe the phase strength scaling for different r_0 values as well as the exact prescribed frozen flow velocities of the layers. The real parts of the amplitudes were used for these results. Additionally, the shown plots have the mean value across all time steps of each layer subtracted so that the layers have zero phase piston. This makes it easier to compare the strength of the aberrations between $r_0 = 0.1$ m and $r_0 = 1$ m.

Using a wave decomposition with $N = 750$ components, l_0 = 1 mm, $\kappa_{\max}$ = 2, L_0 = 100,000 m, $\kappa_{\min}$ = 1/10, Δx = $5 l_0$, and the scaled Kolmogorov power spectrum in Eq. (3.22), the partitioned

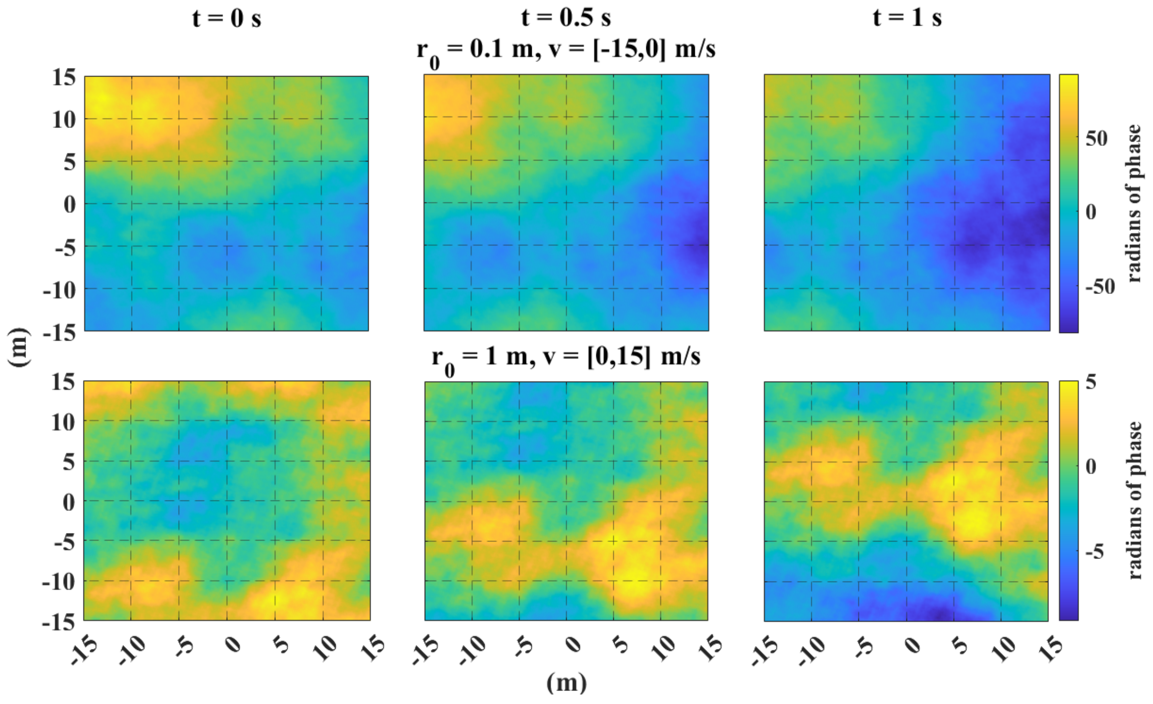


FIGURE 3.1: Simulation of two different layers of Kolmogorov turbulence at times $t = [0, 0.5, 1]$ s. The shown layers were generating using $N = 1,500$ elements, $L_0 = 100,000$ m, $l_0 = 0.001$ m, $\kappa_{\min} = 1/10$, $\kappa_{\max} = 2$, and pixel scale $\Delta x = 0.005$ m. The average layer generation time was 2.3 s using the matrix method described later in this chapter. The top row is for a layer with $r_0 = 0.1$ m and velocity only in $-x$, while the bottom row layer has $r_0 = 1$ m and a velocity which is only in $+y$. The velocity directions and speeds can be estimated visually by following distinct turbulence structures at each time step. The increased phase strength of $r_0 = 0.1$ m turbulence relative to the $r_0 = 1$ m turbulence can be observed by comparing the scale of color bars in the top and bottom rows.

wave vector space and amplitudes were calculated for different values of r_0. The structure function was then estimated directly using the right side of Eq. (3.8), and the process was averaged over 5,000 realizations consisting of both the real and imaginary parts of the wave decomposition. Each layer extended out to 1 km to allow for estimates of widely separated points in fully developed turbulence. The results are shown in Fig. (3.2), plotted alongside the theoretical structure function from the power law in Eq. (3.8) for $C = 6.88$ and $p = 5/3$. There is full agreement between the theoretical curve and the Monte Carlo results indicating that the wave decomposition is capable of generating proper Kolmogorov layers of turbulence.

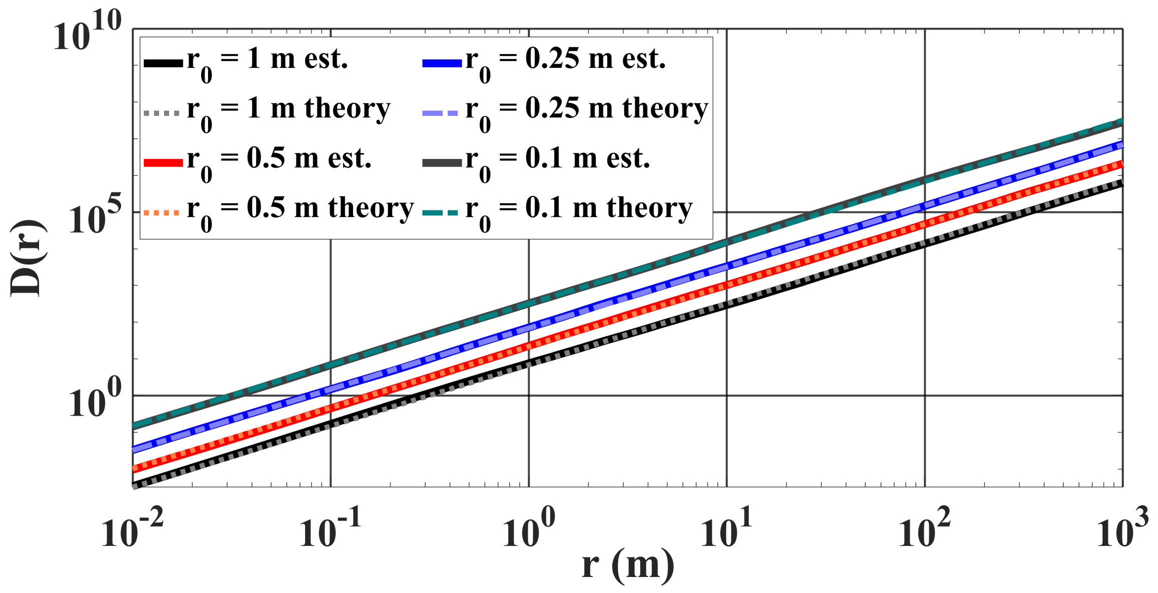


FIGURE 3.2: Results from the Monte Carlo simulation of Kolmogorov layers from the wave decomposition in Eq. (3.4) with $N = 750$ elements. Dashed lines indicate the ideal structure functions using the modeled layer r_0 and Eq. (3.8) which are in excellent agreement.

3.1.3 von Kármán turbulence simulation and verification

A more realistic model for turbulence uses the von Kármán spectrum. This spectrum operates under the same foundational assumptions as Kolmogorov, but it considers the effects of a finite outer scale and non-zero inner scale. With a finite inertial range of fully developed turbulence, the 2D power spectrum takes the form (Charnotskii 2013)

$$S^{(\mathrm{vK})}(k,p) = \frac{B(p)r_0^{-p}}{\left(k^2+k_0^2\right)^{1+p/2}} e^{-k^2/k_m^2}. \tag{3.23}$$

The dependence on the inner and outer scales in the von Kármán spectrum is from the coefficients

$$k_0 = \frac{2\pi}{L_0} \qquad k_m = \frac{2\pi}{l_0}. \tag{3.24}$$

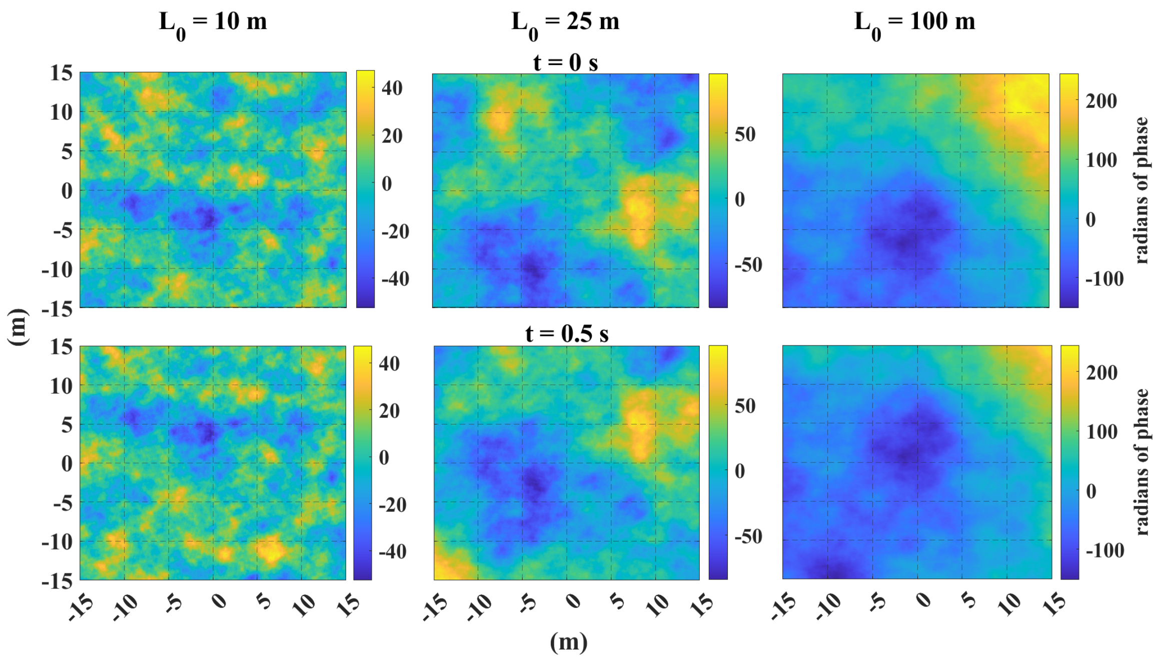


FIGURE 3.3: Simulation of three different layers of von Kármán turbulence at times $t = [0, 0.5]$ s. The shown layers were generating using $N = 1,500$ elements, $\kappa_{\text{min}} = 1/10$, $\kappa_{\text{max}} = 2$, $\Delta x = 0.005$ m, $l_0 = 0.001$ mm, $r_0 = 0.1$ m, and $\vec{v} = [0, 15]$ m/s. The average layer generation time was identical to the Kolmogorov layer generation. Layers with smaller outer scale are dominated by shorter length scale perturbations. At $L_0 = 100$ m, the turbulence looks similar to the Kolmogorov turbulence since the outer scale is much larger than the generated region.

Once again, for atmospheric turbulence we are interested in the case $p = 5/3$ which produces

$$S^{(\text{vK})}(k) = \frac{3.3028 r_0^{-5/3}}{\left(k^2 + k_0^2\right)^{11/6}} e^{-k^2/k_m^2}. \tag{3.25}$$

Three test cases for von Kármán turbulence were generated with three different outer scales and the same r_0. The results are in Fig. (3.3). The effect of changing outer scale is directly observable in the plots. The layers were set at zero phase piston, like in the Kolmogorov layer visualization, allowing us to observe how outer scale also effects the scale of the phase aberrations.

Notice that $L_0 \to \infty$ results in $k_0 \to 0$, which removes the dependence on the outer scale in the power spectrum. When modeling layers, if the maximum separation between two observed points satisfies $r << L_0$ then we should expect the turbulence to appear effectively Kolmogorov. As we approach observation of separation $r > L_0$ the points will become decorrelated resulting in

$$\begin{aligned}&\left\langle [\phi_\ell(\vec{r},t) - \phi_\ell(\vec{r}+\delta\vec{r},t)]^2 \right\rangle \\ &= \left\langle \; |\phi_\ell(\vec{r},t)|^2 \right\rangle + \left\langle |\phi_\ell(\vec{r}+\delta\vec{r},t)|^2 \right\rangle \\ &= 2\sigma^2_{\phi_\ell}.\end{aligned} \tag{3.26}$$

Similarly, when $l_0 \to 0$, $k_m \to \infty$ and the exponent containing the inner scale goes to 1. Therefore, to observe the exponential roll-off in the structure function due to the inner scale separations smaller than the inner scale have to be observed.

Due to the influence of the inner and outer scale on the von Kármán model, its structure function will not be a power law for all r. When observing point separations well inside the inertial range, von Kármán turbulence is fully developed and must appear to follow the power law of Kolmogorov turbulence. As the point separations approach the outer scale, the structure function must roll off to satisfy Eq. (3.26). On the other end, as point separations become smaller than the inner scale there should be an exponential drop off in the phase correlation.

The Monte Carlo validation for von Kármán layers used identical $N, \kappa_{\min}$, and $\kappa_{\max}$ values as the Kolmogorov model validation. Different outer scales of $L_0 = [10, 100, 500]$ m were tested with inner scales $l_0 = [0.001, 0.01, 0.01]$ m, respectively. To observe inner scale decay a pixel scale of $\Delta x = 0.001$ m was used. Each inner and outer scale combination was statistically estimated out to a maximum point separation of 200 m and averaged over 2,000 realization using both the real and imaginary parts of the wave decomposition. The validation was performed for Fried lengths $r_0 = [0.1, 1]$ m. The plots for each Fried length are shown in Fig. (3.4). Once again, there is full agreement in the inertial range to the ideal power law. We also observe the correct roll-off with saturation at $2\sigma^2_{\phi_\ell}$ as $r \gtrsim L_0$ and a decay in correlation for $r \lesssim l_0$.

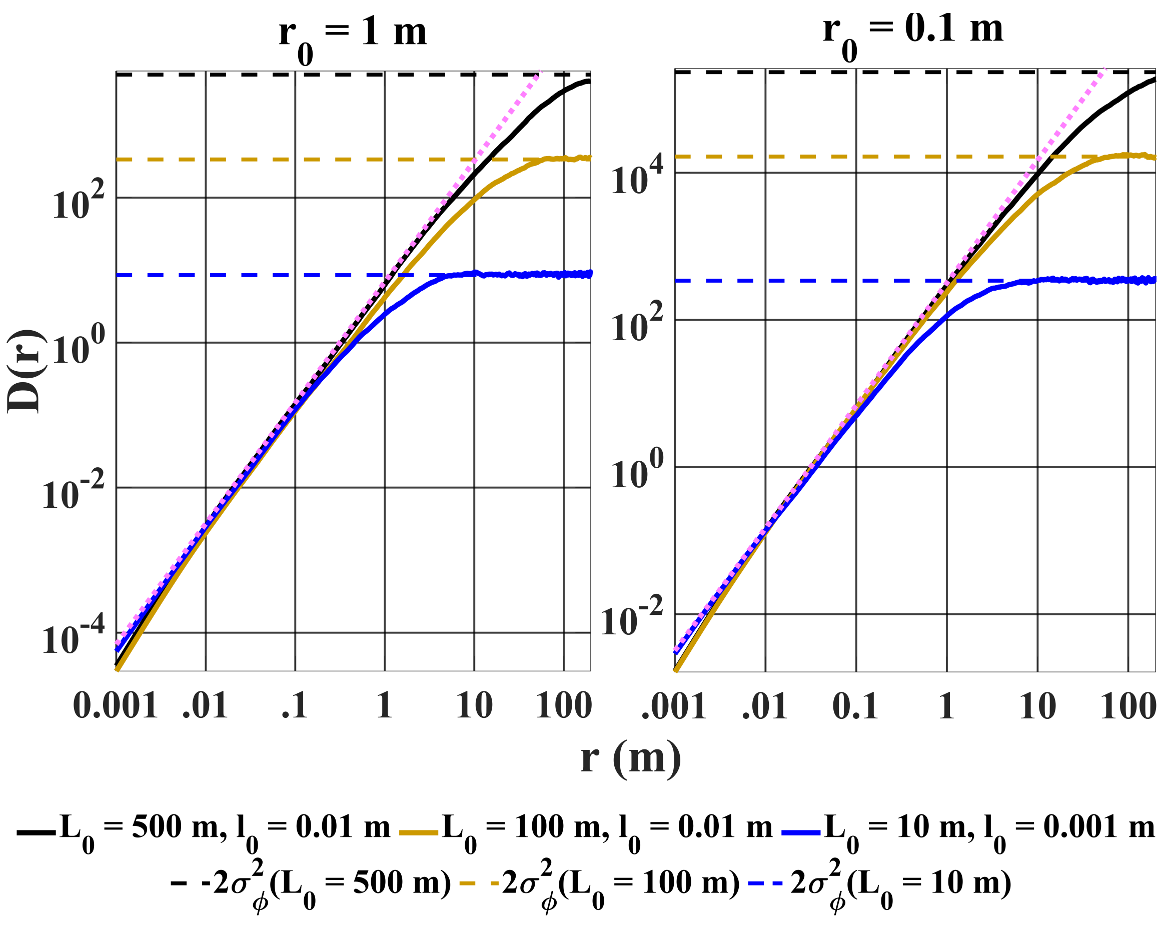


FIGURE 3.4: Results from the Monte Carlo simulation of von Kármán layers from the wave decomposition in Eq. (3.4) using $N = 750$ elements. Dashed horizontal lines indicate the saturation of twice the layer variance for uncorrelated points with separation larger than the outer scale. The sloped dashed line which is tangent to each curve indicates the ideal structure function for fully developed turbulence using the modeled layer r_0 and Eq. (3.8) for separation in the inertial range $l \in [l_0, L_0]$.

3.1.4 Matrix method for fast layer generation

When simulating a dynamic atmosphere many layers have to be repeatedly generated at different times to approximate the continuous evolution of turbulent phase. As we have demonstrated, a complex plane wave decomposition is capable of producing a statistically valid layer of dynamic turbulence by summing as few as $N = 750$ components. If the coherence time and average velocity of layers in the atmosphere demands ~ 1 ms time steps between turbulence realizations, and the simulation is only for one second of data collection, this will quickly require thousands of layer realizations consisting of millions of matrices being summed. Due to the speed at which this problem scales, we have developed an efficient matrix method for fast formation of the layers over large regions which avoids making redundant calculations. Going forward, we will assume that all vectors are column vectors – requiring all row vectors to be the transpose of a vector.

Since we have chosen to use the complex exponential form of the wave decomposition, we can separate the time and space components of the wave element n and write them as a matrix generated over the coordinate sample vectors of the layer ℓ at a specific time:

$$\begin{aligned}\tilde{\boldsymbol{\phi}}_n(t) &= A_n e^{i\vec{k}_n\cdot\left(\vec{x}_\ell^{\mathsf{T}}+\vec{y}_\ell\right)} e^{i\Psi_n(t,\theta_n)}\\ &= A_n\left[e^{ik_n\sin(\theta_n)\vec{y}_\ell}\left(e^{ik_n\cos(\theta_n)\vec{x}_\ell}\right)^{\mathsf{T}}\right] e^{k_n t|\vec{v}_\ell|\cos(\psi_n)}\\ &= A_n\left[\vec{V}_{y_\ell,n}\left(\vec{V}_{x_\ell,n}\right)^{\mathsf{T}}\right] V_{v_\ell,n}(t).\end{aligned} \tag{3.27}$$

If $\vec{x}_\ell$ is a $P\times 1$ vector representing the sampled x-axis of the layer ℓ and $\vec{y}_\ell$ is a $Q\times 1$ vector representing the sampled y-axis, then Eq. (3.27) will produce the $Q\times P$ matrix of the n^{th} wave component at the time t. We now must organize each component $n\in[1,N]$ into matrices such that the sum in Eq. (3.4) is carried out with matrix multiplication and we are left with the $Q\times P$ matrix with all wave elements superimposed.

$\vec{V}_{y_\ell,n}$ and $\vec{V}_{x_\ell,n}$ are $Q\times 1$ and $P\times 1$ vectors, respectively, that form the complex wave spatial vectors given by the frequency component k_n angled at θ_n. We can append each vector for the n^{th} element column-wise to form the matrices

$$\begin{aligned}\mathbf{V}_{y_\ell} &= \begin{bmatrix}\vec{V}_{y_\ell,1} & \vec{V}_{y_\ell,2} & \dots & \vec{V}_{y_\ell,N-1} & \vec{V}_{y_\ell,N}\end{bmatrix}\\ &= \begin{bmatrix}e^{ik_1\sin(\theta_1)\vec{y}_\ell} & e^{ik_2\sin(\theta_2)\vec{y}_\ell} & \dots & e^{ik_{N-1}\sin(\theta_{N-1})\vec{y}_\ell} & e^{ik_N\sin(\theta_N)\vec{y}_\ell}\end{bmatrix}\\ &= e^{i\vec{y}_\ell\left[\vec{k}\odot\sin\left(\vec{\theta}\right)\right]^{\mathsf{T}}},\end{aligned} \tag{3.28}$$

$$\begin{aligned}\mathbf{V}_{x_\ell} &= \begin{bmatrix}\vec{V}_{x_\ell,1} & \vec{V}_{x_\ell,2} & \dots & \vec{V}_{x_\ell,N-1} & \vec{V}_{x_\ell,N}\end{bmatrix} \\ &= \begin{bmatrix}e^{ik_1\cos(\theta_1)\vec{x}_\ell} & e^{ik_2\cos(\theta_2)\vec{x}_\ell} & \dots & e^{ik_{N-1}\cos(\theta_{N-1})\vec{x}_\ell} & e^{ik_N\cos(\theta_N)\vec{x}_\ell}\end{bmatrix} \\ &= e^{i\vec{x}_\ell\left[\vec{k}\odot\cos\left(\vec{\theta}\right)\right]^\intercal},\end{aligned} \tag{3.29}$$

where $\odot$ designates element-by-element multiplication. Therefore, $\mathbf{V}_{y_\ell}$ is $Q \times N$ and $\mathbf{V}_{x_\ell}$ is $P \times N$. We can apply this notation to the time elements in the decomposition,

$$\left[\vec{V}_{v_\ell}(t)\right]^\intercal = \begin{bmatrix}e^{ik_1\cos(\psi_1)t|\vec{v}_\ell|} & \dots & e^{ik_N\cos(\psi_N)t|\vec{v}_\ell|}\end{bmatrix} = e^{i\left[\vec{k}\odot\cos(\vec{\psi})\right]^\intercal t|\vec{v}_\ell|}, \tag{3.30}$$

to produce an $1 \times N$ vector of phase velocity components. Repeating for the amplitudes produces the $1 \times N$ vector

$$\vec{A}^\intercal = \begin{bmatrix}A_1 & A_2 & \dots & A_{N-1} & A_N\end{bmatrix}. \tag{3.31}$$

We can now compute the full layer over $\vec{x}$ and $\vec{y}$ at the time t with the matrix product

$$\begin{aligned}\boldsymbol{\phi}_\ell(t) &= \sum_n^N \tilde{\boldsymbol{\phi}}_n(t) \\ &= \left[\mathbf{V}_{y_\ell} \odot \vec{A} \odot \vec{V}_{v_\ell}(t)\right]\mathbf{V}_x^\intercal.\end{aligned} \tag{3.32}$$

Checking the matrix dimensions:

$$\left[(Q \times N) \odot (N \times 1) \odot (N \times 1)\right](N \times P) = (Q \times P)$$

which is a layer with correct matrix dimensions.

Notice that for a layer of turbulence which is simulated at values of t, the only function of time in Eq. (3.32) is $\vec{V}_{v_\ell}(t)$. This means that before starting a dynamic turbulence simulation, $\mathbf{V}_{x_\ell}$, $\mathbf{V}_{y_\ell}$, and $\vec{A}$ can be pre-generated and stored. At each time step after beginning the simulation, the new $\vec{V}_{v_\ell}(t)$ can be generated and the matrix multiplication can be carried out. Thus, the prescribed method greatly reduces the number of calculations needed to generate each layer in time.

3.2 System simulation for Shack-Hartmann data generation

SHWFS data generation requires determining the phase in each subfield a and subaperture b accumulated at all layers of turbulence ℓ for each simulated time step t. Each cumulative phase can then be reduced to a tip/tilt measurement vector for the subfield-subaperture pair (a, b), notated $\vec{\alpha}_{a,b}$, and organized into the warp maps for each time step, $\mathbf{w}(t)$. As mentioned previously, the system warp maps are required for the layer distance estimation, velocity extraction, and phase reconstruction aspects of MOIC. Due to the compounded dimensionality of this process, we use an indexed matrix method for bookkeeping all of the data when forming the warp maps. Recall from Section (2.1.3) that it is too memory intensive to keep track of the phase matrices and the resulting subaperture image matrices for each subfield-subaperture pair a modern desktop computer. Instead, we will directly find the corresponding 2-element tip/tilt vectors for each subaperture-subfield phase and store them iteratively.

To ease the highly dimensional process of multi-layer subfield-subaperture tip/tilt determination, we propose five assumptions necessary to constrain the geometry of the system:

1. The layers are in the near field compared to the object distance. This forces each subfield to correspond to the same object angle for each subaperture. It also allows us to assume that the subaperture projections to each layer will be the same size as in the pupil.

2. The tip and tilt introduced at the layers are approximately small enough such that any geometric ray going from the object to the pupil will not change angle at the layer.

3. We require that the SHWFS subapertures are squares with side length s which form a regular grid inside of a larger square WFS pupil.

4. The system pupil is a circle with diameter D. The circular pupil can either be inscribed inside the WFS pupil, or the WFS pupil can be inscribed in the system pupil.

5. The object field is assumed to be a uniform grid of square subfields which are angularly separated by no more than the isoplanatic angle to obtain sufficient angular sampling.

From these assumptions, we can define the object space chief and marginal rays for each subfield-subaperture pair as rays with a constant angle between the object and the system pupil. With the known chief and marginal rays, a first order geometric ray trace to each layer can be performed and the cumulative phase in each subaperture along each subfield can be extracted.

3.2.1 Geometric raytrace model

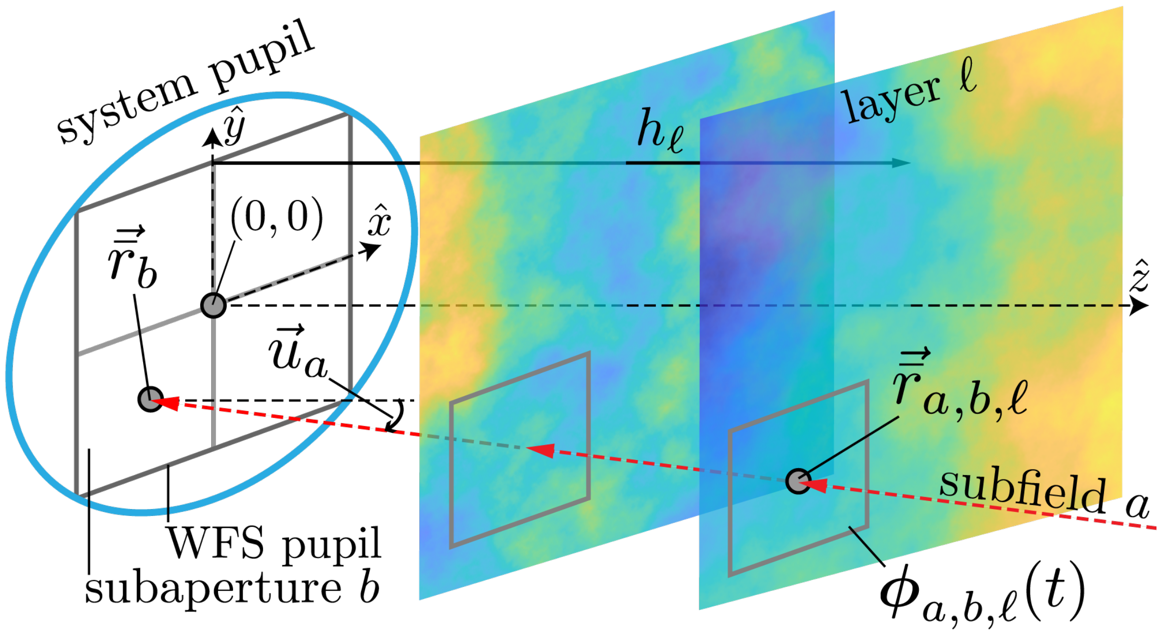


FIGURE 3.5: Geometric raytrace geometry for a single subfield-subaperture combination to a specific layer of turbulence. The chief ray coordinate at the layer, $\vec{\vec{r}}_{a,b,\ell}$, is constrained by the system assumptions made, the chief ray coordinate at the pupil, $\vec{\vec{r}}_b$, the subfield ray angle, $\vec{u}_a$, and the layer distance h_ℓ. The phase in each subaperture at the system pupil is the sum of the matrices of phase at each layer sampled within the aperture projections $\boldsymbol{\phi}_{a,b,\ell}(t)$. The WFS pupil is defined by the outside boundary of the uniform grid of subapertures. The WFS pupil can either be inscribed inside of the system pupil (as shown), or the system pupil can be inscribed in the WFS pupil.

Consider the SHWFS system geometry in Fig. (3.5). From the predetermined system aperture size and by choosing the number of subapertures that fill the WFS pupil, as well as by placing the origin at either the corner of a subaperture for an even array or the center of a subaperture for an odd array, the middle coordinate of each subaperture is constrained. The middle coordinate is stored as the subaperture-specific chief ray coordinate in the system pupil

$$\vec{\vec{r}}_b = \begin{bmatrix} x_b \\ y_b \end{bmatrix}. \tag{3.33}$$

By segmenting the optical system field of view into a uniform grid, each subfield angle is known. Using a compound angle notation, whereby ξ_a is the angle in the xz-plane and ζ_a is the angle in the yz-plane, each subfield ray angle can be written as the vector

$$\vec{u}_a = \begin{bmatrix} \xi_a \\ \zeta_a \end{bmatrix}. \tag{3.34}$$

With the known chief ray coordinates in the pupil, as well as their ray angles, the subfield-subaperture chief rays at any simulated layer ℓ a distance h_ℓ away is given by the geometric raytrace

$$\vec{\vec{r}}_{a,b,\ell} = \vec{\vec{r}}_b - h_\ell \vec{u}_a. \tag{3.35}$$

To determine the coordinates for all system chief rays, an indexed form of the above geometric raytrace has been developed. By stacking the subaperture coordinate vectors column-wise and the subfield angle vectors row-wise, system coordinate matrices can be formed. Following the organization method for this in Appendix (A), matrix methods can be used to perform all required system ray traces to a single layer simultaneously:

$$\bar{\mathbf{r}}_\ell = \bar{\mathbf{r}}_{a,b} - h_\ell \bar{\mathbf{u}}_{a,b}. \tag{3.36}$$

In the above expression, $\bar{\mathbf{r}}_{a,b}$ is the matrix of all subfield-subaperture chief ray coordinates in the system pupil given by Eq. (A.2), and $\bar{\mathbf{u}}_{a,b}$ is the matrix of corresponding subfield angles given by Eq. (A.7). Repeating the calculation for each layer and appending the results along the third matrix dimension,

$$\bar{\mathbf{r}} = \begin{bmatrix} [\bar{\mathbf{r}}_1] & \dots & [\bar{\mathbf{r}}_\ell] & \dots & [\bar{\mathbf{r}}_L] \end{bmatrix}, \tag{3.37}$$

produces the final matrix $\bar{\mathbf{r}}$. $\bar{\mathbf{r}}$ is a $2A \times B \times L$ matrix of chief ray coordinates for subfield a in subaperture b at layer ℓ. Using the matrix indexing notation in Eq. (3.1), the (x,y) chief ray coordinates at the layer are extracted using

$$\bar{x}_{a,b,\ell} = \bar{\mathbf{r}}\,(2a-1, b, \ell)\,, \tag{3.38}$$

$$\bar{y}_{a,b,\ell} = \bar{\mathbf{r}}\,(2a, b, \ell)\,. \tag{3.39}$$

For further detail on the construction of these matrices, as well as their marginal ray counterparts, see Appendix (A).

3.2.2 Using indexed chief ray matrices for efficient simulation

The indexed chief ray matrix from Eq. (3.36) contains an abundance of information to be used when optimizing the simulation environment. Here, we focus on streamlining two key SHWFS data generation processes using $\bar{\mathbf{r}}$. Both optimizations take advantage of the assumption that the subapertures are square and that their projections are the same size at all layers. This allows us to use the chief ray layer intercept coordinates, $(\bar{x}_{a,b,\ell}, \bar{y}_{a,b,\ell})$, as a method for directly sampling the matrix for phase at the layer:

$$\boldsymbol{\phi}_{a,b,\ell}(t) = \boldsymbol{\phi}_\ell\left([\bar{y}_{a,b,\ell} + s/2 : \bar{y}_{a,b,\ell} - s/2]\,, [\bar{x}_{a,b,\ell} - s/2 : \bar{x}_{a,b,\ell} + s/2)]\,, t\right), \tag{3.40}$$

where the sampling notation is that of Eq. (3.3).

Layer matrix size optimization

More distant layers of turbulence must be larger due to the cone spread effect of pupil projection for an extended field. As the size of the layer increases it takes longer to generate, motivating us to generate the smallest patch needed for each layer. The size of each simulated layer is defined by the x-axis sampling vector $\vec{x}_\ell$ and y-axis sampling vector $\vec{y}_\ell$. Therefore, we can use the chief ray sampling matrix to determine the minimum size sampling vectors for each layer distance.

For a system pupil which is inscribed in the WFS pupil, given the matrix indexing method for the matrix $\bar{\mathbf{r}}$ in Appendix (A), the top left chief ray coordinate at each layer is

$$\bar{x}_\ell^{(\mathrm{TL})} = \bar{\mathbf{r}}\left(1,1,\ell\right), \tag{3.41}$$

$$\bar{y}_\ell^{(\mathrm{TL})} = \bar{\mathbf{r}}\left(2,1,\ell\right). \tag{3.42}$$

The bottom right corner is at

$$\bar{x}_\ell^{(\mathrm{BR})} = \bar{\mathbf{r}}\left(2B-1,A,\ell\right), \tag{3.43}$$

$$\bar{y}_\ell^{(\mathrm{BR})} = \bar{\mathbf{r}}\left(2B,A,\ell\right). \tag{3.44}$$

From this, using a layer pixel scale Δx_ℓ and Eq. (3.40), we can see that the minimum sized layer coordinate axis vectors are

$$\vec{x}_\ell = \left[\bar{x}_\ell^{(\mathrm{TL})} - s/2\right] : \Delta x_\ell : \left[\bar{x}_{,\ell}^{(\mathrm{BR})} + s/2\right], \tag{3.45}$$

$$\vec{y}_\ell = \left[\bar{y}_\ell^{(\mathrm{TL})} + s/2\right] : -\Delta x_\ell : \left[\bar{y}_\ell^{(\mathrm{BR})} - s/2\right]. \tag{3.46}$$

The notation $x_1 : \Delta x_\ell : x_2$, similar to that of MATLAB, means that the vector starts at x_1 and changes in steps of $\pm\Delta x_\ell$ to the value x_2.

For the case where the WFS pupil is inscribed inside of the system pupil, the layer must be larger than the previous case to accommodate a system pupil which inscribes the WFS pupil. Fortunately, the system geometry is constrained because we are only considering square WFS pupils and circular system pupils. For a circular pupil of diameter D, the square which inscribes that circle has a diagonal $\sqrt{2}D$. Geometry can be used to show that the extensions of the layer in x and y to accommodate the larger system pupil are

$$x^{(+)} = y^{(+)} = D\frac{\sqrt{2}-1}{\sqrt{2}}. \tag{3.47}$$

Using this, the layer corners for the inscribed WFS pupil case are

$$\vec{x}_\ell = \left[\bar{x}_\ell^{(\mathrm{TL})} - s/2 - x^{(+)}\right] : \Delta x_\ell : \left[\bar{x}_\ell^{(\mathrm{BR})} + s/2 + x^{(+)}\right], \tag{3.48}$$

$$\vec{y}_\ell = \left[\bar{y}_\ell^{(\mathrm{TL})} + s/2 + y^{(+)}\right] : -\Delta x_\ell : \left[\bar{y}_\ell^{(\mathrm{BR})} - s/2 - y^{(+)}\right]. \tag{3.49}$$

Subaperture phase sampling optimization

In the case where the distance of the turbulence is not changing, the matrix $\bar{\mathbf{r}}$ does not change. This means that the the region where each subfield-subaperture combination projects to at each layer is constant at all time steps. Therefore, when the simulation is initialized all phase sampling regions can be predetermined with either Eq. (3.45) or Eq. (3.48). After generating the layers at the time step t, each matrix $\boldsymbol{\phi}_{a,b,\ell}(t)$ can be sampled using the predetermined layer matrix indices. The phase in each subaperture for each subfield can then be found by summing along all layers

$$\boldsymbol{\phi}_{a,b}(t) = \sum_{\ell}^{L} \boldsymbol{\phi}_{a,b,\ell}([\vec{y}_\ell],[\vec{x}_\ell],t). \tag{3.50}$$

At each subsequent time step in the simulation, the shifted patches of turbulence are generated and the same indices are used to sample the correct regions from each layer – no ray tracing needs to be recalculated.

3.2.3 Warp map element extraction

With a fast method for obtaining each matrix $\boldsymbol{\phi}_{a,b}(t)$ from the smallest layer necessary, the phase data can be reduced to the desired SHWFS slope vectors to form the system warp map. As discussed in Section (2.1.3) the image formation process for each subfield-subaperture phase with subsequent matched filtering to back out the pupil tip/tilt is extremely computationally and memory intensive. Instead, we have opted to circumvent that process and extract the subfield-subaperture tip/tilt vectors, $\vec{\alpha}_{a,b}(t)$, directly from each $\boldsymbol{\phi}_{a,b}(t)$.

Using the definition of 2D Legendre polynomials, see Appendix (C), the tilt of the subfield-subaperture specific phase can be found from

$$\begin{aligned} \alpha_{1,a,b}(t) &= \frac{1}{s} \int_{-1}^{1} \int_{-1}^{1} \mathscr{L}_1(x,y) \phi_{a,b}(x,y,t)\, dx\, dy \\ &= \frac{\sqrt{3}}{s} \int_{-1}^{1} \int_{-1}^{1} x \phi_{a,b}(x,y,t)\, dx\, dy. \end{aligned} \tag{3.51}$$

Similarly for tip we get

$$\alpha_{2,a,b}(t) = \frac{\sqrt{3}}{s} \int_{-1}^{1} \int_{-1}^{1} y \phi_{a,b}(x,y,t)\, dx\, dy. \tag{3.52}$$

These integrations are approximated discretely over the phase matrices, $\boldsymbol{\phi}_{a,b}(t)$, by forming sampled matrices of the functions $x \in [-1,1]$ and $y \in [-1,1]$, $\mathbf{x}$ and $\mathbf{y}$, respectively, and calculating

$$\alpha_{1,a,b}(t) = \frac{\Delta x^2}{s} \sum \sum \left[\mathbf{x} \odot \boldsymbol{\phi}_{a,b}(t) \right], \tag{3.53}$$

$$\alpha_{2,a,b}(t) = \frac{\Delta x^2}{s} \sum \sum \left[\mathbf{y} \odot \boldsymbol{\phi}_{a,b}(t) \right]. \tag{3.54}$$

The results are then stored in the vector

$$\vec{\alpha}_{a,b}(t) = \begin{bmatrix} \alpha_{1,a,b}(t) \\ \alpha_{2,a,b}(t) \end{bmatrix}. \tag{3.55}$$

Collecting $\vec{\alpha}_{a,b}(t)\ \forall\ a \in [1,A]\ \forall\ b \in [1,B]$ and storing them in a matrix results in the time-specific warp map $\mathbf{w}(t)$ – which contains all data needed to estimate the turbulence profile.

3.3 System image formation

While the methods for generating SHWFS data did not require an image formation model, we do need to develop one to simulate the image plane for which MOIC is to be applied to. We use a linear imaging system approach where the intensity image is considered to be a copy of the geometric object, $I_g(x,y)$, blurred by the system point spread function (PSF) $\bar{h}(x,y)$:

$$I_i(x,y) = I_g(x,y) \otimes \bar{h}(x,y), \tag{3.56}$$

where $\otimes$ is the convolution operator. However, with a wide field imaging system the field of view is greater than an isoplanatic angle. As discussed in Chapter (1), this means that the system PSF is anisoplanatic and changes for different field angles. The anisoplanatic PSF, $\bar{h}_\theta(x,y)$, can be used to determine the mapping between the geometric object along field θ to the image plane using

$$I_{i;\theta}(x,y) = I_{g;\theta}(x,y) \otimes \bar{h}_\theta(x,y). \tag{3.57}$$

3.3.1 Matrix method for discrete image formation

When simulating image formation using matrices, a complete treatment of the anisoplanatism would use a different PSF for each pixel in the object field. Generating each PSF takes valuable computing time, making the complete treatment costly. Instead, we can take advantage of the fact that we already assumed that the minimum subfield separation would be less than the isoplanatic angle. This means we can approximate the anisoplanatic PSF by determining the PSF for each subfield and using each result for the entire subfield. Additionally, by performing the operation along the established subfields we can take advantage of the phase sampling architecture which has already been developed in the simulation environment.

The phase collection process for the system pupil is procedurally identical to that of the SHWFS phase matrices. Having already ensured that each layer is large enough to capture the projection of the system pupil along every line of sight, the field-specific pupil phase matrix from each layer at the time t can be sampled using

$$\mathbf{\Phi}_{a,\ell}(t) = \mathbf{W} \odot \boldsymbol{\phi}_\ell \left([\bar{y}_{a,\ell} + D/2 : \bar{y}_{a,\ell} - D/2] , [\bar{x}_{a,\ell} - D/2 : \bar{x}_{a,\ell} + D/2)] , t \right), \tag{3.58}$$

where $\mathbf{W}$ is the matrix representation of the system aperture function. The chief ray coordinates at each layer are found using Eq. (3.35) with the initial condition that

$$\vec{\bar{r}} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \; \forall \, a \in [1, A]$$

in the pupil plane. Just as in the subaperture case, the cumulative pupil phase from all layers along each line of sight at time t is given by

$$\mathbf{\Phi}_a(t) = \sum_\ell \mathbf{\Phi}_{a,\ell}. \tag{3.59}$$

Using the field-specific pupil phase, we can define the optical pupil function at the time t as

$$\mathbf{p}_a(t) = \mathbf{W} \odot e^{i\mathbf{\Phi}_a(t)}. \tag{3.60}$$

Because our object-image relationship maintains a linear mapping, the field-specific PSF can be found using the discrete Fourier transform relationship

$$\bar{\mathbf{h}}_a(t) = \left| \mathrm{F}_2^{-1} \left[\mathbf{P}_a(t) \right] \right|^2, \tag{3.61}$$

where $\mathrm{F}_2^{-1}[\cdot]$ indicates the 2D discrete inverse Fourier transform over the spatial coordinates of the matrix.

Notice that instead of the optical pupil matrix, $\mathbf{p}_a(t)$, we are taking the Fourier transform of a new matrix, $\mathbf{P}_a(t)$, called the padded optical pupil matrix. By operating in the discrete domain, we must use zero-padding to obtain the desired pixel scale in the image plane when using discrete

Fourier transforms. When $\mathbf{p}_a(t) = \mathbf{P}_a(t)$, and $\mathbf{W}$ contains a circle of diameter D perfectly inscribed in the bounds of the square matrix, the pixel scale of the PSF is set at the diffraction limited angular sampling λ/D. Now consider the case where $\mathbf{p}_a(t)$ is an $M \times M$ matrix and $\mathbf{P}_a(t)$ is $2M \times 2M$. The scaling property of the discrete Fourier transform results in $\bar{\mathbf{h}}_a(t)$ having pixel scale $\lambda/(2D)$. Note that as the amount of zero-padding increases the time it takes to form system images also increases.

Once the anisoplanatic PSF has been calculated for each subfield, the convolution in Eq. (3.57) can be performed to estimate how any subfield matrix, $\mathbf{I}_{g;a}$, will appear in the image plane. Another option is to apply the convolution theorem which states that convolution in the spatial domain is multiplication in the Fourier domain. This produces

$$\mathbf{G}_{i;a}(t) = \mathrm{F}_2\left[\mathbf{I}_{g;a}\right] \odot \mathrm{F}_2\left[\bar{\mathbf{h}}_a(t)\right], \tag{3.62}$$

which can be returned to the spatial domain via

$$\mathbf{I}_{i;a}(t) = \mathrm{F}_2^{-1}\left[\mathbf{G}_{i;a}(t)\right]. \tag{3.63}$$

The method described by Eq. (3.62) and Eq. (3.63) requires several discrete Fourier transforms compared to the single convolution in Eq. (3.57). When using the fast Fourier transform algorithm, however, this method is reliably faster than computing a convolution. Note that many programming languages, MATLAB included, use the fast Fourier transform method when calling a convolution from a function library.

Anisoplanatic PSF demonstration

Using the methods developed in this chapter, an example of a $D = 2$ m optical system looking along three different lines of sight through an atmosphere with two layers of von Kármán turbulence was modeled for the purpose of observing PSF anisoplanatism. The three modeled fields were $[\xi_a, \zeta_a] = ([-1', 1'], [0, 0], [1', -1'])$, and the results are shown in Fig. (3.6).

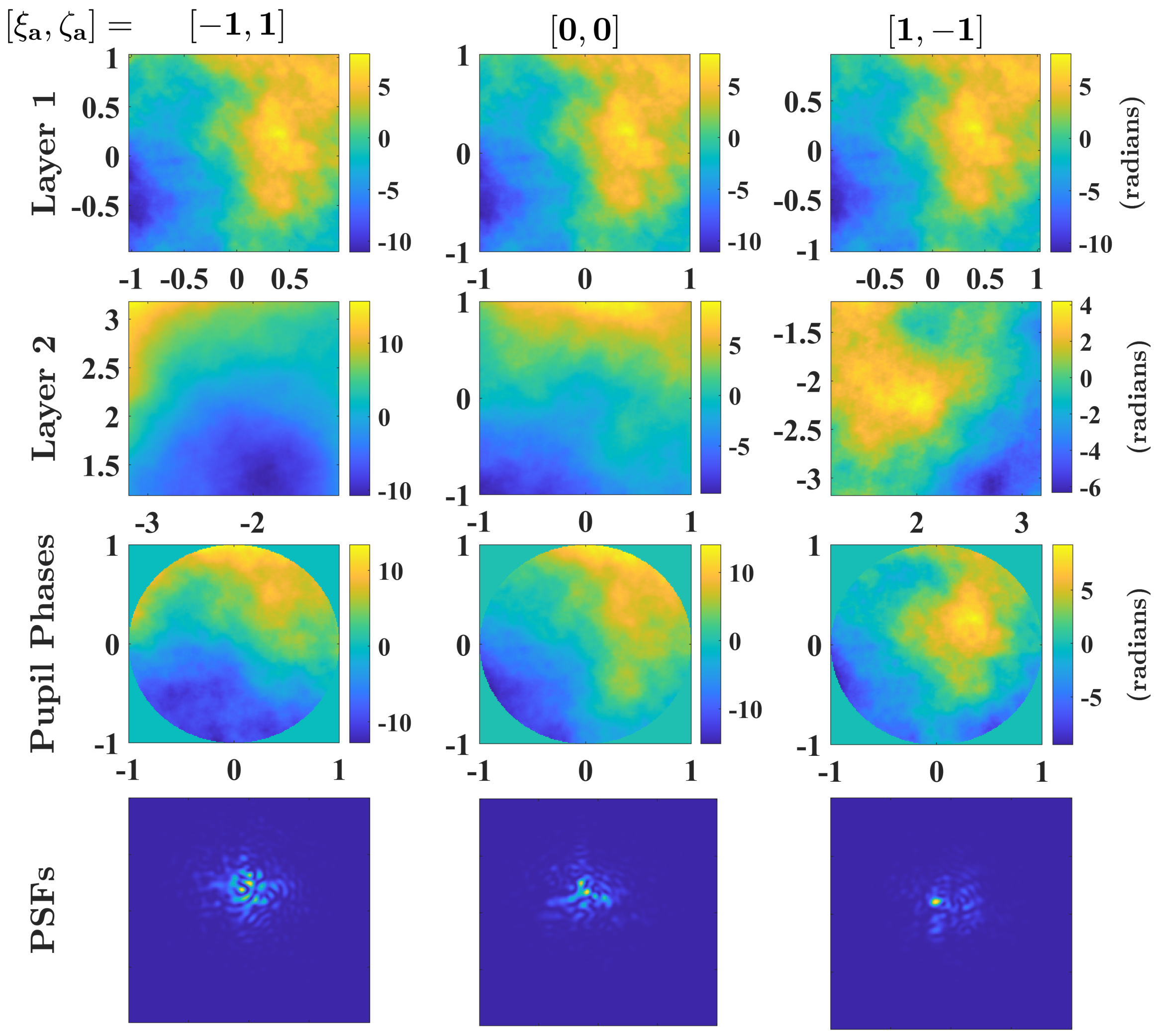


FIGURE 3.6: Each column represents model outputs for the designated field angle $[\xi_a, \zeta_a]$ in units of arcminutes. The first row is layer 1 with $h = 100$ m, $r_0 = 0.25$ m, and $L_0 = 25$ m over the pupil projection region. We can see that the region is almost identical for all three field angles because the layer is close to the system pupil. The second row is for layer 2 with $h = 7500$ m, $r_0 = 0.5$ m, and $L_0 = 250$ m. The isoplanatic angle of this layer is much smaller than the angular separation between the three fields, resulting in drastically different looking regions of phase in the pupil projections. The third row shows the sum of the layer phases in the pupil plane, which are then used to compute the resulting field-specific PSFs in the fourth row. The PSFs were generated with padded array $\mathbf{P}_a(t)$ such that the image pixels are $\lambda/(5D)$.

The first layer was modeled at 100 m with $r_0 = 0.25$ m and $L_0 = 25$ m. Using Eq. (1.6), this results in an isoplanatic angle at the first layer of $\theta_0 = 2.7'$. The second layer was placed at 7500 m with $r_0 = 0.5$ m and $L_0 = 250$ m, resulting in $\theta_0 = 0.07'$. Since 0.07' is much less than the separation between the field angles used in Fig. (3.6), we can predict that layer 2 will dominate the variations in pupil plane phase. This produces drastically different PSFs for each field direction which can be seen in the last row of the figure, confirming the presence of strong anisoplanatism.

3.3.2 Long exposure image and shot noise model

So far the discussion has been limited to using phase to determine the PSF for blurring an ideal geometric image. In a real system, the sensor integrates the blurred ideal geometric image from photons routed by the optics onto a pixel array – requiring some additional modeling steps for proper system image simulations. The quantized nature of counting photons leads to shot noise which we will include in our system images. Accounting for shot noise requires that the pixels of the ideal geometric object, $\mathbf{I}_g$, are in units of photon flux at the image plane. Making sure that this is properly represented, the blurred image matrix output by Eq. (3.63) will also be in units of photons per second.

Any time an image exposure is long enough such that the phase in the pupil changes by a few pixels, a long exposure image model is required. A long exposure model uses multiple instantaneous realizations of pupil subfield phase, $\mathbf{\Phi}_a(t)$, at different sequential values of t to approximate the continuous evolution of the turbulence. Such a model has a turbulence framerate, f_t, and a sensor framerate, $f_s > f_t$. By selecting a turbulence framerate that satisfies

$$\frac{1}{f_t} < \tau_0, \tag{3.64}$$

we can confirm that the pupil phase is not changing drastically between each turbulence frame. At each turbulence frame, we can compute the pupil phase, the resulting PSF, and the system instantaneous images, $\mathbf{I}_i(t)$. Since the units of the image pixels are photons/second and our detector only specifically counts photons, the instantaneous sensor image is

$$\tilde{\mathbf{I}}_i(t) = \frac{1}{f_t}\mathbf{I}_i(t) \tag{3.65}$$

which has pixels in units of incident photons. Doing this at each time step in the time sample vector $\vec{t}$ and stacking the results along a third matrix dimension produces

$$\tilde{\mathbf{I}}_i(\vec{t}) = \begin{bmatrix} [\tilde{\mathbf{I}}_i(t_0)] & [\tilde{\mathbf{I}}_i(t_1)] & \dots & [\tilde{\mathbf{I}}_i(t_T)] \end{bmatrix}. \tag{3.66}$$

If we choose the sensor framerate such that $f_s/f_t = T$, then the final system image for the exposure of length $\Delta t = 1/f_s$ starting at time $t = t_0$ is

$$\bar{\mathbf{I}}_{i;t_0} = \sum_{j=1}^{T} \tilde{\mathbf{I}}_i(t_j). \tag{3.67}$$

Each matrix being summed in the above equation is the distribution of photons in the system image plane resulting from an approximately constant instantaneous PSF along each subfield a over consecutive time windows of $1/f_t$. Thus, the superposition of all of these photon distributions is the sensor photon image from the exposure time Δt.

Shot noise is an additive noise process which obeys the arrival statistics of photons. Photon arrivals are independent and Poisson distributed random events, requiring that it has variance equal to the mean number of photons (Frieden 2011). The expectation of the number of photons is given by the pixel values in the image matrix $\bar{\mathbf{I}}_{i;t_0}$. Thus, the image matrix with shot noise added is

$$\bar{\mathbf{I}}_{i;t_0;s} = \mathscr{P}(\bar{\mathbf{I}}_{i;t_0}). \tag{3.68}$$

$\mathscr{P}(\mathbf{x})$ is an operator which takes each element in the 2D matrix $\mathbf{x}$, $x_{i,j}$ in units of photons, and returns a random number from a Poisson distribution with mean $x_{i,j}$. Such a function exists in packages like MATLAB (poissrnd) and Python (numpy.random.poisson).

Once the treatment of the image in the photon domain has been completed, the photons can be moved into the electrical domain which models the actual output of image sensors. The simplest forms of photon to electron conversion is through quantum efficiency (QE). The quantum efficiency is the wavelength-dependent ratio of photons which are expected to become electrons to the total number of incident photons. The total number of electrons are then sent through an amplifier, with gain g, and an analog-to-digital converter (ADC). The ADC turns the electron signal into a discrete number of bits which can be used to store image data on a computer. The conversion process is written as

$$\bar{\mathbf{I}}_{i;t_0;d} = \text{round}\left\{g(QE)\bar{\mathbf{I}}_{i;t_0;s}\right\}, \tag{3.69}$$

where $\text{round}\{\cdot\}$ is a matrix operator that rounds each pixel to the nearest integer value. Then, if an N-bit detectors is being modeled, we have to clip all pixels with a value greater than 2^N – i.e. the maximum measurable signal for the sensor. The units of the image after this process are analog-digital units (ADU).

Exposure length and noise demonstration

As the exposure time increases, the PSFs which form each image will approach the long-exposure PSF. Recall from Section (1.1.1) that the width of the PSF for long exposures through the atmosphere is $\sim \lambda/r_0$. Therefore, a sufficiently long exposure captured by a system with $D > r_0$ with form an image which is approximately the same as a system with $D = r_0$.

To analyze this effect, the methods developed in this chapter were used to simulate three different exposures times of a $1' \times 1'$ scene of the international space station (ISS). The imaging system was given $D = 2$ m, and the atmosphere consisted of two layers with cumulative $r_0 \approx 0.1$ m. $f_t = 250$ Hz was used such that Eq. (3.64) was approximately satisfied. Within 100 ms, the amount of high spatial frequency information in the frame is noticeably reduced. At a 1 s exposure, the blurred image is comparable to the same scene imaged by an optical system with $D = r_0 = 0.1$ m. These results are shown in Fig. (3.7).

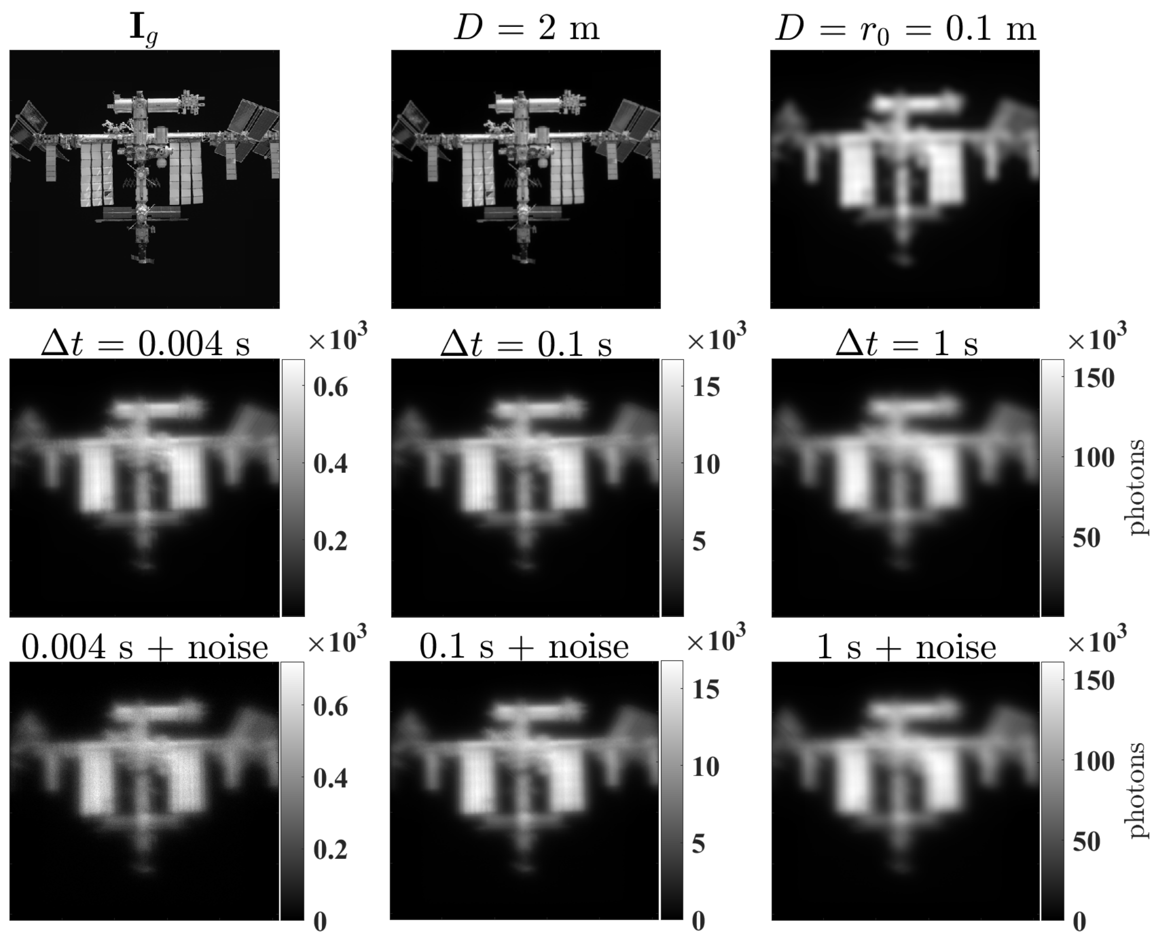


FIGURE 3.7: A series of blurred scenes produced by altering model parameters when imaging ideal geometric image, $\mathbf{I}_g$ shown in the top left plot. The scene is a model of the ISS reflecting enough sunlight to be approximately magnitude -2 in the V-band. The middle plot in the top row is the image produced by a $D = 2$ m optical system without any turbulence, and the plot to the right of that is for a system with $D = 0.1$ m. The middle row shows images simulated for a two layer atmosphere with $\sim r_0 = 0.1$ m at the pupil for three different exposures times. The bottom row shows the same images as the middle row with the addition of shot noise. We can see that while the shortest exposure preserves the most high-spatial frequency content, it is the noisiest image since it has fewer collected photons. By the time $\Delta t = 1$ s the shot noise is negligible, but the atmosphere has effectively blurred the scene to angular resolution λ / r_0. Perfect tracking of the ISS in flight was assumed.

Chapter 4 Layer signal to noise ratio

The previous two chapters provided the theory behind turbulence profiling with a SHWFS and a method for simulating realistic SHWFS data. To explore and demonstrate turbulence profiling with machine learning, we need to simulate a large and diverse set of training data using the developed theory and simulation environment. However, if we were to generate random multi-layer atmospheres and the resulting SHWFS data we would run into a specific problem: weak layers in a turbulence profile can be impossible to find in the presence of much stronger layers.

Layers with signals much weaker than strong layer noise will not be measurable. In the context of neural networks, these unmeasurable layers are unlearnable features. Training neural networks on data containing unlearnable features will result in training errors and poor network performance. Therefore, to obtain accurate neural networks, training data generated in simulation must be conditioned to avoid using layers with an undetectable signals (Hamilton and Hart 2022). The metric which quantifies the layer signal against measurement noise is called the layer signal-to-noise ratio (SNR).

4.1 Statistical SNR from SLODAR geometry

To derive an expression for the layer SNR, consider the analysis system geometry in Fig. (4.1). The measurement pair $\Delta\vec{\theta}_{a,a'}$ and $\Delta\vec{s}_{b,b'}$ overlap perfectly at a distance h_{ℓ_0} following Eq. (2.11). For layers at distances where the subaperture projections do not overlap, the projections are instead separated by the vector $\delta\vec{s}_{\ell_c}$. These layers of imperfect overlap are denoted as noise layers and are indexed $\ell_c \in [1, L-1]$.

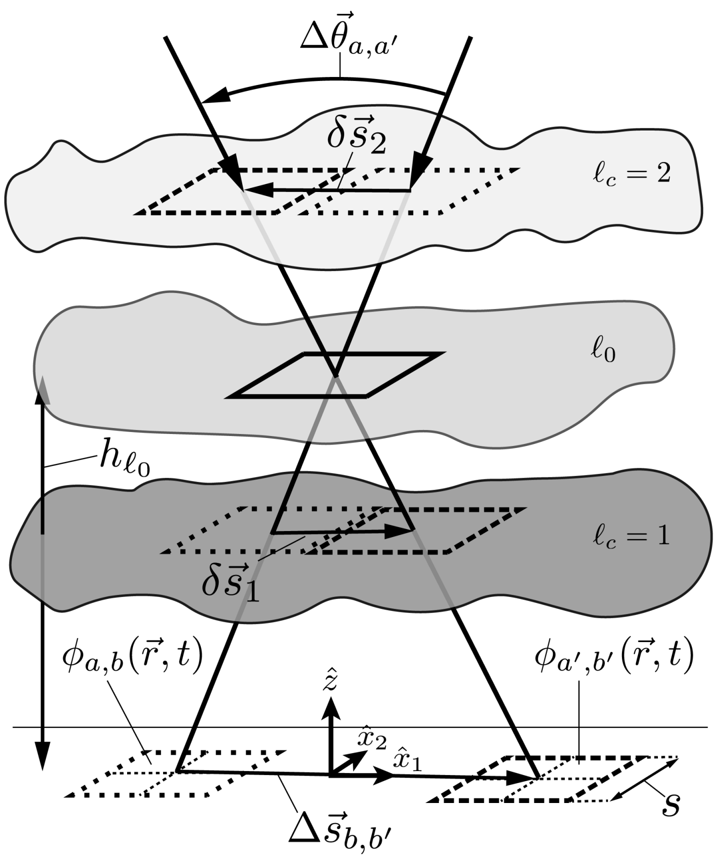


FIGURE 4.1: Analysis SLODAR geometry for deriving the layer SNR. The signal layer is denoted ℓ_0 and the noise layers are labeled $\ell_c \in [1, L-1]$. The assumptions from Section (2.2.3) that $\Delta\vec{\theta}_{a,a'}$ and $\Delta\vec{s}_{b,b'}$ are anti-parallel to each other and parallel to the row vector $\hat{x}_1$ or column vector $\hat{x}_2$ are maintained.

The cumulative phases in the two subapertures along their respective subfield angles are

$$\phi_{a,b}(\vec{r},t) = \sum_{\ell}^{L} \phi_{a,b,\ell}(\vec{r},t), \tag{4.1}$$

$$\phi_{a',b'}(\vec{r},t) = \sum_{\ell}^{L} \phi_{a',b',\ell}(\vec{r},t), \tag{4.2}$$

which are converted to subfield-subaperture slopes using the tip and tilt Legendre mode projection in Eq. (3.51) and Eq. (3.52), respectively. Applying the Legendre tip/tilt projection formula to

Eq. (4.1) and Eq. (4.2), the total tip and tilt vectors in the pupil can be expressed as the sum of the tip and tilt vectors from each layer:

$$\vec{\alpha}_{a,b}(t) = \sum_{\ell}^{L} \vec{\alpha}_{a,b,\ell}(t), \tag{4.3}$$

$$\vec{\alpha}_{a',b'}(t) = \sum_{\ell}^{L} \vec{\alpha}_{a',b',\ell}(t). \tag{4.4}$$

In the analysis geometry, we know that one of the layers is a signal layer and that all other layers are noise layers. Therefore, we can rewrite the summations in Eq. (4.3) and Eq. (4.4) in terms of the signal and noise components:

$$\vec{\alpha}_{a,b}(t) = \vec{\alpha}_{a,b,\ell_0}(t) + \sum_{\ell_c}^{L-1} \vec{\alpha}_{a,b,\ell_c}(t), \tag{4.5}$$

$$\vec{\alpha}_{a',b'}(t) = \vec{\alpha}_{a,b,\ell_0}(t) + \sum_{\ell_c}^{L-1} \vec{\alpha}_{a',b',\ell_c}(t), \tag{4.6}$$

where we have chosen to use the fact that $\vec{\alpha}_{a,b,\ell_0}(t) = \vec{\alpha}_{a',b',\ell_0}(t)$ to include the same form of the signal tip/tilt vector in both expressions.

As covered in Section (2.2.3), layer detection is performed by taking the dot product of all combinations of $\vec{\alpha}_{a,b}$ and $\vec{\alpha}_{a',b'}$ and analyzing the results for peaks in correlation. Taking the dot product of Eq. (4.5) and Eq. (4.6), expanding the sums and regrouping the terms, we can write the tip/tilt correlation as

$$\vec{\alpha}_{a,b}(t) \cdot \vec{\alpha}_{a',b'}(t) = S_{\ell_0}(t) + N_{\ell_c}(t) + R_{\ell}(t) \tag{4.7}$$

where

$$S_{\ell_0}(t) = |\vec{\alpha}_{a,b,\ell_0}(t)|^2, \tag{4.8}$$

$$N_{\ell_c}(t) = \sum_{\ell_c}^{L-1} \vec{\alpha}_{a,b,\ell_c}(t) \cdot \vec{\alpha}_{a',b',\ell_c}(t), \tag{4.9}$$

and

$$R_\ell(t) = \sum_\ell \sum_{\ell' \neq \ell} \vec{\alpha}_{a,b,\ell}(t) \cdot \vec{\alpha}_{a',b',\ell'}(t). \tag{4.10}$$

S_{ℓ_0} is the signal term indicating projected subaperture overlap at the layer ℓ_0. N_{ℓ_c} is a correlated noise term which arises from taking the dot product of tip/tilt vectors generated from spatially separated subaperture projections in the same noise layer. R_ℓ is a random noise term resulting from taking the dot product between tip and tilt vectors in different layers.

The random noise term can be removed by averaging Eq. (4.7) over multiple measurements in time. Since the mean value of tip and tilt in turbulence will be zero, and because each layer is statistically independent from all other layers,

$$\langle R_\ell(t) \rangle = \sum_\ell \sum_{\ell' \neq \ell} \langle \vec{\alpha}_{a,b,\ell}(t) \rangle \cdot \left\langle \vec{\alpha}_{a',b',\ell'}(t) \right\rangle = 0. \tag{4.11}$$

Applying a time average to the random noise requires that the signal and correlated noise components are also time averaged. For the time-averaged signal we find

$$\langle S_{\ell_0}(t) \rangle = \bar{S}_{\ell_0} = \left\langle |\vec{\alpha}_{a,b,\ell_0}(t)|^2 \right\rangle = \sigma^2_{\text{tip}} + \sigma^2_{\text{tilt}}. \tag{4.12}$$

The above expression states that the time-averaged signal of the signal layer will produce SLODAR correlations equal to the sum of the layer tip and tilt variances, σ^2_{tip} and σ^2_{tilt}, respectively. A similar result is observed for the correlated noise term once we acknowledge that each layer is statistically stationary in increments:

$$\begin{aligned} \langle N_{\ell_c}(t) \rangle = \bar{N}_{\ell_c} &= \sum_{\ell_c}^{L-1} \left\langle \vec{\alpha}_{a,b,\ell_c}(t) \cdot \vec{\alpha}_{a',b',\ell_c}(t) \right\rangle \\ &= \sum_{\ell_c}^{L-1} \langle \vec{\alpha}_{\ell_c}(\vec{r},t) \cdot \vec{\alpha}_{\ell_c}(\vec{r} + \delta\vec{s}_{\ell_c}, t) \rangle \\ &= \sum_{\ell_c}^{L-1} B_{\text{tip}}(\delta\vec{s}_{\ell_c}) + B_{\text{tilt}}(\delta\vec{s}_{\ell_c}). \end{aligned} \tag{4.13}$$

$B_{\text{tip}}(\delta\vec{s}_{\ell_c})$ and $B_{\text{tilt}}(\delta\vec{s}_{\ell_c})$ are the tip and tilt autocorrelations for the center-to-center aperture separation $\delta\vec{s}_{\ell_c}$ at the correlated noise layer.

Using the average signal and noise terms, we define the SNR as

$$SNR = \frac{\bar{S}_{\ell_0}}{\bar{N}_{\ell_c}} = \frac{\sigma_{\text{tip}}^2 + \sigma_{\text{tilt}}^2}{\sum_{\ell_c}^{L-1} B_{\text{tip}}(\delta\vec{s}_{\ell_c}) + B_{\text{tilt}}(\delta\vec{s}_{\ell_c})}. \tag{4.14}$$

We can better understand this relationship by recognizing that the autocorrelation of a zero-mean random process can be written as

$$B_{\text{tip}}(0) = \sigma_{\text{tip}}^2, \qquad B_{\text{tilt}}(0) = \sigma_{\text{tilt}}^2. \tag{4.15}$$

This implies that the statistical form of the SNR in Eq. (4.14) is composed entirely of the tip and tilt autocorrelation functions at each layer. The individual autocorrelations depend on the layer strengths and each value of $\delta\vec{s}_{\ell_c}$ resulting from the measurement set $(\Delta\vec{\theta}_{a,a'}, \Delta\vec{s}_{b,b'}, h_{\ell_c})$. When there is a layer corresponding to the distance h_{ℓ_0}, the autocorrelation for tip/tilt vector dot products which sample that distance peak to the sum of the tip and tilt variances. If the sum of the autocorrelations from all noise layers exceeds the signal, the SNR will be less than one and the layer will be difficult or impossible to locate in the measurement data.

The statistical form of layer SNR is useful in the context of understanding what constitutes signal and noise in SLODAR measurements, but it is not practical for computing the SNR of modeled layers. What we seek is a way to estimate the SNR of a layer form basic model parameters before generating the layer matrices. Therefore, we seek an analytical expression form of Eq. (4.14) which can be calculated directly from model parameters, such as measurement geometry and layer Fried length.

4.2 SNR for Kolmogorov and von Kármán layers of turbulence

4.2.1 Generalized Legendre mode autocorrelation in turbulence

Deriving a form of the SNR which is in terms of model parameters requires an analytical expression for the tip and tilt Legendre coefficient autocorrelation functions. We start from the known expression for the crosscorrelation of any two mode coefficients in an orthonormal basis decomposition of a pupil aberrated by turbulence (Whiteley et al. 1998):

$$B_{\alpha_j,\alpha_{j'};\ell}(\delta\vec{s}_\ell) = \int_{-\infty}^{\infty} E(\vec{k};\delta\vec{s}_\ell) S_{\phi_\ell}(\vec{k}) Q_j(\vec{k}s/2) Q^*_{j'}(\vec{k}s/2)\, d\vec{k}, \tag{4.16}$$

where

$$E(\vec{k};\delta\vec{s}_\ell) = e^{i2\pi\vec{k}\cdot\delta\vec{s}_\ell}, \tag{4.17}$$

$$Q_j(\vec{k}s/2) = \frac{1}{(s/2)^2}\int_{-\infty}^{\infty} \mathscr{L}_j\left(\frac{\vec{r}}{s/2}\right) W\left(\frac{\vec{r}}{s/2}\right) e^{-i2\pi\vec{k}\cdot\vec{r}}\, d\vec{r}. \tag{4.18}$$

In our case, $\mathscr{L}_j$ is the j^{th} 2D Legendre polynomial and $W\left(\frac{\vec{r}}{s/2}\right)$ is the square aperture function

$$W\left(\frac{x_1}{s/2},\frac{x_2}{s/2}\right) = \begin{cases} 1 & |x_1| \le s/2 \ \& \ |x_2| \le s/2 \\ 0 & \text{otherwise} \end{cases}. \tag{4.19}$$

As we can see, the autocorrelation depends on the layer power spectrum, $S_{\phi_\ell}(\vec{k})$. This means we will have different analytic expressions for Kolmogorov and von Kármán layer SNRs.

To solve the integral in Eq. (4.18), we choose the Rodrigues' formula definition of 2D Legendre polynomials (see Appendix (C)):

$$\mathscr{L}_j\left(\frac{\vec{r}}{s/2}\right) = c_{n,m}\frac{d^n}{dx_1^n}\left[\left(\left(\frac{x_1}{s/2}\right)^2 - 1\right)^n\right]\frac{d^m}{dx_2^m}\left[\left(\left(\frac{x_2}{s/2}\right)^2 - 1\right)^m\right], \tag{4.20}$$

where n and m are the one-dimensional orders of the 2D Legendre mode along orthogonal axes and

$c_{n,m}$ is the scaling coefficient to ensure orthonormality of the mode. Plugging this into the equation for $Q_j(\vec{k}s/2)$, along with the substitutions

$$y_1 = \frac{x_1}{s/2} \qquad y_2 = \frac{x_2}{s/2}, \tag{4.21}$$

we find

$$Q_j(\vec{k}s/2) = c_{n,m} \int_{-1}^{1} \frac{d^n}{dy_1^n}\left[\left(y_1^2-1\right)^n\right] e^{-i2\pi(k_1 y_1 s/2)}\, dy_1 \int_{-1}^{1} \frac{d^m}{dy_2^m}\left[\left(y_2^2-1\right)^m\right] e^{-i2\pi(k_2 y_2 s/2)}\, dy_2. \tag{4.22}$$

Integration by parts can be used to show that

$$\int_{-1}^{1} \frac{d^n}{dy_1^n}\left[\left(y_1^2-1\right)^n\right] e^{-i2\pi(k_1 y_1 s/2)}\, dy_1 = \ (i\pi s k_1)^n \int_{-1}^{1} \left(y_1^2-1\right)^n e^{-i\pi(s k_1 y_1)}\, dy_1. \tag{4.23}$$

When $\mathrm{Re}\{n\} > -1$, which is true since (n,m) are positive integers, the above integral has the known solution

$$\int_{-1}^{1} \left(y_1^2-1\right)^n e^{-i\pi(s k_1 y_1)}\, dy = (-1)^n \sqrt{\pi}(\pi k_1 s/2)^{-\frac{1}{2}-n}\Gamma(1+n)\mathbf{J}_{\frac{1}{2}+n}(\pi s k_1), \tag{4.24}$$

where $\mathbf{J}_{n+\frac{1}{2}}$ and $\mathbf{J}_{m+\frac{1}{2}}$ are half order Bessel functions of the first kind. Plugging Eq. (4.23) and Eq. (4.24) into Eq. (4.22) and simplifying results in

$$Q_{n,m}(\vec{k}s/2) = \bar{c}_{n,m} \frac{\mathbf{J}_{n+\frac{1}{2}}(\pi s k_1)\,\mathbf{J}_{m+\frac{1}{2}}(\pi s k_2)}{\sqrt{k_1 k_2}} \tag{4.25}$$

where

$$\bar{c}_{n,m} = \frac{c_{n,m}(-2)^{n+m}(i)^{n+m}}{s/2}\Gamma(n+1)\Gamma(m+1). \tag{4.26}$$

Using the solution for $Q_{n,m}(\vec{k}s/2)$ on Eq. (4.16) produces

$$B_{\alpha_j,\alpha'_j;\ell}(\delta\vec{s}_\ell) = \bar{c}_{n,m}\bar{c}_{n',m'} \int_{-\infty}^{\infty} E(\vec{k};\delta\vec{s}_\ell) S_{\phi_\ell}(\vec{k}) \frac{\mathbf{J}_{n+\frac{1}{2}}(\pi s k_1)\mathbf{J}^*_{n'+\frac{1}{2}}(\pi s k_1)\mathbf{J}_{m+\frac{1}{2}}(\pi s k_2)\mathbf{J}^*_{m'+\frac{1}{2}}(\pi s k_2)}{k_1 k_2}\, d\vec{k}. \tag{4.27}$$

Eq. (4.27) is the crosscorrelation between the Legendre mode coefficients α_j and $\alpha_{j'}$ for two square apertures with side length s and center-to-center separation $\delta\vec{s}_\ell$ at the layer of turbulence ℓ. To arrive at the autocorrelation, we set $(n',m') = (n,m)$ and find

$$B_{\alpha_j;\ell}(\delta\vec{s}_\ell) = |\bar{c}_{n,m}|^2 \int_{-\infty}^{\infty} E(\vec{k};\delta\vec{s}_\ell) S_{\phi_\ell}(\vec{k}) \left| \frac{\mathbf{J}_{n+\frac{1}{2}}(\pi s k_1)\mathbf{J}_{m+\frac{1}{2}}(\pi s k_2)}{\sqrt{k_1 k_2}} \right|^2 d\vec{k}. \tag{4.28}$$

4.2.2 Tip and tilt autocorrelations and variances

The tip and tilt autocorrelations, which make up the SNR, are specific cases of Eq. (4.28). Defining tip as $(n,m) = (1,0)$ and tilt as $(n,m) = (0,1)$, and following the proof in Appendix (B.1), it can be shown that

$$B_{\text{tip};\ell}(\delta\vec{s}_\ell) = C \int_{-\infty}^{\infty} E\left(\frac{\vec{\eta}}{\pi};\frac{\delta\vec{s}_\ell}{s}\right) S_{\phi_\ell}\left(\frac{\vec{\eta}}{\pi s}\right) T_1(\vec{\eta}\,)\, d\vec{\eta}, \tag{4.29}$$

$$B_{\text{tilt};\ell}(\delta\vec{s}_\ell) = C \int_{-\infty}^{\infty} E\left(\frac{\vec{\eta}}{\pi};\frac{\delta\vec{s}_\ell}{s}\right) S_{\phi_\ell}\left(\frac{\vec{\eta}}{\pi s}\right) T_2(\vec{\eta}\,)\, d\vec{\eta}, \tag{4.30}$$

where the substitutions

$$\begin{aligned} \eta_1 &= \pi s k_1 \\ \eta_2 &= \pi s k_2 \end{aligned} \tag{4.31}$$

were used to obtain the component functions

$$T_1(\vec{\eta}\,) = \left| \frac{\sin(\eta_2)[\sin(\eta_1) - \eta_1\cos(\eta_1)]}{\eta_1^2 \eta_2} \right|^2, \tag{4.32}$$

$$T_2(\vec{\eta}\,) = \left| \frac{\sin(\eta_1)[\sin(\eta_2) - \eta_2\cos(\eta_2)]}{\eta_1\eta_2^2} \right|^2. \tag{4.33}$$

The complex exponential in the original expression now has modified scaling and takes the form

$$E\left(\frac{\vec{\eta}}{\pi}; \frac{\delta\vec{s}_\ell}{s}\right) = e^{i2(\vec{\eta}\cdot\delta\vec{s}_\ell)/s}. \tag{4.34}$$

Using Eq. (4.15), the expressions for the tip and tilt variances can be found from the corresponding autocorrelation functions. Plugging $\delta\vec{s}_\ell = 0$ into the autocorrelation, the only functional component which is affected is the the complex exponential, defined in Eq. (4.34), which goes to a value of one. Everything else in the integrals remains unchanged, leaving us with layer tip/tilt variances

$$\sigma^2_{\text{tip};\ell} = C \int_{-\infty}^{\infty} S_{\phi_\ell}\left(\frac{\vec{\eta}}{\pi s}\right) T_1(\vec{\eta}\,)\, d\vec{\eta}, \tag{4.35}$$

$$\sigma^2_{\text{tilt};\ell} = C \int_{-\infty}^{\infty} S_{\phi_\ell}\left(\frac{\vec{\eta}}{\pi s}\right) T_2(\vec{\eta}\,)\, d\vec{\eta}. \tag{4.36}$$

4.2.3 SNR for Kolmogorov layers of turbulence

The Kolmogorov power spectrum in Eq. (3.22) can be written in Cartesian coordinates as

$$S^{(\text{kol})}_{\phi_\ell}(k_1,k_2) = C_{\text{kol}} r_0^{-5/3}(k_1^2 + k_2^2)^{-11/6}. \tag{4.37}$$

Scaling the power spectrum variables by $1/(\pi s)$ to match the form in the autocorrelation and variance integrals, and manipulating the resulting expression as in Appendix (B.3), it can be shown that the SNR of Kolmogorov turbulence is

$$\text{SNR}^{(\text{kol})}_{\ell_0} = r_{\ell_0}^{-5/3}\left[\sum_{\ell_c=1}^{L-1} r_{\ell_c}^{-5/3} \frac{\mathbf{R}_{\text{kol}}(\delta\vec{s}_{\ell_c})}{\mathbf{R}_{\text{kol}}(0)}\right]^{-1}, \tag{4.38}$$

where

$$\mathbf{R}_{\text{kol}}(\delta\vec{s}_{\ell_c}) = \int_{-\infty}^{\infty} E\left(\frac{\vec{\eta}}{\pi}; \frac{\delta\vec{s}_{\ell_c}}{s}\right) \frac{T_1(\vec{\eta}) + T_2(\vec{\eta})}{\left[\eta_1^2 + \eta_2^2\right]^{11/6}} d\vec{\eta} \tag{4.39}$$

is the Kolmogorov tip/tilt correlation coefficient (TTCC). The integral in Eq. (4.39) cannot be evaluated any further due to a divergent singularity at $\delta\vec{s}_{\ell_c} = 0$. We can, however, numerically integrate $\mathbf{R}_{\text{kol}}(\delta\vec{s}_{\ell_c})/\mathbf{R}_{\text{kol}}(0)$ in Eq. (4.38), called the normalized TTCC, because the ratio has a value of unity at $\delta\vec{s}_{\ell_c} = 0$.

The Kolmogorov layer SNR can be directly interpreted as a weighted sum of the layer strengths. The signal layer has full strength, designated by $r_{\ell_0}^{-5/3}$. Each noise layer has strength, $r_{\ell_c}^{-5/3}$, which is scaled by the weighting function – i.e. the normalized TTCC – depending on the normalized projected subaperture separation. Since the normalized TTCC peaks at $\delta\vec{s}_{\ell_c} = 0$, and then tends to decrease for increasing $\delta\vec{s}_{\ell_c}$, the noise component of any layer will carry less weight if the aperture projections are further apart.

Notice that $\text{SNR}_{\ell_0}^{(\text{kol})}$ is only a function of the signal layer Fried length, r_{ℓ_0}, the noise layer Fried lengths, r_{ℓ_c}, and the normalized subaperture separations at the noise layers, $\delta\vec{s}_{\ell_c}/s$. To simulate an atmosphere, the altitudes and r_0 values of each layer must already be specified. Simulating the data collection process requires determining where each subaperture for the modeled system intersects each layer, giving us knowledge of all possible values of $\delta\vec{s}_{\ell_c}/s$. Thus, the presented analytic expression for the Kolmogorov SNR can be computed directly from model parameters.

To improve the speed of computing layer SNRs, we can take advantage of the form of the normalized TTCC. Since the integration is for dummy variables (η_1, η_2) for a specific $\delta\vec{s}_{\ell_c}/s$, we can evaluate the normalized TTCC for a range of values of $\delta\vec{s}_{\ell_c}/s$ and store the results before specifying any model parameters. Once a SHWFS and atmosphere model have been defined, the values of $\delta\vec{s}_{\ell_c}$ and s can be used to extract the nearest neighbor value from the stored normalized TTCC for the measurement of interest. Without the need to perform a numerical integration every time a layer SNR is calculated, the SNR of a potentially modeled layer can be determined in microseconds on a standard desktop computer.

4.2.4 SNR for von Kármán layers of turbulence

We can rewrite the von Kármán power spectrum in Eq. (3.25) in Cartesian coordinates as

$$S_{\phi_\ell}^{(\mathrm{kol})}(k_1,k_2) = \frac{C_{\mathrm{vK}} r_0^{-5/3}}{(k_1^2+k_2^2+L_0^{-2})^{11/6}} e^{-(k_1^2+k_2^2)/(5.92/l_0)^2}. \tag{4.40}$$

Following the methods in Appendix (B.4), the von Kármán layer SNR can be put into an identical form as the Kolmogorov layer SNR:

$$\mathrm{SNR}_{\ell_0}^{(\mathrm{vK})} = r_{\ell_0}^{-5/3} \left[\sum_{\ell_c=1}^{L-1} r_{\ell_c}^{-5/3} \frac{\mathbf{R}_{\mathrm{vK}}(\delta \vec{s}_{\ell_c})}{\mathbf{R}_{\mathrm{vK}}(0)} \right]^{-1}. \tag{4.41}$$

Due to the functional form of its the power spectrum, the TTCC for von Kármán turbulence is

$$\mathbf{R}_{\mathrm{vK}}(\delta \vec{s}_{\ell_c}) = \int_{-\infty}^{\infty} e^{-(\eta_1^2+\eta_2^2)/(5.92\pi s/l_0)^2} \frac{E\left(\frac{\vec{\eta}}{\pi}; \frac{\delta \vec{s}_{\ell_c}}{s}\right)[T_1(\vec{\eta})+T_2(\vec{\eta})]}{\left[\eta_1^2+\eta_2^2+(\pi s/L_0)^2\right]^{11/6}} d\vec{\eta}. \tag{4.42}$$

Like the Kolmogorov TTCC, the von Kármán TTCC has a singularity at $\delta \vec{s}_{\ell_c} = 0$ which causes the integral in Eq. (4.42) to diverge. Fortunately, as was the case with Kolmogorov turbulence, the von Kármán normalized TTCC in Eq. (4.41) is unity where the un-normalized TTCC diverges.

In addition to the layer Fried lengths and $\delta \vec{s}_{\ell_c}/s$, the von Kármán layer SNR is a function of s/l_0 and s/L_0. This means that unlike the Kolmogorov turbulence case, the subaperture side length s and its proportion to the inner scale l_0 and outer scale L_0 will affect the von Kármán layer SNR. Based on the functional form of Eq. (4.42), we can see that smaller ratios for s/L_0 will result in a normalized TTCC which decreases more quickly for increasing $\delta \vec{s}_{\ell_c}/s$. This makes logical sense since subaperture projections at a noise layer which are separated by a distance larger than the outer scale should be approximately uncorrelated.

It is worth noting that that the SHWFS side length s will always be much larger than the inner scale of atmospheric turbulence in our analysis. This means that the complex exponential which depends on l_0 in Eq. (4.42) will be ≈ 1 – allowing us to primarily focus on the effects of different turbulence outer scales when discussing layer SNR.

It is still possible to numerically integrate the normalized von Kármán TTCC before simulation and analysis of a particular system. Instead of a single curve, as was the case for Kolmogorov turbulence, there is specific curve for each set $(s/l_0, s/L_0)$. Thus, several von Kármán normalized TTCCs for several different combinations of s/l_0 and s/L_0 must be integrated and stored before simulating to help improve SNR calculation speed.

4.3 Mote Carlo verification of layer SNR

The analytic expressions for the SNR can be verified using a Monte Carlo simulation. The statistical estimates of the SNR from the Monte Carlo simulation can then be directly compared to the SNRs calculated by Eq. (4.38) for Kolmogorov layers and Eq. (4.41) for von Kármán layers.

A computationally efficient method for checking the proposed layer SNR theory is by analyzing two layer atmospheres. Equating either analytical form of the SNR to Eq. (4.14), it can be shown that two layer atmospheres have a normalized TTCC which satisfies

$$\frac{\mathbf{R}(\delta\vec{s}_{\ell_c})}{\mathbf{R}(0)} = \left(\frac{r_{\ell_0}}{r_{\ell_c}}\right)^{-5/3} \frac{B_{\text{tip};\ell_c}(\delta\vec{s}_{\ell_c}) + B_{\text{tilt};\ell_c}(\delta\vec{s}_{\ell_c})}{\sigma^2_{\text{tip};\ell_0} + \sigma^2_{\text{tilt};\ell_0}}. \tag{4.43}$$

By generating several layers of turbulence with different values of r_0, the right side of Eq. (4.43) can be estimated. The statistical estimation of a single layer takes place in three steps:

1. generate a layer which is large enough to support apertures separated by the max $\delta\vec{s}_{\ell_c}$ of interest,

2. for subapertures of side length s, compute the tip and tilt coefficients in each $s \times s$ square region in the layer and store the results with the corresponding subaperture center coordinates,

3. average all pairs of tip/tilt vectors and their corresponding $\delta\vec{s}_{\ell_c}$ values to compute the signal, using Eq. (4.12), and noise, using Eq. (4.13).

This process is repeated for each layer with a different r_0 of interest. We then use each layer pair – e.g. for 4 different r_0 layers there are 10 permutations of a signal and noise layer – and their respective computed $(r_0, \bar{S}_{\ell_0}, \bar{N}_{\ell_c})$ values to estimate the right side of Eq. (4.43). The statistically estimated curves can then be plotted with the ones produced by the analytical expressions to for agreement.

4.3.1 Kolmogorov SNR validation

The Kolmogorov SNR was checked using layers with $r_0 = [0.1, 0.25, 0.3, 0.5, 1]$ m. Each layer was processed for the aperture side lengths $s = [0.05, 0.1, 0.2]$ m out to a maximum normalized aperture separation of $\delta\vec{s}_{\ell_c}/s = 50$. To minimize variation in the statistical estimates, several measurement averaging steps were taken. First, all measurements with identical normalized separations in the same layer were averaged. Second, the signal and noise term calculations outlined in Section (4.3) were repeated for 30 different realization times of each layer. Finally, the entire process of computing and averaging all signal and identically valued $\delta\vec{s}_{\ell_c}/s$ noise terms for 30 different simulation times for each layer was repeated and averaged for 30 independent relizations of a layer with Fried length r_0. We found this to be more than sufficient averaging to obtain good convergence in the statistics of layer signal and noise terms. The results are plotted in Fig. (4.2) along with the theoretical curves as a function of normalized subaperture separation (top plot) and absolute subaperture separation (bottom plot).

The statistically estimated curves are in agreement with the behavior predicted by the analytic expression form of the SNR. The Kolmogorov TTCC is approximately the same for all values of s when plotted as a function of normalized aperture separation. As a result, when plotted as a function of absolute aperture separation at a noise layer, the TTCC is stronger for larger apertures. This is because the larger apertures sample large sections of low frequency turbules as tip/tilt, which will decorrelate after large aperture separations. Opposite to this, smaller subapertures see low frequency turbules as piston which does not contribute to measured tip/tilt coefficients. The smaller subapertures derive their tilt from smaller scale turbules, which will decorrelate over shorter absolute

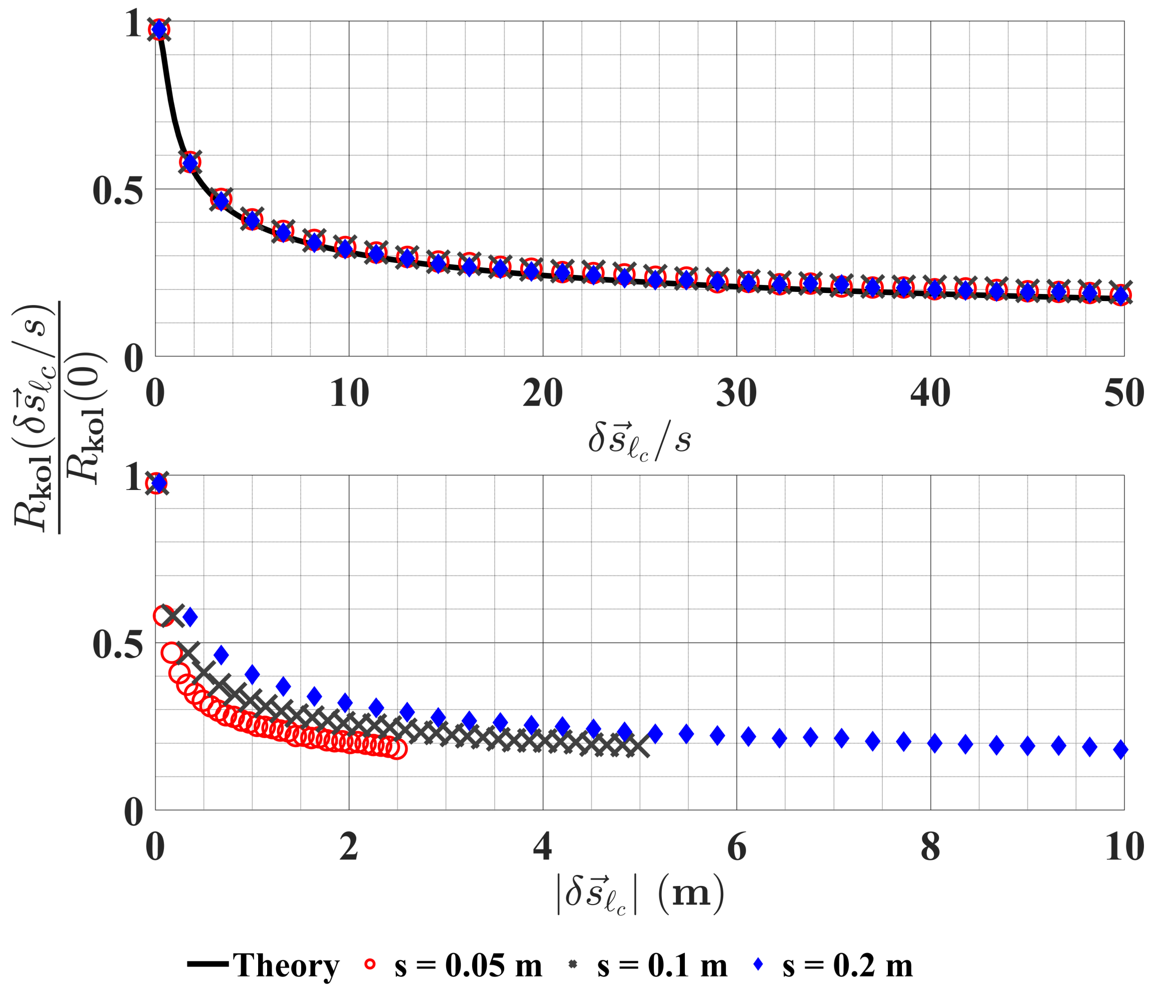


FIGURE 4.2: Top: Monte Carlo results and theoretical expectations for the Kolmogorov TTCC plotted as a function of $\delta\vec{s}_{\ell_c}/s$. In this plotting scheme, we can see that the the TTCC is approximately the same regardless of aperture size at the same value of $\delta\vec{s}_{\ell_c}/s$. Bottom: The Monte Carlo TTCC curves plotted as a function of $|\delta\vec{s}_{\ell_c}|$ in meters. In this plotting scheme, we can see how large apertures have greater correlation for the same physical center-to-center separation as smaller apertures.

separations. Therefore, if a SLODAR system is designed for specific angular sampling we can expect that in the presence of Kolmogorov turbulence the smaller subapertures will tend to have better layer measurement SNR than equally separated large apertures.

4.3.2 von Kármán SNR validation

The von Kármán layer SNR was checked with layer Fried lengths $r_0 = [0.1, 0.25, 0.3, 0.5, 1]$ m, each with outer scales $L_0 = [10, 100, 1000]$ m, for the aperture sizes $s = [0.025, 0.25]$ m. The signal and noise terms for each layer were averaged over 10 time steps for 50 realizations following the same process as the Kolmogorov layers. The results of this process, shown in Fig. (4.3), show excellent agreement between the the statistical estimations of $\mathbf{R}_{\text{vK}}(\delta\vec{s}_{\ell_c})/\mathbf{R}_{\text{vK}}(0)$ and the results computed with the analytic expression.

From the top plot in Fig. (4.3), we can observe behaviors in the layer SNR for different outer scales and subaperture side lengths that were not present in the Kolmogorov SNR. One result is that the normalized TTCCs for identical values of s/L_0 are the same when plotted as a function of $\delta\vec{s}_{\ell_c}/s$. We can also see that the larger the value of the ratio s/L_0, the quicker the von Kármán normalized TTCC goes to zero. This makes intuitive sense since apertures which are separated by the layer outer scale should be approximately uncorrelated.

The bottom plot shows the Monte Carlo results for the von Kármán normalized TTCC plotted as a function of absolute aperture separation at the layer. We can confirm from this that apertures separated by the outer scale have a normalized TTCC of zero. We also see that smaller subaperture will have weaker correlation for the same separation at a layer of turbulence, as was the case with Kolmogorov turbulence. The difference with von Kármán is that the TTCC also drops off more quickly for smaller L_0 - which was not true of Kolmogorov turbulence due to its infinite outer scale. Since layer outer scale is often proportional to altitude (Goodman 2000), the noise from layers close to the ground will tend to be weaker than high altitude layers – an implications which suggests that SLODAR layer detection is well suited for finding ground layers in astronomical applications as well as discriminating between layers when viewing terrestrial scenes.

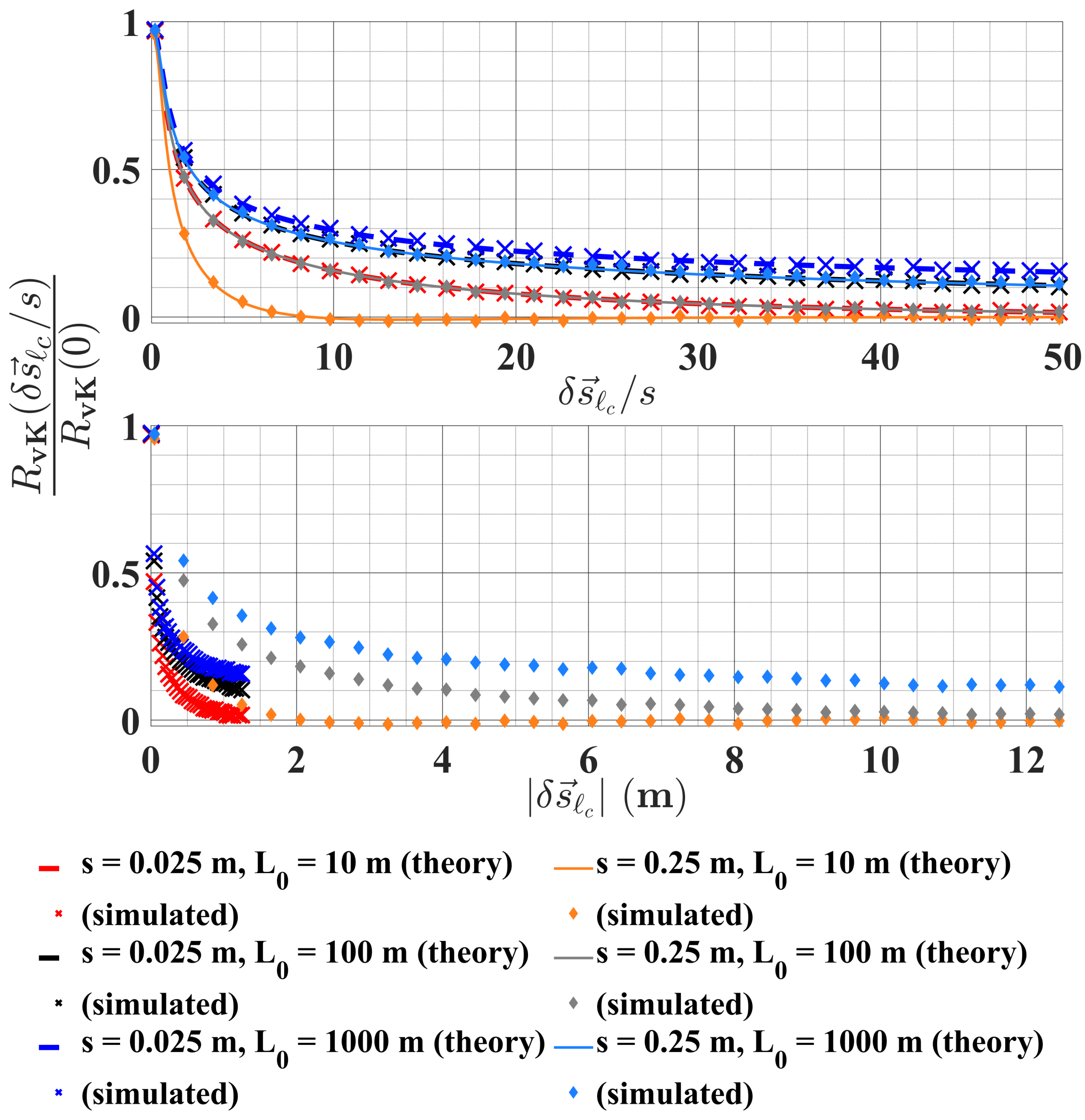


FIGURE 4.3: Top: Monte Carlo results and theoretical expectations for the von Kármán SNR TTCC plotted as a function of $\delta\vec{s}_{\ell_c}/s$. In this plotting scheme, TTCCs from layers with different aperture sizes and interital ranges but identical values of $(s/l_0, s/L_0)$. Bottom: The Monte Carlo TTCC curves plotted as a function of $|\delta\vec{s}_{\ell_c}|$ in meters. In this plotting scheme, we can see how large apertures have greater correlation for the same physical center-to-center separation as smaller apertures.

Chapter 5 Turbulence profiling with neural networks

5.1 Overview of turbulence profile extraction techniques

A primary source of error in wide field image correction systems is a result of incorrectly reconstructing the distribution of turbulence in front of the system (B. Neichel et al. 2009). Obtaining an accurate measurement of the turbulence profile is essential to an accurate reconstruction. There are two properties of the atmosphere which make obtaining an accurate measurement of the turbulence profile difficult:

1. the range of values that layer strengths and distances can take is large,
2. the number of layers, their distances, and their respective strengths can change on the time scale of minutes.

To compensate for property 1, the sample range of the measurement system must meet or surpasses where turbulence can exist out in front of the system and the algorithm which processes the measurements must be able to do so for layers with SNR ≈ 1. If the turbulence profiler can obtain good estimates in less than a minute from when data is collected, then property 2 of the atmosphere is also compensated. While the amount of light coming from the object scene plays a key part in how quickly data can be acquired, the speed at which the data can be processed into layer profile information is highly dependent on the interpretation algorithm used. Therefore, the choice of turbulence profile interpretation algorithm can play a major role in achieving high quality wide field turbulence compensation. The many different interpretation algorithms which have been used for turbulence profiling generally fall into two categories: model fitting and machine learning.

5.1.1 Profile extraction from model fitting

The majority of demonstrated turbulence profiling algorithms process SLODAR or SCIDAR data into an estimate of the $C_n^2(h)$ profile – defining the turbulence strength as a function of the distance h. The $C_n^2(h)$ profile is then distilled into an estimate of the number of layers and their altitude using a fitting method (Costille and Fusco 2012). Depending on the model fitting method employed, layer reconstruction and image correction quality varies.

Model fitting techniques are generally constructed of an initial model, a measured response, and an error metric. The initial model can be informed by typical values at the observation location or based on an initial estimate from contemporaneous system and auxiliary instrument data. The measured response is then compared to the model using an error metric intended to assess the closeness of the fit. The initial model is then adjusted incrementally based on a correction scheme, and the process is repeated until the error metric goes below a predetermined threshold. Since the final model depends on the initial model, and the turbulence profiles can vary drastically, fitting techniques which can quickly adapt to a changing atmosphere reliably perform better than models which assume an unchanging model (Farley et al. 2020).

A notable model fitting technique for wide field image correction is the 'Learn' portion of the Learn & Apply (L&A) algorithm (Vidal et al. 2010b). It was first used successfully on the MOAO system of the William Herschel telescope in La Palma, Canary Islands, in 2010 (Gendron et al. 2011). The technique uses an iterative process to minimize an error metric between the measured covariance of averaged slope data and a model of the calibrated optical system responding to a particular distribution of layers with parameters (r_0, h, L_0). The benefit of L&A is that it is constructed to allows for data from multiple WFSs to be used together to determine the turbulence along the on-axis line of sight. This is particularly useful for MOAO systems which coordinate multiple WFSs and DMs to achieve active correction of multiple layers of turbulence. The biggest challenge for L&A is converging to a solution quickly. The choice of initial model can affect the accuracy of the turbulence profile L&A converges to, as well as the speed at which convergence is reached. The lag time of the fitting process means it is not robust to changes in the turbulence profile, such as layers dissipating or appearing, causing the computed fit to accumulate error over time unless updated.

5.1.2 Profile extraction using artificial neural networks

As computers continue to improve, and large amounts of data become more easily accessible, machine learning techniques have started to gradually replace iterative fitting methods in many applications. The reoccurring challenge with model fitting in turbulence profiling is that algorithm convergence is time consuming and fit accuracy varies with the chosen model. Since machine learning algorithms, such a ANNs, learn to interpret the data during training, the model fitting aspect of turbulence profiling is handled before any on-sky data is even observed – bypassing the need to iterate until an error function threshold is reached.

There are several benefits to using ANNs for turbulence profiling. First, by running the ANNs on a GPU, trained networks can interpret a likely distribution of layer ranges from processed SLODAR data in a matter of seconds (Hamilton and Hart 2022). Second, since the networks interpret data quickly, the trained networks can be tested on a large secondary data set of unseen simulated atmosphere configurations to directly quantify expected system performance. This also makes it easy to check the dynamic range of the neural networks as a turbulence profiling algorithm before using them in the field.

There are downsides to using neural networks. The two primary issues are that training data is difficult and costly to obtain, and the trained networks are black box systems. While we can rationalize the use of black box algorithms for a specific application by validating performance on testing data, the difficult acquisition of training data can only be partially remedied. Real sensor data is ideal, but it can be hard to impossible to obtain. Obtaining data from real atmospheres would require weeks or months of data collection with absolute knowledge of the turbulence profile for each measurement frame. Instead, it is much more realistic to simulate a large and diverse set of atmospheres on a computer and generate synthetic training data from the results.

5.2 Data structures for turbulence profiling with neural networks

In Chapter (2) the theory behind turbulence profiling with SLODAR was presented, and a series of assumptions were provided to help manage the dimensionality of the problem. We saw that layer ranges can be found by finding correlations in measured tip/tilt vectors for different subaperture and subfield pairs. The layer velocity could then be found by also correlating the coefficients in time. Now that a proper set of simulation tools has been developed, we can model data and apply the developed theory to extract turbulence profiles.

The current constraints on the generalized SLODAR geometry are not sufficient to keep track of the many possible correlations between different elements in a simulated warp map. We are yet to define a shape for the warp maps because we will use the warp map elements to build two specific data structures – one for layer ranging and one for layer velocity estimation. Before specifying the data structures, it is important that we consider logistics in regards to generating, storing, reading, and processing the data in the context of neural networks.

Given that a sufficient training set has thousands to tens of thousands of points in it, the biggest cost of generating the training data is time. Having a fast computer and good programming can improve how quickly data generates, but the number of computations required to generate the layers, trace rays, and organize the data for each subfield-subaperture pair is large. This problem scales particularly badly with the system field of view, the number of subapertures in the pupil, and the number of subfields in the object plane.

One way to limit the number of required computations and speed up data generation is to simulate fewer measurement frames. Each $\mathbf{w}(t)$ contains (AB) independent measurements of both tip and tilt coefficients, where A is the number of subfields and B is the number of subapertures. The SLODAR measurements then correspond to the correlation of any two measurements in the warp map – giving us a grand total of (AB) choose 2 possible SLODAR measurements from a single warp map. Many of these combinations are independent measurements of the same layer, meaning they can be averaged together. If we can efficiently identify which warp map element correlations can

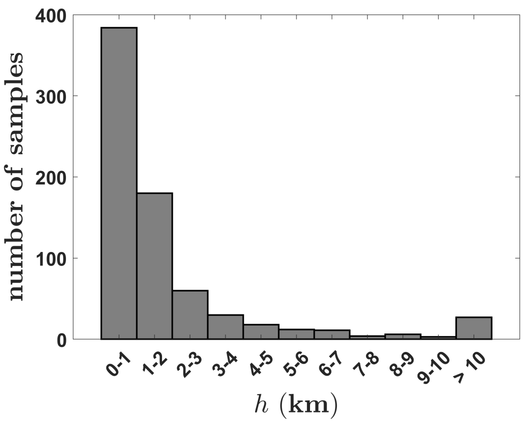


FIGURE 5.1: Histogram of SLODAR sample altitudes of the optical system with the properties of Tab. (5.1).

example system parameters	
D	1.5 m
$FFOV$	5'
θ	8.6"
$\sqrt{A}$	35
s	0.071 m
$\sqrt{B}$	21
h_m	0.05 km
H_M	34.1 km

TABLE 5.1: Parameters of the simulated SLODAR system used to generate example data for data structure visualization in this section

be averaged together, we can produce data structures which are highly averaged from only a few frames – requiring fewer simulation frames to produce high signal training data.

A lesser, but still important, logistical consideration for choosing data structures is matrix size. Large data structure take longer for neural networks to process, which negatively affects network training and data interpretation speeds. Thus, the data structures for ranging and velocity estimation should be dimensionally compact.

With these logistical considerations in mind, we are ready to develop the layer ranging and velocity estimation data structure. After defining one of the data structures, we will need to verify its behavior. To do so, we will simulate an optical system with the SLODAR sampling parameters in Tab. (5.1).

5.2.1 Layer ranging from correlated space-angle maps

Our goal is to process SHWFS data into a metric which can be interpreted by ANNs as the distance to layers of turbulence. As discussed in Section (2.2.3), SLODAR turbulence ranging works by correlating different subaperture and subfield tip and tilt coefficients, $\vec{\alpha}_{a,b}$ and $\vec{\alpha}_{a',b'}$. Peaks in tip/tilt correlation for subfield-subaperture separation vectors $(\Delta\vec{\theta}_{a,a'}, \Delta\vec{s}_{b,b'})$ correspond to the signal of a layer of turbulence at a distance h_ℓ following Eq. (2.11). To obtain a highly averaged estimation of the correlation with only a few measurement frames, we define an organization of the warp map elements called the space angle (SA) map.

SA maps are constructed by extracting elements from the system warp map and organizing them into a new matrix in which subfield separations are along rows and subaperture separations are along columns. Each SA map can only be formed using anti-parallel subaperture and subfield separation vectors to match our geometric requirements from Section (2.2.3). Therefore, each SA map is either constructed from subfields and subapertures both along the row direction of the pupil and object grids (to form vertical SA maps), or both along the column direction of the pupil and object grids (to form horizontal SA maps). Since the warp map contains both the tip and tilt coefficients for each subfield-subaperture, each tip SA map has identically constructed tilt SA map.

Consider the vertical SA maps constructed from tip coefficients. Start by taking the tip coefficient corresponding to column 1 of subapertures along subfield 1 and forming them into a $1 \times \sqrt{B}$ row vector. Then take the tip coefficients corresponding to the same column of subapertures but as seen along the next subfield in the column direction – i.e. angularly separated by θ – and form another $1 \times \sqrt{B}$ row vector. The new $1 \times \sqrt{B}$ vector is then appended to the first $1 \times \sqrt{B}$ vector row-wise, and the process is repeated for each subfield column $a \in [1, A]$. The end result is a SA map which is a $\sqrt{A} \times \sqrt{B}$ matrix with subaperture separations along the column direction and subfield separations along the row direction. This construction is then repeated for each column of subapertures, and then repeated for the tilt coefficients. The horizontal SA maps are constructed identically by using subaperture and subfield rows rather than columns. This process is shown diagrammatically for an $(A,B) = (9,4)$ system in Fig. (5.2)

Once the SA maps have been formed for each pupil row (column) j and each field row (column) i, notated as $\mathbf{S}^{(\mathrm{H})}_{i,j}(t)$ $\left(\mathbf{S}^{(\mathrm{V})}_{i,j}(t)\right)$, layer position information can be found by discretely convolving each matrix with itself. Using δ_b to represent an integer number of subaperture shifts along the space direction of the matrix and δ_a for an integer number of subfield shifts along the angle direction, the

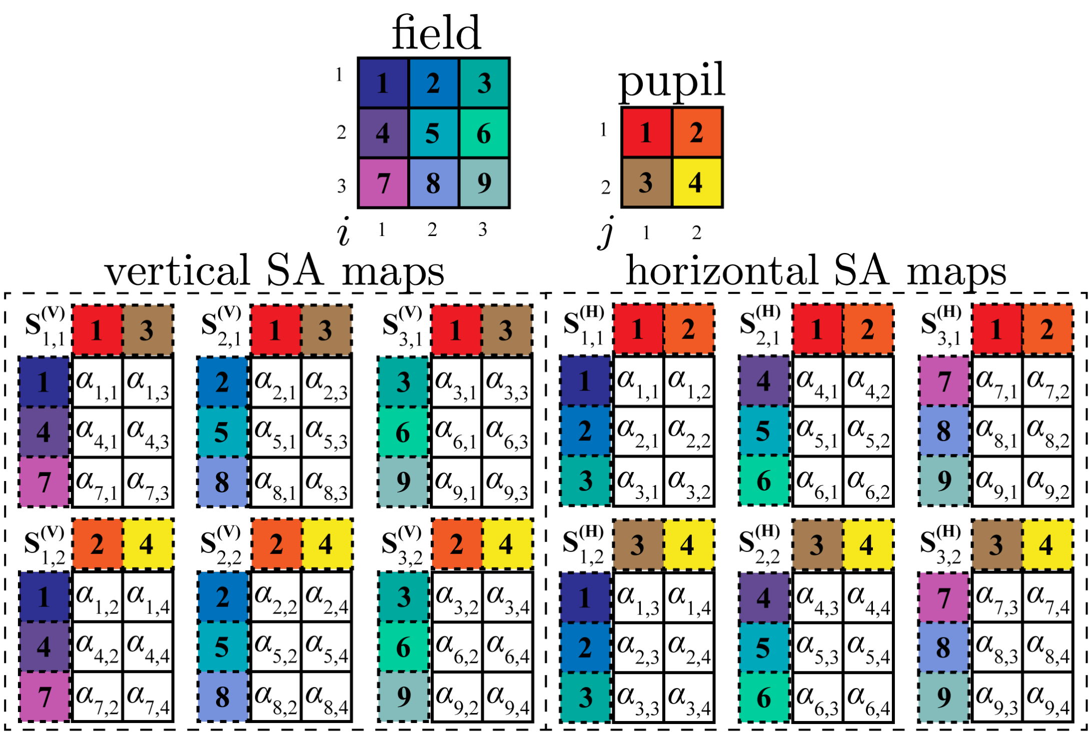


FIGURE 5.2: Illustration of all 3×2 SA maps which can be formed from a 3×3 field and 2×2 pupil. Each SA map matrix, $\mathbf{S}_{i,j}(t)$, is constructed of either a tip or tilt coefficient for the corresponding subfield-subaperture in the warp map matrix. The coefficients are organized such that the columns correspond to row (column) j in the pupil and rows correspond to row (column) i in the field.

SA correlation can be written as

$$\begin{aligned}\mathscr{C}_{i,j}(\delta_a,\delta_b;t) = &\frac{1}{(\sqrt{A}-\delta_a)(\sqrt{B}-\delta_b)} \\ &\times \sum\sum \mathbf{S}_{i,j}\left(\left[1:(\sqrt{A}-\delta_a)\right],\left[(1+\delta_b):\sqrt{B}\right];t\right) \\ &\qquad \odot\ \mathbf{S}_{i,j}\left(\left[(1+\delta_a):\sqrt{A}\right],1:\left[(\sqrt{B}-\delta_b)\right];t\right).\end{aligned} \tag{5.1}$$

We specifically formulate the SA map correlation such that the lines corresponding to a layer extend into quadrant one of a regular Cartesian grid. Note that due to our matrix notation, specified at the beginning of Chapter (3), this requires an inversion between the row and column shifting notation in Eq.(5.1). Also note that we can perform the convolution for the other three quadrants of Cartesian grid space – i.e. for $(\pm\delta_a, \pm\delta_b)$. Doing so provides more information about the symmetry, signal, and noise characteristics of $\mathscr{C}_{i,j}(\delta_a, \delta_b; t)$. However, in the context of visualizing and discussing the SA correlations it is not beneficial to show more than $(+\delta_a, +\delta_b)$ quadrant since it is the most intuitive to relate back to SLODAR measurement theory.

Each SA correlation, $\mathscr{C}_{i,j}(\delta_a, \delta_b; t)$, corresponds to the dot product between tip and tilt coefficients from different subfield-subaperture combinations. At the coordinate (δ_a, δ_b), the corresponding subfield-subaperture separation vectors have magnitude

$$|\Delta\vec{\theta}_{a,a'}| = \theta\delta_a, \qquad |\Delta\vec{s}_{b,b'}| = s\delta_b. \tag{5.2}$$

Looking at Eq. (2.11), Eq. (5.1), and Eq. (5.2), we can deduce that layers will appear in the SA correlation matrix as lines starting at the origin with slope $h_\ell = (s\delta_b)/(\theta\delta_a)$. Thus, to determine the number of layers and their ranges, the SA correlations need to be analyzed to find the number of lines going through the origin as well as the slopes of those lines. Before searching for the lines and their slopes, we can identify all SA maps which can be averaged together to boost signal while using fewer simulation frames.

The construction of the SA maps produce correlations with identical h measurements at each SA correlation matrix pixel. This is true for the vertical and horizontal subfield separation constructions for both tip, α_2, and tilt, α_1. With each pixel being an independent measurement of the same thing, we can obtain a higher degree of averaging by computing

$$\mathbf{C}_{i,j}(\delta_a, \delta_b; t) = \frac{\mathscr{C}^{(\mathrm{H})}_{i,j}(\alpha_1; t) + \mathscr{C}^{(\mathrm{V})}_{i,j}(\alpha_1; t) + \mathscr{C}^{(\mathrm{H})}_{i,j}(\alpha_2; t) + \mathscr{C}^{(\mathrm{V})}_{i,j}(\alpha_2; t)}{4}. \tag{5.3}$$

Also notice that since we setup each SA map to have angular sampling in the column direction and spatial sampling in the row direction, regardless of the subaperture or subfield orientation, each matrix $\mathbf{C}_{i,j}(\delta_a, \delta_b; t)$ contains an independent measurement of the same thing at each matrix element.

Thus, we can average all $\mathbf{C}_{i,j}$ matrices together for all $\sqrt{AB}$ combinations of (i,j):

$$\tilde{\mathbf{C}}(\delta_a,\delta_b;t) = \frac{1}{\sqrt{AB}} \sum_{i=1}^{\sqrt{A}} \sum_{j=1}^{\sqrt{B}} \mathbf{C}_{i,j}(t) \tag{5.4}$$

– resulting in $4\sqrt{AB}$ points being averaged per matrix element. The last option for averaging is in time. For simulation frames indexed $n_t = [1, N_T]$ we compute

$$\bar{\mathbf{C}}(\delta_a,\delta_b) = \frac{1}{N_T} \sum_{n_t=1}^{N_T} \tilde{\mathbf{C}}(\delta_a,\delta_b;t_{n_t}). \tag{5.5}$$

The construction of the SA maps leads us to a final SA correlation which is dimensionally compact, consisting of only $\sqrt{A} \times \sqrt{B}$ elements, and is highly sampled, with $4N_T\sqrt{AB}$ independent correlations averaged together per measurement.

Multi-layer SA map demonstration

Two example atmospheres were modeled to visualize and verify the layer ranging properties of the averaged SA correlation data structure. The properties of each atmosphere are given by Tab. (5.2), and the simulation results for the example system are shown in Fig. (5.3). The presented SA correlations are a result of averaging $\tilde{\mathbf{C}}(\delta_a,\delta_b;t)$ over simulation times $t = [0, 25, 50, 75, 1000]$ ms.

three layer atmosphere	
h (km)	[0.05, 5, 15]
r_0 (m)	[0.25, 0.3, 0.5]
SNR	[>5, 4.42, 1.37]
pupil r_0 (m)	0.18

six layer atmosphere	
h (km)	[0.05, 0.15, 0.5, 5, 7.5, 15]
r_0 (m)	[0.25, 0.3, 0.3, 0.4, 0.3, 0.5]
SNR	[2.41, 1.04, 2.51, 1.23, 1.14, 0.52]
pupil r_0 (m)	0.11

TABLE 5.2: Parameters for the two example atmospheres simulated to visualize the time-averaged SA correlation data metric for layer ranging. Layers 1, 4, and 6 of the six layer atmosphere are the same as layers 1, 2, and 3 of the three layer atmosphere, respectively. The layer SNRs were calculated using the concepts from Chapter (4), the values contained in this table, and values from Tab. (5.1).

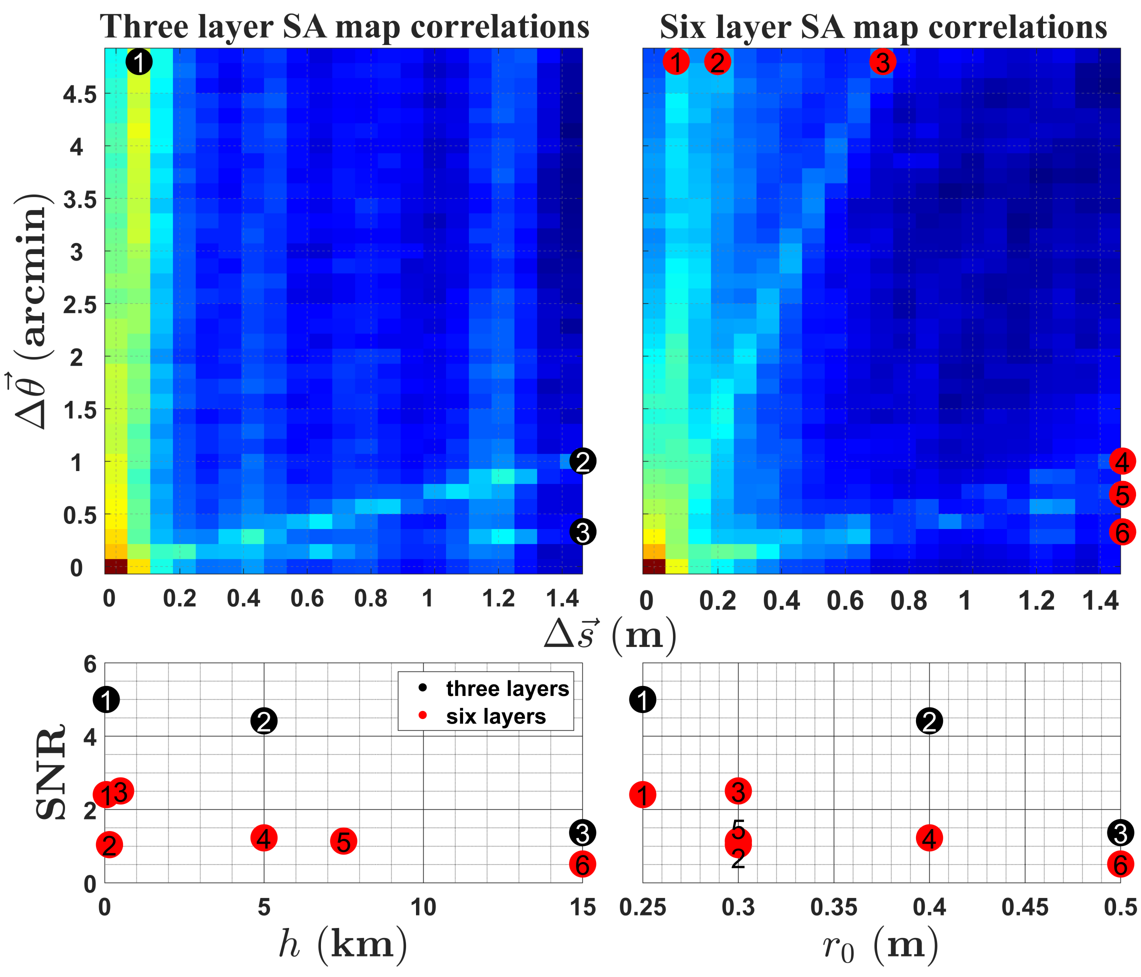


FIGURE 5.3: Top: SA correlations from three layer (left) and six layer (right) atmospheres averaged over 1 s of simulated WFS data. The layers produce lines with slope $(s\delta_b)/(\theta\delta_a) = \Delta\vec{s}/\Delta\vec{\theta}$, corresponding to the layer distance h_ℓ. The layers are labeled at the edge of each matrix. Bottom: The SNR of each layer plotted as a function of layer distance (left) and layer Fried length (right). We can see that layers with low SNR produce weaker lines in the SA map correlation. We can also see that the only layer with SNR < 1, which is layer 6 of the six layer atmosphere, is nearly indistinguishable from the noise component of layers 1, 2, 3 and 5. SNR > 5 were assigned a value of 5 to help visualize the data.

The behavior of the SA correlation for both the three and six layer atmospheres in Fig. (5.3) match the theoretical behavior. The lines in each SA correlation have the correct slopes given by $(s\delta_b)/(\theta\delta_a) = \Delta\vec{s}/\Delta\vec{\theta} = h_\ell$, and there are as many lines passing through the origin as there are layers of modeled turbulence.

By making three of the layers in the six layer atmosphere identical to all layers in the three layer atmosphere, we can see how the number of layers affects SNR. It is clear that as the number of layers increases, the SNR of each layer decreases. This is most obvious for layer 1. Even though layer 1 is identical in both atmospheres, the SNR of the layer is much lower in the six layer atmosphere. We can also see that the high altitude layer has SNR < 1 in the six layer system. For this layer there is some indication of the layer signal, but it is visually ambiguous with the noise from layers 1, 2, 3, and 5. From this, we can assume that layers with SNR ≈ 1 might be measurable but layers with SNR $<< 1$ will likely be undetectable. If a layer is undetectable it will not be learnable by the networks, indicating that it should be treated as noise rather than an actual layer.

5.2.2 Layer wind velocities from correlated space-time maps

Finding the velocity of a layer requires tracking measurements of high tip/tilt correlation across the spatial, angular, and temporal dimensions. Since the velocity is also a vector quantity, the orientation of the spatial components has to be maintained to measure the wind direction. To account for this, we define a new data structure called the space time (ST) map.

Each ST map is laid out identically to the shape of the WFS pupil where each element represents a subaperture tip/tilt coefficient from the same subfield – making it a $\sqrt{B} \times \sqrt{B}$ matrix. Therefore, the axes of the ST maps match the coordinate axes of the system pupil. Each ST map at each measurement time $[1,T]$ is stacked together along the third matrix dimension to produce a $\sqrt{B} \times \sqrt{B} \times T$ matrix. The process is repeated for each subfield $a \in [1,A]$.

To determine the layer velocities, we correlate pairs of ST maps from different subfields (a,a'), with discrete 3D convolutions. There are $A!/(2!(A-2)!)$ combinations of (a,a'), resulting in many different values of $|\Delta\vec{\theta}_{a,a'}|$. Each resulting $|\Delta\vec{\theta}_{a,a'}|$ will have varying levels of identical but independent measurements from different subfield-subaperture pairs, meaning that some ST correlations will be better averaged than others. Each additional ST correlation will take time to compute, making it worthwhile to limit the scope of ST maps which are generated and discretely convolved to values of (a,a') with known sufficient averaging. Simplifying the number of possibilities also makes the process of bookkeeping the correlations easier.

The fist simplification is the row-only or column-only subfield separation requirement that was first established in Chapter (2). This means we will only convolve ST maps from subfields which are both either separated in the horizontal or vertical directions. The second simplification comes from recognizing that convolving the ST maps for the subfield pair (a,a') and also (a',a) is redundant and can be ignored. Together, these two simplifications result in $A(\sqrt{A}-1)/2$ combinations of (a,a') corresponding to ST map pairs, S_a and S'_a, which must be correlated.

Once we have formed each ST map and identified the pairs (a,a') corresponding to subfields that meet our requirements, the ST correlations are computed. For the correlation in the $(+x,+y)$ quadrant, the discrete convolution can be written as

$$\begin{aligned}\mathscr{C}^{(+,+)}_{a,a'}(\delta_y,\delta_x,\delta_t) =& \frac{1}{(\sqrt{A}-\delta_a)(\sqrt{B}-\delta_b)(T-\delta_t)} \\ &\times \sum\sum\sum \mathbf{S}_a\left(\left[1:(\sqrt{B}-\delta_y)\right],\left[(1+\delta_x):\sqrt{B}\right],\left[(1+\delta_t):T\right]\right) \\ &\odot\ \mathbf{S}_{a'}\left(\left[(1+\delta_y):\sqrt{B}\right],\left[1:(\sqrt{B}-\delta_x)\right],\left[1:(T-\delta_t)\right]\right).\end{aligned} \tag{5.6}$$

The matrix indexing is slightly different for the correlations in the other four quadrants but can be formed similarly. Performing the convolution in all four quadrants for a subfield pair (a,a') and tiling them together results in the full ST correlation matrix $\mathscr{C}_{a,a'}(\delta_y,\delta_x,\delta_t)$, which is $(2\sqrt{B}-1)\times(2\sqrt{B}-1)\times T$.

Each $(2\sqrt{B}-1)\times(2\sqrt{B}-1)$ sub-matrix has a vertical axis corresponding to $\Delta\vec{s}_y = s\delta_y$ and a horizontal axis corresponding to $\Delta\vec{s}_x = s\delta_x$. In the zero time delay case, ST correlations from vertically separated objects will produce layer signals along the $\Delta\vec{s}_y$ axis. Similarly, horizontally separated subfield ST correlations will have layer signals distributed along the $\Delta\vec{s}_x$ axis. This is because the subfield-subaperture separation vectors must be anti-parallel to have perfect overlap at a layer of turbulence (see Section (2.2.3)). Since the angular separation vector $|\Delta\vec{\theta}_{a,a'}|$ is fixed for each ST correlation, the position of each layer with distance h_ℓ along $\Delta\vec{s}_y$ or $\Delta\vec{s}_x$ for zero time delay changes between different ST correlations following Eq. (2.11).

ST correlations with zero time delay are actually a suitable data structure for turbulence ranging. However, SA map correlations are more dimensionally compact and are therefore more efficient to store and faster to analyze. SA map correlations are also organized in a way enhances the amount of averaging each distance measurement receives, making it an overall better choice for ranging. Another benefit of using SA maps for layer ranging is that if the distances to layers are already known, the starting position of each layer in each ST correlation is also known – making it easier to theorize a method for velocity estimation than if the starting position of the layer could be anywhere.

While the layer altitude information is stored in the zero time delay ST correlation, the velocity information can only be estimated by analyzing the matrices along the time delay axis. As discussed in Chapter (2), layer signals in SLODAR measurements are due to overlapping projections of different subapertures along different subfields at the same layer of turbulence. We then showed in Sections (2.2.4-2.2.5) that introducing a time shift to one of the tip/tilt coefficient measurements is analogous to sliding the projection of the time delayed subaperture at the layer by $\Delta\vec{x} = \vec{v}\Delta t$. Because of the FFH hypothesis we can deduce that the tip/tilt correlation coefficient will no longer match with the zero time delay case since they no longer correspond to overlapping regions in the measurement layer. Instead, the layer signal will move from the zero time delay coordinate by $\Delta\vec{x}$ to a new coordinate in the ST correlation matrix. If we can determine $\Delta\vec{x}$ for each layer, we can back out the layer wind velocities.

From the $A(\sqrt{A}-1)$ different $\mathscr{C}_{a,a'}(\delta_y, \delta_x, \delta_t)$ matrices, we can boost signal by averaging all ST correlations which are identical and independent measurements. Within the $A(\sqrt{A}-1)$ total subfield combinations, half of them correspond to measured tip coefficients and the other half are for tilt coefficients. Within the tip coefficient ST correlations, half of them are for vertically separated subfields and the other half are for horizontally separated subfields. As mentioned earlier, the vertical and horizontal subfield ST correlations have a different mapping of the layer signal positions which means we cannot average them together. However, the tip and tilt coefficients from either the vertically or horizontally separated fields are independent identically ordered measurements and can be averaged:

$$\mathbf{C}^{(\mathrm{H})}_{a,a'} = \frac{\mathscr{C}^{(\mathrm{H})}_{a,a'}(\alpha_1) + \mathscr{C}^{(\mathrm{H})}_{a,a'}(\alpha_2)}{2}, \tag{5.7}$$

$$\mathbf{C}^{(\mathrm{V})}_{a,a'} = \frac{\mathscr{C}^{(\mathrm{V})}_{a,a'}(\alpha_1) + \mathscr{C}^{(\mathrm{V})}_{a,a'}(\alpha_2)}{2}. \tag{5.8}$$

This results in $A(\sqrt{A}-1)/4$ vertical and horizontal object separation ST correlation matrices each. Within each set of $A(\sqrt{A}-1)/4$ ST correlations there are only $a_u \in [1, \sqrt{A}-1]$ unique subfield angular separations, $\Delta\vec{\theta}_{a_u}$. We can average all ST correlations with identical angular sampling since they all have the same layer sampling coordinates:

$$\tilde{\mathbf{C}}^{(\mathrm{H})}_{a_u} = \frac{\sum_{(a,a')\in a_u} \mathbf{C}^{(\mathrm{H})}_{a,a'}}{A - a_u\sqrt{A}}, \tag{5.9}$$

$$\tilde{\mathbf{C}}^{(\mathrm{V})}_{a_u} = \frac{\sum_{(a,a')\in a_u} \mathbf{C}^{(\mathrm{V})}_{a,a'}}{A - a_u\sqrt{A}}. \tag{5.10}$$

We organize the unique angle indices, a_u, from smallest to largest sampling angle so that we can use system geometry to know there are $(A - a_u\sqrt{A}\ )$ ST correlations corresponding to $\Delta\vec{\theta}_{a_u}$. The dimension for the horizontal or vertical angle-averaged ST correlations is then $(2\sqrt{B}-1)\times(2\sqrt{B}-1)\times T\times(\sqrt{A}-1)$, where each matrix corresponds to a unique angle sample a_u and is the result of average $2(A - a_u\sqrt{A})$ independent measurements.

The angle-averaged ST correlations can still run into noise issues due to insufficient averaging – making it particularly hard to track the velocity of weak layers. To reach sufficient sampling such that the random uncorrelated noise component in each measurement disappears, giving us performance limited by the theoretical SNR of each layer, we can also introduce time averaging. The ST correlations are correlated in time, meaning the individual time samples cannot be averaged together. Instead, the ST correlation with T time steps can be computed at N_T different start times. Since the ST correlations are a function of δt, changing the start time supplies us with independent measurements of the same values. Thus, the time averaged ST correlation can be written as

$$\bar{\mathbf{C}}^{(\mathrm{H})}_{a_u} = \frac{1}{N_T} \sum_{n_t=1}^{N_T} \tilde{\mathbf{C}}^{(\mathrm{H})}_{a_u;n_t}, \tag{5.11}$$

$$\bar{\mathbf{C}}^{(\mathrm{V})}_{a_u} = \frac{1}{N_T} \sum_{n_t=1}^{N_T} \tilde{\mathbf{C}}^{(\mathrm{V})}_{a_u;n_t}. \tag{5.12}$$

The time averaged ST correlation matrices for horizontal and vertically separated subfield pairs, $\bar{\mathbf{C}}^{(\mathrm{H})}$ and $\bar{\mathbf{C}}^{(\mathrm{V})}$ respectively, are each $(2\sqrt{B}-1) \times (2\sqrt{B}-1) \times T \times (\sqrt{A}-1)$ and are the final ST correlation data structure for velocity estimation.

Multi-layer ST map demonstration

Having developed the data metric to be used for velocity estimation, we can use the simulation environment to generate synthetic atmospheres and the resulting ST correlations. We used a three layer atmosphere defined by the parameters in Tab. (5.3) as seen by the example system defined by the parameters in Tab. (5.1). The warp maps were calculated for $T = 10$ time steps with step size $\delta t = 10$ ms at $N_T = 10$ different time windows. Each time window starts 1 s after the previous. The results are shown in Fig. (5.4), Fig. (5.5), and Fig. (5.6).

	h (km)	r_0 (m)	$\vec{v}$ (m/s)	L_0 (m)	SNR
layer 1	0.05	0.2	[10,0]	10	>5
layer 2	1	0.2	[-10,-5]	50	>5
layer 3	5	0.2	[10,-10]	250	2.83

TABLE 5.3: Parameters for the three layer atmosphere simulated to generate warp maps for demonstrating the ST correlation data structure. The layers were all given identical strength to help improve the visibility of each layer.

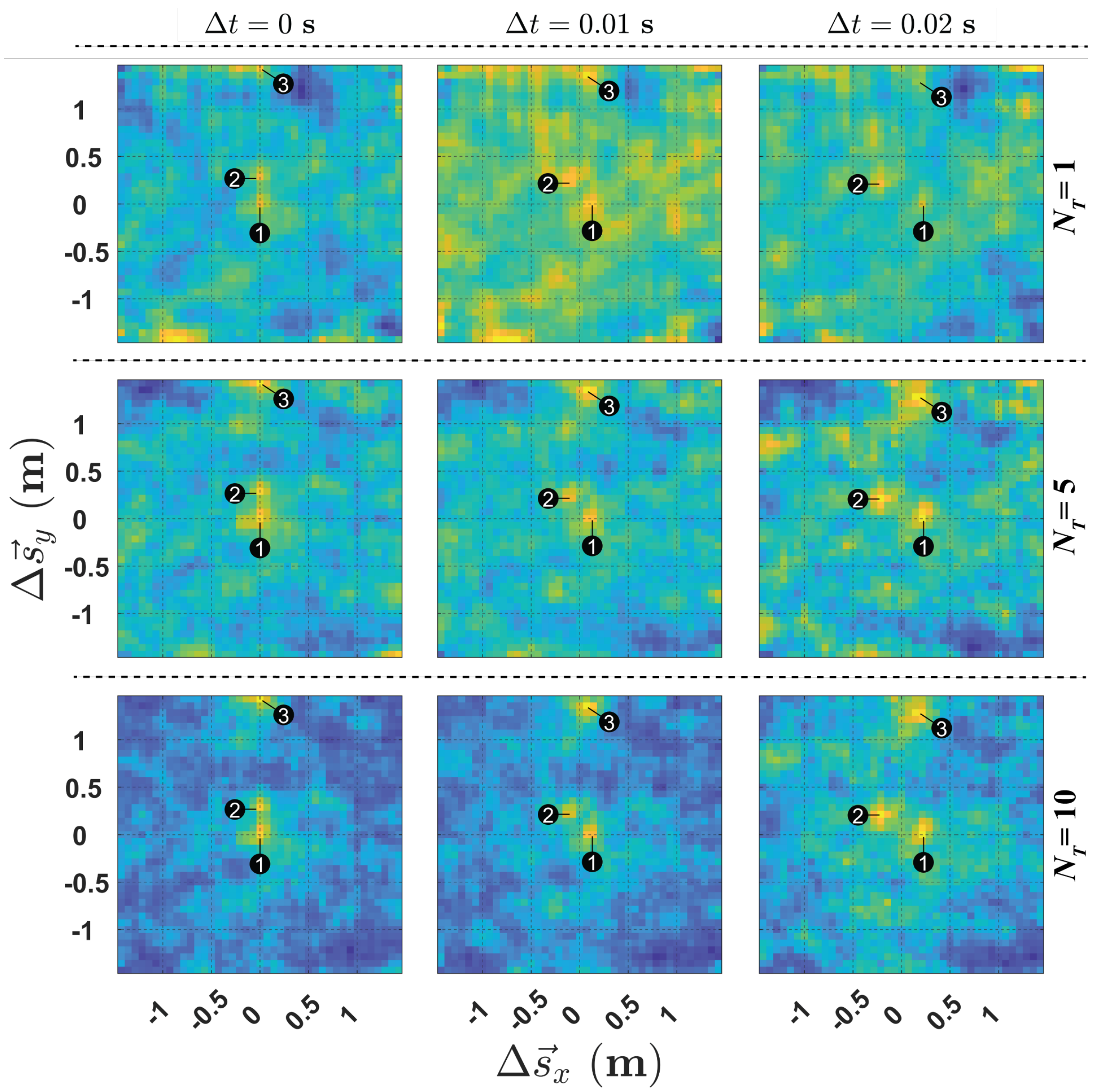


FIGURE 5.4: Demonstration of different amounts of time averaging for the vertical ST correlation data structure computed with an object separation of 1' from the three layer atmosphere given by Tab (5.3). The top row shows time delays of $\Delta t = [0, 10, 20]$ ms for the ST correlation without any time averaging – i.e. $\tilde{\mathbf{C}}_{1'}^{(\mathrm{V})}$. The second and third rows show $\bar{\mathbf{C}}_{1'}^{(\mathrm{V})}$ for the same system when averaged over $N_T = 5$ and $N_T = 10$ time windows, respectively. A delay of 1 second was used between each time window. From this, we can see that the random noise speckles that appear in the ST correlations without any time averaging gradually disappear as more time window averaging is introduced. However, even in the with strong random noise, the layer signals are recognizable.

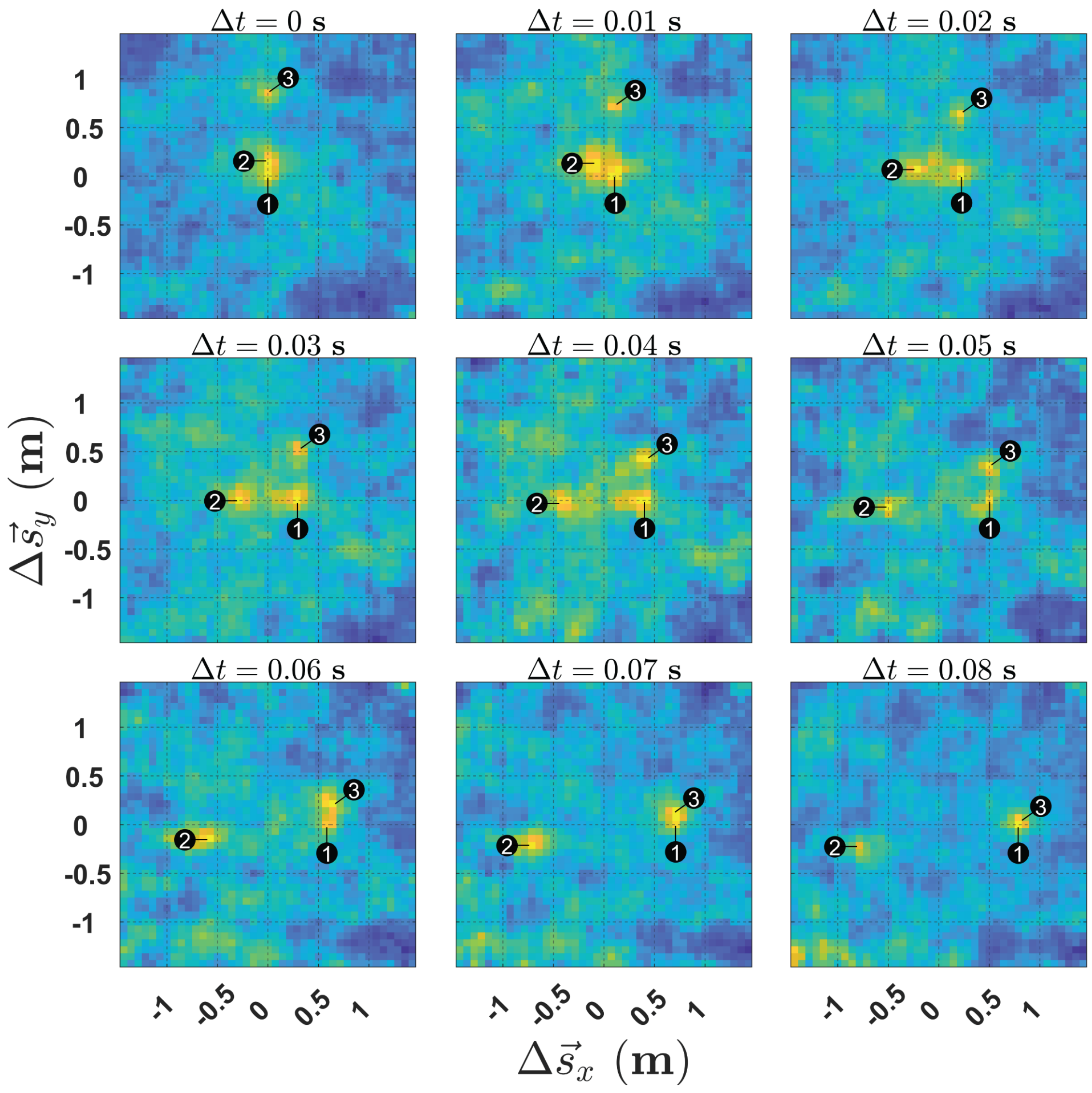


FIGURE 5.5: Demonstration of $\bar{\mathbf{C}}^{(\mathrm{V})}_{0.571'}$ for $\Delta t \in [0, 80]$ ms using the three layer atmosphere in Tab. (5.3) as seen by the example system in Tab. (5.1). The ST correlations were computed by averaging $N_T = 10$ time windows. Each of the three layers start at the expected position along the vertical axis and move with the same magnitude and direction as the layer wind vectors. This subfield separation was chosen to highlight how there are matrices where the layer signals overlap

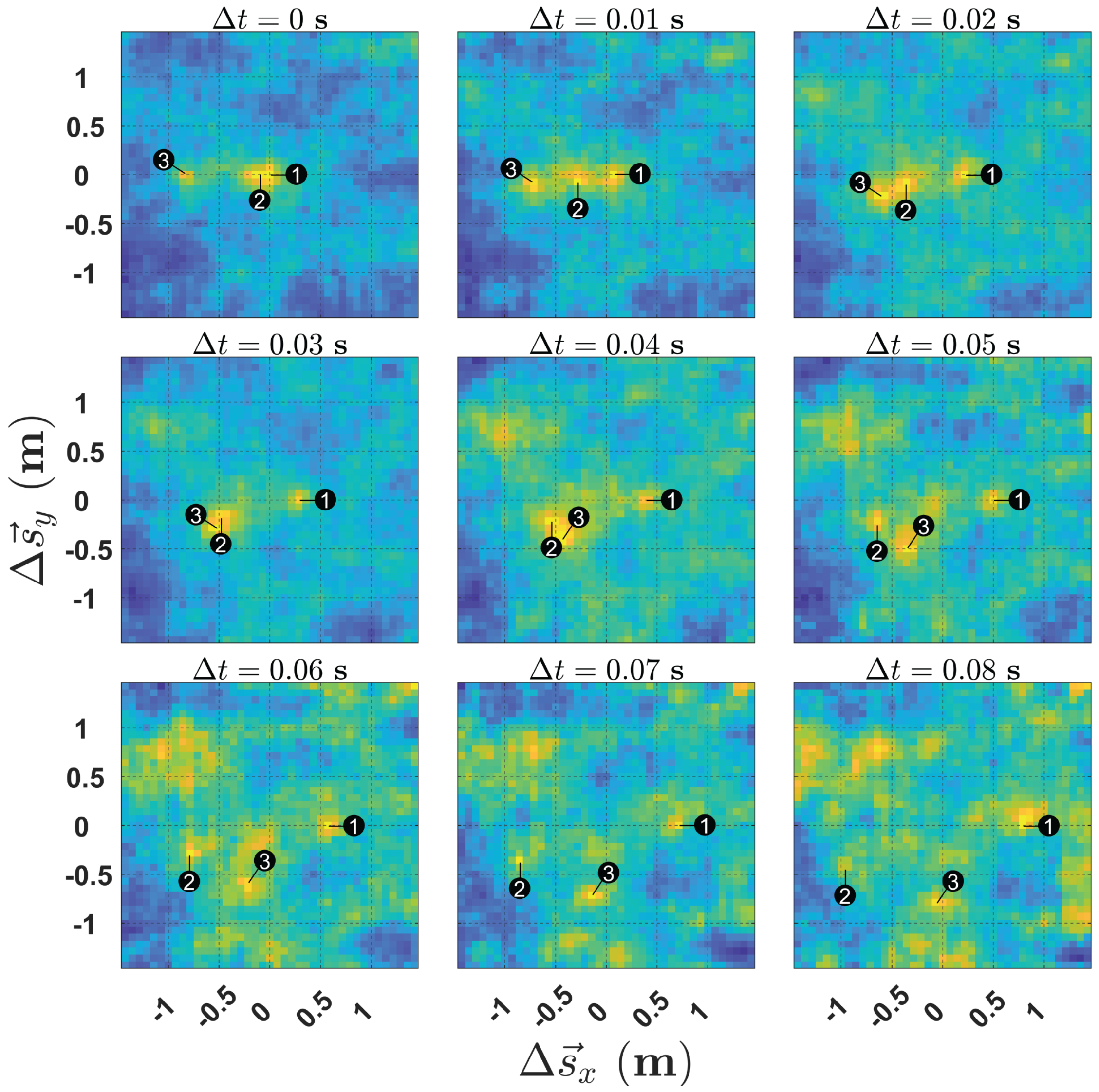


FIGURE 5.6: Demonstration of $\bar{\mathbf{C}}^{(\mathrm{H})}_{0.571'}$ at $\Delta t \in [0, 80]$ ms for the three layer atmosphere in Tab. (5.3) as seen by the example system in Tab. (5.1). This is the horizontal subfield separation counterpart to Fig. (5.5) and provides the same insight into ST correlation behavior. We do find that the horizontal ST correlations contain more random noise than the vertical counterpart in this case. We suspect this has to do with the relative tip and tilt strength of the layers. In the presence of weak tilt, the horizontally separated subapertures will have a weaker magnitude tip/tilt correlation and thus a weaker layer signal relative to random noise levels.

In each ST correlation example figure, the location of the layer signals for $\Delta t = 0$ are in agreement with Eq. (2.11). The layer signals appear along the $\Delta\vec{s}_y$ axis for vertically separated subfields and along the $\Delta\vec{s}_x$ axis for horizontally separated subfields as expected. As the time lag increases, the spots indicating the layer signals move with the direction and magnitude of the layers which confirms the theory developed in the previous section. We see that increasing the number of time windows over which the angle-averaged ST correlations are averaged suppresses random noise making the layer signals more apparent. We also see that as the time lag increases, fewer samples are being averaged and the speckling due to random noise increases. Unexpected, the horizontal and vertical matrices have different amounts of random noise. We suspect that depending on if the layers are tip or tilt dominant, either the horizontal or vertical ST correlation willbe noisier under identical averaging.

5.3 Training data generation

5.3.1 Preparing to generate training data

Properly function ANNs should be able to identify all features of interest in data which was not observed during training. To obtain properly functioning turbulence profiling networks trained on simulated data, three concepts which have not been sufficiently developed yet require further discussion:

1. the training data must sufficiently represent the atmosphere within the measurement dynamic range of the modeled system,
2. the distribution of layer distances and strengths must be organized in such a way as to avoid bias during training,
3. simulated layers must be above a minimum SNR to be measured.

Sufficiently representing the atmosphere

To address the dynamic range issue of topic 1, we can use system geometry and the theory developed in Chapter (2). From the number of subfields A within the system FFOV, the subfield separation angle θ can be determined. Similarly, from the number of subapertures B within the WFS pupil size D the subaperture separation distance s is known. The set (A,θ,B,s) is sufficient to calculate every distance, as well as the minimum and maximum velocities, that the modeled SLODAR system can measure. This set the dynamic range of the optical system, which is by definition the domain over which simulated layers should be bounded.

Having defined the dynamic range, we must now specify a way to ensure that the training data is sufficiently representative of the atmosphere. The most comprehensive training data set for the system would contain all possible data points within the dynamic range. However, this is an unrealistic task to attempt since the atmosphere has infinitely many configurations. Instead, we focus on generating as few training data points as possible while still being representative of the dynamic range of the system to produce high performance networks. What constitutes 'as few training data points as possible' is highly dependent on the features that we are trying to learn. For example, determining if there is or is not turbulence anywhere in the atmosphere is a much easier feature to learn than the velocity of a layer at 10 km. To properly learn a feature, a few data points for each possible presentation of the feature should be supplied during training. By identifying each model parameter change that will alter the signal corresponding to a particular feature, an iterative process can be setup which generates a minimum of tens of data points for each feature for each parameter at a particular setting within the system measurement domain.

A major influence parameter on the feature signal for SLODAR measurements is the number of sample distances. To better understand this, let us compare two systems with different sampling: an $(A,B)=(2,2)$ system and an $(A,B)=(20,20)$ system. $(A,B)=(2,2)$ cannot measure as many discrete distances as a system with $(A,B)=(20,20)$. Thus, the $(A,B)=(20,20)$ system will require more training data to discriminate between its more numerous features than the $(A,B)=(2,2)$

system. The $(A,B) = (20,20)$ SLODAR system will also have many more instances of redundant or closely spaced distance measurements. Because of the relationship between layer separation and layer SNR, closely separated layers can present as a single layer in the measurement data. Our solution to this is to bin sets of adjacent SLODAR sample distances together so that closely separated layers correspond to the same feature.

Binning solves two additional problems in regards to feature discrimination: layers which are between SLODAR sample distances are captured in one feature, and the number of features to learn is consolidated. Binning the measurement distances also increases the number of samples in the data corresponding to the feature of interest which makes learning the feature easier. Going back to the example SLODAR system with $(A,B) = (20,20)$, there are $(A-1)(B-1) = 361$ measurement distances. If we bin the atmosphere into 20 distance bands with equal sampling, each bin could consist of up to ~ 18 measurements. Learning 20 features corresponding to ~ 18 measurements is an easier task than learning 361 features each corresponding to a single measurement – especially when there is varying levels of measurement cross-talk for certain features.

The last parameter which changes feature signal is the number of layers, L. For example, SA maps collected from atmospheres with a different number of layers will have a different number of lines going through the origin. We also know that each layer acts as a source of noise to all other layers, meaning that the layer SNR will generally decrease with increasing L. Thus, if we expect something like $L \in [2,10]$ layers of turbulence to exist out in front of our system, multiple training data points for each feature should be generated for $L = 2,3,...,10$ atmosphere. Since each additional layer will increase the amount of time it takes to calculate the associated warp map, we will limit the discussion here to atmospheres with $L \in [1,4]$.

Obtaining an unbiased distribution of training data

When training neural networks it is important that the training data is unbiased. For a binary classification task, such as layer present/not present, a truly non-biased training set would consist of approximately 50% layer present data and 50% layer not present data. As the number of features increases, the distribution of data points with each feature needs to stay approximately equal to avoid networks which learn to randomly guess or always say one feature is present. In terms of addressing topic 2, preparing to generate training data for turbulence profiling from SLODAR measurements, an important potential source of sample bias is the simulated distribution of layer distances.

As discussed in Section (2.2.3), SLODAR systems have non-uniform distance sampling. For the example system defined by Tab. (5.1), the non-uniformity can be seen in the histogram of sample distances plotted in Fig. (5.1). Notice that while the system has a maximum sampling distance of ~ 30 km, only a few of the SLODAR measurements correspond to altitudes > 10 km. If we were to generate points with distances uniformly distribution over the dynamic range of the system, $\sim 2/3^{\text{rds}}$ of the layers would have signals in the few measurements corresponding to distances > 10 km. If all available training data generation time for a project was spent simulating this distribution of layers, high altitude measuring networks would have a large training data set and low altitude measuring networks would have a small training data set which will produce low performance networks. Even worse, if all simulated training data is used to train a network to find a specific layer which consists of a small percentage of the data, the network will most likely learn to say that there is never a layer at that altitude. Therefore, the most efficient distribution for layer distances when generating training data is one which is uniformly distributed in the SLODAR sample bins. Since the features we seek to teach the ANNs are specific to each altitude bin, this ensures there are an equal number of training data points for each feature.

Conditioning layer SNR

As we saw in Fig. (5.3), layers with an SNR < 1 are potentially indistinguishable from the noise component of other layers. When training ANNs, we make adjustments to the response of the network for a given input based on the difference between its output and the known answer. If there is a layer which is far below the noise floor, and therefore cannot be seen in the input data structure, we will penalizing a network for not detecting something we say is there but it will never be able to detected. This generally results in slower training that could potentially not converge. Thus, if time is being taken to generate a training data set it is important that the chosen atmospheres do not have layers with SNR $<<1$. By conditioning layer strengths to meet a minimum SNR threshold, we can ensure simulation time is not wasted making bad training data which might not produce functioning networks.

5.3.2 Training data generation program

Having recognized what features we are looking for in the data, as well as what parameters influence the features, we can program a simulation to iteratively generate a data set to train turbulence profiling neural networks. The proposed data generation program consists of three functional blocks: the SLODAR model, the atmosphere model, and data simulation.

The SLODAR model block initializes all of the altitude sampling and raytrace parameters that will be needed during simulation. This also tells us the dynamic range of simulated atmospheres. From the dynamic range and altitude sampling we can define the binned SLODAR sampling.

The atmosphere model block uses the binned SLODAR sampling to produce training atmospheres which are in the measurement dynamic range of the SLODAR system and not biased to particular measurement altitudes. Each modeled atmosphere has its first layer generated inside one of the bins. Each subsequent layer $\ell \in [2, n_L]$ is then generated in a different altitude bin. The layer strengths are either randomly selected below a threshold or assigned a pseudo-random value based on the altitude of the layer and a modeled $C_n^2(h)$ profile. A similar approach can be done to select each layer velocity and outer scale. Once all values of h, r_0, $\vec{v}$ and L_0 are selected for a particular atmosphere, the layer SNRs are computed. If there are layers with SNR below the conditioning threshold, the layer strengths are continually adjusted until all layer SNRs are within the desired threshold. The process is repeated for N_a different atmospheres for all altitude bins $z \in [1, Z]$, and number of layer atmospheres $n_L \in [m_L, N_L]$. This produces a total of $N_a Z(N_L - m_L + 1)$ training atmospheres. The atmosphere model block then saves each atmosphere as a collection of layer parameters $(h, r_0, \vec{v}, L_0, \mathrm{SNR})_\ell \; \forall \; \ell \in [1, L]$.

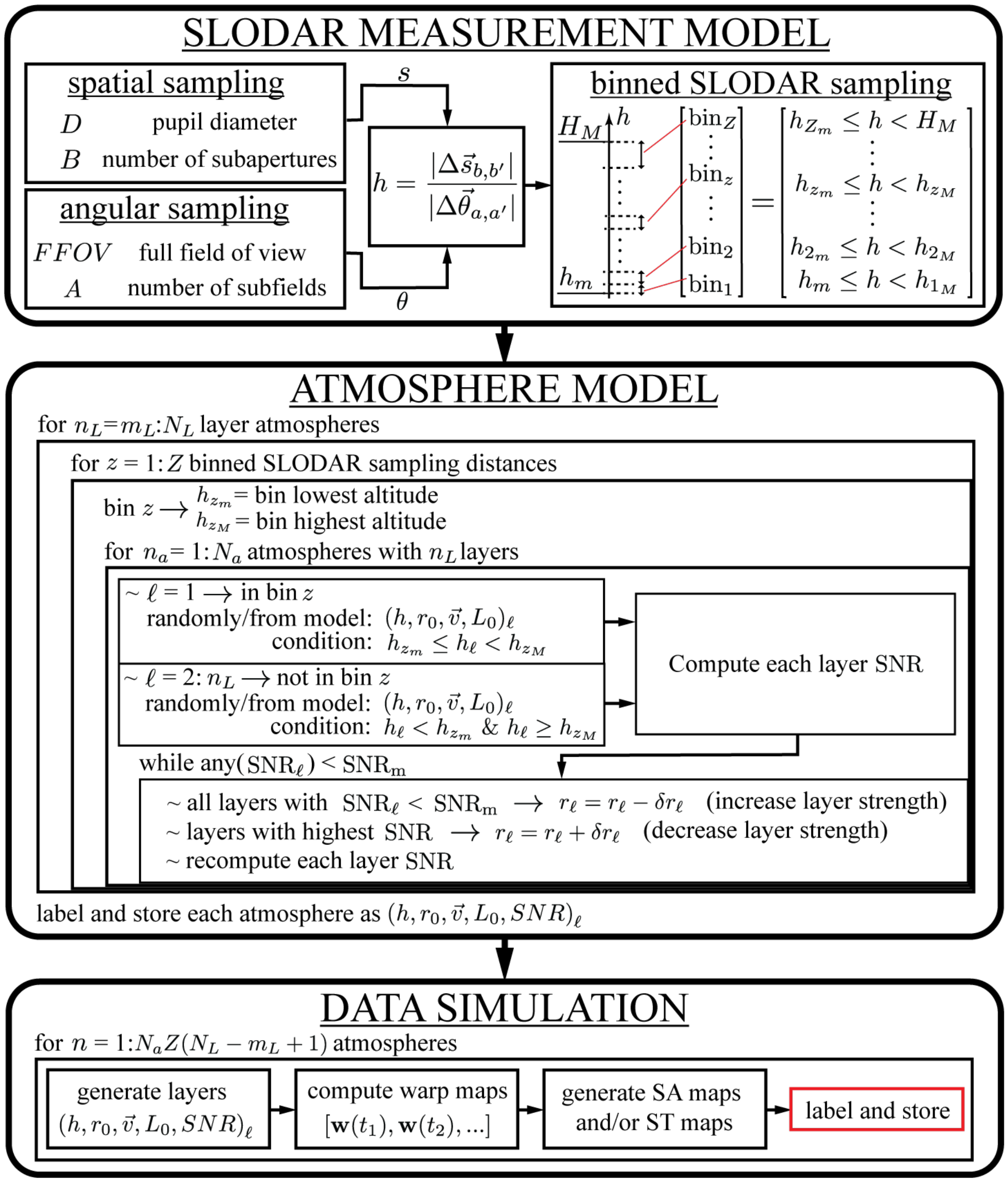

SLODAR MEASUREMENT MODEL
spatial sampling
D pupil diameter
B number of subapertures
angular sampling
$FFOV$ full field of view
A number of subfields
s
θ
$h = \frac{|\Delta\vec{s}_{b,b'}|}{|\Delta\vec{\theta}_{a,a'}|}$
binned SLODAR sampling
H_M
h
h_m
bin_Z
bin_z
bin_2
bin_1
$h_{Z_m} \leq h < H_M$
$h_{z_m} \leq h < h_{z_M}$
$h_{2_m} \leq h < h_{2_M}$
$h_m \leq h < h_{1_M}$
ATMOSPHERE MODEL
for n_L=m_L:N_L layer atmospheres
for z = 1:Z binned SLODAR sampling distances
bin $z \longrightarrow$ h_{z_m}= bin lowest altitude
h_{z_M} = bin highest altitude
for n_a= 1:N_a atmospheres with n_L layers
~ ℓ = 1 $\longrightarrow$ in bin z
randomly/from model: $(h, r_0, \vec{v}, L_0)_\ell$
condition: $h_{z_m} \leq h_\ell < h_{z_M}$
~ ℓ = 2: n_L $\longrightarrow$ not in bin z
randomly/from model: $(h, r_0, \vec{v}, L_0)_\ell$
condition: $h_\ell < h_{z_m}$ & $h_\ell \geq h_{z_M}$
Compute each layer SNR
while any(SNR_ℓ) < SNR_m
~ all layers with $\text{SNR}_\ell < \text{SNR}_\text{m}$ $\longrightarrow$ $r_\ell = r_\ell - \delta r_\ell$ (increase layer strength)
~ layers with highest SNR $\longrightarrow$ $r_\ell = r_\ell + \delta r_\ell$ (decrease layer strength)
~ recompute each layer SNR
label and store each atmosphere as $(h, r_0, \vec{v}, L_0, SNR)_\ell$
DATA SIMULATION
for n = 1:$N_a Z(N_L - m_L + 1)$ atmospheres
generate layers
$(h, r_0, \vec{v}, L_0, SNR)_\ell$
compute warp maps
$[\mathbf{w}(t_1), \mathbf{w}(t_2), ...]$
generate SA maps
and/or ST maps
label and store

FIGURE 5.7: Training data generation program flowchart.

The data simulation block computes the warp map output from every preallocated atmosphere model one at a time. After forming a warp map, the system slopes are processed into SA and/or ST maps which are then correlated are stored with label matching the atmosphere parameters. Once the desired output data for a particular atmosphere is generated, the simulated layers are cleared from memory and the next atmosphere is read in and simulated.

5.3.3 Simulated data for training layer ranging neural networks

The superior averaging of SA correlations over ST correlations makes generating layer ranging training data multiple times faster than a velocity estimation data set. Because of this, we are only able to demonstrate a training data set for layer ranging. Using the program in Fig. (5.7), training data was generated for the system defined by the parameters in Tab. (5.4). The chosen atmosphere binning structure is given by Fig. (5.9). Each simulated SA correlation is the result of averaging the measurements collected at time steps $t = [0, 50, 100, 150, 1000]$ ms.

The system used to generate data has a smaller FOV and fewer subfields and subapertures than the system in Tab. (5.1). The purpose of reducing the measurement dimensions for the training data system was to speed up the data generation program. Even with reduced sampling, the program took a month to run on a powerful computer.

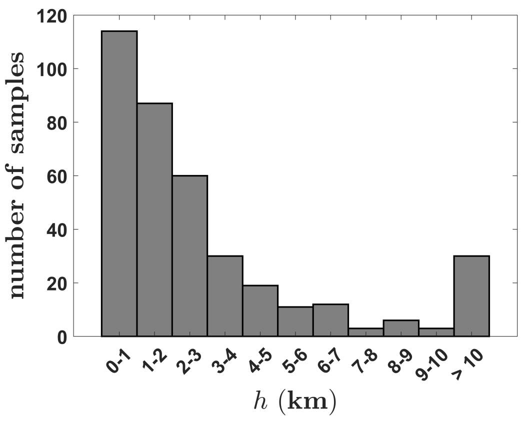
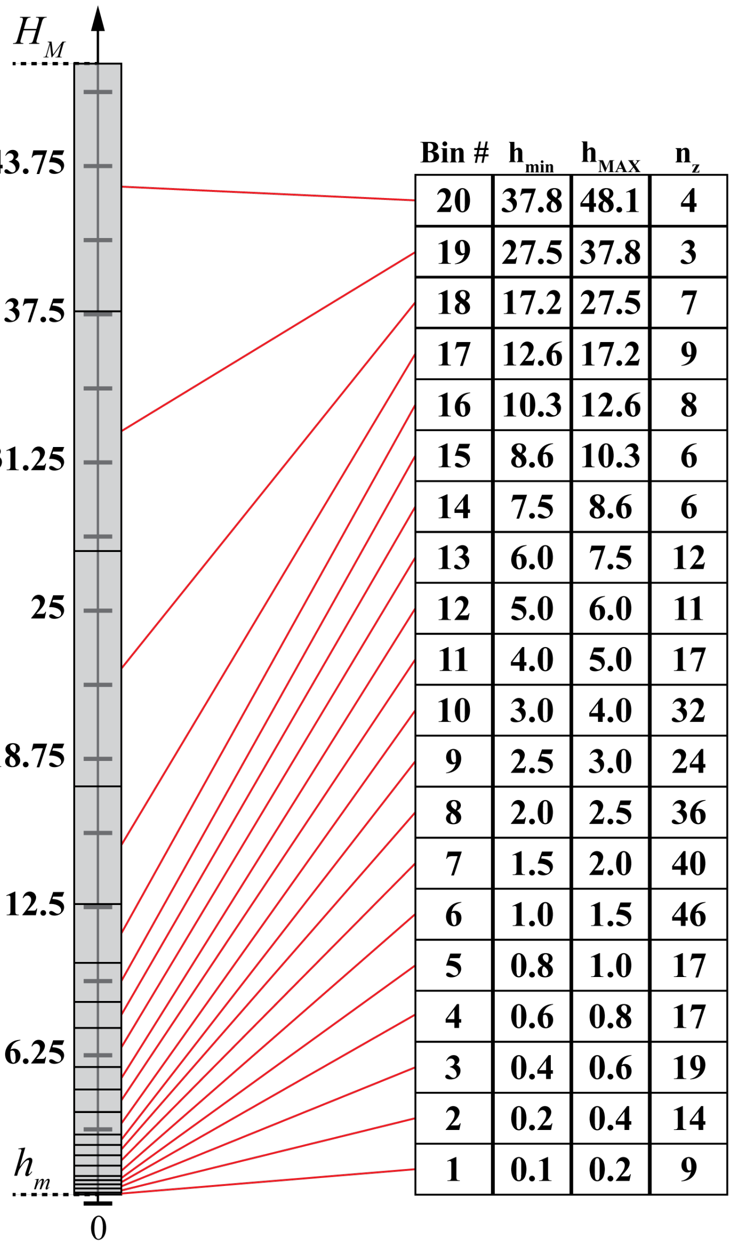


FIGURE 5.8: Histogram of SLODAR sample altitudes of the optical system with the properties of Tab. (5.4).

training system parameters	
D	1.5 m
$FFOV$	2.5’
θ	6"
$\sqrt{A}$	25
s	0.1 m
$\sqrt{B}$	15
h_m	0.14 km
H_M	48.1 km

TABLE 5.4: Parameters for the SLODAR system modeled to generate layer ranging training data.



Bin #	h_{min}	h_{MAX}	n_z
20	37.8	48.1	4
19	27.5	37.8	3
18	17.2	27.5	7
17	12.6	17.2	9
16	10.3	12.6	8
15	8.6	10.3	6
14	7.5	8.6	6
13	6.0	7.5	12
12	5.0	6.0	11
11	4.0	5.0	17
10	3.0	4.0	32
9	2.5	3.0	24
8	2.0	2.5	36
7	1.5	2.0	40
6	1.0	1.5	46
5	0.8	1.0	17
4	0.6	0.8	17
3	0.4	0.6	19
2	0.2	0.4	14
1	0.1	0.2	9

FIGURE 5.9: Binning structure used for the system defined by Tab. (5.4). n_z indicates the number of SA correlation sample pixels which fall into each bin, z.

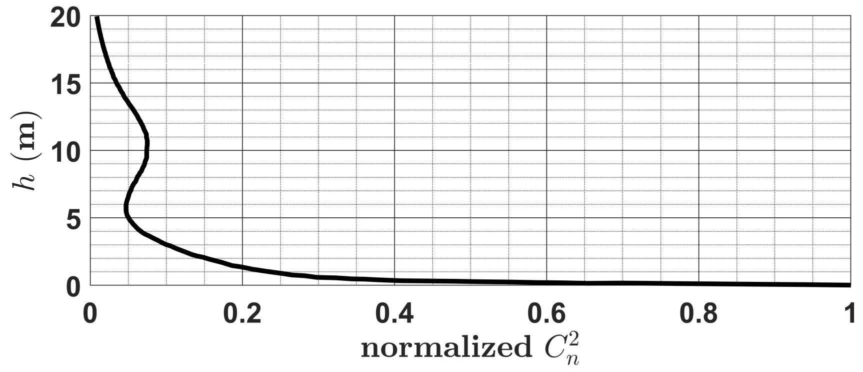


FIGURE 5.10: Hufnagel-Valley model curve used to produce training data with an average turbulence profile similar to an upwards looking telescope.

$N_a = 150$ atmospheres were generated with a first layer in bins $z \in [1,20]$ for $n_L \in [1,4]$ layer atmospheres. The resulting 12,000 simulated atmospheres contained 3,000 training atmospheres for each n_L, for a grand total of 30,000 simulated layers of turbulence. The number of training data points with layers in each sampling bin varies slightly since layers $\ell = 2 : n_L$ are randomly distributed uniformly in each bin which does not already contain a layer for that atmosphere. The distributions of all simulated layer distances are shown in Fig. (5.11).

Layer Fried lengths were chosen pseudo-randomly based on the Hufnagel-Valley $C_n^2(h)$ model, shown in its normalized form in Fig. (5.10), to approximate common average turbulence strength distributions for upward-looking telescope systems (Andrews and Philips 2012). From the relationship between the $C_n^2(h)$ profile and r_0, a Fried length of r_1 at a distance h_1 will tend to have a Fried length r_2 at distance h_2 following

$$r_2 = r_1 \left(\frac{C_n^2(h_2)}{C_n^2(h_1)} \right)^{-3/5} . \tag{5.13}$$

After generating the first layer distance and Fried length, Eq. (5.13) can be used to initiate the mean value of all subsequent layers after their distances are chosen. The first layer Fried length was selected randomly with the constraints that $0.05 \leq r_0$ and $0.31 r_0/h < \theta_0$, where θ_0 is the isoplanatic angle of the layer given by Eq. (1.6). The final r_0 for each layer can then be a random perturbation

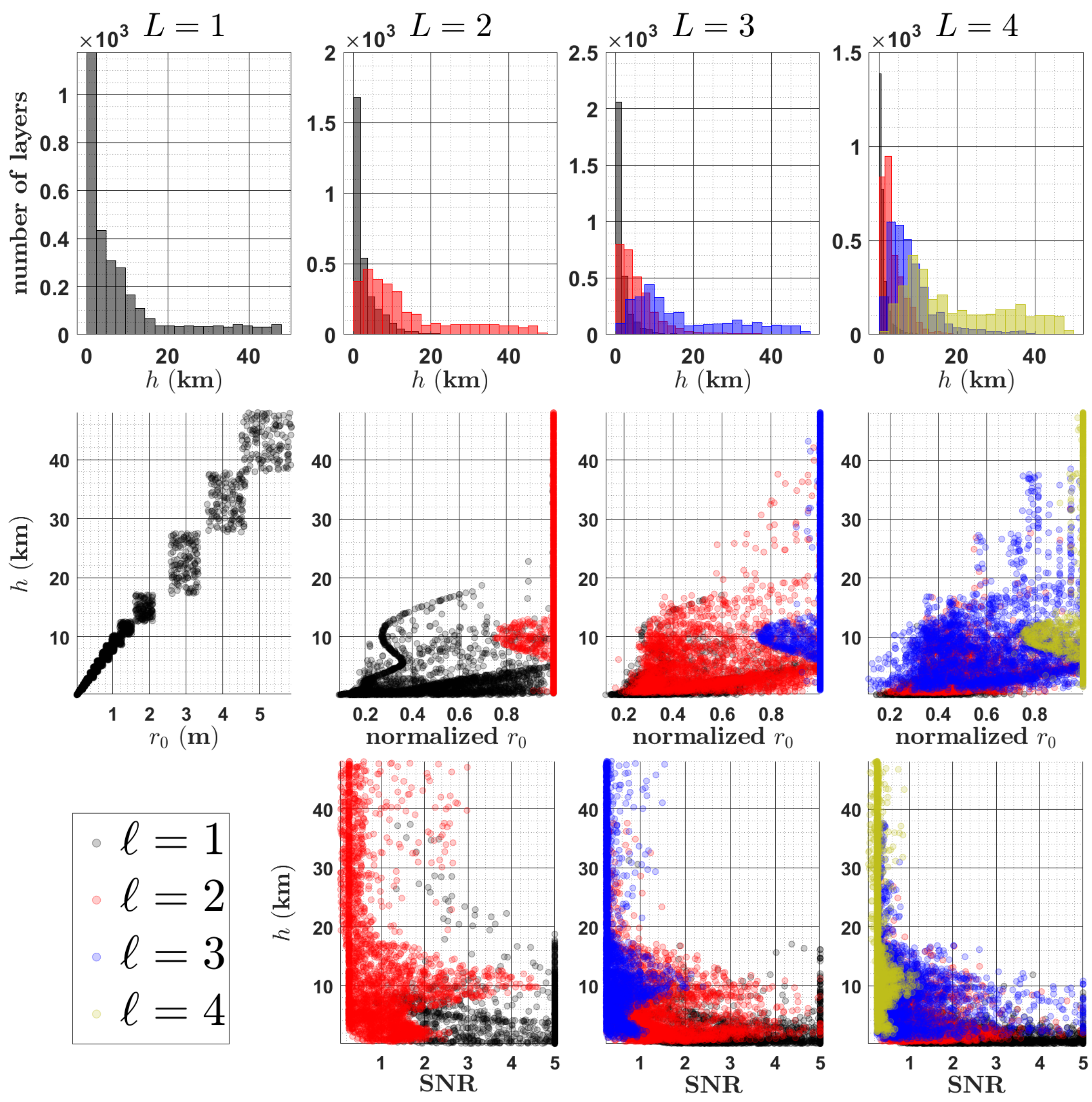


FIGURE 5.11: Simulated layer ranging network training data statistics separated by atmospheres with $L = [1,4]$ layers of turbulence. The layers are indexed $\ell = [1,L]$, where higher index layers are more distant. This allows us to see how conditioning the atmosphere paramters produces specific distributions of layer distances, strengths, and subsequent SNRs. In general, low altitude layers have more strength and higher SNR. We can also see the effect of scaling Fried lengths by the Hufnagel-Valley model, where strong layers tend to cluster below 5 km and around 10 km. Layers with SNR > 5 are displayed as SNR $= 5$ to help visualize the data.

around the mean value based on the $C_n^2(h)$ model. The resulting distribution of layer Fried lengths as a function of altitude from using the Hufnagel-Valley model is shown in Fig. (5.11). The one layer atmosphere Fried lengths are shown as their value in meters, aiding in the visualization of how the conditions on the first layer r_0 force specific distances to have a distribution of possible layer strengths. The multi-layer atmosphere Fried lengths are displayed normalized by the value of the weakest layer. This allows us to see how the chosen $C_n^2(h)$ profile tends to distribute strong layers at $h < 5$ km and around ~ 10 km.

After selecting the layer altitudes, strengths, and outer scales, the SNR is conditioned. We set a soft requirement of SNR ≥ 0.25. Each layer with an initial SNR below 0.25 was repeatedly increased in strength and the strongest layer was weakened until either all SNRs were above 0.25 or 100 adjustment attempts were made. Of the 27,000 layers modeled in multi-layer atmospheres, 12,312 layers have SNR < 1 with only 222 of them having SNR ≤ 0.25. All layer SNRs are displayed in Fig. (5.11)

5.4 Training and evaluating layer ranging neural networks

5.4.1 Network architectures for turbulence profiling

The capabilities of a trained ANN are dependent on the internal architecture of the network and the task it is being asked to perform. For example, the structure of a network built to interpret and generate text as coherent sentences requires a different structure and training process than a network which finds faces in images. This has to be the case since the inputs, outputs, and natures of the two tasks are entirely different. Additionally, simple network architectures will run faster and be easier for others to understand and implement. Given the data structures developed in this chapter, we propose that the two simplest and most suitable types of neural networks for turbulence profiling are multi-layer perceptrons (MLPs) and convolutional neural networks (CNNs).

Multi-layer perceptrons for turbulence profiling

MLPs are feed-forward networks which pass a vector of input data through a series of layers – the last of which is the network output vector. The network layers are made up of interconnected nodes which apply weights to the input data and pass them to linear or non-linear activation functions. The nodes learn specific weighted responses which produce the desired output for a particular input during the training process. Since the input and output vectors of an MLP are related by a combination of linear and non-linear responses, MLPs are capable of learning non-linear multi-variable relationships. This relationship can be probabilistic, such as a classification task, or functional, like a mathematical regression.

In SA correlation data each pixel corresponds to a sum of signal and noise tip/tilt correlations. The location of each measurement tells us information about the distance to turbulence. Thus, MLPs trained on SA correlations should be able to build a relationship between the values at SA correlation pixels and the presence of a layer of turbulence in a specific SLODAR bin in front of the system. We propose posing this relationship as a binary classification problem.

For a binary classification problem, the network output layer has dimension 2 (one for turbulent layer present in the bin and one for not present) which feeds into a softmax activation function. The softmax activation transforms the network output layer responses into a vector of probabilities for each trained feature in the data. The sum of all outputs by the softmax function equals 1, meaning the outputs of the trained MLP will be directly translatable as ‘probability layer of turbulence is present in the bin’ and ‘probability layer of turbulence is not present in the bin’. After training an MLP for each SLODAR bin, the same SA correlation can be fed into each network to produce a probabilistic model of the distribution turbulence in front of the system. The proposed MLP architecture for layer ranging is given by the top diagram in Fig. (5.12), and a process flowchart for building one network per-bin to obtain probability estimates of layer locations is shown in the bottom diagram.

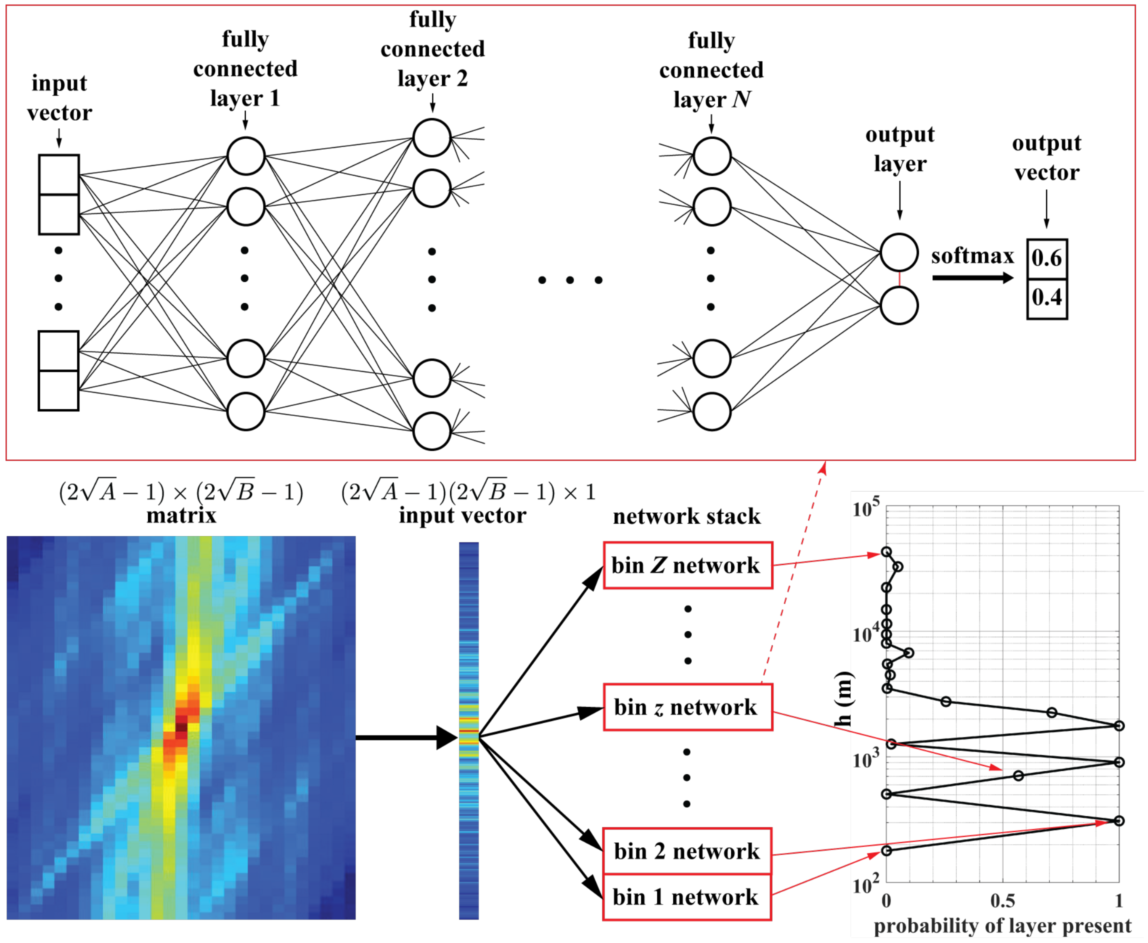


FIGURE 5.12: Top: layer ranging MLP architecture consisting of N layers. Each layer contains a predetermined number of nodes, which can change between layers, where each node contains a weighting function and an activation function. Each node is fully connected to all nodes in the layer before and after its layer. Bottom: process flow for using a stack of trained single distance bin MLPs to interpret an SA correlation matrix as a probability mapping of layer locations. The shown SA correlation matrix is for the full $(\pm\delta_a, \pm\delta_b)$ convolved SA maps.

Convolutional neural networks for turbulence profiling

CNNs are networks which contain at least one convolution layer. The convolution layer generates filter matrices and convolves the input data by each filter before passing the location-specific results through additional processing. The additional processing generally consists of a normalization process, an activation function, and a pooling function. Normalization helps keep the scale of network responses comparable for different input features, making the activation responses more consistent. Pooling is a filtering method which bins pixels in the convolved images and extracts information such as the location of local maxima or local averages. Stacking multiple convolutional layers allows for the network to interpret collections of pixels in the input images into higher levels of abstraction, making CNNs excellent at interpreting images as a distribution of shapes and patterns at specific locations. After the convolution layers, the data is sent to fully connected layers to distill the filtered images into a vector of outputs suitable for understanding classification or regression information from the input data. CNNs can be built for 1D, 2D, and 3D data, making them a viable machine learning technique for interpreting time series data, images, videos, and point-cloud matrices for structure-based information.

In the context of turbulence profiling, CNNs are options for both turbulence layer finding and velocity estimation. The position and angle of lines in the SA correlation give us information about the distance to layers of turbulence. Therefore, 2D convolution layers can assist in understanding the signal-noise relationship in the presence of specific distribution of layers in front of the system. The proposed architecture for interpreting ‘probability layer of turbulence is present in the bin’ and ‘probability layer of turbulence is not present in the bin’ from SA correlations with CNNs is shown in Fig. (5.13).

While not demonstrated in this work, we hypothesize that CNNs are a suitable machine learning technique for extracting turbulent layer velocities. For velocity estimation, the 2D location of layer signals extrapolated along the time dimension – thereby forming lines in the space-space-time coordinate space – tells us the speed and direction of the layers. By building a CNN with a regression output, both 3D and 2D convolutional layers should be suitable to regress the slopes of the layer lines in ST correlation data and interpret that as layer velocity. Having *a priori* knowledge of the layer distances from classified SA correlation data also means that a velocity estimation network can be trained for each SLODAR distance bin, fixing the starting location of each layer signal and assisting in the complexity of the problem the networks are being tasked with learning.

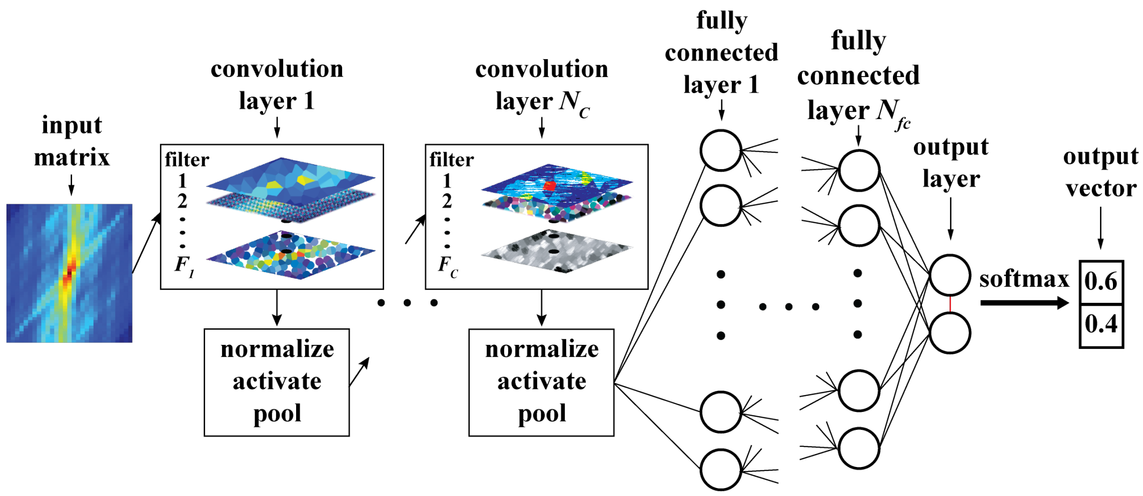


FIGURE 5.13: Layer ranging CNN architecture consisting of N_C convolutional layers feeding out to N_{fc} fully connected layers. Each convolutional layer, indexed $c = [1, C]$, consists of F_c filters with a predetermined shape and filter step size. The 'filtered' images in the figure are illustrative of the process and do not represent actual convolutional layer filters. We show SA correlations which have been convolved in $(\pm\delta_a, \pm\delta_b)$. A network like this would be used identically to the MLP network stack in the turbulence ranging flowchart in the bottom diagram of Fig. (5.12).

5.4.2 Ranging network training and validating

With ranging network architectures planned, we built and trained the MLPs and CNNs using MATLAB's deep learning toolbox. Since there are infinitely many combinations of layers which can be assembled into a network, all of which work to varying degrees, only one MLP and one CNN architecture are investigated. Given that the training data is sufficiently large, containing 30,000 layers of turbulence uniformly distributed in each distance bin across 12,000 different atmospheres, we are able to train and validate simultaneously using a 5-fold cross validation as shown in Fig. (5.14).

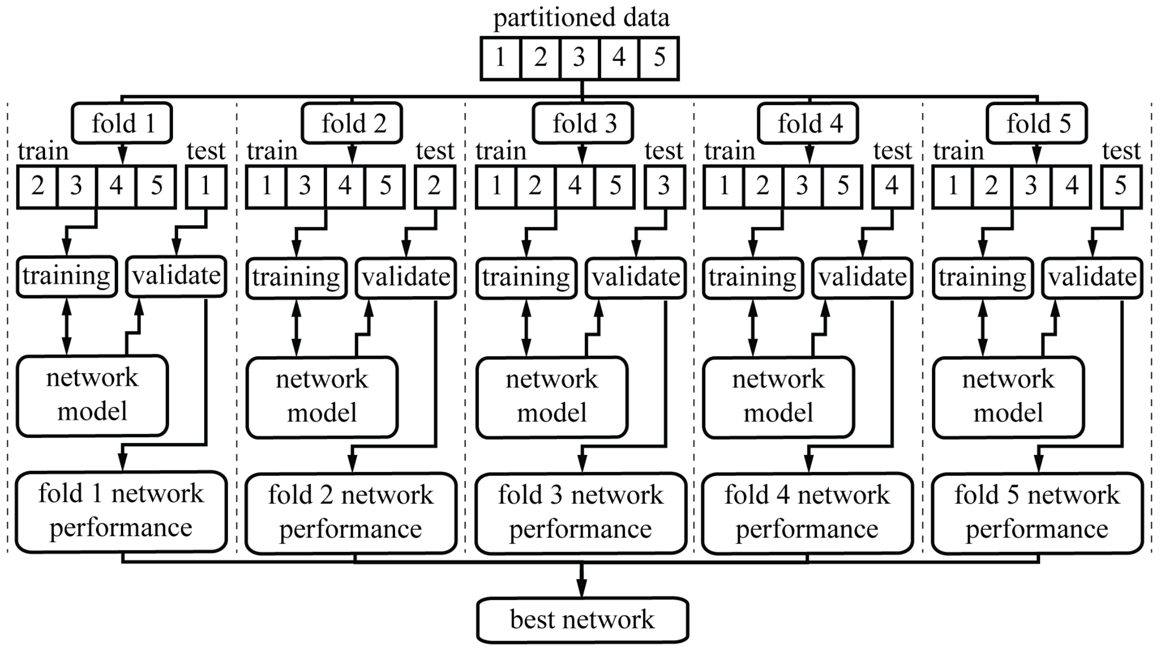


FIGURE 5.14: Process for partitioning data into $k = 5$ folds for simultaneous training and validation of networks. Each partition is made up of data points randomly sampled from the full set. The same initial untrained network model is used in each fold. During the training process, the network is intermittently tested between iterations on the partition which has been held out from the training partitions. The results of each test allows us to asses network learning performance and to check that over-fitting is not occurring – i.e. training performance improving while testing performance declines. The network with the highest final validation accuracy and lowest loss metric across all folds is considered the best network and is saved for future use.

Turbulence ranging MLP training and validation

The layer parameters for the modeled and trained MLPs are given in Tab. (5.5). The fully connected layers are followed by activation functions and drop out layers. Drop out layers are a utility which shuts off a percentage of the nodes in the subsequent fully connected layer at each training step to stop the network from over-fitting to particular batches in the the data. The training process was performed over 30 epochs in mini-batch sizes of 32 SA correlations using the ADAM training algorithm (Kingma and Ba 2014). After every 128 SA correlations (i.e. every 4 mini-batches) validation accuracy and loss are computed using the model at that stage in training. 5-fold validation of the 12,000 data points with all 20 bin-specific networks took ~ 6 hours on an NVIDIA GTX 1060.

layer	type	properties	activation
1	input	$[375 \times 1]$ $[1421 \times 1]$	N/A
2	fully connected	256 nodes	relu
3	drop out	20%	N/A
4	fully connected	512	relu
5	drop out	30%	N/A
6	fully connected	256	tanh
7	drop out	20%	N/A
8	fully connected	64	relu
9	output	classification	softmax

TABLE 5.5: Sequential layout of the MLPs trained on the simulated training data using the 5-fold cross-validation scheme.

The training process was repeated twice – once for networks which interpret four quadrant SA correlations and once for networks which only process SA correlations cropped to the $(+\delta_a, +\delta_b)$ quadrant. The cropped SA correlations are more compact, thereby occupying less space and taking less time to compute, but contain less functional information than the full SA correlations. The purpose of training networks on both forms of the SA correlation is to see if the networks will perform better on one form of the metric than the other. The validation accuracies and losses for the best-fold MLPs corresponding to each distance bin for both full SA correlation and cropped SA correlations are plotted in Fig. (5.15).

As expected, networks attempting to recognize layers in bins corresponding to fewer SA correlation pixels take longer to train and do not perform as well. We can see that while each bin was trained on the same number of points with identical mini-batch size over an identical epoch count, the low altitude layer ranging networks converge to solutions in fewer training steps. The full SA correlations also take longer to converge to the same level of accuracy and loss as the cropped SA correlations, especially for the higher altitude layers. This informs us that the MLPs are not better off when exposed to the additional function symmetry of the full SA correlation matrices. Therefore, if using MLPs to find layers of turbulence from SA correlations, it is better to use SA correlations containing only the $(+\delta_a, +\delta_b)$ quadrant. All MLPs were capable of classifying the presence of a layer of turbulence within particular altitudes, with the worst performing networks still achieving final validation accuracy of 94%.

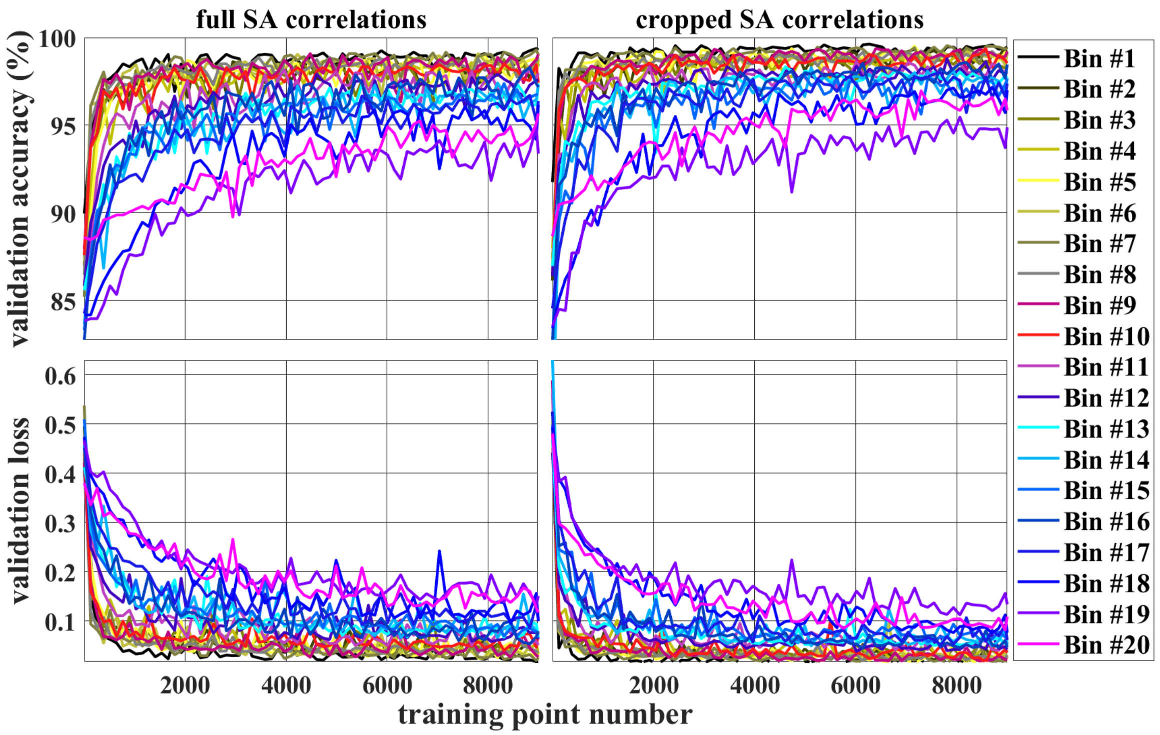


FIGURE 5.15: Ranging MLP validation accuracy (top row) and validation loss (bottom row) for networks trained on four quadrant SA correlations (left column) and the cropped $(+\delta_a, +\delta_b)$ quadrant SA correlations (right column). We can see that high altitude bins tend to take longer to train, and do not perform as well when fully trained.

Turbulence ranging CNN training and validation

The CNN architecture used is given by Tab. (5.6). The exact same training function, mini-bath size, and number of training epochs were used to train the CNNs as the MLPs in the previous section, making their results directly comparable. Similar to the MLPs, the 5-fold CNNs for all distance bins took around 6 hours to train on an NVIDIA GTX 1060. The validation results acquired during training are plotted in Fig. (5.16).

layer	type	properties	activation
1	input	$[25 \times 15]$ $[49 \times 29]$	N/A
2	2D convolution	32 $[6 \times 6]$ filters $[3 \times 3]$ padding	bath normalization relu
3	max pooling	$[2 \times 2]$ binning 2 pixel stride	N/A
4	2D convolution	16 $[4 \times 4]$ filters $[1 \times 1]$ padding	bath normalization relu
5	max pooling	$[2 \times 2]$ binning 2 pixel stride	N/A
6	2D convolution	8 $[2 \times 2]$ filters $[1 \times 1]$ padding	bath normalization relu
7	max pooling	$[2 \times 2]$ binning 2 pixel stride	N/A
8	drop out	30%	N/A
9	fully connected	128	relu
10	output	classification	softmax

TABLE 5.6: Sequential layout of the CNNs trained with the simulated training data using the 5-fold cross-validation scheme.

The CNNs outperform the MLPs in terms of the speed at which they learn the layer ranges as well as final validation – achieving $> 97\%$ accuracy in all bins. We believe that the data abstraction obtained by filtering the SA correlations through convolutional layers produces a better understanding of each layer distance regardless of the full atmosphere profile than the MLPs are able to learn. We also found that the full SA correlations and the cropped $(+\delta_a, +\delta_b)$ quadrant SA correlation matrices produced similar performing networks, with the full matrices obtaining $\approx 0.25\%$ higher validation accuracy on average across all bins. Since the testing data sets are randomized, we do not consider this to be statistically significant enough to state that the full matrix networks perform better than the cropped matrix networks.

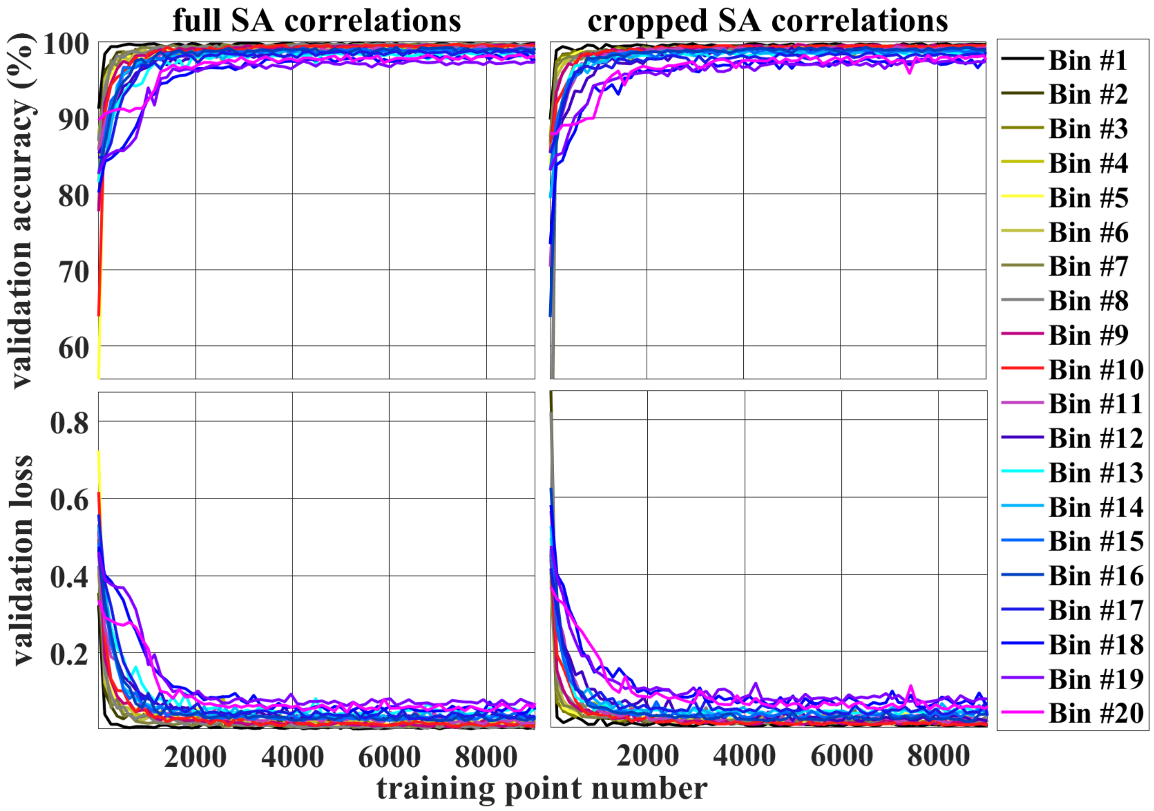


FIGURE 5.16: Ranging CNN training validation accuracy (top row) and validation loss (bottom row) for networks trained on four quadrant SA correlations (left column) and the cropped $(+\delta_a, +\delta_b)$ quadrant SA correlations (right column).

5.4.3 Ranging network performance as a function of layer SNR

Training resulted in ranging networks with high validation accuracy and low loss. However, the validation process does not provide insight into how the the networks perform when attempting to detect layers with different SNRs. From theory, we can predict that low SNR layers will be more difficult features for the networks to learn. We also know that the data generation process, specifically the r_0 selection process based on a specific $C_n^2(h)$ profile and the subsequent SNR thresholding, resulted in drastically different SNR distributions in each altitude bin. Therefore, we can expect the networks at different altitudes to pay attention to specific turbulence strengths based on their training data SNR distributions.

To analyze network performance as a function of layer strength we fed each ‘layer present’ training data point into the full SA correlation CNN and MLP and stored the responses, $P(\text{true})$, along with the corresponding layer SNR. We then sorted the layer responses from low to high SNR and applied a sliding average. The sliding average is computed using a 1D convolution:

$$\overline{\text{SNR}} = \text{SNR} \otimes \frac{\vec{\mathbf{1}}_f}{f}, \tag{5.14}$$

where $\vec{\mathbf{1}}_f$ is a 1's vector with f elements. The same sliding average is then applied to the network responses to produce the average 'layer present' responses, denoted $\bar{P}(\text{true})$. We can then plot $\bar{P}(\text{true})$ vs. $\overline{\text{SNR}}$ for each network to quantify behavioral trends.

We use the sliding average because it is easier to analyze and more representative of general performance trends than the direct responses for each input point. This is because the sliding average smooths variations in the network responses caused by similar SNR data points coming from different noise layer combinations, resulting in differently structured input SA correlations. With the variations in the data smoothed, we can identify three classes of networks which have been trained: high SNR, mid SNR, and low SNR ranging networks

High SNR networks

High SNR networks are networks which have been predominantly trained on data points with SNR > 1. By conditioning the data set r_0 values on the Hufnagel-Valley model, the trained networks which meet the condition for high SNR are for ranging distances < 1 km. These networks have high accuracy across the full data set, but a majority of missed layers corresponding to low SNRs. To analyze these types of networks, we selected the networks for distance bin 1, corresponding to 143-215 m (2 points with SNR < 1), bin 2, corresponding to 215-404 m (11 points with SNR < 1), and distance bin 3, corresponding to 404-607 m (55 points with SNR < 1). The sliding average 'layer present' probabilities and SNRs were computed for each of the chosen networks with sliding average width $f = 10$. The results were plotted for the full SA correlation CNNs and

MLPs, along with histograms of unaveraged SNRs in the data set for ‘layer present’ in each bin, in Fig. (5.17).

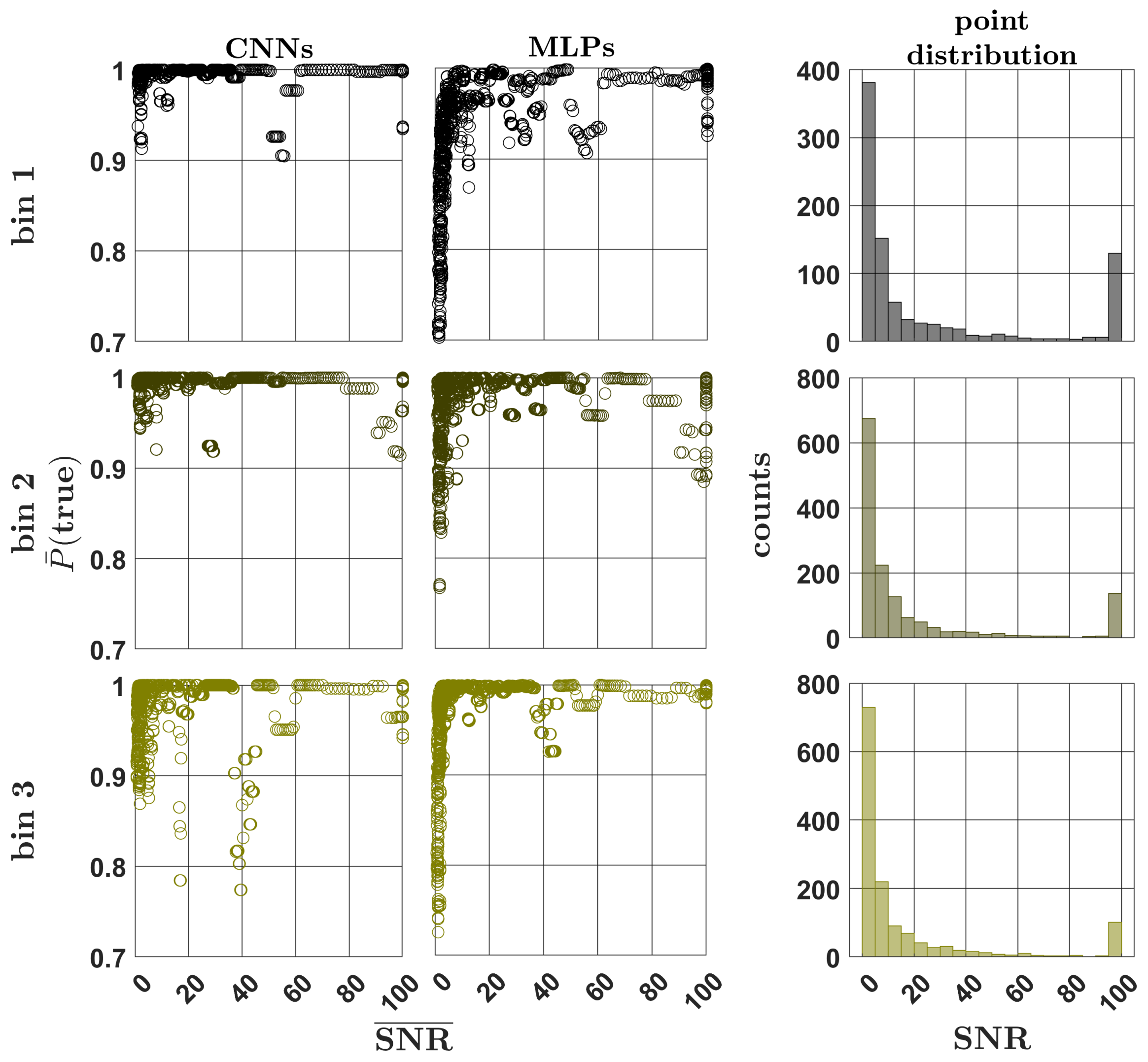


FIGURE 5.17: Sliding average trained network responses for distance bins 1-3 for all ‘layer present’ = true atmospheres in the training data set. The sliding average is for $f = 10$ points to help mitigate over-smoothing of the average network responses for SNRs with low sampling density. Points with SNR > 100 are displayed at a value of 100 to help with visualization.

From Fig. (5.17) there are two immediate takeaways. First, low SNR layers and regions with low SNR sampling density, indicated by the point distribution histogram for each data set as a function of layer SNR, tend to have lower confidence in the probability of a layer being present.

This can be seen noticeably in the decreases in $\bar{P}(\text{true})$ at SNRs closer to zero and throughout the $40 < \text{SNR} < 100$ range. Second, the behavior of the CNNs and MLPs is different when plotted as a function of SNR. MLPs are particularly sensitive to low SNR layers while the CNNs have a higher-order functional relationship. The CNNs generally have more confidence in the presence of low SNR layers, but tend to also trade some additional uncertainty to the higher SNR points. Even with decreased uncertainty at particular layer SNRs, high layer SNR MLPs and CNNs both tend to properly detect layers of turbulence.

Mid SNR networks

Mid SNR networks are networks which have a large number of points with $\text{SNR} \approx 1$. In the training data set, this roughly corresponds to layers around $5 - 15$ km. To analyze these types of networks, we repeated the analysis for high SNR networks but for the networks corresponding to distance bin 12 for 4,966-6,016 m, bin 13 for 6,016-7,448 m and bin 14 corresponding to 7,448-8,594 m. The results are shown in Fig. (5.18).

The same two behaviors found for the high SNR layers are also observed for the mid SNR layers. However, since there are more layers with $\text{SNR} < 1$ in the training data for these networks, the mid SNR average performance also provides insight into the source of variations in the CNN and MLP validation accuracies observed during training. The general trend of quickly decreasing $\bar{P}(\text{true})$ for decreasing $\overline{\text{SNR}}$ seen in the high SNR networks is amplified for the mid SNR networks, resulting in MLPs which are approaching random guess classifiers – i.e. $\bar{P}(\text{true}) = 0.5$ – for layers with $\text{SNR} < 1$. This is not the case for the CNNs which remain robust to low SNR layers on average. While there is a decrease in the probability for lower SNR layers for the CNNs, their average layer detection probability remains > 0.8. We believe this is due to the more abstract interpretation of the layer signals learned by the CNNs which is more capable of discerning weak signal patterns from strong noise patterns. In general, layers with $\text{SNR} > 2$ tend to be properly detected with at least 90% confidence on average for mid SNR networks.

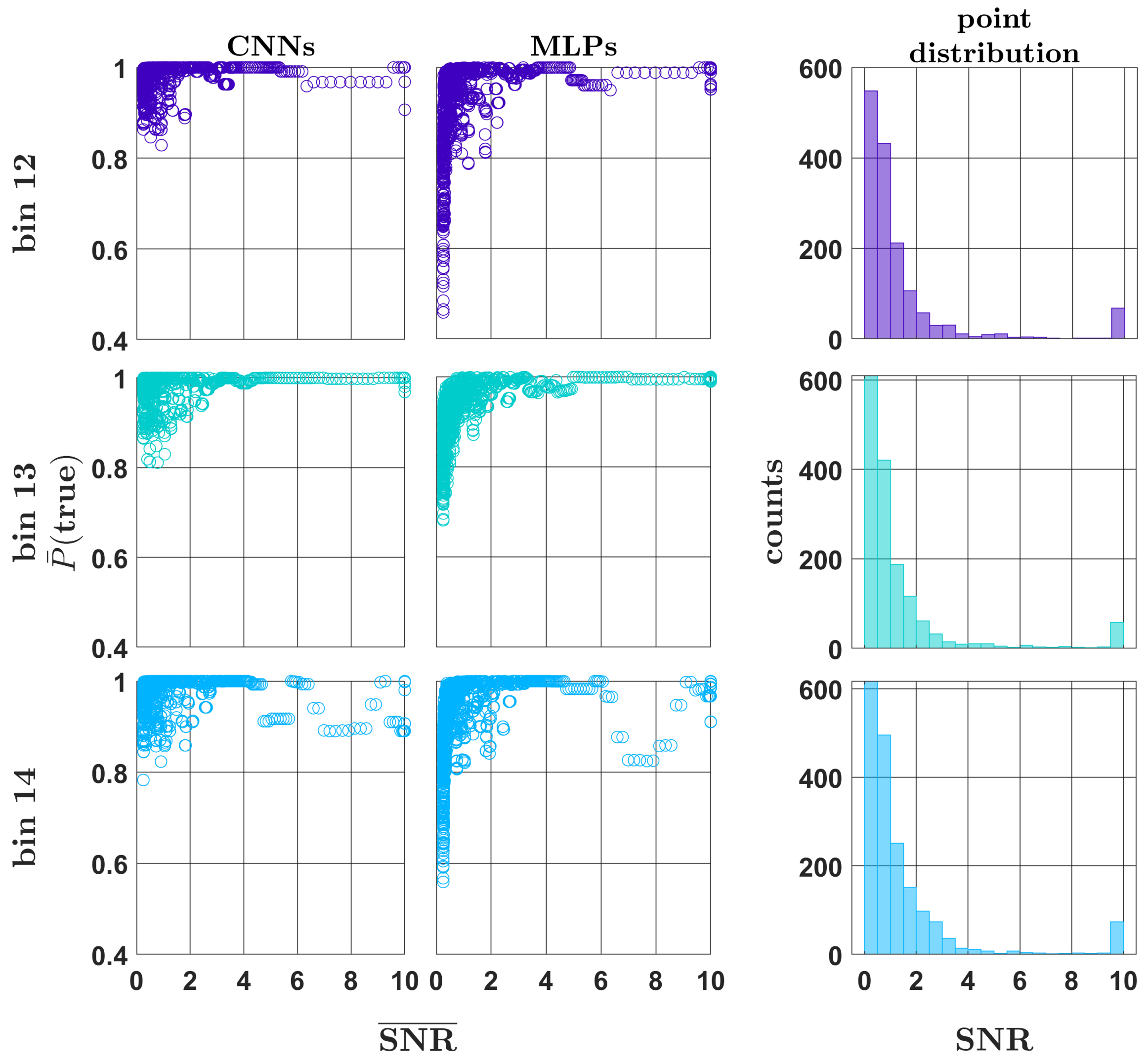


FIGURE 5.18: Sliding average trained network responses for distance bins 12-14 for all 'layer present' = true atmospheres in the training data set. The sliding average is for $f = 10$ points to help mitigate over-smoothing of the average network responses for SNRs with low sampling density. Points with SNR > 10 are displayed at a value of 10 to help with visualization.

Low SNR networks

Low SNR networks are networks which have been trained on a large number of points with SNR $<$ 1. In the data set generated in this work, the SNR conditioning process produced high altitude bin training data with a majority of layers having SNR $\approx$ SNR$_\text{m} = 0.25$. To analyze these types of networks, the same process for investigating high and mid NR networks was repeated on the networks corresponding to distance bin 16 for 10,313-12,605 m (637 points with SNR < 0.5), bin 18 for 17,189-27,502 m (1,542 points with SNR < 0.5), and bin 20 corresponding to 37,815-48,128 m (1,078 points with SNR < 0.5). The results are shown in Fig. (5.19).

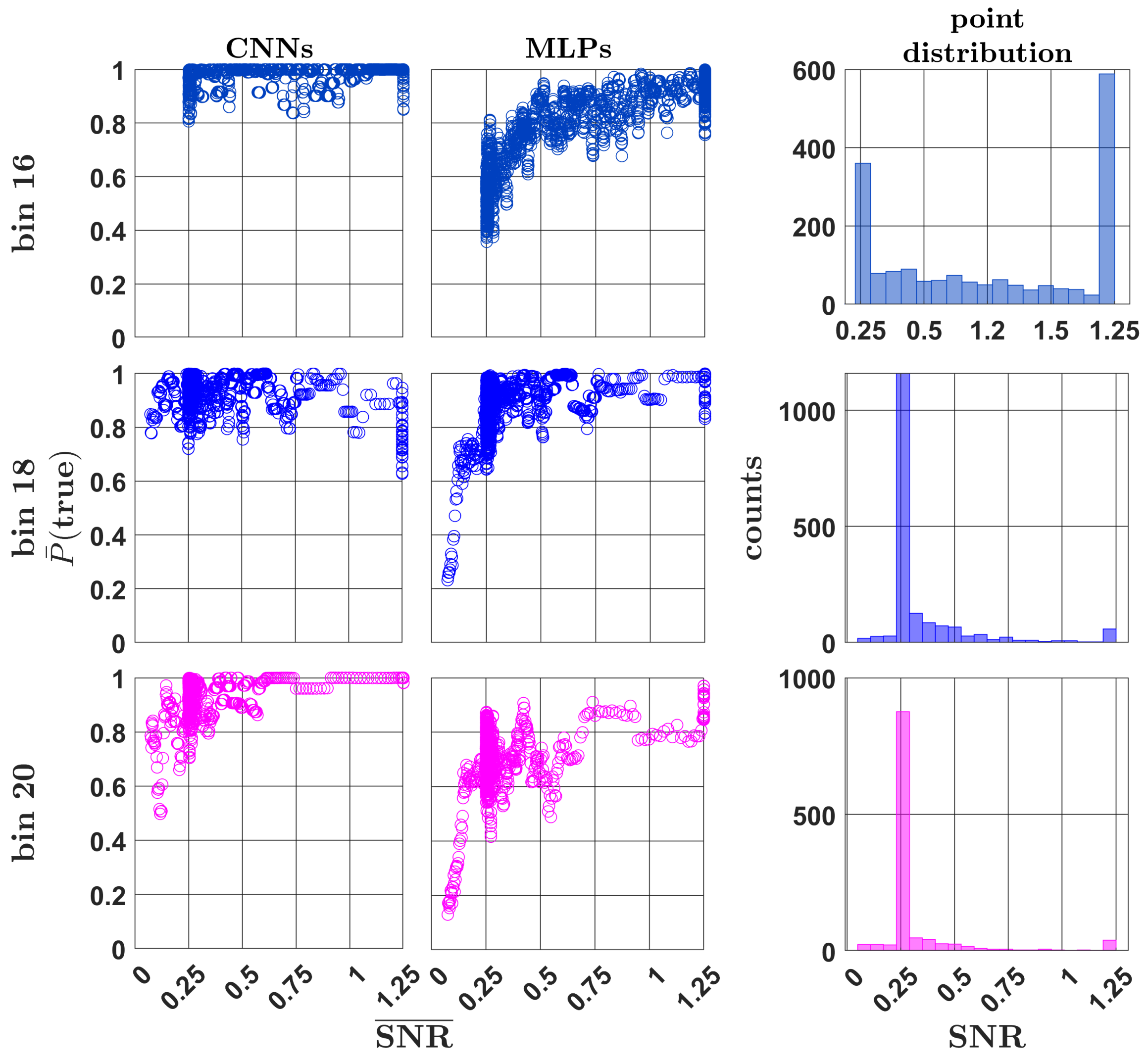


FIGURE 5.19: Sliding average trained network responses for distance bins $(16, 18, 20)$ for all 'layer present' = true atmospheres in the training data set. The sliding average is for $f = 10$ points to help mitigate over-smoothing of the average network responses for SNRs with low sampling density. Points with SNR > 1.25 are displayed at a value of 1.25 to help with visualization.

The differences between the CNN and MLP networks observed in the other two SNR regimes is mirrored for the networks trained on low SNR layer data. We also see an amplification of the dependence on network performance based on the SNR sampling density because of the large number of points with SNR ≈ 0.25. The CNNs and MLPs are able to detect layers with an SNR around 0.25 on average, but performance drops quickly for lower SNRs. This is especially true of the MLPs. However, the observed relationship does highlight that the MLPs can learn the low SNR layer signal relationship when provided a sufficient number of data points with low SNR.

5.4.4 Analysis of misclassified points from ranging networks

The above analysis focused on the average capabilities and sensitivities of the networks for finding layers present in the data, but it does not say anything about the number of missed points, the SNRs of each of those points individually, or the position of the missed layers in each distance bin. Dips in $\bar{P}(\text{true})$ could potentially correspond to something as trivial as a decrease in values for an ensemble of point responses that still satisfy $0.5 \leq P(\text{true}) \leq 1$ – meaning the layer is still properly detected. Another possible explanation is that networks are insensitive to layers of particular SNRs at particular positions in a network's bin and therefore those layers reliably go undetected.

To check what combinations of layer SNRs and bin positions are going undetected in the the data, we use the same networks and data used to generate the plots for $\bar{P}(\text{true})$ vs. $\bar{\text{SNR}}$. but only look at the points with $P(\text{true}) < 0.5$. We split the points into two sets: one set for all points with SNR > 1 and another set with all points SNR < 1. The two sets are then plotted as overlapping hisograms where the bins are split by position in the network distance sampling bin and the counts are number of points with $P(\text{true}) < 0.5$. The results for full SA correlation CNNs are shown in Fig. (5.20), and for full SA correlation MLPs in Fig. (5.21).

There are two key insights into the network ranging performance from this analysis. First, the missed points are dominated by layers with SNR < 1 (histogram bars with black edges). Second, layers which are at a distance near to the bounds of a network distance bin, including layers with SNR > 1, are more likely to be missed by the networks than layers which are in the middle of a bin. We can be sure of this since the distribution of altitudes in each distance bin is approximately uniform, and the trends of the histograms in Fig. (5.20) and Fig. (5.21) are u-shaped.

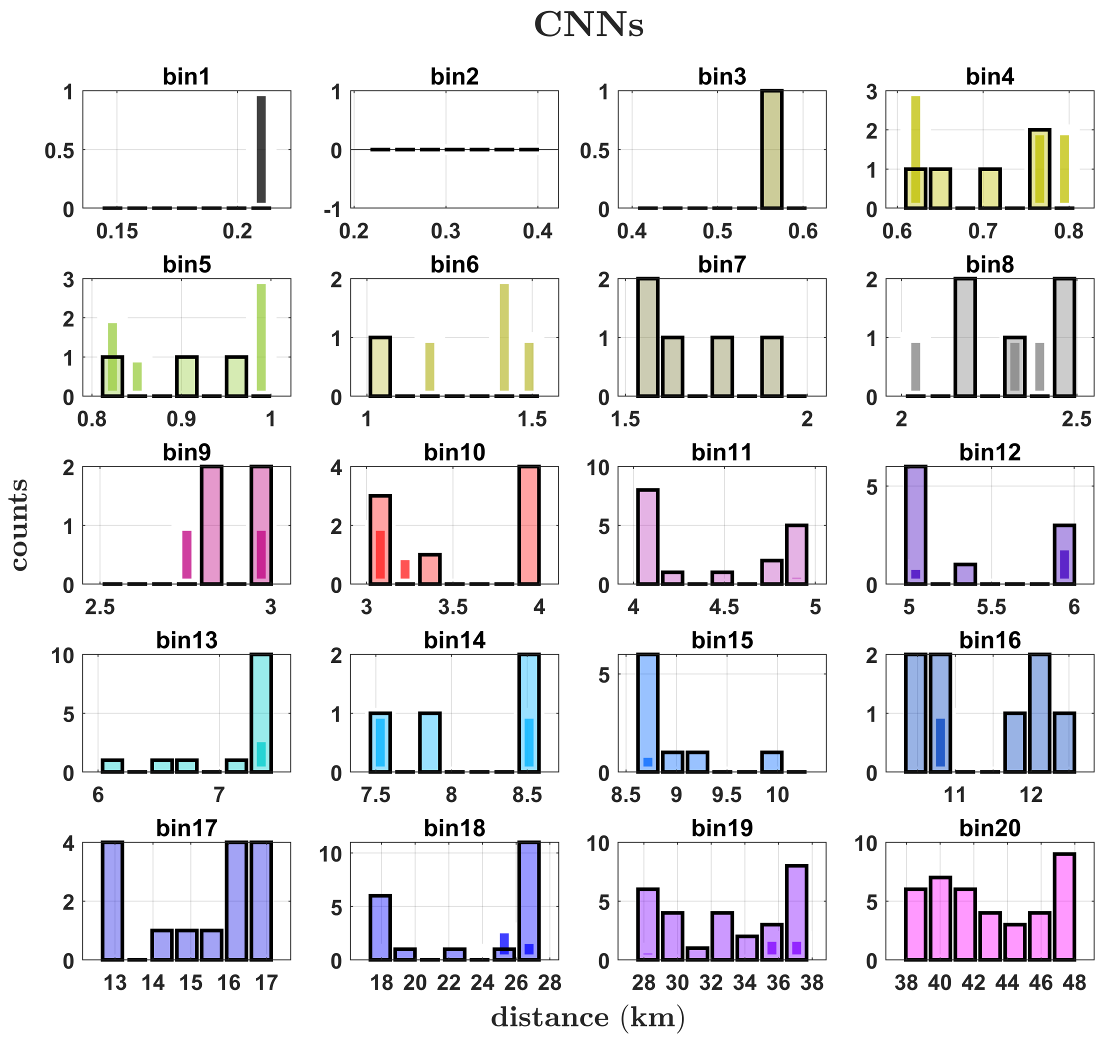


FIGURE 5.20: Histograms of each layer missed by the trained CNNs for the simulated data set organized by layer height in the corresponding network distance bin. Each plot shows two historgrams – one for the layers with SNR < 1, indicated by bars with black borders, and one for layers with SNR < 1, indicated by bars with thick white boarders.

The relationship between layer SNR, distance, and trained network misclassification gives rise to a new question: if layers near the edge of the bins are more likely to be missed, are they also more likely to be detected by an adjacent distance bin network? To check this, we go through all network

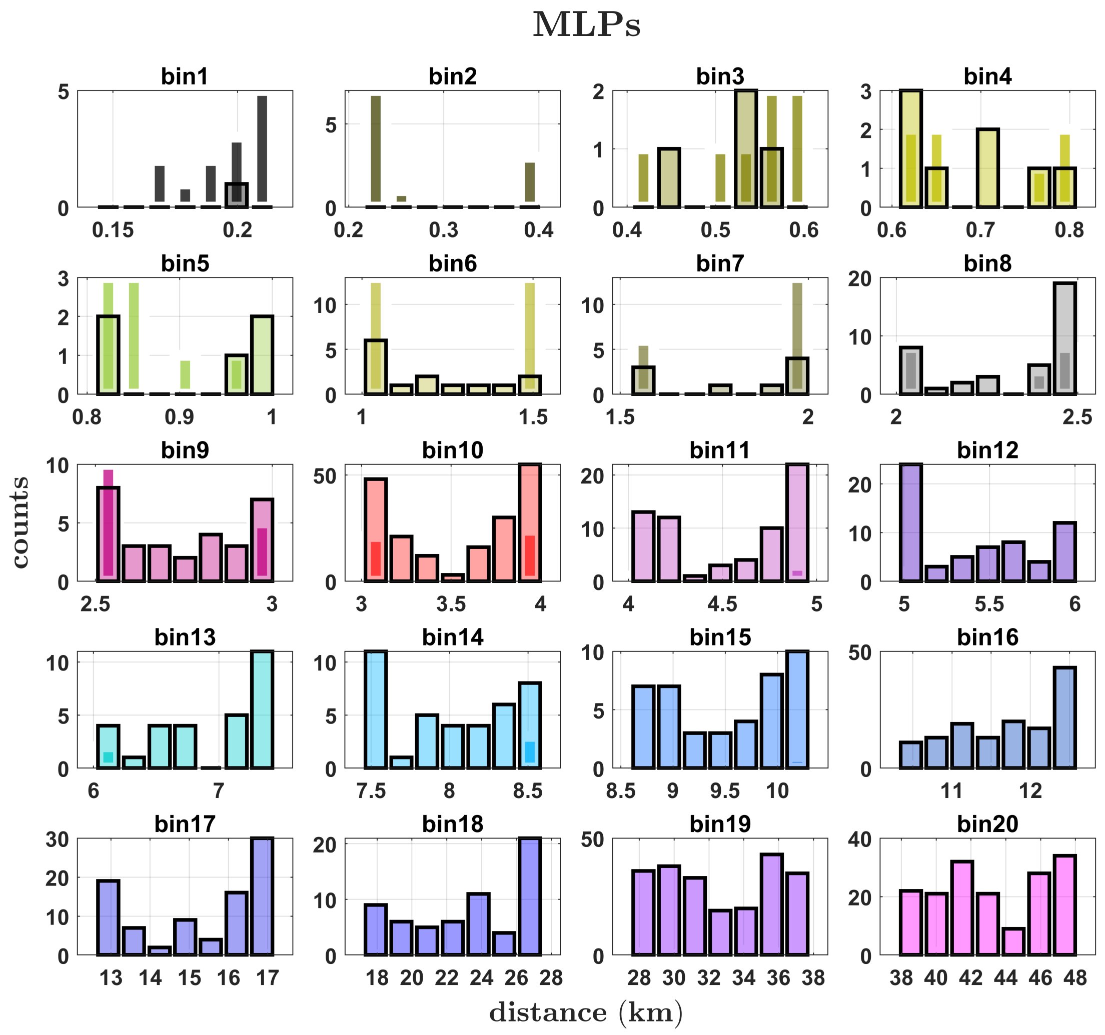


FIGURE 5.21: Histograms of each layer missed by the trained MLPs for the simulated data set organized by layer height in the corresponding network distance bin. Each plot shows two historgrams – one for the layers with SNR < 1, indicated by bars with black borders, and one for layers with SNR < 1, indicated by bars with thick white boarders.

responses for 'layer not present' input data points and store all layer distances for atmospheres which provide a false alarm response. We can then produce histograms of all false alarm atmospheres for each bin, as shown in Fig. (5.22) for full SA correlation CNNs and Fig. (5.23) for full SA correlation MLPs.

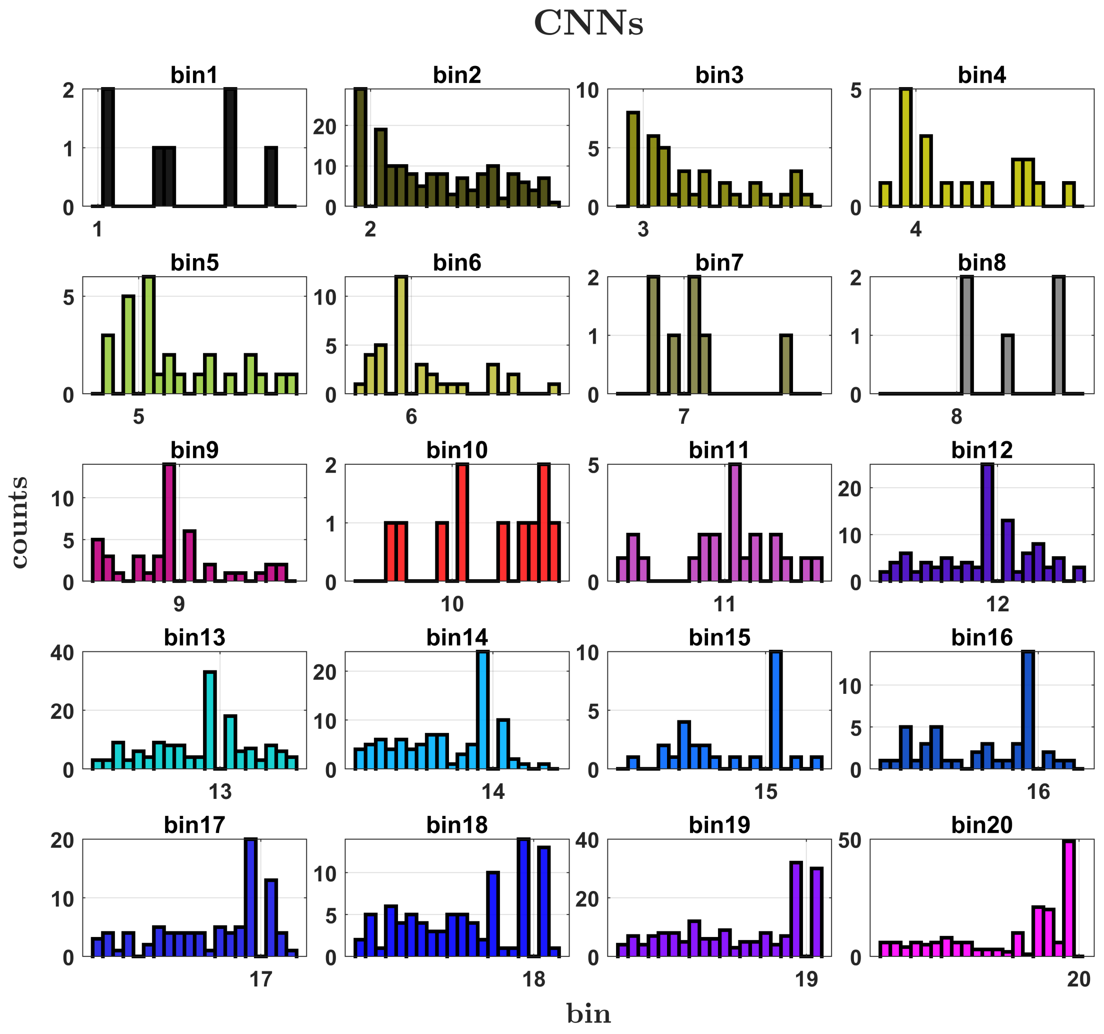


FIGURE 5.22: Distribution of all layers from each atmosphere which triggered a false alarm layer detections in each CNN. The location of the altitude bin corresponding to each network is labeled on the x-axis.

The distribution of false alarm atmospheres for each network bin in Fig. (5.22) and Fig. (5.23) verifies that layers in adjacent bins are a source of misclassification. This is visualized in the histograms as peaks in the distance bins surrounding the missed altitude bin. If a layer is missed by a network but is picked up in an adjacent bin, layer correction can still be obtained to some degree. The possible performance reduction from the shifted layer depends on the scale of the layer displacement, which will depend on the chosen network distance binning structure.

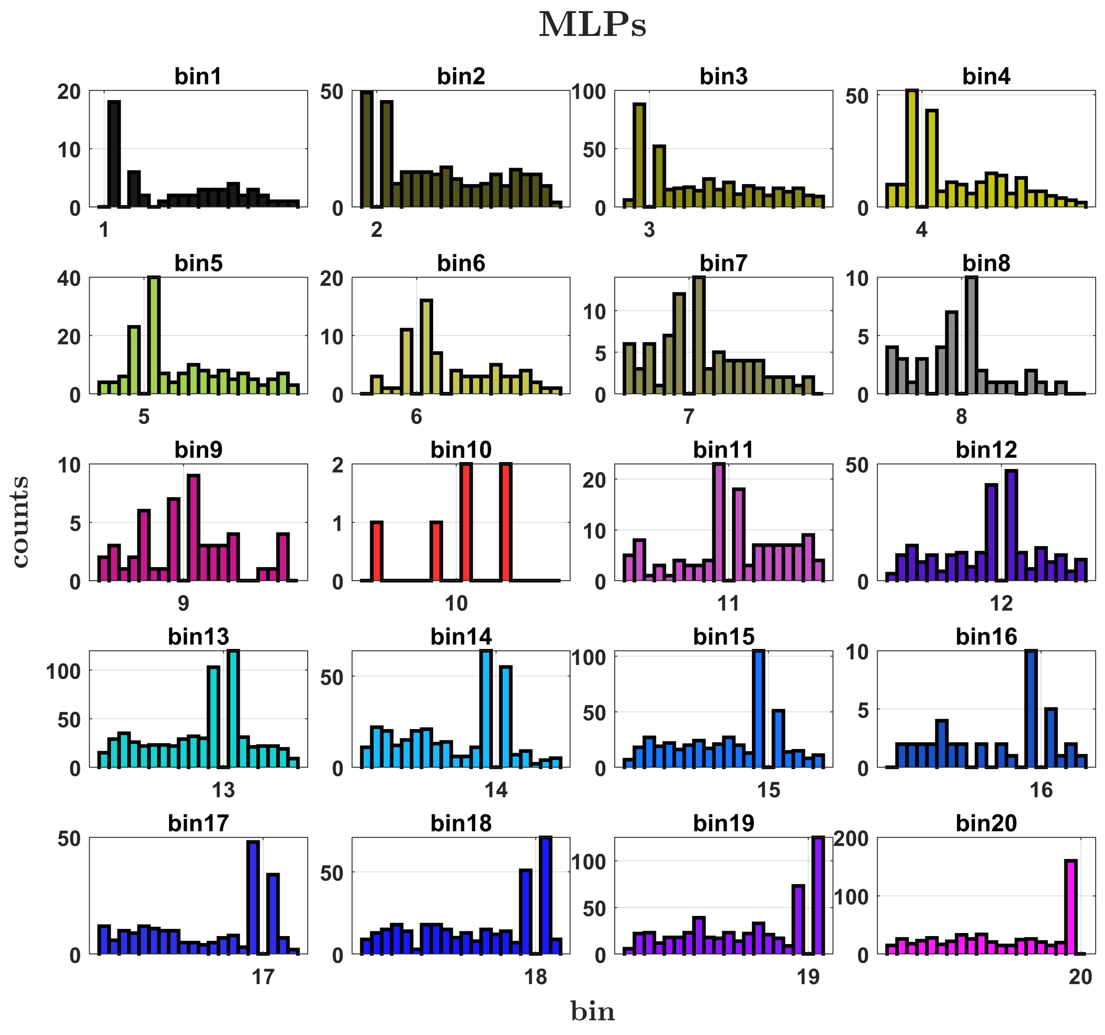


FIGURE 5.23: Distribution of all layers from each atmosphere which triggered a false alarm layer detections in each MLP.

Comparing the counts for each histogram between the MLP and CNN figures, we can see that the MLPs are much more prone to a false alarm from a layer in an adjacent bin. Combining these observation with the missed layer histograms, it clear that misclassification of SA correlations as layer ranges is dominated by two causes: missing low SNR layers and misclassification of layers into adjacent bins. These errors are more present in the MLPs than the CNNs, suggesting that CNNs are a more robust layer ranging network.

5.4.5 Counting networks for data fusion

The outputs of the trained layer ranging networks provides us with a distribution of probable layer locations, but it does not tell us how many layers there probably are. As we found when analyzing the network reposes in the previous section, there is lower accuracy for layers near to the boundaries of the network distance bins. This can lead to a potential data interpretation issue for the case of two ranging networks which occupy adjacent altitude bins both stating there is a layer of turbulence present. The decision to be made in this case, especially for large distance bins, is if this should interpreted as one layer which is somewhere between the bins, one layer which is in one of the bins, or if each bin has a layer present.

A possible solution to this is to introduce additional networks which estimate the number of layers in a given SA correlation. By having an understanding of how many layers are likely present, we can make an informed decision about the edge case described above. Additionally, since it is more likely that a weak layer will go undetected than a non-existent layer being detected, we can also search for weak layers if we come up short using all layers with present probability $> 50\%$. This sort of data fusion is potentially powerful when coupled with additional neural networks. These additional networks could receive the outputs of the counting and ranging networks as inputs, and then interpret a higher resolution layer position vector.

To test the viability of using counting networks, the same initial MLP and CNN models, given by Tab. (5.5) and Tab. (5.6), respectively, were trained on all 12,000 atmospheres. The only modification required from the ranging network layer architectures is to set the output layer size to 4 so that the network can classify the input as being either a 1, 2, 3, or 4 layer atmosphere. Networks were only trained on the full four quadrant SA correlation matrices as inputs. The number of training epochs was increased to 100. The results for the best network across a 5-fold validation are shown in Fig. (5.24).

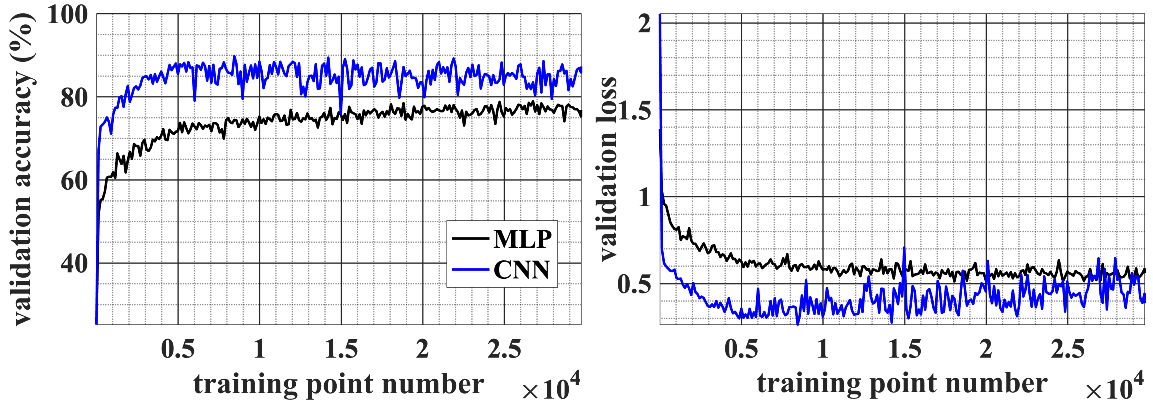


FIGURE 5.24: Layer counting MLP (black) and CNN (blue) training validation accuracy and loss for networks trained on four quadrant SA correlations. The CNNs converge to a more accurate model much faster than the MLPs. We also see the CNNs slowly beginning to over fit after the first 5000 training rounds, indicated by steadily decreasing validation accuracy and increasing validation loss with continued training.

The CNN is both faster at learning to count the layers, and the added abstraction of the data from the convolution layers produces more accurate estimations. The peak validation accuracy for the CNN is around 5000 training points with a value of 89.79%. The MLPs consist of more dropout layers and therefore did not begin to over fit, but only reached a maximum validation accuracy of 78.95%.

To test what the networks are getting wrong, we organized the trained network responses to all training data points in Fig. (5.25). In this form, we can see that the counting CNN tends to misclassify atmospheres as having one less layer and the counting MLPs often states there is an additional layer. If we take every misclassified atmosphere and generate a histogram of the lowest SNR for each misclassified point, as in Fig. (5.26), we see that have a layer with a low SNR drives counting network errors. While the trained CNN begins to sharply misclassify atmospheres containing a layer with SNR<0.5, the MLP has a more gradual misclassification behavior which starts mislabeling more inputs around SNR<2 then accelerates for SNR<1.

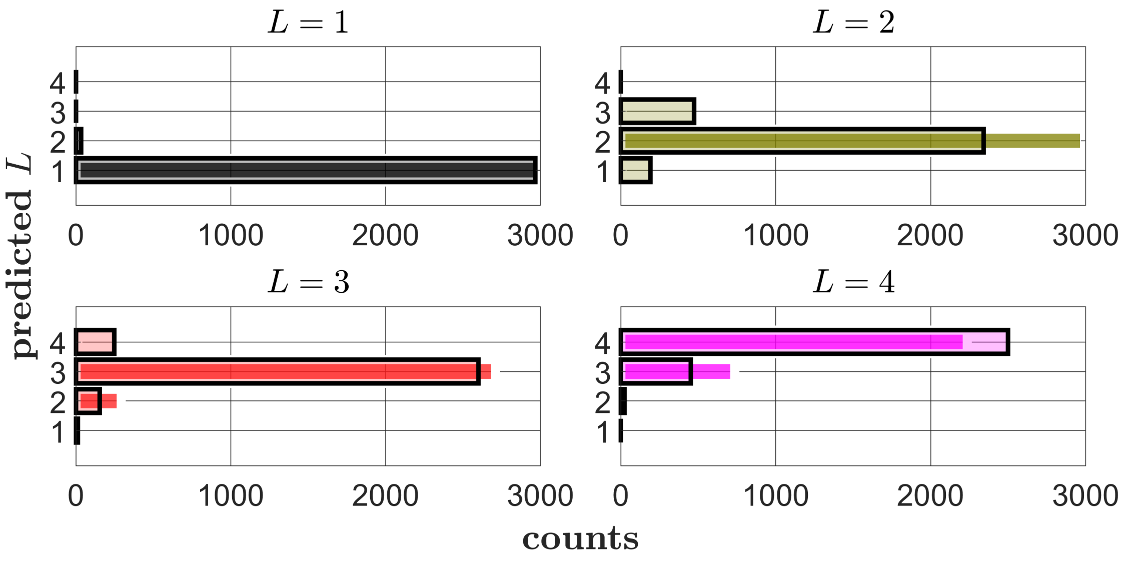


FIGURE 5.25: Histograms of all atmospheres interpreted by the counting CNN (white boarders) and the counting MLP (black boarders) grouped by the correct number of layers L.

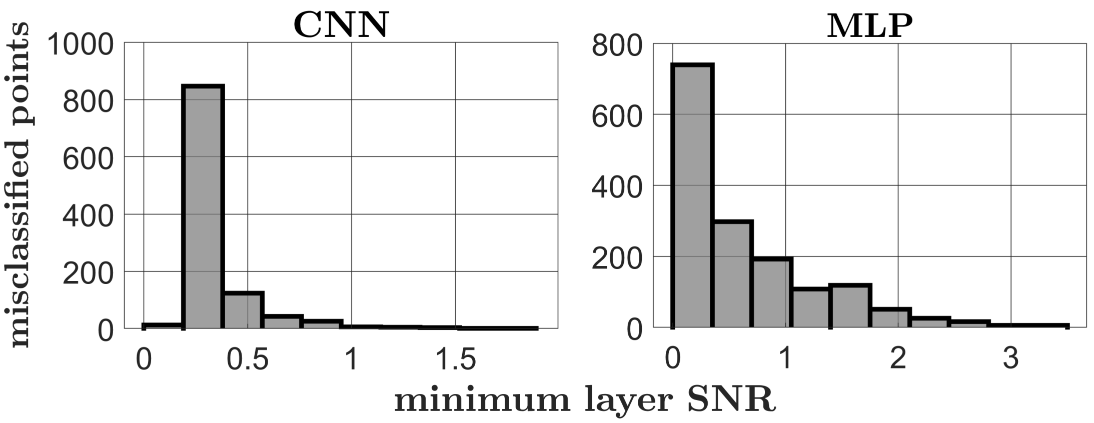


FIGURE 5.26: Histograms of the lowest layer SNR in each misclassified atmosphere for the counting CNN (left) and the counting MLP (right).

Chapter 6 MOIC using a measured turbulence profile

6.1 Turbulence decomposition model

Referring back to the software-based MOIC process outlined in Fig. (1.3), correction over a wide field is enabled by reconstructing the layers of turbulence so that the anisoplanatic PSF can be estimated. To reconstruct layers of turbulence, we need to establish a turbulence decomposition model. The decomposition model is a set of modal functions, $\Upsilon_{v_\ell}(\vec{r}_\ell)$, which can be summed together with different amplitudes, υ_{v_ℓ}, to recreate each layer:

$$\phi_\ell(\vec{r}_\ell) = \sum_{v_\ell}^{V_\ell} \upsilon_{v_\ell} \Upsilon_{v_\ell}(\vec{r}_\ell). \tag{6.1}$$

The most common choice of modes used in a pupil phase decomposition are orthonormal basis function sets, such as Zernike polynomials or disk harmonics (Milton 2009). These functions are chosen because they have convenient mathematical properties which make them easy to calculate and scale, they are orthonormal over the unit circle – making them effective at representing low order aberrations in optical systems – and contain shapes which can be recreated on a deformable mirror. In the context of recreating square layers of turbulence, however, it is more suitable to use the basis set which is orthonormal over the unit square: Legendre polynomials.

As the field of view of an optical system increases, larger and larger regions of turbulence must be reconstructed to capture the aperture projections along each line of sight of interest. As the region of interest in a layer of turbulence increases, the physical size of mid and high frequency turbules in the layer also grow in size relative to subapertures which must satisfy $s \leq r_0$. While standard basis functions are effective at representing low frequency components, such as tip, tilt, defocus, and astigmatism, mid and high frequency turbules are difficult to form without a large number of orthonormal polynomial modes. Scott and Hart recognized that due to self-similar geometry, the functions which have similar mid and high frequency structure as turbulence is other regions of turbulence. This suggests that a weighted sum of turbulence screen modes are a viable option for reconstructing large layers of turbulence (Scott 2021). The primary downsides of turbulence screen modes is that they are time consuming to form, making them unrealistic to use in active correction, and they contain high frequency shapes which cannot be recreated on a DM. Fortunately, neither of these downsides exclude turbulence screens from use in software-based MOIC since correction is applied offline without a DM.

Before presenting the tested decomposition models, low order mode ambiguity in an L-layer atmosphere must be addressed. Namely, when there is an L-layer decomposition estimated from tomographic data the distribution of modal components of power $(x^1, x^2, ..., x^{L-1})$ in each layer are ambiguous (Lloyd-Hart and Milton 2003). Therefore, when estimating the multi-layer decomposition from measurement data, the real tip, tilt, defocus, etc, from each layer will be distributed across all layers in a non-trivial way. One method to remedy the ambiguity of low order modes is to build reconstructors which are layer position dependent. We can then provide specific layers exlcusive access to specific low order modes, thereby forcing the modes of the order x^ℓ to the first ℓ layers. This makes it so all of the system tilt is optimized into the first layer, defocus and astigmatism into the first two layers, and so on. There are clear benefits to doing this for systems which achieve wide field correction with DMs; by knowing where all the tip and tilt will be reconstructed, we can conjugate a tip/tilt steering mirror to that altitude. However, it is not known if there is a benefit of forcing the low order modes to specific layers in the context of software-based MOIC reconstructions.

Regardless of the chosen model, each layer decomposition is defined by the coefficient vector

$$\vec{\upsilon}_\ell = \begin{bmatrix} \upsilon_{1_\ell} \\ \upsilon_{2_\ell} \\ \vdots \\ \upsilon_{V_\ell} \end{bmatrix}. \tag{6.2}$$

The multi-layer decomposition can then be formed by concatenating each individual layer decomposition vector:

$$\vec{\boldsymbol{\upsilon}} = \begin{bmatrix} \vec{\upsilon}_1 \\ \vec{\upsilon}_2 \\ \vdots \\ \vec{\upsilon}_L \end{bmatrix}. \tag{6.3}$$

By properly bookkeeping the location of each mode in $\vec{\boldsymbol{\upsilon}}$ with the correct modal function $\Upsilon_{v_\ell}(\vec{r}_\ell)$, each layer in the atmosphere can be formed from the multi-layer decomposition vector through the projection in Eq. (6.1). The specifics of the mode coefficient bookkeeping and the accuracy of the decomposition when using a finite number of modes depends on the chosen model.

6.1.1 Legendre modal decomposition model

To decompose a layer of turbulence into Legendre modes we use

$$\phi_\ell(\vec{r}_\ell) = \sum_{v_\ell}^{V_\ell} \upsilon_{v_\ell} \mathscr{L}_{v_\ell}(\vec{r}_\ell), \tag{6.4}$$

where $\mathscr{L}_{v_\ell}(\vec{r}_\ell)$ are the 2D Legendre polynomials as defined in Appendix (C). The amplitude coefficients in the decomposition are found by projecting the layer $\phi_\ell(\vec{r}_\ell)$ onto the v_ℓ^{th} mode with the inner product

$$\upsilon_{v_\ell} = \frac{1}{W_\ell^2} \int_{-1}^{1} \phi(\vec{r}_\ell) \mathscr{L}_{v_\ell}(\vec{r}_\ell) \, d\,\vec{r}_\ell, \tag{6.5}$$

where W_ℓ is the width of the region of interest defined by the bounds of $\vec{r}_\ell$. The discrete form of this calculation, whereby a modeled layer matrix $\boldsymbol{\phi}_\ell$ generated with the methods from Chapter (3) is projected onto the Legendre mode matrix $\mathbf{L}_{m;\ell}$, is found using

$$\upsilon_{v_\ell} = \frac{\Delta x^2}{W_\ell^2} \sum \sum \boldsymbol{\phi}_\ell \odot \mathbf{L}_{m;\ell}. \tag{6.6}$$

This multiplication requires that the phase and Legendre mode matrices are generated over the region $\vec{r}_\ell$ of width W_ℓ with identical pixel scale Δx.

One option for defining the layer decomposition with Legendre polynomials is to decompose each layer with the same set of modes. This means that in the multi-layer decomposition, each layer can contain different amounts of tip, tilt, defocus, and so on. When reconstructing the decomposition from tomographic measurement data with this method, the true distribution of low-order modes will be randomly distributed amongst each layer. We can control this by using a modified Legendre decomposition called the distributed Legendre (DL) mode set.

For the ℓ^{th} layer in the DL decomposition, the first Legendre polynomial used corresponds to the first mode of power x^ℓ. Using the Legendre polynomial notation from Appendix (C), the first two modes in the decomposition are of order x^1, modes 3-5 are of order x^2, and so on. Thus, for a multi-layer DL decomposition, where the first layer has V_1 modes, the total number of modes used in the decomposition of layer $\ell > 1$ can be written as

$$V_\ell = V_1 - \sum_2^\ell \ell. \tag{6.7}$$

When using tomographic measurement data to reconstruct the DL decomposition, we can be certain that system tip and tilt will be confined to the first layer, defocus and astigmatism to the first two layers, and so on.

6.1.2 Turbulence screen modal decomposition model

The turbulence screen decomposition is written as

$$\phi_\ell(\vec{r}_\ell) = \sum_{v_\ell}^{V_\ell} \upsilon_{v_\ell} \varphi_{v_\ell}(\vec{r}_\ell). \tag{6.8}$$

The turbulence screen modes, $\varphi_{v_\ell}(\vec{r}_\ell)$, must be numerous enough and sufficiently diverse in eddie scales to adequately cover the range of possible structures in any given layer $\phi_\ell(\vec{r}_\ell)$. This is accomplished by using turbulence screen modes with a range of different r_0 and L_0 values. Variations in these parameters diversifies the scales of turbules in the mode set, suggesting greater likelihood of matching the structures of ϕ_ℓ to specific modes in the decomposition. Once a distribution of different r_0 and L_0 values for each mode has been chosen, each turbulence screen modal function matrix $\boldsymbol{\varphi}_{v_\ell}$ can be generated and the inner product

$$\upsilon_{v_\ell} = \frac{\Delta x^2}{W_\ell^2} \sum\sum \boldsymbol{\phi}_\ell \odot \boldsymbol{\varphi}_{v_\ell} \tag{6.9}$$

is taken to find the mode amplitudes for the real layer undergoing decomposition. Since there are no specific low order modes for the decomposition model, each layer in a multi-layer decomposition has identical mode indexing $v_\ell = v \in [1,V] \ \forall \ \ell$.

An important note in regards to the practicality of using turbulence screen modes is the amount of memory that is required. The size of each turbulence screen matrix, $\boldsymbol{\varphi}_v$, is different at each layer distance. For distant layers, this can be a matrix which exceeds $25,000 \times 25,000$ pixels for a highly sampled layer. If this is the case, storing hundreds of these matrices may be cumbersome or impossible. Thus, we suggest storing the screens as the plane wave elements needed to generate the layers over any region $\vec{r}_\ell$ from Eq. (3.4). The necessary components are the random vectors $\vec{k}$, $\vec{\theta}$, $\vec{\psi}$, and $\vec{A}$, as described in Sec. (3.1.4). For a layer which is the superposition of N plane waves, this means the layer is defined by 4 $N \times 1$ vectors – which more memory efficient than $\geq 25,000^2$ elements which make up the phase screen matrix.

While this technique enables turbulence screen modal decomposition with limited computer memory resources, it is computationally inefficient. By storing the turbulence layers in coefficient form we must generate the matrix $\boldsymbol{\varphi}_v$ each time we want to use it for something. When this corresponds to hundreds of modes in multiple layers in the atmosphere, this requires the repeated generation of thousands of layers of turbulence which is redundant and time consuming.

6.1.3 Mixed mode decomposition model

Even though the turbulence screen mode set does not contain individual modes which correspond to the terms $(x^1, x^2, ...)$, it is possible to resolve the L-layer ambiguity in a turbulence screen based model with a mixed mode (MM) decomposition. The MM decomposition takes the turbulence screen modal decomposition and injects Legendre polynomials of the correct order at each layer. Adapting this idea to the notation of Eq. (6.1), the v_ℓ^{th} mixed mode is written as

$$\Upsilon_{v_\ell}(\vec{r}_\ell; L) = \begin{cases} \mathscr{L}_m(\vec{r}_\ell) & 1 \leq v_\ell \leq M_{\ell;L} \\ \varphi_n(\vec{r}_\ell) - \sum_{m=1}^{M_{\ell;L}} \alpha_{m;n,\ell} \mathscr{L}_m(\vec{r}_\ell) & M_{1;L} < v_\ell \leq V_{\ell;L} \end{cases}, \tag{6.10}$$

where

$$\alpha_{m;n,\ell} = \frac{1}{W_\ell^2} \int_{-1}^{1} \varphi_n(\vec{r}_\ell) \mathscr{L}_m(\vec{r}_\ell) \, d\,\vec{r}_\ell \tag{6.11}$$

is the amplitude of the m^{th} Legendre mode in the n^{th} turbulence screen mode. We subtract all of the Legendre modes that have been injected into the decomposition from each turbulence mode to ensure that the low order components which naturally occur in the modeled turbulence do not share the decomposition with any of the pure Legendre modes. The indices of the Legendre modes which are included in the ℓ^{th} layer decomposition are $m \in [1, M_{\ell;L}]$, where

$$M_{\ell;L} = \frac{L(L+1) - \ell(\ell+1)}{2}. \tag{6.12}$$

Therefore, the number of Legendre modes is different for each layer $\ell \in [1, L]$ and changes for a different number of total layers, L. The turbulence screen modes are $n \in [1, N]$, which does not change with ℓ and L, resulting in MM decomposition indices $v_\ell \in [1, V_{\ell;L}] = [1, N + M_{\ell;L}]$.

Eq. (6.12) is formulated so that when used with the Legendre polynomial notation in Appendix (C) – whereby $m = 0$ is piston, $m = 1$ is tilt, and so on – the correct number of low order terms are decomposed into the ℓ^{th} layer of an L-layer atmosphere. An example of the distribution of Legendre and turbulence screen modes for an $L = 4$ layer MM decomposition is outlined in Tab. (6.1).

To describe each $\boldsymbol{\phi}_\ell$ discretely in terms of a decomposition of mixed modes, the layer of phase must be projected onto each mode Υ_{v_ℓ} to find the corresponding amplitude coefficients υ_{v_ℓ}. When the mode corresponding to v_ℓ is a Legendre polynomial, Eq. (6.6) is used to determine the MM coefficient. When v_ℓ corresponds to a turbulence screen mode, Eq. (6.9) is used with the additional step of subtracting the first $M_{1;L}$ Legendre modes from $\varphi_n(\vec{r}_\ell)$.

ℓ	Legendre	turbulence	$(x^1,y^1)_\ell$	$(x^2,y^2,xy)_\ell$	$(x^3,y^3,x^2y,xy^2)_\ell$
1	$v_1 \in [1,9]$	$v_1 \in [10,N+9]$	✓	✓	✓
2	$v_2 \in [1,7]$	$v_2 \in [8,N+7]$	×	✓	✓
3	$v_3 \in [1,4]$	$v_3 \in [5,N+4]$	×	×	✓
4	N/A	$v_4 \in [1,N]$	×	×	×

TABLE 6.1: Mixed mode decomposition using Legendre and turbulence screen modes in a 4 layer atmosphere. The Legendre modes column indicate how many of the mixed modes are Legendre polynomials. The turbulence modes then indicates the number of phase screens which compose the rest of the modes such that $v_\ell \in [1, V_{\ell;L}]$. The remaining columns indicate the low order terms, which have been removed from all turbulence screen modes, that are explicitly contained as Legendre modes in the decomposition of layer ℓ.

6.2 Multi-layer turbulence reconstruction

Having defined several multi-layer turbulence decomposition models, we are ready to use a known turbulence profile and warp map measurements for tomographic turbulence reconstruction. The known turbulence profile tells us the number of layers, L, and their distances – which constrains their position vectors $\vec{r}_\ell$. If we assume that the warp map measurements $\mathbf{w}$ are a direct result of turbulence defined by the multi-layer decomposition vector $\vec{\boldsymbol{\upsilon}}$, we can write the relationship between the decomposition coefficients and warp maps as

$$\mathbf{I}\vec{\boldsymbol{\upsilon}} = \vec{\mathbf{w}}, \tag{6.13}$$

where the warp map elements have been vectorized. The influence matrix **I** in the above relationship is a linear operator which maps each coefficient in each layer of the decomposition to the measured system warp map. In practice, however, we only have knowledge of the warp map and we want to use it to determine the decomposition vector. Inverting Eq. (6.13), the relationship we need to solve for becomes

$$\vec{\boldsymbol{\upsilon}} = \mathbf{I}^{-1}\vec{\mathbf{w}}. \tag{6.14}$$

Thus, to reconstruct the layers from warp map measurements, we must generate the influence matrix and then solve for its inverse.

6.2.1 Multi-layer influence matrix generation

The structure of the influence matrix can be deduced from the shape of the decomposition vector and the number of warp map elements. The decomposition vector consists of the mode coefficients υ_{v_ℓ}, where each layer has a total of V_ℓ modes defined in the decomposition model. Using the notation

$$V = \sum_{\ell=1}^{L} V_\ell, \tag{6.15}$$

the decomposition vector is of dimension $V \times 1$. The warp map contains the tip and tilt coefficients for each subfield $a \in [1, A]$ in each subaperture $b \in [1, B]$. When vectorized, the warp maps produce a vector of length $(2AB) \times 1$. To satisfy the mapping from the decomposition elements to the warp map elements in Eq. (6.13), the influence matrix must be of size $(2AB) \times V$.

By knowing the dimensionality of the influence matrix, we can understand the operation it is performing. Each column, indexed v_ℓ, is mapping the mode Υ_{v_ℓ} to a system warp map vector of size $(2AB) \times 1$. If each layer and mode are independent, which we assume is true, then the matrix multiplication in Eq. (6.13) states that the system warp map is the sum of the responses to each individual mode from each layer. Thus, because of layer and mode independence, the influence matrix can be constructed one column at a time using the warp map responses for each mode at each layer. This process is outlined in Fig. (6.1). Once the influence matrix is formed for a particular multi-layer decomposition model, the inverse of the influence matrix can be computed using a Moore-Penrose inversion.

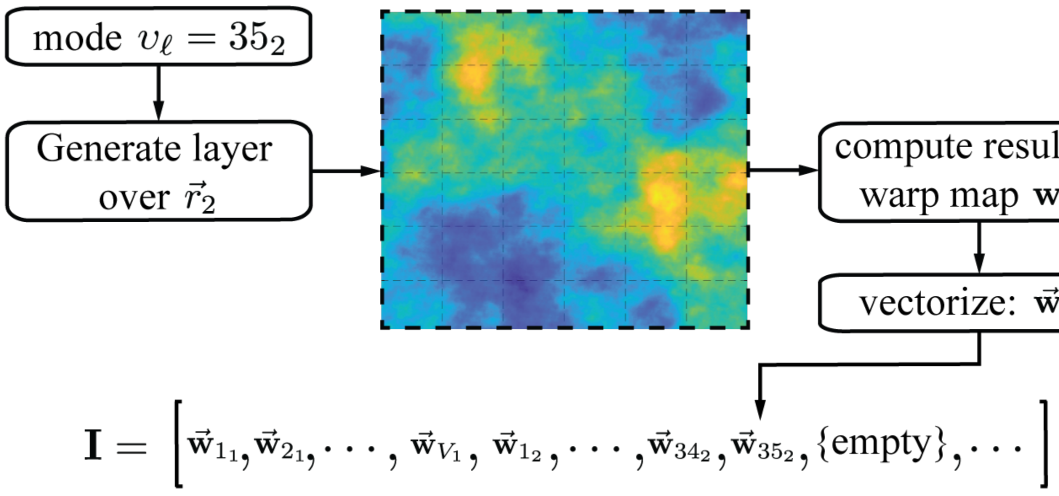

mode $\upsilon_\ell = 35_2$
Generate layer over $\vec{r}_2$
compute resulting warp map $\mathbf{w}_{35_2}$
vectorize: $\vec{\mathbf{w}}_{35_2}$
$\mathbf{I} = \left[\vec{\mathbf{w}}_{1_1}, \vec{\mathbf{w}}_{2_1}, \cdots, \vec{\mathbf{w}}_{V_1}, \vec{\mathbf{w}}_{1_2}, \cdots, \vec{\mathbf{w}}_{34_2}, \vec{\mathbf{w}}_{35_2}, \{\text{empty}\}, \cdots\right]$

FIGURE 6.1: Example for the generation and storing process of a turbulence screen mode as an influence matrix response. The example is for the 35th mixed mode of the second layer in the atmosphere, requiring that the mode is generated over the region $\vec{r}_2$.

The biggest challenges with generating an influence matrix are identical to those of most software-based MOIC processes: computer memory and computation time. Each column in the influence matrix requires performing a system ray trace to obtain the mode-specific warp map. If using hundreds of modes per layer to obtain highly structured reconstructions, this corresponds to hundreds or thousands of warp map calculations – making it unreasonable to generating the influence matrix during real data processing. Therefore, for a system which uses software-based MOIC, it is best to use the developed simulation environment to calculate every possible inverse influence matrix that could be needed and store them beforehand. For reference, the computer which simulated the training data in Chapter (5) took ~ 5 hours to form 4 layer reconstructor with 250 modes per layer.

When using layer ranging neural networks, like those developed in Chapter (5), layer positions are estimated at the center of each network distance bin rather than at a precise distance. This makes preallocation of system influence matrices a more manageable task. By only reconstructed layers at the centers of each network distance bin, the number of layers which can be reconstructed and their positions are constrained. In the case of decomposition models which do not manage low order mode distribution, preallocation is straightforward: model each mode at each ranging network bin center and store the warp maps. If a model which uses low order mode distribution is employed, preallocation of the system influence matrices is much more involved since the number of modes and their distributions at each layer changes with the number of layers and their ordering.

6.2.2 Test case: 4-layer atmosphere reconstructions

Using the presented decomposition models and the influence matrix generation process, we can use the four layer atmosphere model defined by Tab. (6.2) as a test case for reconstructor performance evaluation. We use the optical system model defined by Tab. (5.4) as our test system. Since this is the same system we trained layer ranging networks for, we are able to use the network distance binning structure defined in Fig. (5.9) when building influence matrices. The measured full SA correlation matrix and the resuling turbulence profile predictions from the trained networks for the test atmosphere are shown in Fig. (6.2). Each tested reconstructor model in this section uses the same preallocated real layer matrices and resulting system warp maps to ensure the results are directly comparable.

Since the reconstruction of the layer coefficients from Eq. (6.14) is a least squares fit to the warp map vector given the multi-layer influence matrix model, each reconstructed layer can appear visually different from the actual layers of turbulence. This makes comparing each reconstructed layer to each actual layer uninformative – especially since we are only concerned with the effectiveness of the reconstruction as a method of compensating for field-specific aberrations. Thus, to score reconstructor performance we need to use a metric which considers the tomography of the reconstruction rather than just the reconstruction of each individual layer.

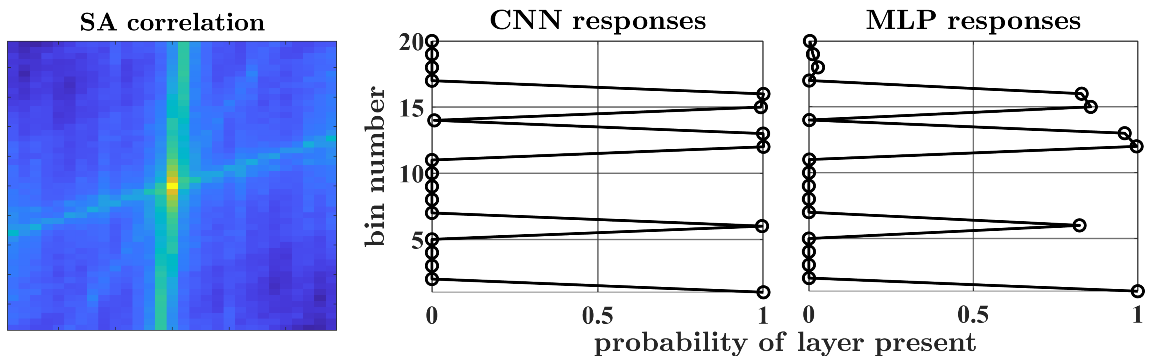


FIGURE 6.2: Left: SA correlation matrix computed by averaging 100 ms of slope data from the reconstructor test atmosphere specified in Tab. (6.2). Middle: reponses to the SA correlation matrix from the ranging CNNs trained in Chapter (5). Right: responses to the SA correlation matrix from the trained ranging MLPs.

	h (km)	bin #	bin center (km)	r_0 (m)	L_0 (m)	SNR
layer 1	0.15	1	0.18	0.3	15	24.79
layer 2	1.25	6	1.27	0.75	50	0.88
layer 3	6.0	12	5.49	0.5	125	1.78
layer 4	10.5	16	11.46	1.0	250	0.173

TABLE 6.2: Parameters for the four layer atmosphere simulated to test different reconstruction models. The bin # corresponds to the altitude bin in the trained ranging network model developed in Chapter (5) for the 2.5' field of view system prescribed in Tab. (5.4) – allowing us to specify the layer SNRs.

We can use simulation tools developed in Chapter (3) to estimate the tomographic error in the reconstruction as a function of field. To do this, the field specific pupil phase matrices from Eq. (3.58) and Eq. (3.59) can be computed for the real and reconstructed atmospheres. The field-specific pupil phase matrix resulting from the real atmosphere is denoted $\mathbf{\Phi}_a$, and the one from the reconstructed layers is denoted $\bar{\mathbf{\Phi}}_a$. Once we have both pupil phases we subtract their means:

$$\mathbf{\Psi}_a = \mathbf{\Phi}_a - \langle \mathbf{\Phi}_a \rangle, \qquad \bar{\mathbf{\Psi}}_a = \bar{\mathbf{\Phi}}_a - \left\langle \bar{\mathbf{\Phi}}_a \right\rangle. \tag{6.16}$$

We do this because phase piston does not change the incoherent point spread function which is the function used in the software MOIC process. Next, the root mean squared difference between the real atmosphere and the reconstructed model is evaluated:

$$\sigma_\phi(\theta_a) = \sqrt{\frac{1}{N_p}\sum(\bar{\mathbf{\Psi}}_a - \mathbf{\Psi}_a)^2}, \tag{6.17}$$

where N_p is the number of elements inside the pupil over which $\bar{\mathbf{\Psi}}_a$ and $\mathbf{\Psi}_a$ are defined. $\sigma_\phi(\theta_a)$ is the tomographic error and is in units of radians of wavefront error. It tells us how close the reconstructed phase along field angle θ_a is to the actual phase which produces the system warp map.

If we assume that the reconstruction can be perfectly implemented during the correction phase, we can approximate the Strehl ratio after correction using

$$\text{Strehl}(\theta_a) = e^{-\sigma_\phi^2(\theta_a)}. \tag{6.18}$$

The Strehl ratio is a convenient supplemental quality metric to the tomographic error since it directly states how close the corrected performance is to that of the diffraction limited system – i.e. Strehl = 1. While there are deficiencies in the correction process which make the values computed from Eq. (6.18) optimistic, it provides an estimate of the upper threshold of correction which can be extracted given the tomographic error computed by Eq. (6.17).

Comparison of reconstructors with and without forced low order mode distribution

The first reconstructor model test conducted was to compare the models which contain forced low order mode distributions to those which do not. The reconstructors were modeled at the exact values of h in Tab. (6.2) to remove any performance variations that might occur from shifting the reconstruction. The tomographic error and ideal Strehls computed along each subfield center are shown in Fig. (6.3).

Fig. (6.3) immediately demonstrates that the Legendre mode model performs better for this atmosphere than the turbulence screen mode model. The Legendre mode reconstructions most notably outperform the turbulence mode models at the edges of the field. We can also see that while the MM and pure turbulence mode models have a similar distribution of performance as a function of field, the MM model has several fields with lower tomographic error. The DL model appears to have slightly better performance than the pure Legendre mode model, but the performance differences are noticeably more minor than in the turbulence screen mode case.

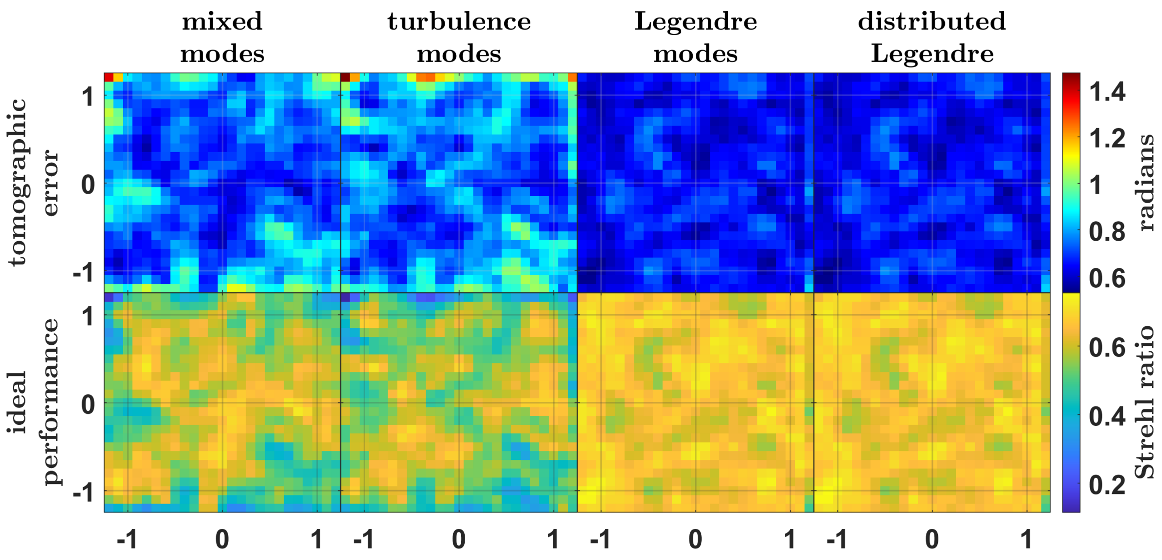


FIGURE 6.3: Tomographic error along each subfield angle θ_a for the four presented reconstructor models on the 4 layer atmosphere in Tab. (6.2). The horizontal and vertical axes are shown in units of arcminutes. Each reconstruction model reconstructed the layers at the exact model distances. The MM model and turbulence mode model both used $N = 250$ turbulence screen modes, the pure Legendre mode model used 250 modes for each layer, and the DL model used $V_1 = 250$ modes.

Distributed Legendre mode reconstruction test

Next we took a closer look at the Legendre mode reconstruction model. In this test, the influence matrix was setup to reconstruct each layer at the ranging network bin center rather than the exact layer distances.

The first tested reconstructor was the 4 layer DL model, the results of which are plotted in the second row of Fig. (6.4). Visually the reconstructed layers do not look anything like the real layers. However, if we project the system pupil through each layer along a line of sight and add up the phases, as shown in Fig. (6.5), we can see that the reconstructed phase tomography is in agreement with the real atmosphere at the pupil. Taking the difference between the real and reconstructed pupil phases in each case shows that the phase residuals are dominated by high frequency turbules. Given that the tomographic error of the model remains low, as shown in Fig. (6.6), the high frequency residual appears to have less impact on performance than the low order turbulence structures which are reconstructed effectively by the Legendre modes.

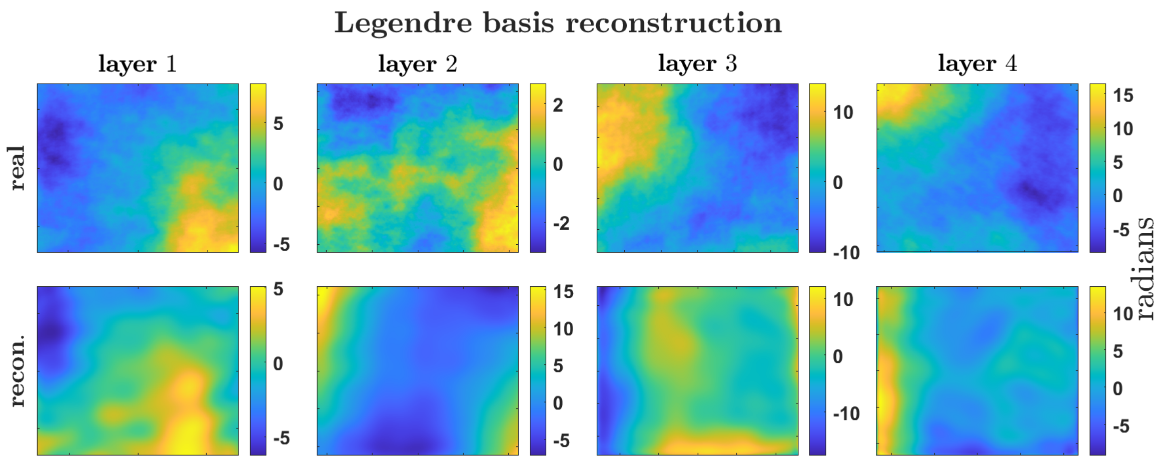


FIGURE 6.4: Modeled layers (top row) and DL model reconstruction layers using $V_1 = 250$ (bottom row). The layers are shown with 100 pixels cropped from each side to help with visualization since Legendre modes tend to have large values at the edges. One possible way to mitigate the effect of the large values at the edge of the reconstructions is to include a small buffer region around the reconstructed layers outside of where the system pupil will land.

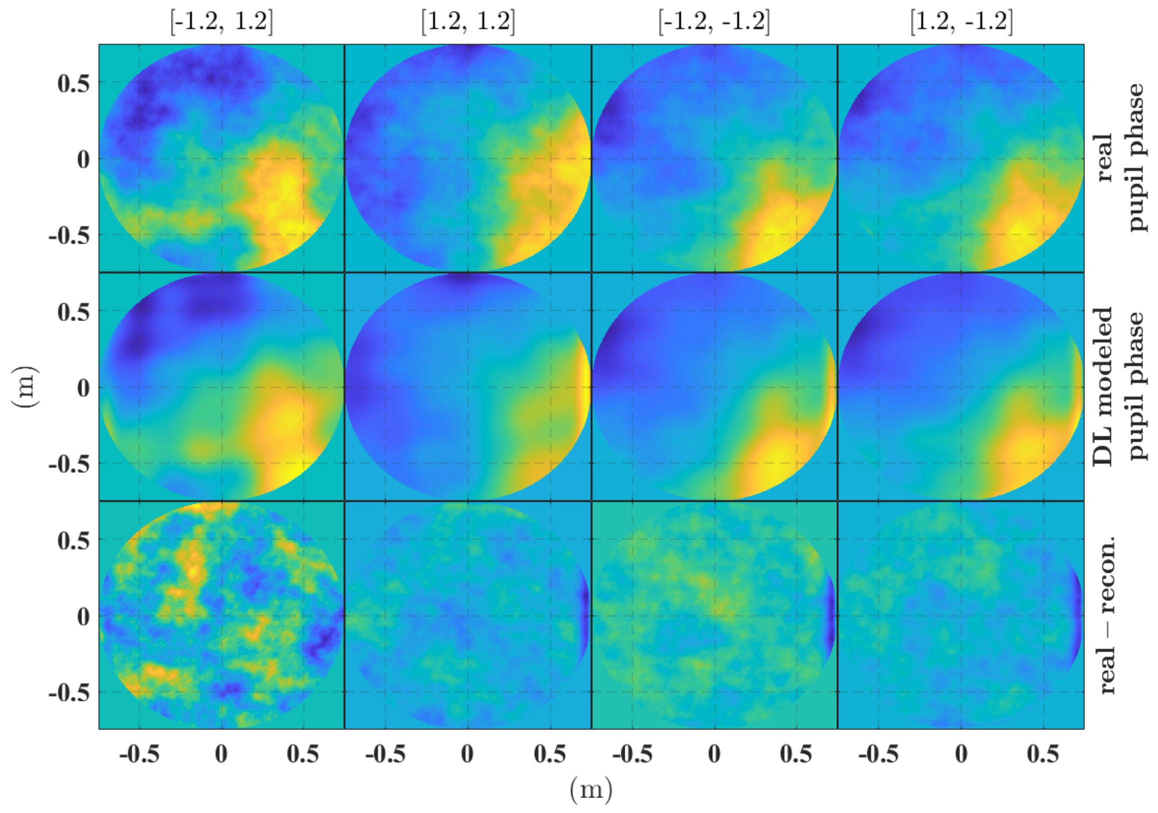


FIGURE 6.5: Top: pupil phase accumulated from all layers along each specified field angle $[\xi_a, \zeta_a]$ shown in units of arcminutes. From left to right these correspond to the top left, top right, bottom left, and bottom right subfields for the system. Middle: same field-specific pupil phases using the DL model reconstructed layers. Bottom: difference between the real and DL model phases.

To analyze the effects of missing layers of turbulence, DL reconstructions at the binned altitudes for 3, 2, and 1 layer models were generated. The performance analysis of the reconstructions are shown in Fig. (6.6). The 3 layer reconstruction model is assumed to have missed the lowest SNR layer – i.e. layer 4 from Tab. (6.2). We can see that this produces a slight decrease in performance from the 4 layer model. Even though the layer is high altitude and will therefore have strong anisoplanatism, it only has $r_0 = 1$ m which will correspond to weak phase perturbations in the 1.5 m diameter entrance pupil modeled here. The 2 layer reconstruction model is assumed to have missed the two lowest SNR layers, layer 4 and layer 2. This produces another drop in performance across the field as the missed layers are not sufficiently compensated at the two remaining reconstructed layers. The 1 layer reconstruction is assumed to be missing layers 2-4, where we see a significant increase in tomographic error – particularly at the edges of the field. Since layer 3 contributes a significant amount of phase aberration and anisoplanatism, missing it in the reconstruction causes the greatest drop in reconstructor performance when compared to dropping any of the other layers from consideration.

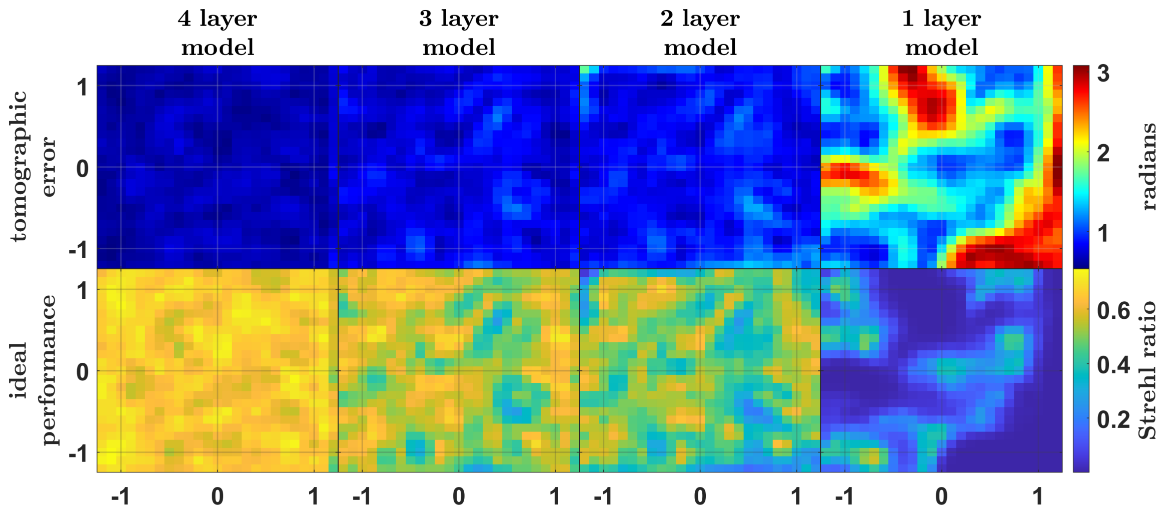


FIGURE 6.6: Analysis of the field-specific tomographic error on the 4 layer atmosphere from Tab. (6.2) when using 4, 3, 2, and 1 layer DL reconstruction models. The reconstructed layers are set to the ranging network bin center distances rather than the true layer distance. The missed layers in each reconstruction model correspond to the layers with the lowest measurement SNR.

Mixed mode reconstruction test

The last reconstructor model test conducted used 4, 3, 2, and 1 layer MM reconstructions at the ranging network bin center distances. First, from the 4 layer MM reconstruction results in Fig. (6.7) we can see that the real and reconstructed layers appear more visually similar than the Legendre modes. Their similarities are made more apparent when we account for the fact that the MM model has low-order aberrations (LOA) forced into specific layers. Thus, by subtracting the LOAs from each layer we can see that the MM model does actually reproduce something that looks like each individual layer of real turbulence.

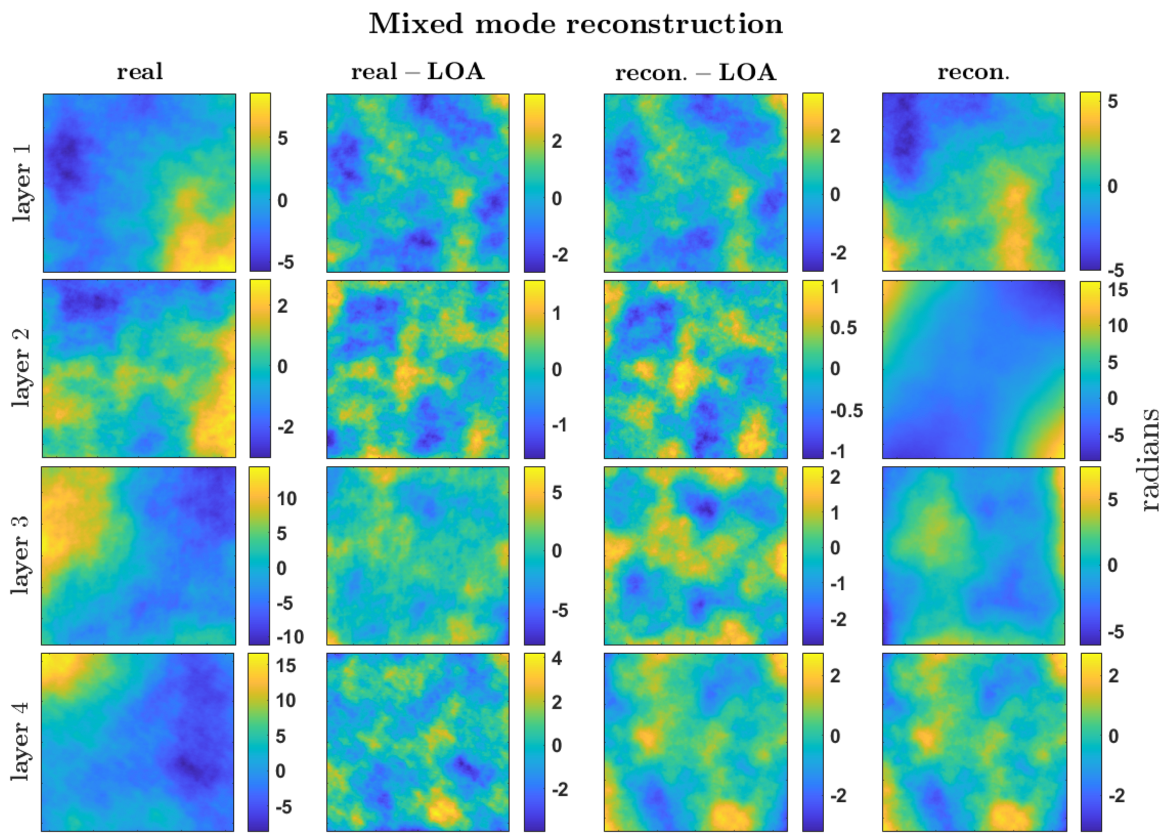


FIGURE 6.7: The left-most column shows the real modeled layers. The second column shows the same layers but with the first 9 orthonormal 2D Legendre modes subtracted. The right-most column shows the layers which were reconstructed using the MM model with $V_{1;4} = 259$. In these plots, the presence of low order Legendre modes like tip/tilt/defocus are visible in the bottom layers. The third column shows the MM reconstructions with the same low order Legendre modes removed. We can see that the underlying structure of turbules from the MM reconstruction is strikingly similar to real modeled layers.

As discussed in the DL mode reconstruction test results, the structure of each reconstructed layer is not as important in the context of software-based MOIC as the tomography of the field-specific pupil phase. To analyze how this is managed by the MM reconstruction, we plot the field-specific pupil phases for the four corner subfields and the resulting deviation from the real pupil phases in Fig. (6.8). Unlike the the DL reconstruction, which has residuals which appear to be dominated by high spatial frequencies, the MM model residual has a very high frequency background with some missed tubules superimposed. These mid to large frequency phase residuals are likely the culprit for the lower performance of the MM model compared to the DL model found in Fig. (6.3). It may be possible to mitigate the large structure residuals using more turbulence screen modes, but that will come at the cost of increased computer memory and processing time.

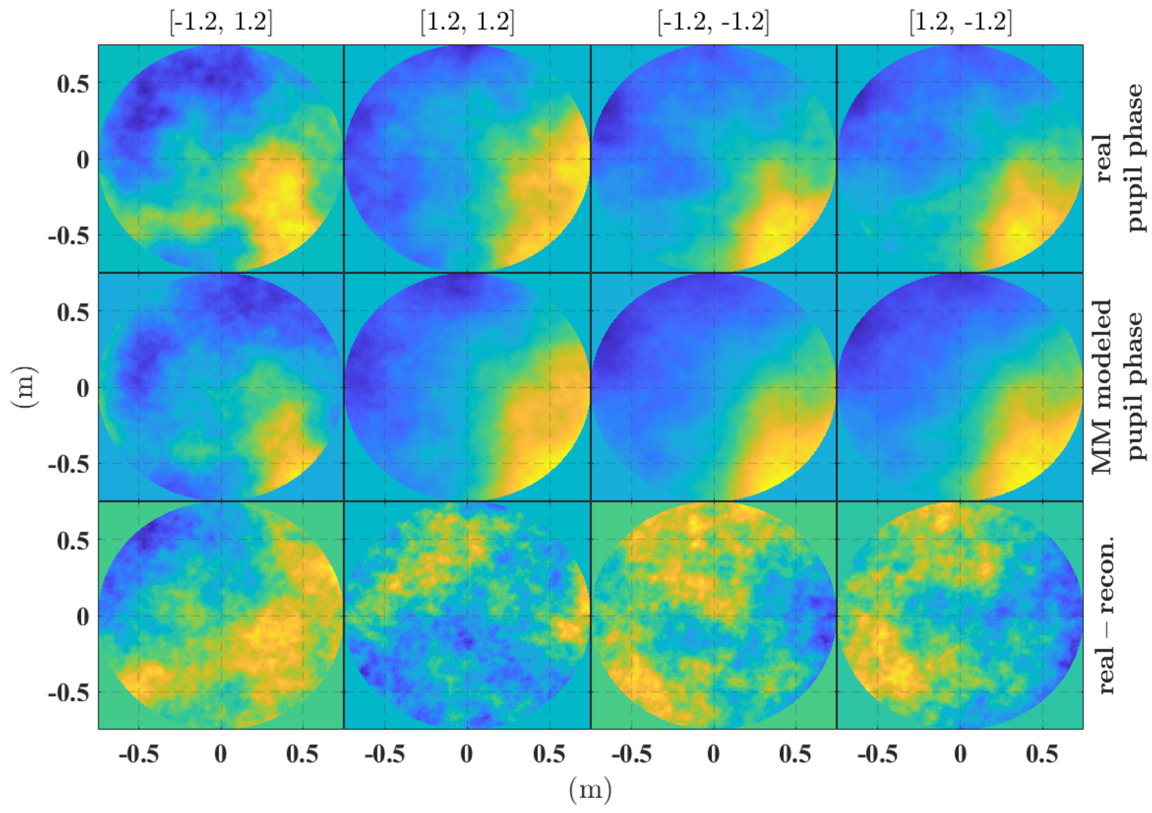


FIGURE 6.8: Top: identical to the top row of Fig. (6.5). Middle: same field-specific pupil phases using the MM model reconstructed layers. Bottom: difference between the real phase and the MM model reconstructed phase.

To analyze how the MM model is affected by missed layers, the same SNR-based layer dropout test used on the DL modes was repeated. The performance of the 4, 3, 2, and 1 layer MM reconstructors are shown in Fig. (6.6). Like the DL model, tomographic error increases when a layer is missed during reconstruction. The decrease in performance is once again largest when going from the 2 layer to the 1 layer reconstruction since it corresponds to missing the strong distant layer of turbulence in the profile.

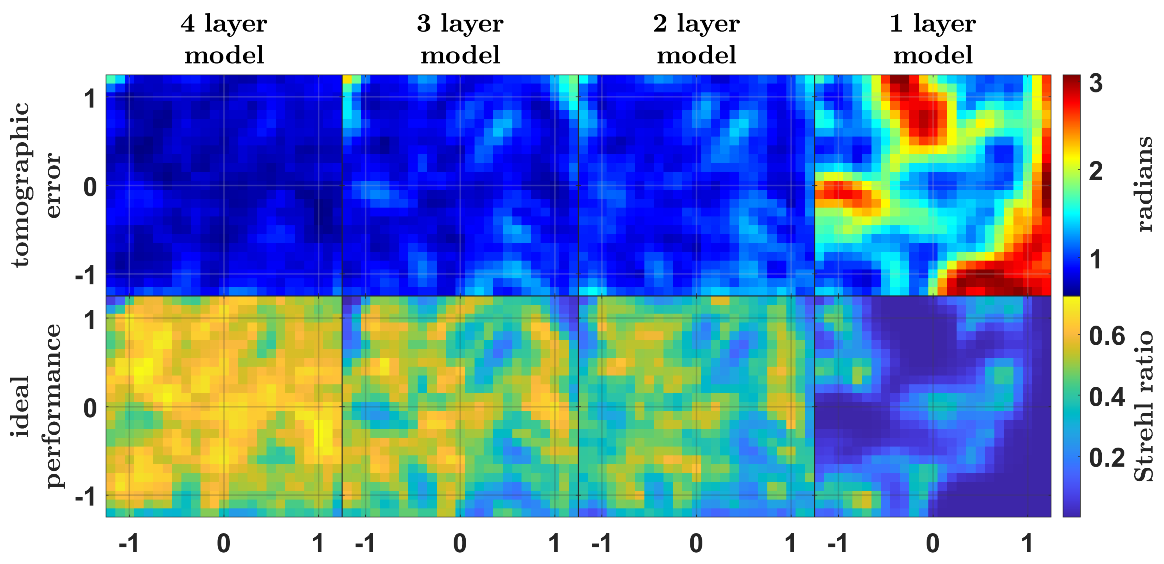


FIGURE 6.9: Analysis of the field-specific tomographic error on the 4 layer atmosphere from Tab. (6.2) when using 4, 3, 2, and 1 layer MM reconstruction models. The reconstructed layers are set to the ranging network bin center distances rather than the true layer distance. The missed layers in each reconstruction model correspond to the layers with the lowest measurement SNR.

Reconstructor test summary

Having built multiple different layer reconstruction techniques for the simulated 4 layer atmosphere, we can use the results to directly compare the performance of each model. By taking the average of the ideal performance Strehl matrices for each model and plotting them side-by-side, we can see how well each reconstructed model could perform on average across the entire field. The results are shown in Fig. (6.10).

The most immediate insight from Fig. (6.10) is that the greatest source of tomographic error is from missing layers in the reconstruction model. We see that even though layer 4 is weak, defined by $r_0 = 1$ m and SNR$= 0.173$, missing it from the reconstruction drops the average Strehl by about 10% in both the DL and MM models. Even though layer 2 is stronger than layer 4, missing it from the reconstruction model produces a smaller decrease in performance. This is because layer 2 is both closer to the optical system – resulting in less anisoplanatism – and nearer to other layers than layer 4 – thereby allowing the remaining layers included in the model to absorb some of the missed

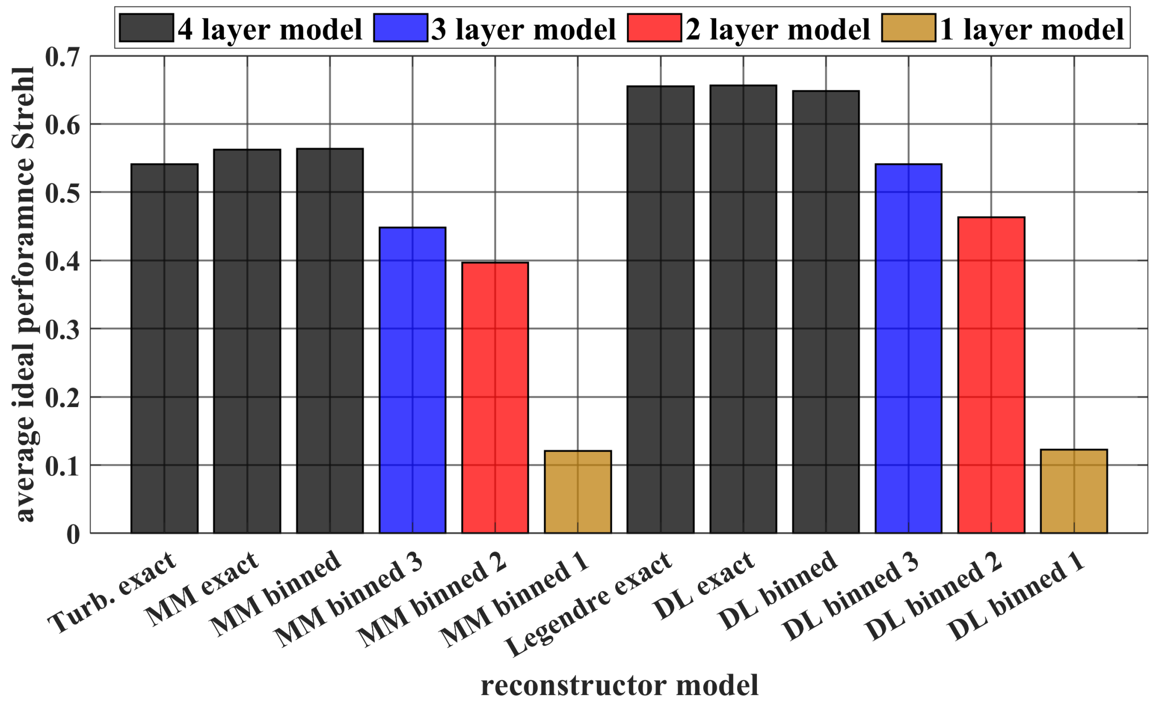


FIGURE 6.10: Average ideal performance Strehl from perfect compensation of the reconstructed phase across the 2.5' system FFOV for each reconstructor model. We can see that the greatest source of error comes from missing layers. We can also deduce that Legendre-based models out perform the turbulence screen mode models when using ~ 250 modes per layer in a multi-layer reconstruction.

phase components. The largest drop in performance occurs when going from the 2 layer model to the 1 layer model. This is because the model no longer accounts for the strong layer 6 km out in front of the system which dominates the system anisoplanatism.

Interestingly, the performance of the MM and DL 1 layer models is approximately the same on average. By comparing the tomographic error maps for the two 1-layers models in Fig. (6.6) and Fig. (6.9), we also find that they have almost identical field-dependent performance mapping. While the 4 layer reconstructions for the DL and MM models, shown in Fig. (6.4) and Fig. (6.7), respectively, look different layer-by-layer, we find that when these two models reconstruct a single layer the results are almost identical. The results for the 1 layer reconstruction for both the MM and DL models, as well as their difference, are shown in Fig. (6.11).

Fig. (6.10) reinforces the previous observation that Legendre models outperform turbulence

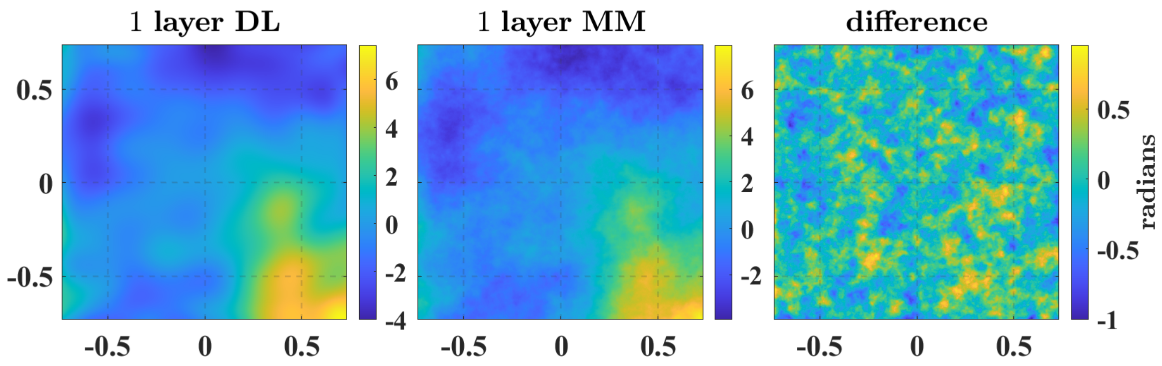


FIGURE 6.11: Direct comparison of the single reconstructed layer at $h = 0.18$ km for the 1 layer DL and MM models. Note that when only one layer is used, the DL model is just a Legendre decomposition and the MM model is an unmodified turbulence screen model. The difference between the reconstructions is characterized by weak high frequency components which explains why the tomographic error and ideal Strehl for these models is nearly identical.

screen models for this test case. The degree to which this is true likely depends on the number of turbulence screen modes used. If a turbulence screen model is used, distributing the low-order terms with a MM could provide a slight improvement in performance. Distributing the low-order terms in a Legendre model seems to have less of an impact on performance. It is worth noting that since these conclusions were reached based on a single test case, additional testing on a range of different atmosphere is needed to make concrete conclusions about decomposition model performance.

Lastly, Fig. (6.10) suggests that the exact location of the reconstructed layers is not of great importance. In the case of the MM reconstruction model, the binned 4 layer model performed approximately identically to the 4 layer model with exact reconstruction positions. For the DL model, reconstruction at the exact layer distances performed $\sim 1.5\%$ better than the model reconstructed at the bin centers. This is reassuring in the context of using binned ranging networks since they cannot provide exact layer distance information. From this observation, we conclude that it is more important that the turbulence profiling method used is better at finding the total number of layers than precisely estimating their location.

6.3 Wide field image correction from reconstructed layers

With reconstructed layers in hand, the estimates of the anisoplanatic PSFs, $\tilde{\mathbf{h}}_a$, can be calculated from each pupil phase matrix $\bar{\mathbf{\Phi}}_a$ following the methods in Section (3.3.1). Notice that we can rewrite the discrete image formation relationship, given by Eq. (3.62), as

$$\mathbf{I}_{i;a} = \mathrm{F}_2^{-1}\left[\mathrm{F}_2\left[\mathbf{I}_{g;a}\right] \odot \mathrm{F}_2\left[\tilde{\mathbf{h}}_a\right]\right], \tag{6.19}$$

where we have swapped out the original system PSF matrix, $\bar{\mathbf{h}}_a$, for our estimate. The image collected on the detector in field region a is the matrix $\mathbf{I}_{i;a}$, and $\mathbf{I}_{g;a}$ is the original unblurred object. Solving for the ideal geometric object matrix:

$$\mathbf{I}_{g;a} = \mathrm{F}_2^{-1}\left[\frac{\mathrm{F}_2\left[\mathbf{I}_{i;a}\right]}{\mathrm{F}_2\left[\tilde{\mathbf{h}}_a\right]}\right]. \tag{6.20}$$

Eq. (6.20) is known as an ideal inverse filter and can be used to deconvolve the PSF from the image. The ideal inverse filter works for the unique case that $\tilde{\mathbf{h}}_a = \bar{\mathbf{h}}_a$.

When there is noise in the estimate of the PSFs, the division in Eq. (6.20) can produce unstable solutions. In the case of additive noise, we can use a Wiener filter instead:

$$\mathbf{I}_{g;a} = \mathrm{F}_2^{-1}\left[\mathbf{G}_{i;a}\frac{\tilde{\mathbf{H}}_a}{\left|\tilde{\mathbf{H}}_a\right|^2 + \frac{\mathbf{S}_n}{\mathbf{S}_s}}\right]. \tag{6.21}$$

In the above expression, $\tilde{\mathbf{H}}_a$ is the Fourier transform of the PSF, $\mathbf{G}_{i;a}$ is the Fourier transform of the blurred image, and $\mathbf{S}_s$ and $\mathbf{S}_n$ are the signal and noise power spectrum, respectively. The Wiener filter is the generalized form of the inverse filter which minimizes the mean-squared error in the presence of additive noise (Barrett and Myers 2003). Given that the PSF estimates are known to deviate from the real PSFs due to reconstructor tomographic error, the Wiener filter is a more suitable deconvolution method than the ideal inverse filter.

6.3.1 MOIC images from reconstructions with varying tomographic error

We can test the Wiener filter method of deconvolution by first investigating the ideal case where an exact reconstruction of the tomographic pupil phase is achieved. This means $\tilde{\mathbf{h}}_a = \bar{\mathbf{h}}_a$ which should perfectly reconstruct the ideal geometric image matrix. Two test cases were run using a simulation of the detailed image of the center capsule of the ISS, originally shown in Fig. (1.1), but artificially expanded to fill the 2.5' FFOV of the system from Chapter (5). One simulation was for the 4 layer atmosphere from the reconstructor test defined in Tab. (6.2). The seeing in the pupil for this atmosphere is $r_0 \approx 0.21$ m for green light which is typical of good conditions at an astronomical observatory. The second test used the same turbulence profile, but using a stronger layer 1, with $r_0 = 0.15$ m, and stronger layer 2, with $r_0 = 0.2$ m. This models a seeing of $r_0 \approx 0.1$ m at 550 nm – more typical of bad seeing at an observatory. Ten image frames were simulated over

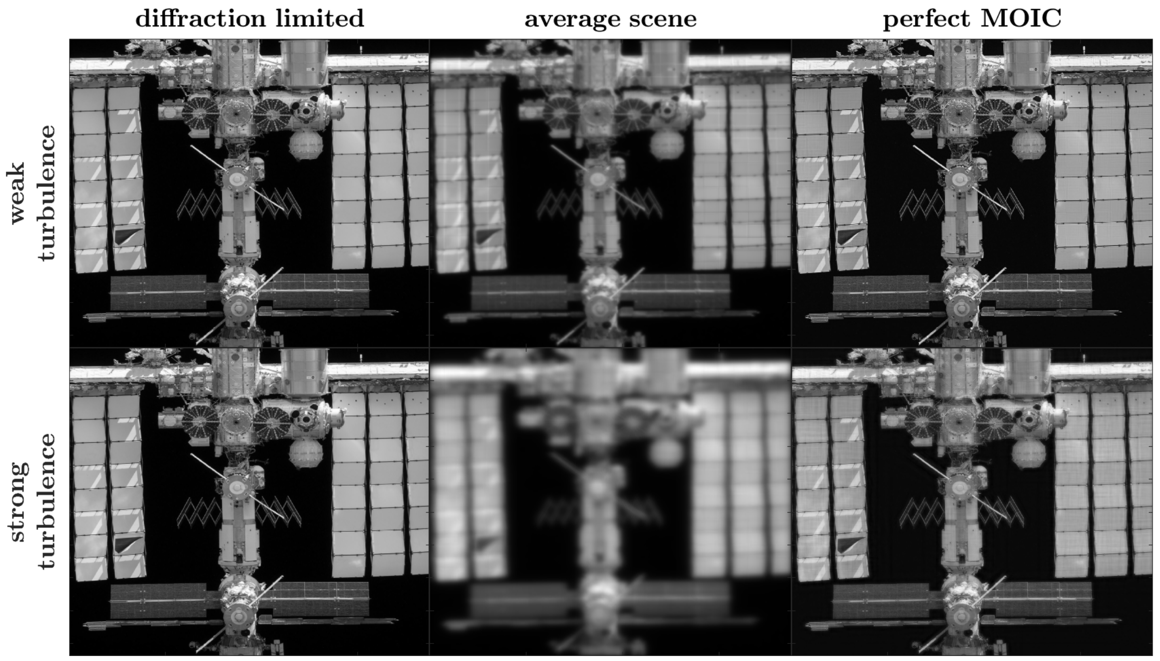


FIGURE 6.12: Left: diffraction limited image simulation of the ISS center capsule for the $D = 1.5$ m system. The size of the capsule has been artificially expanded to fill the 2.5' FFOV. Middle: average scene observed through the weak (top) and strong (bottom) turbulence profiles. The weak profile corresponds to $r_0 \approx 0.2$ m in the pupil and the strong profile corresponds to $r_0 \approx 0.1$ m. Right: average result of Wiener filtering each blurred scene image with a perfectly estimated PSF.

a 200 ms time window in both atmospheres. The resulting average blurred scene and perfectly estimated PSF MOIC images are shown in Fig. (6.12). Note that photon noise was not simulated in this test.

We find that with perfect knowledge of the anisoplanatic PSF, the ideal geometric image can be recovered almost exactly. All of the original high frequency content blurred by the atmosphere appears to be rectified to the quality of the diffraction limited image in both the weak and strong turbulence cases. Thus, if we are able to get a layer reconstruction model which is close enough to the real atmosphere, we suspect it is possible to get this degree of correction from software-based MOIC.

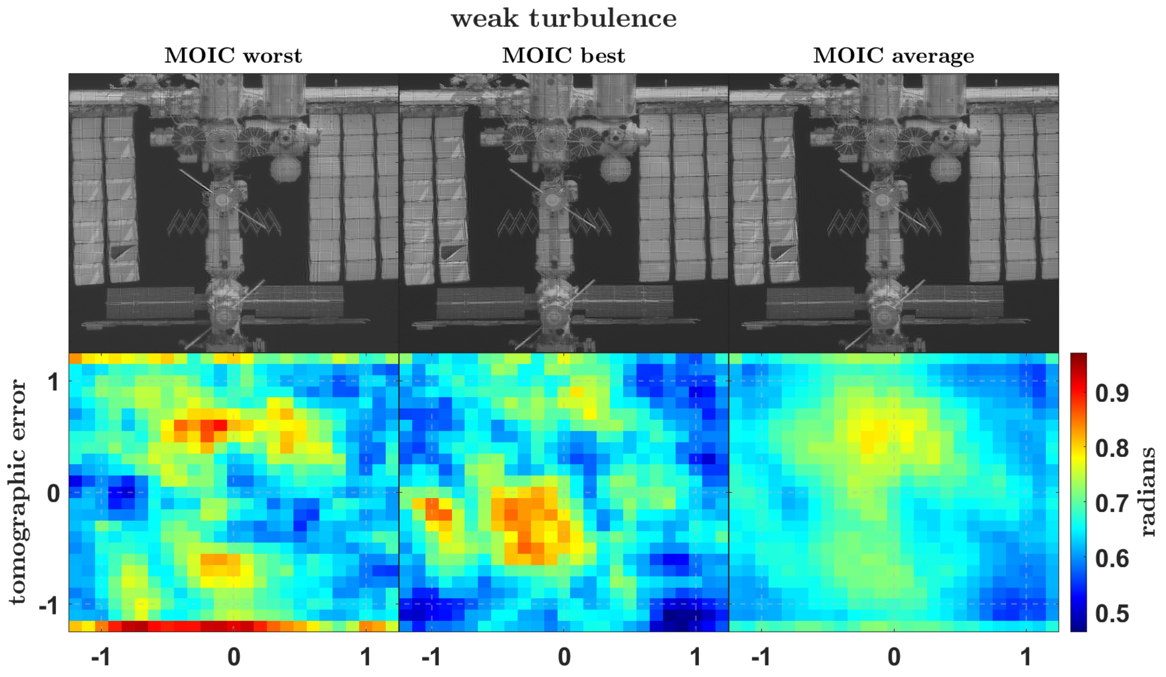


FIGURE 6.13: MOIC from the reconstructed weak turbulence profile (top) and the associate tomographic error maps (bottom). From the scale of the colorbar, we can see that there is small residual phase from the reconstruction. This indicates that the estimate of the anisoplanatic PSF are close to the real PSF. The average MOIC image is sharper than the average scene, shown in the top middle plot of Fig. (6.12), but there is some artifacting and a drop in image dynamic range caused by the phase residual and errors in the noise model which did not manifest in the perfect MOIC image.

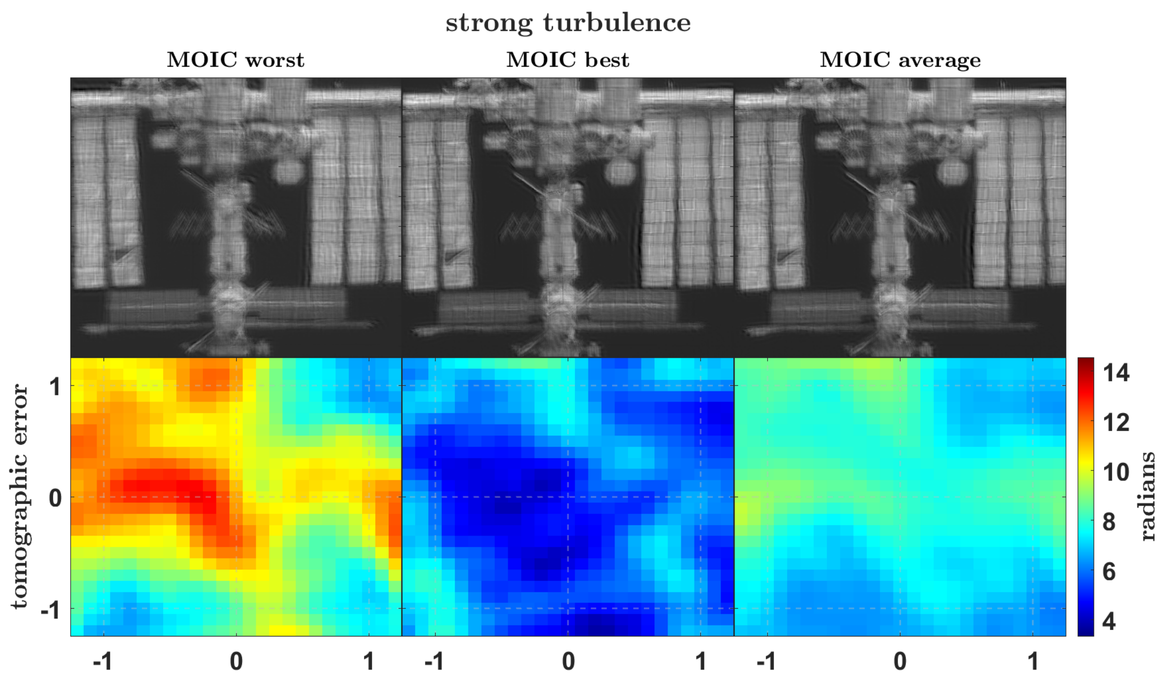


FIGURE 6.14: MOIC from the reconstructed strong turbulence profile (top) and the tomographic error maps (bottom). From the scale of the colorbar, we can see that there is multiple waves of residual phase from the reconstruction, indicating that the estimate of the anisoplanatic PSF is not close to the real PSF. The average MOIC image is sharper than the average scene, shown in the bottom middle plot of Fig. (6.12), but there is significant artifacting caused by the phase residual and errors in the noise model which did not manifest in the perfect MOIC image.

Having found that the Wiener filter can recover the original geometric object matrix, we can test how it works with reconstructed layers. Using the 4 layer DL reconstructor, the weak and strong atmospheres used to generate Fig. (6.12) were reconstructed and the anisoplanatic PSFs were estimated. The estimated PSFs were then used with the Wiener filter assuming a weak white noise model to restore the images. The results are shown in Fig. (6.13-6.14).

From our tests, we deduce that the quality of the recovered images depends strongly on the estimated PSF accuracy as well as the the signal and noise models used for filtering. In the case of the weak turbulence, the phase residuals are small and the PSF estimates are close enough to the real PSFs that a simple Wiener filter can restore most of the image. There is minor artifacting, primarily in the subfield regions which do not have much structure (which is a deficiency inherent to the Wiener filtering process). In the case of strong turbulence, the $V_1 = 250$ mode DL model produced a layer reconstruction with multiple waves of phase residual. While the resulting PSFs were capable of restoring some of the sharpness to the image, there is significant artifacting in the output of the Wiener filter. Thus, we conclude that high quality wide field turbulence compensation with software-based MOIC relies heavily on the accuracy of the tomographic layer reconstruction.

6.3.2 Frequency domain analysis of an MOIC corrected imaging system

While the ideal corrected Strehl used in the previous section gives us an indication of corrected image performance without going through the PSF formation and image deblurring process, it is not a practical method for quantifying the quality of an MOIC restored image. Since Strehl is formally the peak of the diffraction limited PSF divided by the peak of the image PSF, and since the deconvolution process will introduce a change in corrected PSF scale, an MOIC corrected image will have artificially low or high Strehl. Additionally, MOIC is intended for use with extended scenes while Strehl is best for analyzing point source images. Therefore, we suggest assessing MOIC image quality with the modulation transfer function (MTF) rather than Strehl. By using MTF, we can analyze how effective the filtering process was at restoring different spatial frequency content in the image without worrying about changes in scale resulting from the filtering process.

The MTF is formally defined as

$$\mathrm{MTF} = \left|\mathrm{F}_2\left[\bar{\mathbf{h}}_a\right]\right|, \tag{6.22}$$

where the result is then normalized so that $\mathrm{MTF} \in [0,1]$. The MTF describes the contrast in the image as a function of spatial frequency. In the context of the simulation presented in this work, each field-specific PSF, $\bar{\mathbf{h}}_a$, is already known so that the system image can be formed. The estimated field-specific PSFs, $\tilde{\mathbf{h}}_a$, are also known because it is required to perform MOIC. Thus, the MOIC

MTF can be found by computing

$$\tilde{\bar{\mathbf{h}}}_a = \mathrm{F}_2^{-1}\left[\bar{\mathbf{H}}_a \frac{\tilde{\mathbf{H}}_a}{\left|\tilde{\mathbf{H}}_a\right|^2 + \frac{\mathbf{S}_n}{\mathbf{S}_s}}\right] \tag{6.23}$$

and then plugging $\tilde{\bar{\mathbf{h}}}_a$ into Eq. (6.22).

To explore using the MTF as an MOIC image performance metric, 100 frames of the artificially enlarged ISS object viewed through the optical system from Tab. (5.4) were simulated through a two layer atmosphere. Layer 1 was defined by $h = 0.15$ km, $r_0 = 0.15$ m, and $L_0 = 15$ m, and layer 2 was defined by $h = 6$ km, $r_0 = 0.5$ m, and $L_0 = 125$ m. Reconstructions were computed with a 500 mode DL model. The MTFs computed from the scene PSF and the corrected PSF for the on-axis subfield are shown for three time steps in Fig. (6.15).

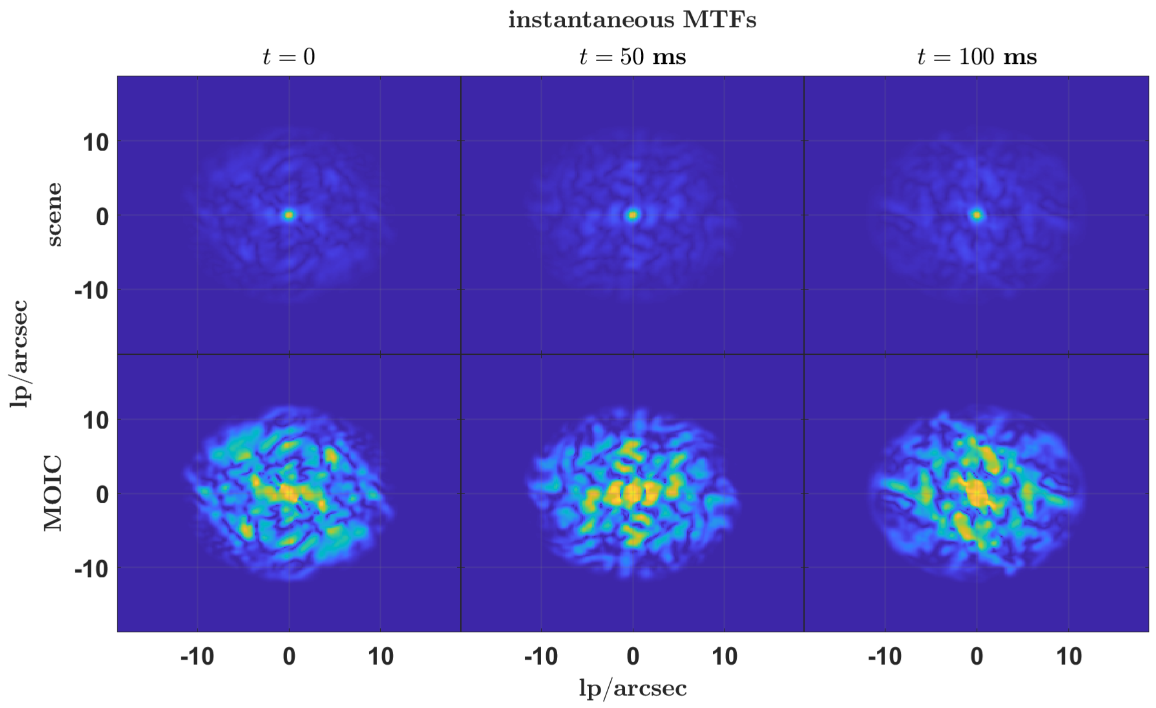


FIGURE 6.15: Demonstration of how MOIC amplifies signal from mid and high spatial frequency information in the scene. At each time step, the scene and MOIC MTFs both contain the same frequency information, but the mid and high frequencies have been amplified in their contrast by the MOIC process.

From Fig. (6.15) we can see that the scene MTF and MOIC corrected MTF at each frame contain the same functional shape. Both contain a bright core at the low frequencies which decreases out to a cutoff frequency. Between the bright core and the cutoff frequency, there are patches of different size and shape which indicate mid and high frequency information which is in the image but has been blurred to varying degrees by the atmosphere. The shapes of the patches match between the scene and MOIC MTFs, but the MOIC MTFs have amplified the signals of the mid and high frequency elements which were not destroyed by the atmosphere. Therefore, when we average the MOIC MTF over multiple time steps, as shown in Fig. (6.16), we find that the amplified mid and high frequency components in the scene are restored. Without the amplification from MOIC, the time-averaged scene MTF approaches the MTF of a system with $D = r_0$.

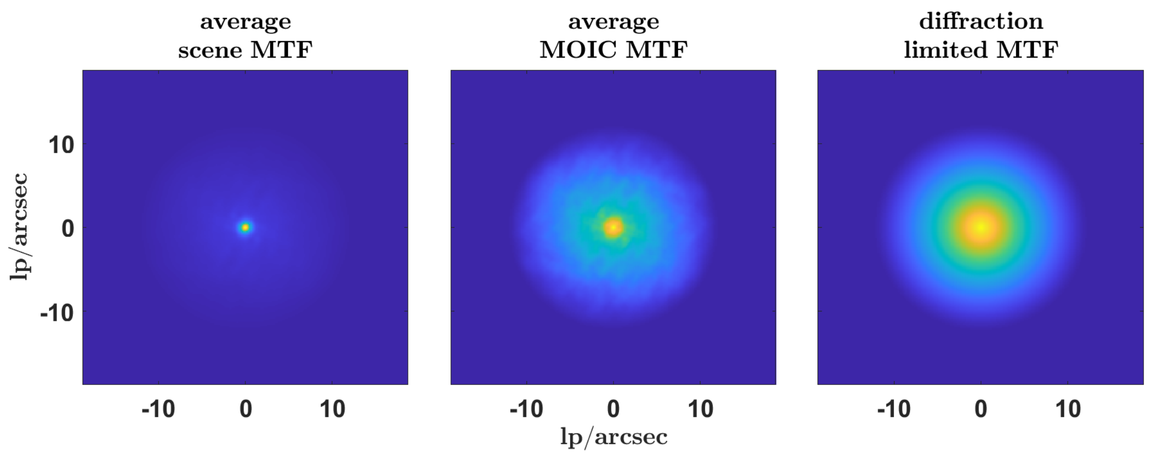


FIGURE 6.16: The average MTF for the scene and the MOIC filtered scene. The average was computed using all fields in the 2.5' FFOV over 100 measurement frames collected during $t \in [0, 1]$ s. The average MOIC MTF is much closer to the diffraction limited MTF for $D = 1.5$ m, while the average scene MTF is approaching the MTF of a diffraction limited system with $D = r_0$.

To check frequency restoration properties of the MOIC filtering process, we can reduce the 2D MTFs to 1D field-specific radial MTFs which can be plotted on the same graph. The MTFs from our circularly symmetric entrance pupil are also necessarily circular. Thus, the time-averaged field-specific 2D MTFs can be integrated azimuthally at each radial distance from the zero frequency pixel to produce a highly sampled estimate of the 1D system MTF. The 1D MTF for the scene and the MOIC images for the four corner and the on-axis fields are shown in Fig. (6.17) along with the diffraction limited MTF.

In Fig. (6.17), we can see that the average MOIC MTF is much closer to the diffraction limited MTF than the average scene MTF. This remains true for each line of sight as long as the tomographic error is similar in each field direction. As $\tilde{\mathbf{h}}_a \to \bar{\mathbf{h}}_a$, the corrected PSF will approach the diffraction limited system PSF and the average MOIC MTF will approach the diffraction limited system MTF – once again solidifying that high quality MOIC requires low tomographic error in the reconstructed atmosphere model.

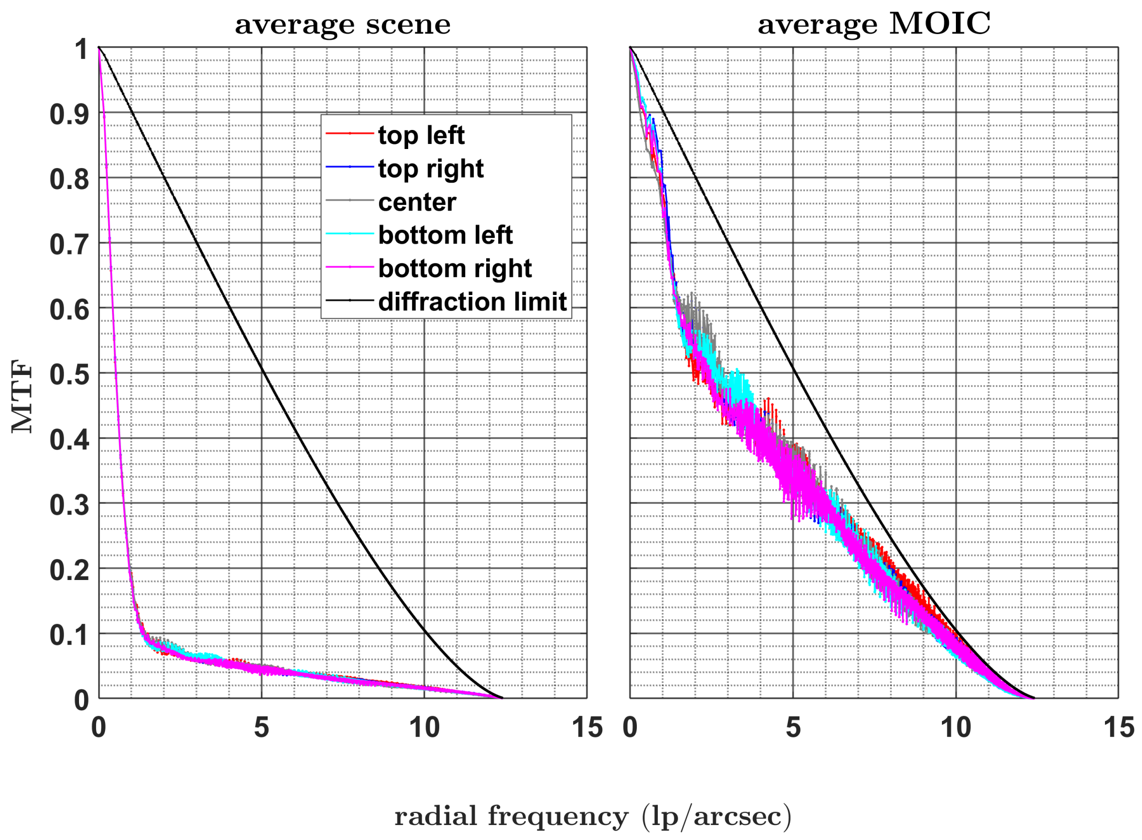


FIGURE 6.17: The time-averaged and azimuthally-averaged MTFs for the scene and the MOIC images along 5 different field angles. The top left field corresponds to $[\xi_1, \zeta_1] = [-1.2', 1.2']$, the top right field is $[\xi_{25}, \zeta_1] = [1.2', 1.2']$, center is $[\xi_{13}, \zeta_{13}] = [0, 0]$, bottom left is for $[\xi_1, \zeta_{25}] = [-1.2', -1.2']$, and bottom right is $[\xi_{25}, \zeta_{25}] = [1.2', -1.2']$. Since the tomographic error is similar for each field direction, the MOIC MTF for each field is approximately the same.

The result of the quantified improvement in mid and high frequency information obtained with MOIC can be observed qualitatively in Fig. (6.18). The average scene is blurred substantially, while the average MOIC image looks nearly identical to the ground truth MOIC image – which is almost identical to the diffraction limited image. Weak high frequency artifacts from the Wiener filter, which appeared in the single frame corrections in Section (6.3.1), do not appear consistently in each image, ultimately resulting in their erasure in the averaged images below. The size of the on-axis isoplanatic patch for this atmosphere is shown as the red rectangle in the ground truth MOIC image, putting into perspective the enhancement over traditional AO corrected field of view which can be obtained from high-quality MOIC.

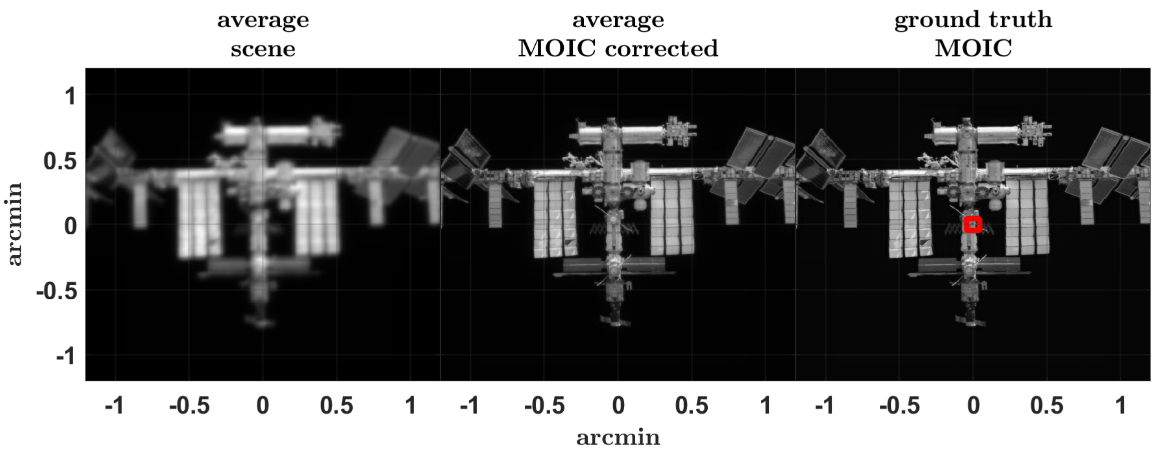


FIGURE 6.18: Scene with no correction (left), with MOIC from the reconstructed layers (center), and with MOIC using the real atmosphere (right), averaged over 100 frames simulated for time $t \in [0,1]$ s. The red square in the ground truth MOIC image is the isoplanatic angle resulting from the second layer with $h = 6$ km and $r_0 = 0.5$ m.

Chapter 7 Summary and future work

7.1 Summary

In this work, a complete simulation of a $D = 1.5$ m optical system with FFOV $= 2.5'$ and multi-layer atmospheric turbulence compensation using MOIC was demonstrated. The simulation involved generating layers of turbulence, simulating SHWFS slope measurements, estimating layer positions, tomographic layer reconstruction, and anisoplanatic PSF deconvolution. The purpose of this end-to-end demonstration is to advance understanding of MOIC as a wide field image correction technique, as well as to develop tools for possible future systems which use the methods herein for turbulence correction.

The first tool developed here was a simulation environment which can form SHWFS measurement warp maps and system images through multiple layers of statistically valid dynamic Kolmogorov and von Kármán turbulence. The simulation environment uses methods which make it suitable to run on a desktop computer. Once the simulation environment was verified, we used it as a tool to help develop three additional MOIC tools: layer SNR, turbulence profiling neural networks, and layer reconstructors.

Layer SNR is a statistical quantity which describes how difficult a layer of turbulence will be to pick out in a multi-layer atmosphere when using correlated slope measurements. We presented an analytical form of the SNR for both Kolmogorov and von Kármán turbulence models which can be calculated from known system parameters, then verified that it is equivalent to the statistical form of the SNR using a Monte Carlo simulation. Even though both SNRs are written as weighted sums of signal and noise layer Fried lengths, the functional form of the noise layer weighting function varies between the the two models. The weight of a Kolmogorov noise layer only depends on measurement aperture size and projected aperture separation. The von Kármán noise layer weight depends on layer inner and outer scale in addition to aperture size and projected separation. Since Kolmogorov turbulence has infinite outer scale, the noise layer weight asymptotically approaches zero as measurement aperture separation goes to infinity. The dependence on finite outer scale for von Kármán turbulence influences the noise layer weighting function, which hits zero when projected measurement aperture separation reaches $\sim L_0$. The fact that real turbulence has an outer scale suggests that modeling the atmosphere with Kolmogorov turbulence will produce a pessimistic model in the context of layer detection SNR.

To estimate turbulence profiles we developed the SA correlation data metric, which can count and range layers, and the ST correlation data metric, which can be used to estimate layer wind velocity. The simulation environment was then used to simulate 3,000 1, 2, 3, and 4 layer atmospheres, for a total of 12,000 data points, where the resulting warp maps were processed into SA correlations and saved. The simulated layers were conditioned on SNR to ensure that most layers had an SNR > 0.25. The conditioned data set was then used to train layer ranging and counting neural networks following a binned atmosphere structure. The effectiveness of the neural networks as layer ranging and counting algorithms was then assessed based on the layer SNR metric. We found that low layer SNR is a strong determining factor in layer detection. We also found that if trained on enough low SNR layers, neural networks can reliably detect layers with SNR well below a value of 1.

Once the turbulence profile has been properly estimated, the layers can be tomographically reconstructed for MOIC. We developed both Legendre mode and turbulence screen mode layer decomposition models for mutli-layer reconstruction from wavefront sensor measurements. The models were tested on an example 4 layer atmosphere using decompositions consisting of ~ 250 modes. The process of distributing low order modes to control multi-layer mode ambiguity was explored for both layer decomposition models. It was found that missing a layer in the reconstruction produces the largest increase in tomographic error. To the benefit of the binned layer ranging network model, it was found that displacing the layer from its true altitude by a few percent minorly impacts reconstructed atmosphere performance. We also found that distributing low order modes in turbulence screen reconstruction models could offer some improvement over a pure turbulence screen decomposition. In every test case analyzed in this work, the Legendre-based reconstruction models outperformed the turbulence mode models.

Given an accurately reconstructed atmosphere, we showed that a Wiener filter is capable of restoring wide field images which have been blurred by the atmosphere to near diffraction limited quality. We also found that when there is ≈ 1 wave of tomographic error in the reconstructed profile for a particular field in the scene, a straightforward Wiener filter with the computed PSF will have artifacting and residual blurring. Therefore, it is of the upmost importance in MOIC that the layer reconstructions are accurate with respect to the real field-specific pupil phase which blurred the image originally.

7.2 Future work

7.2.1 Simulation environment

All of the turbulence profiling and reconstruction methods developed in this work require preprocessing before working with real data. By assuming many simplifications, such as no ray bending at layers of turbulence and automatic pupil slope extraction, the SHWFS data collection process was fast enough to simulate data from 12,000 training atmospheres in about a month. A huge barrier to improving the effectiveness of the trained neural networks is the amount of training data used and the accuracy of the data with respect to real data. Since the current simulation environment is implemented in a single-threaded MATLAB program, adding additional realism, such as sources of measurement noise and more layers per profile, could slow down the simulation until it is impractical to use.

The next step with the simulation environment is to implement it in a parallel computing architecture. Coupling this with a super-cluster of processors could make it possible to generate larger data sets in a week or less. Once the current layer formation and SHWFS data generation process is fast enough, more accurate physics and a wider range of atmospheres could be implemented without slowing the simulation down too much. A parallel architecture would also make influence matrix formation and layer reconstructions faster. Faster layer reconstruction would make it more practical to test reconstructor models and how they can be implemented with different PSF deconvolution techniques.

7.2.2 Turbulence profiling neural networks

In Section (5.4.3) we found that ranging neural network performance is connected to layer SNR. We also found that low SNR layers can be detected if the networks are exposed to enough low SNR layers during training. While we used r_0 and the Hufnagel-Valley model to condition layer strengths in our data set, there is not a direct relationship between r_0 and network performance. This is because each correlation matrix is normalized before being fed into the networks which removes the actual value of r_0 from the data. The layer SNR, however, still appears in the normalized data since it is tied to the relative strengths of each layer given the measurement geometry. Therefore, next generation training data sets should use SNR to determine each layer r_0 rather than the other way around.

Conditioning each layer strength directly from SNR allows us to select the distribution of layer SNRs each network will be exposed to during training. With a faster simulation environment, we could generate several different SNR distributed training data sets – e.g. exponential density, Gaussian distributed around the mean based on a $C_n^2(h)$ model, uniform, etc. The best training distribution could then be deduced by finding which networks perform best over all data sets. We could also use this to more accurately quantify how many data points are required during training to learn varying degress of tweak layer detection

Before developing the theory of layer SNR, we actually generated a training data set which was not conditioned. The networks were never able to converge as layer ranging or counting algorithms. Upon proving the layer SNR, we went back and computed the SNRs of the training data and found that the dataset was dominated by layers with SNR > 100 and SNR < 0.01 – including some even as low as 0.00001. From this experience, we can deduce that ultra-low SNR layers will never be learnable features. However, the location of that threshold is still unknown. Coupling this with knowledge of how many points at a specific SNR are needed to learn a particular layer for a specific system, the exact structure of a good training data set can be formulated.

There are still unknowns to be explored in terms of atmosphere binning structures. The binning structure used in this analysis, given by Fig. (5.9), was not chosen following any specific guideline other than to obtain high resolution near the ground and low resolution at high altitudes. More structured approaches should be explored, such as an equal number of SLODAR samples in each bin, which has an equal number of tip/tilt correlations averaged in each bin, or an equal number of SA correlation samples in each bin.

An effect which was not modeled here but should be characterized for real systems is the effect of multiple layers existing in a bin. We see two possible paths for exploration: multiple layers in the same bin are treated as a single layer, and multiple layers are picked up by a data fusion network which receives the outputs from all ranging and counting networks. Treating multiple layers as one is easier since it requires fewer neural networks and reduces the reconstruction requirements, but the loss in performance as a result of this simplification is uncharacterized. If tomographic error can e reduced by using multiple network responses fed into a data fusion network for sub-bin layer finding, then it would be beneficial to develop this method for MOIC.

Velocity estimation with neural networks is still only a theory. The ST correlations developed in Chapter (5) contain all of the information needed to extract the layer wind speeds and directions – it is only a matter of obtaining a sufficient training data set to learn the relationship. We were actually able to build a velocity estimation training data set of 9,000 simulated ST correlations. The ST correlations contained ten time steps, but did not contain any time averaging ($N_t = 1$). None of the networks were capable of learning the layer velocity relationship. Upon analysis of the generated data, we concluded that the random correlated noise phenomenon for unaveraged ST correlations demonstrated in Fig. (5.4) was to blame. Thus, training data sets for layer velocity estimation simulated in future iteration must contain several time averages to mitigate the random noise effect.

An additional path to explore for velocity estimation networks is using time-correlated SA correlations. In the ST correlations, we are able to visualize the velocity of a layer as a line through the space-space-time plane. Since SA correlations are in the space-angle plane, where layers are lines rather than points, the velocities for a plane. This is much more difficult to visualize and discuss so it was not explored in this work. However, there is nothing which fundamentally bars neural networks from interpreting layer velocity from time-correlated SA correlations. The compact nature of the SA correlation and the high degree of averaging it offers make it a convenient data metric to use for velocity estimation – especially if the SA correlation is already being used for layer ranging.

The true test of the theory of turbulence profiling neural networks developed in this work requires exposing the trained networks to real SHWFS data. If enough precautions were taken during modeling the training data, it is expected that the networks will be able to extract layer positions and velocities. As discussed in Chapter (1), the infinitely thin layer model is non-physical. The real atmosphere contains volumes of turbulence which can appear as thin sheets when observed from the ground. The impacts of this model degeneracy on turbulence profiling network performance cannot be truly be quantified until tested on real data – which may lead to additional considerations being taken during data generation.

7.2.3 MOIC research

Tomographic multi-layer reconstruction

Many facets of tomographic layer reconstruction have already been researched. This includes detailed analyses of different fitting methods (B. Neichel et al. 2009), developing methods to reduce the number of required layers to increase system speed (Saxenhuber et al. 2017), using temporal wavefront sensor information for improved reconstruction and predictive control (Ono et al. 2016), finding methods to mitigate turbulence profile estimation errors (Farley et al. 2020), and even using neural networks to perform the reconstruction process (Osborn et al. 2014, García Riesgo et al.

2021). The agreement across most of these studies is that the turbulence profile used in the reconstruction process plays a major role in corrected performance, and it is a challenge to compute reconstructions fast enough for on-sky correction.

The four multi-layer reconstructors analyzed here are relatively unexplored methods. The concept of using a least-squares reconstruction is not novel, but using Legendre modes is unusual since square regions of phase are not often reconstructed in astronomy applications. Other than the work of Phil Scott (Scott 2021), we are unaware of any other research on turbulence screen mode reconstructors. In the context of offline turbulence compensation, the speed at which correction occurs only needs to be doable during offline hours – it does not need to happen during data collection. This is what enabled the possibility of using turbulence screen modes. With the 250 mode decompositions used in this work, the Legendre basis modes outperformed the turbulence modes. We believe this was driven by the mid-sized turbule residuals between the reconstructed layers and the real atmospheres which were not present in the Legendre mode decompositions. It is possible that with a large set of turbulence modes there would be a superposition of better screen matches to any given layer. Thus, more analysis is needed for turbulence mode reconstructors using a varying number of modes for a diverse set of turbulence profiles. Given the computational requirements of such an analysis, a faster simulation environment will be needed to explore this topic sufficiently.

Another potential method for turbulence screen based reconstruction would be to develop a process for determining the plane wave decomposition vectors, $\vec{k}$, $\vec{\theta}$, $\vec{\psi}$, and $\vec{A}$, needed to generate each layer with Eq. (3.4). If temporal wavefront sensor data can be used to deduce these parameters accurately, a dynamic model of each layer could be estimated. If the dynamics are accurate enough to model the layers for seconds or minutes, the reconstructed layer models could be used for on-sky predictive control.

In our analysis, we found that missing a layer of turbulence in the reconstruction is a strong source of tomographic error. The decrease in performance is stronger when the missed layer has a smaller r_0, and therefore contributes more aberration, as well when there is larger separations between a missed layer and a reconstructed layer. We suspect that if a missed layer is close to another layer, part of the phase in the missed layer will be absorbed by the reconstruction. The degree to which this is true and how it is related to layer SNR remains unknown. Additionally, there is a point where a sufficiently weak and distant undetected layer does not contribute nominally to the tomographic error. A better understanding of these relationships would help inform the training data generation program and help constrain real MOIC system requirements.

Anisoplanatic PSF deconvolution

The Wiener filtering demonstrated here was only qualitative. Using the real anisoplanatic PSF, it is clear that the degree of correction which is possible with MOIC is powerful. However, no quantitative relationships between the tomographic error metric and final corrected image quality were developed. Such a metric would give us an idea of what sort of tomographic error is tolerable for a specific imaging application.

The Wiener filter is only one type of filtering process which could be used with the estimated PSFs to restore images. Given that most programming languages have fast Wiener filtering algorithms available, it is easy to implement. This does not mean that the Wiener filter is the best option for general MOIC. Depending on the types of scenes a particular MOIC system will view, it is likely that there is an alternative image restoration filter which will outperform the Wiener filter. Therefore, future research on anisoplanatic PSF deconvolution for MOIC should investigate the use of different image restoration filters for different applications such as astronomical telescopes, Earth-viewing satellites, and long distance terrestrial imagers.

Appendix A Ray tracing matrices

From the geometric ray trace model developed in Section (3.2.1), the subaperture chief rays intersect the system pupil at coordinate vectors $\vec{\bar{r}}_b$ organized into the matrix $\bar{\mathbf{r}}_b$. This matrix is constructed by appending each $\vec{\bar{r}}_b$ along a new column for $b \in [1,B]$. We choose to start at the top left subaperture in the WFS pupil, then append the next subaperture coordinate going across the WFS pupil top row from left to right. After reaching the end of the first row of subapertures, we move down one to the second row and repeat until all $\vec{\bar{r}}_b$ has been accounted for:

$$\bar{\mathbf{r}}_b = \begin{bmatrix} \vec{\bar{r}}_1 & \vec{\bar{r}}_2 & \ldots \vec{\bar{r}}_{B-1} & \vec{\bar{r}}_B \end{bmatrix}, \tag{A.1}$$

which is a $2 \times B$ matrix. To prepare for computing each subaperture coordinate for each subfield angle, we now duplicate the matrix $\bar{\mathbf{r}}_b$ row-wise for subfield indices $a \in [1,A]$. The results is the $2A \times B$ matrix of the form

$$\bar{\mathbf{r}}_{a,b} = \begin{bmatrix} \bar{\mathbf{r}}_b \\ \vdots \\ \bar{\mathbf{r}}_b \end{bmatrix}. \tag{A.2}$$

We can also use each $\vec{\bar{r}}_b$ to determine the coordinates of the corner marginal rays. First we organize each vector into the matrix

$$\begin{bmatrix} \vec{\bar{r}}_b & \vec{\bar{r}}_b \\ \vec{\bar{r}}_b & \vec{\bar{r}}_b \end{bmatrix}.$$

Now we use the subaperture side length, s, to define the shift matrix

$$\mathbf{s} = \begin{bmatrix} \vec{s}_1 & \vec{s}_2 \\ \vec{s}_3 & \vec{s}_4 \end{bmatrix} \tag{A.3}$$

with vector elements

$$\begin{aligned} \vec{s}_1 &= \begin{bmatrix} -s/2 \\ s/2 \end{bmatrix} & \vec{s}_2 &= \begin{bmatrix} s/2 \\ s/2 \end{bmatrix} \\ \vec{s}_3 &= \begin{bmatrix} -s/2 \\ -s/2 \end{bmatrix} & \vec{s}_2 &= \begin{bmatrix} s/2 \\ -s/2 \end{bmatrix}. \end{aligned}$$

The marginal ray coordinates for the subaperture b in the pupil plane is

$$\mathbf{r}'_b = \begin{bmatrix} \vec{\vec{r}}_b & \vec{\vec{r}}_b \\ \vec{\vec{r}}_b & \vec{\vec{r}}_b \end{bmatrix} + \mathbf{s} = \begin{bmatrix} \vec{r}_{b1} & \vec{r}_{b2} \\ \vec{r}_{b3} & \vec{r}_{b4} \end{bmatrix}. \tag{A.4}$$

Stacking these 4×2 matrices of coordinates column-wise produces the matrix of all subaperture marginal rays in the pupil plane:

$$\mathbf{r}_b = \begin{bmatrix} \mathbf{r}'_1 & \mathbf{r}'_2 & \ldots \mathbf{r}'_{B-1} & \mathbf{r}'_B \end{bmatrix}, \tag{A.5}$$

which is a $4 \times 2B$ matrix. Again stacking these row-wise for fields $a \in [1, A]$ produces

$$\mathbf{r}_{a,b} = \begin{bmatrix} \mathbf{r}_b \\ \vdots \\ \mathbf{r}_b \end{bmatrix}, \tag{A.6}$$

which is a $4A \times 2B$ matrix

By approximating that all objects are infinitely far, each object will have the same field angle at each subaperture. Additionally, the field angle from each object at the center of each subaperture will be identical for the field angle at the subaperture corners. If we assume the object scene is arbitrary we can segment it into regions labeled $a \in [1,a]$ centered at a specific angle

$$\vec{u}_a = \begin{bmatrix} \xi_a \\ \zeta_a \end{bmatrix}$$

where ξ_a is the angle in the xz plane and ζ_a is the same for the yz plane. We can then build object field angle matrices structured to project the coordinates in $\bar{\mathbf{r}}_{a,b}$ and $\mathbf{r}_{a,b}$ onto any layer between the object and the entrance pupil. First we take each $\vec{u}_a$ and stack the individual object angle vectors row-wise to form a $2A \times 1$ vector. We then make B copies of the vector and stack them column-wise to produce

$$\bar{\mathbf{u}}_{a,b} = \begin{bmatrix} \vec{u}_{1,1} & \cdots & \vec{u}_{1,B} \\ \vdots & \ddots & \vdots \\ \vec{u}_{A,1} & \cdots & \vec{u}_{A,B} \end{bmatrix} \tag{A.7}$$

$\bar{\mathbf{u}}_{a,b}$ is a $2A \times B$ matrix of ray angles connecting subfield a to the center of subaperture b. The process is repeated with 4×2 copies of each angle coordinate to produce $\mathbf{u}_{a,b}$ which is a $4A \times 2B$ matrix of ray angles connecting object a to the four corners of each subaperture.

We can project each subaperture along each field angle to the corresponding layer distance h_ℓ. For the subaperture center coordinates at layer ℓ we compute

$$\bar{\mathbf{r}}_\ell = \bar{\mathbf{r}}_{a,b} - h_\ell \bar{\mathbf{u}}_{a,b}. \tag{A.8}$$

Repeating the calculation for each layer and appending the results along the third matrix dimension,

$$\bar{\mathbf{r}} = \begin{bmatrix} [\bar{\mathbf{r}}_1] & \ldots & [\bar{\mathbf{r}}_\ell] & \ldots & [\bar{\mathbf{r}}_L] \end{bmatrix}, \tag{A.9}$$

produces the final chief ray matrix $\bar{\mathbf{r}}$. $\bar{\mathbf{r}}$ is a $2A \times B \times L$ matrix of chief ray coordinates for subfield a in subaperture b at layer ℓ.

For the corner marginal rays,

$$\mathbf{r}_\ell = \mathbf{r}_{a,b} - h_\ell \mathbf{u}_{a,b}. \tag{A.10}$$

Repeating the appending process along the third dimension for each layer produces $\mathbf{r}$ – the $4A \times 2B \times L$ matrix of corner marginal rays for each subfield a in subaperture b at layer ℓ.

The above matrix construction allows us to directly access any specific subfield-subaperture combination at a particular layer. Using MATLAB's row-column indexing, each chief ray (x,y) coordinate intercepts a layer at

$$\begin{aligned} \bar{x}_{a,b,\ell} &= \bar{\mathbf{r}}\left(2a-1,b,\ell\right), \\ \bar{y}_{a,b,\ell} &= \bar{\mathbf{r}}\left(2a,b,\ell\right). \end{aligned}$$

The top left corner marginal rays are

$$x^{(TL)}_{a,b,\ell} = \mathbf{r}\left(4a-3,2b-1,\ell\right), \tag{A.11}$$

$$y^{(TL)}_{a,b,\ell} = \mathbf{r}\left(4a-2,2b-1,\ell\right), \tag{A.12}$$

top right are

$$x^{(TR)}_{a,b,\ell} = \mathbf{r}\left(4a-3,2b,\ell\right), \tag{A.13}$$

$$y^{(TR)}_{a,b,\ell} = \mathbf{r}\left(4a-2,2b,\ell\right), \tag{A.14}$$

bottom left are

$$x^{(BL)}_{a,b,\ell} = \mathbf{r}\left(4a-1,2b-1,\ell\right), \tag{A.15}$$

$$y^{(BL)}_{a,b,\ell} = \mathbf{r}\left(4a,2b-1,\ell\right), \tag{A.16}$$

and bottom right are

$$x^{(BR)}_{a,b,\ell} = \mathbf{r}\,(4a-1, 2b, \ell)\,, \tag{A.17}$$

$$y^{(BR)}_{a,b,\ell} = \mathbf{r}\,(4a, 2b, \ell)\,. \tag{A.18}$$

Appendix B Analytic layer SNR from Legendre coefficient autocorrelation integrals

B.1 Tip and tilt autocorrelation and variance integrals

The generalized Legendre coefficient autocorrelation integral,

$$B_{\alpha_j;\ell}(\delta\vec{s}_\ell) = |\bar{c}_{n,m}|^2 \int\limits_{-\infty}^{\infty} E(\vec{k};\delta\vec{s}_\ell) S_{\phi_\ell}(\vec{k}) \left| \frac{\mathbf{J}_{n+\frac{1}{2}}(\pi s k_1)\, \mathbf{J}_{m+\frac{1}{2}}(\pi s k_2)}{\sqrt{k_1 k_2}} \right|^2 d\vec{k}, \tag{B.1}$$

can be used to determine the tip and tilt mode autocorrelations and variances needed for the layer SNR in Eq. (4.14). Tip is defined as the Legendre mode $(n,m) = (1,0)$, which when plugged into the Bessel functions in Eq. (B.1) produces

$$\mathbf{J}_{\frac{3}{2}}(\pi s k_1) = \frac{\sin(\pi s k_1) - \pi s k_1 \cos(\pi s k_1)}{2\pi^2 (s k_1/2)^{3/2}}, \tag{B.2}$$

$$\mathbf{J}_{\frac{1}{2}}(\pi s k_2) = \frac{\sin(\pi s k_2)}{\pi (s k_2/2)^{1/2}}. \tag{B.3}$$

Applying the substitutions

$$\begin{aligned} \eta_1 &= \pi s k_1 \\ \eta_2 &= \pi s k_2 \end{aligned} \tag{B.4}$$

and taking the product of the two Bessel functions to match the from in Eq. (B.1) results in

$$\mathbf{J}_{\frac{3}{2}}(\eta_1)\mathbf{J}_{\frac{1}{2}}(\eta_2) = \frac{2\sin(\eta_2)[\sin(\eta_1) - \eta_1\cos(\eta_1)]}{\pi\, \eta_1^{3/2} \eta_2^{1/2}}. \tag{B.5}$$

Using Eq. (B.5) to simplify the expression inside the absolute value signs:

$$\left| \frac{\mathbf{J}_{\frac{3}{2}}(\eta_1)\mathbf{J}_{\frac{1}{2}}(\eta_2)}{(\eta_1\eta_2)^{1/2}(\pi s)^{-1}} \right|^2 = 4s^2 T_1(\vec{\eta}\,) \tag{B.6}$$

where

$$T_1(\vec{\eta}\,) = \left| \frac{\sin(\eta_2)[\sin(\eta_1) - \eta_1\cos(\eta_1)]}{\eta_1^2\eta_2} \right|^2. \tag{B.7}$$

Replacing the components in the expression for $B_{\alpha_j;\ell}$ with Eq. (B.6) and carrying through the substitutions for η_1 and η_2 we find

$$B_{\text{tip};\ell}(\delta\vec{s}_\ell) = C \int_{-\infty}^{\infty} E\left(\frac{\vec{\eta}}{\pi}; \frac{\delta\vec{s}_\ell}{s}\right) S_{\phi_\ell}\left(\frac{\vec{\eta}}{\pi s}\right) T_1(\vec{\eta}\,)\, d\vec{\eta}. \tag{B.8}$$

The complex exponential is now in terms of $\vec{\eta}$ and takes the form

$$E\left(\frac{\vec{\eta}}{\pi}; \frac{\delta\vec{s}_\ell}{s}\right) = e^{i2(\vec{\eta}\cdot\delta\vec{s}_\ell)/s}, \tag{B.9}$$

and the scaling coefficient in front of the integral is

$$C = \left|4c_{n,m}(-2)^{n+m}(i)^{n+m}\Gamma(n+1)\Gamma(m+1)\right|^2. \tag{B.10}$$

Repeating this process for tilt – which is defined as $(n,m) = (0,1)$ – results in

$$B_{\text{tilt};\ell}(\delta\vec{s}_\ell) = C \int_{-\infty}^{\infty} E\left(\frac{\vec{\eta}}{\pi}; \frac{\delta\vec{s}_\ell}{s}\right) S_{\phi_\ell}\left(\frac{\vec{\eta}}{\pi s}\right) T_2(\vec{\eta}\,)\, d\vec{\eta}. \tag{B.11}$$

The new component function in this integral is

$$T_2(\vec{\eta}\,) = \left| \frac{\sin(\eta_1)[\sin(\eta_2) - \eta_2\cos(\eta_2)]}{\eta_1\eta_2^2} \right|^2. \tag{B.12}$$

Finally, using the definition for variance in Eq. (4.15) we find that the variance integrals are

$$\sigma^2_{\text{tip};\ell} = C \int\limits_{-\infty}^{\infty} S_{\phi_\ell}\left(\frac{\vec{\eta}}{\pi s}\right) T_1(\vec{\eta}\,)\, d\vec{\eta}, \tag{B.13}$$

$$\sigma^2_{\text{tilt};\ell} = C \int\limits_{-\infty}^{\infty} S_{\phi_\ell}\left(\frac{\vec{\eta}}{\pi s}\right) T_2(\vec{\eta}\,)\, d\vec{\eta}. \tag{B.14}$$

B.2 The statistical SNR in integral form

Plugging the autocorrelation and variance integrals into Eq. (4.14) for a signal layer ℓ_0 and noise layers ℓ_c:

$$\begin{aligned}
\text{SNR}_{\ell_0} &= \frac{\sigma^2_{\text{tip};\ell_0} + \sigma^2_{\text{tilt};\ell_0}}{\sum_{\ell_c}^{L-1} B_{\text{tip};\ell_c}(\delta\vec{s}_{\ell_c}) + B_{\text{tilt};\ell_c}(\delta\vec{s}_{\ell_c})} \\
&= \frac{C\int_{-\infty}^{\infty} S_{\phi_{\ell_0}}\left(\frac{\vec{\eta}}{\pi s}\right) T_1(\vec{\eta}\,)\, d\vec{\eta} + C\int_{-\infty}^{\infty} S_{\phi_{\ell_0}}\left(\frac{\vec{\eta}}{\pi s}\right) T_2(\vec{\eta}\,)\, d\vec{\eta}}{\sum_{\ell_c}^{L-1} C\int_{-\infty}^{\infty} E\left(\frac{\vec{\eta}}{\pi};\frac{\delta\vec{s}_{\ell_c}}{s}\right) S_{\phi_{\ell_c}}\left(\frac{\vec{\eta}}{\pi s}\right) T_1(\vec{\eta}\,)\, d\vec{\eta} + C\int_{-\infty}^{\infty} E\left(\frac{\vec{\eta}}{\pi};\frac{\delta\vec{s}_{\ell_c}}{s}\right) S_{\phi_{\ell_c}}\left(\frac{\vec{\eta}}{\pi s}\right) T_2(\vec{\eta}\,)\, d\vec{\eta}} \\
&= \frac{\int_{-\infty}^{\infty} S_{\phi_{\ell_0}}\left(\frac{\vec{\eta}}{\pi s}\right) [T_1(\vec{\eta}\,) + T_2(\vec{\eta}\,)]\, d\vec{\eta}}{\sum_{\ell_c}^{L-1} \int_{-\infty}^{\infty} E\left(\frac{\vec{\eta}}{\pi};\frac{\delta\vec{s}_{\ell_c}}{s}\right) S_{\phi_{\ell_c}}\left(\frac{\vec{\eta}}{\pi s}\right) [T_1(\vec{\eta}\,) + T_2(\vec{\eta}\,)]\, d\vec{\eta}}.
\end{aligned} \tag{B.15}$$

A convenience of this form is that the scaling coefficients are identical in the numerator and denominator resulting in cancellation. This is because the only difference between the two integrals is the complex phase component $E\left(\frac{\vec{\eta}}{\pi};\frac{\delta\vec{s}_\ell}{s}\right)$. Additionally, since the complex exponential is not divergent, any singularities which cause divergence in the individual integrals for a particular value of $\vec{\eta}$ will be identical – thus allowing the potential for the SNR to converge when the integrals themselves do not. Now all that is needed to solve for the SNR of a particular layer of turbulence is the layer power spectrum.

B.3 Kolmogorov SNR in terms of model parameters

The Kolmogorov power spectrum is

$$S_{\phi_\ell}^{(\mathrm{kol})}(k_1,k_2) = C_{\mathrm{kol}} r_0^{-5/3} (k_1^2+k_2^2)^{-11/6}. \tag{B.16}$$

Using the substitution in Eq. (B.4),

$$S_{\phi_\ell}^{(\mathrm{kol})}\left(\frac{\vec{\eta}}{\pi s}\right) = C_{\mathrm{kol}}(\pi s)^{11/3} r_0^{-5/3}\left(\eta_1^2+\eta_2^2\right)^{-11/6}. \tag{B.17}$$

Plugging Eq. (B.17) into the integral form of the SNR results in the expression

$$\mathrm{SNR}_{\ell_0}^{(\mathrm{kol})} = \frac{r_{\ell_0}^{-5/3}\int_{-\infty}^{\infty}\left(\eta_1^2+\eta_2^2\right)^{-11/6}\left[T_1(\vec{\eta}\,)+T_2(\vec{\eta}\,)\right]d\vec{\eta}}{\sum_{\ell_c}^{L-1} r_{\ell_c}^{-5/3}\int_{-\infty}^{\infty}E\left(\frac{\vec{\eta}}{\pi};\frac{\delta\vec{s}_{\ell_c}}{s}\right)\left(\eta_1^2+\eta_2^2\right)^{-11/6}\left[T_1(\vec{\eta}\,)+T_2(\vec{\eta}\,)\right]d\vec{\eta}.} \tag{B.18}$$

By packaging the integrals into the Kolmogorov tip/tilt correlation coefficient (TTCC),

$$\mathbf{R}_{\mathrm{kol}}(\delta\vec{s}_{\ell_c}) = \int_{-\infty}^{\infty} E\left(\frac{\vec{\eta}}{\pi};\frac{\delta\vec{s}_{\ell_c}}{s}\right)\frac{T_1(\vec{\eta})+T_2(\vec{\eta})}{\left[\eta_1^2+\eta_2^2\right]^{11/6}}\,d\vec{\eta}, \tag{B.19}$$

we can rearrange the final form of Eq (B.15) into a weighted sum:

$$\begin{aligned}\mathrm{SNR}_{\ell_0}^{(\mathrm{kol})} &= r_{\ell_0}^{-5/3}\frac{\mathbf{R}_{\mathrm{kol}}(0)}{\sum_{\ell_c}^{L-1} r_{\ell_c}^{-5/3}\mathbf{R}_{\mathrm{kol}}(\delta\vec{s}_{\ell_c}).}\\ &= r_{\ell_0}^{-5/3}\left[\sum_{\ell_c=1}^{L-1} r_{\ell_c}^{-5/3}\frac{\mathbf{R}_{\mathrm{kol}}(\delta\vec{s}_{\ell_c})}{\mathbf{R}_{\mathrm{kol}}(0)}\right]^{-1},\end{aligned} \tag{B.20}$$

which is the analytic expression form of the SNR we choose since it can be determined using known simulation parameters.

B.4 von Kármán SNR in terms of model parameters

We can rewrite the von Kármán power spectrum

$$S_{\mathrm{vK}}(k_1,k_2) = \frac{C_{\mathrm{vK}} r_0^{-5/3}}{(k_1^2+k_2^2+L_0^{-2})^{11/6}} e^{-(k_1^2+k_2^2)/k_m^2}. \tag{B.21}$$

Using the substitution in Eq. (B.4),

$$S_{\mathrm{vK}}\left(\frac{\vec{\eta}}{\pi s}\right) = \frac{C_{\mathrm{vK}}(\pi s)^{11/3} r_0^{-5/3}}{\left[\eta_1^2+\eta_2^2+\left(\frac{\pi s}{L_0}\right)^2\right]^{11/6}} e^{-(\eta_1^2+\eta_2^2)/(5.92\pi s/l_0)^2}. \tag{B.22}$$

Plugging Eq. (B.22) into the integral form of the SNR results in the expression

$$\mathrm{SNR}_{\ell_0}^{(\mathrm{vK})} = \frac{r_{\ell_0}^{-5/3}\int_{-\infty}^{\infty} e^{-(\eta_1^2+\eta_2^2)/(5.92\pi s/l_0)^2}\frac{T_1(\vec{\eta})+T_2(\vec{\eta})}{\left[\eta_1^2+\eta_2^2+(\pi s/L_0)^2\right]^{11/6}}d\vec{\eta}}{\sum_{\ell_c}^{L-1} r_{\ell_c}^{-5/3}\int_{-\infty}^{\infty} e^{-(\eta_1^2+\eta_2^2)/(5.92\pi s/l_0)^2} E\left(\frac{\vec{\eta}}{\pi};\frac{\delta\vec{s}_{\ell_c}}{s}\right)\frac{T_1(\vec{\eta})+T_2(\vec{\eta})}{\left[\eta_1^2+\eta_2^2+(\pi s/L_0)^2\right]^{11/6}}d\vec{\eta}.} \tag{B.23}$$

By packaging the integrals into the von Kármán TTCC,

$$\mathbf{R}_{\mathrm{vK}}(\delta\vec{s}_{\ell_c}) = \int_{-\infty}^{\infty} e^{-(\eta_1^2+\eta_2^2)/(5.92\pi s/l_0)^2}\frac{E\left(\frac{\vec{\eta}}{\pi};\frac{\delta\vec{s}_{\ell_c}}{s}\right)\left[T_1(\vec{\eta})+T_2(\vec{\eta})\right]}{\left[\eta_1^2+\eta_2^2+(\pi s/L_0)^2\right]^{11/6}}d\vec{\eta}, \tag{B.24}$$

we can rearrange Eq (B.23) into a weighted sum:

$$\begin{aligned}\mathrm{SNR}_{\ell_0}^{(\mathrm{vK})} &= r_{\ell_0}^{-5/3}\frac{\mathbf{R}_{\mathrm{vK}}(0)}{\sum_{\ell_c}^{L-1} r_{\ell_c}^{-5/3}\mathbf{R}_{\mathrm{vK}}(\delta\vec{s}_{\ell_c}).} \\ &= r_{\ell_0}^{-5/3}\left[\sum_{\ell_c=1}^{L-1} r_{\ell_c}^{-5/3}\frac{\mathbf{R}_{\mathrm{vK}}(\delta\vec{s}_{\ell_c})}{\mathbf{R}_{\mathrm{vK}}(0;s,l_0,L_0)}\right]^{-1},\end{aligned} \tag{B.25}$$

which is the same weighted sum form as for Kolmogorov turbulence. The only difference between the two SNR expressions is the form of the TTCC. Most notably, $\mathbf{R}_{\mathrm{vK}}(\delta\vec{s}_{\ell_c})$ depends on the inner and outer scale of the turbulence which the Kolmogorov TTCC does not.

Appendix C Legendre polynomials

Functions which are square integrable over a finite region of support can be decomposed into a summation of orthonormal basis functions. The natural basis function set for square regions are 2D Legendre polynomials. Since we have decided to model the WFS pupil and layers of turbulence as square regions, we can decompose these phases into the form

$$\phi(x,y) = \sum_{m=1}^{\infty} \alpha_m \mathscr{L}_m(x,y) \tag{C.1}$$

where $\mathscr{L}_m(x,y)$ is the m^{th} orthonormal 2D Legendre polynomial satisfying

$$\frac{1}{4}\int_{-1}^{1}\int_{-1}^{1} \mathscr{L}_m(x,y)\mathscr{L}_{m'}(x,y)\,dx\,dy = \delta_{mm'} \tag{C.2}$$

with $\delta_{mm'}$ being the Kronecker delta function (Mahajan 2010). The corresponding coefficient, α_m, for each 2D Legendre polynomial in the phase can be determined by computing the inner product

$$\alpha_m = \frac{1}{4R_\ell^2}\int_{-1}^{1}\int_{-1}^{1} \phi_\ell(x,y)\mathscr{L}_m(x,y)\,dx\,dy, \tag{C.3}$$

where R_ℓ is the half-width of the square layer of phase, $\phi_\ell(x,y)$. Scaling by the half width of the layer coordinate grid ensures that the projection of the phase onto the Legendre polynomial is scaled to the region of support, $x \in [-1,1]$ and $y \in [-1,1]$.

Each 2D Legendre polynomial is the multiplication of two 1D Legendre polynomials

$$\mathscr{L}_m(x,y) = c_m P_u(x) P_{u'}(y), \tag{C.4}$$

u	$P_u(x)$ (not normalized)
0	1
1	x
2	$\frac{1}{2}(3x^2-1)$
3	$\frac{1}{2}(5x^3-3x)$
4	$\frac{1}{8}(35x^4-30x^2+3)$
5	$\frac{1}{8}(63x^5-70x^3+15x)$
u	$\frac{1}{2^u u!}\frac{d^u}{dx^u}(x^2-1)^u$

m	$\mathscr{L}_m(x,y)=c_m P_u(x)P_{u'}(y)$	Associated Aberration
0	$P_0(x)P_0(y)$	Piston
1	$\sqrt{3}P_1(x)P_0(y)$	Tilt
2	$\sqrt{3}P_0(x)P_1(y)$	Tip
3	$\sqrt{5}P_2(x)P_0(y)$	x Defocus
4	$3P_1(x)P_1(y)$	Astigmatism
5	$\sqrt{5}P_0(x)P_2(y)$	y Defocus
6	$\sqrt{7}P_3(x)P_0(y)$	x Coma

TABLE C.1: Left: 1D Legendre polynomials for a range of values including arbitrary index u in the form of the Legendre Rodrigues' formula. Right: 2D Legendre polynomials constructed for wavefront representation over a unit square aperture.

where

$$c_m = \sqrt{(2u+1)(2u'+1)} \tag{C.5}$$

is the orthonormal scaling coefficient. The 1D Legendre polynomials, $P_u(x)$ and $P_{u'}(y)$, can be derived from the Legendre Rodrigues' formula which is provided in the last row of the left table in Tab. (C.1).

Works Cited

Andrews, L. C., and R. L. Philips. *Field Guide to Probability, Random Processes, and Random Data Analysis*. Ed. by J. E. Greivenkamp, SPIE P, 2012.

Barrett, H. H., and K. J. Myers. *Foundations of Image Science*. Wiley, 2003.

Beckers, J. M. "Increasing the Size of the Isoplanatic Patch with Multiconjugate Adaptive Optics". *Very Large Telescopes and their Instrumentation, Vol. 2*. Oct. 1988, pp. 693–703.

Butterley, T., et al. "Determination of the profile of atmospheric optical turbulence strength from SLODAR data". *Monthly Notices of the Royal Astronomical Society*, vol. 369, no. 2, June 2006, pp. 835–45.

Charnotskii, M. "Sparse spectrum model for a turbulent phase". *J. Opt. Soc. Am. A*, vol. 30, no. 3, Mar. 2013, pp. 479–88.

Charnotskii, M. "Comparison of four techniques for turbulent phase screens simulation". *J. Opt. Soc. Am. A*, vol. 37, no. 5, May 2020, pp. 738–47.

Costille, A., and T. Fusco. "Impact of Cn2 profile on tomographic reconstruction performance: application to E-ELT wide filed AO systems". *SPIE Astronomical Telescopes + Instrumentation*. SPIE, Adaptive Optics Systems III, Sept. 2012.

Dekany, R. G., et al. "Adaptive optics requirements definition for TMT". *Advancements in Adaptive Optics*. Ed. by Domenico Bonaccini Calia et al., SPIE, 2004, pp. 879–90.

Farley, O. J. D., et al. "Limitations imposed by optical turbulence profile structure and evolution on tomographic reconstruction for the ELT". *MNRAS*, vol. 494, 2020, pp. 2773–84.

Fried, D. L. “Statistics of a Geometric Representation of Wavefront Distortion”. *JOSA*, vol. 55, no. 11, 1965, pp. 1427–35.

Fried, D. L. “Anisoplanatism in adaptive optics”. *JOSA*, vol. 72, no. 1, 1982, pp. 52–61.

Frieden, R. B. *Probability, Statistical Optics, and Data Testing*. 3rd ed., Springer, 2011.

Fuchs, A., et al. “Focusing on a Turbulent Layer: Principle of the “Generalized SCIDAR””. *Publications of the Astronomical Society of the Pacific*, vol. 110, no. 743, 1998, pp. 86–91. Accessed 20/1/2023.

García Riesgo, F., et al. “Overview and Choice of Artificial Intelligence Approaches for Night-Time Adaptive Optics Reconstruction”. *Mathematics*, vol. 9, no. 11, 2021.

Gendron, E., et al. “MOAO first on-sky demonstration with CANARY”. *Astronomy and Astrophysics*, vol. 529, May 2011.

Goodman, J. W. *Statistical Optics*. John Wiley & Sons, 2000.

Hamilton, R. J., and M. Hart. “Turbulence profiling neural networks using imaging Shack-Hartmann data for wide-field image correction”. *Adaptive Optics Systems VIII*. International Society for Optics / Photonics, ed. by Laura Schreiber et al., SPIE, 2022, 121855T.

Hardy, John W. *Adaptive optics for astronomical telescopes*. Oxford Univ. P, 1998.

Johnston, D. C., and B. Welsh. “Analysis of multiconjugate adaptive optics”. *J. Opt. Soc. Am. A*, vol. 11, no. 1, Jan. 1994, pp. 394–408.

Kingma, D. P., and J. Ba. “Adam: A Method for Stochastic Optimization”. 2014.

Kolmogorov, A. N. “The local structure of turbulence in incompressible viscous fluids for very large Reynolds numbers”. Trans. by V. Levin. *Doklady Akademii Nauk SSSR*, 1941.

Kornilov, V., et al. “MASS: a monitor of the vertical turbulence distribution”. *Adaptive Optical System Technologies II*. Ed. by Peter L. Wizinowich and Domenico Bonaccini, Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, Feb. 2003, pp. 837–45.

Lloyd-Hart, M., and N. M. Milton. “Fundamental limits on isoplanatic correction with multiconjugate adaptive optics”. *J. Opt. Soc. Am. A*, vol. 20, no. 10, Oct. 2003, pp. 1949–57.

Lombardi, G., et al. “Combining turbulence profiles from MASS and SLODAR: a statistical study of the evolution of the seeing at Paranal”. *Ground-based and Airborne Telescopes II*. International Society for Optics / Photonics, ed. by Larry M. Stepp and Roberto Gilmozzi, SPIE, 2008, p. 701221.

Mahajan, V. N. “Orthonormal aberration polynomials for anamorphic optical imaging systems with rectangular pupils”. *Applied Optics*, vol. 49, no. 36, 2010, pp. 6924–29.

McCulloch, W. S., and W. A. Pitts. “A logical calculus of the ideas immanent in nervous activity”. *Bulletin of Mathematical Biophysics*, vol. 478, no. 5, Dec. 1943, pp. 115–33.

Milton, N. M. *Tomographic Reconstruction of Wavefront Aberrations using Multiple Laser Guide Stars*. Doctoral Thesis, U of Arizona, 2009.

Mitchell, Tom. *Machine Learning*. McGraw Hill, 1997.

Neichel, B., et al. “Tomographic reconstruction for wide-field adaptive optics systems: Fourier domain analysis and fundamental limitations”. *J. Opt. Soc. Am. A*, vol. 26, no. 1, Jan. 2009, pp. 219–35.

Ono, Y. H., et al. “Multi time-step wavefront reconstruction for tomographic adaptive-optics systems”. *J. Opt. Soc. Am. A*, vol. 33, no. 4, Apr. 2016, pp. 726–40.

Osborn, J., et al. “Optical turbulence profiling with Stereo-SCIDAR for VLT and ELT”. *Monthly Notices of the Royal Astronomical Society*, vol. 478, no. 1, Apr. 2018, pp. 825–34.

Osborn, J., et al. “Open-loop tomography with artificial neural networks on CANARY: on-sky results”. *Monthly Notices of the Royal Astronomical Society*, vol. 441, no. 3, May 2014, pp. 2508–14.

Poyneer, L. A. “Scene-based Shack-Hartmann wave-front sensing: analysis and simulation”. *Appl. Opt.*, vol. 42, no. 29, Oct. 2003, pp. 5807–15.

Rigaut, F., and B Neichel. “Multiconjugate Adaptive Optics for Astronomy”. *Annu. Rev. Astron. Astrophys.*, vol. 56, 2018, pp. 277–314.

Rocca, A., et al. “Detection of atmospheric turbulent layers by spatiotemporal and spatioangular correlation measurements of stellar-light scintillation”. *J. Opt. Soc. Am.*, vol. 64, no. 7, July 1974, pp. 1000–04.

Roggemann, M. C., and B. Welsh. *Imaging through turbulence*. CRC P, 1996.

Sarazin, M., and F. Roddier. “The ESO differential image motion monitor”. *Astronomy and Astrophysics*, vol. 227, no. 1, Jan. 1990, pp. 294–300.

Saxenhuber, D., et al. “Comparison of methods for the reduction of reconstructed layers in atmospheric tomography”. *Appl. Opt.*, vol. 56, no. 10, 2017, pp. 2621–29.

Schöck, M., and E. J. Spillar. “Method for a quantitative investigation of the frozen flow hypothesis”. *JOSA A*, vol. 17, Sept. 2000.

Scott, R. P. *Development and Simulation of a Wide-Field Tomographic Wavefront Sensor for Use with an Extended Scene*. Doctoral Thesis, The U of Arizona, July 2021.

Shikhovtsev, A. Y. “A Method of Determining Optical Turbulence Characteristics by the Line of Sight of an Astronomical Telescope”. *Atmospheric and Oceanic Optics*, vol. 35, June 2022, pp. 303–09.

Tatarski, V. I. *Wave Propagation in a Turbulent Medium*. Trans. by R. A Silverman, McGraw-Hill Book Company, 1961.

Thomas, S., et al. “Comparison of centroid computation algorithms in a Shack-Hartmann sensor”. *MNRAS*, vol. 371, no. 1, Sept. 2006, pp. 323–36.

Tokovinin, A., and V. Kornilov. “Measuring turbulence profile from scintillations of single stars”. *Astronomical Site Evaluation in the Visible and Radio Range*. Ed. by Jean Vernin et al., Astronomical Society of the Pacific Conference Series, Jan. 2002, p. 104.

Tyson, R. *Principles of Adaptive Optics*. third, CRC P, 2011.

USAF. *Weather for Aircrews*. Vol. 1, Air Force, 1997.

Vidal, F., et al. “Tomography approach for multi-object adaptive optics”. *J. Opt. Soc. Am. A*, vol. 27, no. 11, Nov. 2010, A253–A264.

Vidal, F., et al. “Tomography reconstruction using the Learn and Apply algorithm”. *1st AO4ELT conference - Adaptive Optics for Extremely Large Telescopes*. 2010.

Whiteley, M. R., et al. “Temporal properties of the Zernike expansion coefficients of turbulence-induced phase aberrations for aperture and source motion”. *JOSA*, vol. 15, no. 4, 1998, pp. 993–1005.

Wilson, R. W. “SLODAR: measuring optical turbulence with a Shack-Hartmann wavefront sensor”. *Monthly Notices of the Royal Astronomical Society*, vol. 337, 2002, pp. 103–08.

Wizinowich, Peter L., et al. “The W. M. Keck Observatory Laser Guide Star Adaptive Optics System: Overview”. *Publications of the Astronomical Society of the Pacific*, vol. 118, no. 840, Feb. 2006, p. 297.