University of California
Santa Barbara

Readout and Calibration of Large Format Optical/IR MKID Arrays and Applications to Focal Plane Wavefront Control

A dissertation submitted in partial satisfaction
of the requirements for the degree

Doctor of Philosophy
in
Physics

by

Neelay Fruitwala

Committee in charge:

Professor Benjamin A. Mazin, Chair
Professor Omer Blaes
Professor Timothy Brandt

August 2021

The Dissertation of Neelay Fruitwala is approved.

Professor Omer Blaes

Professor Timothy Brandt

Professor Benjamin A. Mazin, Committee Chair

July 2021

Readout and Calibration of Large Format Optical/IR MKID Arrays and Applications to Focal Plane Wavefront Control

by

Neelay Fruitwala

To My Family

Acknowledgements

First I'd like to thank my advisor, Prof. Ben Mazin, for giving me the opportunity to work on a diverse range of exciting projects, and for his valuable mentorship and support throughout my PhD. I'd also like to thank my labmates for creating a fun and supportive work environment.

The work in chapter 2 was a collaborative effort between teams at Fermilab and UCSB. The Fermilab team (Gustavo Cancelo, Ted Zmuda, Ken Treptow, Neal Wilcer, and Chris Stoughton) designed the RF/IF and ADC/DAC boards and developed much of the Virtex 7 firmware for the ADC/DAC board. Br. Paschal Strader developed the Virtex 6 (ROACH 2) signal processing chain. Giulia Collura, Isabel Lipartito, and Ben Mazin designed and assembling the readout cartridges, power supplies, and other supporting hardware. Alex Walter, Nicholas Zobrist, Jeb Bailey (and really almost the entire Mazin Lab!) wrote key pieces of the control and calibration software.

For the work in chapter 3, I would like to thank Nicholas Zobrist, Isabel Lipartito, Sarah Steiger, Noah Swimmer, Jenny Smith, Kristina Davis, Rupert Dodkins, Jeb Bailey, Alex Walter, Ben Mazin, and Clint Bockstiegel for spending many tedious hours to help generate training and validation data.

Seth Meeker, Clint Bocksteigel, Alex Walter, Isabel Lipartito, and Nick Zobrist at UCSB supported the speckle nulling tests with DARKNESS at Palomar. Mike Bottom provided valuable assistance with setting up SDC and integrating it with DARKNESS, and provided me with a starting point for my own speckle nulling implementation. Mike's help was also crucial for performing the software integration between the P3K controller and MKID-based speckle nulling code. Thanks also to the Palomar Day crew and support astronomers for their continued support.

Commissioning/integration of MEC would not have been possible without the SCExAO

team in Hilo. Julien Lozi and Sebastian Vievard performed many of the critical in-person support tasks at the summit, including optical alignment, craning the instrument, and helping us troubleshoot issues with the readout and control electronics. Thanks as well to the Subaru day crew for taking time out of their day on numerous occasions to help us troubleshoot malfunctioning equipment. Olivier Guyon and Vincent Deo provided crucial support for integrating the MEC data stream and speckle nulling code with the SCExAO realtime control computer. At UCSB, on-sky speckle nulling tests were supported by Sarah Steiger, Noah Swimmer, Isabel Lipartito, and Alex Walter.

The work in chapter 2 was funded in part by an NSF ATI grant AST-1308556, and by US Department of Energy, Office of Science, Office of High Energy Physics contract No. DE-AC02-07CH11359 w/ Fermi Research Alliance, LLC. Special thanks to Xilinx for donating the FPGAs used for the readout system. For the work in chapter 4, I was supported by a grant from the Heising-Simons foundation.

Finally, I'd like to thank my friends and family for their continued support throughout my graduate school career.

Curriculum Vitæ

Neelay Fruitwala

Education

2021	Ph.D. in Physics (Expected), University of California, Santa Barbara.
2018	M.A. in Physics, University of California, Santa Barbara.
2015	B.S. in Physics, California Institute of Technology

Selected Publications

"End-to-end deep learning pipeline for microwave kinetic inductance detector resonator identification and tuning," **Neelay Fruitwala**, Alex B. Walter, John I. Bailey, Rupert Dodkins, and Benjamin A. Mazin, *Journal of Astronomical Telescopes, Instruments, and Systems*, 7(2), 028003 (2021).

"Second generation readout for large format photon counting microwave kinetic inductance detectors," **Neelay Fruitwala**, Paschal Strader, Gustavo Cancelo, et al, *Review of Scientific Instruments*, 91(12), 124705 (2020).

"The MKID exoplanet camera for Subaru SCExAO," Alex B. Walter, **Neelay Fruitwala**, Sarah Steiger, et al, *Publications of the Astronomical Society of the Pacific*, 132(1018), 125005 (2020).

Abstract

Readout and Calibration of Large Format Optical/IR MKID Arrays and Applications to Focal Plane Wavefront Control

by

Neelay Fruitwala

Microwave Kinetic Inductance Detectors (MKIDs) are a cryogenically cooled, superconducting detector technology with applications in astronomical observations in the radio through gamma-ray wavelengths. In the optical and IR, MKIDs have zero read noise, single photon sensitivity, microsecond time resolution, and broadband energy resolution. These properties make MKIDs ideal for directly imaging exoplanets, as high sensitivity/low noise are required for observing faint companions, and energy and time resolution are useful for resolving atmospheric aberrations which are the dominant noise source for such observations. Two MKID cameras have been built for this purpose: the 20,000 pixel MKID Exoplanet Camera (MEC) at Subaru Observatory and 10,000 pixel DARK-speckle Near-infrared Energy-resolving Superconducting Spectrophotometer (DARKNESS) camera at Palomar Observatory.

The first chapter of this thesis will discuss the development and testing of a second generation digital readout system for large format optical/IR MKID arrays. Our system retains much of the core signal processing architecture from the first generation system, but with a significantly higher bandwidth, enabling readout of kilopixel MKID arrays. Each set of readout boards is capable of reading out 1024 MKID pixels multiplexed over 2 GHz of bandwidth; two such units can be placed in parallel to read out a full 2048 pixel microwave feedline over a $4 - 8$ GHz band. As in the first generation readout, our system is capable of identifying, analyzing, and recording photon detection events in real

time with a time resolution of order a few microseconds. We describe the hardware and firmware, and present an analysis of the noise properties of the system. We also present a novel algorithm for efficiently suppressing IQ mixer sidebands to below -30 dBc.

The second chapter will focus on the development of a machine learning pipeline for automating the calibration of large MKID arrays. This process involves determining the resonant frequency and optimal drive power of every pixel (i.e. resonator) in the array, which is typically done manually. We instead use a deep-learning based object detection scheme to localize correctly-driven resonators in the 2D (power x frequency) frequency response data. Our method has performance equal to that of manual tuning, and only takes 12 minutes of computational time per 2000 pixels, as opposed to 4-6 hours for the manual method.

The final chapter will focus on the development of wavefront sensing and control algorithms to enable more sensitive exoplanet imaging. We implement an additional feedback loop between the MKID focal plane camera and adaptive optics (AO) system deformable mirror (DM) to correct residual optical aberrations in real time. Using a simple Fourier mode based technique (speckle nulling), we demonstrate in-lab convergence times < 1 second at a control rate of 30 Hz. We also show on-sky suppression of quasistatic speckles by 25% with a convergence time of 10 seconds. Preliminary work on a system identification scheme for improved calibration and control will also be discussed.

Contents

Curriculum Vitae **vii**

Abstract **viii**

1 Introduction **1**
1.1 High Contrast Imaging . . . 1
1.1.1 Adaptive Optics . . . 3
1.1.2 Coronagraphy . . . 5
1.1.3 Speckles . . . 6
1.1.4 Focal Plane Wavefront Control . . . 7
1.2 Microwave Kinetic Inductance Detectors . . . 8
1.2.1 Digital Readout for Large Arrays . . . 10
1.2.2 Frequency Comb Calibration . . . 11
1.3 MKIDs for High Contrast Imaging . . . 12
1.3.1 DARKNESS . . . 13
1.3.2 MEC . . . 14
1.4 Permissions and Attributions . . . 15

2 Second Generation Digital Readout **17**
2.1 System Overview . . . 17
2.1.1 ADC/DAC Board . . . 18
2.1.2 ROACH-2 Board . . . 20
2.1.3 RF/IF Board . . . 21
2.1.4 Timing . . . 22
2.2 Firmware and Software . . . 22
2.2.1 ADC/DAC (Virtex-7 FPGA) . . . 22
2.2.2 ROACH-2 (Virtex-6 FPGA) . . . 24
2.3 Performance . . . 29
2.3.1 Phase Noise . . . 29
2.4 Sideband Suppression . . . 33
2.5 Conclusion . . . 38

3 Deep Learning for Array Calibration **39**
3.1 Resonator Identification and Tuning: Baseline Methodology 39
3.1.1 Resonator Identification 40
3.1.2 Resonator Drive Power Tuning 41
3.1.3 Problems with the Baseline Methodology 43
3.2 End-to-end Machine Learning: System Overview 45
3.2.1 Overall Architecture 45
3.2.2 Neural Network 47
3.2.3 Resonator Identification 50
3.3 Final Model Selection and Training 52
3.4 Performance Evaluation 53
3.4.1 Resonator Identification 54
3.4.2 Drive Power Tuning 55
3.4.3 Frequency Placement 56
3.4.4 Computational Performance 57
3.5 Conclusion 57

4 Speckle Control **60**
4.1 Fourier-based Optical Propagation Model 60
4.2 Speckle Nulling Algorithm 63
4.2.1 Probing 66
4.2.2 Calculation of Nulling Solution 67
4.3 Realtime Data Pipeline 70
4.3.1 Raw Data Format 71
4.3.2 Packetmaster 71
4.3.3 Shared Memory Interface 76
4.4 MKID Speckle Nulling Code 78
4.4.1 Speckle Detection 78
4.4.2 Aperture Intensity Measurement 80
4.4.3 Multiple Probe Iterations 81
4.4.4 Codebase 82
4.5 In-lab Testing 83
4.5.1 DARKNESS 83
4.5.2 MEC 86
4.6 On-sky Test 91
4.7 Preliminary Tests of Expectation Maximization (EM) Algorithm for Calibration 97

A Room Temperature Phase Noise Calculation **103**

B Correcting for Line Noise **106**

C Noise Contributed by Sideband Tones **109**

Bibliography **111**

Chapter 1

Introduction

1.1 High Contrast Imaging

High contrast imaging (or direct imaging) is a technically challenging but useful technique for characterizing faint stellar companions such as debris disks and exoplanets. It is sensitive to regions of (mass/separation) parameter space complementary to other exoplanet detection and characterization techniques (figure 1.1), and is more naturally suited to spectroscopic characterization of companions. High contrast imaging using large thirty-meter-class ground based telescopes may even enable characterization of habitable-zone planets around M-stars.

Ground-based high contrast imaging systems generally require an adaptive optics (AO) system to correct for short-timescale ($\sim$ 10 ms) optical aberrations caused by atmospheric turbulence, followed by a coronagraph to optically suppress the diffraction pattern of the central star (known as the point spread function, or PSF). Current instruments are limited to a performance of $\sim 10^{-6}$ due to a "speckle" background of scattered starlight.

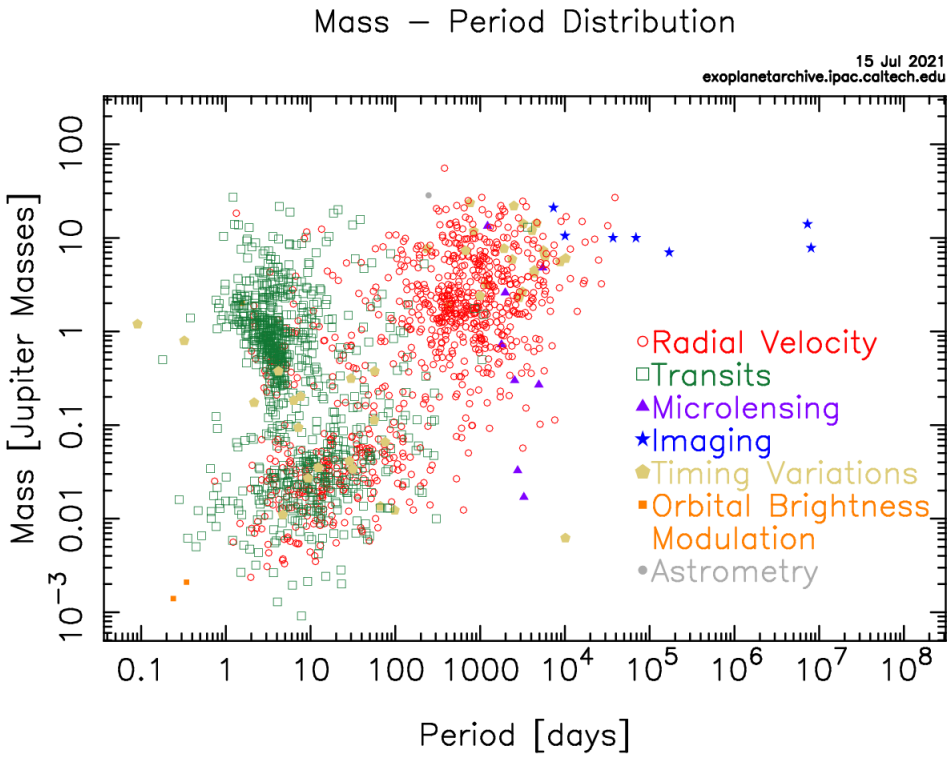


Figure 1.1: Mass vs orbital period plot of all currently known exoplanets. Unlike the radial velocity and transit methods, the sensitivity of high contrast imaging increases with separation (indicated here by orbital period). Figure reproduced from: `https://exoplanetarchive.ipac.caltech.edu`.

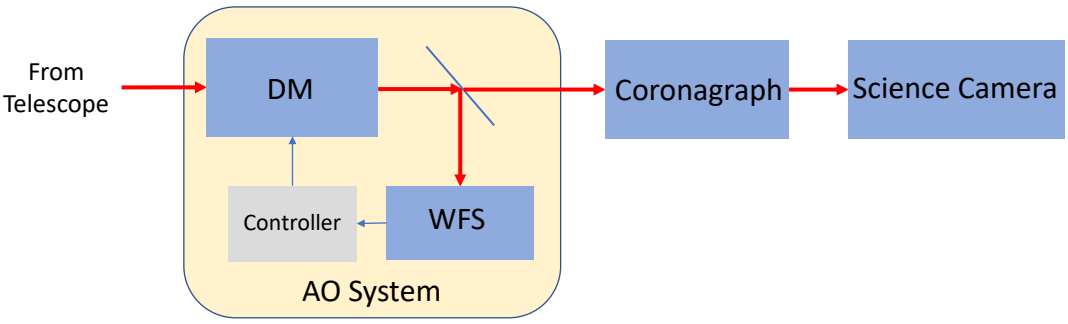


Figure 1.2: Block diagram of a basic high contrast imaging setup. Incoming light from the telescope is first stabilized/corrected by the AO system, then passed through the coronagraphic optics and finally imaged by the science camera. A beamsplitter or dichroic controls how the light is divided between the WFS and science camera; typically, shorter ($\lesssim$ 1000 nm) wavelengths are used for wavefront sensing while longer wavelengths are used for science imaging.

1.1.1 Adaptive Optics

AO is a control system used for correcting atmosphere-induced optical abberations. It is crucial for ground-based high contrast imaging, as atmospheric turbulence causes light from astronomical sources to diffract into a "seeing-limited" point spread function (PSF) with characteristic angular size ($\lambda/r_0 \sim 1$ arcsec) in the focal plane. This is comparable to the scale (i.e. angular distance between the primary and it's companions) of extrasolar systems of interest [1, 2], making it nearly impossible to distinguish between the primary and companion in the focal plane. Modern adaptive optics systems are capable of concentrating 70-90% [3] of starlight (and light from the companion) into a *diffraction limited* image; i.e. the only resolution limit is that imposed by the optics inside the telescope. The remaining scattered light outside of this diffraction pattern still poses a significant problem for high contrast imaging, which we discuss in section 1.1.3.

In its simplest form, an AO system has a wavefront sensor camera (WFS) for measuring the phase of the aberrated wavefront, and a deformable mirror (DM), for correcting the wavefront in real time. These components are generally connected in a closed-loop fashion; i.e. the WFS is downstream of the DM, such that it measures *residual* aberrations that remain uncorrected by the DM.

The most common wavefront sensors (and the ones used in this thesis) are Shack-Hartmann WFS and Pyramid WFS. A Shack-Hartman WFS consists of a lenslet array located in the pupil plane of the optical system. Each lenslet in the array will image that region of the pupil plane into a spot in the focal plane; the location of the spot can be used to infer the local slope of the wavefront [4]. A pyramid WFS consists of a pyramid-shaped optic with a tip located in the focal plane. This optic creates four images of the pupil (one for each pyramid face), the relative intensity patterns of which can be used to solve for the wavefront phase [5, 6]. Relative to the pyramid WFS, the Shack-Hartman

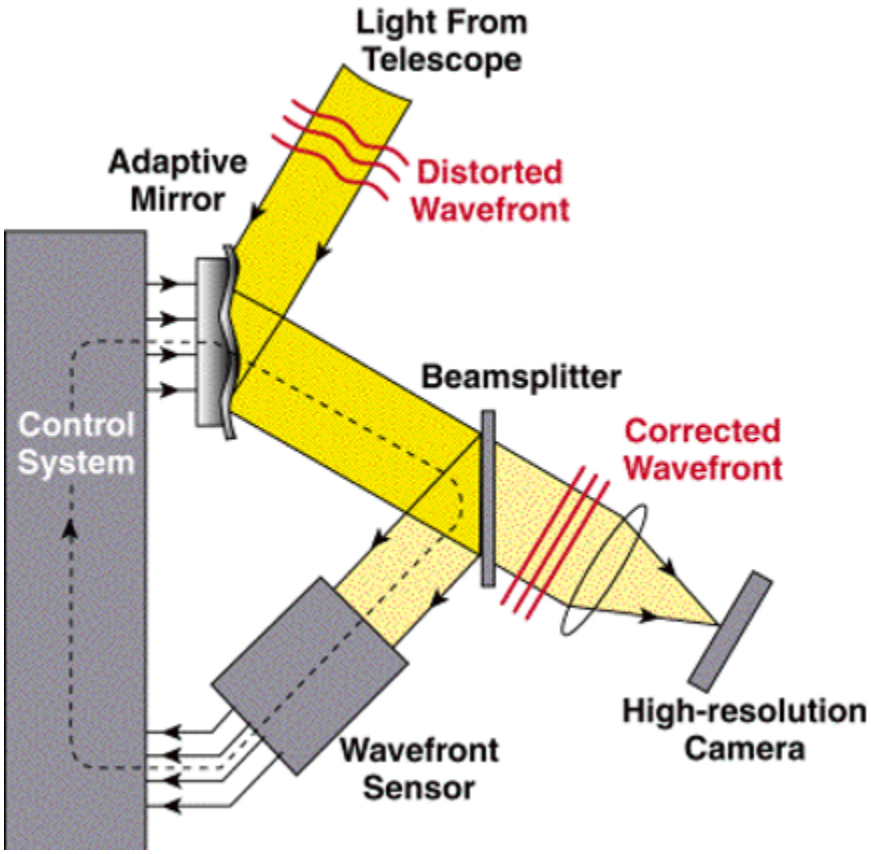


Figure 1.3: Schematic of a simple AO system. The phase of the distorted wavefront is measured by the WFS, and corrected in a closed-loop fashion by the DM. The DM is a flexible mirror whose shape is modulated by an array of actuators. WFS images are not generally useful for scientific measurements, so light is typically split between the WFS and a dedicated science camera. Reproduced from the Lyot Project (`lyot.org`)

is relatively stable and highly linear (i.e. the spot displacements are linear in wavefront phase error over a relatively large range), but suffers from low sensitivity to low-order aberrations [7].

Modern AO systems built with an eye towards high contrast imaging often have feedback rates upwards of 2 kHz and DM actuator counts in the few thousands [3, 8]. These high actuator count DMs allow for the correction of high-order (i.e. high spatial frequency) aberrations that may have relatively little effect on the stellar PSF but could interfere with signal from a companion. It is also common to have a multi-stage system, where each stage is tuned for correcting a certain aberration regime. One example of this is a "woofer-tweeter" architecture, where a high-stroke, low actuator count DM stage is used for correcting low-order aberrations, and is followed by a low-stroke, high actuator count DM stage for correcting high-order aberrations [3, 8, 9].

1.1.2 Coronagraphy

A coronagraph generally consists of a series of focal plane and pupil plane masks designed to suppress the diffraction pattern resulting from on-axis starlight, while maximizing the thoughput of any off-axis signal. The Lyot coronagraph (figure 1.4) is one of the simplest and most widespread designs; its advantages include broadband wavelength coverage and robustness to wavefront errors (especially tip/tilt PSF jitter). Other designs, such as the phase induced amplitude apodization (PIAA) or vector-vortex coronagraph offer increased suppression at smaller separations at the cost of increased sensitivity to wavefront error [7].

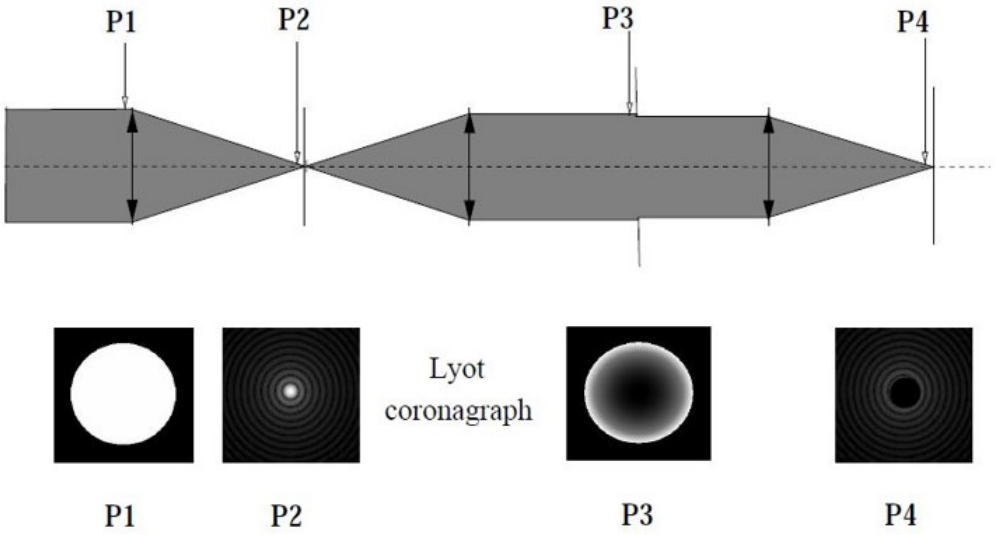


Figure 1.4: Schematic of a Lyot coronagraph, showing two conjugate pupil planes (P1 and P3), and two focal planes (P2 and P4). A focal plane mask at P2 blocks out the on-axis PSF core, while an optic known as the "Lyot stop" at P3 obscures the resulting diffraction ring around the edge of the pupil. The final (under ideal conditions!) focal plane image is shown at P4. Figure reproduced from Ref. [10].

1.1.3 Speckles

The dominant noise source in high contrast imaging is the speckle background; i.e. a pattern of scattered starlight that can obscure the light from a companion. This speckle background has two major sources:

1. **Quasistatic** speckles, or non-common path aberrations (NCPA), result from optical aberrations that are downstream of the AO system (and so aren't sensed or corrected for). These aberrations are caused by mechanical changes/imperfections in the instrument, and vary on relatively slow timescales (tens of minutes).

2. **Atmospheric** speckles result from atmosphere-induced aberrations that are not corrected by the AO system. These vary on timescales of ~ 10 ms [11]. These have a variety of possible causes, including [7]:

    - Chromaticity: difference in wavelength between wavefront sensor and science

images

- Bandwidth Error: atmospheric fluctuations that have higher frequency than the AO system control bandwidth (i.e. feedback loop frequency)
- Amplitude aberrations
- WFS measurement noise: depending on the camera and system configuration this could be either detector read noise or photon noise

A variety of post-processing techniques exist for mitigating the impact of quasistatic speckles, such as angular differential imaging (ADI) and spectral differential imaging (SDI). Atmospheric speckles too quickly to be effectively imaged for subtraction in post-processing in conventional imagers; they will generally average down into a diffuse "halo" around the stellar PSF.

1.1.4 Focal Plane Wavefront Control

Focal plane wavefront control (FPWC) involves measuring and correcting wavefront error using data from the focal plane science camera. Since FPWC uses the same images for both science and wavefront sensing, it avoids both chromatic wavefront error and NCPA, making it a useful tool for reducing the speckle background. FPWC requires a stable, well calibrated system, so the main AO loop is still required to perform the initial correction; FPWC is run concurrently as an additional input to the DM (figure 1.5).

Focal plane images only measure electric field intensity, so there is no straightforward way to directly extract the wavefront phase. This is usually accomplished by applying "probe" patterns to the DM which coherently interfere with the speckle pattern. With enough probes, the set of interference patterns can be used to calculate the electric field phase, allowing the aberrations to be corrected by the DM. Performing this calculation requires some form of a calibrated system model; i.e. knowledge of the focal plane electric

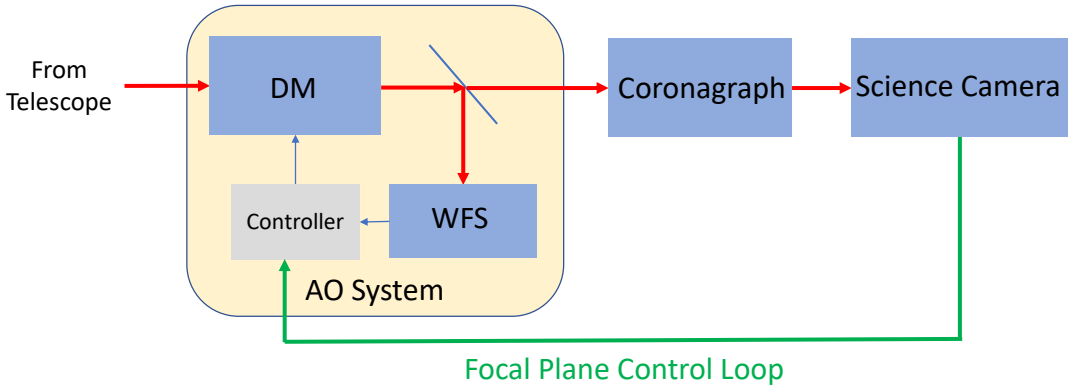


Figure 1.5: Schematic of focal plane wavefront control (FPWC). Abberations are measured in the focal plane by the science camera, and corrected in real time by the deformable mirror (DM). FPWC has several advantages: it is not limited by WFS sensitivity or chromatic differences between the WFS and science camera, and allows for the correction of NCPA not seen by the WFS. Since FPWC is implemented as a secondary control loop on top of the main AO loop, corrections must be applied as offsets to the convergence point of the primary loop (if corrections were applied directly to the DM, it would be sensed as an aberration by the WFS and removed by the main loop).

field response to each applied DM probe. Some control algorithms, such as electric field conjugation (EFC) [12], require the full Jacobian matrix mapping DM actuator offsets to per-pixel electric field response, while others, such as speckle nulling [13, 14], use a Fourier-based approximation requiring only a simple calibration. In general, simpler approaches sacrifice performance (in terms of achievable contrast) for robustness.

Speckle nulling at low framerates ($< 1Hz$) has been demonstrated on ground-based systems; including in-lab tests at the Palomar and Keck Observatories [13], and an on-sky test at Subaru Observatory [15].

1.2 Microwave Kinetic Inductance Detectors

An MKID is a cryogenically cooled superconducting resonator consisting of an inductor/capacitor pair. The circuit gets its inductance primarily from the kinetic energy of the Cooper pairs in the supercurrent (hence the term kinetic inductance). When a photon

is absorbed by the inductor, some of the Cooper pairs are broken (creating free electrons known as "quasiparticles"), changing the inductance and hence the resonant frequency and quality factor of the circuit. The number of quasiparticles created is proportional to the energy of the incident photon, which gives the MKID intrinsic energy resolution. To read out the detector, the resonator is driven by a microwave probe signal tuned to its resonant frequency and the probe signal is monitored for changes in its relative phase. A photon detection event manifests as a pulse in phase with a steep rise ($\sim 1\ \mu s$) and an (approximately) exponential decay with ($\tau \approx 15\ \mu s$). Our devices are designed to operate in the linear regime, where $\delta\phi \propto \delta f \propto \delta E$ [16, 17].

One of the primary advantages of MKIDs over other superconducting detector technologies is the intrinsic frequency domain multiplexibility; the resonant frequency of each pixel can be easily tuned in fabrication. This allows large numbers of resonators, each tuned to a different frequency, to be placed in parallel on a single microwave channel. This greatly simplifies the cryogenic wiring and electronics, shifting the complexity to the room temperature readout system [18, 19, 20, 21].

MKIDs' single photon sensitivity, intrinsic energy resolution, and microsecond-scale time resolution make them ideally suited to the study of highly time-variable phenomena such as pulsars [22] and compact binaries [23]. MKIDs are also suited to serve as IFUs (integral field units) for exoplanet direct imaging, which requires high sensitivity (due to the extremely low flux from companions), and benefits from high time resolution (to resolve atmospheric effects). The first generation optical/IR MKID instrument, ARCONS (Array Camera for Optical to Near-IR Spectrophotometry, a 2 kilopixel IFU at Palomar Observatory) [19], was built primarily for high time resolution astronomy, while two second generation instruments, DARKNESS (DARK-speckle Near-infrared Energy-resolving Superconducting Spectrophotometer, a 10 kilopixel IFU at Palomar)[20], and MEC (MKID Exoplanet Camera, a 20 kilopixel IFU at Subaru Observatory)[21], were

purpose-built for high contrast imaging. A third instrument, PICTURE-C, a 10 kilopixel IFU for balloon-borne high contrast imaging, is currently under development [24].

1.2.1 Digital Readout for Large Arrays

The readout procedure for large format arrays generalizes the single pixel readout described in section 1.2. A comb of probe tones, one at each resonator frequency, is used to drive the MKIDs. Each resonator acts as a notch filter centered on its resonant frequency, modifying the phase and amplitude of its probe tone while leaving other tones comparatively untouched. After transmission through the array, the tones are downconverted and digitized, and each tone is independently montitored for changes in its relative phase.

A first generation readout for optical MKIDs was developed in 2011 for the ARCONS instrument. This system is capable of reading out 256 channels in a 512 MHz band on each readout board [25]. Eight boards were used simultaneously to readout the 2048 pixel ARCONS instrument. In order to feasibly scale up to the kilopixel arrays used by the second generation of optical MKID instruments, we have developed a new readout system capable of reading out up to 1024 channels per unit over 2 GHz of bandwidth. This system largely retains the signal processing algorithm chain found in the first generation readout, albeit with major changes to the hardware and firmware to accommodate the higher bandwidth.

The second generation of MKID arrays are in the 10-20 kilopixel range, with pixels split up into several microwave feedlines, each containing 2000 pixels inside the 4-8 GHz band. We expect a maximum photon event rate of 2500 counts/pixel/second. Due to data transmission and storage constraints, we require that each event be identified, analyzed, and recorded in real time. Following these constraints, we define the following system

requirements:

- Generation of a 2000 tone comb in a 4-8 GHz band, with arbitrary tone frequencies and powers.
- Channelization: the ability to isolate a 200-500 kHz wide channel centered around each tone
- Low noise: the noise floor of each channel must be lower than the device noise and cryogenic amplifier noise
- Sample each channel at a rate of $\sim 1\ \mu s$
- Implement a filtering and triggering system that is capable of detecting photon events in real time and accurately determining the associated phase pulse amplitude

In chapter 2, we describe the design, development, and testing of the second generation readout system.

1.2.2 Frequency Comb Calibration

Calibrating the comb of probe tones used to drive the MKID array consists of the following two stages: 1) identification of resonator frequencies; and 2) tuning the drive power of each resonator probe tone. Due to imperfections in the array fabrication process, resonator frequencies deviate significantly more (from their design values) than the average resonator frequency spacing, and must be measured empirically. Additionally, the photon detection SNR of each resonator is very sensitive to the power of its input probe tone [26, 27]. This "ideal" drive power varies considerably across the array and must be tuned individually for each resonator.

The calibration process requires significant manual intervention and can be prohibitively time consuming; calibrating a single 2000 pixel feedline can take 4-6 hours.

Because calibration is required to fully characterize an MKID array, this process can present a significant bottleneck for testing new devices.

Machine learning has been shown to be a promising method for tuning resonator drive power [28]. However, using machine learning to tune power alone (and not frequency) suffers from a few pitfalls, and we find that manual tuning is still required to refine the ML selected powers. Building on this earlier work, we developed a unified deep learning framework for performing both frequency and power calibration steps simultaneously. In chapter 3, we provide an overview of our algorithm and present the results of an end-to-end performance test.

1.3 MKIDs for High Contrast Imaging

MKIDs' single photon sensitivity, microsecond time resolution, and intrinsic energy resolution in the near-IR (700-1400 nm) wavelength band make it an ideal detector technology for both high-contrast science and for characterizing and removing the speckle background. A number of speckle subtraction techniques utilizing MKIDs are being explored, both in realtime and post-processing.

One such post-processing technique is stochastic speckle discrimination (SSD). SSD exploits the fact that the temporal intensity distribution resulting from coherently interfering speckles (e.g. "pinned" atmospheric speckles interfering with quasistatic speckles) obeys a Modified-Riccian (MR) function, while the intensity from an incoherent source (e.g. companion object) is Poisson distributed. MKIDs are ideally suited for SSD analysis, as observing the MR distribution requires exposures on similar timescales as atmospheric speckle convergence times ($\sim$ 10 ms). SSD has been implemented using both MEC and DARKNESS data. Recently, SSD with MEC has aided in the detection of a ≈ 6 AU companion around HIP 109427 [29]. A bin-free version of SSD, which operates

directly on the distribution of photon arrival times, is also being explored [30].

In realtime, both coherent differential imaging (CDI) based techniques and FPWC are being explored. For CDI, the DM is modulated by a pre-defined probe pattern at a rate faster than the atmospheric speckle coherence time. The interference pattern produced by this modulation can be used to characterize (both atmospheric and quasistatic) speckles and distinguish them from any incoherent source.

FPWC with MKIDs can enable faster, more efficient subtraction of quasistatic speckles (leaving more time for science observations), and could also potentially be used to subtract fast-varying atmospheric speckles. In chapter 4, we describe an implementation of the speckle nulling algorithm using MKIDs as a focal plane wavefront sensor.

1.3.1 DARKNESS

The Dark-speckle Near-IR Energy-resolved Superconducting Spectrophotometer (DARKNESS) is a 10,000 pixel MKID IFU (integral field unit), designed to interface with the stellar double coronagraph (SDC) behind the PALM-3000 (P3K) AO system [9] at the Palomar Observatory 200" Hale telescope. DARKNESS is tuned to near-IR (700-1400 nm) wavelengths, optimized for observing companions around low-mass K and M type stars. SDC is a modular coronagraphy platform that is capable of using a variety of coronagraph masks, including a vector vortex coronagraph (VVC) capable of achieving inner working angles $\sim 1\lambda/D$ [31]. P3K utilizes a Shack-Hartman wavefront sensor, and has two deformable mirrors (one with 3388 actuators and another with 241) for both high and low order corrections. It can achieve control loop frequencies of up to 2 kHz. [32].

My contributions to DARKNESS include readout integration and testing at Palomar, developing calibration and control software, and device testing. I also performed on-sky

Figure 1.6: Photograph of DARKNESS (left) mounted underneath P3K in the Palomar Observatory AO lab.

observations and in-lab speckle nulling tests.

1.3.2 MEC

The MKID Exoplanet Camera (MEC) is a scaled up version of DARKNESS, with a 20,000 pixel MKID array. It serves as a science camera and focal-plane wavefront sensor (FPWS) behind SCExAO (Subaru Coronagraphic Extreme Adaptive Optics) at the 8.2 meter Subaru telescope on Mauna Kea. SCExAO is a high contrast imaging instrument with an integrated coronagraph and AO system, operating behind the Subaru telescope AO188 facility AO system. It has a 2000 actuator deformable mirror (DM) and pyramid wavefront sensor for high-order wavefront correction, complementing the low order facility AO system. The system is capable of control loop frequencies of up to 3.5 kHz. With a modified PIAA or vortex coronagraph, SCExAO can achieve a contrast of 10^{-6} at 1 λ/D [33]. Unlike DARKNESS, MEC is permanently mounted to the backend of SCExAO (along with several other science cameras), greatly increasing the availability and ease of

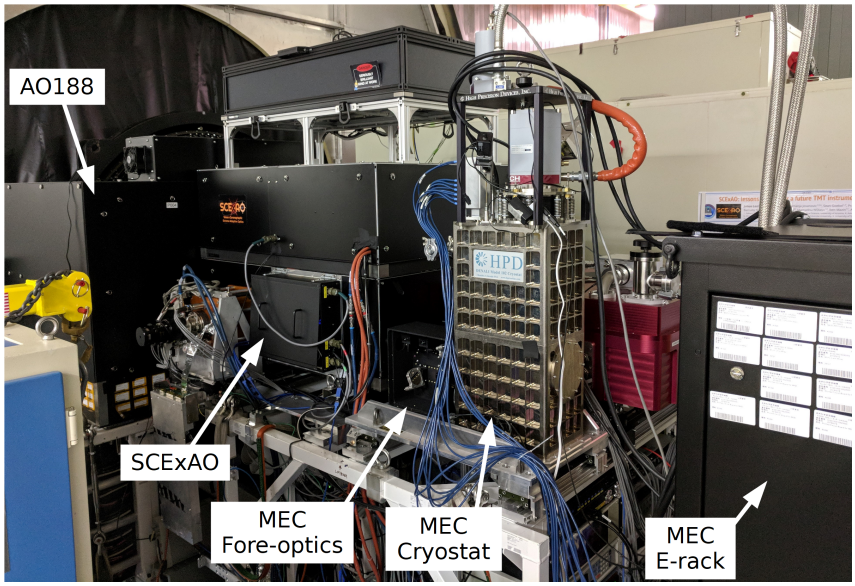


Figure 1.7: SCExAO bench at the nasymth-IR platform at Subaru Observatory, with MEC attached as a science camera. Figure reproduced from [21].

in-lab bench testing.

My contributions to MEC include device simulations and fabrication mask layout, readout system testing/deployment, optical alignment at SCExAO, and software integration with the SCExAO realtime controller (RTC).

The SCExAO realtime control software was written using Compute And Control for Adaptive Optics (CACAO) [34]. CACAO is a modular, hardware agnostic software package that provides a framework for implementing complex control algorithms using multiple sensor inputs and controller outputs (e.g. DM, tip tilt mirror). It provides a unified shared memory format for realtime I/O, and has utilities for calibration, telemetry, and process management.

1.4 Permissions and Attributions

1. The content of chapter 2 and appendices A, B, and C is the result of a collaboration with Paschal Strader, Gustavo Cancelo, Ted Zmuda, Ken Treptow, Neal Wilcer,

Chris Stoughton, Alex B. Walter, Nicholas Zobrist, Giulia Collura, Isabel Lipartito, John I. Bailey III, and Benjamin A. Mazin, and was originally published in the AIP Review of Scientific Instruments [35].

2. The content of chapter 3 is the result of a collaboration with Alex B. Walter, John I. Bailey III, Rupert Dodkins, and Benjamin A. Mazin, and was originally published in the Journal of Astronomical Telescopes, Instruments, and Systems [36].

Chapter 2

Second Generation Digital Readout

2.1 System Overview

The full readout system consists of several readout units, each of which can read out up to 1024 resonator channels in a 2 GHz band. Two such units can be placed in parallel to read out a full 4 GHz feedline. Because the resonant frequency band lies outside the range of readily available ADCs, we use an IQ modulation scheme to perform up/down conversion between the digital room temperature electronics (-1 to 1 GHz) and the resonators (4 to 6 GHz or 6 to 8 GHz). Each readout unit has three major components: 1) CASPER (Collaboration for Astronomy Signal Processing and Electronics Research) ROACH-2 (Reconfigurable Open Architecture Computing Hardware) board with a Xilinx Virtex-6 FPGA for channelization and pulse detection, 2) Custom ADC/DAC board with dual 2 GSPS Analog Devices AD9625 ADCs and AD9136 DACs and Xilinx Virtex-7 FPGA for control, and 3) RF/IF board for IQ modulation.

The basic outline of the readout procedure (illustrated in figure 2.1) is as follows:

1. Tone generation in [-1 GHz, 1 GHz] (IF band) using dual 2 GSPS DACs.

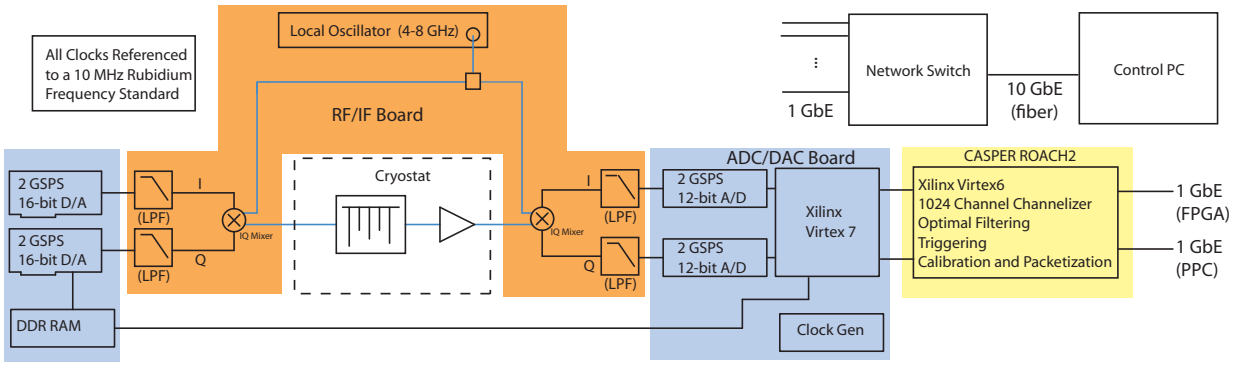


Figure 2.1: System block diagram showing a single 2 GHz, 1024 channel readout unit. Reproduced with permission from Publ. Astron. Soc. Pac 130, 988 (2018). Copyright 2018 The Astronomical Society of the Pacific.

2. Upconversion to RF band via IQ mixing tones with local oscillator (LO) (figure 2.2).

3. Tones filtered through resonators, amplified by cryogenic HEMT (high electron mobility transistor).

4. Output is downconverted by IQ mixer, then digitized by dual 2 GSPS ADCs.

5. ADC stream is sent to Virtex-6 FPGA for channelization, filtering, and photon triggering.

6. Photon events (time and pixel tagged pulse height) are streamed over ethernet to data server.

2.1.1 ADC/DAC Board

Due to the complexity of modern, high sample rate ADCs and DACs, we have developed a dedicated circuit board for signal routing and control of these components. The board features dual 12-bit 2 GSPS ADCs (Analog Devices AD9625), and dual 16-bit 2 GSPS DACs (Analog Devices AD9136). Dual ADC/DACs are required for complex

sampling; one unit is required for each of I and Q. The board also has an onboard Xilinx Virtex-7 FPGA (XC7VX330T-2FFG1761C) for control of the ADCs and DACs, routing of the ADC signal to the ROACH-2, and communication with the ROACH-2 and RF/IF boards. A LMK04821 synthesizer is used to generate all clocks required by the ADCs, DACs, and FPGAs on both the ROACH-2 and ADC/DAC boards, synchronized to an external 10 MHz reference signal.

The DAC output is generated using lookup table (LUT) that contains the sum of all of the resonator probe tones in the time domain. The LUT is generated in software and stored in onboard DDR3 RAM. Two DACs are used to generate a complex output signal, one for I and one for Q. A table size of 262144 values (for each of I and Q) imposes a tone frequency quantization of 7.629 kHz. The LUT approach allows us to generate the DAC output for an arbitrary list of resonator frequencies, and independently specify the drive power for each resonator (required to optimize photon pulse SNR [27, 26]). To generate the LUT, tones are first calculated and summed together in software according to their IF band frequencies and relative powers. Tone phases are randomized to maximize DAC dynamic range utilization. The full LUT is then scaled to fit the DAC dynamic range, and the programmable output attenuators are adjusted to set the tones to the desired power. Programmable attenuators in the RF output chain ensure full utilization of the DAC dynamic range over a wide range of tone powers.

The IQ ADC/DAC inputs/outputs are connected to the RF/IF board via SMP connectors. There are also several SPI interfaces to enable the Virtex-7 to program the attenuators and local oscillator on the RF/IF board.

The raw ADC output is sent to the Virtex-6 FPGA on the ROACH-2 over a ZDOK connector 16 samples (8 each from I and Q) per clock in a 192-bit parallel bus clocked at 250 MHz.

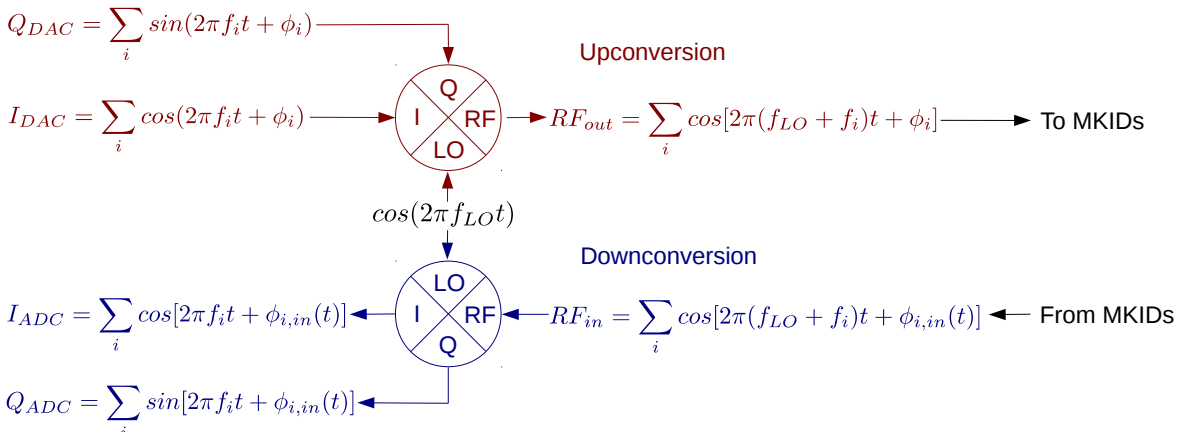


Figure 2.2: Schematic of RF up/downconversion scheme. The frequency comb is written explicitly as a sum over the probe tones, where $f_i \in [-1\ GHz,\ 1\ GHz]$ is the IF band tone frequency, and $f_{LO} \in [4\ GHz,\ 8\ GHz]$ is the LO frequency. ϕ_i is the probe tone phase in the DAC LUT. $\phi_{i,in}(t) = \phi_i + \phi_{i,cable} + \delta\phi_{i,res}(t)$ is a time dependent phase term that includes the DAC LUT phase, cable delay, and phase change induced by the resonator. The same LO is used for both upconversion and downconversion to suppress phase/frequency errors in the LO signal.

2.1.2 ROACH-2 Board

The ROACH-2 was developed by the CASPER collaboration at UC Berkeley [37, 38]. It was designed around the Xilinx Virtex-6 FPGA (XC6VSX475T-1FFG1759C). It has a PowerPC CPU, DDR3 and QDR memory modules, and various communications interfaces such as ethernet. All of the core digital signal processing for the readout system (channelization, filtering, and pulse detection) is implemented on the Virtex-6 FPGA. We chose the ROACH-2 platform because CASPER provides an extensive dedicated toolset and programming interface for the FPGA. The CASPER toolflow contains blocks for performing complex signal processing tasks (such as the polyphase filter bank), as well as interfaces to much of the hardware on the ROACH-2. CASPER has also released a series of python libraries for programming and communication with the FPGA over ethernet.

2.1.3 RF/IF Board

IQ Upconversion and Downconversion

The RF/IF board implements IQ modulation for analog up/downconversion between the IF band (-1 – 1 GHz) and the RF band (4 – 6 GHz or 6 – 8 GHz). For the upconversion process, the complex IF band frequency comb is mixed with a local oscillator (LO) tone at the center of the RF band (i.e. a 4 – 6 GHz resonator band would use a 5 GHz LO), which generates a real-valued output that is shifted into the resonator frequency band. Downconversion is the reverse process, where the real valued RF output from the cryostat is mixed with the same LO to generate a complex-valued IF band signal which can be digitized by the ADCs. This process is illustrated schematically in figure 2. The LO is generated on the board by a programmable phase locked loop (PLL) IC (Texas Instruments TRF3765). The PLL is referenced to an external 10 MHz signal provided by the ADC/DAC board. A frequency doubler is used on the output of the TRF3765 to shift the frequency range into the resonator band (600 MHz – 9.6 GHz). Analog devices HMC525LC4 I/Q mixers are used for upconversion and downconversion.

RF Output Chain

Two Peregrine Semiconductor PE43705 programmable step attenuators provide 0 to 63.5 dB (steps of 0.25 dB) of attenuation on the output frequency comb, which enables full utilization of DAC dynamic range over a range of resonator readout powers. The maximum output power of the full RF chain is -12.5 dBm.

RF Input Chain

In order to ensure full utilization of the ADC dynamic range over a wide range of resonator drive powers, the RF/IF board has an amplifier/attenuator chain before the

IQ mixer. Four Hittite HMC3587 provide 60 dB gain, and two programmable step attenuators (63.5 dB total in steps of 0.25 dB) are used to optimize the input power at the ADC and ensure that the amplifiers are kept away from saturation (see figure A.1 for schematic). The optimal signal power at the end of the amplifier chain (IQ mixer input) should be approximately -7 dBm; this implies an input power range of -64 dBm to -0.5 dBm [1]. In practice, we find that in order to keep the noise from the room temperature chain sufficiently low, the total programmable attenuation should not exceed $\approx$ 30 dB, providing an actual input power range of -64 dBm to -34 dBm (see Appendix A for full calculation).

2.1.4 Timing

The ROACH-2 firmware has a timing block that keeps track of the absolute (UTC) time with microsecond precision. In order to ensure accurate absolute timing, this block can be synchronized to an external reference using a pulse-per-second (PPS) input. In our current implementation, the PPS input and the 10 MHz reference used to generate the FPGA clocks are generated by a spectracom timing module that is synchronized to a GPS satellite reference.

2.2 Firmware and Software

2.2.1 ADC/DAC (Virtex-7 FPGA)

The Virtex-7 firmware is responsible for controlling the ADCs, DACs, and programmable components on the RF/IF board, and for routing the ADC and DAC data streams. Communication with the Virtex-7 is conducted over a UART interface to the

[1] There is a fixed 3 dB of attenuation in addition to the programmable attenuators, so the power at the end of the chain is given by $P_{out} = P_{in} + 60\ dB - A_{prog} - 3\ dB$

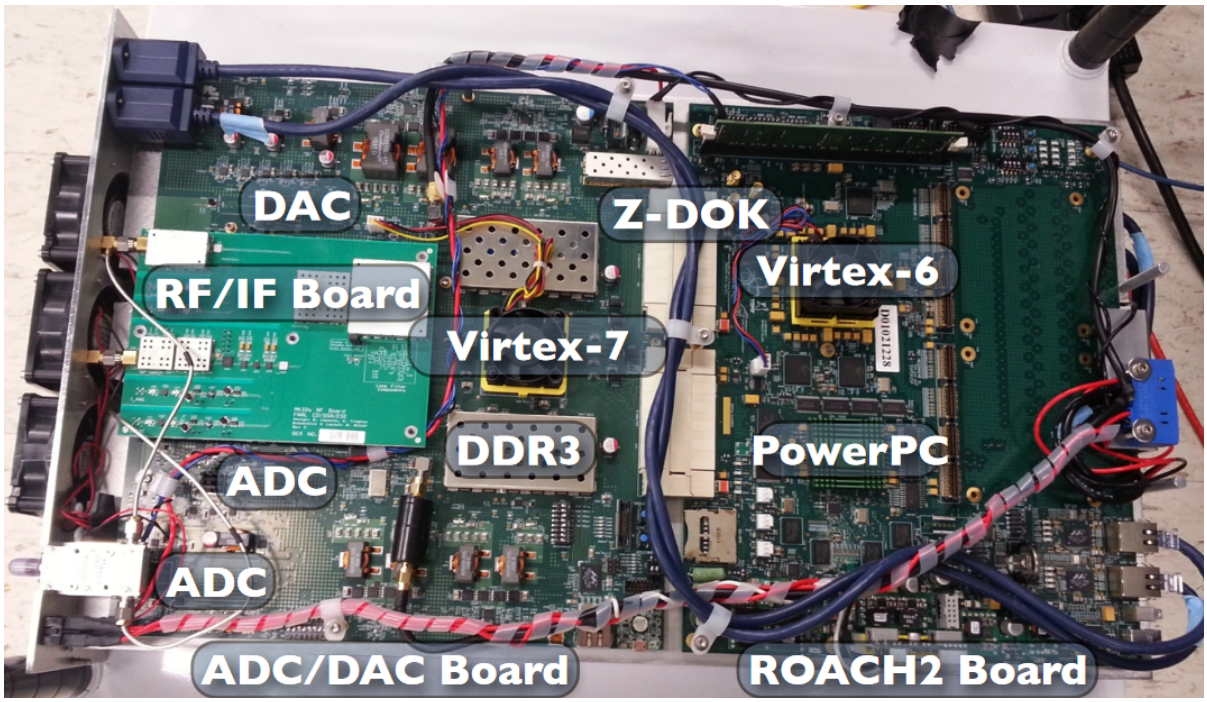


Figure 2.3: Cartridge containing an assembled readout unit. The ROACH-2 board is connected to the ADC/DAC board by two ZDOK connectors. The RF/IF board is mounted on the ADC/DAC board using SMP blind-mate connectors for signals and GPIO pins for programming. Another readout unit is mounted to the underside of this cartridge. Figure and Caption reproduced with permission from Paschal Strader, "Digital Readout for Microwave Kinetic Inductance Detectors and Applications in High Time Resolution Astronomy", Ph.D. dissertation (University of California at Santa Barbara, 2016).

ROACH-2 Virtex-6 FPGA, which can receive and forward commands from the control computer over ethernet. The low-level firmware and communications interfaces are controlled using a Xilinx MicroBlaze soft processor. The MicroBlaze runs a C script which initializes all major peripherals, then recieves and executes commands sent by the ROACH-2 over the UART interface. These commands include programming the RF attenuators and LO, receiving the DAC LUT and writing it to DDR3, and resetting the ADCs and DACs. The firmware was developed primarily at Fermilab, and testing and integration with the ROACH-2 were performed at both Fermilab and UCSB.

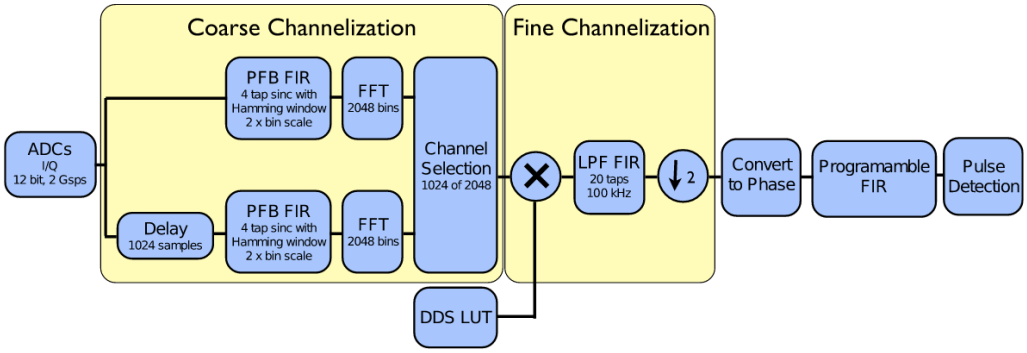


Figure 2.4: ROACH-2 firmware block diagram. Data is streamed over the ZDOK over a 16-sample bus (8 12-bit I/Q pairs) clocked at 250 MHz. The data is then copied and processed by two parallel PFBs; one copy is delayed by 128 clocks (half of the complete FFT cycle) to maintain a uniform sampling interval of each bin at 2 MHz [25]. After the PFB stage, up to 1024 resonators are selected from the bins. Following channel selection, DDC is performed on each channel to center the resonator tone within the frequency bin. Channels are then low pass filtered (100 kHz on each of I and Q) and converted to phase. The phase is filtered using a channel-specific "optimal" FIR. Phase pulses meeting the trigger conditions are recorded and streamed to the data server over 1 GBit ethernet. Figure adapted with permission from Paschal Strader, "Digital Readout for Microwave Kinetic Inductance Detectors and Applications in High Time Resolution Astronomy", Ph.D. dissertation (University of California at Santa Barbara, 2016).

2.2.2 ROACH-2 (Virtex-6 FPGA)

The Virtex-6 firmware is responsible for the bulk of the digital signal processing performed by the system: conversion from the raw 2 GHz I/Q stream from the ADCs to time and phase (or energy) tagged photons. This process can be divided into four stages: channelization, phase conversion, optimal filtering, and photon event triggering. The firmware was developed at UCSB using the MATLAB Simulink and ISE based CASPER ROACH toolflow. All of the firmware blocks are clocked at 250 MHz using a clock supplied by the ADC/DAC board over the ZDOK connectors.

Channelization

Channelization is the process of converting the raw 2 GHz I/Q stream from the ADC into a time-multiplexed series of frequency bins, each centered around a resonator probe tone. We use the same two stage approach as the first generation readout, as described in (Ref. [25]). The first stage is a dual 2048-sample (more specifically, 2048 branch, 4 tap) polyphase filter bank (PFB), which consists of a finite impulse response (FIR) filter followed by a Fast Fourier Transform (FFT). The configuration is the exact same as in (Ref. [25]), except for the total bandwidth/number of points (2048 samples over 2 GHz, instead of 512 samples over 512 MHz). As in (Ref. [25]), oversized 2 MHz bins are used, so two parallel PFBs are required to Nyquist sample each bin. Oversized bins ensure that there exists a bin with at least 500 kHz of bandwidth around every resonator, no matter its resonant frequency. The output of the PFB+FFT is time multiplexed at $(2 \times FFT) \times (8\ bins/clock) = 16\ bins/clock$. The FPGA is clocked at 250 MHz, so the sample rate of each bin is $(250\ clocks/s) \times (16\ bins/clock)/(2048\ bins) \approx 2$ MHz.

After the FFT, the bins are sorted into 1024 resonator channels; for each resonator frequency, the bin with the closest center frequency is selected. The allowed spacing between resonators can be as low as 200 kHz, so there may be as many as 10 resonators per bin. These channels are split into 4×256-channel time-multiplexed "streams", with each resonator getting 2 samples per clock (one from each FFT). This maintains the 2 MHz channel sample rate: (250 MHz) $\times$ $(2\ samples)/(256\ channels) \approx 2$ MHz. Digital down conversion (DDC) is then performed on each channel to center it at 0 Hz: each channel is multiplied by a (complex) sinusoid with $f = f_{bin} - f_{res}$. These sinusoids are generated using a lookup table which is stored on the ROACH-2 QDR memory.

Low Pass Filtering and Phase Conversion

After the channelization stage, each channel is low pass filtered to set the desired channel bandwidth. This is done by separately convolving I and Q with a 20-tap Hanning-windowed sinc function. The following considerations should be taken into account when setting the channel bandwidth: 1) desired time resolution/response time of the detector (1 μs); 2) photon event (phase pulse) SNR; 3) desired minimum resonator spacing. Exoplanet imaging requires framerates of at most a few kHz ($\sim$ 100 μs), so time resolution was not a major consideration. We found that a channel bandwidth of 200 kHz ($\pm$ 100 kHz around the tone center; which gives 100 kHz bandwidth after phase conversion) provides good performance for considerations (2) and (3). This bandwidth preserves the frequency content of a typical pulse (figure 2.5) while attenuating much of the high frequency line noise present in many resonator channels (figure 2.7). It also allows us to impose a minimum channel spacing of 200 kHz, reducing the number of frequency collisions to $\approx$ 6% of identified resonators. After the low pass filter, the channels are downsampled to 1 MHz, and the I/Q values are converted to phase using a CORDIC algorithm that implements $arctan(Q/I)$.

Optimal Filtering

After phase conversion, each channel is filtered using a 50-point FIR (finite impulse response) filter. The filter coefficients are programmable and are unique to each channel. This customizability is a key feature, as it allows us to tailor each filter to that channel's noise PSD and photon pulse shape. To generate these filters, we use the formalism described in (Ref. [39]) and extended in (Ref. [40]). This "optimal filter" formalism uses the noise covariance and a template of the photon pulse to produce a minimum variance linear estimator of the photon pulse amplitude. The number of filter coefficients is well

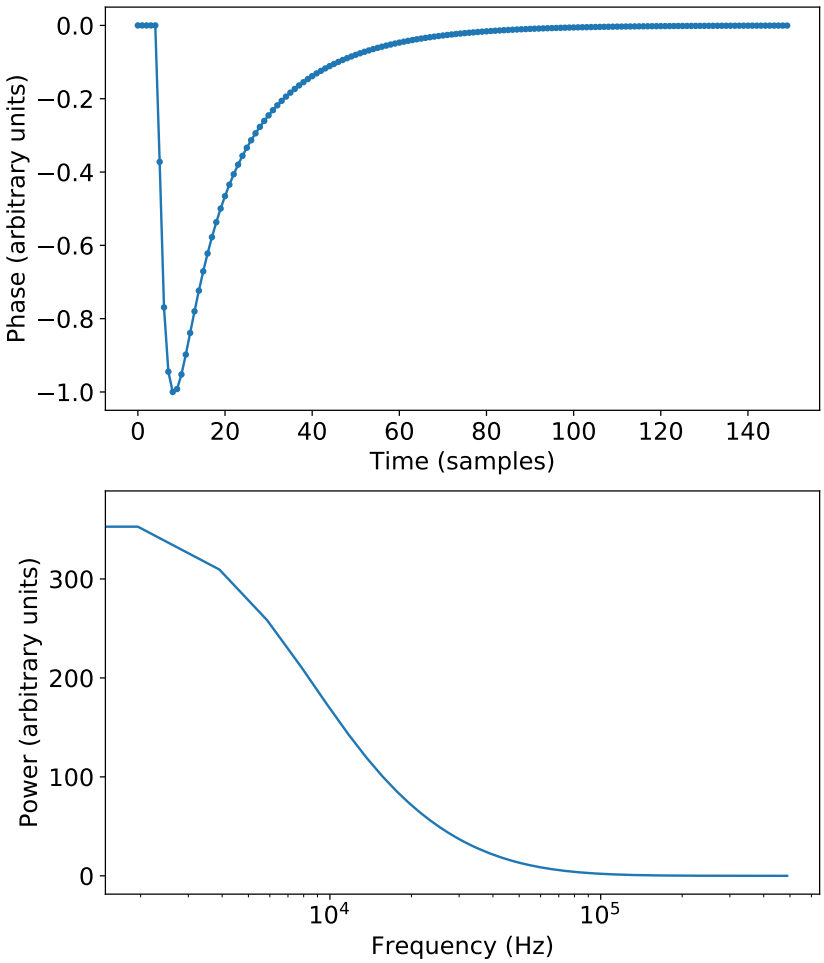


Figure 2.5: Top: representative phase pulse template, made by averaging together phase collected on a single resonator channel, then fitting to a triple exponential function. Bottom: power spectrum of the above pulse. Note that there is very little power above 100 kHz.

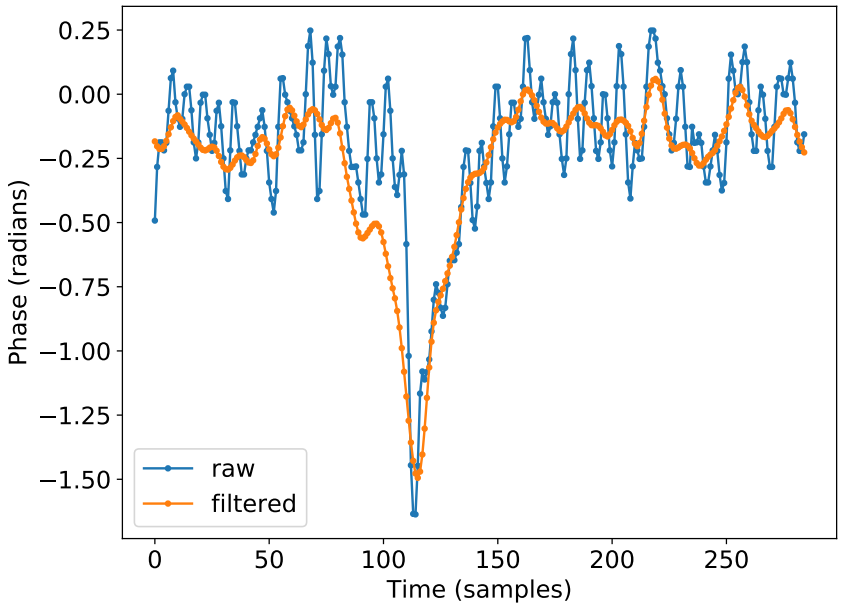


Figure 2.6: Example photon signal on a single resonator channel. (blue) is data collected after the 100 kHz LPF and phase conversion steps; (orange) is the same data stream after applying a 50 coefficient optimal FIR filter. Photon wavelength is 1050 nm. The sampling interval is 1.024 μs

matched to the characteristic pulse timescale ($\tau \approx 15\ \mu s$), as indicated by the filtered pulse being fairly symmetric (figure 2.6).

Photon Event Triggering and Streaming

The total phase data rate for a single readout unit is approximately 18 Gbps (1024 channels $\times$ 1 MHz per channel $\times$ 18 bits). This data rate is difficult to stream over ethernet (a 20 kpix array would require 360 Gbps of bandwidth) and infeasible to save for a full night of observing. To reduce the data rate, we use a firmware trigger to find and record photon pulses. The trigger consists of two conditions:

1. Local minimum detection - pulse must be preceded by 9/10 points w/ negative derivative, and immediately proceeded by two points w/ positive derivative.

2. Threshold - pulse amplitude must exceed a threshold value. This threshold is

programmable and can be specified individually for each channel. Typically, we use some multiple of the standard deviation of the phase value ($3.5 - 4\sigma$).

If the trigger conditions are met, the pulse amplitude, arrival time, and channel ID are recorded in a 64-bit data packet. Multiple such data packets are packed into UDP frames (up to 100 packets per frame) and streamed to the data server over ethernet. This is done using a dedicated 1 GBit ethernet interface connected directly to the FPGA, which is different from the Power PC ethernet interface used for firmware configuration and instrument control.

2.3 Performance

2.3.1 Phase Noise

A key system performance metric is the phase noise in each resonator channel (i.e. the noise PSD of the phase timestream used for photon detection). The phase noise has three major sources: 1) noise intrinsic to the device, 2) noise contributed by the cryogenic amplification chain, 3) noise contributed by the readout system [27]. Ideally, phase noise is dominated by sources (1) and (2); contributions from the readout system should be negligible. To characterize this, we measure the readout system phase noise in loopback (RF input connected directly to RF output), and compare this to the expected best case performance of the device and cryogenic amplifier.

The noise contributed by the device comes primarily in the form of a $1/f$ contribution due to two-level system (TLS) noise [41, 42]. We ignore this contribution, and instead use the cryogenic amplifier noise as a basis for comparison. This is because the TLS noise varies significantly between devices and is difficult to estimate, and using cryogenic amplifier noise alone provides a more stringent performance criterion than using the

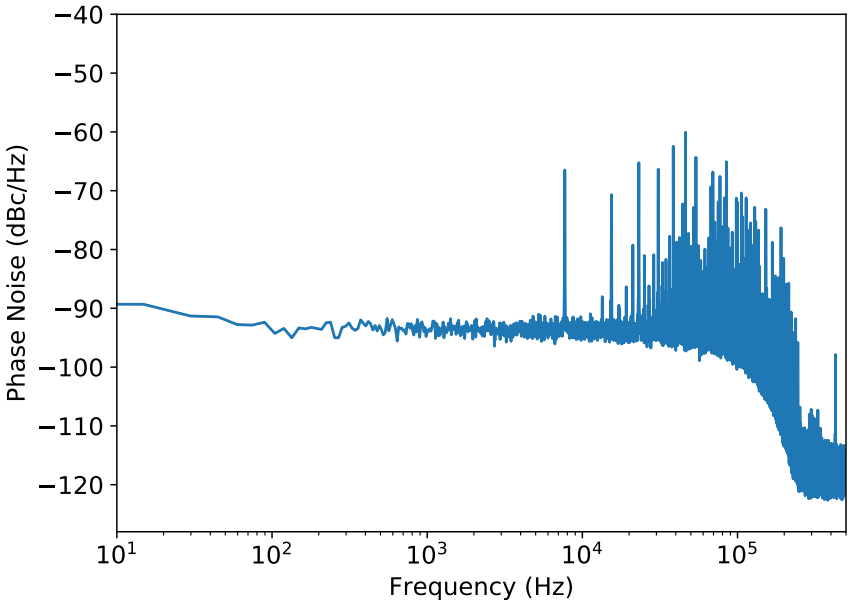


Figure 2.7: Phase noise spectrum of a single resonator channel. Measurement was taken in loopback with a full 1024 tone comb in $4-6\ GHz$ band. Aside from spurious signals, the noise floor is approximately -93 dBc/Hz.

combination of TLS noise and cryogenic amplifier noise. For the cryogenic amplifier we assume a 2.3 K HEMT (high electron mobility transistor), which is used in both MEC [21] and DARKNESS [20]. We assume that it has 40 dB gain and contributes thermal white noise at the device stage with a 2.3 K noise temperature (eqn. A.1). We report all phase noise measurements in dBc/Hz, where dBc is the noise power relative to the probe (carrier) tone in a single quadrature.

The noise contributed by the digital readout electronics is dominated by room temperature thermal noise and amplifier noise in the input RF chain. The exact phase noise contribution from the room temperature stage depends on the programmable attenuator settings; we calculate that for typical values the contribution of room temperature noise floor should be at least 6 dB lower than the HEMT noise contribution (see appendix A for calculation). The measured phase noise floor (figure 2.8) is $\approx$ 3 dB higher than expected, likely due to unaccounted for attenuation in the RF input chain.

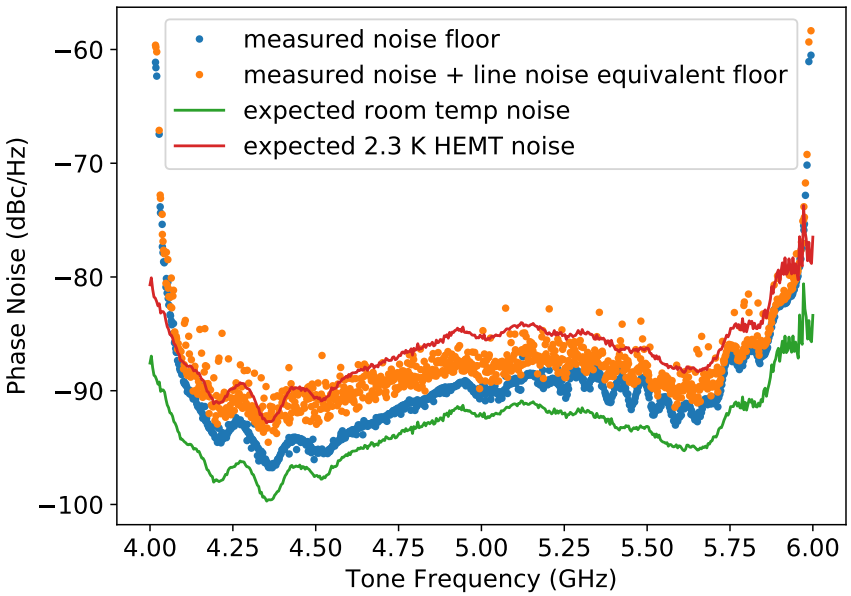


Figure 2.8: Loopback phase noise measurement for 1024 tone comb on a single readout unit. The measured phase noise value for each channel was computed by fitting the white noise component of the phase noise spectrum (example spectrum in figure 2.7). The equivalent phase noise floor after accounting for line noise is also plotted (calculation in appendix B). The expected phase noise contributions from the HEMT and room temperature chain were estimated using the measured tone power at the RF input (see appendix A for calculation).

To verify system performance, we performed a loopback measurement of the phase noise of each tone in a full 1024 tone frequency comb. The tones were given frequencies in 4 – 6 GHz range and uniform output power in the DAC LUT. The output power of each RF tone ranged from approximately -65 dBm to -70 dBm, where the variation results from the nonuniform frequency response of the RF output chain (which we calibrate out when reading out actual devices). The exact readout power (power at the MKID) that this corresponds to depends on the array properties and experimental setup, but the typical RF attenuation before the device is $\approx$ 35 dB, which corresponds to a readout power of approximately -100 dBm to -105 dBm. Results are plotted in figure 2.8, along with expected contributions from the room temperature amplifier chain and HEMTs.

The measured phase noise has significant line noise (figure 2.7). The line noise occurs at multiples of 7.629 kHz, which is the quantization frequency of the DAC comb. So, we suspect that the lines are generated as intermodulation products in the input RF chain. In order to compare the line noise to the HEMT and room temperature noise, we examine the effect each has on the variance of the measured photon pulse amplitude. We do this by removing the spectral lines from the noise PSD, then scaling this "flattened" PSD such that its corresponding pulse height estimator variance is equal to that of the original PSD (see appendix B for full calculation).

There is a significant apparent difference between our phase noise measurements (≈ -93 dBc/Hz), and the measured phase noise for the first generation system (-106 dBc/Hz, including HEMT noise [25]). This is because the first generation system measurements were performed at -85 dBm [25] (significantly higher than used in practice with ARCONS), and our measurements were performed at an expected readout power of -100 dBm to -105 dBm (which is close to the -106 dBm readout power [27] of the PtSi devices used in DARKNESS [20] and MEC [21]). So, we expect that any thermal noise source (which includes both HEMT and room temperature amplifier noise) will

contribute up to 20 dB higher phase noise (phase noise is inversely proportional to tone power; see eqns. A.1, A.2, A.7) for readout tone powers characteristic of our system as compared to the first generation system; this is reflected in our measurements in figures 2.7 and 2.8.

It is difficult to determine the effect of readout system noise on the ultimate resolving power of the MKIDs, as this depends on detector sensitivity (i.e. pulse height as a function of photon energy), quality factor, and device noise, which vary significantly between pixels [21, 17]. However, the majority of channels are below the expected HEMT noise floor, so these would obey the HEMT noise limits measured in (Ref. [27]). We also note that the measurements in figures 2.7 and 2.8 are close to worst-case performance, with RF input attenuators $A_1 = 14.25$ dB and $A_2 = 14.5$ dB (see figure A.1 for block diagram of RF input chain). A more typical use case would have $A_{1,2} < 10$ dB, which will reduce the phase noise floor by > 2.7 dB (eqn. A.8), making almost every channel HEMT noise limited.

2.4 Sideband Suppression

Imperfections in IQ mixer performance lead to the presence of "sidebands", or reflections across the LO frequency when the mixer is used to upconvert or downconvert RF tones. For example, a 6.5 GHz tone generated using a 6 GHz LO will have a sideband tone at 5.5 GHz (which may also show up at -500 MHz in the IF band after downconversion). Ideally, these sidebands are > 30 dB down from the original tone. In our system, we have observed sidebands with powers as high as $10 - 15$ dB below the original tone. A sideband is problematic if it happens to be within 100 kHz of another tone, as it will leak into that channel and cause its phase timestream to oscillate at $f = f_{tone} - f_{sideband}$, degrading the performance of that pixel.

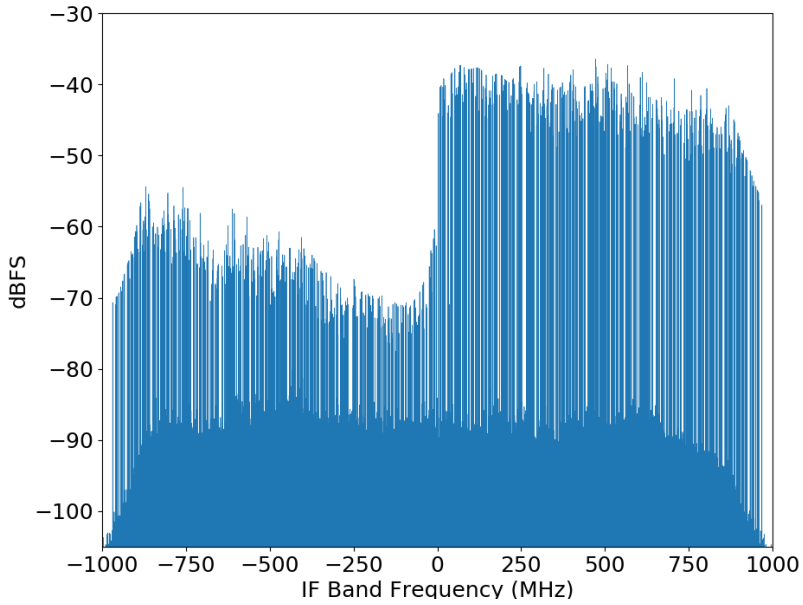


Figure 2.9: Example sideband spectrum (taken using the ADC on loopback). Tones on the right half ($f > 0$) are actual tones generated by the DAC, while tones on the left half are undesirable sideband reflections of these generated tones. In the worst case, sideband power is only 10-15 dB down from the tone power.

The input to the I/Q mixer can be adjusted to compensate for mixer errors and suppress sidebands. This can be done for both upconversion and downconversion:

- Upconversion: adjust the relative phase/amplitude of the I and Q inputs to the mixer. This effectively introduces sidebands in the IF input to cancel them out in the output.

- Downconversion: introduce real RF sidebands to cancel out sidebands produced in the downconverted I and Q.

Performing these adjustments separately for upconversion and downconversion is infeasible, as injecting RF sidebands after upconversion would require major design changes. So, we instead adjust the output I and Q generated by the DAC to minimize total sideband power remaining after both upconversion and downconversion.

For our system, we experimentally determined that the optimal phase and amplitude

adjustments are sensitive to tone power and frequency and are not always smooth functions of these. So, a global fitting approach would be suboptimal, and any adjustments must be made on a per-tone basis in the DAC LUT. We have developed an algorithm to perform this optimization for an arbitrary list of tone frequencies and powers. Our algorithm minimizes total sideband power remaining after both RF upconversion and downconversion (and everything else that might be in the RF chain, including resonators, etc). The optimization problem for N tones can be written as:

$$\max_{\Delta\phi_1, r_{IQ,1} \ldots \Delta\phi_N, r_{IQ,N}} \sum_i P_i - P_{sideband,i}(\Delta\phi_i, r_{IQ,i}) \tag{2.1}$$

where i indexes the tones, P_i is the tone power, and $P_{sideband,i}(\Delta\phi_i, r_{IQ,i})$ is the sideband power of tone i as a function of the phase offset $\Delta\phi_{IQ,i}$ (from 90°) between I_i and Q_i and $r_{IQ,i} = |I_i/Q_i|$ (I_i and Q_i are the DAC outputs for tone i). Sideband power is measured in the input IF band by the ADCs, in units dBFS (dB relative to ADC full scale). To simplify the problem, we have assumed that the tones are independent (i.e. adjusting the phase or amplitude balance of one tone will not affect the sideband power of any other tone); so the problem can be thought of as a series of N independent maximizations over the two dimensional domain given by $\{\Delta\phi_{IQ} \times r_{IQ}\}$.

The following procedure is used to evaluate the objective function for a specific set of $\{(\Delta\phi_{IQ,1}, r_{IQ,1}), (\Delta\phi_{IQ,2}, r_{IQ,2}), \ldots, (\Delta\phi_{IQ,N}, r_{IQ,N})\}$:

1. Load DAC LUT containing these offsets
2. Take a 8,388,608-point snapshot (set by the amount of available memory) of the ADC input
3. Take an FFT of the snapshot to find sideband powers

This evaluation process takes approximately one minute, so a brute force optimiza-

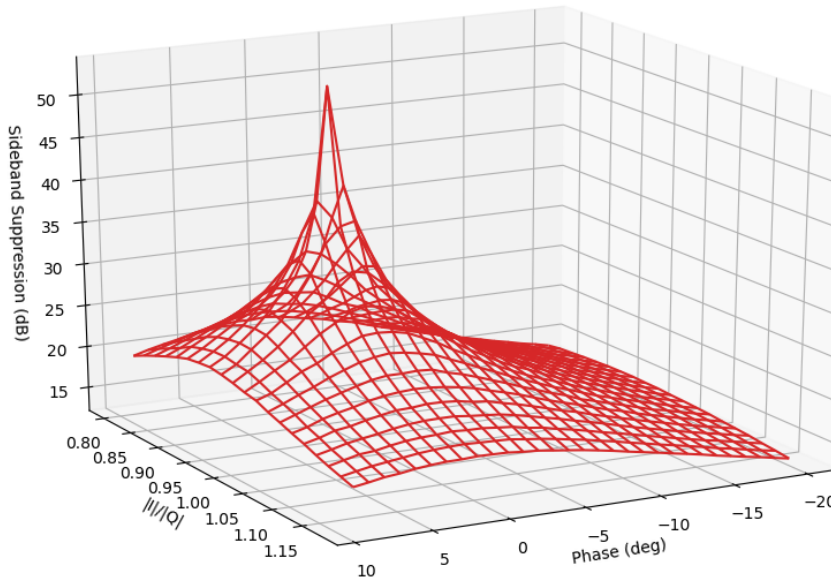


Figure 2.10: Plot of sideband suppression of a single tone as a function of ϕ and r_{IQ}. Close to the optimal value, the sideband suppression decays approximately exponentially, which is why an exponential fit is used in the search algorithm.

tion with the required range and precision (phase: $[-20°, 20°]$, $1°$ increments; amplitude ratio: $[0.8, 1.2]$ in increments of 0.2) would take approximately 13 hours. Since the optimization must be repeated for each new configuration of tones, the brute force approach is infeasible and a more efficient algorithm is needed. We have developed the following approach:

1. Sample the objective function at n randomly chosen sets of $\{(\Delta\phi_{IQ,1}, r_{IQ,1}), (\Delta\phi_{IQ,2}, r_{IQ,2}), \ldots, (\Delta\phi_{IQ,N}, r_{IQ,N})\}$ (in addition to $(\Delta\phi_{IQ,i}, r_{IQ,i}) = (0, 0)$ for all i)

2. For each tone, fit $P_i - P_{sideband,i}(\Delta\phi_i, r_{IQ,i})$ to an exponential decay function (see figure 2.10).

3. Randomly sample a new set of $\{(\Delta\phi_{IQ,1}, r_{IQ,1}), (\Delta\phi_{IQ,2}, r_{IQ,2}), \ldots, (\Delta\phi_{IQ,N}, r_{IQ,N})\}$, either close to the fit centers or uniformly over the domain (choose this with some probability ϵ).

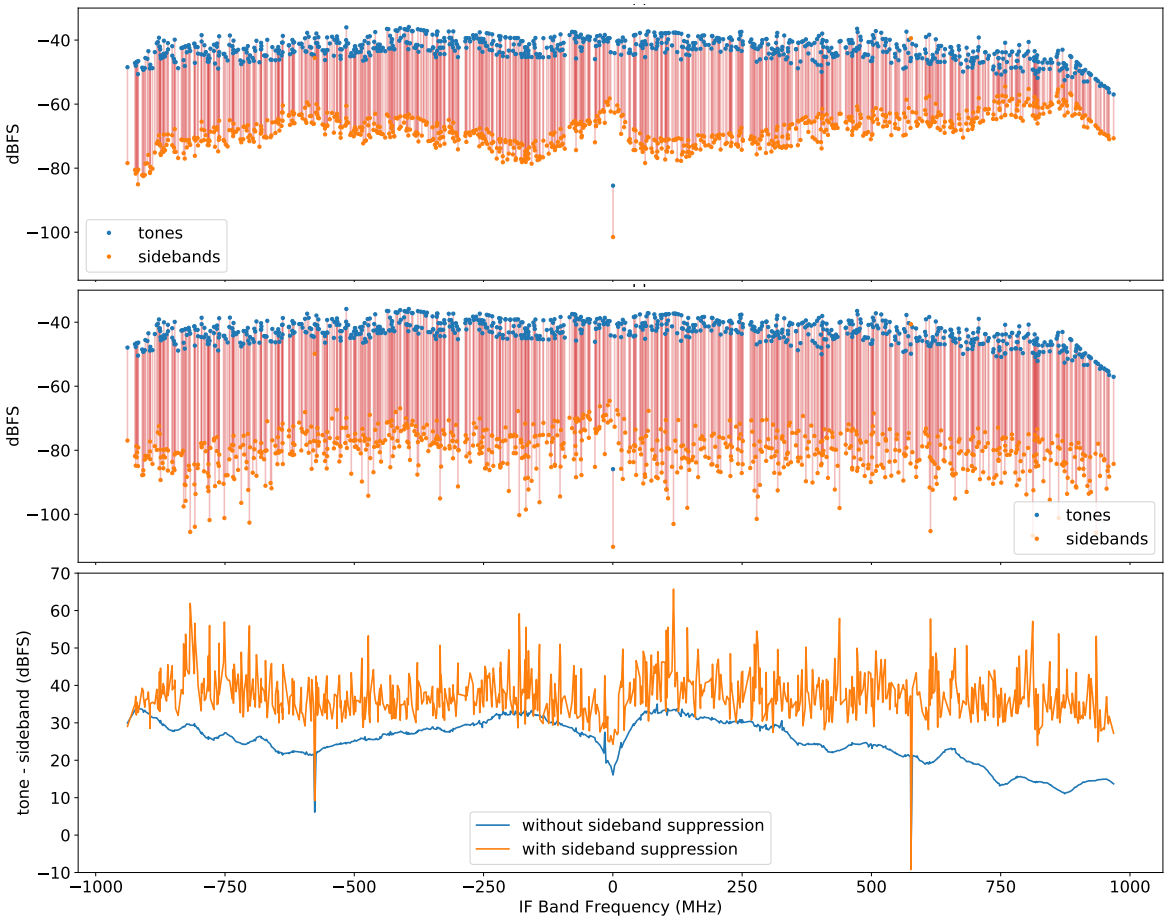


Figure 2.11: Test of the sideband suppression algorithm on loopback using a 868 tone MEC array frequency comb. Top/middle: comparison of each tone's power with its corresponding sidedband power, before (top) and after (middle) running the sideband suppression algorithm. Bottom: Magnitude of sideband suppression across the frequency comb.

4. Repeat (2) and (3) until all sidebands are sufficiently suppressed (typically $P_{tone} - P_{sideband} > 30$ dB), or the number of iterations has reached some threshold value.

Using this approach, we are able to suppress sidebands by at least 30 dB for almost every tone, with a convergence time of 30 - 45 minutes. Results from a loopback test using a MEC array frequency comb with 868 tones are shown in figure 2.11. It is difficult to provide a general estimate of the improvements to energy resolution resulting from suppressing sidebands to ≥ 30 dBc, as this depends on highly variable device characteristics (including sensitivity, quality factor, device noise), as well as the location of of the interfering sideband in the spectrum of a nearby resonator channel. We feel that the 30 dBc benchmark is appropriate, as it has an equivalent RMS phase > 3 dB below that of a typical resonator channel (appendix C), and is the same benchmark used for the first generation system [25].

2.5 Conclusion

We have developed a digital readout system which, for the first time, makes it feasible to read out kilopixel-scale photon counting optical/IR MKID arrays. Our system is actively being used by two MKID cameras; MEC at Subaru Observatory, and DARKNESS at Palomar Observatory. All of our array calibration and control software, as well as the Virtex-6 firmware, is open source and can be found at: `https://github.com/MazinLab/MKIDReadout`.

Chapter 3

Deep Learning for Array Calibration

3.1 Resonator Identification and Tuning: Baseline Methodology

The standard procedure for MKID array calibration treats frequency identification and drive power tuning independently. We outline this procedure below, then explain some of the issues that require manual tuning to address.

All calibration steps are performed using measurements of the complex frequency response $S_{21}(f) = I(f) + iQ(f)$ of the MKID array in the resonator frequency band (approximately 4-8 GHz for the current generation of instruments). When $S_{21}(f)$ is plotted on the complex plane, resonators manifest as "loops" when driven at the correct power (figure 3.1). Resonators absorb power close to resonance, so they also correspond to local minima in $|S_{21}|$. Frequency response data is taken over a wide range of drive powers for the tuning stage.

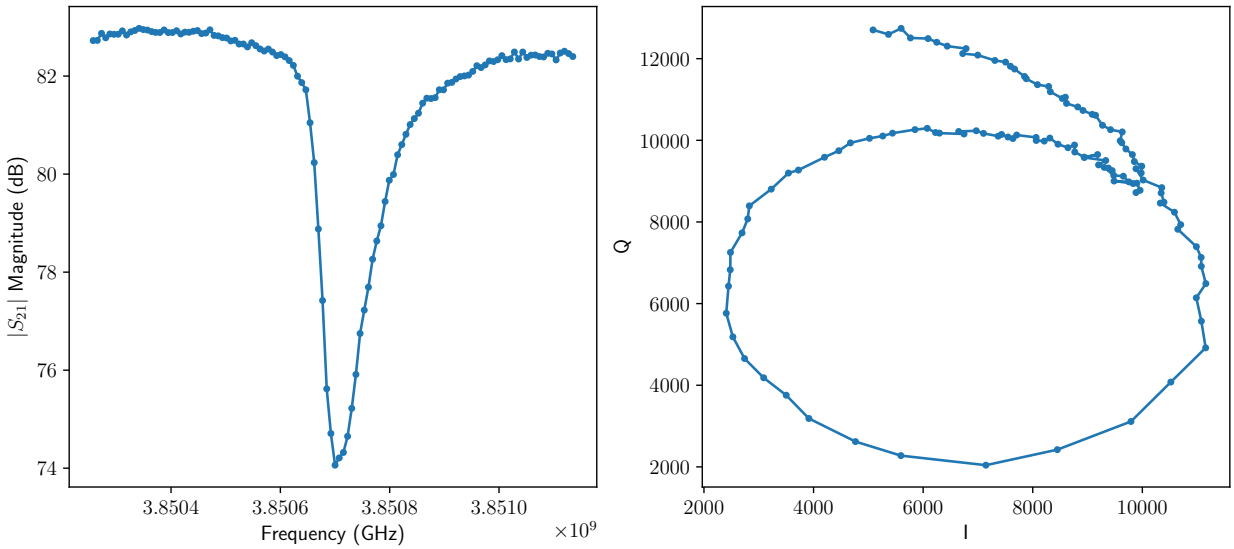


Figure 3.1: Resonators absorb power in a $\approx$ 200 kHz band around the resonant frequency (hence a dip in $|S_{21}|$; left), but leave tones outside this band relatively unmodified. If $S_{21}(f)$ is plotted in the complex plane over this frequency band, the resonator will trace out a "loop" (right).

3.1.1 Resonator Identification

Due to imperfections in the MKID array fabrication process, there is an $\approx$50 MHz spread between the nominal resonator design frequency and the actual measured frequency. Since pixels are designed to be separated by $\approx$2 MHz, this spread makes the nominal design frequency essentially useless for identifying resonators. Instead, we measure the magnitude of the frequency response ($|S_{21}|$) across the full 4-8 GHz bandwidth of each microwave feedline ("widesweep"), where resonators appear as narrow 5-20 dB deep "dips" in transmission (figure 3.1). These dips can be accurately flagged automatically using a spatial bandpass filter (to select for the band corresponding to resonator width in the $|S_{21}(f)|$ signal) followed by a peak-detection algorithm such as SciPy's signal.find_peaks function [43].

3.1.2 Resonator Drive Power Tuning

In order to maximize photon detection SNR, resonators should be driven at the highest possible power, as amplifier induced phase noise decreases linearly with drive power [27, 35]. However, if a resonator is driven too hard, it can enter a bistable "bifurcation" state, rendering it useless for photon detection [26]. So, we take the "ideal" resonator drive power to be slightly (1-3 dB) below the bifurcation state. For most resonators, the bifurcation state can easily be identified in the complex frequency response of the resonator across its linewidth ($\approx$ 200 kHz) (see figure 3.2), making it relatively straightforward for a human operator (or machine learning algorithm) to identify the correct drive power.

To tune each resonator, we select a 450-750 kHz window around the frequency dip identified in the previous step, and measure the complex (I and Q) frequency response in this window at a variety of drive powers (typically a 31 dB range in steps of 1 dB). We then feed this data into a neural net classifier that estimates the drive power [28].

To refine the estimate of optimal resonator drive frequency at a given power, we define the quantity IQ-velocity (IQV), which is the magnitude of the derivative of S_{21} in the complex plane wrt frequency:

$$IQV(f_i) = |\Delta_f S_{21}(f_i)| = \sqrt{[I(f_{i+1}) - I(f_i)]^2 + [Q(f_{i+1}) - Q(f_i)]^2} \tag{3.1}$$

where i indexes frequency, and all S_{21} measurements are performed at a single power. The point along a resonator loop with maximum IQV is the ideal drive frequency, since it is where the phase response of the resonator is most sensitive to changes in frequency. This is often different from the $|S_{21}(f)|$ minimum (figure 3.3). So, after the optimal drive power is estimated, the frequency identified in the previous stage (section 3.1.1) is changed to the IQV maximum at the estimated power. This is typically done using a

peak finding algorithm in conjuction with a boxcar smoothing filter.

After automated power and frequency determination, a human operator may look through the data set to refine the neural network's initial guesses.

If two resonators are separated by less than 200 kHz, we only read out one of them to prevent their probe tones from interfering [35]. We choose the higher frequency resonator to prevent undesirable resonator crosstalk during photon detection events [1]. After the identification and tuning process, the final resonator list is pruned according to this criteria; we define the resulting list to be the set of "usable resonators".

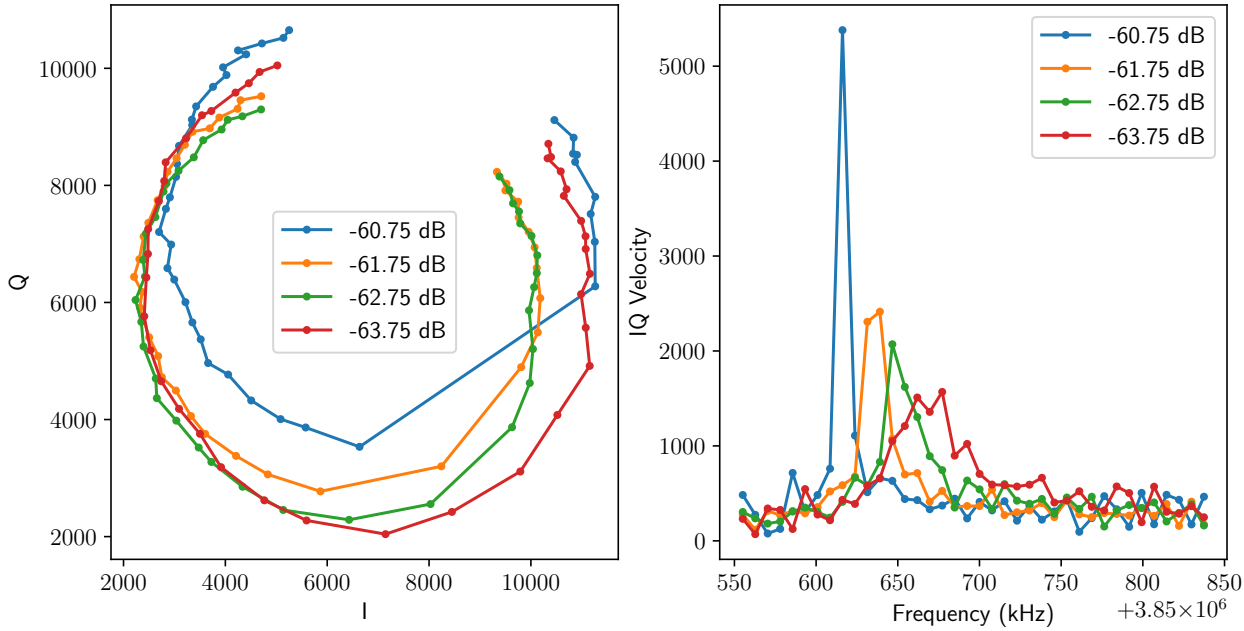


Figure 3.2: Complex frequency response $S_{21}(f)$ (left) and corresponding IQV (right) of a single resonator measured at four different powers. The resonator is clearly bifurcated when driven at -60.75 dB, as indicated by the discontinuity in the resonator loop. There is still some apparent discontinuity at -61.75 dB; -62.75 dB or -63.75 dB would be reasonable drive powers for this resonator. Power is measured relative to the DAC full scale output.

[1] Since the resonant frequency decreases during detection events, the higher frequency resonator might enter the readout band of the lower frequency resonator, interfering with that resonator's probe tone.

3.1.3 Problems with the Baseline Methodology

We find that the automated steps of the above method are inadequate for providing an accurate calibration solution; 15-20% of resonators may be miscalibrated. Most of these calibration errors are related to choosing a frequency window around each identified resonator (section 3.1.1) to feed into the neural network power classifier. We detail these issues below, using an analysis of a single representative feedline from the MEC array "Hypatia" (call this feedline "Hypatia6").

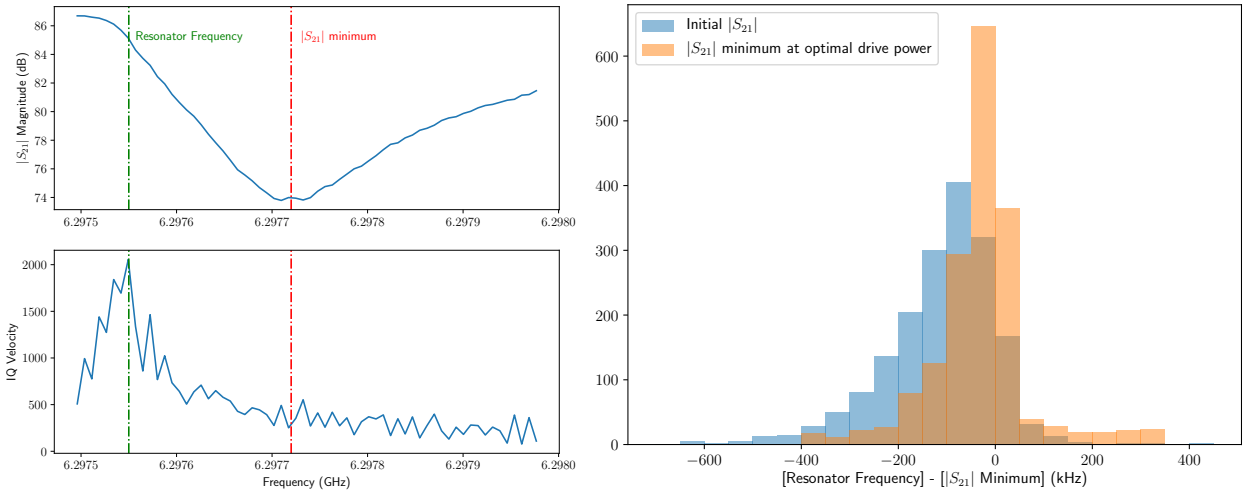
Choosing a frequency window around each resonator is complicated by the following factors: 1) Resonators often move significantly in frequency space as a function of power (figure 3.2); and 2) The local minimum in $|S_{21}|$ that is used to identify the resonator is often not aligned with the IQV maximum; this discrepancy can be larger than the linewidth of the resonator (figure 3.3).

Issue (1) can be dealt with to some extent by re-centering the window around the $|S_{21}|$ dip at each power. However, this is complicated by issue (2), since the $|S_{21}|$ dip doesn't always shift by the same amount as the actual resonator frequency.

Because of issue (2), we find that windows must be quite wide; e.g. for the Hypatia6 feedline, a 500 kHz window will only include ≈90% of the resonators [2]. Using larger windows however increases the chance that there will be multiple resonators within the same window; which degrades the capability of the machine learning algorithm to tune the resonator (figure 3.4).

We find that while the neural network power classifier generally has good performance on standalone resonators, manual intervention is required when there are multiple resonators within the same window, or the resonator is too far from the frequency identified in the previous stage and lies outside the window. For the Hypatia6 feedline, this com-

[2]The median resonator linewidth is 140 kHz, so to be included in a 500 kHz window, we assume that the resonator frequency is ±180 kHz from the window center.



(a) (b)

Figure 3.3: Illustration of issue (2). a) Plot of $|S_{21}|$ and IQ velocity for a single resonator, measured at its optimal drive power. For this resonator, the $|S_{21}|$ minimum and IQ velocity maximum (taken to be the "actual" resonator frequency) are separated by 170 kHz. b) Histograms of this discrepancy across Hypatia6. Initial $|S_{21}|$ (blue) shows the difference between the actual resonator frequency and that flagged in the identification step. Some of this is due to the resonator frequency changing with power, and can be corrected for to some extent by re-centering the machine learning window accordingly (orange).

prises 15-20% of resonators. It is difficult to determine a priori which resonators meet these criteria (particularly when the resonator lies outside the initial window), so manual inspection is required for all resonators in order to fully optimize array performance. This is a time consuming process, requiring 4-6 hours per feedline. To address these issues, we present an approach for doing both frequency identification and tuning in a single step using the same convolutional neural network.

[3]This histogram includes only resonators satisfying the "usable resonator" criterion defined in section 3.1.2. However, we allow resonators that violate this criterion to count as nearest neighbors, since they may still interfere with the tuning process even if they aren't in the final resonator list.

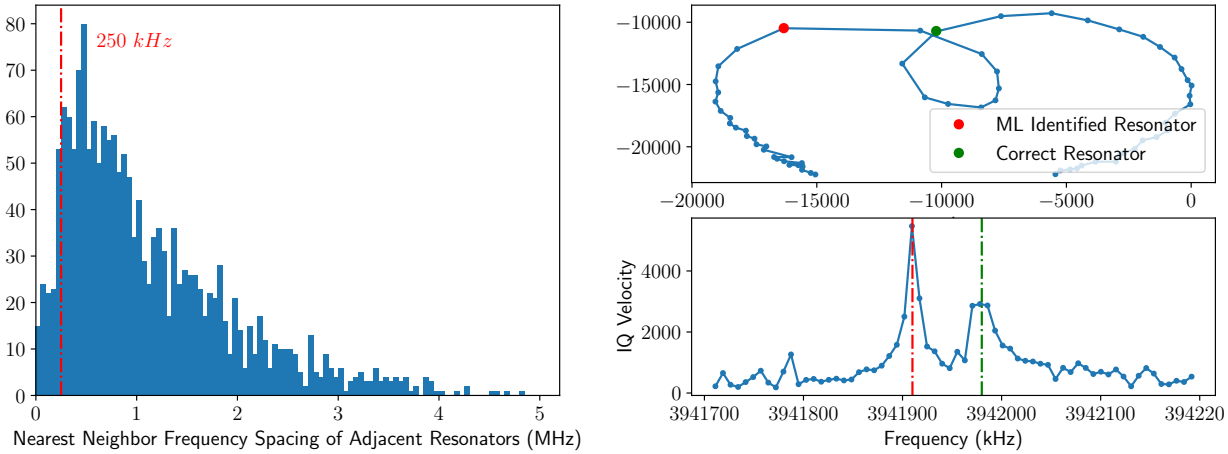


Figure 3.4: left: histogram of the frequency spacing between resonator and it's nearest neighbor in frequency space for Hypatia6 [3]. Approximately 8% of resonators contain another resonator within a 500 kHz window; right: example of a window with multiple resonators. In this case, the machine learning algorithm selected the correct power for the higher frequency resonator (which was also the one flagged during the identification stage), but selected the frequency of the lower frequency resonator (which is bifurcated at this power).

3.2 End-to-end Machine Learning: System Overview

3.2.1 Overall Architecture

The full S_{21} dataset can be conceptualized as a 2D "image", with a real (I) and imaginary (Q) value at each point in $power \times freq$ space. Each resonator at its optimal drive power and resonant frequency is located at a single point in this image, and its influence extends to a small region surrounding this point. Resonator identification/tuning can then be posed as an object detection problem in $power \times freq$ space.

One of the simplest methods of implementing machine learning based object detection is to use a sliding window classifier. In this scheme, we take a small window around each point in the $S_{21}(p, f)$ image and feed it into a deep convolutional neural network (CNN) [44], which assigns the window a set of scores representing the likelihood of it being a member of each of the following classes:

1. window is centered around a resonator at the correct power and frequency
2. window is centered around a bifurcated (saturated) resonator
3. window is centered around an underpowered resonator
4. there is no resonator inside the window

The output of the classification stage is four $N_{power} \times N_{freq}$ images, each containing the classification score at each point for one of the above classes. To find resonators, we use a relatively simple criteria to select N local maxima above some threshold of the image representing the scores for class (1) (detailed in section 3.2.3).

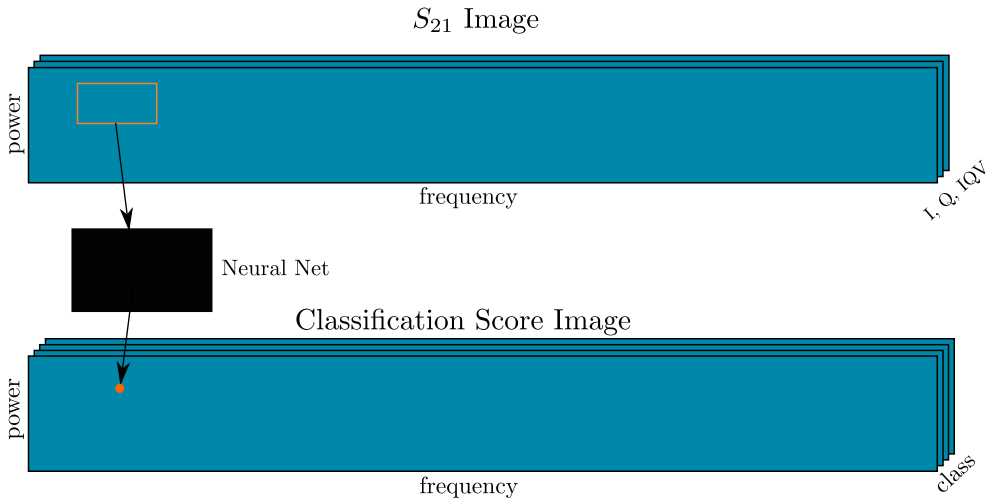


Figure 3.5: Schematic of overall architecture. The input data set is a 3-color image (one for each of I, Q, and IQV) containing the frequency response of the MKID feedline at multiple powers. A small window is generated around each (p, f) point, and fed into a convolutional neural network to generate a set of classification scores for that point. This results in a $N_{power} \times N_{freq}$ output image with four colors, one for each classification score.

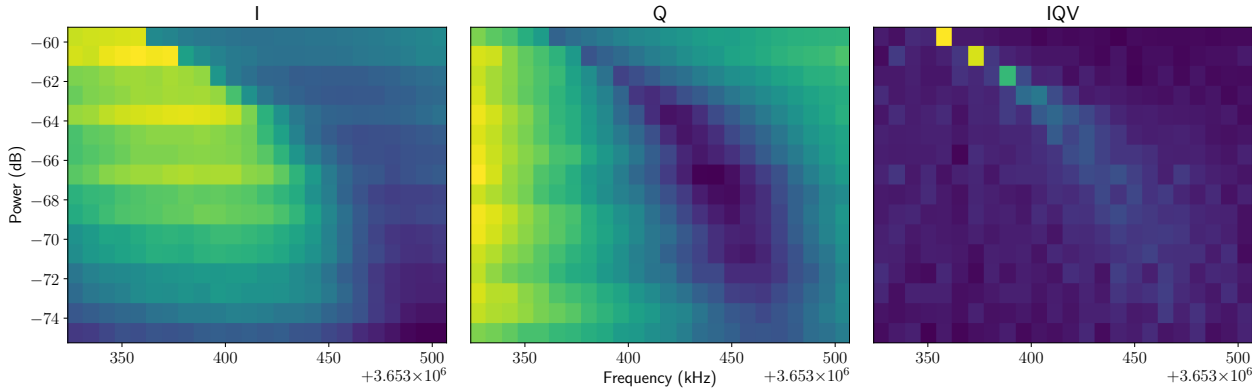


Figure 3.6: Example S_{21} images of I, Q, and IQV around a resonator. This is before any normalization or other preprocessing, so units/scaling is arbitrary. The classification score image assigned to this input data is plotted in figure 3.8.

3.2.2 Neural Network

Preprocessing

A typical S_{21} dataset will contain the full 4-8 GHz band, sampled at 7.7 kHz, and 31 powers sampled in 1 dB increments. For the sliding window classifier, we use a window size of 30 points in frequency ($\approx$ 230 kHz) by 7 points in power. In frequency space, this provides good coverage across the full linewidth of most resonators, and in power space, it allows the transition from saturation $\rightarrow$ underpowered to be fully sampled. In addition to the S_{21} I and Q data, we include IQV as well since it is a useful visual aid when manually inspecting resonators. I, Q, and IQV are implemented as separate "color channels" in the neural network input [4].

We perform the following normalizations on each input window:

1. Normalize by S_{21} magnitude, independently at each power:

   - $M_p = \sqrt{\frac{1}{N_{freq}} \sum_i I_{p,i}^2 + Q_{p,i}^2}$, where p indexes power and i indexes frequency. N_{freq} is the window size along the frequency axis.

[4] "color channels" can be thought of as analogous to RGB channels in conventional images

- $I_p \rightarrow I_p/M_p$; $Q_p \rightarrow Q_p/M_p$; $IQV_p \rightarrow IQV_p/M_p$;

  The value of M_p depends on how the MKID readout ADCs and DACs were configured for the frequency sweep at that particular power p; this can change arbitrarily with power and has no physical relevance. Normalizing separately at each p removes this nonuniformity.

2. Center the data, also independently at each power: $I_p \rightarrow I_p - mean_i(I_p)$, $Q_p \rightarrow Q_p - mean_i(Q_p)$, $IQV_p \rightarrow IQV_p - mean_i(IQV_p)$

Normalizing by M_p before centering ensures that the relative loop size (i.e. the amount it attenuates the probe tone on resonance) is preserved; this is an important metric of resonator quality [5]. If a window overlaps with the boundaries of the S_{21} image, it is edge-padded to the correct size.

Network Architecture

The convolutional neural network is implemented using the TensorFlow [45] package version 1.8. The network has three convolutional/pooling layers followed by a fully connected output layer (figure 3.7). Each convolutional/pooling layer has three stages: 1) 2D convolution (generalized across input/output color channels) with filter coefficients determined during training; 2) Rectilinear activation function, which applies $RELU(x) = max(0, x)$ to each pixel in the output of the convolution[46]; 3) max-pooling, which downsamples the output of the previous stage by selecting the maximum value within a $n \times m$ window. The fully connected layer multiplies the flattened output of the final convolutional/pooling layer by a matrix of trainable weights to arrive at the final four element output vector. A softmax function is applied to this output to normalize the

[5] loop size is proportional to Q/Q_c, where Q is the resonator quality factor and Q_c is the coupling quality factor; ideal resonators have $Q \approx Q_c$, hence a relative loop size close to 1

vector of classification scores. During training, batch normalization is applied to the output of each convolution to improve performance/convergence time [47], and dropout is used to minimize overfitting [48].

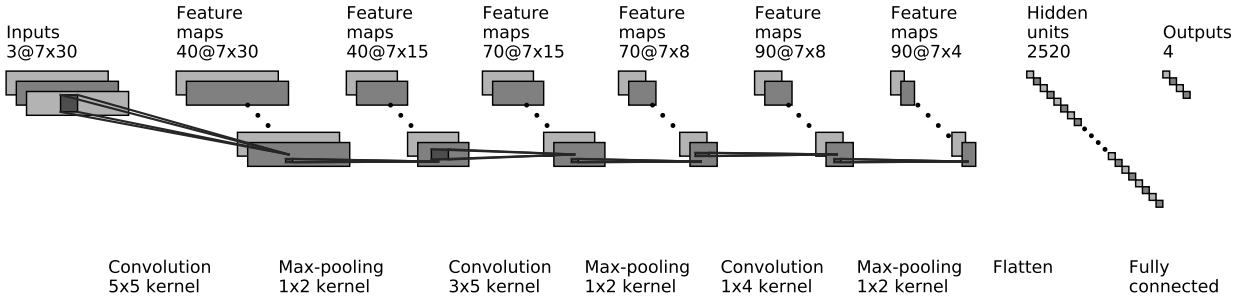


Figure 3.7: Schematic of neural network architecture, showing alternating convolutional/pooling layers, followed by a fully connected output layer. The row axis in each layer is along the power dimension, and the column axis is frequency. Batch normalization and a RELU activation function are applied to the output of each convolution. During training, dropout is applied to the output of each max-pooling layer. Each of the four outputs represents the classification score for each of the four possible classes. This figure was generated by adapting the code from `https://github.com/gwding/draw_convnet`.

Training

The network is trained using human-reviewed resonator data from the "baseline methodology" analysis pipeline described in the previous section. Sixteen training images are made for each identified resonator (four per class):

1. Four copies centered around correct power/frequency ("good" class)
2. Four "bifurcated" resonators, from powers $[p_c + p_{sat}, p_c + p_{sat} + 3]$
3. Four "underpowered" resonators, from powers $[p_c - p_{up}, p_c - p_{up} - 3]$
4. Four "no resonator" images randomly selected from regions of S_{21} without resonators

All of the above powers are in dB; p_c is the correct power, and p_{sat} and p_{up} represent bifurcation/underpowered thresholds above/below the correct power, respectively. These thresholds are arbitrary and can be tuned as hyperparameters. For the bifurcated training images, a random offset within $[0, -75 \text{ kHz} \times \frac{f_{res}}{4 \text{ GHz}}]$ is added to the resonant frequency; this was found to correct for resonant frequency movement with power, and prevent the network from choosing frequencies away from the IQV peak during inference.

Training a neural network is the process of finding the set of convolutional filter coefficients and fully connected layer weights that minimize a "loss function" that is computed across all of the training data. We use a "cross-entropy" loss function given by: $L = -\sum_i \sum_c y_{i,c} log(p_{i,c})$, where i indexes the training samples and c indexes classes. $y_{i,c}$ represents the "true" class label for sample i; $y_{i,c} = 1$ for the correct class c and 0 for all other c. $p_{i,c}$ is the score assigned by the neural network for image i and class c. Because the final layer of the neural network is a softmax function, $0 < p_{i,c} < 1$. For the training process, we use the Adam optimization algorithm, which is a modified version of stochastic gradient descent [49], with a batch size of 50 training images and the learning rate left as a tunable hyperparameter.

3.2.3 Resonator Identification

After running the full S_{21} inference dataset through the neural network, we obtain a four-color $N_{power} \times N_{freq}$ output image containing the classification scores for each data point in the input image (figure 3.5). For the purposes of inference, we only use the first "color" of the output image, which corresponds to the likelihood of a resonator at the correct drive power and frequency being centered on a given point (p, f). Correctly driven resonators correspond to local maxima in this output layer above a certain threshold (figure 3.8). We find that this threshold varies between datasets, so it is generally

sufficient to pick the N_{res} highest local maxima, where N_{res} is the expected number of resonators. In practice, we use both of these criteria to limit false positive detections (i.e. we pick the N_{res} highest local maxima but require them to be above a certain threshold). For a typical 4-6 GHz sweep, we use $N_{res} = 1024$, since this is the maximum number of pixels supported by the readout system over this bandwidth [35]. It is also very close to the target pixel count for both DARKNESS [20] ($N = 1000$) and MEC [21] ($N = 1022$).

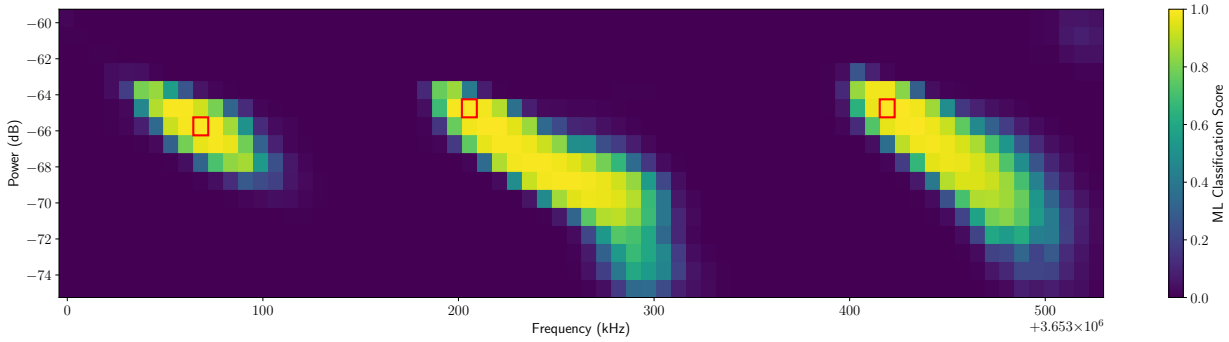


Figure 3.8: Output classification image for the "good resonator" class over a 500 kHz band. Each of the three local maxima (outlined in red) corresponds to a resonator. The input data corresponding to the rightmost resonator is plotted in figure 3.6.

One issue with this approach is that if a resonator's frequency changes a significant amount with power, the resonator may show up as multiple distinct local maxima. One way to mitigate this is to use a slightly modified version of the notion of topographic peak prominence [6] : for all adjacent (in frequency space) local maxima meeting the initial threshold or N_{res} cut, we take the minimum value along the shallowest monotonic path between the two maxima (see figure 3.9). If this value exceeds a certain threshold (t_{pr}; see table 3.1), we flag the larger of the two maxima as a resonator and discard the other one.

[6]Peak prominence is a measure of how "well-separated" a peak is from its neighbors; two neighboring resonators are more likely to manifest as distinctly separated peaks in the output classification than a single resonator moving in frequency space.

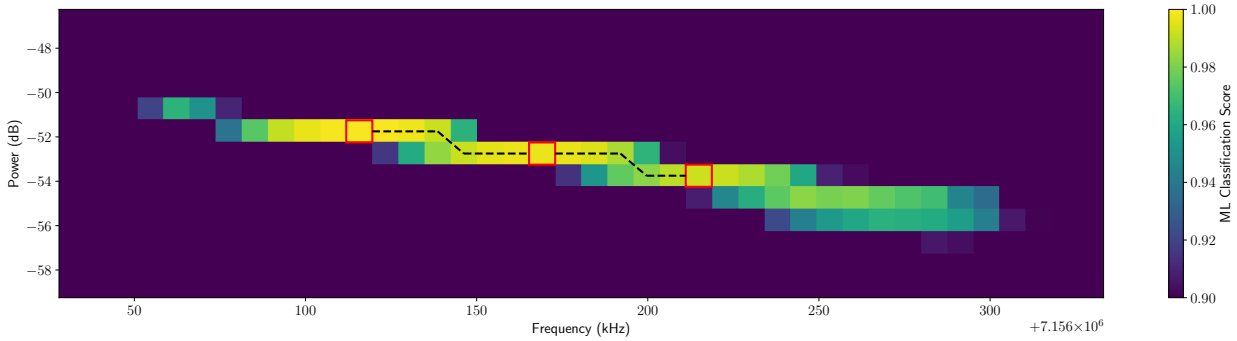


Figure 3.9: Output classification image for the "good resonator" class for a single resonator. The resonator frequency changes significantly enough with power that the resonator generates three distinct local maxima with high classification scores (> 0.99), outlined in red. The shallowest paths between these maxima are shown by the dashed lines. The minimum value along each of these paths is very high (> 0.98), so these maxima are not "prominent" enough to indicate the presence of three separate resonators. So, the algorithm will only flag the highest scoring peak as a resonator and discard the other two.

3.3 Final Model Selection and Training

To develop, tune, and train our model, we used a dataset consisting of five feedlines from the device "Hypatia" (8490 resonators), which is the device currently mounted in the MEC instrument. For each feedline, the dataset spanned 4 GHz of bandwidth in steps of 7.629 kHz, and 31 dB of power in steps of 1 dB. Resonators were identified and tuned using the baseline methodology (section 3.1), and after the automated steps, both resonant frequency and drive power were corrected by a manual operator. All hyperparameter and model architecture tuning was performed within this dataset, with all hyperparameters being manually selected (i.e. no grid search or similar algorithms were used). Once we settled on a model architecture and hyperparameter set, we used all five feedlines to train the final model for evaluation.

During training, 10% of the training data is held out for testing; accuracy and cross-entropy are measured throughout the training process. We do this to ensure that the neural network isn't overfitting to the training examples. After the last training step,

our model has approximately 97% accuracy on on both the training data and held-out test data (figure 3.10), indicating good performance with minimal overfitting [7].

Preprocessing		Training		Identification	
p_{sat}	1 dB	learning rate	$10^{-4.5}$	threshold score	0.9
p_{up}	4 dB	training epochs	170	N_{res}	1024
		batch size	50	t_{pr}	0.85
		dropout probability	0.4		

Table 3.1: Table of model hyperparameters used during data preprocessing, training, and identification. Neural network hyperparameters can be found in figure 3.7.

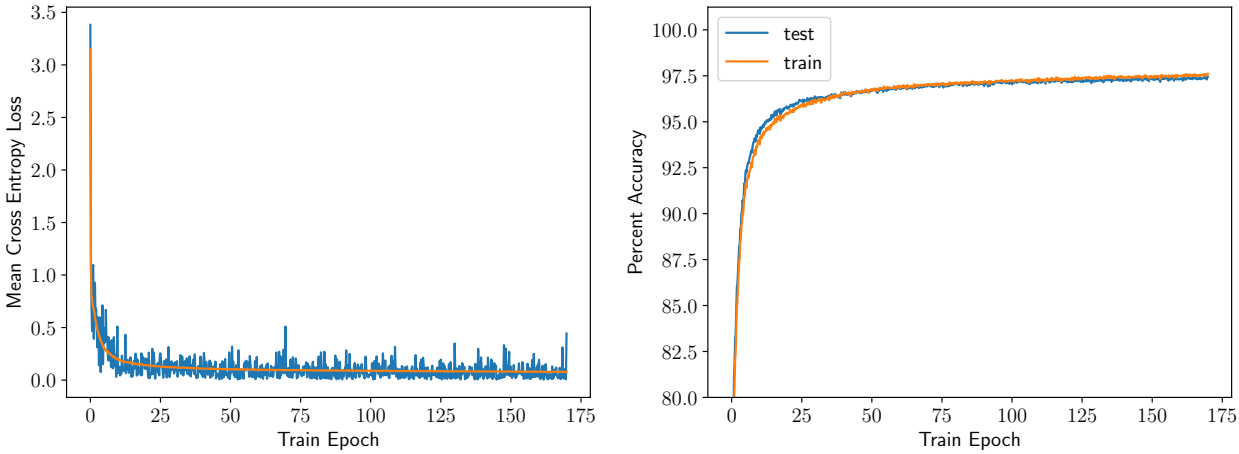


Figure 3.10: Mean cross entropy (left) and classification accuracy (right) for the training data and held out test images during the training process. Both of these indicators track very closely, indiating minimal to no overfitting.

3.4 Performance Evaluation

To evaluate full system performance, we used our trained model to perform inference on a "fresh" S_{21} test dataset, which we never used to train, develop, or tune our model. This dataset consists of a single feedline (call this Elson0) from a different MEC array

[7]Note that this is not a test of end-to-end system performance, as it only indicates the classification accuracy within a set of training images selected using the methods described in 3.2.2

than the one we used in section 3.3, which should provide a good test of model generalization. The dataset has the same span and step size as the training data in both the frequency and power dimensions. To provide a basis for comparison, we also performed resonator identification and tuning using the "baseline methodology", and had two different manual operators tune the drive power of each resonator. We evaluated our model on three different performance criteria, which we detail below: resonator identification, drive power tuning, and frequency placement.

3.4.1 Resonator Identification

For both the baseline methodology and end-to-end ML pipeline, we evaluated the number of true positive, false positive, and false negative identifications. To determine the true and false positive values, we had a manual operator screen all of the resonators identified by each method and flag all false identifications. Since we don't have a "ground truth" list of all of the resonators in Elson0, we estimate the number of false negatives of each method by comparing against the other method; i.e. if a resonator is found by the baseline method but not by the end-to-end ML method, it is considered a false negative of end-to-end ML. To correlate resonators across methods for this analysis, we matched each resonator from the baseline method frequency list to its unique nearest neighbor within 200 kHz in the end-to-end ML method frequency list. Unmatched resonators remaining in each list after this matching process were labeled as false negative identifications for the other method. The results of this analysis are detailed in table 3.2. The end-to-end ML and baseline methods have similar performance on this metric, although the ML algorithm appears to be slightly more sensitive.

	Baseline Method	End-to-end ML
True Positive	1840	1857
False Positive	11	14
False Negative	62	45

Table 3.2: Resonator identification performance on Elson0. The dataset spans 4 Ghz of bandwidth over a single feedline, so in the absence of fabrication errors we expect at most 2044 resonators. All resonators included in this table satisfy the "usable resonator" criteria defined in section 3.1.2.

3.4.2 Drive Power Tuning

There is some uncertainty in the manual selection of resonator drive powers; since it is done heuristically by-eye [28], different operators can assign the same resonator different drive powers and each might be considered "correct". This uncertainty is reflected in the training data, and will fundamentally limit the performance of our pipeline. To quantify this uncertainty and provide a benchmark for model performance, we had two different human operators (call these human1 and human2) tune the drive power of each resonator in the test dataset. In figure 3.11a, we histogram the difference in selected drive power for each resonator between the two manual operators. We find that human1 on average assigns a power approximately 1 dB higher than human2, with $\sigma = 1.4$ dB. In light of this uncertainty, we expect a well performing ML algorithm to behave as a third human operator; i.e. the distributions of power differences between it and each of the human operators should be comparable to figure 3.11a. We histogram these distributions in figure 3.11b. We find these three distributions to be broadly similar; each has a systematic offset (i.e. median) ≤ 1 dB, and a similar spread, with $\approx$90% of resonators falling within 1 dB of the median drive power difference.

We can also perform an analysis similar to the one in (Ref. [28]). For this analysis, we define a "ground truth" power for each resonator by averaging together the powers assigned to it by each human operator. We then define a per-resonator binary accuracy

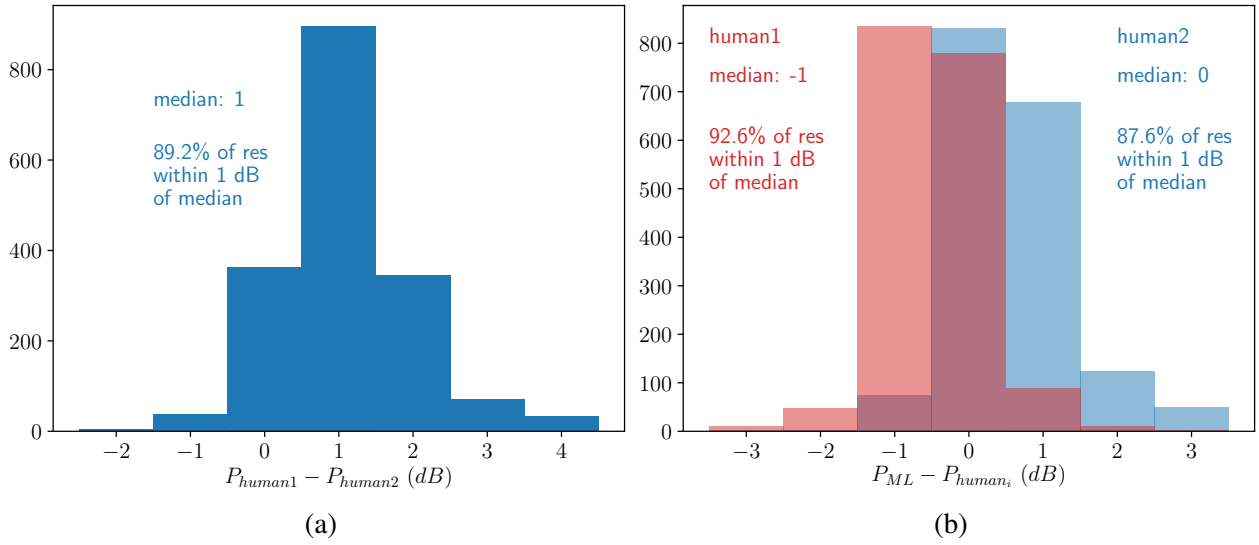


Figure 3.11: Histograms of the per-resonator difference in selected resonator drive power between different tuning methods for the Elson0 dataset. a) Difference between the two human operators; b) Difference between the end-to-end ML method and each human operator. For these comparisons, resonators were correlated between frequency lists using the same method as in section 3.4.1. Power differences are quantized to 1 dB (as per the dataset), which sets the histogram bin spacing

metric: a resonator was assigned the correct drive power by the ML algorithm if and only if $|P_{ML} - P_t| \leq 1$, where P_{ML} and P_t are the ML assigned and "ground truth" powers in dB, respectively. By this metric, our pipeline accurately classifies 93% of the Elson0 resonators (figure 3.12). This is comparable to the 90% accuracy achieved by the ML algorithm (and manual operators!) in (Ref. [28]); both are likely limited by training data nonuniformity.

3.4.3 Frequency Placement

Since the resonator frequency shifts significantly with power, the ideal frequency placement depends on resonator drive power. So, we cannot simply compare the ML selected frequency with the baseline solutions. Instead, we had a human operator check through the end-to-end ML solutions and correct any misplaced resonator frequencies.

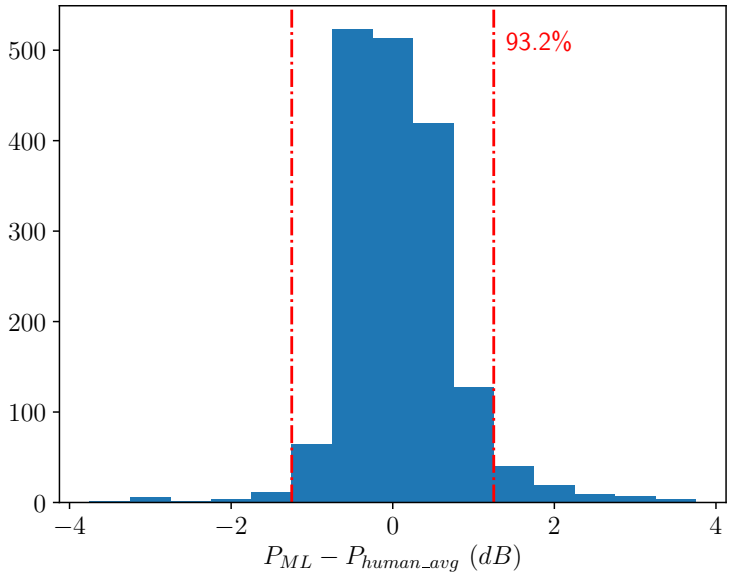


Figure 3.12: Histogram of the per-resonator difference in selected drive power between end-to-end ML and the average human-selected drive power for Elson0. The end-to-end ML method assigns 93% of the resonators a drive power within 1 dB of the human average.

These results are plotted in figure 3.13. 97.8% of resonators were assigned a frequency within 7.629 kHz of the correct value; this represents an error of less than 10% of the typical resonator linewidth, so the impact on performance is likely negligible.

3.4.4 Computational Performance

Using three desktop CPU cores (Intel Core i5), along with an NVIDIA RTX-2080 GPU, our pipeline takes approximately 12 minutes to calibrate a full feedline. Image generation and normalization forms the bulk of this time.

3.5 Conclusion

We have developed a deep learning based analysis pipeline, which, for the first time allows us to completely automate the calibration of optical/IR MKID array frequency

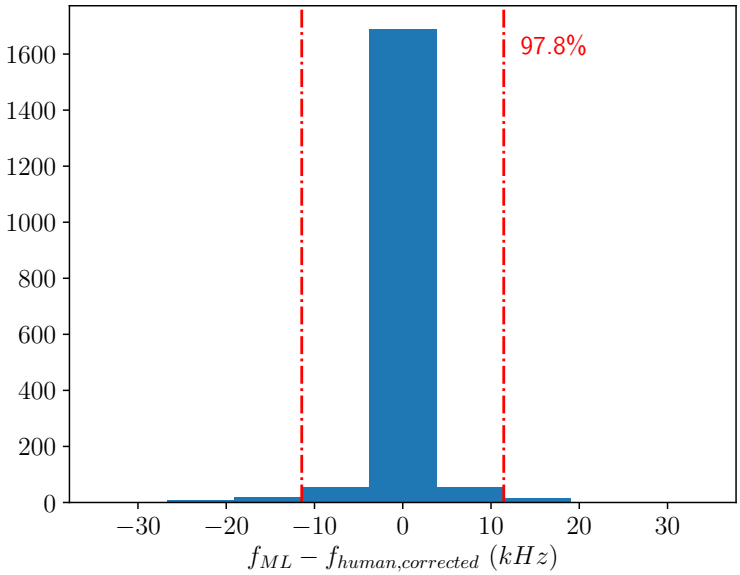


Figure 3.13: Histogram of difference between ML selected frequency and any human correction (performed by a single human operator). Frequency differences are quantized to 7.629 kHz, which sets the histogram bin spacing. 90.4% of resonators required no correction, and 97.8% of resonators were within one quantization step of the corrected frequency.

combs. Our pipeline is in active use, reducing the 40+ hour manual tuning process for the MEC instrument array to less than two hours of computational time. This has signficantly streamlined the array characterization process, and reduced instrument downtime in the event of a warm-up to room temperature (which often causes resonators to shift in frequency). It is also a crucial step towards feasibly scaling up MKID array pixel counts.

When calibrated using our method, MEC has a demonstrated energy resolution $R = 3.8$ and 75% pixel yield on working feedlines ($R = 4.8$ and the yield is 82% on the well-filled "sweet spot" of the array)[21]. See (Ref. [21]) for more details; all array performance metrics provided were obtained using an array calibrated by our pipeline.

The precision of the calibration solutions (particularly resonator drive power) is inherently limited by the lack of consistency of the human-generated training data. Future work will aim to address this; potential solutions involve using analytical fitting methods

to generate or refine training data.

Our pipeline is open source and can be found here: `https://github.com/MazinLab/MKIDReadout`.

Chapter 4

Speckle Control

For FPWC with DARKNESS and MEC, we chose the speckle nulling algorithm for its simplicity: the algorithm doesn't require an optical propagation model, and instead assumes a Fourier-series mapping between the DM pupil and focal plane science camera. The speckle nulling controller operates independently on single Fourier modes, which correspond to sine waves on the DM and point sources (speckles) in the focal plane image. In addition to being relatively straightforward to implement, this simplicity also affords some degree of robustness to changing on-sky conditions [13].

4.1 Fourier-based Optical Propagation Model

Following ([13]) and ([14]), we described the Fourier based model below. The unaberrated pupil plane electric field from an on-axis point source is given by:

$$E(\vec{\rho}) = P(\vec{\rho})A_0 \tag{4.1}$$

where $\vec{\rho}$ is the position in the pupil plane, A_0 is the time averaged electric field amplitude, and $P(\vec{\rho})$ is the pupil transmission function, which is generally defined as:

$$P(\vec{\rho}) = \begin{cases} 1 & |\vec{\rho}| \leq r_p \\ 0 & |\vec{\rho}| > r_p \end{cases} \tag{4.2}$$

where r_p is the radius of the pupil. If we add a sinusoidal phase aberration in the pupil plane, the electric field becomes:

$$E(\vec{\rho}) = P(\vec{\rho})A_0 e^{ia_p cos(\vec{k}\cdot\vec{\rho}+\phi)} \tag{4.3}$$

Assuming the amplitude of the aberration is small ($a_p << 1$), we obtain:

$$E(\vec{\rho}) \approx P(\vec{\rho})A_0[1 + ia_p cos(\vec{k}\cdot\vec{\rho}+\phi)] \tag{4.4}$$

Similarly, we can describe a sinusoidal amplitude aberration using:

$$E(\vec{\rho}) = P(\vec{\rho})[1 + a_a cos(\vec{k}\cdot\vec{\rho}+\phi)]A_0 \tag{4.5}$$

We can combine eqns. 4.4 and 4.5 into a single complex sinusoidal aberration:

$$E(\vec{\rho}) = P(\vec{\rho})[1 + (a_a + ia_p)cos(\vec{k}\cdot\vec{\rho}+\phi)]A_0 \tag{4.6}$$

where we have discarded the product term since $a_a a_p << a_{a,p}$. The corresponding focal plane speckle complex amplitude can be determined by taking the 2D Fourier transform

of eqn. 4.6 as follows:

$$\hat{E}(\vec{r}) \equiv \mathcal{F}[E(\vec{\rho})] = \int P(\vec{\rho})[1 + (a_a + ia_p)cos(\vec{k} \cdot \vec{\rho} + \phi)]A_0 e^{2\pi\vec{r}\cdot\vec{\rho}/\lambda s} d\vec{\rho} \tag{4.7}$$

where $\vec{r}$ is the position in the focal plane, in units of camera pixel coordinates, λ is the optical wavelength, and s is the platescale of the camera; i.e. the number of pixels per angular coordinate (radians) in the focal plane. Using the convolution theorem:

$$\hat{E}(\vec{r}) = A_0\hat{P}(\vec{r}) + A_0\hat{P}(\vec{r}) * \int (a_a + ia_p)cos(\vec{k} \cdot \vec{\rho} + \phi)]e^{2\pi\vec{r}\cdot\vec{\rho}/\lambda s} d\vec{\rho} \tag{4.8}$$

$$= A_0\hat{P}(\vec{r}) + \frac{1}{2}A_0(a_a + ia_p)\hat{P}(\vec{r}) * \left[e^{i\phi}\delta\left(\vec{r} + \frac{\vec{k}\lambda s}{2\pi}\right) + e^{-i\phi}\delta\left(\vec{r} - \frac{\vec{k}\lambda s}{2\pi}\right)\right] \tag{4.9}$$

$$= A_0\hat{P}(\vec{r}) + \frac{1}{2}A_0(a_a + ia_p)\left[e^{i\phi}\hat{P}\left(\vec{r} + \frac{\vec{k}\lambda s}{2\pi}\right) + e^{-i\phi}\hat{P}\left(\vec{r} - \frac{\vec{k}\lambda s}{2\pi}\right)\right] \tag{4.10}$$

The first term of eqn. 4.10 describes the diffraction pattern of the unaberrated on-axis point source, while the second term describes the aberration pattern. For a circular aperture, $\hat{P}(\vec{r})$ is given by:

$$\hat{P}(\vec{r}) = \frac{2J_1(4\pi r_p|\vec{r}|/\lambda s)}{4\pi r_p|\vec{r}|/\lambda s} \tag{4.11}$$

where J_1 is the Bessel function of the first kind, order one. The intensity profile $|\hat{P}(\vec{r})|^2$ is given by an Airy disk profile, and in general is known as the point spread function (PSF) of the imaging system. From the second term of eqn. 4.10, we can see that the sinusoidal aberration creates two copies of the PSF (i.e. "speckles"), centered at $\vec{r} = \pm\frac{\vec{k}\lambda s}{2\pi}$. The relative phase of the speckles is given by the relative magnitudes of a_a and a_p; for a pure phase aberration, the electric field phase is given by $e^{i(\pi/2\pm\phi)}$, and for an amplitude aberration it is $e^{\pm i\phi}$. The full intensity profile is given by:

$$I(\vec{r}) = |\hat{E}(\vec{r})|^2 \tag{4.12}$$

$$= A_0^2|\hat{P}(\vec{r})|^2 + \frac{1}{4}A_0^2(a_a^2 + a_p^2)\left(\left|\hat{P}\left(\vec{r} + \frac{\vec{k}\lambda s}{2\pi}\right)\right|^2 + \left|\hat{P}\left(\vec{r} - \frac{\vec{k}\lambda s}{2\pi}\right)\right|^2\right) \tag{4.13}$$

where we assume $\frac{\vec{k}\lambda s}{2\pi}$ is sufficiently large such that the cross terms $\hat{P}(\vec{r})\hat{P}(\vec{r} \pm \frac{\vec{k}\lambda s}{2\pi})$ are negligible.

4.2 Speckle Nulling Algorithm

The speckle nulling controller is designed to identify and measure the strongest Fourier mode aberrations (i.e. brightest speckles), and use the DM to apply speckles of opposite electric field phase to destructively interfere with ("null") these aberrations. Our implementation broadly follows [13], with a few modifications to optimize for the high-framerate regime.

The DM can apply pupil plane phase aberrations of the following form:

$$\varphi_{DM}(\vec{\rho}) = a_p cos(\vec{k} \cdot \vec{\rho} + \phi_{DM}) \tag{4.14}$$

$$= \frac{2\pi a_{DM}}{\lambda} cos(\vec{k} \cdot \vec{\rho} + \phi_{DM}) \tag{4.15}$$

where $2\pi/\lambda$ describes the conversion factor between the optical phase a_p in radians, and the physical DM stroke a_{DM}.

To determine the value of $\lambda s/2\pi$ (i.e. the relationship between pupil plane spatial frequency $\vec{k}$ and speckle position $\vec{r}$ in the focal plane), we use a calibration script that applies DM modes of a specified amplitude and frequency and measures the position of the resulting speckles relative to the PSF center. In practice, the relationship between

the modal amplitude (a_{DM}) and focal plane intensity is difficult to calculate and so is also calibrated in the same manner. Denote these calibration constants α and β, respectively. Then, plugging in α and β into eqn. 4.10, the focal plane electric field corresponding to the DM mode in eqn 4.15 is given by:

$$\hat{E}_{DM}(\vec{r}) = \frac{\beta a_{DM}}{\sqrt{I_{ap}}} \left[e^{i(\pi/2+\phi_{DM})} \hat{P}\left(\vec{r} + \frac{\vec{k}}{\alpha}\right) + e^{i(\pi/2-\phi_{DM})} \hat{P}\left(\vec{r} - \frac{\vec{k}}{\alpha}\right) \right] \tag{4.16}$$

with corresponding intensity:

$$I_{DM}(\vec{r}) = \frac{(\beta a_{DM})^2}{I_{ap}} \left[\left| \hat{P}\left(\vec{r} + \frac{\vec{k}}{\alpha}\right) \right|^2 + \left| \hat{P}\left(\vec{r} - \frac{\vec{k}}{\alpha}\right) \right|^2 \right] \tag{4.17}$$

where $(\beta a_{DM})^2$ is the measured speckle intensity inside of some aperture centered around the speckle (usually with angular size $\approx \frac{\lambda}{D}$), and I_{ap} is a constant describing the relationship between the peak intensity of the Airy disk and the total intensity inside of the speckle aperture:

$$I_{ap} \equiv \int_{ap} \left| \hat{P}(\vec{r} - \vec{r}_s) \right|^2 d^2\vec{r} \tag{4.18}$$

where the integral is over a fixed-radius circular aperture centered around $\vec{r}_s$.

In light of the calibrations defined above, we can rewrite the intensity of a general sinusiodal aberration (eqn. 4.13) in terms of measurable quantities:

$$I(\vec{r}) = \frac{I_0}{I_{ap}} |\hat{P}(\vec{r})|^2 + \frac{I_s}{I_{ap}} \left(|\hat{P}(\vec{r} + \vec{r}_s)|^2 + |\hat{P}(\vec{r} - \vec{r}_s)|^2 \right) \tag{4.19}$$

where I_0 is the intensity of the unocculted PSF, I_s is the unocculted speckle intensity, and $\pm\vec{r}_s$ are the measured speckle centroids in the focal plane image. This has corresponding

electric field:

$$E(\vec{r}) = \sqrt{\frac{I_0}{I_{ap}}}\hat{P}(\vec{r}) + \sqrt{\frac{I_s}{I_{ap}}}\left[e^{i(\phi_0+\phi)}\hat{P}(\vec{r}+\vec{r_s}) + e^{i(\phi_0-\phi)}\hat{P}(\vec{r}-\vec{r_s})\right] \tag{4.20}$$

where we've re-introduced the pupil plane phase ϕ. ϕ_0 is determined by the relative magnitudes of the phase and amplitude aberrations (i.e. $\phi_0 = \arctan a_p/a_s$).

If we sum eqns 4.16 and 4.20, we can determine the DM amplitude a_{DM}, phase ϕ_{DM}, and spatial freqency k_{DM} required to null the original speckle. By inspection, we can immediately determine:

$$a_{DM} = \frac{\sqrt{I_s}}{\beta} \tag{4.21}$$

$$k_{DM} = \alpha\vec{r_s} \tag{4.22}$$

For the speckle phase, we can only null both speckles at $\pm\vec{r_s}$ if $\phi_0 = \pi/2$; i.e. the speckles result from a pure phase-type aberration. This is intuitively clear, since the DM can only control the electric field phase[1]. So, in general we choose a region on a single side of the PSF (know as the "control region") over which to null speckles. Then, the DM phase is given by:

$$\pi/2 \pm \phi_{DM} = \phi_0 \pm \phi + \pi \tag{4.23}$$

$$\pm\phi_{DM} = \phi_0 \pm \phi + \pi/2 \tag{4.24}$$

$$\phi_{DM} = \phi \pm (\phi_0 + \pi/2) \tag{4.25}$$

where the $+$ or $-$ depends on which speckle in the pair we are choosing to null.

Focal plane images measure intensity, so we cannot directly measure ϕ (or ϕ_0). We

[1]This is true for both P3K and SCExAO, but some high contrast imaging testbeds have conjugate DMs to correct both phase and amplitude

instead use the DM to apply "probe" speckles of known phase (with amplitude and spatial frequency given by eqns. 4.21 and 4.22) to interfere coherently with the speckle we are trying to correct. With enough probe measurements we can solve for the electric field phase (and more robustly estimate the amplitude). We describe the probing process and the calculation of nulling solutions in the following subsections.

4.2.1 Probing

For each identified speckle at position $\vec{r_s}$ with intensity I_s, generate DM probes of the form $\varphi_p(\vec{\rho}) = \frac{\sqrt{I_s}}{\beta} cos(\alpha \vec{r_s} \cdot \vec{\rho} + \phi_p)$. These probes will coherently interfere with the identified speckle, allowing us to infer the speckle phase. The amplitude of the applied probe was chosen to match the expected amplitude of the identified speckle, which is important for maximizing SNR [13]. In our implementation, we use four probe phases $\phi_p = [0, \frac{\pi}{2}, \pi, \frac{3\pi}{2}]$. After each probe is applied, we measure the resulting speckle intensity over some predetermined aperture; denote these measurments $[I_1, I_2, I_3, I_4]$. The focal plane electric field for each probe measurement is given by:

$$\hat{E}_p(\vec{r}) = \sqrt{\frac{I_s}{I_{ap}}} e^{i(\phi_0 - \phi)} \hat{P}(\vec{r} - \vec{r_s}) + \frac{\beta a_{DM}}{\sqrt{I_{ap}}} e^{i(\pi/2 - \phi_p)} \hat{P}(\vec{r} - \vec{r_s}) \tag{4.26}$$

where we've picked the speckle at $+\vec{r_s}$. The corresponding intensity is:

$$\hat{I}_p(\vec{r}) = \frac{1}{I_{ap}} \Big[I_s + (\beta a_{DM})^2 + \beta a_{DM} \sqrt{I_s} \left(e^{i(-\phi_0 + \phi + \pi/2 - \phi_p)} + e^{i(\phi_0 - \phi - \pi/2 + \phi_p)} \right) \Big] \left| \hat{P}(\vec{r} - \vec{r_s}) \right|^2 \tag{4.27}$$

$$\hat{I}_p(\vec{r}) = \frac{1}{I_{ap}}\left[I_s + (\beta a_{DM})^2 + 2\beta a_{DM}\sqrt{I_s}\cos\left(\phi - \phi_0 + \pi/2 - \phi_p\right)\right]\left|\hat{P}(\vec{r} - \vec{r_s})\right|^2 \quad (4.28)$$

$$= \frac{1}{I_{ap}}\left[I_s + (\beta a_{DM})^2 + 2\beta a_{DM}\sqrt{I_s}\cos\left(\phi_s - \phi_p\right)\right]\left|\hat{P}(\vec{r} - \vec{r_s})\right|^2 \quad (4.29)$$

where we set $\phi_s \equiv \phi - \phi_0 + \pi/2$ for clarity. Since our intensity measurements $I_{1..4}$ are defined over the speckle aperture (eqn. 4.18), we have:

$$I_p \equiv \int_{ap} \hat{I}_p(\vec{r}) d^2\vec{r} \quad (4.30)$$

$$= I_s + (\beta a_{DM})^2 + 2\beta a_{DM}\sqrt{I_s}\cos\left(\phi_s - \phi_p\right) \quad (4.31)$$

Then, plugging in each of the four probes defined above we have:

$$I_1 = I_s + (\beta a_{DM})^2 + 2\beta a_{DM}\sqrt{I_s}\cos\left(\phi_s\right) \quad (4.32)$$

$$I_2 = I_s + (\beta a_{DM})^2 + 2\beta a_{DM}\sqrt{I_s}\cos\left(\phi_s - \pi/2\right) \quad (4.33)$$

$$I_3 = I_s + (\beta a_{DM})^2 + 2\beta a_{DM}\sqrt{I_s}\cos\left(\phi_s - \pi\right) \quad (4.34)$$

$$I_4 = I_s + (\beta a_{DM})^2 + 2\beta a_{DM}\sqrt{I_s}\cos\left(\phi_s - 3\pi/2\right) \quad (4.35)$$

4.2.2 Calculation of Nulling Solution

Combining the above measurements, we have:

$$I_1 - I_3 = 2\beta a_{DM}\sqrt{I_s}\left[\cos(\phi_s) - \cos(\phi_s - \pi)\right] \quad (4.36)$$

$$= 4\beta a_{DM}\sqrt{I_s}\cos(\phi_s) \quad (4.37)$$

$$I_2 - I_4 = 2\beta a_{DM}\sqrt{I_s}\left[\cos(\phi_s - \pi/2) - \cos(\phi_s + \pi/2)\right] \quad (4.38)$$

$$= 4\beta a_{DM}\sqrt{I_s}\sin(\phi_s) \quad (4.39)$$

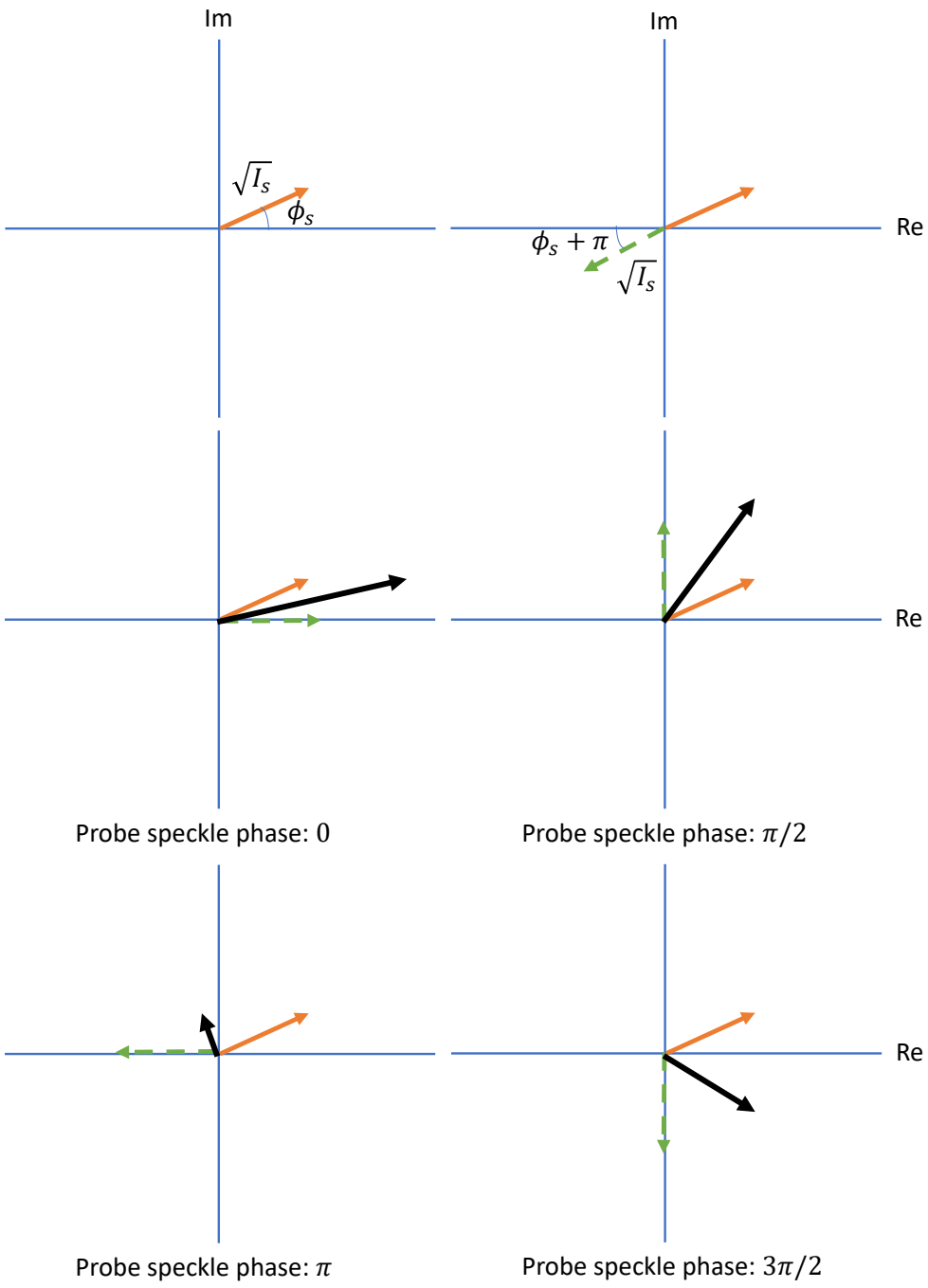


Figure 4.1: Schematic of probing process. Speckle with amplitude $\sqrt{I_s}$ and phase ϕ_s (top left) is measured using four DM probes (bottom four plots). The orange and dashed green lines show the complex amplitude of the original and probe speckles, respectively. The amplitude of the resulting speckle produced by the interference of these speckles are indicated by the solid black line, the intensity of which (i.e. length of the arrow squared) is measured in the focal plane. The top right plot shows the nulling solution.

By combining eqns. 4.37 and 4.39, we obtain:

$$\phi_s = \arctan \frac{I_1 - I_3}{I_2 - I_4} \tag{4.40}$$

A potentially more robust estimate of the speckle amplitude $\sqrt{I_s}$ can also be obtained from the probe measurements:

$$(I_1 - I_3)^2 + (I_2 - I_4)^2 = 16\beta a_{DM}^2 I_s \left[cos^2(\phi_s) + sin^2(\phi_s)\right] \tag{4.41}$$

$$I_s = \frac{(I_1 - I_3)^2 + (I_2 - I_4)^2}{16\beta a_{DM}^2} \tag{4.42}$$

This method will reject any incoherent sources/background, since these will show up as equal additive terms in all four probe measurements. Depending on the strength of the modulation, this method may also improve SNR in the photon noise limited regime.

It is often useful to write down the measured phase and amplitude as a single complex number:

$$A_s \equiv \sqrt{I_s} e^{i\phi_s} \tag{4.43}$$

$$= \sqrt{I_s}\cos\phi_s + i\sqrt{I_s}\sin\phi_s \tag{4.44}$$

$$= \frac{(I_1 - I_3) + i(I_2 - I_4)}{4\beta a_{DM}} \tag{4.45}$$

Eqn. 4.45 is linear in the probe measurements, making it straightforward to combine multiple measurements, propagate errors, and apply temporal filtering schemes (e.g. Kalman filtering).

It is important to determine the SNR of the estimated speckle complex amplitude, especially in the high-framerate regime, where the probe measurements are often photon-starved. If the estimate has insufficient SNR, using it to apply a correction actually in-

crease the intensity inside the control region, worsening the image quality. In this regime, our measurements are Poisson noise limited ($\sigma_I^2 = I$). Propagating this error through eqn. 4.45, we can determine the uncertainties on the real and imaginary components of A_s:

$$\sigma^2_{\mathbf{Re}[A_s]} = \frac{I_1 + I_3}{16\beta^2 a_{DM}^2} \tag{4.46}$$

$$\sigma^2_{\mathbf{Im}[A_s]} = \frac{I_2 + I_4}{16\beta^2 a_{DM}^2} \tag{4.47}$$

The associated SNR of each component is:

$$SNR_{\mathbf{Re}[A_s]} = \frac{|I_1 - I_3|}{\sqrt{I_1 + I_3}} \tag{4.48}$$

$$SNR_{\mathbf{Im}[A_s]} = \frac{|I_2 - I_4|}{\sqrt{I_2 + I_4}} \tag{4.49}$$

4.3 Realtime Data Pipeline

The raw MKID data is formatted as a list of photon detection events, each tagged with the time of detection, pulse amplitude, and pixel coordinates. These photon lists can then be calibrated and binned into images with a user-defined exposure time and wavelength range for analysis in realtime or post-processing. In this section, we describe a realtime data pipeline for generating wavelength calibrated images on-demand using the raw photon list data streaming from the readout FPGAs. The raw photon lists are also saved to disk, so the pipeline has no effect on the analysis steps in post-processing. Our pipeline is a key component of the MKID speckle control codebase and is also used in the realtime observing display.

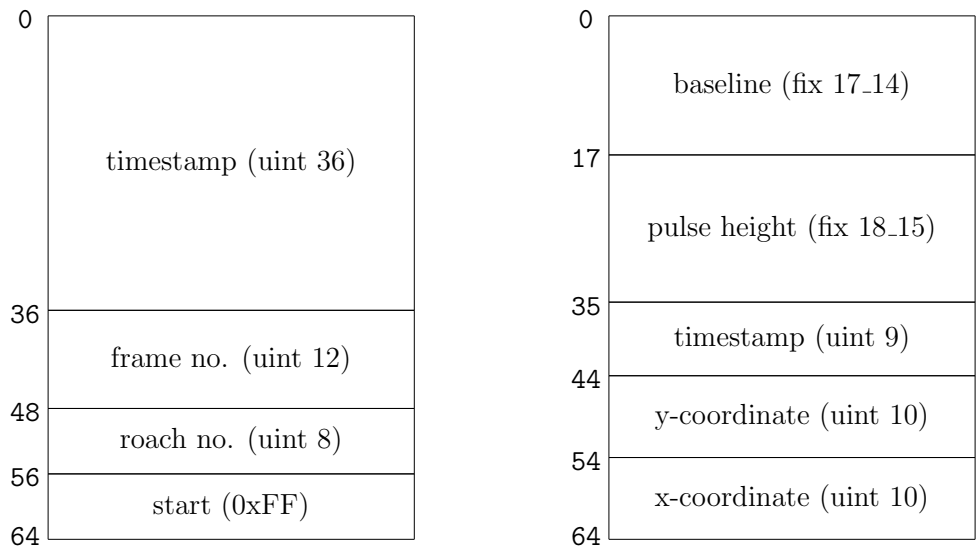
4.3.1 Raw Data Format

The readout FPGAs encode each photon event as a 64-bit word, and pack these words into UDP frames for transmission to the data server. Each UDP frame contains a global "header" word, containing the current time (as a UNIX timestamp) to the nearest 500 μs and a serial number identifying the readout unit (figure 4.2a). Each photon word contains the detection time in microseconds relative to the header word timestamp, pulse amplitude, pixel coordinates, and baseline filter value (figure 4.2b). To ensure that all photon words in a UDP frame occur within 500 μs after the header timestamp, frames are sent at least every 500 μs. We also impose an upper limit of 100 photon words per frame, so frames can be sent more often if necessary.

Pixel coordinates are matched to resonator readout channels using a calibration process known as "beammapping" [50, 51]; this mapping is loaded into the firmware prior to taking any observational data.

4.3.2 Packetmaster

Packetmaster is the primary application used for receiving, parsing, and saving photon data. It is primarily written in C, but can be fully configured and run using a Python API (written in Cython). Packetmaster can be configured to have up to three threads: Reader, for receiving UDP packets; Writer, for dumping the contents of these packets to disk for later analysis; and SharedImageWriter, for generating calibrated realtime images. Multithreading, sharing memory, and process synchronization are implemented using standard POSIX tools.



(a) header word (b) photon word

Figure 4.2: Data format descriptions of the header (left) and photon (right) words. The MSB of the header word is set to 0xFF (i.e. 8 ones in binary) in firmware to distinguish it from the photon words and indicate the start of a new UDP frame. These data structures are implemented in Packetmaster as packed bitfield C structs.

Reader

The "Reader" thread listens for UDP frames on a specified port and dumps their contents to a shared memory buffer that is accessible to the "Writer" and "SharedImageWriter" threads. For receiving packets, we find that it is important to increase the size of the system ethernet buffer to 512 MB to prevent overflow. The shared memory buffer is implemented as a read-only ring buffer (figure 4.3). This provides a safe, straightforward way to share resources while allowing each thread to control how and when it accesses the raw photon data. The ring buffer is implemented as a C struct containing a pointer to the data, and indices specifying the current write location in the buffer (denote this i_p where p indicates that this is the index of the "producer" thread) and the number of total write cycles (i.e. the number of times the reader thread has reached the end of the buffer and looped back to the beginning; denote this n_p). Each thread reading data (i.e.

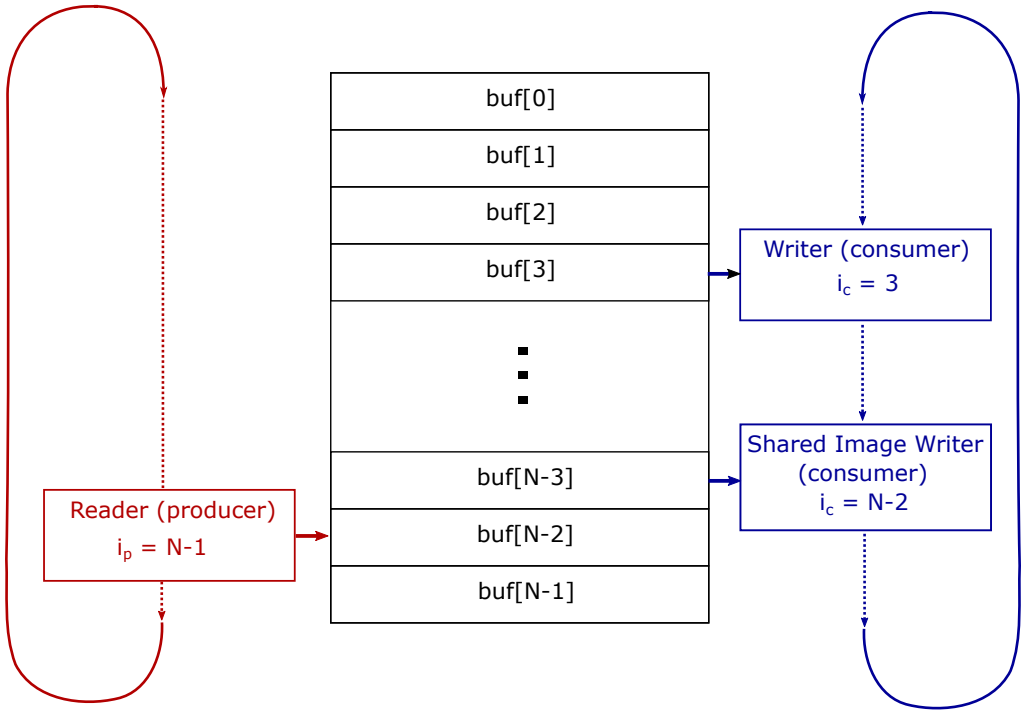


Figure 4.3: Schematic of the ring buffer data structure. Each cell (`buf[i]`) represents a photon or header word. Reader is the "producer" thread which continuously writes to the buffer as it receives UDP frames. Once it reaches the end of the buffer (`buf[N-1]`), Reader will return to the buffer start (`buf[0]`) and continue writing, overwriting existing data. So, in the diagram above, the oldest word is at (`buf[N-1]`), and the most recent is being written to `buf[N-2]`. The "consumer" threads read data sequentially from the buffer as needed. It is important to make the buffer large enough such that data isn't overwritten by Reader before they can be read by the consumer thread(s).

"consumer") from the ring buffer has a pointer to the current read location (i_c) and a counter tracking the number of total read cycles (n_c). Each time the buffer is read from, the consumer thread will compare its own read and total cycle pointers with those of the Reader thread, to ensure continuity of the data stream (e.g. if $i_p > i_c$ and $n_p > n_c$, we know that the producer thread has overwritten the data between i_c and i_p before it was read by the consumer).

Writer

The "Writer" thread simply reads the raw data from the ring buffer and saves it to disk for later analysis. A shared memory interface is used by the master process (e.g. Python program that spawned the Packetmaster instance) to specify the write path and command Writer to start/stop saving to disk. Data is saved in files labeled by the current UNIX timestamp in seconds (as measured by the data server system clock). A new file is started every second. These filename timestamps are not guaranteed to line up with the photon/header timestamps in the raw data, since there may be misalignments between the data server system clock and PPS signal used to synchronize the readout FPGAs.

SharedImageWriter

The "SharedImageWriter" thread parses the raw data and bins photon events into (optionally) wavelength calibrated images as specified by the shared memory interface. In this section, we focus on the packet parsing aspects of SharedImageWriter; the shared memory API and its interfacing with SharedImageWriter is covered in section 4.3.3.

Both the system ethernet buffer and the Reader provided ring buffer don't distinguish between header words and photon words; data is simply extracted from the raw UDP frames and stored sequentially in memory with no reference to the frame boundaries. It is however guaranteed that data from a single UDP frame will occupy a contiguous location in memory (modulo any wrapping at the ring buffer boundaries). So, when parsing raw data we must first check each 64-bit word to determine whether it is a header or photon word. If it is a header word, we know that all subsequent photon words between it and the next header word belong to the same UDP frame, and thus share a common header timestamp. To distinguish between header and photon words, we can simply check the most significant byte; for header words, this is set to 0xFF by the firmware. This will

never occur in a photon word, since it would require the x-coordinate to be ≥ 1020, which is well outside the boundaries of current generation MKID arrays.

Once a header word is identified, subsequent photon words can be parsed to determine the full UTC timestamp, pulse amplitude, and coordinates of each photon event. In general, parsing header/photon words is done simply by typecasting it into the proper C struct and examining the relevant fields. Before parsing, the byte order of each word must be swapped, since the UDP frames use big-endian (network) byte order, while the x86 systems running Packetmaster are little-endian [2]. Once photon events are fully parsed, they may be calibrated and/or binned into images as specified by the shared memory interface. SharedImageWriter will continuously parse out the raw data stream in this manner; decisions about whether/how to add a photon to an image are made based on the current state of the shared memory interface. Photon events are immediately discarded after being parsed (and possibly added to an image); SharedImageWriter has no "memory" of previous photons.

To convert phase pulse height to wavelength, we use a function of the form:

$$E = a\phi^2 + b\phi + c \tag{4.50}$$

where $E = hc/\lambda$ is the photon energy and ϕ is the pulse height in radians. The coefficients a, b, and c are unique to each pixel, and are determined using calibration data taken at known monochromatic wavelengths [21]. To bin shared memory images according to wavelength, we provide Packetmaster with a buffer containing these coefficients; SharedImageWriter can then calculate the wavelength of every parsed photon.

[2] "Endianness" refers to the ordering of bytes within numerical data types; "big-endian" systems have the most significant byte at the lowest memory address, while "little-endian" systems start with the least significant byte. The format descriptions in figure 4.3.1 assume little-endian byte ordering

4.3.3 Shared Memory Interface

The MKID shared memory interface provides an API for requesting on-demand images from Packetmaster with relatively low latency ($\sim$ 1 ms). The API allows a client program (e.g. speckle nulling controller) to specify the start time, integration time, and optionally a set of wavelength bins. It has hooks written in both C and Python.

The start time and integration time can be specified to 0.5 ms precision (corresponding to the UDP header timestamps), and can be changed on the fly for each new image. The wavelength range can also be set dynamically, but the number of wavelength bins must be specified when creating the image buffer.

The interface is implemented as a system-wide shared memory buffer containing the image data (call this `image_buffer`), and metadata (call this `image_metadata`) parameters that can be set by the client program. A set of C functions is used to interface with this buffer; the client program can optionally use a dedicated Python API instead. Synchronization of the client program with the SharedImageWriter thread is done using POSIX semaphores (which are named in the shared metadata). A semaphore is a data structure used for sharing resources between processes. In its simplest form, it is a counter indicating the number of a certain resource that is currently available. A process requiring the resource can perform a `wait` operation on the semaphore, pausing execution (hence access to the resource) until the value of the semaphore is greater than zero. A process using the resource will `post` the semaphore when it is finished, indicating that the resource has freed up. Our interface uses two semaphores: one to trigger SharedImageWriter to start taking an exposure (call this `take_image`), and another for SharedImageWriter to indicate that the exposure is complete (call this `done_image`). The communication protocol for taking a single exposure is detailed below (and outlined in figure 4.4):

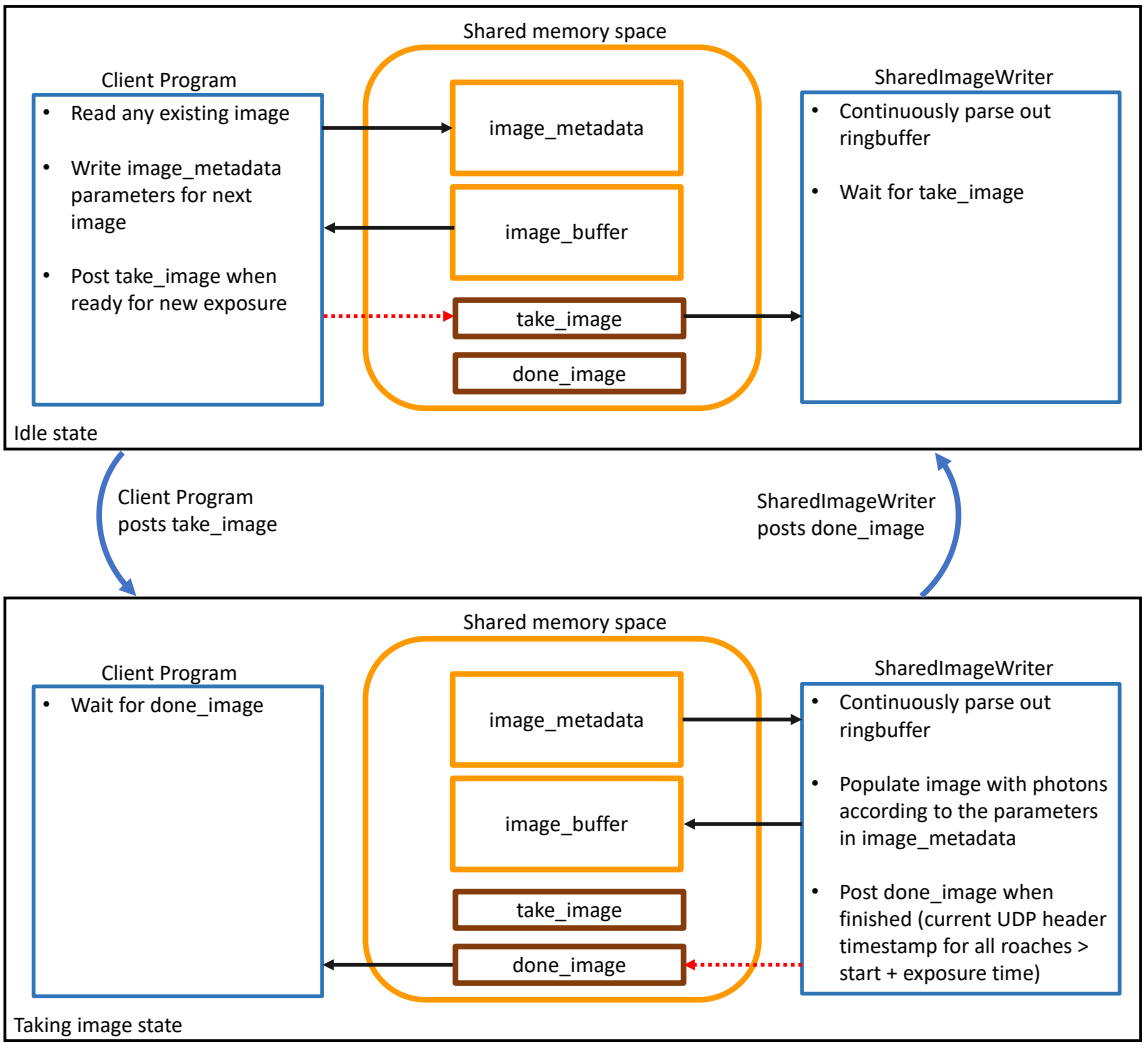


Figure 4.4: State machine diagram of shared memory interface. In the idle state (top), no image data is being written to `image_buffer`, and the client program is free to read from it and write metadata parameters for subsequent exposures. To trigger a new exposure, the client program will post `take_image`. In the taking image state (bottom), SharedImageWriter will overwrite `image_buffer` with data from a new exposure. Posting `done_image` will transition back to the idle state.

1. Client program sets the desired image parameters, then posts (`take_image`). It then waits on `done_image` before reading from the image buffer.

2. When `take_image` is posted, SharedImageWriter reads the image parameters from the shared memory buffer, clears the content of `image_buffer`, and begins populating `image_buffer` with photons within the start time, integration time, and wavelength range specified by the client program. If the specified start time is 0, SharedImageWriter will set it to the header timestamp of the most recently parsed UDP frame.

3. SharedImageWriter will continue to populate the image until all ROACH header timestamps exceed the start time plus the integration time. It will then `post done_image`, at which point the client program is free to read `image_buffer`.

4.4 MKID Speckle Nulling Code

Our implementation of the speckle nulling algorithm largely follows [13, 14], but we made a few modifications to improve performance in the high framerate, photon noise limited regime, as well as to account for the relatively low pixel yield of the MEC and DARKNESS arrays. These changes are described in the following subsections; a full account of the algorithm is given below in algorithm 1.

4.4.1 Speckle Detection

The speckle detection step finds the brightest speckles within the control region, and measures the speckle position and intensity for the purposes of calculating the DM probe to apply (eqns. 4.21 and 4.22).

Algorithm 1: Speckle Nulling Loop

input: num_iters, snr_thresh, int_time, exclusion_radius, max_n_speckles

phases = $[0, \pi/2, \pi, 3\pi/2]$ `// list of probe phases`
speckles = [] `// list of speckles being probed/nulled`
for $n \leftarrow 1$ **to** num_iters **do**
    image $\leftarrow$ `new_exposure(int_time)`
    new_speckles $\leftarrow$ `detect_speckles(image)`
    `/* detect_speckles returns a list of speckles (up to max_n_speckles), and measures the position and intensity of each speckle according to sections 4.4.1 and 4.4.2, respectively */`
    delete any speckle in new_speckles with a focal plane position that is within exclusion_radius of any speckle in speckles
    append the brightest m speckles in new_speckles to speckles, where m = max_n_speckles $- len$(speckles)
    `/* probe measurements: */`
    for $i \leftarrow 1$ **to** 4 **do**
        $\phi_p \leftarrow$ phases [i]
        for speckle *in* speckles **do**
            apply speckle probe $a_{DM} \cos(\vec{k}_{DM} \cdot \vec{\rho} + \phi_p)$ to DM
            `/*` $\vec{k}$ `and` a_{DM} `are calculated from the measured speckle intensity and position as specified in eqns. 4.21 and 4.22 */`
        end
        image $\leftarrow$ `new_exposure(int_time)`
        for speckle *in* speckles **do**
            speckle.`add_probe_measurement(image)`
            `/* add_probe_measurement measures the speckle aperture intensity and` σ `according to eqns. 4.53 and 4.54, and combines with any previous measurements (section 4.4.3) */`
        end
        remove all speckle probes from DM
    end
    `/* caclulate nulling solutions: */`
    for speckle *in* speckles **do**
        calculate speckle complex amplitude A_s according to eqn. 4.45
        calculate speckle signal to noise (snr) according to eqn. 4.61
        if snr $>$ snr_thresh **then**
            apply nulling solution: $|A_s| \cos(\vec{k}_{DM} \cdot \rho + \arg\{A_s\})$ to DM
            delete speckle from speckles
        end
    end
end

To help fill in missing pixels, we first apply a normalized Gaussian blur filter of the following form:

$$\text{Filt}\left[I(\vec{r})\right] \equiv \frac{I(\vec{r}) * g(\vec{r})}{\text{GPM}(\vec{r}) * g(\vec{r})} \tag{4.51}$$

where $I(\vec{r})$ is the image of the control region, $*$ denotes the convolution operator, and $\text{GPM}(\vec{r})$ is the "good pixel mask", such that $\text{GPM}(\vec{r}) = 1$ if the pixel at $\vec{r}$ is usable, and $\text{GPM}(\vec{r}) = 0$ if not. The exact usability criteria depend on the configuration of the system; e.g. if we are specifying a wavelength range, we require that all usable pixels be wavelength calibrated. If not, we usually only require that they be photosensitive. $g(\vec{r})$ is a normalized Gaussian kernel, with $\sigma = 0.42 \times 2\pi/\alpha$ (this value was chosen to best approximate the core of the Airy function profile). Intuitively, this filter normalizes the output of an ordinary Gaussian blur by the expected output given uniform illumination. We also find that upsampling $I(\vec{r})$ before applying the filter improves speckle centroiding precision; we typically upsample by a factor of 2. After upsampling and filtering, we identify the speckle locations simply by picking the N brightest local maxima. Speckle intensity is measured using the procedure described in section 4.4.2.

4.4.2 Aperture Intensity Measurement

To measure the speckle intensity, we first define a circular aperture mask centered around the identified speckle location. The size of the aperture is left as a tunable parameter, but we generally define it to be close to the resolution limit (λ/D, which is ≈ 3 pixels in diameter). Within the aperture, we define the good pixel fraction:

$$\text{GPF} = \frac{\Sigma_{ap}\text{GPM}(\vec{r})}{N_{pix}} \tag{4.52}$$

where N_{pix} is the total number of pixels in the aperture. We then perform the following correction to the raw intensity measurement:

$$I_{meas} = \frac{I_{ap}}{GPF} \tag{4.53}$$

where I_{ap} is the raw intensity inside the aperture. This has associated variance:

$$\sigma^2_{meas} = \frac{I_{ap}}{GPF^2} \tag{4.54}$$

This method of measuring intensity and variance is used for both the initial intensity measurement and the probe measurements.

4.4.3 Multiple Probe Iterations

In the high framerate, photon noise limited regime, a single iteration of probe measurements might not provide enough signal-to-noise to accurately measure the speckle phase and amplitude. So, we integrate signal over multiple probe iterations, and only apply the correction once the speckle SNR has exceeded some threshold value. The resulting probe intensity is simply an average over all of the probe cycles:

$$I_i = \frac{1}{N}\sum_{n=1}^{N} I_{i,n} \tag{4.55}$$

where i indexes the probe speckle phase, and n indexes the probe iterations. The resulting variance is given by:

$$\sigma_i^2 = \frac{1}{N^2}\sum_{n=1}^{N} \sigma_{i,n}^2 \tag{4.56}$$

Plugging eqn. 4.55 into eqn. 4.45, the measured speckle complex amplitude is:

$$A_s = \frac{\sum_{n=1}^{N} \left[(I_{1,n} - I_{3,n}) + i(I_{2,n} - I_{4,n})\right]}{4N\beta a_{DM}} \tag{4.57}$$

To calculate the SNR of A_s, define

$$\sigma^2_{A_s,avg} = \frac{1}{2}\left(\sigma^2_{\mathbf{Re}[A_s]} + \sigma^2_{\mathbf{Im}[A_s]}\right) \tag{4.58}$$

$$= \frac{\sigma_1^2 + \sigma_2^2 + \sigma_3^2 + \sigma_4^2}{32\beta^2 a_{DM}^2} \tag{4.59}$$

where we've combined eqns. 4.46 and 4.47, and replaced the raw intensities with the combined variances defined in eqn. 4.56. If we make the approximation that $\sigma^2_{\mathbf{Re}[A_s]} \approx \sigma^2_{\mathbf{Im}[A_s]}$, and set $\sigma^2_{\mathbf{Re}[A_s]} = \sigma^2_{A_s,avg}$ and $\sigma^2_{\mathbf{Im}[A_s]} = \sigma^2_{A_s,avg}$, the SNR of the speckle focal plane amplitude ($|A_s|$) is given by:

$$SNR_{|A_s|} = \frac{|A_s|}{\sigma_{A_s,avg}} \tag{4.60}$$

$$= \sqrt{\frac{2\left[(I_1 - I_3)^2 + (I_2 - I_4)^2\right]}{\sigma_1^2 + \sigma_2^2 + \sigma_3^2 + \sigma_4^2}} \tag{4.61}$$

4.4.4 Codebase

The speckle nulling code was written in C++ for performance, and for efficient interfacing with the various hardware modules (e.g. the MKID realtime pipeline). There is also a Python API for accessing high-level functionality, such as starting the control loop, gathering telemetry, and configuring parameters.

Most of the core functionality is implemented as C++ classes. The control loop itself (described in algorithm 1) is implemented as a state machine (inside a class), with

hooks for updating it with new images, advancing the control iteration, and updating the DM. Each identified speckle is also implemented as a state machine object, with hooks for updating the speckle with new images and retrieving DM probe/null maps. These objects have attributes to store the speckle position, intensity and probe measurements, and internal methods for measuring the intensity within the aperture, calculating SNR, etc. The code has interchangeable, modular interfaces to the CACAO and P3K DM controllers, abstracting away most hardware differences.

We also include a simple simulator with hooks that are identical to those of the MKID realtime pipeline and CACAO DM interface, making any control code interchangeable between hardware and simulation.

The codebase is open source and can be found at: `https://github.com/MazinLab/MKIDSpeckleControl`.

4.5 In-lab Testing

4.5.1 DARKNESS

We performed in-lab speckle nulling tests with DARKNESS at Palomar Observatory, behind the Stellar Double Coronagraph (SDC), and PALM-3000 (P3K) adaptive optics system. We used the P3K internal white-light source to provide the PSF. The speckle nulling code was run on the DARKNESS data server. The MKID data stream was provided over a 10 Gbit fiber link. Commands to the P3K controller (DM actuator maps, etc.) were sent over TCP via a LAN connection.

Actuator maps can be applied to the DM either directly, or as a wavefront sensor convergence point offset (figure 1.5). For the latter method, the response of the wavefront sensor to the actuator map is calculated (usually with a matrix multiplication);

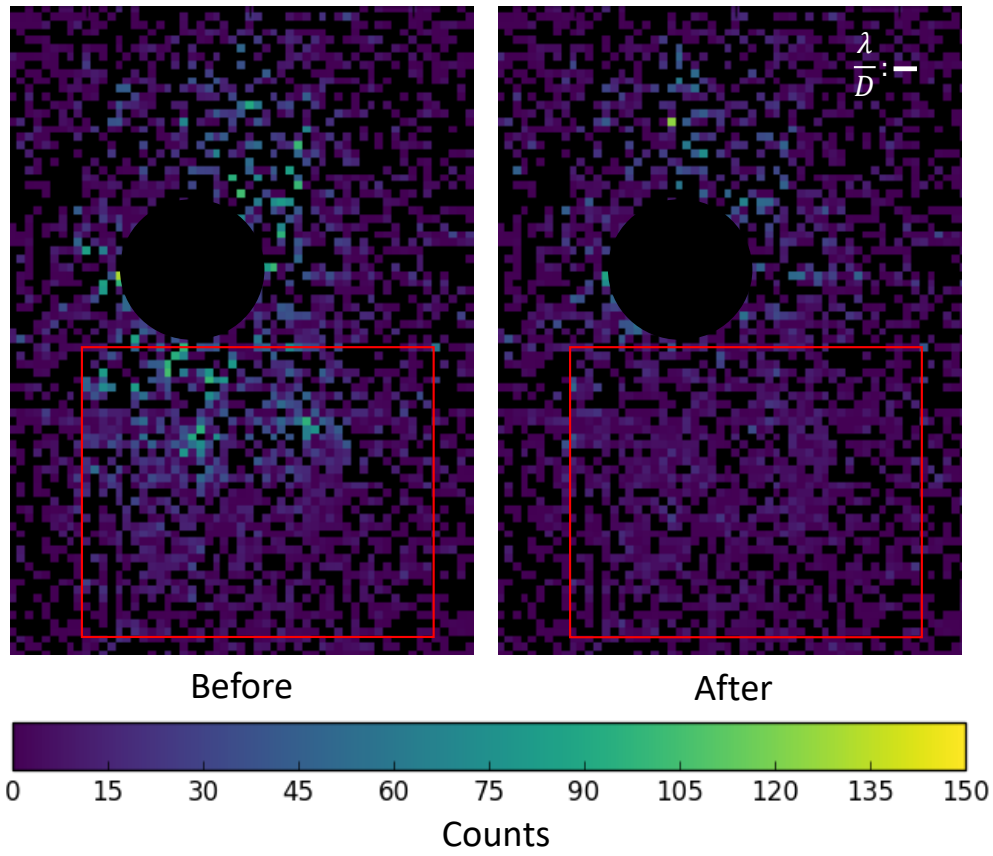


Figure 4.5: DARKNESS focal plane images before (left) and after (right) running the speckle nulling controller. The control region is outlined in red, and the black circle marks the approximate position of the coronagraph.

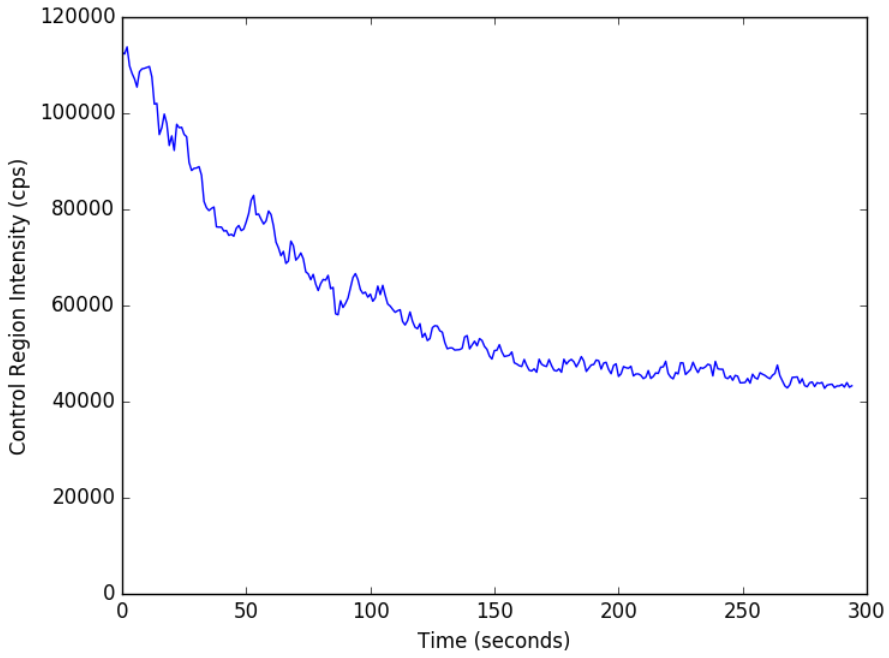


Figure 4.6: Light curve inside the control region over the duration of the speckle nulling loop, corresponding to the DARKNESS test in figure 4.5

this is then added to the current wavefront sensor convergence point (which generally corresponds to a flat wavefront). This will result in the control loop converging to the applied actuator map. It is necessary to apply actuator maps as wavefront sensor offsets while the main AO control loop is running; if a map is simply written directly to the DM, the WFS will measure it as an aberration and the AO loop will remove it. P3K uses a Shack-Hartmann wavefront sensor, so wavefront sensor measurements are encoded as positional "centroid offsets"; i.e. the displacement of each imaged lenslet spot away from its nominal center position [13]. Actuator maps are converted to centroid offsets via matrix multiplication by the system response matrix (which encodes the per actuator response of the wavefront sensor to the DM).

For our tests, we configured SDC with a Lyot coronagraph, and applied a J-band filter in the DARKNESS optical stack. We used a detection/probing exposure time

of 0.1 seconds, for a nominal control bandwidth of 2 Hz (1x detection followed by 4x probe frames gives 0.5 s per iteration). We experimented with faster framerates, but found that this did not improve final contrast achieved or convergence time. The speckle background for our tests was purely quasistatic, resulting from optical aberrations within the instrument. Results are shown in figures 4.5 and 4.6, demonstrating a factor of ~ 2.5 reduction in flux over a ≈ 150 second convergence time.

Tests at Palomar were performed with an older version of the speckle nulling codebase, which can be found at: `https://github.com/neelay893/DarknessSpeckleSuppression`.

4.5.2 MEC

In lab speckle nulling tests with MEC were performed behind SCExAO at Subaru Observatory. Unlike at Palomar, the speckle nulling controller was run directly on the SCExAO realtime controller (RTC), for easy interfacing with the main AO control loop. The raw MKID datastream was forwarded by the MEC data server to the RTC over a 10 Gbit SFP+ link.

Actuator maps can be applied to the SCExAO DM using a CACAO provided shared memory interface; one simply writes the map to the shared memory buffer, then posts a semaphore to trigger a write to the DM. CACAO provides ten such shared memory buffers to accommodate multiple control processes. As with P3K, actuator maps can be written either directly or as wavefront sensor convergence point offsets. This happens entirely within CACAO; one simply toggles an option to convert a particular DM shared memory channel into a convergence point offset.

For these tests, we used a Lyot coronagraph with a 113 mas focal plane mask. Tests were performed using both broadband and narrowband (900 ± 40 nm) internal source configurations. The energy resolution of MEC is currently not high enough to mean-

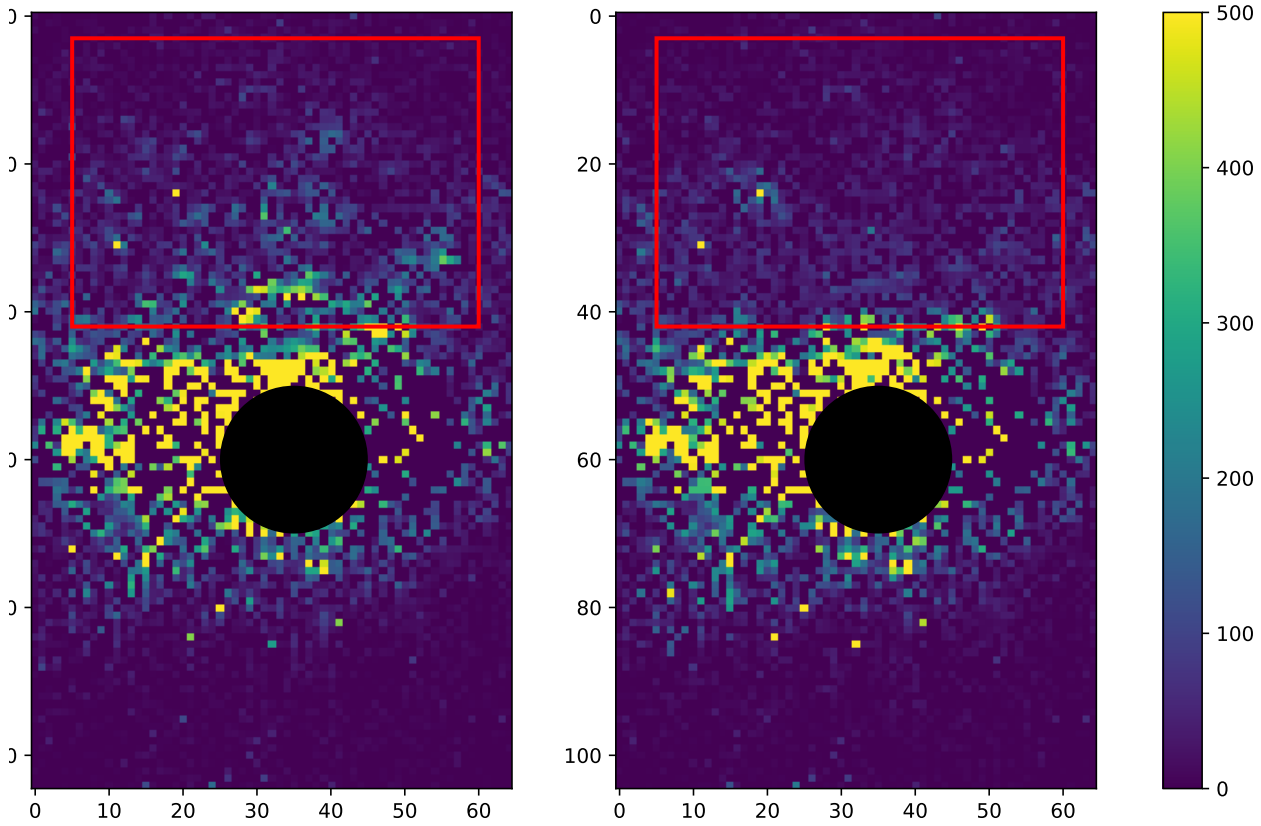


Figure 4.7: MEC focal plane images before (left) and after (right) a quasistatic speckle nulling test using the SCExAO internal white light source. Intensity is in counts/second. For this test, we applied a narrowband filter (900 ± 40 nm), and used 5 ms frames, for a nominal feedback rate of 40 Hz.

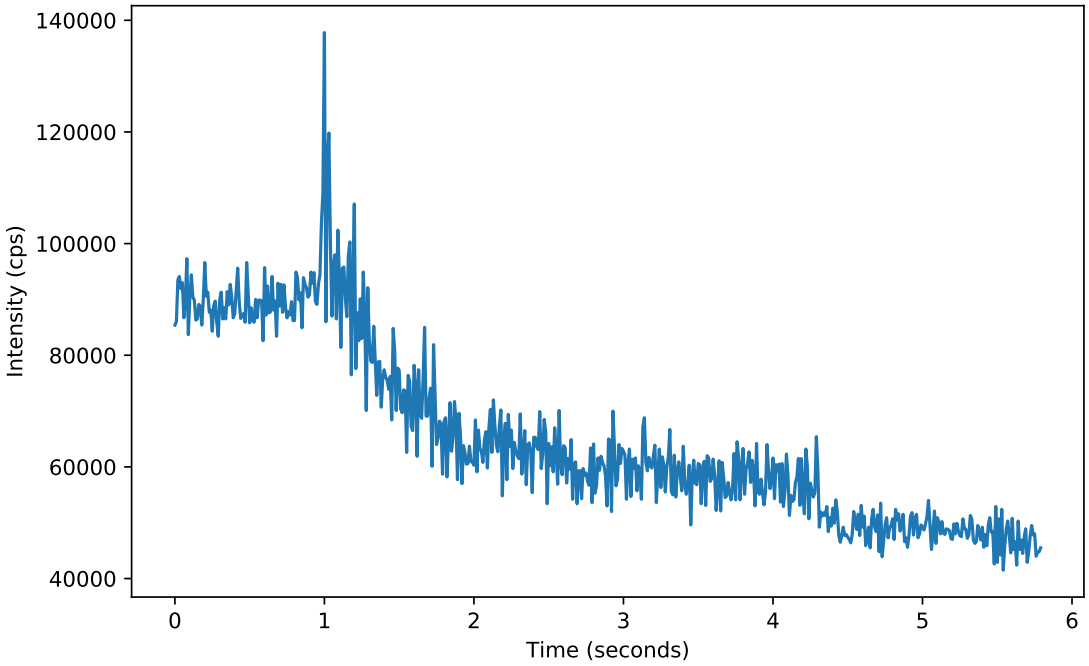


Figure 4.8: Light curve inside the control region corresponding to the test in figure 4.7. The loop was started at $t = 1$ s. The loop fully converged after approximately 1 second, though significant suppression was achieved in ≈ 0.5 s. The controller was turned off at $t \approx 4.3$ s; the drop in intensity occurs because the DM is no longer being modulated by probe speckles.

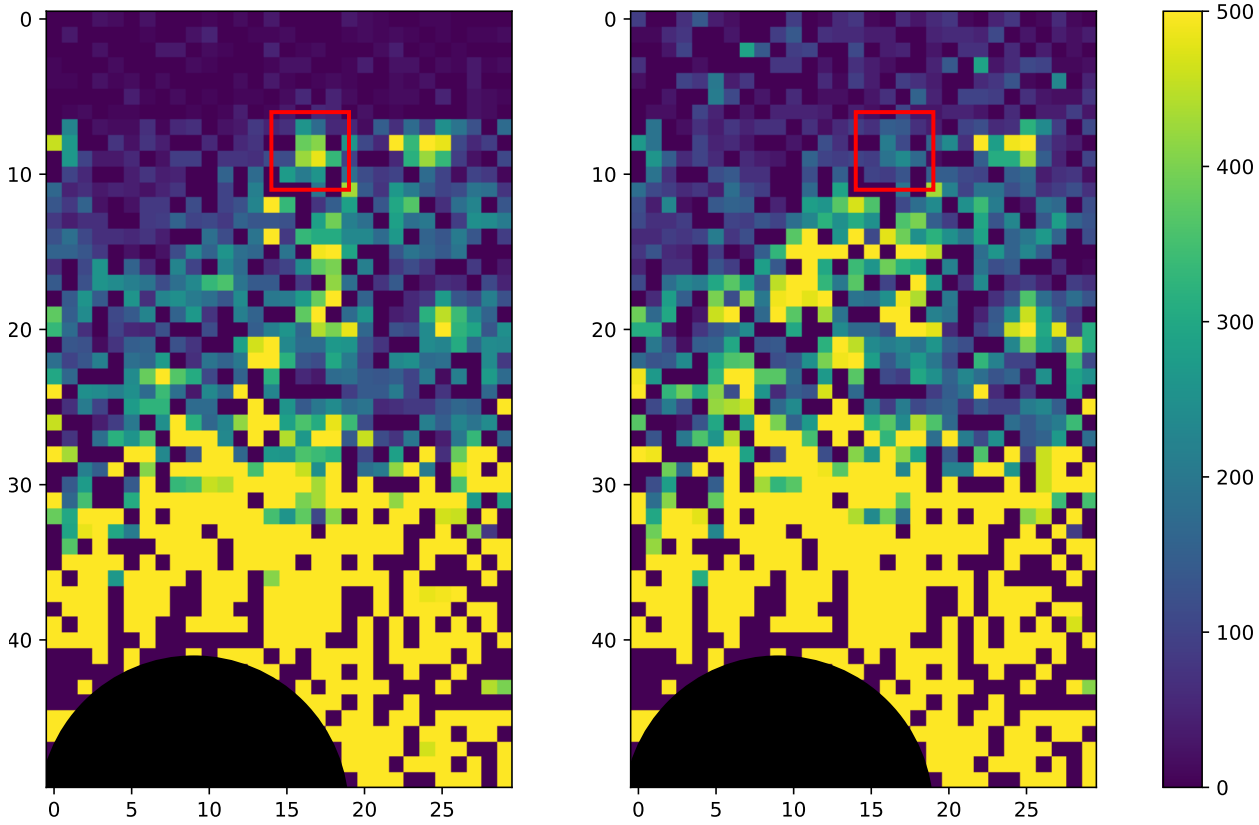


Figure 4.9: Before/after images for a speckle nulling test within a small ($\approx$ 2.5 resolution element) aperture (outlined in red). The aperture was chosen to be small enough to only contain a single speckle, but large enough to provide the algorithm with a nontrivial speckle centroiding/measurement task. These images correspond to the (blue) light curve in figure 4.10; intensity was suppressed by a factor of 2 within 50 ms using 5 ms exposures. As in figure 4.7, these tests use the SCexAO internal source with a narrowband filter applied

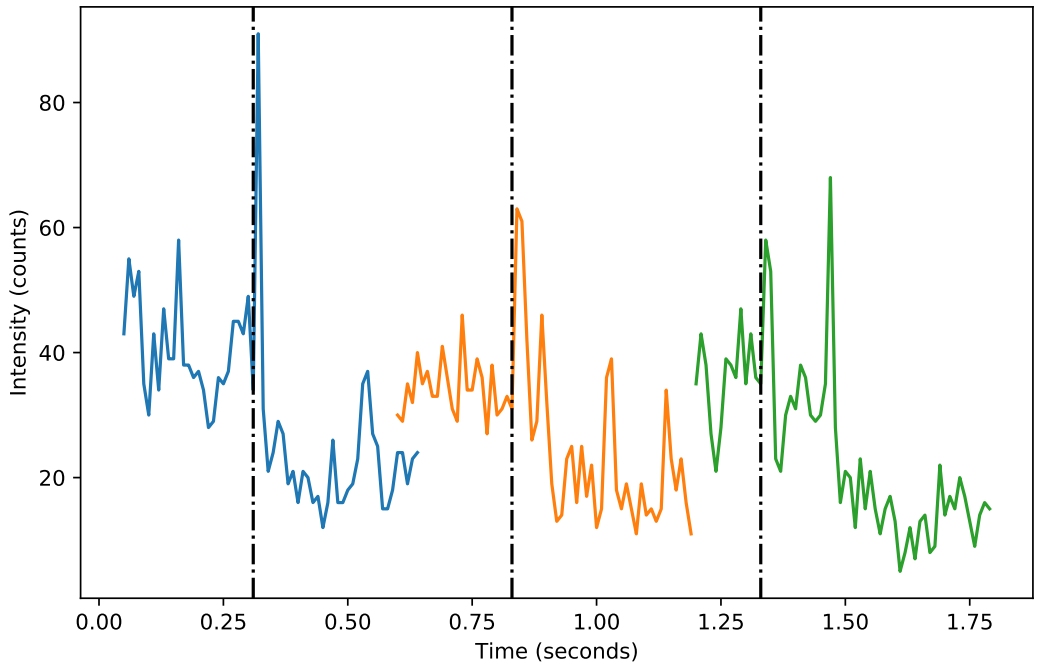


Figure 4.10: Light curves corresponding to three different speckle nulling tests within the aperture (and test configuration) defined in figure 4.9. The black lines mark the start of each test. All three curves show suppression by approximately a factor of two, but differ in convergence time: (blue) shows convergence time $t \leq 50$ ms, (orange) shows $50 < t < 100$ ms, and (green) shows $t \approx 200$ ms.

ingfully discriminate within this wavelength band, so we imposed a wavelength range of < 1200 nm for all MEC images. We experimented with a variety of exposure timescales, but most tests were done using ≈ 5 ms exposures for both speckle detection and probing (for a nominal control bandwidth of 40 Hz). No apparent performance gains (in terms of convergence time and final contrast) were found by using exposures shorter than this.

In figures 4.7 and 4.8, we show results from a narrowband test over a relatively large (55x39-pixel, or 27x20 resolution element) control region, using 5 ms exposures and controlling up to 10 speckles simultaneously. The algorithm reduced total speckle flux in the control region by approximately 1/3 after 0.5 s, and 1/2 after 1 s. In this test, 25% of the control loop runtime is used for computations and data transmission/parsing (≈ 1.2 ms per 5 ms probe frame), so the actual control bandwidth is closer to 30 Hz. For an isolated speckle (figure 4.9), the algorithm will converge in 50-200 ms (figure 4.10). This discrepancy in convergence times can be largely attributed to errors in the initial speckle amplitude and position measurement.

In the broadband regime (figures 4.11 and 4.12), performance is degraded significantly, likely because the speckle chromaticity causes them to appear diffuse and elongated. An MKID array with higher energy resolution ($R \approx 10$) will likely mitigate these effects by allowing us to impose more stringent wavelength cuts and/or correct for the speckle position based on measured photon wavelength.

4.6 On-sky Test

We performed an on-sky test using MEC+SCExAO, on the target HD148112. HD148112 is an A-type star with an apparent magnitude of 4.56 in J-band [52]. For our test, we used wavelengths shorter than J-band (950 nm - 1100 nm). As with our in-lab tests, we imposed a wavelength cut of < 1200 nm on all MEC images. We ran the speckle

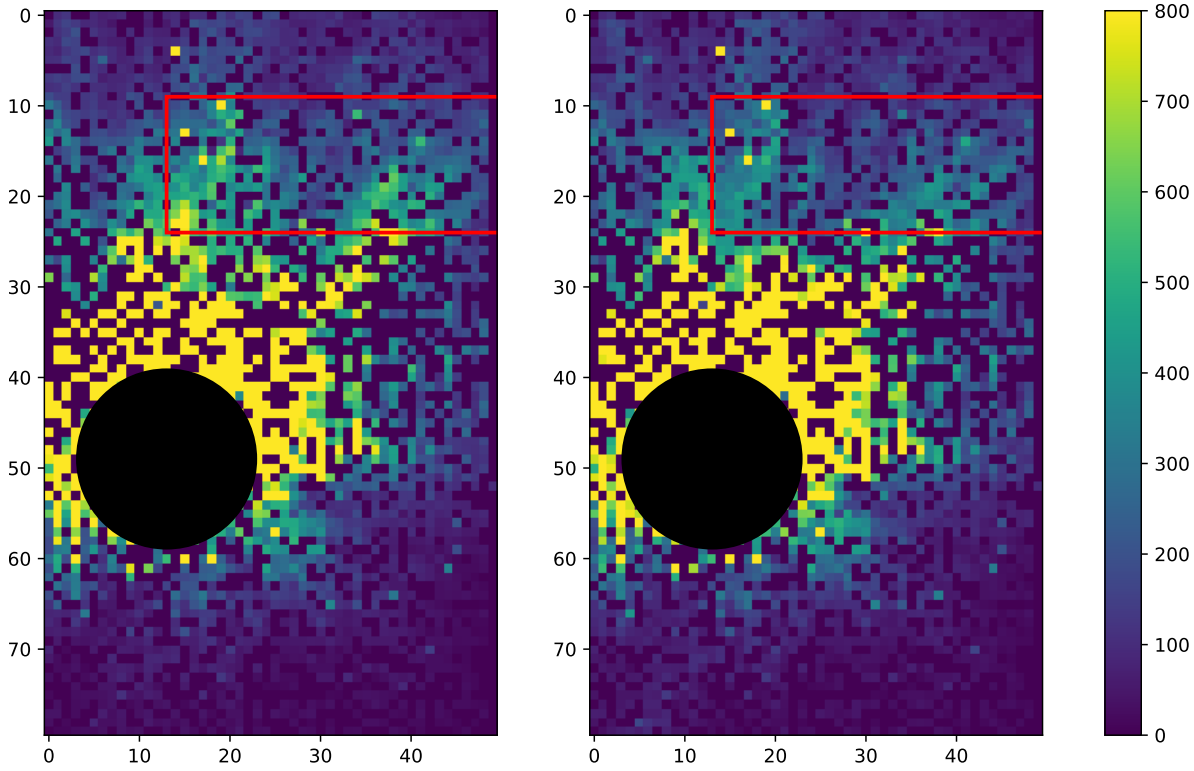


Figure 4.11: Before/after images for an in-lab speckle nulling test in the broadband regime. Speckles appear to extend outward from the PSF center due to chromaticity.

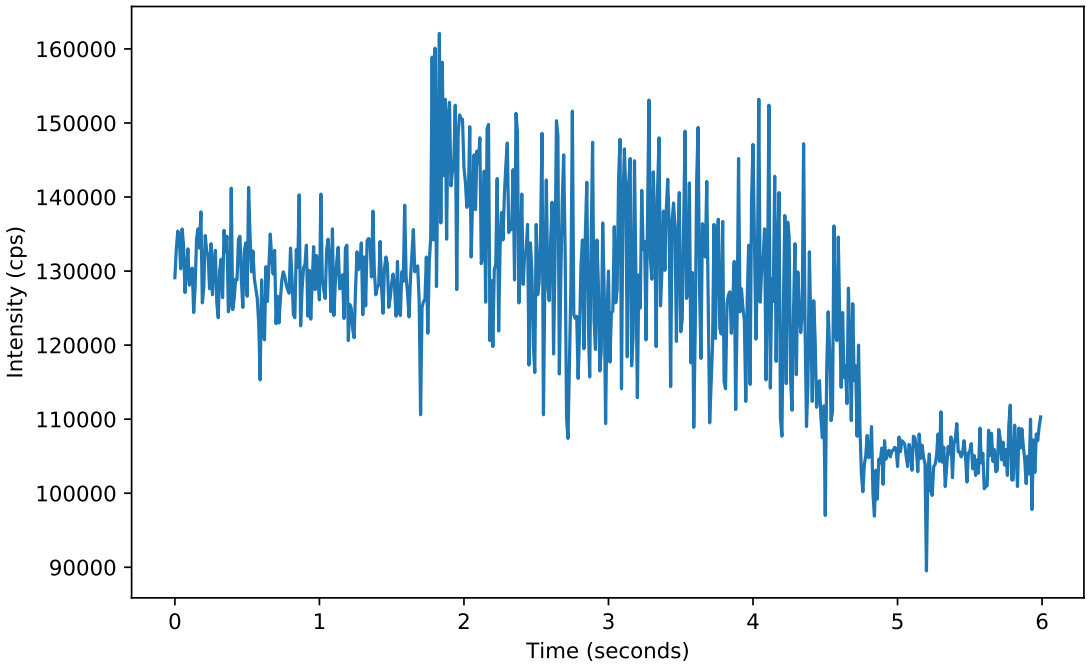


Figure 4.12: Light curve corresponding to the test in figure 4.11. The speckle nulling loop starts at ≈ 1.75 s and ends at 5 s. 5 ms exposures were used for control.

nulling control loop concurrently with the SCExAO wavefront sensor loop, so our DM commands were applied as WFS convergence point offsets. We expermented with both fast feedback rates (exposures ≈ 5 ms), as well as slower feedback rates more suited to quasistatic speckle nulling (exposures ≥ 0.5 s).

In figures 4.13, 4.14, and 4.15, we show results from a 5-iteration test of quasistatic speckle nulling using 0.5 second frames, for a nominal feedback rate of 0.4 Hz. The overall quasistatic speckle background was suppressed by $\approx 25\%$ within the control region (figure 4.15).

Unfortunately, the algorithm did not converge when using fast feedback rates (i.e. exposure times ≈ 5 ms). This is likely because the speckle background was fluctuating too quickly to allow the speckle phase/amplitude to be measured with sufficient SNR for correction. The fast framerate would also makes it more difficult to average down

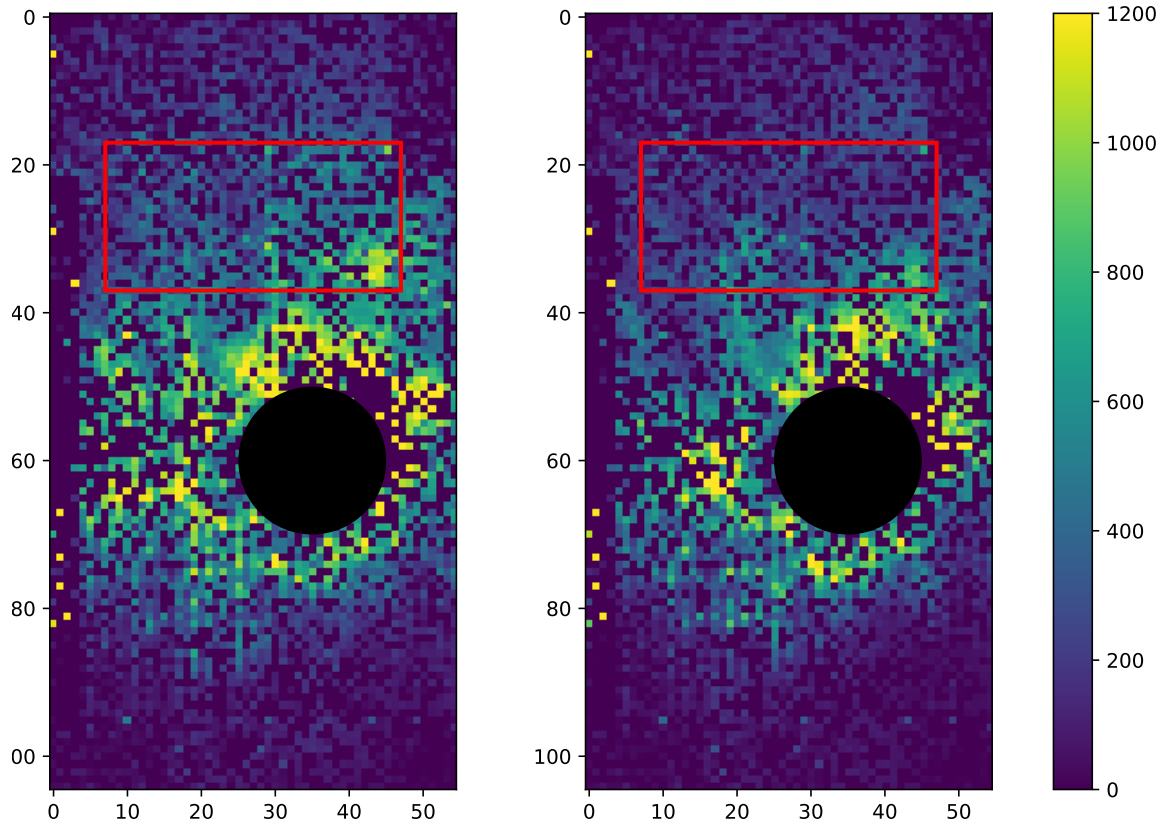


Figure 4.13: MEC focal plane images before (left) and after (right) quasistatic on-sky speckle nulling. Each image is a 0.5 second exposure (taken immediately before/after running the control loop); the intensity is given in counts/second. The control region is outlined in red, and the black circle marks out the approximate location of the coronagraph.

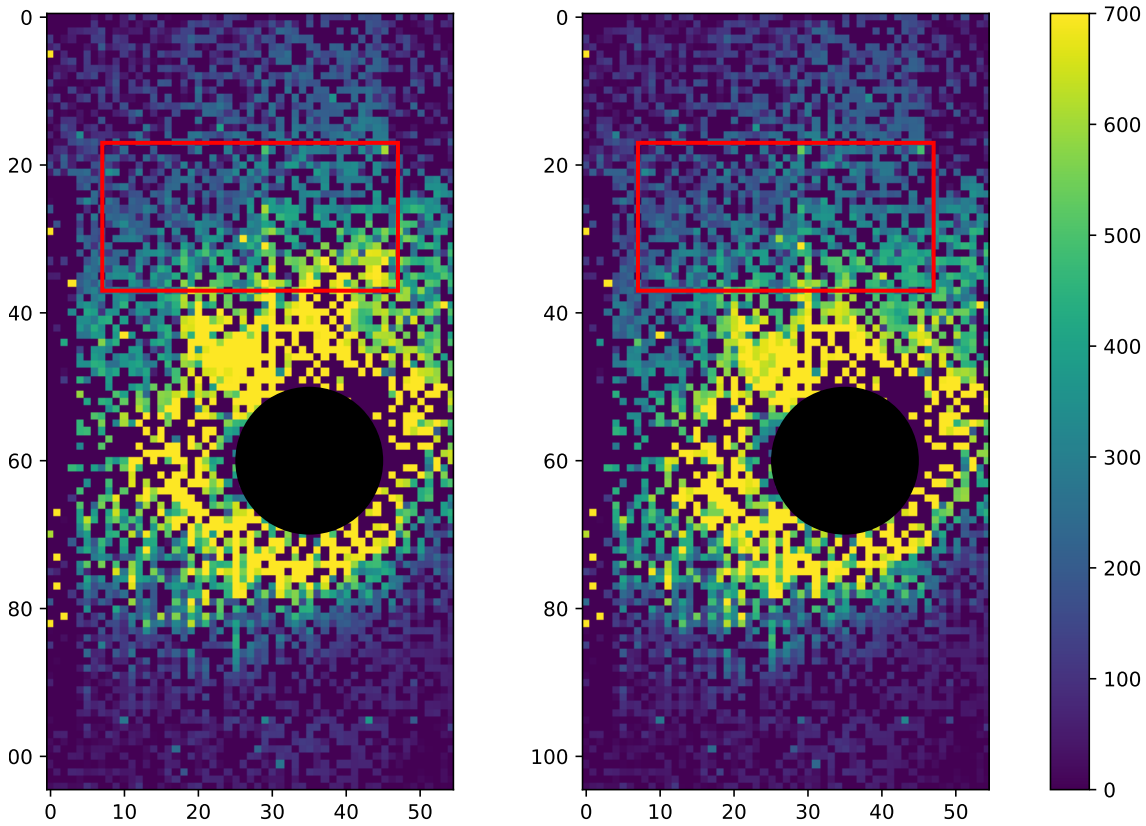


Figure 4.14: Long (10 second) exposure limit of figure 4.13. The control region is outlined in red, and the black circle marks out the approximate location of the coronagraph.

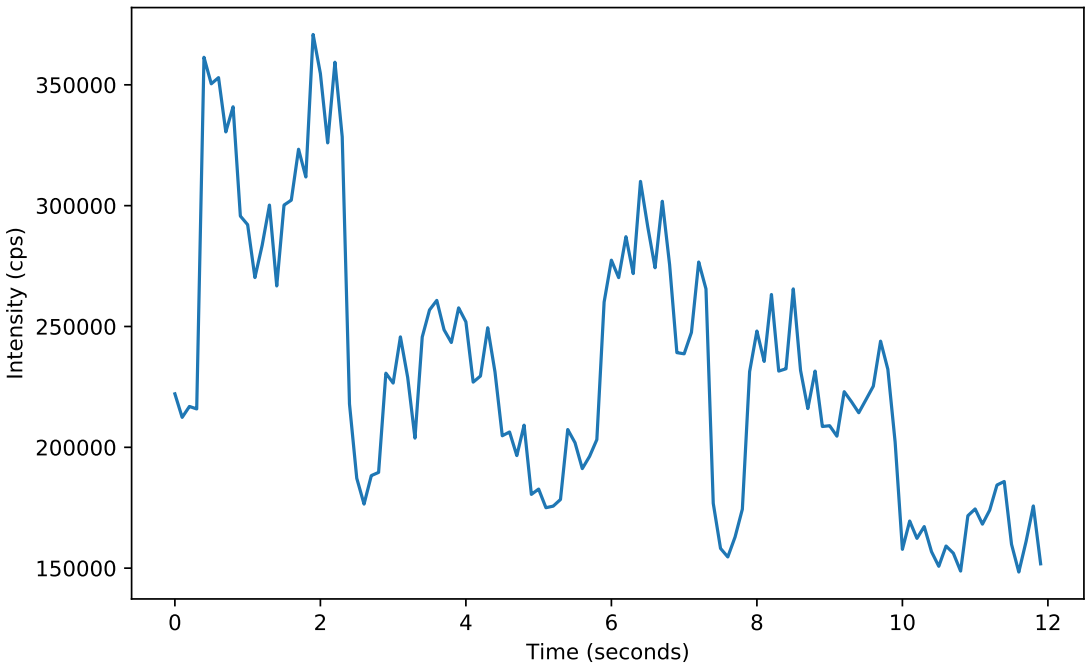


Figure 4.15: Light curve of control region during the speckle nulling loop, made using 0.1 second exposures. The dips in intensity occuring every ≈ 2.5 seconds correspond to detection frames, while the peaks correspond to probes. The loop stops running at $t = 10$ s

fluctuations that are too fast to correct (the way one can in the quasistatic long-exposure limit). Future work is needed to develop more efficient methods for accurately measuring the speckle complex amplitude in the high-framerate limit.

4.7 Preliminary Tests of Expectation Maximization (EM) Algorithm for Calibration

In order to improve the accuracy of speckle probe measurements, we experimented with a calibration method to determine the *per-pixel* (rather than per aperture) complex amplitude response to the DM speckle modes. This could also allow for more precise speckle localization during the probing stage.

We first discretize the space of speckle modes to match the pixel grid; i.e. we define a DM mode control vector $\vec{u}$ such that each element corresponds to the real or imaginary amplitude of a speckle centered around a pixel in the control region. This allows us to specify a list of speckles to apply to the DM in a single control vector. For a vector with M total control modes, we have:

$$\varphi_{DM}(\vec{\rho}) = \sum_{i=1}^{M} u_i cos(\vec{k}_i \cdot \rho) + u_{i+M} sin(\vec{k}_i \cdot \rho) \tag{4.62}$$

where i indexes speckle modes, and the first M elements of $\vec{u}$ specify the real amplitudes, and the subsequent M elements specify imaginary amplitudes. We then define the matrix G, which specifies the per-pixel response in the focal plane to $\vec{u}$. As with $\vec{u}$, for an N pixel control region, the first N rows of G specify real amplitudes, and the next N are imaginary. The per-pixel complex electric field in the focal plane is given by $\vec{x}$; such that the intensity of pixel i is given by $I = \sqrt{x_i^2 + x_{i+N}^2}$. So, if our calibration matrix G has perfect accuracy then applying a control vector $\vec{u}$ would result in $\vec{x} = G\vec{u}$.

Above we have defined G such that it is analogous to the Jacobian control matrix in EFC (electric field conjugation) or CDI (coherent differential imaging) control schemes. This allows us to utilize the expectation maximization (EM) algorithm described in Ref. [53] to refine our estimate of G.

We first take a calibration dataset consisting of a series of random control vectors $\vec{u}_{c,n}$, each followed by a probe sequence of randomly selected speckle modes. We use the same set of probe phases $\phi = [0, \pi/2, \pi, 3\pi/2]$ as speckle nulling. We can define the per-pixel probe measurement vector $\vec{z}$, such that for pixel i, $z_i = I_{i,1} - I_{i,3}$ and $z_{i+N} = I_{i,2} - I_{i,4}$. Here $I_{i,j}$ corresponds to the intensity measured at pixel i and probe phase j. These are analogous to the real and imaginary speckle amplitudes defined in 4.45; we can estimate $x_{i,i+N}$ from these measurements using:

$$x_i = \frac{z_i}{4G_i \vec{u}_{\mathbf{Re}}} \tag{4.63}$$

$$x_{i+N} = \frac{z_{i+N}}{4G_{i+N} \vec{u}_{\mathbf{Im}}} \tag{4.64}$$

where $\vec{u}_{\mathbf{Re}}$ is the speckle mode vector corresponding to the positive real probe ($\phi = 0$), and $\vec{u}_{\mathbf{Im}}$ is the speckle mode vector corresponding to the positive imaginary probe ($\phi = \pi/2$). Probes of opposite phase are found by negating these probe vectors; e.g. the probe vector for $\phi = \pi$ corresponds to $-\vec{u}_{\mathbf{Re}}$.

Following [53], we use a Kalman filter to track the focal plane amplitude estimate $\vec{x}_n$ across each of the iterations n in our calibration dataset. We can then fit the elements of G to minimize the per-iteration Kalman filter measurement and control residuals using a gradient descent algorithm. Our definitions of the Kalman filter and the loss function used to optimize G are identical to those in Ref. [53], substituting in our alternative definitions of the calibration matrix and DM control vector. This optimization process is performed iteratively, alternating between application of the Kalman filter to estimate

$\vec{x}_n$ and running 100-500 gradient descent steps to optimize G.

To test this approach, we took an in-lab calibration dataset using MEC+SCExAO. We used a narrowband configuration (900 ± 40 nm), and 0.1 second exposures over 1000 probe+control cycles. For the fitting procedure, we inititalized G using the standard speckle nulling calibration procedure, estimating the per-pixel response to adjacent speckle modes using a Gaussian function of appropriate width. We ran the optimization algorithm on a per-pixel basis, using approximately 500 iterations per pixel. The results of the optimization for a single pixel are shown in figure 4.16. For this pixel, the algorithm has appeared to have "found" a one-pixel offset error in the initial calibration, as well as the first Airy ring around each speckle (corresponding to the "dips" below zero surrounding each peak). Over the full control region, the optimization algorithm appears to have reduced the per-pixel measurement residual by a factor of 2-5 (figure 4.18), indicating a significant improvement in the overall accuracy of the system calibration. Work is ongoing to integrate this calibration scheme into the speckle nulling controller.

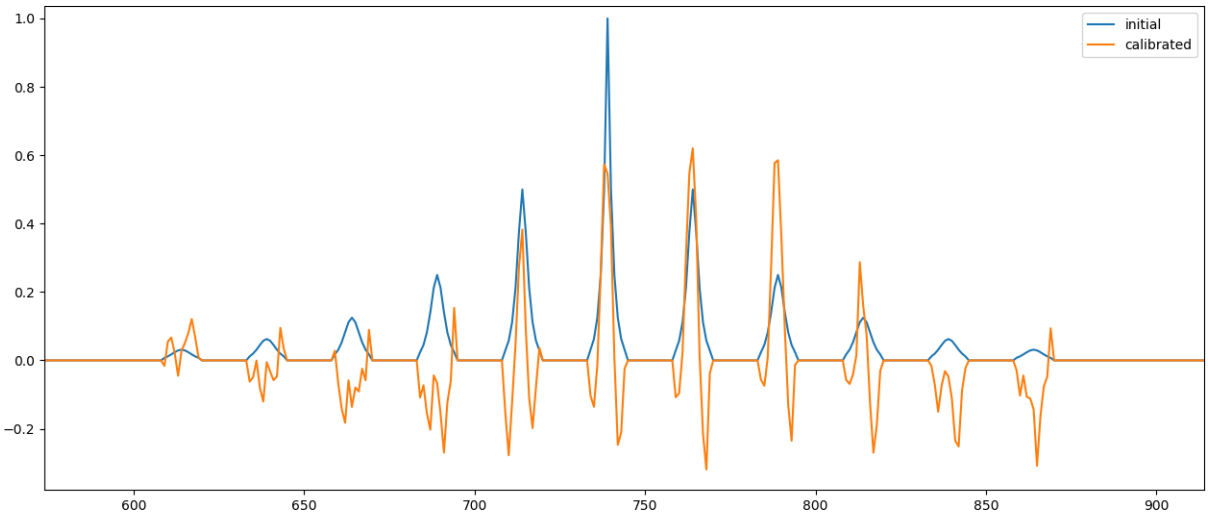


Figure 4.16: G-matrix elements for a single pixel before (blue) and after (orange) running the EM algorithm. Each index along the x-axis corresponds to a different mode in the speckle grid. The grid is flattened in row-major order; horizontally-adjacent modes occur at consecutive indices, while vertically-adjacent modes correspond to adjacent clusters in the above plot.

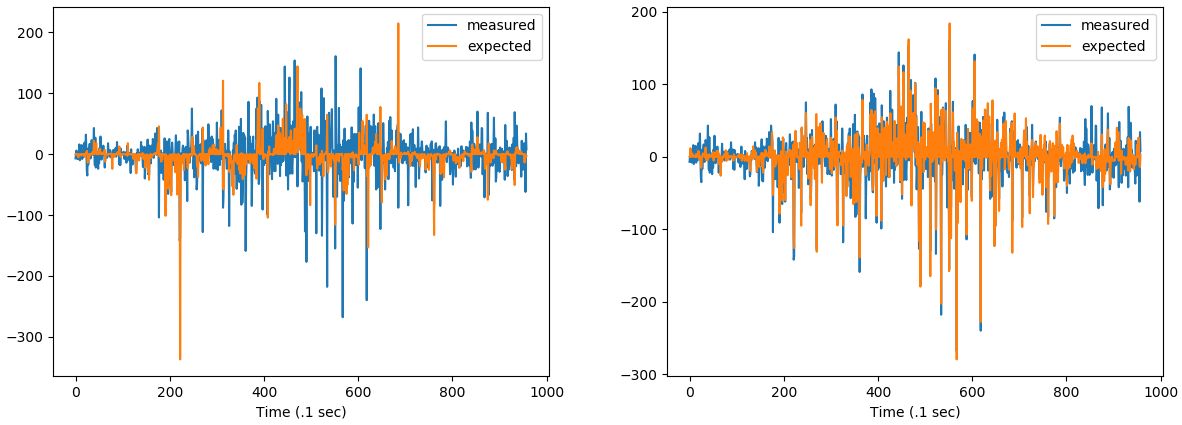


Figure 4.17: Expected and actual probe measurement ($z_{i,n}$) for a single pixel's imaginary probe quadrature, before (left) and after (right) the optimization procedure. We use the same pixel as in figure 4.16. (blue) corresponds to the actual probe measurement, while (orange) shows the expected z_i, given the focal plane electric field estimate $x_{i,n}$, calibration matrix G, and applied probe(s) $\vec{u}_{\mathbf{Im},n}$. Expected and actual probe measurements appear to agree much more closely after the calibration procedure, indicating an improvement in the ability of G to predict the focal plane response to the DM probes.

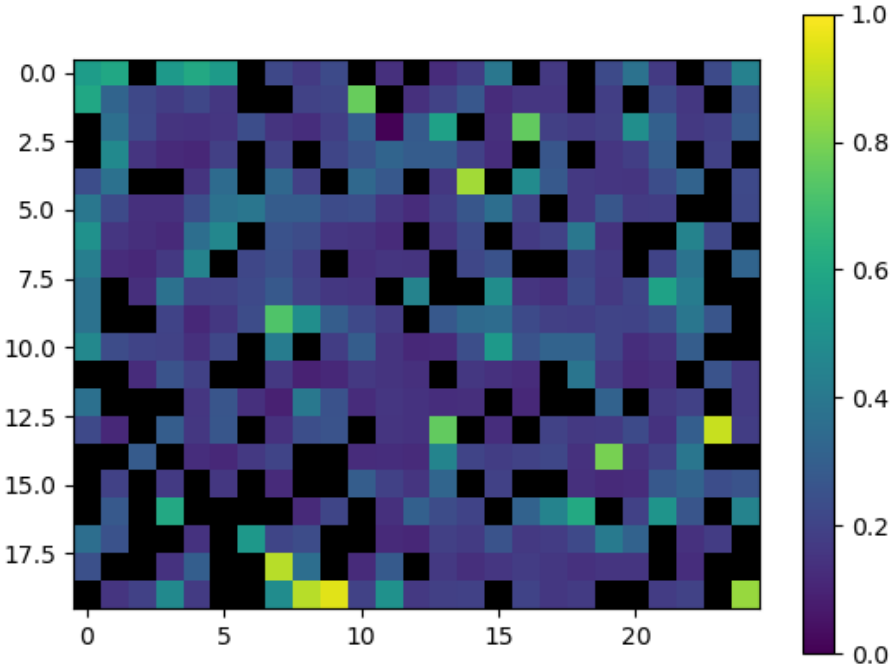


Figure 4.18: Image showing the ratio $r_{opt}/r_{initial}$ of the mean squared probe measurement residual before ($r_{initial}$) and after (r_{opt}) the optimization procedure. For each pixel, r is calculated by averaging the squared difference between the actual and expected probe measurements over the full calibration dataset.

Appendix A

Room Temperature Phase Noise Calculation

The single-sided phase noise spectral density $S_{\delta\phi}$ due to Johnson noise from the HEMT is given by [54]:

$$S_{\delta\phi,HEMT} = \frac{k_B}{P_{device}}(T_{device} + T_{HEMT}) \tag{A.1}$$

To facilitate comparison with loopback data taken at room temperature, we can instead use the tone power at the readout system RF input; $P_{RF} = G_{HEMT}P_{device}$, where G_{HEMT} is the HEMT amplifier gain. So,

$$S_{\delta\phi,HEMT} = \frac{k_B G_{HEMT}}{P_{RF}}(T_{device} + T_{HEMT}) \tag{A.2}$$

The noise from the readout system is dominated by the room temperature amplifier chain immediately following the RF input of the RF/IF board. This chain consists of four HMC3587, each with a noise temperature of 438 K and two programmable attenuators (figure A.1). The total single sided noise spectral density at the end of this chain (IQ

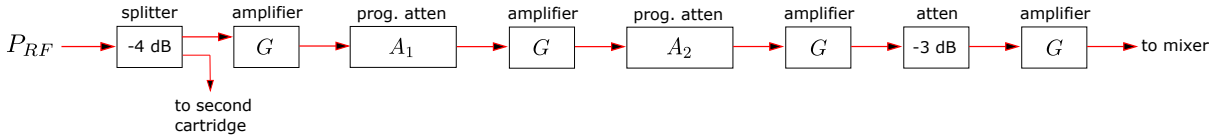


Figure A.1: Schematic of room temperature amplifier/attenuator chain. An RF power splitter is used to send the full $4-8\ GHz$ output comb to both the $4-6\ GHz$ and $6-8\ GHz$ readout units. The amplifier gain G is 15 dB and $0 \leq A_{1,2} \leq 31.75$ dB. In practice, we require $A_{1,2} \leq 15$ dB.

mixer input) is given by:

$$S_{mixer} = G\left\{\frac{G}{2}\left(\frac{G}{A_2}\left[\frac{G}{A_1}\left(S_{RT}+S_{amp}\right)+S_{RT}+S_{amp}\right]+S_{RT}+S_{amp}\right)+S_{RT}+S_{amp}\right\} \tag{A.3}$$

$$= \left(\frac{G^4}{2A_1A_2}+\frac{G^3}{2A_2}+\frac{G^2}{2}+G\right)\left(S_{RT}+S_{amp}\right) \tag{A.4}$$

where $S_{RT,amp} = 2k_bT_{RT,amp}$ is the Johnson noise PSD at room temperature and from the amplifier, respectively, G is the room temperature amplifier gain, $A_{1,2}$ are the programmable attenuator values, and $T_{RT,amp}$ are noise temperatures at room temperature (290 K) and of the amplifiers (438 K).

The tone power at the end of the amplifier chain is given by:

$$P_{mixer} = \frac{G^4}{5A_1A_2}P_{RF} \tag{A.5}$$

To obtain the single quadrature phase noise spectral density, we have:

$$S_{\delta\phi,RT} = \frac{S_{mixer}}{2P_{mixer}} \tag{A.6}$$

$$= \frac{k_B}{P_{RF}} \left(\frac{5}{2} + \frac{5A_1}{2G} + \frac{5A_1A_2}{2G^2} + \frac{5A_1A_2}{G^3} \right) (T_{RT} + T_{amp}) \tag{A.7}$$

Comparing the HEMT and room temperature terms:

$$\frac{S_{\delta\phi,RT}}{S_{\delta\phi,HEMT}} = \left(\frac{5}{2} + \frac{5A_1}{2G} + \frac{5A_1A_2}{2G^2} + \frac{5A_1A_2}{G^3} \right) \frac{T_{RT} + T_{amp}}{G_{HEMT}(T_{device} + T_{HEMT})} \tag{A.8}$$

In normal operation we require $A_{1,2} \leq G$. So we can set $A_{1,2} = G$ to determine an upper bound on $S_{\delta\phi,RT}$:

$$\frac{S_{\delta\phi,RT}}{S_{\delta\phi,HEMT}} \leq \left(\frac{15}{2} + \frac{5}{G} \right) \frac{T_{RT} + T_{amp}}{G_{HEMT}(T_{device} + T_{HEMT})} \tag{A.9}$$

For $T_{RT} = 290\ K$, $T_{amp} = 438\ K$, $T_{device} = 0.1\ K$, $T_{HEMT} = 2.3\ K$, $G_{HEMT} = 40\ dB$, and $G = 15\ dB$, $\frac{S_{\delta\phi,RT}}{S_{\delta\phi,HEMT}} \leq -6.3\ dB$.

Appendix B

Correcting for Line Noise

According to the optimal filtering formalism, the pulse amplitude estimator variance is given by [55]:

$$\sigma^2 = \left(\sum_k \frac{|\hat{s}(f_k)|^2|}{J(f_k)} \Delta f \right)^{-1} \tag{B.1}$$

where k indexes frequency, Δf is the frequency spacing, $\hat{s}(f_k)$ is the DFT (discrete Fourier transform) of the pulse template, and $J(f_k)$ is the noise PSD.

Consider some $J(f_k)$ with significant spectral lines. We can remove these lines – denote this $J'(f_k)$ – then compute a new pulse height variance $\sigma^2[J'(f_k)]$, where we've written σ^2 as a function of $J(f_k)$. Define:

$$C = \frac{\sigma^2[J(f_k)]}{\sigma^2[J'(f_k)]} \tag{B.2}$$

Since σ^2 scales proportionally with $J(f_k)$,

$$\sigma^2[J(f_k)] = C\sigma^2[J'(f_k)] = \sigma^2[CJ'(f_k)] \tag{B.3}$$

$CJ'(f_k)$ is a "clean" spectrum with the same σ^2 as $J(f_k)$, so we define the noise floor of

$CJ'(f_k)$ as the "true" channel noise floor after accounting for spectral lines. Since $J'(f_k)$ has the same noise floor as $J(f_k)$, we can perform this correction by scaling the fitted noise floor of $J(f_k)$ by C.

To compute $J'(f_k)$, we set all spectral lines that are 3 dB above the fitted noise floor to the local spectral floor. This correction is performed over 500 Hz to 300 kHz band; it does not account for the $1/f$ "knee" present in some channels. The IQ low pass filter rolls off transmission significantly past 300 kHz, so it is not necessary to remove lines in this region.

In order to keep this calculation consistent with the optimal filtering scheme used in firmware, we calculate σ^2 directly from the optimal filter and pulse template. In the frequency domain, the optimal filter is given by:

$$\hat{\phi}(f_k) = \frac{\hat{s}^*(f_k)}{J(f_k)} \tag{B.4}$$

Substituting eqn. B.4 into B.1, we have:

$$\sigma^2 = \left(\sum_k \hat{\phi}(f_k)\hat{s}(f_k)\Delta f \right)^{-1} \tag{B.5}$$

We can convert this to the time domain using the convolution theorem (up to a normalization constant, since we are only interested in the ratio $\sigma^2[J(f_k)]/\sigma^2[J'(f_k)]$):

$$\sigma^2 = \left(\sum_k \hat{\phi}(f_k)\hat{s}(f_k)e^{if_k t}\Delta f \right)^{-1}\Bigg|_{t=0} \tag{B.6}$$

$$= iFFT[\hat{\phi}(f_k)\hat{s}(f_k)](t)\Big|_{t=0} \tag{B.7}$$

$$= [\phi * s](t)|_{t=0} \tag{B.8}$$

$$\tag{B.9}$$

where $*$ denotes the convolution operator. We use equation B.9 to compute all σ^2, using a 50 coefficient optimal filter and representative pulse template (figure 2.5). To avoid potential periodicity artifacts inherent to DFT, optimal filters are computed in the time domain [40].

Appendix C

Noise Contributed by Sideband Tones

Consider a sideband reflection falling within 100 kHz (inside the channel bandwidth) of some resonator tone. The power contributed by the sideband reflection to the resonator tone phase power spectrum (in units dBc) is given by:

$$P_{SB,\delta\phi} = \frac{1}{2}\frac{P_{SB}}{P_{tone}} \tag{C.1}$$

where P_{tone} resonator tone power and P_{SB} is the sideband reflection power. The corresponding RMS amplitude (in radians) is given by:

$$A_{SB,\delta\phi} = \sqrt{P_{SB,\delta\phi}} = \sqrt{\frac{P_{SB}}{2P_{tone}}} \tag{C.2}$$

For $\frac{P_{SB}}{P_{tone}} = -30\ dB$, $A_{SB,\delta\phi} = 1.3°$. This is below the RMS phase noise for a typical resonator channel; the channel in figure 2.7 has $A_{\delta\phi,RMS} \approx 2°$. Depending on the location of the sideband reflection relative to the in-band tone, this can likely be further reduced

by optimal filtering.

Bibliography

[1] C. Marois, B. Macintosh, T. Barman, B. Zuckerman, I. Song, J. Patience, D. Lafrenière, and R. Doyon, *Direct imaging of multiple planets orbiting the star hr 8799*, *science* **322** (2008), no. 5906 1348–1352.

[2] B. P. Bowler, *Imaging extrasolar giant planets*, *Publications of the Astronomical Society of the Pacific* **128** (2016), no. 968 102001.

[3] T. Currie, O. Guyon, J. Lozi, *et. al.*, *Performance and early science with the subaru coronagraphic extreme adaptive optics project*, in *Techniques and Instrumentation for Detection of Exoplanets IX*, vol. 11117, p. 111170X, International Society for Optics and Photonics, 2019.

[4] B. C. Platt and R. Shack, *History and principles of shack-hartmann wavefront sensing*, 2001.

[5] R. Ragazzoni, *Pupil plane wavefront sensing with an oscillating prism*, *Journal of modern optics* **43** (1996), no. 2 289–293.

[6] S. Esposito and A. Riccardi, *Pyramid wavefront sensor behavior in partial correction adaptive optic systems*, *Astronomy & Astrophysics* **369** (2001), no. 2 L9–L12.

[7] O. Guyon, *Limits of adaptive optics for high-contrast imaging*, *The Astrophysical Journal* **629** (2005), no. 1 592.

[8] B. Macintosh, J. Graham, D. Palmer, R. Doyon, D. Gavel, J. Larkin, B. Oppenheimer, L. Saddlemyer, J. K. Wallace, B. Bauman, *et. al.*, *The gemini planet imager*, in *Advances in Adaptive Optics II*, vol. 6272, p. 62720L, International Society for Optics and Photonics, 2006.

[9] R. Dekany, J. Roberts, R. Burruss, A. Bouchez, T. Truong, C. Baranec, S. Guiwits, D. Hale, J. Angione, T. Trinh, *et. al.*, *Palm-3000: exoplanet adaptive optics for the 5 m hale telescope*, *The Astrophysical Journal* **776** (2013), no. 2 130.

[10] O. Guyon, C. Roddier, J. E. Graves, F. Roddier, S. Cuevas, C. Espejo, S. Gonzalez, A. Martinez, G. Bisiacchi, and V. Vuntesmeri, *The nulling stellar*

coronagraph: Laboratory tests and performance evaluation, *Publications of the Astronomical Society of the Pacific* **111** (1999), no. 764 1321.

[11] S. B. Goebel, O. Guyon, D. N. Hall, and N. Jovanovic, *The scexao nir speckle lifetime experiment*, in *5th Adaptive Optics for Extremely Large Telescopes, AO4ELT 2017*, 2017.

[12] A. Give'on, *A unified formalism for high contrast imaging correction algorithms*, in *Proc. SPIE*, vol. 7440, p. 74400D, 2009.

[13] M. Bottom, B. Femenia, E. Huby, D. Mawet, R. Dekany, J. Milburn, and E. Serabyn, *Speckle nulling wavefront control for palomar and keck*, in *Adaptive Optics Systems V*, vol. 9909, p. 990955, International Society for Optics and Photonics, 2016.

[14] P. J. Bordé and W. A. Traub, *High-contrast imaging from space: speckle nulling in a low-aberration regime*, *The Astrophysical Journal* **638** (2006), no. 1 488.

[15] F. Martinache, O. Guyon, N. Jovanovic, C. Clergeon, G. Singh, T. Kudo, T. Currie, C. Thalmann, M. McElwain, and M. Tamura, *On-sky speckle nulling demonstration at small angular separation with scexao*, *Publications of the Astronomical Society of the Pacific* **126** (2014), no. 940 565.

[16] P. Day, H. Leduc, B. Mazin, A. Vayonakis, and J. Zmuidzinas, *A broadband superconducting detector suitable for use in large arrays*, *Nature* **425** (10, 2003) 817–21.

[17] P. Szypryt, S. R. Meeker, G. Coiffard, N. Fruitwala, B. Bumble, G. Ulbricht, A. B. Walter, M. Daal, C. Bockstiegel, G. Collura, N. Zobrist, I. Lipartito, and B. A. Mazin, *Large-format platinum silicide microwave kinetic inductance detectors for optical to near-ir astronomy*, *Opt. Express* **25** (Oct, 2017) 25894.

[18] B. Mazin, P. Day, J. Zmuidzinas, and H. LeDuc, *Multiplexable kinetic inductance detectors*, in *AIP Conference Proceedings*, vol. 605, pp. 309–312, American Institute of Physics, 2002.

[19] B. A. Mazin, S. R. Meeker, M. J. Strader, P. Szypryt, D. Marsden, J. C. van Eyken, G. E. Duggan, A. B. Walter, G. Ulbricht, M. Johnson, B. Bumble, K. O'Brien, and C. Stoughton, *ARCONS: A 2024 pixel optical through near-IR cryogenic imaging spectrophotometer*, *Publ. Astron. Soc. Pac.* **125** (2013), no. 933 1348–1361.

[20] S. R. Meeker, B. A. Mazin, A. B. Walter, P. Strader, N. Fruitwala, C. Bockstiegel, P. Szypryt, G. Ulbricht, G. Coiffard, B. Bumble, G. Cancelo, T. Zmuda, K. Treptow, N. Wilcer, G. Collura, R. Dodkins, I. Lipartito, N. Zobrist,

M. Bottom, J. C. Shelton, D. Mawet, J. C. van Eyken, G. Vasisht, and E. Serabyn, *DARKNESS: A microwave kinetic inductance detector integral field spectrograph for high-contrast astronomy*, *Publ. Astron. Soc. Pac.* **130** (Jun, 2018) 065001.

[21] A. B. Walter, N. Fruitwala, S. Steiger, J. I. Bailey III, N. Zobrist, N. Swimmer, I. Lipartito, J. P. Smith, S. R. Meeker, C. Bockstiegel, *et. al.*, *The mkid exoplanet camera for subaru scexao*, *Publications of the Astronomical Society of the Pacific* **132** (2020), no. 1018 125005.

[22] M. J. Strader, A. M. Archibald, S. R. Meeker, P. Szypryt, A. B. Walter, J. C. van Eyken, G. Ulbricht, C. Stoughton, B. Bumble, D. L. Kaplan, and B. A. Mazin, *Search for optical pulsations in PSR J0337+1715*, *Monthly Notices of the Royal Astronomical Society* **459** (03, 2016) 427–430, [https://academic.oup.com/mnras/article-pdf/459/1/427/8115535/stw663.pdf].

[23] P. Szypryt, G. E. Duggan, B. A. Mazin, S. R. Meeker, M. J. Strader, J. C. van Eyken, D. Marsden, K. O'Brien, A. B. Walter, G. Ulbricht, T. A. Prince, C. Stoughton, and B. Bumble, *Direct detection of SDSS J0926+3624 orbital expansion with ARCONS*, *Monthly Notices of the Royal Astronomical Society* **439** (02, 2014) 2765–2770, [https://academic.oup.com/mnras/article-pdf/439/3/2765/3850573/stu137.pdf].

[24] N. Swimmer, B. A. Mazin, C. Bockstiegel, J. I. Bailey III, G. Coiffard, M. Daal, K. Davis, N. Fruitwala, I. Lipartito, J. Smith, *et. al.*, *The picture-c mkid camera*, in *Ground-based and Airborne Instrumentation for Astronomy VIII*, vol. 11447, p. 114479B, International Society for Optics and Photonics, 2020.

[25] S. McHugh, B. A. Mazin, B. Serfass, S. Meeker, K. O'Brien, R. Duan, R. Raffanti, and D. Werthimer, *A readout for large arrays of microwave kinetic inductance detectors*, *Review of Scientific Instruments* **83** (Apr, 2012) 044702.

[26] L. Swenson, P. Day, B. Eom, H. Leduc, N. Llombart, C. McKenney, O. Noroozian, and J. Zmuidzinas, *Operation of a titanium nitride superconducting microresonator detector in the nonlinear regime*, *Journal of Applied Physics* **113** (2013), no. 10 104501.

[27] N. Zobrist, B. H. Eom, P. Day, B. A. Mazin, S. R. Meeker, B. Bumble, H. G. LeDuc, G. Coiffard, P. Szypryt, N. Fruitwala, I. Lipartito, and C. Bockstiegel, *Wide-band parametric amplifier readout and resolution of optical microwave kinetic inductance detectors*, *Applied Physics Letters* **115** (2019), no. 4 042601, [https://doi.org/10.1063/1.5098469].

[28] R. Dodkins, S. Mahashabde, K. O'Brien, N. Thatte, N. Fruitwala, A. B. Walter, S. Meeker, P. Szypryt, and B. Mazin, *MKID digital readout tuning with deep learning*, *Astronomy and Computing* **23** (2018) 60–71.

[29] S. Steiger, T. Currie, T. D. Brandt, O. Guyon, M. Kuzuhara, J. Chilcote, T. D. Groff, J. Lozi, A. B. Walter, N. Fruitwala, *et. al.*, *Scexao/mec and charis discovery of a low-mass, 6 au separation companion to hip 109427 using stochastic speckle discrimination and high-contrast spectroscopy*, *The Astronomical Journal* **162** (2021), no. 2 44.

[30] A. B. Walter, C. Bockstiegel, T. D. Brandt, and B. A. Mazin, *Stochastic speckle discrimination with time-tagged photon lists: Digging below the speckle noise floor*, *Publications of the Astronomical Society of the Pacific* **131** (2019), no. 1005 114506.

[31] M. Bottom, J. C. Shelton, J. K. Wallace, R. Bartos, J. Kuhn, D. Mawet, B. Mennesson, R. Burruss, and E. Serabyn, *Stellar double coronagraph: a multistage coronagraphic platform at palomar observatory*, *Publications of the Astronomical Society of the Pacific* **128** (2016), no. 965 Art–No.

[32] E. Cady, C. Baranec, C. Beichman, D. Brenner, R. Burruss, J. Crepp, R. Dekany, D. Hale, L. Hillenbrand, S. Hinkley, *et. al.*, *Electric field conjugation with the project 1640 coronagraph*, *arXiv preprint arXiv:1309.6357* (2013).

[33] N. Jovanovic, F. Martinache, O. Guyon, C. Clergeon, G. Singh, T. Kudo, V. Garrel, K. Newman, D. Doughty, J. Lozi, *et. al.*, *The subaru coronagraphic extreme adaptive optics system: Enabling high-contrast imaging on solar-system scales*, *Publications of the Astronomical Society of the Pacific* **127** (2015), no. 955 890.

[34] O. Guyon, A. Sevin, F. Ferreira, H. Ltaief, J. Males, V. Deo, D. Gratadour, S. Cetre, F. Martinache, J. Lozi, *et. al.*, *Adaptive optics real-time control with the compute and control for adaptive optics (cacao) software framework*, in *Adaptive Optics Systems VII*, vol. 11448, p. 114482N, International Society for Optics and Photonics, 2020.

[35] N. Fruitwala, P. Strader, G. Cancelo, T. Zmuda, K. Treptow, N. Wilcer, C. Stoughton, A. B. Walter, N. Zobrist, G. Collura, *et. al.*, *Second generation readout for large format photon counting microwave kinetic inductance detectors*, *Review of Scientific Instruments* **91** (2020), no. 12 124705.

[36] N. Fruitwala, A. B. Walter, J. I. Bailey, R. Dodkins, and B. A. Mazin, *End-to-end deep learning pipeline for microwave kinetic inductance detector resonator identification and tuning*, *Journal of Astronomical Telescopes, Instruments, and Systems* **7** (2021), no. 2 028003.

[37] D. Werthimer, *The casper collaboration for high-performance open source digital radio astronomy instrumentation*, in *2011 XXXth URSI general assembly and scientific symposium*, pp. 1–4, IEEE, 2011.

[38] J. Hickish, Z. Abdurashidova, Z. Ali, K. D. Buch, S. C. Chaudhari, H. Chen, M. Dexter, R. S. Domagalski, J. Ford, G. Foster, *et. al.*, *A decade of developing radio-astronomy instrumentation using casper open-source technology*, *Journal of Astronomical Instrumentation* **5** (2016), no. 04 1641001.

[39] S. H. Moseley, R. L. Kelley, R. J. Schoelkopf, A. E. Szymkowiak, D. McCammon, and J. Zhang, *Advances toward high spectral resolution quantum x-ray calorimetry*, *IEEE Transactions on Nuclear Science* **35** (1988), no. 1 59–64.

[40] B. K. Alpert, R. D. Horansky, D. A. Bennett, W. B. Doriese, J. W. Fowler, A. S. Hoover, M. W. Rabin, and J. N. Ullom, *Note: Operation of gamma-ray microcalorimeters at elevated count rates using filters with constraints*, *Review of Scientific Instruments* **84** (2013), no. 5 056107, [https://doi.org/10.1063/1.4806802].

[41] J. Gao, J. Zmuidzinas, B. A. Mazin, H. G. LeDuc, and P. K. Day, *Noise properties of superconducting coplanar waveguide microwave resonators*, *Applied Physics Letters* **90** (2007), no. 10 102507.

[42] J. Gao, M. Daal, A. Vayonakis, S. Kumar, J. Zmuidzinas, B. Sadoulet, B. A. Mazin, P. K. Day, and H. G. Leduc, *Experimental evidence for a surface distribution of two-level systems in superconducting lithographed microwave resonators*, *Appl. Phys. Lett.* **92** (2008), no. 15 152505.

[43] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, *SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python*, *Nature Methods* **17** (2020) 261–272.

[44] Y. LeCun, Y. Bengio, and G. Hinton, *Deep learning*, *nature* **521** (2015), no. 7553 436–444.

[45] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, *Tensorflow: A system for large-scale machine learning*, in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, (Savannah, GA), pp. 265–283, USENIX Association, Nov., 2016.

[46] A. Krizhevsky, I. Sutskever, and G. E. Hinton, *Imagenet classification with deep convolutional neural networks*, *Communications of the ACM* **60** (2017), no. 6 84–90.

[47] S. Ioffe and C. Szegedy, *Batch normalization: Accelerating deep network training by reducing internal covariate shift*, *arXiv preprint arXiv:1502.03167* (2015).

[48] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, *Dropout: a simple way to prevent neural networks from overfitting*, *The journal of machine learning research* **15** (2014), no. 1 1929–1958.

[49] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings* (Y. Bengio and Y. LeCun, eds.), 2015.

[50] A. B. Walter, *MEC: The MKID exoplanet camera for high speed focal plane control at the subaru telescope*. PhD thesis, 2019.

[51] M. Bottom, L. S. Neat, L. K. Harding, P. Morrissey, S. R. Meeker, and R. T. Demers, *Smartphone scene generator for efficient characterization of visible imaging detectors*, in *High Energy, Optical, and Infrared Detectors for Astronomy VIII*, vol. 10709, p. 107092R, International Society for Optics and Photonics, 2018.

[52] J. R. Ducati, *VizieR Online Data Catalog: Catalogue of Stellar Photometry in Johnson's 11-color system.*, *VizieR Online Data Catalog* (Jan., 2002).

[53] H. Sun, N. J. Kasdin, and R. Vanderbei, *Identification and adaptive control of a high-contrast focal plane wavefront correction system*, *Journal of Astronomical Telescopes, Instruments, and Systems* **4** (2018), no. 4 049006.

[54] J. Zmuidzinas, *Superconducting microresonators: Physics and applications*, *Annual Review of Condensed Matter Physics* **3** (02, 2012) 169–214.

[55] S. R. Golwala, *Exclusion limits on the WIMP nucleon elastic scattering cross-section from the Cryogenic Dark Matter Search*. PhD thesis, UC, Berkeley, 2000.