ABSTRACT

Title of Dissertation: WAVEFRONT SHAPING IN A COMPLEX REVERBERANT CAVITY WITH A BINARY TUNABLE METASURFACE

Benjamin W. Frazier
Doctor of Philosophy, 2021

Dissertation Directed by: Professor Thomas M. Antonsen, Jr.
Department of Electrical and
Computer Engineering
and
Professor Steven M. Anlage
Department of Physics

Electromagnetic environments are becoming increasingly complex and congested, creating a growing challenge for systems that rely on electromagnetic waves for communication, sensing, or imaging. The use of intelligent, reconfigurable metasurfaces provides a potential means for achieving a radio environment that is capable of directing propagating waves to optimize wireless channels on-demand, ensuring reliable operation and protecting sensitive electronic components. The capability to isolate or reject unwanted signals in order to mitigate vulnerabilities is critical for any practical application.

In the first part of this dissertation, I describe the use of a binary programmable metasurface to (i) control the spatial degrees of freedom for waves propagating inside an electromagnetic cavity and demonstrate the ability to create nulls in the transmission coefficient between selected ports; and (ii) create the conditions for

coherent perfect absorption. Both objectives are performed at arbitrary frequencies. In the first case a novel and effective stochastic optimization algorithm is presented that selectively generates coldspots over a single frequency band or simultaneously over multiple frequency bands. I show that this algorithm is successful with multiple input port configurations and varying optimization bandwidths. In the second case I establish how this technique can be used to establish a multi-port coherent perfect absorption state for the cavity.

In the second part of this dissertation, I introduce a technique that combines a deep learning network with a binary programmable metasurface to shape waves in complex electromagnetic environments, in particular ones where there is no direct line-of-sight. I applied this technique for wavefront reconstruction and accurately determined metasurface configurations based on measured system scattering responses in a chaotic microwave cavity. The state of the metasurface that realizes desired electromagnetic wave field distribution properties was successfully determined even in cases previously unseen by the deep learning algorithm. My technique is enabled by the reverberant nature of the cavity, and is effective with a metasurface that covers only $\sim$1.5% of the total cavity surface area.

WAVEFRONT SHAPING IN A COMPLEX REVERBERANT CAVITY WITH A BINARY TUNABLE METASURFACE

by

Benjamin West Frazier

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2021

Advisory Committee:
Professor Thomas M. Antonsen Jr., Chair/Advisor
Professor Steven M. Anlage, Co-Advisor
Professor Edward Ott
Professor Sennur Ulukus
Professor Michelle Girvan

Preface

This dissertation is long overdue. I started down the path of electromagnetics, specifically adaptive optics, as an undergraduate student at the University of North Carolina at Charlotte (UNCC) back in 1999 under the instruction of Bob Tyson. My career as an electro-optical engineer began at Xinetics, Inc. in Devens, MA in 2002, where I characterized deformable mirrors and active optical components. I had completed a Master's degree and expected that I would return to school to pursue a Ph.D. at some point in the future. I chose an occupational path instead of an academic one however, and spent over a decade in the field of adaptive optics with a specialty in beam control for high energy lasers. I joined Adaptive Optics Associates in 2005 and worked out of offices in Wichita, KS, Lancaster, CA, and East Hartford, CT until 2013. I then moved to the Johns Hopkins University Applied Physics Laboratory (JHU/APL) in Laurel, MD, where my career expanded to include radio frequency (RF) and intelligence, surveillance, and reconnaissance (ISR) systems. Finally, in 2015, under the encouragement of my wife (then fiance), I returned to the University of Maryland (UMD) as a part time graduate student.

At UMD, I took the electrophysics/applied electromagnetics route. I joined the Wave Chaos group in 2017, researching electromagnetic wave propagation in chaotic microwave cavities. When the COVID19 pandemic hit in early 2020, the labs at UMD were closed and experimental work ground to a halt. Professor Anlage

was able to get a binary programmable metasurface through our Office of Naval Research (ONR) contacts (and my JHU/APL colleagues). My immediate thought was "adaptive optics with microwaves", and I was hooked. I built a smaller scale cavity (< 1 m^3 volume) in my basement and proceeded with experimentation. The chaotic nature of the environment required a control approach more complex than found in traditional adaptive optics systems. This worked well as a research project and culminated in 2 publications that form the basis of this dissertation: 1) Frazier, Antonsen, Anlage, and Ott. "Wavefront Shaping with a Tunable Metasurface: Creating Cold Spots and Coherent Perfect Absorption at Arbitrary Frequencies", *Physical Review Research*, 2:043422, 2020; and 2) Frazier, Antonsen, Anlage, and Ott, "Deep Learning Estimation of Complex Reverberant Wave Fields with a Programmable Metasurface", *arXiv*, 2103.13500 [physics.app-ph], 2021.

Ben Frazier,

Ellicott City, MD, October, 2021

Dedication

To my father, Richard T. Frazier, whom I still miss profoundly.

Acknowledgments

First and foremost, I am exceptionally grateful to Professors Thomas M. Antonsen and Steven M. Anlage for guiding my research over the last several years and presenting me with interesting and challenging projects. They were always willing to offer their time whenever I needed help or advice. I am also indebted to Professor Edward Ott for providing valuable insight along the way, and would like to thank both the UMD Wave Chaos group and the UNM Extreme Electromagnetics group for allowing me the opportunity to present my work at various stages and receive feedback. I have learned a tremendous amount and will always be appreciative of the experience. Funding for this research was provided through AFOSR COE Grant FA9550-15-1-0171 and ONR Grant N000141912481.

Bob Tyson will always be thanked for starting me on the path to adaptive optics (and in general, electromagnetics) at UNC Charlotte all those years ago. This kick started my career and has been an area of great interest for me ever since. I am further indebted to him for bringing me on as a co-author for the "Field Guide to Adaptive Optics" SPIE books and the upcoming "Principles of Adaptive Optics, 5th Edition" book through Taylor & Francis/CRC Press (currently scheduled for publication in January 2022).

I am also grateful to my colleagues at JHU/APL, who were patient and supportive (most of the time) as I pursued my educational goals. In particular I would

like to thank Eric Rose, Ray Sova, Coire Maranzano, Cris Fernandes, Andy Newman, and Matt Zuber for support and encouragement. Zerotti Woods and Chad Hawes were also instrumental in teaching me about deep learning.

Many additional people have helped shaped my career both academically and professionally, too many to adequately thank each and every one. A roughly chronological (but certainly not exhaustive) list of additional people with significant impact is: Mark Ealey, Jeff Cavaco, Mike and Jackie Roche, Harold Schall, Paul Shattuck, Frank Lewis, John Hernandez, Steve Headrick, Nick Hilton, Gino Delizo, Lawton Lee, Terry Brennan, Greg Luther, Sue Lathan, Mike Reilly, Jack Keane, Erik Johnson, Bob Cameron, Kem White, Eric Fisher, and Jeff Barton.

Finally, I would never have made it without the support of my amazing family: my wonderful wife Jana, step son Max, and son Luke. I owe you guys everything.

Table of Contents

List of Tables

List of Figures

Listings

List of Abbreviations

AFOSR	Air Force Office of Scientific Research
ANN	Artificial Neural Network
CDF	Cumulative Distribution Function
CNN	Convolutional Neural Network
CPA	Coherent Perfect Absorption
DOF	Degrees of Freedom
DQN	Deep Q Network
EMI	Electromagnetic Interference
FFT	Fast Fourier Transform
GOE	Gaussian Orthogonal Ensemble
HPM	High Power Microwave
IFFT	Inverse Fast Fourier Transform
IFT	Inverse Fourier Transform
JHU/APL	Johns Hopkins University Applied Physics Laboratory
LOS	Line-of-Sight
MAE	Mean Absolute Error
MEMS	Micro-Electrical-Mechanical System
ML	Machine Learning
MSE	Mean Squared Error
ONR	Office of Naval Research
PA	Perfect Absorption

PDF	Probability Density Function
PDP	Power Delay Profile
PMN	Lead Magnesium Niobate
PNA	Performance Network Analyzer
RCM	Random Coupling Model
RCS	Radar Cross Section
ReLU	Rectified Linear Unit
RF	Radio Frequency
RMT	Random Matrix Theory
RNN	Recurrent Neural Network
SAR	Synthetic Aperture Radar
SDR	Software Defined Radio
SGD	Stochastic Gradient Descent
SLM	Spatial Light Modulator
SPGD	Stochastic Parallel Gradient Descent
SRE	Smart Radio Environment
SWAP-C	Size, Weight, Power, and Cost
TRS	Time Reversal Symmetry
UMD	University of Maryland
UNM	University of New Mexico
UWB	Ultra Wide Band

List of Symbols

α	Loss parameter	unitless
γ	Learning Rate	unitless
λ	Wavelength	m
λ_s	Scattering matrix eigenvalues	unitless
ν	Slope of power delay profile	dB/s
ω	Angular temporal frequency	rad/s
ρ_c	Pearson correlation coefficient	unitless
τ_c	Cavity time constant (reverberation time)	s
$\delta\mathbf{a}$	Metasurface command perturbation vector	controller units
ξ	Universal (fluctuating) component of impedance	unitless
$\mathcal{E}$	Environment	arbitrary
Δf	Cavity mean mode spacing	Hz
Δk_n^2	Cavity eigenvalue spacing	$(\text{rad/s})^2$
ΔP_2	Difference in average power metric	dB
ΔS_{21}	Normalized difference in reflection coefficient	unitless
$\Delta\phi$	Phase Sweep Difference	deg
$\mathbf{a}$	Metasurface command vector	controller units
$\mathbf{a}^*$	Trial metasurface command vector	controller units
c	Speed of Light (2.997×10^8)	m/s
f	Temporal frequency	Hz
i	Iteration number	unitless
j	Complex number, $\sqrt{-1}$	unitless
k	Wave number/spatial frequency	rad/m
n	Element number	unitless
A_{CPA}	Amplitude for the CPA State eigenvector	W
$\mathbf{B}$	Geometry matrix	arbitary
J	Cost function	arbitrary
L	Loss function	unitless
M	Number of modes/number of elements to toggle	unitless
N	Number of points	unitless

P_{out}	Power emerging from the cavity	W
P_{in}	Power injected into the cavity	W
Q	Cavity quality factor	unitless
$\mathbf{R}$	Reconstruction matrix	arbitrary
S	Scattering parameters	V/V
S^{cav}	Cavity scattering parameters	V/V
S^{rad}	Radiation (free-space) scattering parameters	V/V
T	Convergence criteria	unitless
V	Cavity volume	m^3
$\mathbf{Z}^{\text{avg}}$	Average impedance	Ω
$\mathbf{Z}^{\text{cav}}$	Cavity impedance	Ω
$\mathbf{Z}^{\text{rad}}$	Radiation (free-space) impedance	Ω

Chapter 1: Introduction

1.1 Motivation

Reverberant and highly scattering environments scramble electromagnetic waves, producing interference among the multiple paths between source and receiver. The resulting spatio-temporal fluctuations can seriously degrade imaging, sensing, and communication systems at microwave and optical wavelengths, disrupting operation or even damaging sensitive components. Large enclosed spaces, such as offices or compartments on ships or aircraft can act as "chaotic" reverberating chambers for short-wavelength electromagnetic waves [1]. Additional emissions in these environments, whether from unintentional coupling between components or from an intentional electromagnetic attack, can have serious consequences. Some platforms, such as aircraft and spacecraft, can experience devastating consequences, resulting in mission failure or even casualties [2].

Future smart radio environments (SREs) are envisioned to handle such dynamic conditions, adapting on-the-fly to optimize a given wireless channel through a spatial light modulator (SLM) [3–5]. Intelligently controlling wave fields in the presence of multi-path reflections is therefore a critical factor for enabling SREs. In addition, an intelligent and self-adaptive approach will benefit applications such

as micromanipulation of objects in complex scattering environments [6], and time reversal mirrors that can selectively focus a wavefront or enhance communication system performance [7,8]. A necessary step along this path is to identify approaches for wavefront reconstruction, i.e., determining the configuration of the SLM that accurately produces a given scattering response, that work in reverberant scattering environments.

In optics, SLMs have been used to control waves under strong scattering conditions for some time. Applications range from focusing through general disordered media [9,10] to sophisticated biomedical imaging instruments that fall under the umbrella of adaptive optics [11, 12]. In the last several years, SLMs in the form of programmable metasurfaces have also become widely available at radio frequency (RF) wavelengths [13–16]. Programmable electromagnetic metasurfaces are metamaterial sheets that can modify their local surface impedance over unit cells (meta-atoms) and have emerged as powerful tools for shaping waves inside complex microwave cavities [17–25], increasing the available degrees of freedom (DOF) by manipulating boundary conditions.

Wavefront shaping techniques with metasurfaces have been well studied; however, control in complex environments still relies on simple, online brute force optimization methods. While these approaches work, they require a large number of iterations to reach convergence, are not guaranteed to achieve a global minimum, and can produce undesirable scattering configurations through the intermediate steps of the optimization process. Our approach, as shown in Fig. 1.1 and described in the remainder of this thesis, leverages a deep learning network to enable wave-

front reconstruction and control in complex and highly reverberant environment. The metasurface is placed in a reverberant scattering environment, with a signal injected at Port 1 and the resulting field measured at a specific point of interest (Port 2). The environment is defined by irregular walls and inclusions and is probed by waves with wavelengths much smaller than the characteristic dimension of the enclosure.

The reverberating nature of the environment provides capabilities not present in non-reverberant environments. The requirement for establishing direct LOS is removed, which allows the location of the metasurface to be arbitrarily chosen. In addition, the ability to modify the wave field distribution at arbitrarily chosen regions is enhanced, which allows a relatively small metasurface to be used. In our configuration, the metasurface covers only $\sim$1.5% of the total surface area of the cavity. Establishing a viable method to control a metasurface in a reverberant environment will unlock novel applications and encourage research in new and under-explored domains.

One such unexplored area is coherent perfection absorption (CPA), where coherent excitation of a lossy system can result in complete absorption of all incident waves [26, 27]. It has applications in highly efficient notch filtering, energy conversion, and even detection; since the CPA state is extremely sensitive to parameter variation, it can be used to identify small changes in a complex scattering system [28]. The ability of a metasurface to manipulate additional DOF presents a novel capability for realizing CPA states.

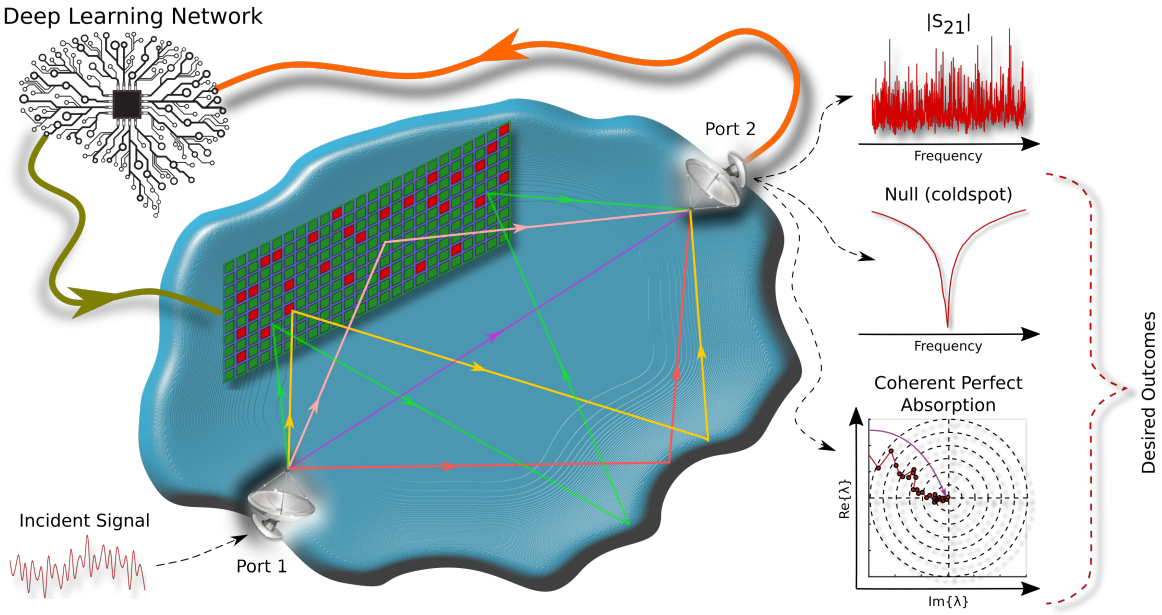

Figure 1.1: **Conceptual view of a deep learning enabled programmable metasurface in a complex electromagnetic environment.** Constructive and destructive interference between multiple propagation paths in a reverberating environment induces randomness in the scattering parameters and scrambles electromagnetic waves that are injected at Port 1. A reconfigurable metasurface is used to tune the interference to create cold spots for protection of sensitive electronic components, realize coherent perfect absorption states for long range wireless power transfer, or unscramble the output fields to enable smart radio environments. The metasurface, along with a sensing antenna at Port 2, is coupled with a deep learning network that provides control. Measurements are used as training data, enabling the network to determine the control settings of the metasurface, and allowing the system to adapt to changing environmental conditions on-the-fly. The metasurface is shown here as large relative to the cavity. In our configuration however, the metasurface is much smaller, covering only $\sim$1.5% of the total surface area of the cavity.

1.2 Outline

The rest of the dissertation is organized as follows. Chapter 2 presents relevant background information for metasurfaces, wavefront reconstruction, complex reverberant scattering environemnts, the random coupling model (RCM), and deep learning. Chapter 3 discusses the stochastic optimization approach for creating coldspots and coherent perfect absorption states. Much of this chapter was published in Frazier, Antonsen, Anlage, and Ott, "Wavefront Shaping with a Tunable Metasurface: Creating Cold Spots and Coherent Perfect Absorption at Arbitrary Frequencies", *Physical Review Research* 2:043422, 2020 [24]. Chapter 4 presents the deep learning approach, most of which was published in Frazier, Antonsen, Anlage, and Ott, "Deep Learning Estimation of Complex Reverberant Wave Fields with a Programmable Metasurface", *arXiv* , 2103.13500 [physics.app-ph], 2021 [29]. Finally, Chapter 5 provides a summary and discussion of the impacts and potential future research directions.

Chapter 2: Background

2.1 Metasurfaces

A "meta"-material is an engineered material that has characteristics not found in naturally occuring materials. A metasurfaces is then a 2-D sheet of metamaterials that is capable of changing its boundary conditions over local regions. Because the incident, reflected, and transmitted electromagnetic waves must have a continuous spatial phase profile across the boundary, metasurfaces need to approximate a continuous phase distribution. This requires the elements to be subwavelength [14]. Smaller dimensions relative to the wavelength provide better approximation [13,30], but are more difficult to produce. In addition, the subwavelength nature can reduce the effective number of DOF, as elements may need to operate in groups rather than individually. Due to the circuit design complexity, many devices only provide binary control over the local spatial phase at each element (relative phase shifts of π radians); however recent advances in fabrication have led to devices with 2-bit control (relative phase shifts of $\pi/2$ radians) [31–34]. Higher fidelity in the realizable phase profile provides additional flexibility for shaping fields, but increases the device complexity in terms of size, weight, power, and cost (SWAP-C). In communications, metasurfaces have been used to generate optical beams with orbital angular momen-

tum that have reduced sensitivity to turbulence [35–38], and for spatial phase coding with visible light [39]. At radio frequencies, metasurfaces have been demonstrated to work as a transmitter for quadrature amplitude modulation [40, 41], simplifying the processing design and providing the potential to enable beyond 5G capabilities. In addition, they have been used to control nonlinear reflections to harness higher order harmonics for spectrum conversion [42], and improving communications using backscatter of WiFi signals [43].

Computational imaging has seen a tremendous boost from the use of metasurfaces [44]. Applications include digital holography at microwave frequencies [45–47], aperture generation for synthetic aperture radar (SAR) [48–50], imaging through walls [51], in cavities [52], and in large scale apertures for human sized objects [53].

Enhanced diversity with the increased number of available DOF is what makes many of these concepts possible, so there has also been significant work in exploring what types of diversity can be leveraged. This includes exploiting spatial and temporal DOF [17, 19, 20, 54, 55], transforming a regular cavity into a chaotic one [21], performing electronic mode stirring [56–58], cancelling electromagnetic fields at a point [18], realizing a high Q open cavity [59], radar cross section (RCS) reduction [60–62], and cloaking objects [63].

Metasurfaces are not limited to shaping only electromagnetic waves. In seismology, control over surface acoustic waves has been demonstrated using metasurfaces made of elastic metamaterials for Love waves (horizontally polarized) [64] and Rayleigh waves (containing both longitudinal and transverse motion) [65]. In the case of quantum waves, a metasurface created from an array of trapped neutral

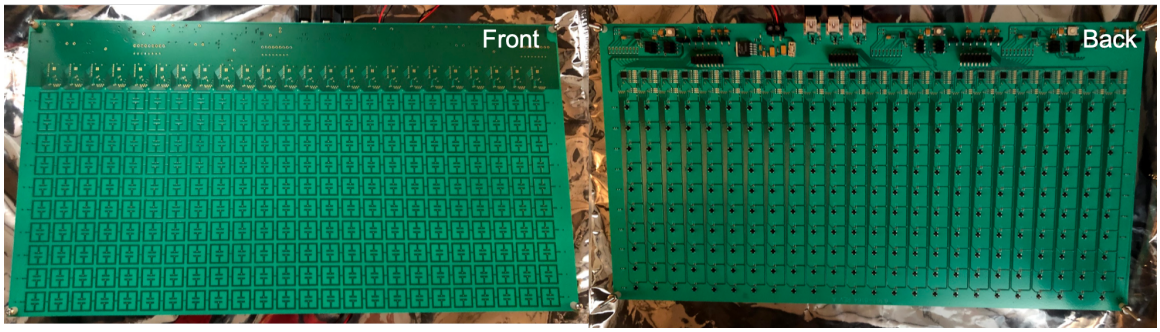

Figure 2.1: **Reflectarray device used for the research.** The left-hand side shows the front of the board with the individual unit cells visible. The right-hand side shows the back of the board with the GaAs FET amplifiers and control electronics visible.

atoms was used to manipulate light at the quantum level [66]. While the underlying physics of these metasurfaces is vastly different, the overall operation and process of wave interaction is essentially the same, implying that strategies for wavefront shaping in one domain can be readily adapted to another.

The metasurface used for this dissertation is a reflectarray fabricated by the Johns Hopkins University Applied Physics Laboratory (JHU/APL) that is designed to operate from 3-3.75 GHz and is shown in Figure 2.1. It has a lattice of 10×24 squares occupied by unit cells with size $< \lambda/6$ [30], where λ is the wavelength. These 240 unit cells can be independently set to one of two states, which approximate electric or magnetic boundary conditions and provide a relative 180° phase shift for waves reflected by the element. This results in the local surface impedance of the array varying from cell to cell and state to state. The array surface thus has 2^{240} (1.8×10^{72}) independent states, each of which reflects waves in a uniquely different set of directions.

2.2 Wavefront Reconstruction and Control

Wavefront control has a rich history and has been well studied from the perspective of adaptive optics. Conceived by Horace Babcock in 1953 [67], and realized in the 1970's [68], adaptive optics provides a method of correcting wavefront aberrations induced by propagation through random media. It has been successfully used in a diverse array of applications including astronomical imaging [69], biomedical imaging [70], high energy laser propagation [71], free space optical communications [72], quantum networking with satellites [73], and laser processing of materials [74].

Adaptive optics systems typically employ reflective deformable mirrors that provide mechanical phase compensation [75, 76]. Conventional deformable mirrors are built with ferroelectric actuators [77] with as small as 5 mm spacing, though monolithic deformable mirrors manufactured from a block of lead magnesium niobate (PMN) material can have 1 mm spacing between actuators [78]. Micro-electrical-mechanical-system (MEMS) mirrors using electrostatic actuation have made great strides over the past decade [79, 80] and are very competitive with conventional deformable mirrors, particularly where a large actuator density is desired. Refractive liquid crystal devices have also been proposed and developed [81, 82], but tend to be slow and are uncommon outside of microscopy [12], or laser processing [74].

The availability of inexpensive reconfigurable metasurfaces has driven research into a field known as wavefront shaping [9, 10, 18, 83–85]. While there is not a strict

convention or definition, adaptive optics is generally associated with controlling distorted wavefronts for a single propagation path while wavefront shaping is generally associated with controlling (or shaping) a combination of multiple scattered wavefronts. We will adopt this convention here and refer to adaptive optics and wavefront shaping in general as wavefront control. While many applications are built around scattering systems possessing time reversal symmetry (TRS), the presence of TRS is not a requirement for all wavefront control systems.

Conventional adaptive optics systems measure the wavefront directly and use an operator, called the reconstructor, to solve the inverse problem between measurements and control signals. Wavefront reconstruction is therefore at the heart of any wavefront control system. The process is generally indirect, as the reconstructor evaluates the wavefront in the basis of command signals, rather than explicitly in phase. Wavefront reconstruction is a specialized area of system identification [86], and relies heavily on methods for solving inverse problems.

For a linear system, or one that can be linearized, the standard reconstructor, $\mathbf{R}$, is a regularized optimal Wiener filter given as [87]

$$\mathbf{R} = \left[\mathbf{B}^T\mathbf{C}_n^{-1}\mathbf{B} + g\mathbf{W} + \mathbf{B}^T\mathbf{C}_\varphi^{-1}\mathbf{B}\right]^{-1}\mathbf{B}^T\mathbf{C}_n^{-1} \tag{2.1}$$

Here, $\mathbf{C}_n$ is the measurement noise covariance matrix, $\mathbf{C}_n = \left\langle \mathbf{n}^T\mathbf{n} \right\rangle$, $\mathbf{C}_\varphi$ is the covariance matrix of the environmental disturbance, $\mathbf{C}_\varphi = \left\langle \varphi^T\varphi \right\rangle$, $\mathbf{W}$ is a weighting/regularization matrix, g is a scalar gain term, and $\mathbf{B}$ is a system configuration (geometry) matrix that relates control signals to wavefront measurements. Both $\mathbf{C}_n$

and $\mathbf{C}_\varphi$ are defined in sensor space, while $\mathbf{W}$ is defined in command space.

The inverse problem is often ill-posed due to singularities or the presence of highly correlated responses with different commands [88], which effectively means we do not have enough information to solve the problem. We can add the necessary information through a process known as regularization. The weighting matrix, $\mathbf{W}$, in Eq. 2.1 implicitly provides Tikhonov regularization, which acts as a spectral filter on the singular values [89, 90]. $\mathbf{W}$ can be the identity matrix to raise all the singular values, or a projection matrix to suppress specific modes that either induce singularities or expend control energy in ways we wish to avoid. The gain term, $g < 1$, then determines how strongly these modes are suppressed. Using the inverse of the environmental covariance matrix, $\mathbf{C}_\varphi^{-1}$, preconditions the solution towards expected spatial modes with the appropriate statistics.

For nonlinear systems, we can apply iterative methods to handle the reconstruction process. These are typically Krylov subspace methods such as the conjugate gradient method [89,91,92]. Regularization can be applied through Landweber iteration, where the gradient is allowed to decay with a relaxation parameter [89,91]. In this case, the iterations are performed "offline", meaning that each iteration is evaluated numerically on the measurements. The convergence rate is therefore only limited by the computational power we can throw at the problem.

Wavefront reconstruction is viable for some metasurface applications, but depends on being able to solve the inverse problem. Complex environments include uncertainties in determining the system configuration ($\mathbf{B}$ matrix), and multiple reflection paths create intricate interference patterns at the antennas, producing chaotic

fluctuations [93, 94]. In addition, short orbits that manifest as persistent features in the ensemble [95] are not removed, leading to a complicated relationship between metasurface commands and cavity scattering parameters. For a metasurface that is small relative to the cavity, the effective strength of the metasurface commands on the cavity scattering parameters is reduced. This results in high correlation between measurements taken with different sets of commands and creates problems for uniqueness, as many potential solutions are extremely similar. The effective strength is enhanced by reverberant environments, however these add additional complications that will be discussed later. In addition, the scattering process is linear, but the relationship between metasurface commands and measured scattering parameters is not necessarily so, particularly for measurements in the temporal domain. In these extreme scattering environments, we are limited to partial information and may not be able to define the system, let alone determine the inverse. Therefore, model-free control approaches that do not require knowledge of the system configuration, and bypass an explicit wavefront reconstruction step altogether, have traditionally been preferred.

Early metasurface control approaches used brute force trial and error, toggling every element or combination of elements [96, 97]. This guarantees that a global minimum is reached, but becomes infeasible with large numbers of elements. The simplest practical approach is an extension of iterative multidither techniques in adaptive optics [98], where, at the i^{th} iteration, the algorithm updates a trial command vector, $\mathbf{a}^*$, with a small perturbation, $\delta\mathbf{a}$, so that

$$\mathbf{a}^*_{i+1} = \mathbf{a}_i + \delta\mathbf{a}_i \tag{2.2}$$

The impact on performance is evaluated through a metric or cost function, J, that is positive and real-valued, and dependent on both the command vector and the environment, $\mathcal{E}$. If the cost function, $J\left(\mathbf{a}^*_{i+1}, \mathcal{E}\right)$, is improved, the trial command vector becomes the new command vector, $\mathbf{a}_{i+1} = \mathbf{a}^*_{i+1}$. Otherwise, the trial command vector is rejected and a new trial command vector is generated. The iterative process continues until either a specified number of iterations, T, are performed without improving the metric, or the cost function reaches a predetermined value, at which point we claim convergence. While simple to implement and not reliant on knowledge of the system configuration, the dithering approach is by no means optimal, so we next turn to stochastic optimization.

Gradient based approaches have proven extremely successful for general stochastic optimization problems [99]. In a stochastic gradient descent (SGD) optimization, the descent is performed by taking steps along the gradient of the cost function with respect to the element command vector. The step size, γ, which may or may not depend on the iteration, determines how quickly the algorithm descends, and is sometimes referred to as a "learning rate". Tuning the step size is an important aspect of SGD methods. If γ is too small, the algorithm will take a long time to converge and may not be able to escape a local minimum. On the other hand, if γ is too large, the algorithm may become unstable. The basic SGD is implemented as

$$\mathbf{a}_{i+1} = \mathbf{a}_i - \gamma_i \nabla_{\mathbf{a}_i} J\left(\mathbf{a}_i, \mathcal{E}\right) \tag{2.3}$$

In most cases, it is not possible to evaluate the gradient, $\nabla_{\mathbf{a}} J$, directly, so it must be approximated. The general approach is to apply a small perturbation to the current command vector and estimate the gradient from a one-sided or two-sided finite difference. The perturbation is applied to all elements of the command vector simultaneously (in parallel) to increase the convergence rate.

A specialty of wavefront control, known as wavefront sensorless, or target-in-the-loop adaptive optics [100], leverages stochastic optimization in a sensor agnostic manner, indirectly evaluating the wavefront through the cost function. In target-in-the-loop approaches, the iterations are performed "online", meaning that each iteration requires applying commands and measuring the result. The convergence rate is therefore limited by the sampling rate of the system.

Target-in-the-loop methods are not as easily analyzed through modern multivariable control theory as conventional methods, but in theory they are applicable to the problem of controlling metasurfaces in complex scattering environments. In particular, stochastic parallel gradient descent (SPGD) [101, 102] has enjoyed great success in target-in-the-loop adaptive optics systems. For SPGD, the gradient is estimated from a one-sided finite difference,

$$\nabla_{\mathbf{a}} J\left(\mathbf{a}, \mathcal{E}\right) \approx \left[J(\mathbf{a} + \delta\mathbf{a}, \mathcal{E}) - J(\mathbf{a}, \mathcal{E})\right] \delta\mathbf{a}^{-1} \tag{2.4}$$

The cost function is arbitrarily defined, allowing SGD methods to be applied

based on the specific need. It can be an image quality metric such as Strehl ratio [101] for imaging systems, signal strength for free space optical communications [103], transmission coefficient for cold spot generation, or scattering matrix eigenvalue magnitudes for CPA state realization. Specific to the problem of controlling metasurfaces in microwave wavebands, energy efficiency in terms of bits-per joule is an attractive metric for wireless networks [104]. Energy efficiency optimization using a reconfigurable metasurface has been proposed and simulated using both SGD and sequential programming in an open scattering environment [105].

SGD methods work well in principle for controlling a metasurface. However, they begin to fail with coarse quantization, which limits the ability to tune both the size of the applied perturbation and the size of the step taken along the gradient. In the extreme case of a binary (1-bit) metasurface, applying a perturbation boils down to simply toggling or not toggling each element, so that for the n^{th} element, $\delta a_n = \{0, 1\}$. This leads to singularities in estimating the gradient (Eq. 2.4), as well as approximation errors with driving the solution along the gradient (Eq. 2.3), since the resulting command must also be quantized to either 0 or 1. While metasurfaces can be manufactured with more bits of resolution for phase control, this increases complexity, cost, and power consumption considerably, making them less attractive for wide scale use. In addition, the capability of binary metasurfaces has been demonstrated many times; these devices can be expected to be utilized whenever power and cost are drivers for implementation.

Since gradient based approaches are problematic with coarse quantization, dithering methods have dominated wavefront control applications with binary tun-

able metasurfaces. We can however modify the dithering technique in Eq. 2.2 to use shaped or intelligent perturbations. When the algorithm is initialized, we do not know where the optimal commands are located with respect to the solution space. We would like to apply "larger" effective changes that induce highly diverse responses with large scale global changes. As the algorithm proceeds, effectively moving along the gradient, we want the changes to become "smaller", and more localized. In this manner, we are able to continue the optimization process without wasting trials on global changes that are less likely to improve the specific metric of interest. Finally, once the algorithm has converged, we would like to be able to make sure we are not stuck in a local minimum.

This shaped perturbation approach is discussed in Chapter 3, and was demonstrated to successfully enable generating cold spots and realizing CPA states for a binary metasurface with 240 elements [24]. In this case, the perturbations were "shaped" by changing the number of elements that were toggled (perturbed) each time the algorithm converged. The algorithm cycled through perturbations that toggled 120, 48, 24, 12, and then 6 elements, with a convergence criteria of $T = 30$ trials. This can also be thought of as a simple policy-iteration method of reinforcement learning [106]. To ensure the solution was not stuck in a local minimum, the algorithm then entered a "single element" phase where three trial command vectors were generated at each iteration that toggled the individual element, the nearest neighbors of that element, and the diagonal neighbors of that element. Cold spots were generated with this technique and provided 4-40 dB of suppression over a 1 GHz frequency range and CPA states were realized and verified with power absorption

ratios $\sim 10^6$ [24].

While dithering allows us to provide wavefront control in some capacity, a true wavefront reconstruction method is desired. Deep learning may provide a viable approach, as it has already been successfully demonstrated for general ill-posed inverse problems [107]. This is the basis for Chapter 4.

2.3 Complex Reverberant Scattering Environments

Complex reverberant scattering environments contain universal fluctuations with statistics governed by random matrix theory (RMT) [108], as well as deterministic behavior from the system specific configuration of the ports and short orbits, i.e., prompt or direct paths, between the ports [95, 109, 110]. For the particular case of chaotic cavities, electromagnetic wave fields have specific statistical properties that depend upon a limited number of parameters [111]. Among these is the fact that the wave field is statistically equivalent to a random superposition of plane waves [112]. As such, we can leverage analytical tools from the active research area of quantum chaos [108, 113] in the more generalized framework of wave chaos [114, 115].

Complex scattering environments are often characterized by their scattering matrix, or S-matrix, which is a frequency dependent transfer function matrix containing the complex-valued reflection and transmission coefficients between the ports. While useful for describing the overall behavior, separating the universal and deterministic features is difficult when working with the S-matrix [116].

An analytical approach known as the random coupling model (RCM), has been shown to accurately predict fluctuation statistics and allows separation of the universal and deterministic contributions in a simple additive manner [114,115]. The RCM extends RMT and is characterized by a single parameter, α, that describes the losses in the system. It is supported by wealth of experimental validation data with chaotic microwave cavities [93,94,117–119]. The behavior in large, thee-dimensional enclosures has also been studied to understand these statistics and the potential impact of high power microwave (HPM) attacks [1, 120]. A table of characteristic parameters of chaotic microwave cavities derived from the RCM is given in Appendix A.

The RCM works in the impedance domain to separate the universal contributions; conversion between impedance and scattering is handled through standard bilinear transformations [121]. In the RCM, the fluctuating cavity impedance, $\mathbf{Z}^{\text{cav}}$, is defined as

$$\mathbf{Z}^{\text{cav}} = j\text{Im}\{\mathbf{Z}^{\text{rad}}\} + \text{Re}\{\mathbf{Z}^{\text{rad}}\}^{1/2}\xi\text{Re}\{\mathbf{Z}^{\text{rad}}\}^{1/2} \tag{2.5}$$

Here, $\mathbf{Z}^{\text{rad}}$ is the radiation or free-space impedance of the ports and ξ is the fluctuating or universal component, which is described by RMT [114]. For lossless systems, ξ is a Lorentzian distributed random variable. With loss, the distribution becomes much more complicated, but it is well suited to Monte Carlo simulations [111] (see Appendix G). $\mathbf{Z}^{\text{cav}}$ represents a single realization of a cavity, experimental results typically collect an ensemble of realizations with a mechanical mode stirrer

used to realize new cavity configurations.

The RCM has been shown to apply to fading statistics in open systems, such as wireless communication paths, as well as closed systems, such as microwave cavities. [122, 123]. A chaotic system is characterized by extreme sensitivity to initial conditions and is qualitatively different than an open one. In open systems, fading statistics are often modeled with empirically fit distributions [124]: the Rayleigh distribution when no line-of-sight path is present, the Rician distribution when a strong line-of-sight path is present, or the K distribution for propagation over the ocean [125]. The limiting cases of Rayleigh and Rician distributions are captured by the RCM with the σ parameter related to the loss parameter, as $\alpha = (8\pi\sigma^2)^{-1}$, and the ν parameter equal to the magnitude of the short orbits [122, 123].

Operating in a reverberant scattering environment presents an additional set of challenges in comparison to an open environment. In addition to the fundamental difference in the character of the fluctuations, in the semi-classical case or short wavelength limit, we can look at the behavior of ray trajectories. Specifically, we are interested in the change in ray trajectories in response to a change in the metasurface configuration. In an open system, there is a single ray (or ray bundle) that is observed by the sensor, with at most a single reflection off the metasurface. In a chaotic system, that single ray will reflect off multiple walls and obstacles in the cavity and possibly off the metasurface itself multiple times before reaching the sensor. This creates a cascading effect, so that the wavefront at the sensor is a combination of constructive and destructive interference of the multiple rays. The effect of these multiple interference paths is highly dependent on the configuration of the cavity.

For an open system, a wavefront reconstruction approach is only dependent on the geometry between the sensor and the correcting device and is invariant to environmental changes (provided the disturbances remain within the dynamic range of the sensor and corrector). Wavefront reconstruction is therefore very robust for an open system. For a chaotic system however, small environmental changes can cause a wavefront reconstruction technique that was previously successful to no longer be viable, so that being "close" is not good enough.

2.4 Common Simplifying Assumptions

To build environmental models for simulations, we typically need to make assumptions or approximations for simplicity or computational tractability. We also need to ensure that these assumptions are valid for the environments that are being modeled. Otherwise, the models may neglect potentially significant effects. We will outline some of the most problematic simplifying assumptions here.

The first simplifying assumption often made is that the channels are assumed to be perfectly known by the transmitter, so the only uncertainty in the environment is random thermal noise at the receiver(s). In complex scattering environments, there is always uncertainty due to multi-path reflections, which can be significant. In addition, inside chaotic cavities, measured scattering responses with the same initial conditions will change over time, a phenomenon known as scattering fidelity decay [126–128], which means that perfect knowledge of a complex scattering environment has a finite lifetime. This lifetime can be several days in controlled conditions, but

is sensitive to temperature and humidity and will be reduced in scenarios such as dense urban environments. Having only partial knowledge of the system limits deterministic control approaches and encourages learning algorithms. Scattering fidelity has an impact here as well; as the scattering responses start to change, any machine learning algorithm will require periodic retraining.

The second simplifying assumption often made is that the equivalent channel matrix is assumed to be invertible, so the inverse problem is well-posed. Complex scattering environments generally contain short orbits, or prompt direct paths, that are persistent across measurements [95, 118]. These short orbits induce correlations that are difficult for simple machine learning approaches to unwrap and typically lead to ill-posed inverse problems. Excluding multi-path reflections and short orbits can overestimate the performance of a given algorithm.

The third simplifying assumption often made is that all the propagation paths are assumed to have a single reflection off the metasurface, so direct line-of-sight and multi-path trajectories are not included. Channel fading is then modeled with Rayleigh amplitude statistics. In real-world systems, strong direct line-of-sight paths induce Rician statistics and the presence of multi-path reflections drives statistics that are governed by RMT [122, 123]. Neglecting these statistics can lead to algorithms that are not properly tuned. The longer tails in the distributions lead to large amplitude signal spikes that can degrade imaging performance or disrupt signal processing algorithms. This phenomenon is well known for the maritime synthetic aperture radar (SAR) systems and has lead to the development of the K-distribution [125].

The final simplifying assumption is that the metasurface is often assumed to have infinite phase resolution, so quantization effects are not included. As stated previously, most commercially available metasurfaces have a single bit of control, though custom devices with 2 bits of control are becoming available [31–33, 129], which means quantization effects are important and likely significant. In addition, as discussed in Section 2.2, coarse quantization can cause gradient based controllers to fail, so neglecting quantization effects may lead to poorly performing real world controllers. The metasurfaces are also idealized, with identical responses across all elements. In real devices, manufacturing defects produce nonuniformities between the phase at each element, and the metasurface may also include uncontrollable losses or gain.

In addition to simplifying assumptions, testing and verification is often done in anechoic chambers to remove the environment and capture only the impact of the reconfigurable metasurface. Anechoic chambers are very good at covering up emissions problems, which become immediately apparent in reverberation chambers [130]. A complex scattering system is reverberant in nature, so mismatches that seem negligible in anechoic chambers may be significant in real world environments.

Caution should be taken when applying any of these simplifying assumptions. Otherwise, they can overestimate the performance or install a false sense of confidence in a particular approach.

2.5 Microwave Coldspots

A microwave coldspot is simply a null in the transmission coefficient between two ports. References [17–19] present a sequence of research using a 102 element metasurface with $\lambda/2$ spacing to generate coldspots. First, a small office was used as the reverberating environment and the ability to null transmission at a point was demonstrated when the metasurface covered only $\sim 1\%$ of the overall surface area [18]. This work was extended by placing the metasurface inside a cavity where it covered $\sim 7\%$ of the surface area [17] and culminated in loading the cavity to test cases with low, medium, and high Q values [19]. In each of these cases, the optimization process was performed at a single frequency corresponding to the element spacing of $\lambda/2$. These experiments show evolving capability; however, practical applications require operation over a range of frequencies and need to simultaneously handle inputs from different directions.

2.6 Coherent Perfect Absorption

CPA is an exciting research area where coherent excitation of a lossy system can result in complete absorption of all incident waves [26, 27]. Creating CPA requires coherent excitation of all the ports in an eigenvector whose corresponding S-matrix eigenvalue is zero. Operationally, the first step in establishing CPA is to find an eigenvalue of the scattering matrix that is close to zero. For example, a 2 x 2 scattering matrix will have a pair of eigenvalues at each frequency. However,

realizing CPA only requires driving 1 eigenvalue to zero, as the other eigenvalue corresponds to the anti-CPA state [28].

CPA has typically been investigated in simple, regular scattering scenarios and cavities but recently it has been demonstrated in more complex systems, specifically in the realm of wave chaos, and graphs [131–134]. These works analytically demonstrate the use of RMT to explore CPA states with semiclassical tools without relying on the limit of weak coupling. CPA states have also been experimentally investigated in multiple scattering environments [27], and in graphs that break time-reversal invariance [28]. The use of enhanced spatio-temporal diversity from a metasurface for realization of CPA has not yet been explored and presents a novel capability.

While practical applications are still being developed, research has demonstrated that a high fraction of the power was absorbed by the target in a CPA demonstration using a tuned absorber embedded in a lossy environment [135]. This work also showed that the target absorbed virtually nothing in the "anti-CPA" state, demonstrating a high degree of control over absorption by a specific target in a CPA scenario. An interesting future application is to utilize a generalized Wigner-Smith operator [136] to apply a high absorption fraction to a target with a modulated impedance or loss.

Recent research has investigated the use of metasurfaces for Perfect Absorption (PA) inside a cavity and demonstrated a secure communication system as an application [22]. PA is a complementary idea to CPA for a single port system that relies only on the reflection coefficient, S_{11} [137]. Full coherent multi-channel CPA is more complicated than single channel perfect absorption. However, one advan-

tage is the increase in delivered power by a factor of N, where N is the number of channels. This is a significant gain and worth the difficulty of additional phase and amplitude control.

2.7 Deep Learning

Learning is natural to humans but difficult for machines and should be explicitly differentiated from simple memorization. As a classification example, learning means developing the ability to recognize something new as similar to something previously observed. We want the machine to be able to learn patterns that we may not be able to explicitly describe from data that we believe contains these patterns. Machine learning (ML) is therefore dependent on the machine being able to generate representations of the underlying data. Bengio et al. discuss the many qualities of a "good" representation in great length [138]. These qualities grow exponentially more complex with the number of features due to the curse of dimensionality.

Artificial neural networks (ANNs) started to rise in the 1980's as an attempt to leverage the architecture of the human brain and get away from specific hand tailored representations that required significant engineering and domain expertise [139]. Early ANNs contained only a few layers due to limitations with computational resources and difficulties in training and stagnated in the mid 1990's. In the mid 2000's however, ANNs made a comeback in the form of deep learning due to advances in high performance computing platforms, innovations in architecture and training, advanced regularization techniques, and the availability of labeled training data

[139].

The concept of depth comes from complexity theory as defined for circuits, with the depth being the longest path from an input to an output [140]. The number of potential paths or ways to reuse features grows exponentially with depth, which leads to progressively more abstract features [138]. Depth therefore provides an exponential advantage in the expressive power of a network [141]. The use of a deep network allows the machine to learn a rich, hierarchical feature representation with data-learned features progressing from simple to abstract concepts [142].

Depth is provided in terms of "hidden" layers, that are unobservable from the input or output of the network. A representative deep network with 2 hidden layers in addition to the input and output layers is given in Figure 2.2. This is an example of a feedforward network as the connections are in only one direction with no explicit dependence on neurons further upstream.

Many networks are designed for classification of objects, as such, it is common to refer to the inputs to the network as "features" and the outputs from the network as "labels". The incremental outputs from the various layers are often referred to as feature maps. Supervised learning means labels are available for use, while unsupervised learning means labels are not available.

The ubiquity of open source tools such as Tensorflow [143] and Keras [144], both in Python, greatly simplifies the design and use of deep ANNs. However, these tools mask many important and subtle details, so it is quite easy to develop broken designs without realizing it. Unfortunately, many online resources also only provide a superficial discussion, focusing instead on the software layout.

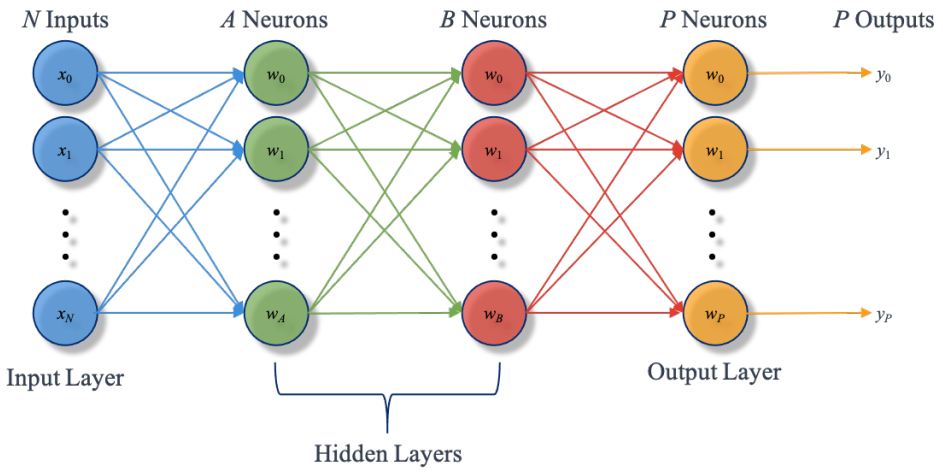

Figure 2.2: **Deep Neural Network.** A deep neural network consists of an input and an output layer around 1 or more hidden layers. Intermediate outputs after each layer are a linear combination of the weights with the inputs. Nonlinear activation functions (not shown) operate on these intermediate outputs. Shown here is a case with N inputs in the input layer, followed by 2 hidden layers containing A and B neurons, and then an output layer with P neurons, providing P total outputs.

2.7.1 Types of Layers

As a consequence of the "no free lunch" theorem [145], no single method or technique is optimal for all problems. This has led to many different varieties of artificial neurons as well as architectures.

2.7.1.1 Dense Neural Networks

The simplest type of layer is "dense" or "fully connected" and provides a single scalar weight for each neuron. An example of a dense neural network is given in Figure 2.3, which shows that the outputs for each neuron are linear combinations of the inputs. The number of outputs is equal to the number of neurons, and a nonlinear activation function acts as a gating function for each output. In some

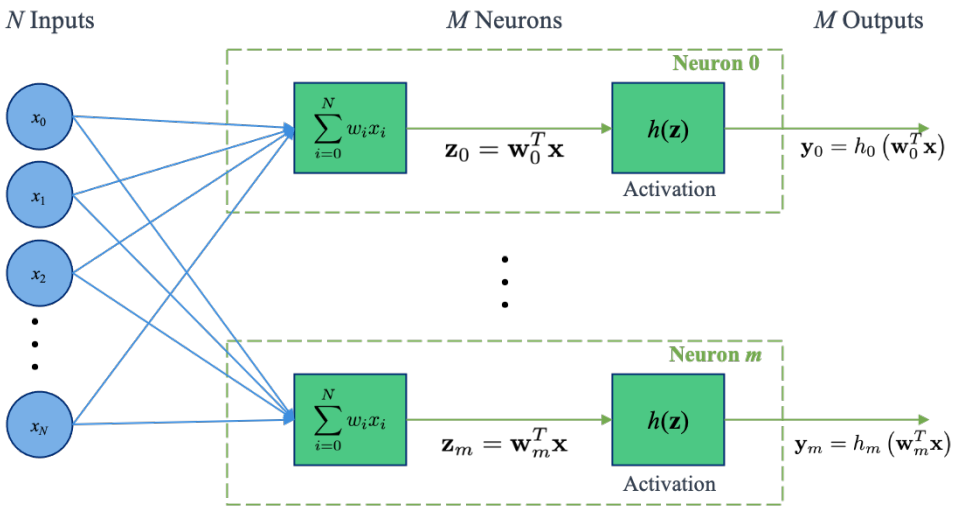

Figure 2.3: **Dense Neural Network Layer.** A dense neural network processes N inputs with M neurons. The intermediate output vector from each neuron, $\mathbf{z}$, is a linear combination of the inputs with a vector of trained weights, $\mathbf{w}$, for each input. The final output from each neuron, $\mathbf{y}$, is then the result of a nonlinear activation function applied to the intermediate output. Because $\mathbf{z}$ is the input to the activation function, it is referred to by some authors as "activations".

literature the intermediate calculations, $\mathbf{z}$, are referred to as "activations".

The output layer in almost all ANNs will be a dense layer, this ensures that the output of the ANN will have the appropriate size and consist of a linear combination of the outputs from the previous layer. The output layer should be thought of as a final conditioning step and does not perform feature extraction.

2.7.1.2 Convolutional Neural Networks

A convolutional layer simply convolves a kernel, h, with the input and provides an optional bias, b. For a 1-D kernel of length L, the output y at point n for an input x is given by

$$y[n] = \sum_{n-(L-1)/2}^{n+(L-1)/2} x[k]h[n-k] + b \tag{2.6}$$

As shown here, the output of the convolution will have a different size than the input. If the input has length N, the output will have length M, given by $M = N - L + 1$. We can zero pad the signal to force $M = L$ if desired.

A convolutional layer is defined by the number of filters (number of neurons), the length of the kernel, the stride, the padding, and the activation function. We can also control various parameters that affect how the kernel and bias are updated during training. Padding specifies whether to use the zero padded result of the same size as the input or the unpadded result with a smaller size than the input. Stride is the number of samples to skip during the operation and is left at 1 in most cases. Kernel lengths are typically odd for alignment (centering).

A transposed convolutional layer is also available and often incorrectly described as deconvolution. As forward convolution without padding results in an output with a smaller size than the input, transposed convolution can be thought of as convolution with upsampling and provides an output with a larger size than the input. These are generally found in image processing applications with U-shaped networks to preserve the output size [146].

2.7.1.3 1D Convolution

An aspect that is not well understood outside of the signal processing community is how convolutional layers are implemented for inputs containing multiple

features. In signal processing, the feature dimensionality is referred to as the number of channels and is sometimes defined as the width or the depth of the data. This arises from color image processing with 3 color channels for red, green, and blue. To perform the convolution over the desired dimension and ensure all the features are captured, the convolution kernel is multidimensional as shown in Fig. 2.4. For a specified kernel length, k, the size of the kernel for a 1D convolutional layer with an input containing N features is $k \times N$. The kernel will only be shifted along a single dimension, the local frequency window in our case, but will contain optimized weights for each element. This means that the number of trainable parameters for a 1D convolutional layer scales as kN, not just k. For an input data set X, the output, y, of the convolutional layer with kernel K is given by

$$y[n] = \sum_{i=0}^{k-1} \sum_{j=0}^{N-1} K[i,j]X[n-i,j] \tag{2.7}$$

For our purposes, we will zero pad the input data by appending $(k-1)/2$ rows of zeros to either end and keeping the central part of the result, so the number of points along the convolution dimension is constant in the output. By designing a convolutional layer consisting of N_l filters, there will be N_l such outputs or new features for the next layer.

2.7.1.4 Receptive Field

The receptive field of a CNN defines the number of points in input space that contribute to the result at a single point in a given layer. Our CNNs use zero padding to keep the output size fixed, and the stride and dilation are always set

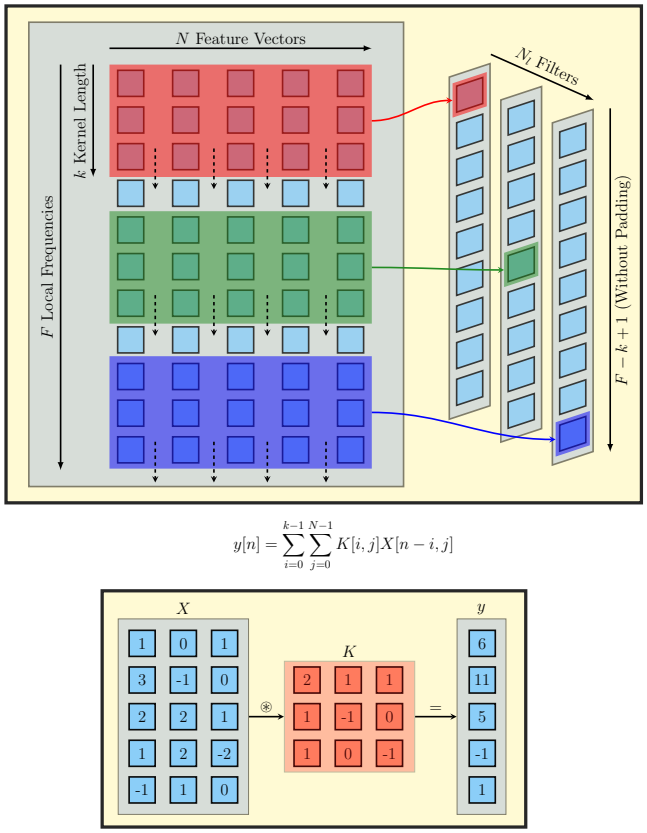

Figure 2.4: **1D Convolution with Multiple Features. Top:** Graphical representation showing the N feature vectors and F local frequencies processed by 3 different filters with kernel length k. The kernel only moves along a single dimension (vertically) even though the data is represented in a 2D format. Each position of the kernel results in a single point in the output vector which has length $F - k + 1$ if zero padding is not used and length F if padded as described in the text. The weights for each of the kN_l elements of the kernel are computed collectively, but can be different. **Bottom:** Numerical example with 3 features containing 5 points each convolved with a kernel of length 3. The input data is zero padded with a row of zeros at the top and bottom and the outputs for the 5 central rows are kept.

to 1. This means the receptive field at any layer, r_l, is given by a simple recursive equation dependent on the receptive field at the previous layer, r_{l-1}, and the kernel length of the current layer, k_l [147].

$$r_l = r_{l-1} + k_l - 1 \tag{2.8}$$

As shown in Fig. 2.5, the receptive field for a sequential architecture grows monotonically with depth, with each layer only seeing the receptive field from the preceding layer. An architecture that utilizes parallel branches along with concatenation conserves the intermediate receptive fields, making them available for all subsequent layers and introduces width as well as depth to the network and providing the motivation for the inception module.

2.7.1.5 Pooling Layers

Pooling layers are used to reduce the size of output feature maps and can help prevent networks from learning features only at specific locations. They operate by reducing the size of the feature map by taking either an average or the maximum value of the input values over the pool size, commonly 2.

2.7.1.6 Dropout Layers

Dropout layers provide a regularization method that randomly sets a specified percentage, $0 < d < 1$, of input values to 0. The sum of the inputs is held constant, so inputs that are not dropped out are scaled by $1/(1-d)$. Dropouts are only applied during training and are not present when evaluating the neural network.

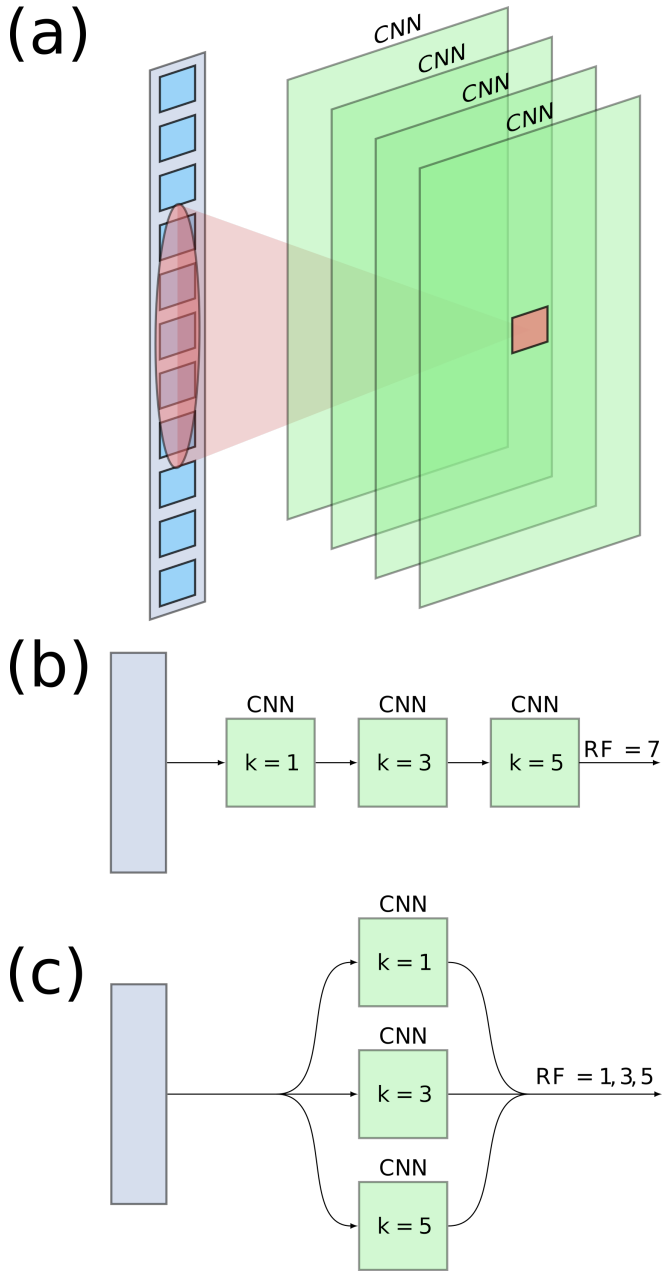

Figure 2.5: **Receptive field.** **(a)** The receptive field of a convolutional neural network (CNN) layer indicates the number of points in input space that contribute to a single point at a given layer. **(b)** For a purely sequential architecture, the receptive field increases monotonically. **(c)** A parallel architecture with concatenation produces multiple receptive fields with each available for subsequent layers, promoting sparsity in the representations.

2.7.2 Training Neural Networks

Training a network leverages an objective or loss function, J, evaluated over the neuron weights, $\mathbf{w}$. For the mean squared error (MSE), the loss function over K samples of data is given as

$$J(\mathbf{w}) = \frac{1}{K}\sum_{i=0}^{K}\frac{1}{2}||\mathbf{c}_i - h(\mathbf{z}_i)||^2 \tag{2.9}$$

Here, $\mathbf{c}_i$ is the true set of outputs (labels) from the i^{th} sample in the data and $h(\mathbf{z}_i)$ is the collection of outputs from the neural network, explicitly showing the dependence on the activation function. Training is performed iteratively, with each iteration referred to as an epoch. Networks are generally trained in reverse using an algorithm known as backpropagation. Backpropagation is a layer-wise recursive process that starts at the output of the network and propagates errors back through the network layer by layer; this allows us to compute the desired gradients for each neuron.

The workhorse algorithm is the SGD [139], where the weights at each epoch, e, are updated based on the estimated gradient of the cost function scaled by a factor γ.

$$\mathbf{w}_e = \mathbf{w}_{e-1} - \gamma\nabla J \tag{2.10}$$

SGD has problems with "pathological curvature", or narrow ravines, which are

common around local optima and the response tends to oscillate back and forth across the ravine. To address this, we can introduce the concept of momentum, "forgetting" a portion of the previous gradient [148].

$$\begin{aligned} \mathbf{v}_e &= \mu \mathbf{v}_{e-1} - \gamma \nabla J \\ \mathbf{w}_e &= \mathbf{w}_{e-1} - \mathbf{v}_e \end{aligned} \tag{2.11}$$

Here, $\mathbf{v}$ is an intermediate calculation and μ is the momentum, typically around 0.9. Momentum can be thought of as a very coarse approximation of the curvature or 2nd derivative.

The gradient can be looked at in terms of the chain rule for derivatives [139],

$$\nabla J = \frac{\partial J}{\partial \mathbf{w}} = \frac{\partial J}{\partial h} \frac{\partial h}{\partial \mathbf{z}} \frac{\partial \mathbf{z}}{\partial \mathbf{w}} \tag{2.12}$$

The selection of activation function and the existence of a gradient is therefore important in training the network. An example of typical activation functions is given in Figure 2.6, showing the tanh, sigmoid, and rectified linear unit (ReLU). The tanh and sigmoid activation functions experience saturation where the gradient goes to zero. When the intermediate outputs, $\mathbf{z}$, are near these saturation regimes, we experience a vanishing gradient and the network is unable to train. The ReLU does not have this issue and its development enabled deep networks by improving training convergence and providing better solutions than tanh or sigmoid [149].

Another activation function used almost exclusively in the output layer of

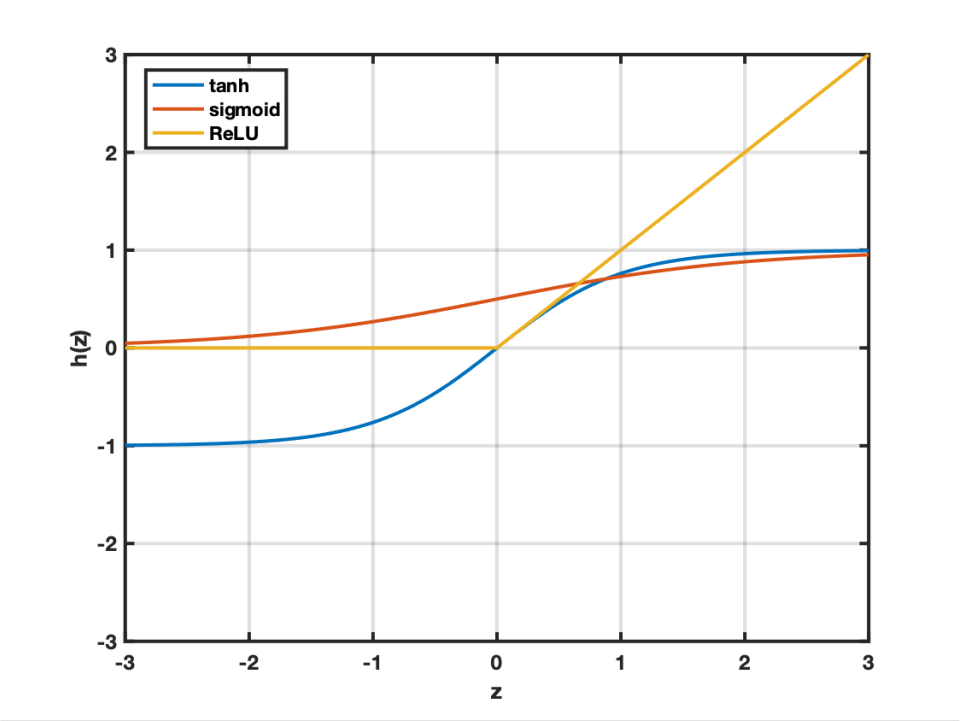

Figure 2.6: **Activation Functions.** Three commonly used activation functions are the tanh, sigmoid, and rectified linear unit (ReLU). The ReLU significantly improved convergence of training and is the standard activation function in deep learning.

classifier ANNs is softmax. Softmax forces the sum of probabilities to 1 and guarantees that the network will always choose one and only one class from the list of possibilities.

Finally, it is also important to understand how the inputs to the activation functions, $\mathbf{z}$, behave during training. In general, the distributions (mean and standard deviation) between layers change over the course of training, a phenomenon known as internal covariate shift [150]. This causes problems as the inputs to the next layer in the network become a moving target for the learning process and the weights tend to get learned sequentially from the output layer to the input layer.

Internal covariate shift can be addressed with batch normalization, which normalizes each feature (dimension) of the input independently across the samples. The normalized input for the n^{th} feature is then

$$\hat{x}_n = \frac{x_n - \langle x_n \rangle}{\sqrt{\langle x_n^2 \rangle - \langle x_n \rangle^2}} \tag{2.13}$$

Simply normalizing the inputs to a layer may change the underlying representation of the layer. To account for this, batch normalization scales and shifts the normalized values to ensure that the identity transform is represented. This provides an additional set of parameters, γ and β, to be learned [150]

$$\tilde{x}_k = \gamma_k \hat{x}_k + \beta_k \tag{2.14}$$

Chapter 3: Stochastic Optimization Approach

In this chapter we describe the use of a binary programmable metasurface to create microwave coldspots at arbitrary frequencies, and to realize CPA states, both within the 1 GHz band of operation of the metasurface. The conceptual overview is shown in Figure 3.1 where the metasurface is installed in a complex reverberating cavity and controlled in a closed loop manner. Input directional diversity is introduced by simultaneously driving multiple ports with arbitrary relative phase shifts. An iterative optimization algorithm is used to generate coldspots at the output port, or to drive candidate scattering matrix eigenvalues towards the origin to achieve CPA.

3.1 Cavity Configuration

As shown in Figure 3.2, the array was installed in a 0.76 m^3 cavity where it covers $\sim$ 1.5% of the total interior surface area. The cavity has 3 ports with one acting as a target for scoring and two used for signal injection; the input ports can be driven either individually or collectively with a relative phase shift. Although 3 ports are present, we are typically using the cavity as a 2-port system because we have a 2-port network analyzer. All 3 ports are used when driving ports 1 and 3 simultaneously, in which case the underlying scattering system is represented by a

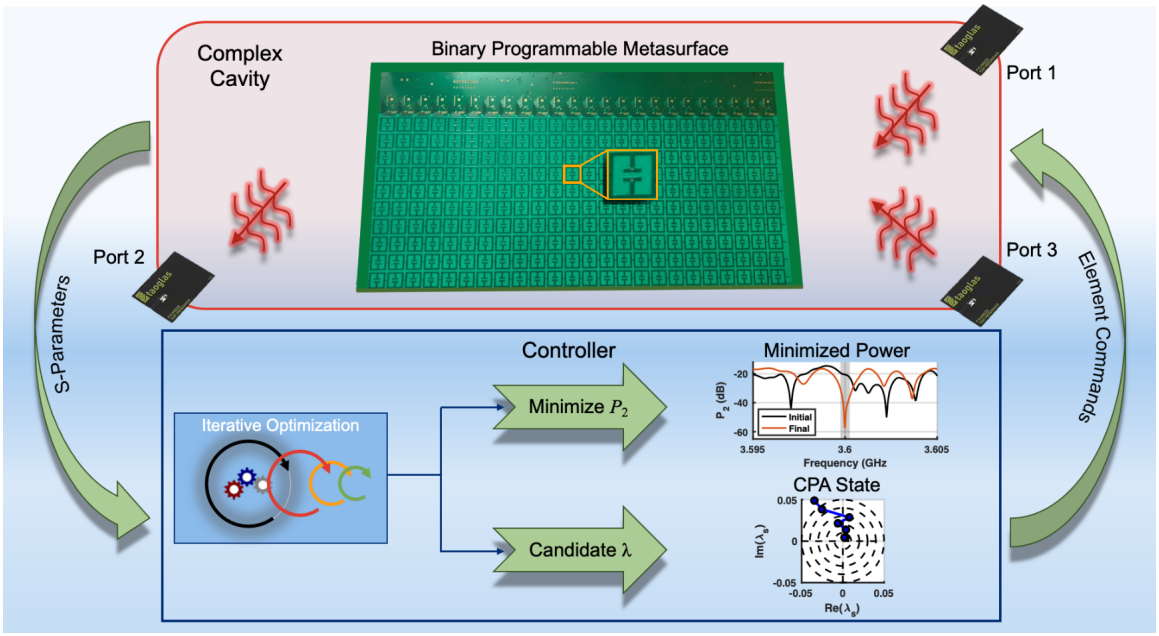

Figure 3.1: **Conceptual overview of the metasurface enabled cavity as a closed loop system.** The cavity S-parameters (scattering parameters) are measured with a network analyzer and passed to a controller that updates the metasurface elements with a new set of commands. The controller can generate coldspots at port 2 at an arbitrary set of frequencies, or drive candidate S-matrix eigenvalues towards the origin, and includes a stochastic iterative optimization algorithm. The three ports allow additional angular and spatial diversity to be added at the inputs. The inset shows a closeup view of one of the metasurface unit cells.

3×3 scattering matrix. While the experiment has a physically fixed number of ports, the results can be generalized to an arbitrary number of ports. The cavity has both low and high loss configurations to test how the behavior varies with the typical quality factor, Q, of the modes; here we consider the high Q case. Introducing the metasurface to the cavity reduced the average Q in the frequency band of operation of the metasurface by a factor of $\sim$2. However, once the metasurface was installed, the average Q was found to be independent of the number of active or inactive elements on the surface. The quality factor was determined to be roughly 5.5×10^3, by measuring the power decay time, $\tau_c = Q/\omega$ (250 ns with the metasurface installed). A method for estimating the time constant is given in Appendix C. Further details of the metasurface, cavity construction, experimental setup, and impact of the metasurface on losses are provided in Appendix E.2.

The cavity mean mode spacing in frequency, Δf, is found from the Weyl formula as $\Delta f = \pi c^3 \left(2\omega^2 V\right)^{-1}$ [119]. A measure of the loss in the cavity is the Q-width of a mode normalized to the mode spacing, $\alpha = f/(2\Delta f Q) = 3$ for this cavity. For our cavity, the mean mode spacing is roughly 115 kHz at a 3.5 GHz center frequency. As discussed in Appendix E.1, the mean spacing between nulls in the transmission coefficient, $|S_{21}|$, was found to be $\sim$2 MHz and the average width of the nulls was found to be $\sim$200 kHz. This indicates a transmission coefficient null contains about 2 modes. Alternatively, it corresponds to a path difference of 750 m between two interfering signals.

We are interested in the steady state response, so the average Q does put a bound on the effective speed with which we can switch the cavity scattering matrix

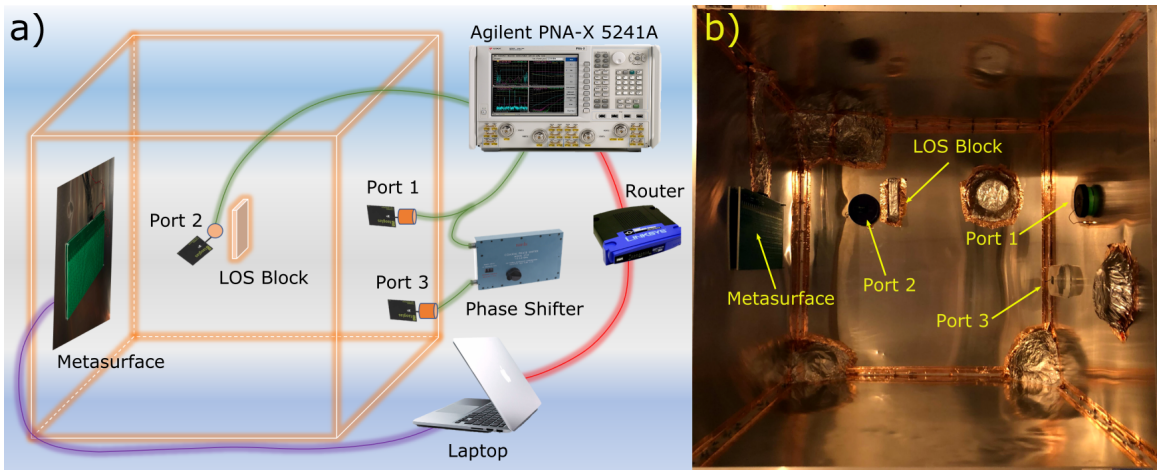

Figure 3.2: **Experimental schematic and cavity configuration. a)** Schematic of the experimental setup in the configuration driving all 3 ports. A network analyzer (Agilent PNA-X 5241A) is used to measure cavity S-parameters, with channel 1 connected to both Port 1 and Port 3 (through a phase shifter) and channel 2 connected to Port 2. The ports are terminated with ultra wide band (UWB) antennas. The metasurface is mounted on the cavity wall opposite Ports 1 and 3, and a block is used next to Port 2 to break the line-of-sight (LOS) between Port 2 and Ports 1 and 3. A laptop controls the system and is connected to the metasurface through a USB interface and to the network analyzer through a wired ethernet link. **b)** Photograph looking inside the cavity, with the metasurface and ports labeled. Also shown in the photo are the line of sight block and the irregular scattering elements installed on the cavity walls as discussed in Appendix E.1.

between fixed states. This can be seen through the power decay time, which measures how quickly energy in the cavity dissipates. In order to guarantee a steady state result, the metasurface commands must be toggled at a rate slower than $1/\tau_c$. The time between measurements must also be staggered by several τ_c to ensure each measurement corresponds to the desired metasurface commands. To observe transient behavior in a cavity with a 250 ns power decay time, the metasurface commands must be switched at rates greater than 4 MHz. This assumes the bandwidth of the measured phenomena is wider than the switching rate; an additional complication arises with narrow bandwidth responses that are of interest. When considering finite bandwidth nulls (200 kHz as stated above), the narrower bandwidth process will determine the bound. Our experimental setup is limited to switching rates $<$ 1 Hz, so neither limit presents a practical concern for our configuration. However, experiments with an embedded microcontroller demonstrated that the metasurface itself can be switched at rates up to 15 kHz [30], so this may need to be considered with high speed operation in higher Q cavities.

A complex scattering system such as our cavity exhibits both universal fluctuations, which can be described by RMT [108], and deterministic behavior arising from the system specific configuration of the ports and short orbits between the ports [109, 110, 151]. Due to the small relative size of the metasurface, the chaotic ray paths with many multipath bounces will experience the strongest influence. Since minimizing the power received at a port is accomplished by creating destructive interference of the ray paths, the relationship between commands and responses is quite complicated. This leads to utilizing stochastic iterative approaches, or ma-

chine learning, in place of linear deterministic methods for control. Here we consider iterative processes. A metasurface covering a larger fraction of the interior surface would likely produce stronger results [18] and allow us to use a transmission matrix based approach to determine the optimal metasurface commands [152–154]. For this reason most prior research utilizes metasurfaces that cover a significant portion of a wall (or multiple walls). However, using a relatively small metasurface coverage is better suited for real world applications where it is not practical to build or use a larger device.

A key step in evaluating system performance is to determine the range of possible responses of the scattering properties of the system so as to ensure that we have a sufficiently diverse command set. Unfortunately, with 2^{240} possible commands (approximately 1.8×10^{72}), it is not feasible to test every one and we need to find a reduced number that produces the full range of outcomes. As discussed in Appendix E.3, deterministic decomposition of commands into orthogonal basis functions, such as Hadamard bases [155, 156], generated a very narrow range of system scattering responses. Diversity in the responses requires a distribution of commands with a variety of spatial frequencies, ratios of active to inactive elements, and localized groupings of active elements. Doubly random methods or compound distributions, such as a biased coin toss, or power law spectral density with the bias, or power exponent itself a random draw, were found to yield the widest range of responses. Details of our novel stochastic algorithm are discussed in the next section.

3.2 Generating Coldspots

Our goal is to program the metasurface to minimize the transmission between two ports in a complex scattering system at an arbitrary frequency. Cases are scored by evaluating the difference in average power, ΔP_2, in a specified frequency range at a given center frequency between the initial inactive (all 0s) state and the current state of the metasurface. To maximize this difference we take a directed random walk approach in which at each step a number of array element states are toggled (changed), ΔP_2 is evaluated, and the new state is accepted or rejected based on whether or not it decreases ΔP_2. As discussed in the previous section, we need to have a mix of large and small spatial groupings of elements and a varied number of active elements to ensure a diverse set of responses. To meet this requirement, our iterative algorithm operates in 2 distinct phases: multiple element toggling and individual element toggling.

In the multiple element toggling phase, we select M elements at random as a trial and toggle their state ($0 \rightarrow 1$ and $1 \rightarrow 0$). If ΔP_2 is decreased, the trial set of commands becomes the new reference set and we repeat the process selecting another M elements at random and toggling their state. When T consecutive trials have been made without improving ΔP_2 we claim convergence and move to the next value of M. In a typical experimental run, $M = [120, 48, 24, 12, 6]$, and $T = 30$. After all values of M have been exhausted, we move to the individual element phase.

The individual element toggling phase has 3 cases associated with each trial.

We select a single element at random and toggle it and then, in an adaptation of the neighbor toggling method of Ref. [97], we toggle the 4 nearest neighbors and the 4 diagonal neighbors. ΔP_2 is evaluated for each of these cases and the algorithm continues as in the 1st phase until T consecutive trials are performed without improving ΔP_2.

The multiple element toggling phase tends to result in a local minimum which is difficult to escape when toggling only a single element. Adding neighbor toggling significantly improves the performance, as it provides larger localized changes in the command set and allows us to escape the local minimum. Even with the neighbor toggling, however, our stochastic approach does not guarantee that a global minimum is found. Increasing the convergence criteria, T, can increase the probability of finding the global minimum, but comes with the cost of increased time. The absolute minimum is not necessarily required, and our stochastic algorithm is able to provide substantially deep nulls at arbitrary frequencies in a reasonable amount of time.

A typical experimental run will provide $\sim$350 trials, $\sim$25 iteration updates, and take $\sim$1.5 hours, as the experimental setup is not optimized for run time. We use an ethernet connection to transfer 32,001 frequency samples over the full 1 GHz band for each of the 4 complex S-parameter measurements using 64-bit precision. With the frequency values themselves included, this means 2.3 MB of data are transferred for each trial. In addition, the commands and measured $|S_{21}|$ are plotted at each trial for operator feedback, resulting in a delay of $\sim$15 seconds per trial. Disabling plotting and capturing only the processed frequency band could potentially reduce

the time to 1-2 seconds per trial, or 6-12 minutes for an experimental run. In general, the cavity should not be used for other purposes while generating a coldspot, as trials that move in the wrong direction in the solution space may produce undesirable responses. Reducing the time to find a solution in only a few minutes may present an acceptable interruption in service. To move towards a faster, real time operational system, we would replace the network analyzer with software defined radios (SDRs), such as the HackRF One [157] or BladeRF [158] commercial devices which retail for ∼$300 or ∼$500-1000 respectively. In addition, an embedded micro-controller, such as an Arduino or Raspberry Pi, could be used to reduce the USB communications overhead induced by traditional desktop operating systems when interfacing with the metasurface. This would mean measuring signal I and Q channels rather than S-parameters; however, this is a realistic requirement for a practical fielded solution that would not use a bulky, expensive network analyzer anyway. Trial rates approaching 1 kHz could be achieved in this fashion, though substantial engineering effort would be required to reduce the latency to approach the metasurface switching limit of 15 kHz [30].

Figure 3.3 shows the results obtained when minimizing the average power at the output port and compares the results of many different experiments and configurations. All the cases are scored by the change in average power, ΔP_2, between the initial inactive (all 0s) state and the final state. The optimization algorithm was performed with ΔP_2 evaluated over a single frequency band as well as simultaneously over multiple separated frequency bands. As discussed previously, the widths of the nulls were observed to be ∼200 kHz. The initial bandwidth was

selected to be 500 kHz in order to ensure that ΔP_2 was evaluated over an entire null. In addition, the cavity configuration was switched between driving a single input port and driving two input ports simultaneously with varying relative phase shifts. The achieved suppression ranges from 4-40 dB with most cases providing $>$ 10 dB. The lower values of ΔP_2 arise in the following cases: working near the edges of the metasurface operational window, evaluating ΔP_2 over a large bandwidth, or evaluating ΔP_2 over multiple separated bands. This is not surprising as more bandwidth results in more features in the region where ΔP_2 is evaluated, which then means more degrees of freedom are required to be manipulated for destructive interference. The metasurface provides some benefit outside of the 3-3.75 GHz design window; the reflection phase change of the pixels is limited near the edges of the operational bandwidth, so performance is expected to be reduced under those conditions.

Since ΔP_2 is inherently a relative measurement, there is an implicit dependence on the initial state. Using the inactive (all 0s) state as the reference ensures the metasurface is always initialized with the same command even though the specific value is dependent on the selected frequency window. Starting with a condition where there was already a deep null would result in limited improvement; the average power in that case would already be quite low and there would not be a need for further reduction. Starting with a condition where there is a transmission peak however, would result in significant reduction. When using a single frequency band metric, we were able to drive deep nulls in each of the windows that were tested, as can be seen by the circles in Fig. 3.3a and the power at port 2 in panels b) through

e). Panels b) and d) show moderate cases where there is not a clear peak in the initial P_2 measurement, while panels c) and e) show cases with a clear peak in the initial P_2 measurement and demonstrate significant improvement. This highlights the dependence of ΔP_2 on the initial state.

Deep transmission nulls were also observed when driving two input ports simultaneously with varying relative phase shifts, as shown by the diamonds in Fig. 3.3a. This indicates our approach is self-adaptive and can compensate for multiple input signals as well as signals coming in from different directions. With dual frequency bands however, we were generally unable to drive deep nulls in both bands simultaneously, which can be seen by the hexagrams in Fig. 3.3a and the power at port 2 over the frequency band in panels f) and g). This is because the metasurface frequency response in separated bands is correlated, as the metasurface induces wide bandwidth effects on the scattering properties of the enclosure. Different choices of metrics produce different out of band behavior. These results show that our approach provides 3 distinct advantages over previous works: 1) we are able to generate coldspots at arbitrary frequencies and are not limited to a single operating frequency; 2) we are able to generate coldspots simultaneously in multiple separated frequency bands as well as at single frequencies; and 3) we are able to generate coldspots when the injected signal comes from multiple directions with an arbitrary relative phase shift and are not limited to a single direction.

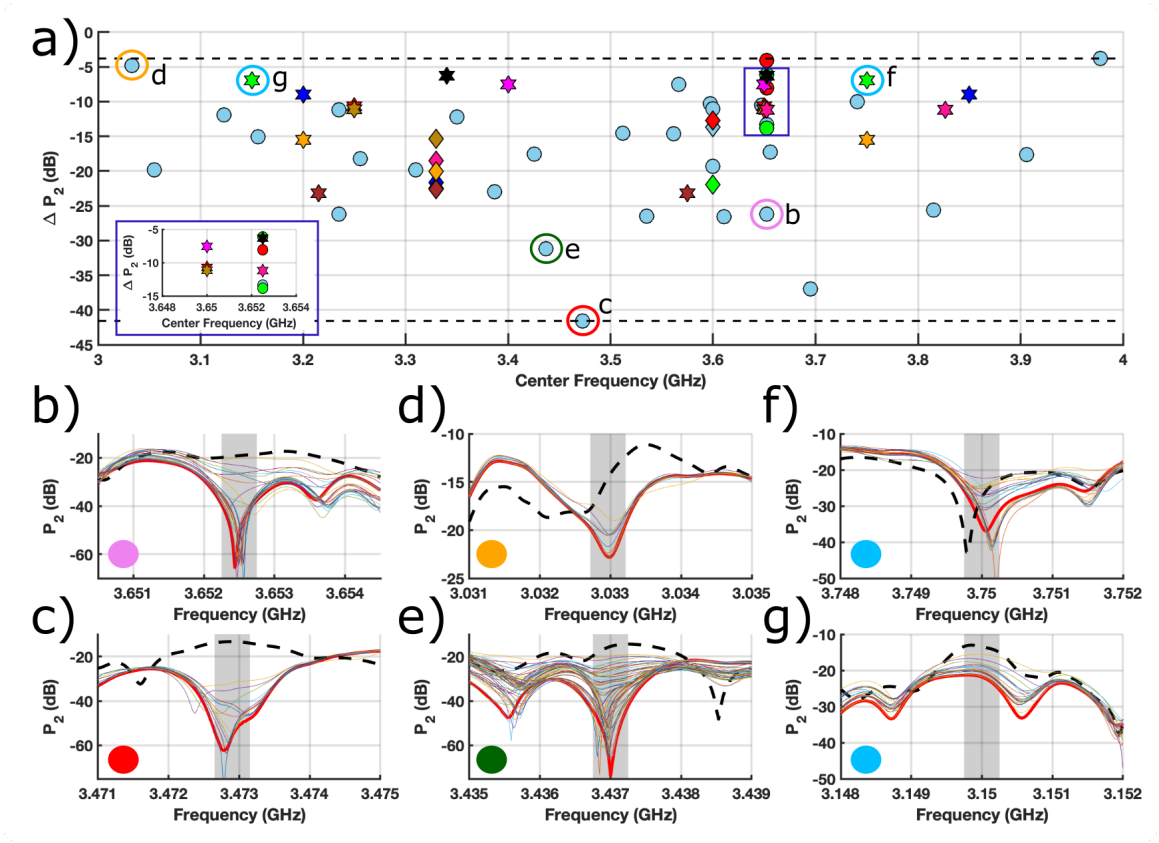

Figure 3.3: **Results of minimizing power at port 2 with the iterative optimization algorithm.** **a)** Plot of ΔP_2 at various frequencies in the range of operation of the metasurface. Circles represent cases where the metric was evaluated over a single frequency band and are color coded by bandwidth (sky blue is 500 kHz, green is 5 MHz, and red is 10 MHz). Hexagrams represent dual frequency band metrics and are color coded by matching pairs. Diamonds represent driving both ports 1 and 3 collectively and are color coded by relative phase shift (0, 8, 15, 25, and 50 deg/GHz). Letters indicate points shown in detail in the following panels. The dashed black lines indicate the smallest reduction (-4dB) and largest reduction (-41 dB). **b** through **g)** P_2 evolution from initial (all 0s) to final state. The dark shaded region represents the frequency band where ΔP_2 was evaluated, the dashed black line shows the initial response, the solid bold red line shows the final response, and the remaining lines show a few of the incremental steps. **b** and **c)** Single band examples centered at 3.033 GHz and 3.6525 GHz, with 28 and 5 dB of suppression, respectively. **d** and **e)** Single band examples centered at 3.473 GHz and 3.437 GHz, with 41 and 31 dB of suppression, respectively. **f** and **g)** Dual band example centered at 3.75 GHz and 3.15 GHz, with 7 dB of suppression averaged over the 2 bands.

3.3 Generating Coherent Perfect Absorption

CPA is a situation in which all energy injected into a system is absorbed, no matter how small the losses are in the system. Creating CPA requires coherent excitation of all the ports in an eigenvector whose corresponding S-matrix eigenvalue is zero. Operationally, the first step in establishing CPA is to find an eigenvalue of the scattering matrix that is close to zero. For example, a 2 x 2 scattering matrix will have a pair of eigenvalues at each frequency. However, realizing CPA only requires driving 1 eigenvalue to zero, as the other eigenvalue corresponds to the anti-CPA state [28]. For the following discussion and experimental results, we only consider the smallest eigenvalue of each pair.

CPA has typically been investigated in simple, regular scattering scenarios and cavities but recently it has been demonstrated in more complex systems, specifically in the realm of wave chaos, and graphs [131–134]. These works analytically demonstrate the use of RMT to explore CPA states with semiclassical tools without relying on the limit of weak coupling. CPA states have also been experimentally investigated in multiple scattering environments [27], and in graphs that break time-reversal invariance [28]. The use of enhanced spatio-temporal diversity from a metasurface for realization of CPA has not yet been explored.

Recent research however has investigated the use of metasurfaces for Perfect Absorption (PA) inside a cavity and demonstrated a secure communication system as an application [22]. PA is a complementary idea to CPA for a single port system that relies only on the reflection coefficient, S_{11} [137]. Coherent excitation of a single

port with complete absorption has been demonstrated to enhance wireless power transfer [159]. Our work extends this to coherent operation with the full scattering matrix for a 2-port system, and can be generalized to an arbitrary number of ports.

Realizing a true absolute zero of the S-matrix eigenvalues is generally difficult, because the eigenvalues are complex numbers. Thus two parameters must be varied independently to drive an eigenvalue to zero. Further, a CPA state is highly dependent on the structure of the underlying scattering system. This is best understood in the framework of the RCM [114, 115]. The eigenvalues accessible by means of the programmable metasurface tend to cluster around values determined by the coupling properties of the ports, which are characterized by the radiation S-matrix, S^{rad}. We define S^{rad} as the S-matrix corresponding to the free-space radiation condition with the cavity walls taken out to infinity such that no waves come back to the ports [94]. S^{rad} can be determined by a number of means [160]. Here we employ the ensemble average of the time gated measured S-parameters in the cavity [119], as described in Appendix B. Deviations of the scattering matrix from S^{rad} have a number of causes. First there are deviations resulting from relatively direct ray paths between the ports [110]. These deviations are removed by averaging the S-matrix over a frequency window that is the reciprocal of the time of flight on the path. However, in finding the eigenvalues of the S-matrix in a narrow frequency range, these deviations are present. Second, there are deviations due the multitude of longer paths, and these are characterized statistically by RMT within the RCM. These fluctuations in S tend to be of the order of $1/(\pi\alpha)^{1/2}$ [109, 110, 151] where the loss parameter $\alpha = f/(2\Delta f Q) = 3$ in the present experimental case. Finally,

there are deviations dependent on the state of the metasurface. These deviations are constrained to be less than or equal to either the direct path or the statistical long path deviations.

Thus, to find a CPA state it is necessary for the ports to be sufficiently matched so that the statistical fluctuations can shift the eigenvalues to zero. If the ports are poorly matched and losses within the cavity are sufficiently high, the eigenvalues will naturally fall near values determined by the properties of the ports with statistical fluctuations around those values dictated by the amount of cavity loss. As such, it is generally not possible to realize a CPA state at arbitrary frequencies when limited to a single DOF [28]. The availability of additional DOF, such as those produced by the metasurface, allows greater control over the underlying scattering system and provides a greater likelihood of potential CPA states.

Characterization of the S-matrix eigenvalues from a distribution of 2000 command sets is presented in Figures 3.4 and 3.5. Figure 3.4 shows the probability distributions for all of the S-matrix eigenvalues over all frequencies and commands. Panel a) shows that the magnitude follows a Rician distribution as predicted by Ref. [123], which also tells us that the ν parameter of the Rician distribution is due to the presence of persistent short orbits [95]. Panel b) shows that the phase of the S-matrix eigenvalues is not truly uniformly distributed. The deviation of the eigenphase from uniformity indicates that the random distribution is not statistically independent and again tells us there are persistent short orbits present in the system. These short orbits are not captured explicitly in S^{rad}, and will cause the eigenvalues of S^{rad} to be offset from the center of the point cloud of S-matrix

eigenvalues. Short orbits can be explicitly included analytically in the RCM [95,118] and do not prevent us from proceeding. Panel c) shows the cumulative distribution function (CDF) of the eigenvalue magnitudes and is useful in establishing thresholds for potential CPA candidates.

Figure 3.5 shows point clouds of the S-matrix eigenvalues at selected frequencies and demonstrates that the eigenvalues can have very different behavior in how they approach the origin. The panels show the collection of eigenvalues of the 2000 S-matrices at selected frequencies along with the eigenvalues of S^{rad} at that frequency. We can see that the eigenvalues of the distribution tend to cluster around the eigenvalues of S^{rad}; the offset from the center of the point cloud is due to the presence of short orbits, as discussed above. In panel a), the S-matrix eigenvalues are clustered in the upper right quadrant far from the origin and do not enter the inner rings. The S^{rad} eigenvalue is in the upper right-hand quadrant outside of the plot area, at $0.1862 + j0.2288$. In panel b), the S-matrix eigenvalues are clustered in the upper half, with some getting close to the origin. In panel c), the S-matrix eigenvalues show a fairly uniform density throughout the full $|\lambda_s| < 0.15$ range. In panel d), the S-matrix eigenvalues show a high density clustered around the origin. The results in these panels are from a random distribution of commands rather than a targeted search. During optimization, we will take smaller dithering steps for finer control as we approach the origin, and expect to see slightly different behavior.

The variance in eigenvalue magnitudes means we need to use a large threshold for identifying candidates because the overall global minimum S-matrix eigenvalue may not be identified as a candidate in every realization. In practice, we found that

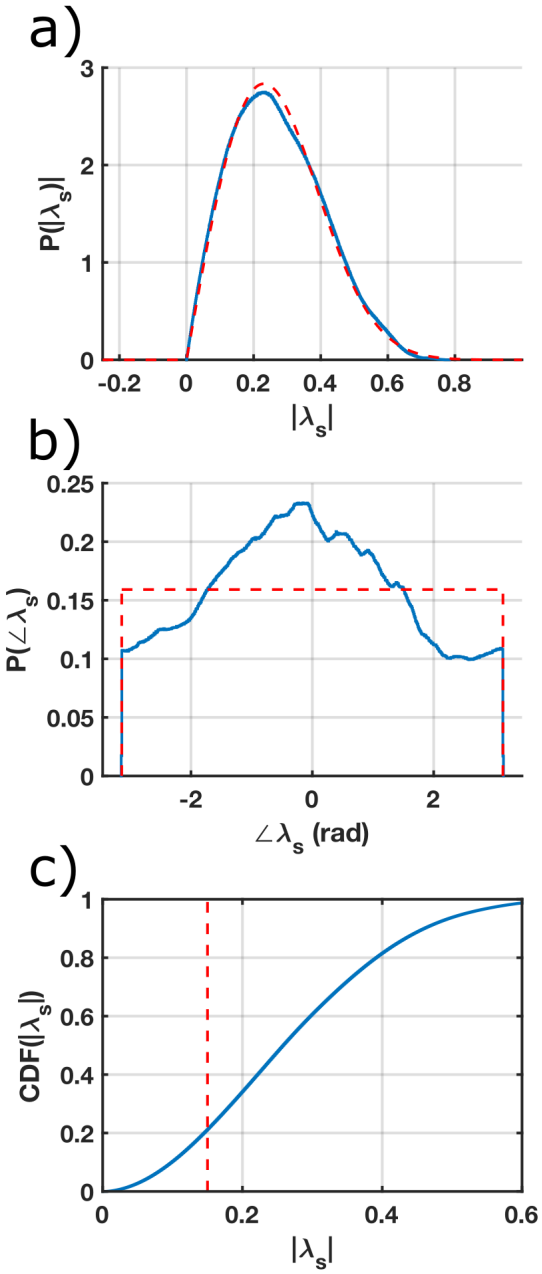

Figure 3.4: ***S*-matrix statistics for a random distribution of 2000 metasurface commands.** Panels show statistics for the scattering matrix eigenvalues covering all commands and all frequencies from 3-4 GHz. **a)** PDF for the magnitude of scattering matrix eigenvalues, $|\lambda_s|$. The dashed red line shows the fit to a Rician distribution with $\sigma = 0.173$ and $\nu = 0.177$. **b)** PDF for the phase of scattering matrix eigenvalues, $\angle\lambda_s$. The dashed red line shows the distribution for a perfectly uniform phase. **c)** CDF for the magnitude of scattering matrix eigenvalues. The dashed red line shows the threshold of $|\lambda_s| < 0.15$.

we were unable to realize CPA states when starting with a magnitude $|\lambda_s| \geq 0.2$ but were generally able to realize CPA states when starting with a magnitude $|\lambda_s| \leq 0.15$. Moving an eigenvalue far from the origin requires modifying the underlying scattering matrix more strongly than moving an eigenvalue that is already near the origin, so this behavior is expected. Assessing the probability of finding a CPA state in a given frequency range a priori is difficult. The universal properties of a complex scattering system are not easily separated from the deterministic properties when working with S-parameters, as the statistics are dominated by S^{rad} [93]. This means the existence of a CPA state is highly dependent on the coupling properties of the ports and therefore the specific antennas chosen. An analytical approach is possible through the framework of the RCM and will be left to future work.

An open question is how small do the eigenvalues need to be to realize CPA? This is dependent on the specific application and scattering system, as that determines how accurately the eigenvalues can be measured and maintained. For our experimentation, we set $|\lambda_s| \leq 5 \times 10^{-3}$ as the upper bound and $|\lambda_s| \leq 1 \times 10^{-3}$ as the goal for realizing CPA.

We adopt the same basic algorithm used for power minimization but initialize it differently. We apply a random set of commands to the metasurface and then select a candidate eigenvalue with a specified magnitude. Figure 3.6 presents the results of 27 separate CPA eigenvalue optimization experiments. Fig. 3.6a shows the behavior of 4 selected cases away from the origin for $|\lambda_s| < 0.15$, and Fig. 3.6b shows the behavior at the CPA condition for $|\lambda_s| < 5 \times 10^{-3}$. Only 3 of the 4 selected cases reach the CPA threshold. Fig. 3.6c presents the collection of all 27

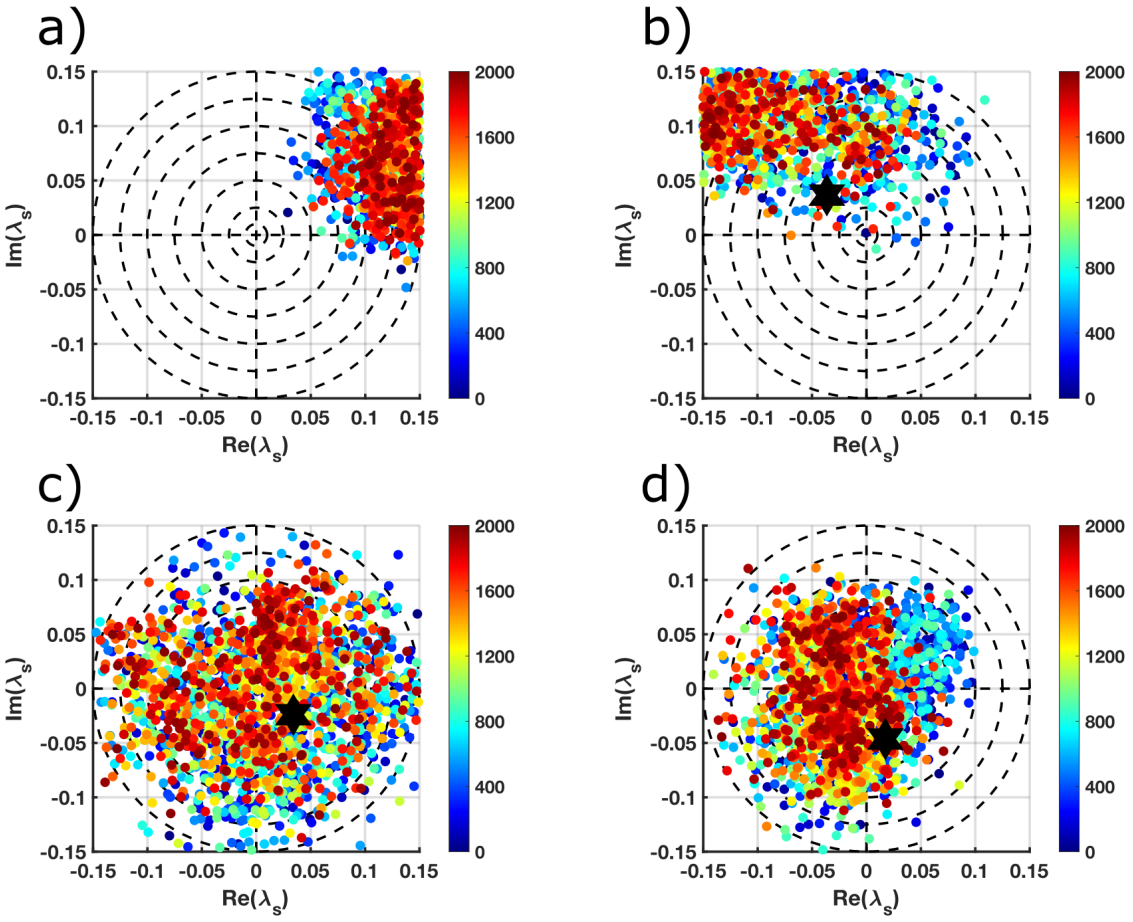

Figure 3.5: **Point clouds of selected S-matrix eigenvalues for a random distribution of 2000 metasurface commands.** Panels show the point clouds of the smaller eigenvalues of the 2000 scattering matrices at 4 selected frequencies. The colored circles are the S-matrix eigenvalues, and are color coded by the specific command, from 1 to 2000. The large black hexagrams indicate the position of the eigenvalue of S_{rad}. **a)** Candidate at $f = 3.0055$ GHz, minimum $|\lambda_s| = 6 \times 10^{-2}$. **b)** Candidate at $f = 3.4021$ GHz, minimum $|\lambda_s| = 2 \times 10^{-3}$. **c)** Candidate at $f = 3.6564$ GHz, minimum $|\lambda_s| = 5 \times 10^{-3}$. **d)** Candidate at $f = 3.9991$ GHz, minimum $|\lambda_s| = 5 \times 10^{-4}$.

experiments, with the four shown in detail in panels a) and b) color coded. Each case was initialized with an eigenvalue magnitude chosen in the range $0.075 \leq |\lambda_s| \leq 0.5$. The case that started with $|\lambda_s| = 0.5$ is enclosed by a triangle, the cases that started with $|\lambda_s| = 0.2$ are enclosed by squares and the case that started with $|\lambda_s| = 0.175$ is enclosed by a circle. All the rest started with $|\lambda_s| \leq 0.15$. Three cases initialized with $|\lambda_s| = 0.15$ did not quite make the CPA threshold, $|\lambda_s| \leq 5 \times 10^{-3}$. Two cases were within a factor of 2, $|\lambda_s| \leq 9 \times 10^{-3}$, while the third was within $\sim$20%, $|\lambda_s| = 6 \times 10^{-3}$.

Utilizing the iterative optimization algorithm to change the metasurface, we are able to drive eigenvalues towards the origin in all cases, but the algorithm stalls at different points. The closer we get to the origin, the more difficult it becomes to reduce the eigenvalue further. As with the coldspot optimization, the stochastic nature of the algorithm plays a role in where convergence is reached. The overall performance could be improved by increasing the convergence criteria or making the algorithm adaptive so that it tracks multiple candidates and switches to another candidate when the optimization stalls.

As a final step, we want to verify that the CPA state has been achieved. Because the CPA state is found by minimizing the eigenvalues of the scattering matrix, verification requires that we apply the corresponding S-matrix eigenvector. This can be done using a network analyzer with 2 independent sources and an external phase shifter [28]. After directing a particular eigenvalue towards the origin, the network analyzer was configured for independent source operation and the amplitude and phase were adjusted to generate the eigenvector, as described in Appendix E.4. The

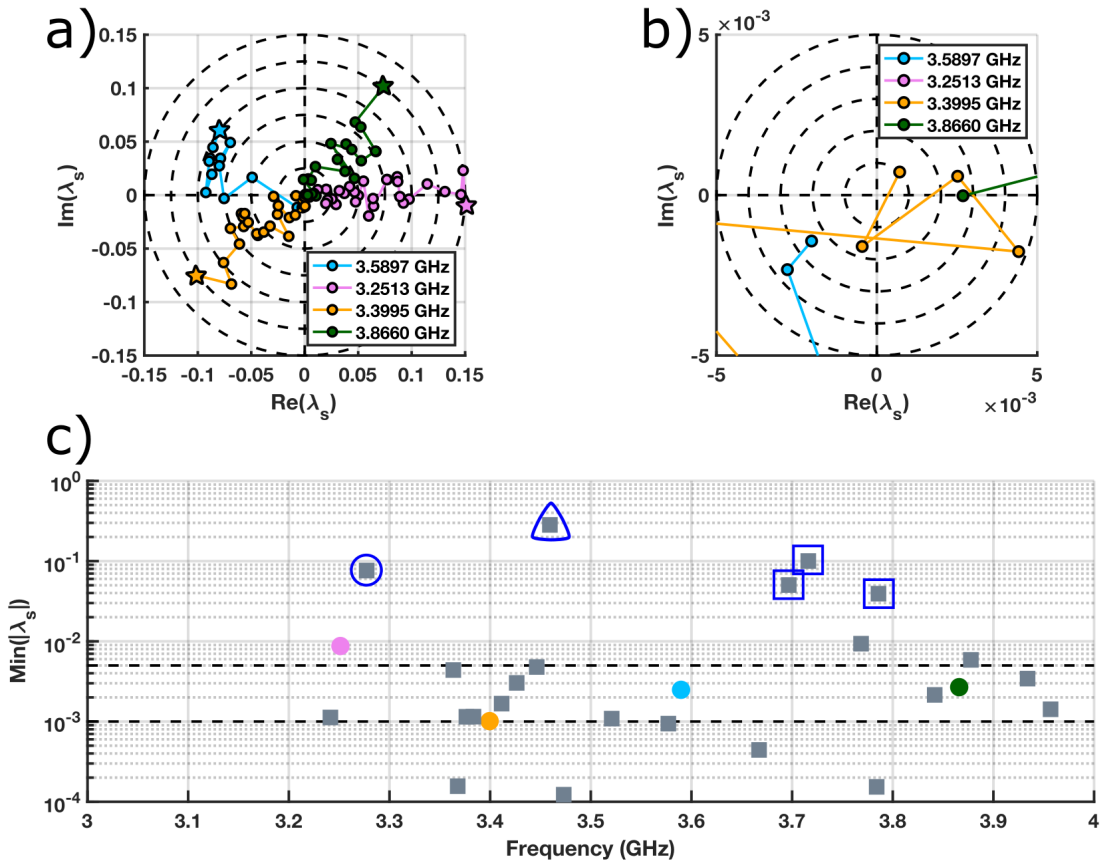

Figure 3.6: **Experimental S-matrix eigenvalue trajectories for realization of Coherent Perfect Absorption (CPA) states. a)** and **b)** Directed trajectories for eigenvalues showing the random walk nature of the algorithm and demonstrating mobility of selected eigenvalue candidates. **a)** Zoomed out view showing selection of initial S-matrix eigenvalue candidates and behavior away from the origin, the bulls-eye circles are spaced at radii incrementing by 2.5×10^{-2}. The starting eigenvalue magnitude in each case is identified by a star. **b)** Close up view showing behavior near the origin, the bulls-eye circles are spaced at radii incrementing by 1×10^{-3}. Of the four cases shown, only 3 were able to get inside the inner rings near the origin where $|\lambda_s| < 5 \times 10^{-3}$. **c)** Minimum achieved eigenvalue magnitude for each performed experiment. The circles indicate data that is shown in the upper plots and are color coded to match. The gray squares indicate an experiment that was performed but whose detailed trajectory is not shown in the upper plots. The dashed black lines indicate the cross over points of 5×10^{-3} and 1×10^{-3}. The enclosed squares indicate cases where the initial eigenvalue magnitude $|\lambda_s| > 0.15$. $|\lambda_s| = 0.5$ for the triangle, $|\lambda_s| = 0.2$ for the squares, and $|\lambda_s| = 0.175$ for the circle.

presence of a CPA state is verified by looking at the ratio of all the power emerging from the cavity to all the power injected into the cavity, $P_{\text{out}}/P_{\text{in}}$. Sensitivity to changes in the eigenvector can be determined by making small deviations in the relative phase shift or amplitude between the two sources. Sensitivity to the eigenvalue can be determined by small changes in frequency.

A set of parameter sweeps that verify a CPA state was realized are presented in Figure 3.7. Fig. 3.7e shows the S-matrix eigenvalue magnitude trajectory during optimization prior to performing the verification sweeps. The overall experimental setup is shown in Fig. 3.7f, which shows that a 2-source network analyzer was configured with independent source operation and connected to the cavity with an external phase shifter on port 1. This allows us to produce the appropriate eigenvector by controlling the relative amplitude with the network analyzer and the relative phase with the phase shifter. The metric for the sweeps is the power ratio, $P_{\text{out}}/P_{\text{in}}$, of all the power emerging from the cavity to all the power injected into the cavity. At the CPA condition, all the energy should be absorbed. However, due to instrumentation limitations with the system noise floor, the smallest measurable power ratio is $\sim 10^{-6}$. Before performing the sweeps, the eigenvector was tweaked to provide the closest CPA state realization and then the parameters were varied to determine the sensitivity of the power ratio.

Fig. 3.7a shows the results of the frequency sweep performed in a $\pm$ 10 MHz window around 3.6697 GHz, with the inset showing a closeup in a $\pm$ 200 kHz window. The width of the deep null is $\sim$200 kHz, which matches the null widths found during cold spot generation. Fig. 3.7b shows the results of the phase sweep,

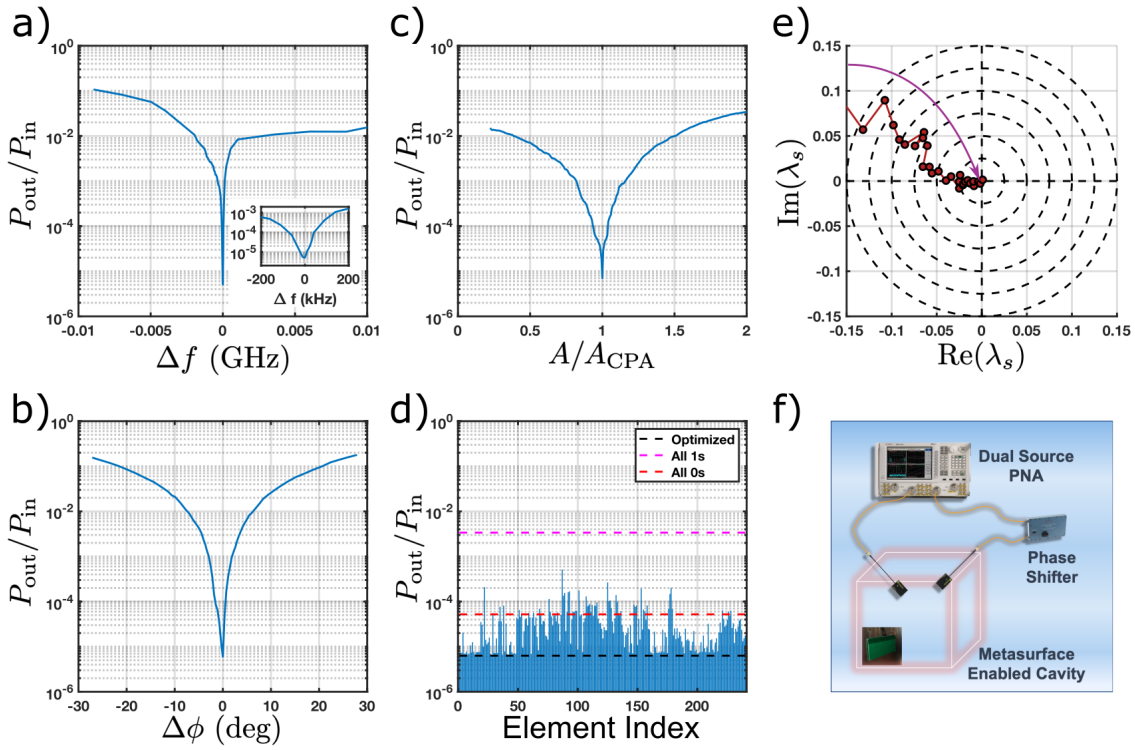

Figure 3.7: **Coherent Perfect Absorption (CPA) state verification at 3.6697 GHz. a)** Frequency sweep showing the power ratio, $P_{\text{out}}/P_{\text{in}}$, over a $\pm$ 10 MHz window, with the inset providing a closeup of the null in a $\pm$ 200 kHz window. **b)** $P_{\text{out}}/P_{\text{in}}$ vs. phase difference, $\Delta\phi$, showing the power ratio over a $\pm$ 30° window. **c)** $P_{\text{out}}/P_{\text{in}}$ vs. relative amplitude showing the power ratio when driving port 1 with an amplitude ~0-2 times the CPA amplitude (A_{CPA}). **d)** Metasurface command sweep showing the power ratio when toggling individual elements relative to the optimized set. Each bar indicates the power ratio when that particular element was flipped between a 1 or a 0. The black dashed line shows the power ratio for the optimized state, the red dashed line shows the power ratio for the all 0s state, and the magenta dashed line shows the power ratio for the all 1s state. **e)** Eigenvalue magnitude trajectory during optimization of the CPA state prior to performing the verification sweeps. Minimum achieved $|\lambda_s| = 4 \times 10^{-4}$. **f)** Diagram showing experimental setup for applying CPA eigenvector excitation and verification sweeps.

which was performed by adjusting the external phase shifter. Here, $\Delta\phi$ represents the phase shift at port 1 away from the CPA eigenvector phase. Fig. 3.7c shows the results of the relative amplitude sweep. This was performed by sweeping the power injected into port 1 from -10 dBm to +10 dBm. The x-axis is then scaled to show the relative change in injected amplitude from the initial CPA state. In each of these cases, the minimium power ratio is $\sim 6 \times 10^{-6}$ and shows a steep cusp-like increase with the various parameters. Fig. 3.7d shows the results of the metasurface command sweep. In this case, the 240 individual metasurface elements were toggled to determine the impact of a single element on the CPA state. Several elements had negligible impact on the power ratio in comparison with the optimized value as seen in the dashed black line, but no toggles were found with clearly better performance. The elements in the center of the metasurface have a stronger impact than those at the edges of the metasurface, but the largest change from the CPA condition was observed by setting all the elements to 1s as shown in the dashed magenta line.

Chapter 4: Deep Learning Approach

In this chapter, we again use a binary programmable metasurface to shape radio frequency electromagnetic waves inside a chaotic microwave cavity, and present a deep learning network that solves the wavefront reconstruction problem, enabling real-time operation once trained. We emphasize that our method is enabled by the use of a reverberant environment, which allows the metasurface to interact with multiple ray trajectories, often more than once. A reverberant environment provides two major capabilities that are not present in non-reverberant environments: 1) the ability to control the distribution of wave fields at arbitrary regions inside the cavity is enhanced. This allows the use of relatively small metasurfaces, e.g., in our configuration, the metasurface covers only $\sim$1.5% of the total surface area of the cavity; and 2) the requirement on establishing a line-of-sight path between the metasurface and the ports is removed, which allows the location of the metasurface to be arbitrarily chosen, increasing the flexibility and versatility of the approach. We anticipate that realization of this concept, as shown in Fig. 1.1, will help usher in the new era of smart radio environments, as well as allow on-demand creation of microwave cold spots to protect sensitive electronic components and coherent perfect absorption states for wireless power transfer.

4.1 Wavefront Control in Reverberant Environments

Microwave experiments have shown that programmable metasurfaces can provide fine control over the scattering parameters of a cavity, with the most recent work demonstrating perfect absorption [22] and coherent perfect absorption [24, 25] states inside the cavity. The relationship between metasurface commands and cavity scattering parameter responses is extremely complicated (there are 10^{18} possible configurations of the metasurface in our case). Therefore, optimization of the metasurface is typically handled through brute force trial and error or stochastic search algorithms [24, 96, 97]. As discussed in Sections 2.2 and 2.3, rapid and accurate wavefront reconstruction techniques that solve the inverse problem between measurements and metasurface commands are necessary to realize practical intelligent wavefront shaping systems. Conventional methods fall apart in complex scattering environments with binary metasurfaces; however, the inherent complexity makes it an ideal place to utilize deep learning. Ma et al. explored the use of deep learning networks with wave chaotic systems, demonstrating the ability to successfully distinguish between different types of wave chaotic cavities through the measured S-parameters [161]. We now tackle a more difficult problem, quickly identifying a set of metasurface commands required to achieve a specific wave scattering requirement, even for cases where that set of commands has not been previously encountered.

In 2018, artificial neural networks were demonstrated to be applicable to solving inverse scattering problems in electromagnetics [162]. Since then, deep learning has been successfully used to design metasurfaces for wavefront shaping applica-

tions in both the photonic and microwave domains [163–173]. However, most of the publications so far have focused on designing and arranging the individual unit cells of the metasurface for static use cases. Active deep learning approaches with programmable metasurfaces have been demonstrated for microwave imaging applications [174–179]. Li et al. used a two-bit coding metasurface to generate radiation patterns for a machine learning algorithm that detects and classifies human movement [174, 175]. del Hougne et al. started with a pair of metasurfaces as a transmitter and receiver to feed a dense neural network that detects and classifies objects in a learned integrated sensing paradigm [176, 177]. Further research by this group used a dense neural network to classify the position of a scattering object inside a complex cavity with a metasurface acting as a coded aperture [178]; this work was recently extended to predict a continuous position with sub-wavelength precision [179].

These examples demonstrate how a programmable metasurface can enhance the processing power of a deep learning network for microwave imaging, but they do not leverage the deep learning network for wavefront reconstruction. This is a key component of intelligent wavefront shaping, which has so far been an underexplored area of research. Qian et al. used a simple dense network to enable cloaking of an object [63], while Shan et al. used a 2D convolutional network to optimize the steering of multiple beams [180]. Both cases utilize an idealized testing environment inside an anechoic chamber, where multi-path reflections from the environment are intentionally excluded. In addition, both cases are built around a propagation path with a direct reflection off the metasurface, which means that the metasurface in-

teracts with virtually all ray trajectories from the source to the receiver.

As discussed in Section 2.3, a single propagation path eliminates redundancies from persistent short orbits, reducing the measured correlation between metasurface configurations. These cases can be treated with more traditional system identification techniques or simple neural network models. When the metasurface is placed inside a complex reverberant scattering volume [178, 179], determining the relationship between metasurface commands and scattering responses becomes substantially more difficult due to the presence of multiple scattering paths. A reverberant scattering system is qualitatively different from an open system. It is characterized by extreme sensitivity to initial conditions [108, 113], and the fluctuations are only well represented by Rayleigh or Rician distributions under high loss conditions. This means accurate wavefront reconstruction must account for chaotic behavior and be sensitive to small environmental changes, as well as handle non-negligible large amplitude signal spikes that include phenomena such as rogue waves [181]. This difficulty is further compounded as we wish to optimize the metasurface response over a wide bandwidth or even over multiple separated bandwidths simultaneously. Finally, real world chaotic systems contain short orbits, or prompt direct paths that do not ergodically sample the enclosure [95, 118]. Short orbits are persistent and manifest as correlations between realizations.

The reverberating nature of the cavity enables operation with a smaller metasurface than would be possible in a non-reverberating environment. Longer reverberation times (lower cavity losses) mean that the rays will survive longer in the cavity, resulting in more reflections from scattering objects and more ray trajecto-

ries that interact with the metasurface, often multiple times. Longer reverberation times then provide the metasurface more flexibility in controlling constructive and destructive interference at the ports, allowing for larger relative changes when toggling metasurface states. We demonstrate this to be the case and show that the performance of the deep learning network degrades as the losses in the cavity increase because the metasurface has a smaller relative impact on the S-parameters. This is another distinction between a reverberant environment and an open one, where environmental losses only impact the signal magnitude through absorption.

We further show that our trained network can successfully determine the metasurface configuration from the measured scattering response in the cavity several days after the training data was collected. Measured S_{21} responses with the same initial conditions inside a chaotic cavity will change over time, a phenomenon known as scattering fidelity decay [126–128]. Scattering fidelity decay is a property of wave chaotic systems, and is sensitive to boundary conditions and the scattering environment. This is in contrast to ray chaos and the sensitivity of bouncing ray trajectories to initial conditions in billiards. This decay means that any deep learning system that learns scattering responses inside a chaotic cavity will require periodic retraining. As discussed in Chapter 5, the fact that we are still able to determine the metasurface configuration accurately after several days means our technique is operationally useful, as it can function at a high level of accuracy for a long period of time before requiring retraining. Our approach is robust and highly accurate in determining metasurface commands from measured cavity S_{21} spectra, providing an enabling capability for intelligent wavefront shaping applications. In addition, our

method is general enough to operate in arbitrary complex scattering systems and does not requires a specifically engineered environment.

Our technique is achieved through the development and combination of four major novel aspects: 1) adaptive configuration of the metasurface unit cells by binning elements together to dynamically alter the relative size of the elements; 2) representation of the complex system S-parameters in a pseudo-2D "image" to promote extraction of features that are correlated over both local and global frequencies; 3) complex-valued deep learning layers to exploit both phase and amplitude information, accelerating training and improving the accuracy when applied to complex scattering environments; and 4) introduction of the Terrapin Module to parallelize the deep learning network, promoting sparse feature representation and improving training robustness.

Future efforts will refine our technique to intentionally scramble (or unscramble) waves propagating through a complex scattering environment. Three aspirational goals include: 1) tuning the scattering responses through a controller that optimizes the system for a given application at arbitrary frequencies and bandwidths. Specific metrics include minimizing transmitted power for coldspot generation, minimizing scattering matrix eigenvalue magnitudes for coherent perfect absorption, or minimizing the bit error rate for wireless communication; 2) introducing feedback from the environment to dynamically update the controller and react to changing environmental conditions; and 3) realizing a fully autonomous system that enables persistent and robust smart radio environments that do not require human intervention.

4.2 Experimental Configuration

The complex, ray chaotic cavity used for experimentation has a binary programmable metasurface installed on an interior wall. It has a volume of 0.76 m^3 and is in the same configuration used in our previous work [24], as discussed in Appendix F.1 and Fig. F.1. The metasurface was fabricated by the Johns Hopkins University Applied Physics Laboratory and is designed to operate in the frequency range of 3-4 GHz. It contains 240 binary meta-atoms (LC resonators) arranged in a rectangular grid of 10×24 elements. Each element has a characteristic length of $\sim \lambda/6$ and is switched by a GaAs transistor amplifier to 1 of 2 states (0 or 1), changing the phase of the reflection coefficient by $\sim 180^\circ$ [30]. The metasurface covers a small region of the interior surface area of the cavity, 1.5%, so only a limited number of rays are intercepted by the metasurface.

The goal of our deep learning network is to enable wavefront reconstruction inside a complex cavity. The network will accept a given S_{21} spectra from 3-4 GHz and accurately determine the metasurface commands necessary to closely realize that specific scattering response. The relatively small size of the metasurface and its unit cells leads to high correlation between system scattering responses with minor changes in metasurface commands, which means the inverse problem is ill-posed. Deng et al. recently introduced a neural-adjoint approach for solving the ill-posed inverse problem of designing unit cell geometries to match specified absorption spectra [107]. In this case, a fully connected deep learning network was used to model the forward problem, acting as a Green's function to predict the spectrum from a

given design. The inverse problem was then solved iteratively, driving the design along an estimated gradient towards the optimal solution. As discussed in Section 2.2, gradient methods work best for a continuous or near continuous solution space rather than a binary one such as ours; however, the adjoint method from [107] can be adapted into a reinforcement learning approach as discussed in Section 5. In addition, inside a chaotic reverberating environment, the spectra will have more structure, resulting in higher frequency oscillations or local features that must also be learned. Therefore, we require a different deep learning approach.

The complexity of the cavity scattering responses combined with the enormous number of possible metasurface command configurations (2^{240}) means the direct development of a deep learning network for the full space of 240 elements is overly ambitious. To simplify the problem, we reduced the number of degrees of freedom of the metasurface by binning together neighboring pixel elements, or grouping them together so that each element in a group is always commanded with the same value. Binning the metasurface elements reduces the total number of elements that must be determined and strengthens the relative change in cavity scattering parameters when driving a single effective element. Binning also promotes generality, as a metasurface with smaller elements can always approximate one with larger elements. This provides the first major novel aspect of our approach and allows us to explore the use of deep learning models in simpler configurations before working our way up to the more difficult cases. We used 4 different metasurface binning configurations, as shown in Fig. 4.1 and discussed in Appendix F.2, to progressively decrease the number of elements.

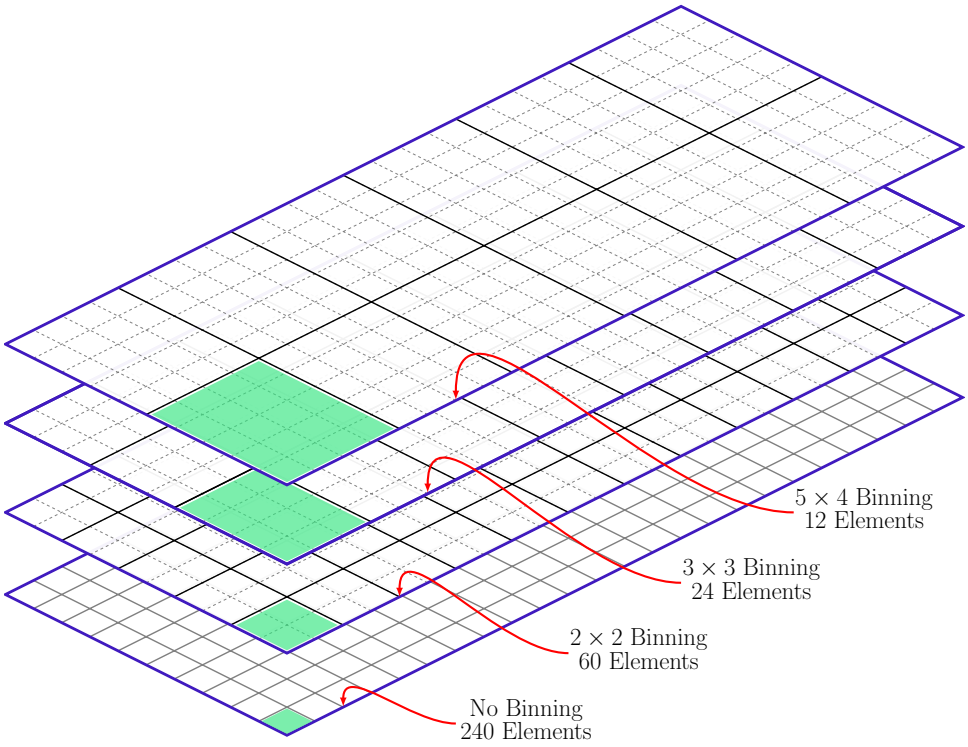

Figure 4.1: **Metasurface binning configurations.** Binning configurations showing the relationship between the various options. The shaded green region identifies a single effective element for the specified configuration and the thin gray lines show the layout of the unbinned elements. With no binning, there are 240 elements, binning into groups of 2×2 yields 60 elements, binning into groups of 3×3 yields 24 elements, and binning into groups of 5×4 yields 12 elements. For the 3×3 binning case, the bottom row of elements consists of a 4×3 group so that all the elements are utilized.

An important step for deep learning is preparation of the measured data. As discussed in Appendix F.3, the raw data consists of M sets of complex two-port S-parameter values, each containing 32,001 points measured over a 3-4 GHz window. We use a pseudo-2D "image" to capture both long-range and short-range correlations in frequency, as shown in Fig. F.2. This represents the complex S_{21} measurements in a basis set convenient for deep learning networks. The pseudo-2D format provides the second major novel aspect of our approach.

The primary limitation is that we are not guaranteed to be able to generate any arbitrary S_{21} response, as a configuration of the metasurface that produces that response does not necessarily exist. The small size of the metasurface relative to the cavity limits its ability to interact with all possible ray trajectories, emphasizing the importance of a binning capability to adapt the effective pixel size to the environment. This limitation is therefore a function of the system configuration, and not the deep learning network. The small relative size of the metasurface does represent a realistic configuration for practical smart radio environments, however.

We show that the deep learning network achieves an accuracy exceeding 95%. This high success rate is achieved with a limited amount of training data, requiring the collection of far fewer sets than the number of possible combinations of metasurface commands. To be specific, we have a dataset of 10^4 S_{21} measurements, each containing 32,001 frequency points, whereas there are 10^{18} possible configurations of the metasurface. The data sets are split into 75% training data and 25% validation data. At each step (epoch), validation of the trained model is performed by testing the model with a new set of data not present in the training set.

Training is performed in a parallelized fashion over all the training data at once, e.g., for our computational resources, taking $\sim$ 4 hours to collect a sufficient set of data and train the deep learning network. Testing, however, is performed on single shot measurements and takes less than 1 second to measure the S_{21} response and make a determination of the metasurface commands, enabling real-time operation. In contrast, the iterative approach in our earlier work [24] required $\sim$300 measurements to converge to a desired configuration, taking $\sim$10 minutes to reach the answer for each configuration. Online iterative optimization does not require training, but may produce undesirable configurations due to the randomly applied perturbations. When time is available for offline training, the deep learning approach is preferred.

4.3 Deep Learning Network Design

The goal was to design a deep learning network that provided the best performance for the wavefront reconstruction problem, and determine metasurface commands that approximately realize the desired complex transmission coefficient, S_{21}, vs. frequency (3-4 GHz). A discussion of the different types of neural network layers used and the overall training approach is provided in Appendices F.4 and F.5.

The added complexity resulting from placing the metasurface inside a chaotic cavity requires a correspondingly complicated deep learning network to extract the relevant features. Rather than only designing progressively more intricate network topologies, we can also introduce complex-valued layers [182], which serve as the third major novel aspect of our approach. The wave scattering phenomenon is

fundamentally complex-valued, so using complex-valued layers allows the network to better match the underlying physical system and exploit both phase and amplitude information. This is shown to be true for our system in Appendix F.6 and Fig. F.3, which further demonstrates that the real-valued networks are more sensitive to tuning parameters than the complex-valued networks and ultimately do not reach the same level of accuracy for extremely complicated scattering systems. Complex-valued multiplicative layers have been used to invert propagation through multi-mode fibers [183, 184], but, to the best of our knowledge, have not previously been used for wavefront reconstruction. Unfortunately, as discussed in Appendix F.6, there are no officially supported complex-valued modules in any of the major deep learning frameworks. Multiplicative layers are straightforward to implement, but more complicated modules, such as convolutional layers, are not. For the research described here, we leveraged the open source complexPyTorch library [185] as the basis for our complex-valued network layers.

As described in Appendices F.7 and F.8, the 5×4 binning case performed extremely well using a straightforward sequential CNN architecture. After training, we were able to accurately realize 100% of the target responses over both the training and validation sets. Unfortunately, the purely sequential architecture of the network did not work as well for the 3×3 binning configuration (see Appendix F.10). The increased complexity implies that we need a more complex network, so we turned to approaches successfully used in modern image classification, specifically inception modules [186, 187]. As discussed in Appendix F.9, we modified the general architecture to perform 1D convolutions over the 10 MHz local frequency windows. The 1D

convolutional filters then extract local features over the 10 MHz windows, while the relationship between the filters acts as a dense or fully connected layer, extracting global features over the full 1 GHz measurement window. The final version, which we refer to as a "Terrapin Module", is shown schematically in Fig. 4.2, and provides the fourth and final major novel technical contribution of our approach.

With a deep learning network containing 4 Terrapin Modules in series, we were able to get excellent performance for the 3×3 binning configuration with only 4,000 sets of training data, as discussed in Appendix F.10. The 2×2 configuration required 10,000 sets of data for a similar level of performance (see Appendix F.11). The smaller effective elements in this configuration produce responses with a larger degree of correlation. Thus, more data is required for the network to learn and distinguish the more subtle relationships between metasurface command configurations and scattering matrix responses, $S_{21}(f)$.

4.4 Results

The primary objective of this work is to demonstrate that deep wavefront shaping is a viable technique for wavefront reconstruction inside complex scattering environments, enabling intelligent wavefront shaping in a chaotic cavity. In this section, we shown how our deep wavefront shaping approach accurately determines metasurface commands from measured cavity scattering parameters.

Training results for the 5×4 and 3×3 binning configurations are provided in Appendices F.8 and F.7, while training results for the 2×2 binning configuration are provided in Appendix F.11 and shown in Fig. 4.3. The training data consists

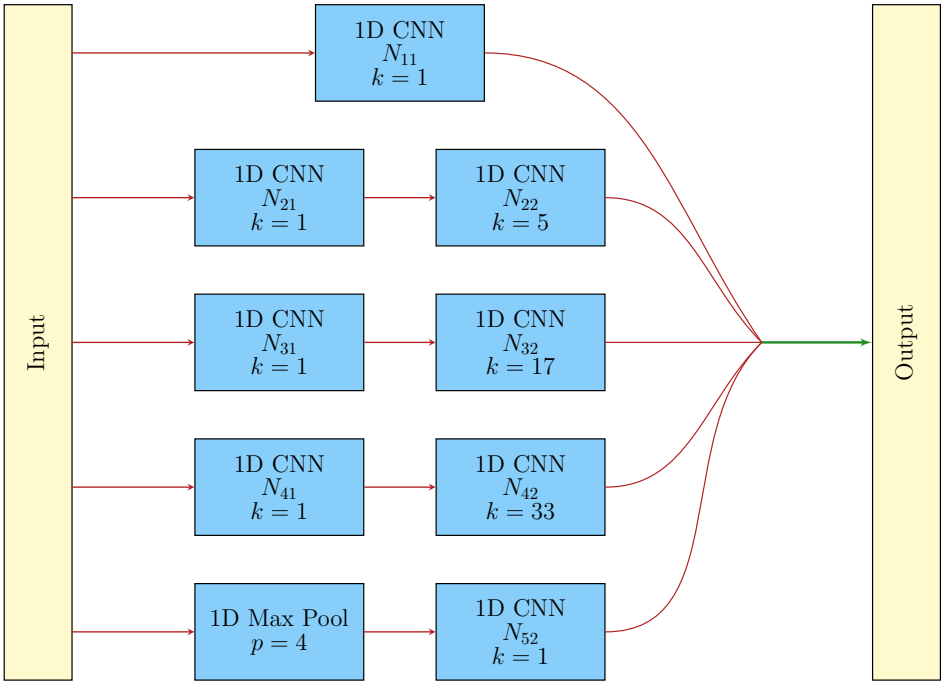

Figure 4.2: **Terrapin Module Architecture.** Five parallel branches with 8 1D convolutional neural network (CNN) layers and a max pool layer are used in the module. The module operates on the pseudo-2D data format discussed previously, and the input layer can ingest either the raw measured S_{21} parameters or the outputs from a previous Terrapin module. The output layer is then connected either to a subsequent Terrapin module for additional processing or to the final output layer for conversion to metasurface commands. Each CNN includes a 1D convolution, a batch normalization, and a rectified linear unit activation function. The 2nd level CNNs have kernel lengths of 5, 17, and 33 to increase the receptive field by 125 kHz, 500 kHz, and 1 MHz, respectively. A 1D max pooling layer with pool size of 4 is included to provide a pooling window of 125 kHz as well. The quantity N_{xx} indicates a tunable parameter for the number of convolutional filters at each branch and stage, acting as a dense or fully connected layer for the global correlations. The convolutions with unit length kernels serve to buffer and condition the inputs to each stage, and the single layer 1st branch maintains the receptive field sizes from previous modules. The outputs of each branch are concatenated together to form the module output, preserving the receptive field sizes for subsequent layers.

of 10,000 random realizations of metasurface commands, representing an extremely small fraction of the 1.2 $\times 10^{18}$ possible configurations. The data was split into 7500 sets for training and 2500 sets for validation to ensure the validation process is unbiased. The evolution of the loss function is shown in Fig. 4.3 (a) and the evolution of the accuracy is shown in Fig. 4.3 (b). The loss function was chosen as the mean absolute error (MAE), or the $\mathcal{L}_1$ norm, between determined and actual metasurface commands. Accuracy is the fraction of sets that was determined without error and provides a more conservative estimate of performance. The variation around Epoch 45 is due to choosing an aggressive initial learning rate and the impact of reducing the learning rate on a plateau can be seen at Epoch 55. These panels demonstrate that we were able to obtain high accuracy and a small loss function for both the training and validation data sets.

The trained model did not have perfect accuracy but was able to determine 2,443 out of 2,500 sets in the validation data without error for an accuracy of 97.7%. One set had 2 errors and 56 sets had a single error, as shown in Fig. 4.3 (c). A comparison of the determined and true commands is shown in Fig. 4.3 (d), (e), (g), and (h). These panels show that for the worst case set with two errors, the network was not highly confident in the results as the erroneous determined command probabilities were 0.41 and 0.73. Finally, example scattering responses are shown in Fig. 4.3 (f) and (i), which demonstrate both the complexity of the S_{21} responses and the fact that the difference between the measured and predicted responses are $\sim$20 dB lower than the signal magnitudes themselves.

To further validate the trained deep learning network, we adopted the on-line,

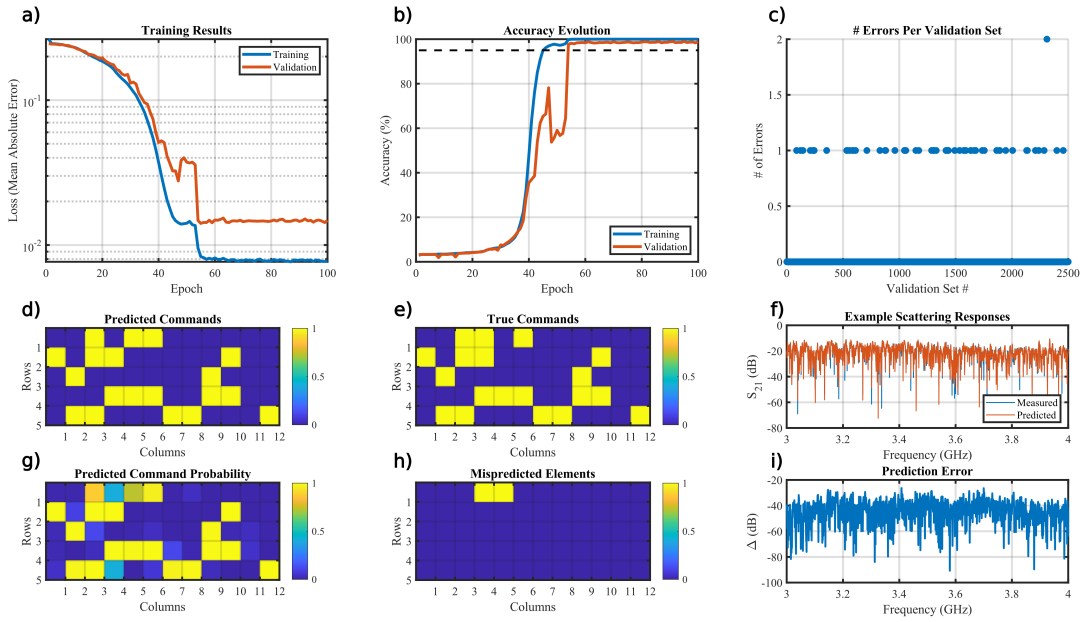

Figure 4.3: **Deep learning performance with complex-valued layers for 2x2 binning.** **(a)** Evolution of the loss function for the training and validation sets over 100 epochs. The learning rate is reduced at Epoch 55, inducing an additional reduction in the loss function. The initial learning rate was aggressive, resulting in a large variation in the validation loss function between Epochs 45-50. **(b)** Evolution of the accuracy for the training and validation sets over 100 epochs, the dashed black line identifies 95% accuracy. **(c)** Number of errors over the validation set for the trained model, showing a total of 58 errors and 97.7% accuracy. The maximum number of errors in a single set was 2 (out of 60 elements), which occurred once. **(d)** Determined commands for validation set #2311 showing the output from the deep learning network. **(e)** True commands for validation set #2311 showing what was actually applied to the metasurface. **(f)** Example scattering responses for online validation showing the measured and predicted S_{21}responses. **(g)** Determined command probability for validation set #2311, showing the raw outputs from the deep learning network prior to rounding. This panel shows that the 2 elements determined incorrectly have command probabilities of 0.41 and 0.73, meaning the network was not highly confident in the result. **(h)** Errors, or incorrectly determined commands for validation set #2311, showing the 2 elements that were determined incorrectly. **(i)** Prediction errors or difference between the measured and predicted S_{21} responses.

closed loop configuration as shown in Fig. 4.4 (a). Commands were applied to the metasurface and the S_{21} response was measured and then passed through the trained deep learning model to verify accuracy. This provides a 3rd set of data that was not seen during training (or the initial validation). When the deep learning determined commands had errors, the determined commands were applied to the metasurface so that the difference in S_{21} responses could be computed. We define the difference, ΔS_{21}, between two measured S_{21} responses, S_{21}^a and S_{21}^b through the $\mathcal{L}_2$ (Euclidean) norm, $||S_{21}(f)||_2 = \sqrt{\sum_f |S_{21}(f)|^2}$. The summation is taken over the full measured frequency range (3-4 GHz) and ΔS_{21} is defined as

$$\Delta S_{21} = 2\frac{||S_{21}^a(f) - S_{21}^b(f)||_2}{||S_{21}^0(f)||_2 + ||S_{21}^1(f)||_2} \tag{4.1}$$

The normalization factor here is determined by the average of the $\mathcal{L}_2$ norms of the active commands (all 1s), S_{21}^1, and the inactive commands (all 0s), S_{21}^0. To understand how ΔS_{21} depends on the difference between commands, we first identified the minimum Hamming distance for each of the 10,000 sets in the training data. The Hamming distance is simply the number of elements that are different between 2 sets of commands. It is a useful metric for comparing command sets, but does not include scaling or correlation based on position; in some cases, the impact of toggling an element in the center may be significantly different than the impact of toggling an element on the edge of the metasurface.

The smallest Hamming distance between the training data sets ranged from a single element to 19 elements (out of 60). A series of whisker box plots showing

ΔS_{21} for the various Hamming distances is shown in Fig. 4.4 (b). The general trend shows an increase in ΔS_{21} with an increase in the Hamming distance. While the relationship is nonlinear, the dynamic range in ΔS_{21} for Hamming distances up to 1/3 of the total number of elements is large, approximately an order of magnitude.

Validation of the deep learning network in the configuration shown in Fig. 4.4 (a) was performed periodically after collecting the training data and the results are shown in Fig. 4.4 (c) at 2 hours, (d) at 36 hours (1.5 days), and (e) at 72 hours (3 days). Over time the scattering environment is expected to "age" and systematic changes to the scattering environment will occur. The blue diamonds show cases where there was a single error, and the black dots show cases where there were 2 errors. Each on-line validation experiment showed $\sim 95\%$ accuracy and the resulting ΔS_{21} for errors was small compared to the observed statistical extent of ΔS_{21} for single element Hamming distances. This suggests that even when the deep learning network is unable to determine the commands completely accurately, the resulting difference in S_{21} is very small. As shown in Appendix F.12, the accuracy was still $>85\%$ after 120 hours (5 days), but dropped to $\sim 65\%$ after 9 days.

The number of errors and number of cases with more than one error increases with time, showing the "aging" effect of the cavity, which can be quantified through the concept of scattering fidelity. Scattering fidelity is the normalized correlation as a function of time between two cavity responses with the same initial conditions [188]. Because a chaotic cavity is sensitive to small changes in the boundary conditions, such as volume perturbations, the scattering fidelity will decay over time [126–128]. Loss in scattering fidelity means that the accuracy of any trained deep learning

network has a finite lifetime, so we must periodically retrain the network on new training data to maintain accuracy. In our case, we have demonstrated that the deep learning network can determine metasurface commands with high accuracy ($>$ 95%) for at least 72 hours (3 days) after the initial training data collection, and with reasonable accuracy ($> 85\%$) up to 120 hours (5 days) after the initial training data collection. Large variations in environmental conditions, such as temperature or humidity, will introduce larger perturbations and more rapidly degrade the scattering fidelity.

An additional set of experiments was performed to determine the impact of cavity reverberation time on the performance of the deep learning network. To increase the losses in the cavity (and decrease the reverberation time), RF absorbent materials were placed inside the cavity. For each loss configuration, an ensemble of measurements was collected using the mechanical mode stirrer and the reverberation time was estimated from the power delay profile (PDP) [189]. See appendix C for details on estimating the reverberation time and Appendix D for a description of the ensemble statistics and the complex correlation coefficient. The mode stirrer was then set to a fixed position and another ensemble was collected for training data with 10,000 random metasurface configurations (following a biased random coin toss approach). The correlation coefficient was computed over all possible measurement pairs, for $\sim 5 \times 10^7$ combinations, to assess how highly correlated the training sets were. The deep learning network was then trained using the same network and parameters as previously discussed and the results are shown in Fig. 4.5. The cavity reverberation time ranged from 23 ns to 179 ns. The statistics of

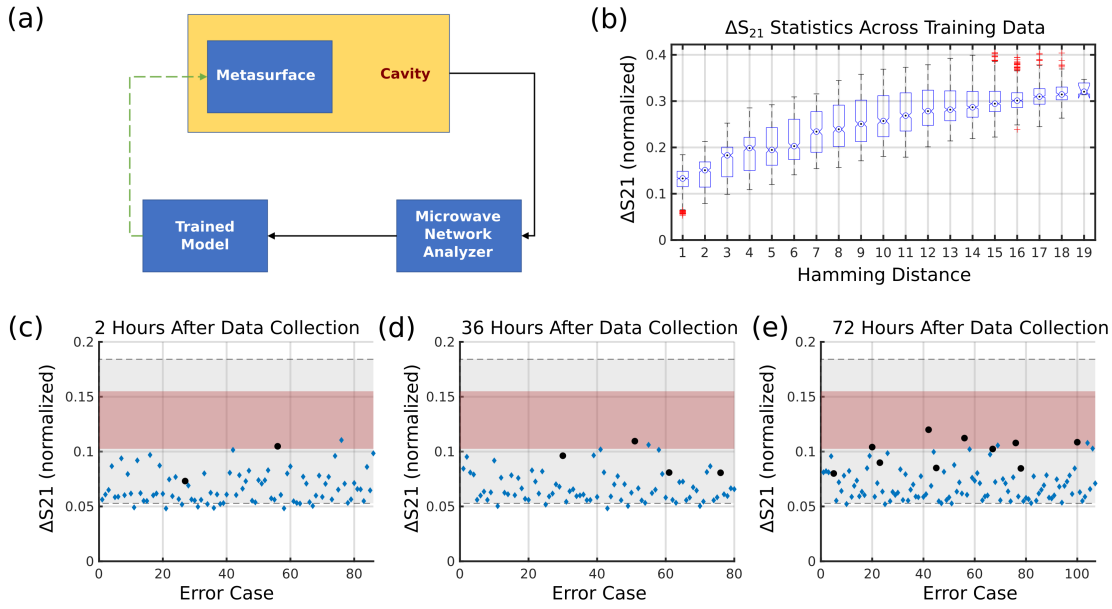

Figure 4.4: **On-line performance verification.** **(a)** Closed loop validation configuration. Commands were applied to the metasurface inside the cavity, the corresponding S_{21} response was measured on the PNA, and the results were passed through the trained deep learning network. Errors for the trained model were then measured to determine the difference in S_{21}, ΔS_{21}, between the two command sets. **(b)** ΔS_{21} statistics for the minimum Hamming distance across the 10,000 sets from the training data. Whisker plots are given for the smallest Hamming distance for each case, and show the mean value, 25^{th} and 75^{th} percentiles, and maximum and minimum values. **(c)** through **(e)** ΔS_{21} for online validation sets taken a specified time after the training data was collected. The shaded regions show the extent of the single element Hamming distance whisker box plot from panel (b). The grey region shows the full range from maximum to minimum, and the red region shows the 25^{th} and 75^{th} percentiles. The blue diamonds indicate cases with a single error, while the black circles indicate cases with 2 errors. These panels show that the ΔS_{21} for errors is very small, and in the lower region of the statistics covered by observed cases with single element Hamming distances. **(c)** Validation 2 hours after collecting training data, 2000 sets of commands were tested with 86 errors for an accuracy of 95.7%. **(d)** Validation 36 hours after collecting training data, 2000 sets of commands were tested with 80 errors for an accuracy of 96%. **(e)** Validation 72 hours after collecting training data, 2000 sets of commands were tested with 107 errors for an accuracy of 94.7%.

the correlation coefficients are shown relative to the left-hand axis, and show the median value, quartiles, and full extent. The achieved accuracy of the deep learning network on the training set is shown as the dashed red line relative to the right-hand axis and indicates that the accuracy and correlation coefficients are inversely related. The deep learning network is capable of operating in extremely complicated scattering environments, but the performance degrades as the cavity losses increase. This is because ray trajectories do not persist as long for high loss systems; the number of bounces for a given trajectory is reduced, which means there are fewer rays intercepted by the metasurface.

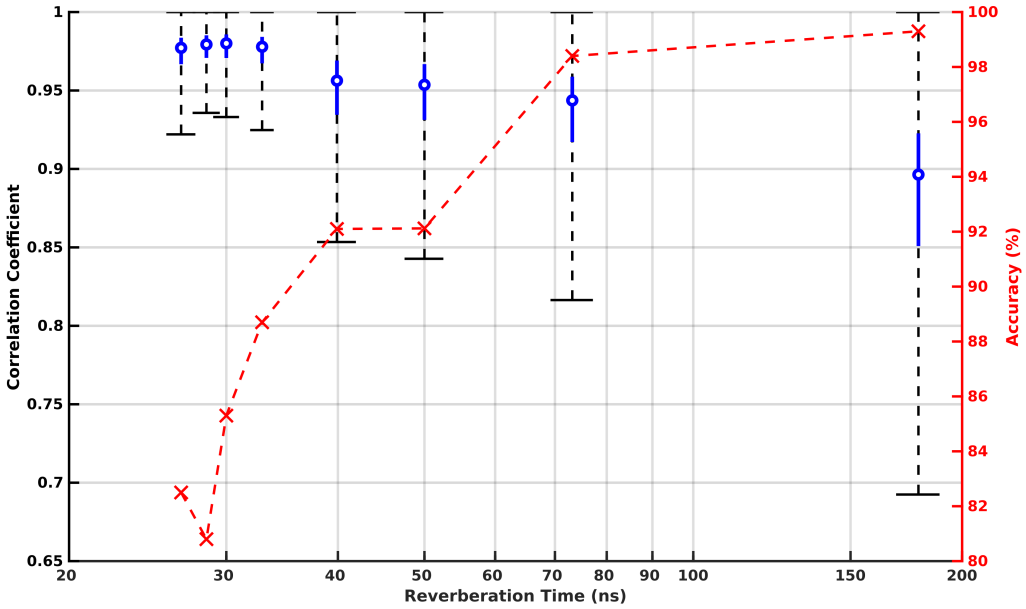

Figure 4.5: **Deep wavefront shaping performance vs. cavity reverberation time.** Performance of the deep learning network for different cavity loss configurations as specified by the cavity reverberation time (x-axis). The reverberation time is shown on a log scale to highlight the behavior for high loss configurations (short reverberation times). The statistics of the correlation coefficient over the ∼50 million combinations of measurement sets are shown relative to the left-hand y-axis. The blue line shows the extent between the quartiles, the blue circle indicates the median value, and the dashed black line shows the full extent. The achieved accuracy of the deep learning network is shown as the dashed red line relative to the right-hand y-axis, with individual points represented by a cross. The trend shows an inverse relationship between the correlation coefficient and the achieved accuracy of the deep learning network, indicating that the deep learning network struggles to identify features in the data when it is highly correlated.

Chapter 5: Summary and Discussion

In the first part of this dissertation, we demonstrated the ability of a programmable metasurface to generate microwave coldspots in a chaotic cavity at arbitrary frequencies and showed this capability exists even when applied over multiple frequency bands simultaneously.

The coldspots can be generated for different bandwidths and multiple input port configurations that induce additional angular and spatial diversity. We have also utilized the programmable metasurface to control the eigenvalues of the scattering matrix and direct them towards the origin to realize a CPA state for the cavity. Finally, we verified the existence of a CPA state and demonstrated the sensitivity to parameter sweeps in frequency, phase, amplitude, and metasurface configuration. All of this is accomplished with a metasurface that covers only 1.5% of the interior surface area of the cavity and a unique and effective stochastic algorithm to find desired outcomes despite the enormous space of possible metasurface commands.

A potential worst-case scenario could be experienced where many trials are attempted with no improvement in the metric. With sufficient diversity in the responses, this should only occur when the baseline metasurface commands (all 0s) already produce a deep null at the desired frequency. In the experimentation, we were always able to reduce the measured power by at least 4 dB; however, no

experimental cases were initialized with the bandwidth centered directly over a deep null. This could be addressed by shifting the center frequency or the bandwidth of the metric so that the metric is not initialized at a deep null. Increasing diversity unfortunately requires changing the cavity configuration so that the metasurface intercepts more rays. A larger metasurface, multiple metasurfaces, or even a smaller cavity may be necessary in this case.

In the second part of this dissertation, we demonstrated the use of a deep learning network for wavefront reconstruction to enable intelligent wavefront shaping in complex environments. Major novel aspects include complex-valued deep learning layers that exploit both phase and amplitude information, binning of the metasurface elements, a pseudo-2D data format that allows features to be extracted over both narrow and wide bandwidths, and a Terrapin module that enhances the receptive field, providing width and depth to the network.

One of the primary limitations of traditional deep learning is the amount of data required to train the networks. This is especially concerning in light of the fact that loss of scattering fidelity requires periodically collecting new training data. We have demonstrated the ability to train highly accurate networks with a limited amount of training data, requiring far fewer sets than the number of possible combinations of commands. We have also demonstrated that the accuracy can be maintained for a period of at least several days. This indicates that successful training on a reduced amount of data is possible, provided it is sufficiently diverse. Diversity in both the metasurface commands and the measured responses is then a key aspect in setting up any potential autonomous system.

For on-the-fly learning and adaptation to changing environmental conditions, we propose the future use of reinforcement learning [106, 190], which is at the intersection of artificial intelligence and optimal control. Reinforcement learning uses an agent that interacts with an environment to learn about it and then manipulate that environment in order to maximize (minimize) a reward (cost function), leading to the development of optimal control policies. In particular, the subset of reinforcement learning known as deep or double deep "Q" learning is gaining traction as a method for controlling quantum states [191–193]. Deep "Q" learning uses a deep learning network to estimate a quality matrix that scores the result of taking a particular action, while double deep "Q" learning uses two estimates to limit the implementation of poor control policies from overestimation [194]. The deep learning network architecture developed in this dissertation is well suited for estimation of this quality matrix.

Specification of an arbitrary scattering condition in the current implementation is cumbersome, as the complete S_{21} response over the full 3-4 GHz measurement window must be defined. For practical engineering applications, we prefer a simpler method of defining a desired wave scattering condition. Deep reinforcement learning also helps in this case, as it scores the performance of an agent through a scalar, positive, and real-valued metric. The agent uses the deep learning network to learn the relationship between metasurface commands and S_{21} responses, but the complicated details are hidden from the user. There are therefore 2 learning components to deep reinforcement learning: an inner deep learning network that learns how to map S_{21} responses onto metasurface commands, and an outer agent based loop that

learns how to use the inner deep learning network to optimize the desired metric. This metric can be the total power in a specified bandwidth for cold spot generation, the magnitude of the eigenvalues of the full S-matrix at a given frequency for coherent perfect absorption, or the bit error rate for communications systems.

Learning from scratch can be slow and may not be fast enough to adapt to changing environmental conditions. In this case, transfer learning, or using information about a similar problem to accelerate training for another one, can be incorporated into the reinforcement learning strategy [195].

Several concerns must be addressed to enable practical fielded hardware systems. First, the sensing component must be reduced in cost and size. The availability of SDR architectures presents an ideal path here, with many inexpensive platforms readily available. Compact devices such as the bladeRF [158] can replace the bulky network analyzer. SDRs have limited instantaneous bandwidth, typically 10-20 MHz, so modifications would be required to the pseudo-2D data representation. Second, processing large deep learning models on power hungry GPUs may exceed the allowable footprint in terms of both cost and power consumption. Deep learning models can be compressed by pruning and quantization [196], and the explosion of edge intelligence for connected devices in the Internet of Things is leading to more efficient embedded deep learning systems. An example is the Jetson series of embedded GPUs from NVIDIA; the currently available TX2 series can provide up to 1.26 trillion floating point operations per second on a 256-core GPU while consuming only 10-20 W of power [197].

In addition, cabling and interface requirements grow with the number of unit

cells provided by a metasurface. The ability to address individual unit cells and switch states as needed is critical to achieve a practical SRE. Connectors and large cable runs form bottlenecks and tend to be the weakest links in a system, so the capability of addressing unit cells wirelessly without further corrupting the environment is highly desired for large element counts.

In closing, we have shown that deep wavefront shaping provides an important step towards realizing intelligent reconfigurable metasurfaces for smart radio environments. Potential applications in the domain of electromagnetics include wireless power transfer, protection of sensitive electronic components, optimization of wireless networks, micromanipulation of objects, and nonlinear time reversal. Our technique is applicable to general wave chaotic scattering systems and is not strictly limited to electromagnetic waves. Adopting deep wavefront shaping to control the system scattering response with metasurfaces that interact with seismic waves [64, 65] or quantum waves [66] will unlock many innovative applications for wave chaotic systems.

Appendix A: Characteristic Parameters of Chaotic Cavities

This appendix lists the characteristic parameters of chaotic cavities.

Parameter	**Equation**	**Description**
Mean Mode Spacing	$\Delta f = \frac{\pi c^3}{2\omega^2 V}$	Specific to a 3D cavity [116]
Mean Mode Spacing	$\Delta k_n^2 = \frac{2\pi^2}{kV}$	Specific to a 3D cavity [116]
Loss Parameter	$\alpha = \frac{k^3 V}{2\pi^2 Q}$	Specific to a 3D cavity [116]
Mean Mode Spacing	$\Delta f = \frac{c^2}{\omega A}$	Specific to a 2D cavity [116]
Mean Mode Spacing	$\Delta k_n^2 = \frac{4\pi}{A}$	Specific to a 2D cavity [116]
Loss Parameter	$\alpha = \frac{k^2 A}{4\pi Q}$	Specific to a 2D cavity [116]
Heisenberg Time	$\tau_H = \frac{1}{2\pi\Delta f}$	Timescale after which dynamics are governed by quantum fluctuations [198, 199]
Ballistic Flight Time	$\tau_f = \frac{L_c}{c}$	Time required for a ray to traverse the cavity once
Power Delay Profile	$\mathrm{PDP} = \left\langle \lvert \mathrm{IFT}\{S_{21}\} \rvert^2 \right\rangle$	Decay in transmitted energy through the cavity [200]
Reverberation Time	$\tau_c = 4.34/\nu$	Time constant of the energy decay in the cavity, ν is the slope of the PDP in dB/s [119]
Cavity Quality Factor	$Q = \omega\tau_c$	See Appendix C
Number of Modes	$N = \frac{Vk^3}{3\pi^2}$	Weyl approximation [201]
Radiation Impedance	$\mathbf{Z}^{\mathrm{rad}} = \left\langle \frac{[GW]*\mathbf{Z}^{\mathrm{cav}}}{[GW]*1} \right\rangle$	See Appendix B
Average Impedance	$\mathbf{Z}^{\mathrm{avg}} = \langle \mathbf{Z}^{\mathrm{cav}} \rangle$	See Appendix B

Table A.1: Cavity Characteristic Parameters.

Appendix B: Time Gating in the Frequency Domain

This appendix presents a method to determine the radiation S-parameters, S^{rad}, in the frequency domain. In many cases, it is not feasible to directly measure S^{rad} but it can extracted from time gating the measured cavity S-parameters, S^{cav} [119]. To ensure that the important features of S^{rad} are maintained, the optimal gate width is determined by examining the S-parameters in the time domain and selecting a point in time before significant contributions from the ensemble average are present. This prevents short orbits from having a large impact on the estimated S^{rad}. Figure B.1 shows an example ensemble of S_{11} measurements. The largest short orbit contribution occurs at ~ 8 ns, so we will select a gating time of 7 ns.

A simple time-gating approach is to convert the signal into the time domain with an Inverse Fast Fourier Transform (IFFT), multiply the signal with a rectangular gate, and then convert the signal back to the frequency domain with a Fast Fourier Transform (FFT). However, this approach compounds truncation effects through the IFFT/FFT sequence and induces band edge roll off effects due to the fact that we are using a finite, single-sided spectrum [202]. In addition, the lack of low frequency (DC) information induces a roll off at long times when using a simple IFFT. To account for this, the signal must be conditioned to increase the length so the low frequency roll off is outside the region of interest. It must also be resampled

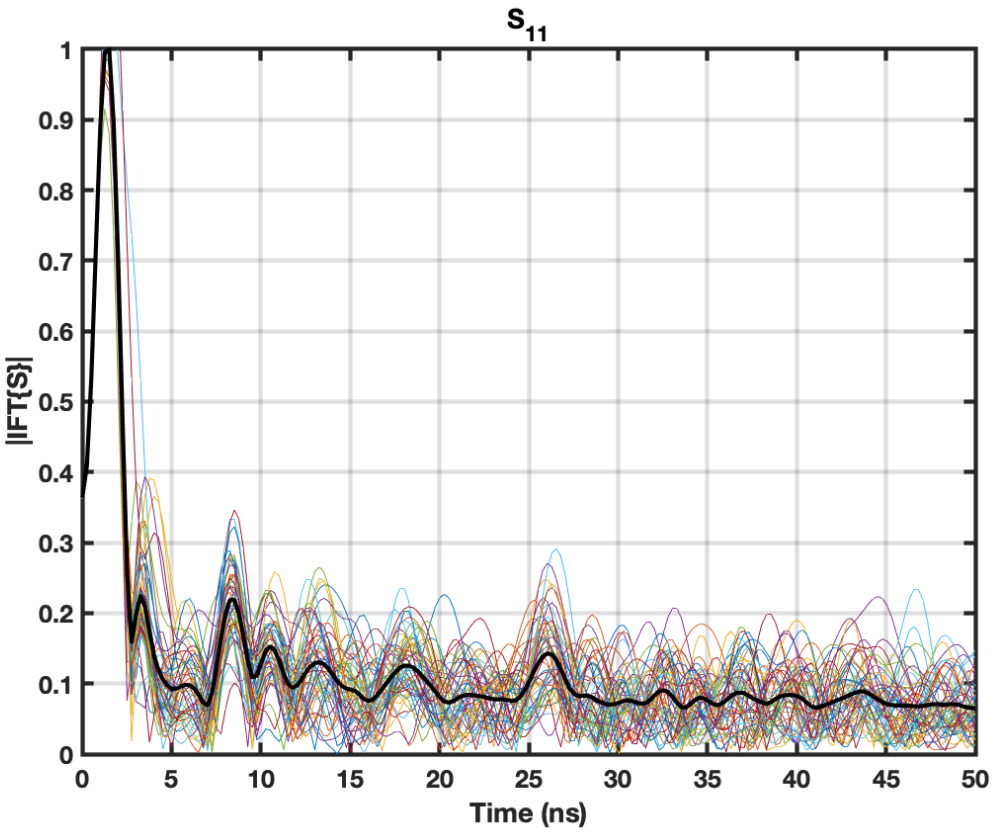

Figure B.1: **Example S_{11} Time Domain Response.** The solid black line is the ensemble average, S^{avg}, and the colored lines are the individual realizations. The dominant short orbit is present at $\sim$8 ns, as seen by the largest bump after the initial prompt reflection.

and zero padded to obtain the desired resolution and then windowed to remove the edge effects. The implementation then requires significant book-keeping to ensure the desired frequency domain information is preserved.

An alternative approach is to perform gating in the frequency domain through a convolution operation and use the concept of renormalization to remove band edge artifacts. In order to optimize frequency domain gating, the gate needs to be centered at the time of the response being gated [202]. Because we are interested in gating the initial response, we will center the time window at 0 seconds; accordingly, the overall width of the gate will be double the desired end time to include both positive and negative time extents (due to the periodicity of the discrete Fourier transform).

The gate can be designed in 2 segments: a rectangular gate in time transformed to the frequency domain, and a window to reduce side lobes and ringing artifacts due to the sharp transitions of the rectangular gate. The generalized gate function in the frequency domain, $G(f)$, for a rectangular time domain gate defined between times t_1 and t_2 is given by a sinc function, where we define $\mathrm{sinc}(x) = \sin(\pi x)/(\pi x)$.

$$G(f) = (t_2 - t_1)\mathrm{sinc}\left[f(\,t_2 - t_1\,)\right]\exp\left[-j2\pi f(t_2 + t_1)\right] \tag{B.1}$$

For a gate centered at $t = 0$ with end time t_g ($t_1 = -t_g, t_2 = t_g$), this expression is simplified.

$$G(f) = 2t_g\mathrm{sinc}\left[2ft_g\right] \tag{B.2}$$

The next step is to design a window function in the frequency domain to suppress side lobes. A common window is the Kaiser-Bessel window, which is defined by a shape parameter, β, and the window length, M [203]. For the analysis described here, the Agilent PNA provided 32,001 points of data and the window was defined with $\beta = 6.5$ and $M = 23,897$. The filter response in both the time and frequency domains is shown in Figure B.2. Because we are gating the signal at 7 ns, the -3dB width of the gated window is 14 ns.

Finally, we can apply the gate by convolving the product of $G(f)$ and $W(f)$ with the measured S-parameter for a given cavity realization. Renormalization is handled by dividing the gated S-parameter by the convolution of the product of $G(f)$ and $W(f)$ with a constant unit frequency response, which removes band edge roll off effects [202]. S^{rad} is then found by taking the average over the ensemble of cavity configurations as shown in Equation B.3.

$$S^{\text{rad}}(f) = \left\langle \frac{[G(f)W(f)] * S(f)}{[G(f)W(f)] * 1(f)} \right\rangle \tag{B.3}$$

Figure B.3 shows the overall ensemble with the average and radiation S-parameters superimposed. Note that in general, $S^{\text{avg}} \neq S^{\text{rad}}$. The persistent short orbits longer than the time gate are removed from S^{rad} but remain in S^{avg}. Short orbits are due to the specific configuration of the cavity and are not representative of the free-space responses, so time gating provides a better estimate of the radiation

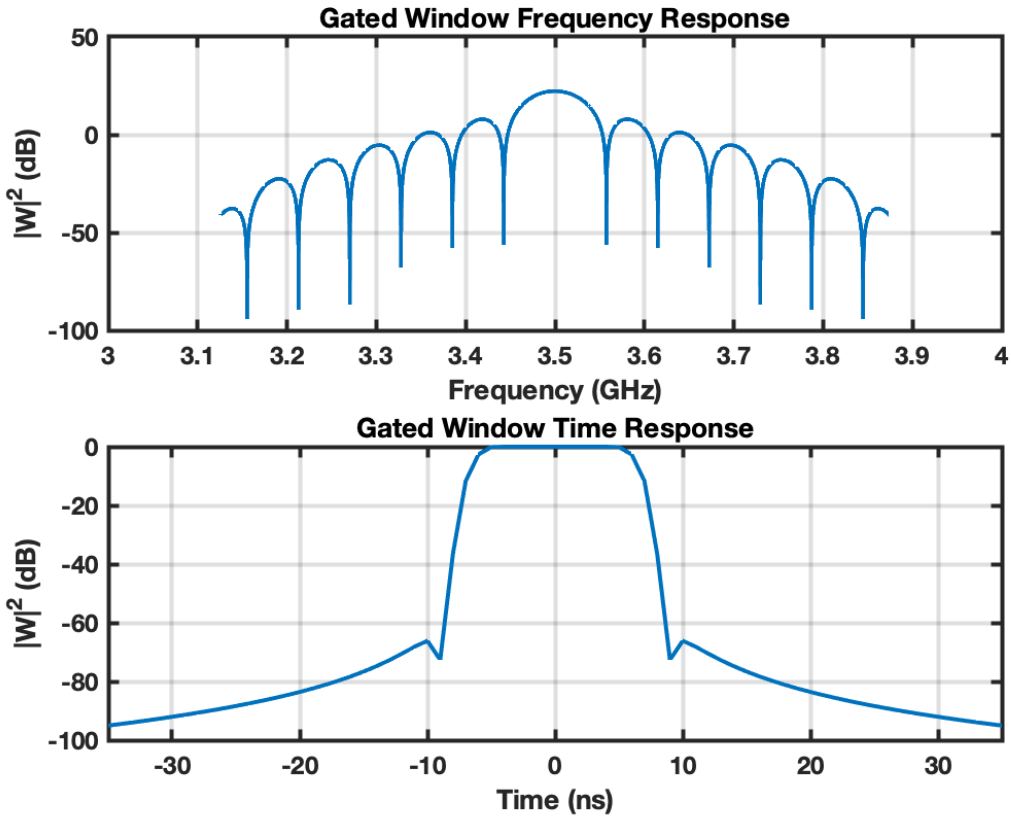

Figure B.2: **Time Gating Filter Response.** The top shows the gate function W in the frequency domain and the bottom shows the gate function in the time domain.

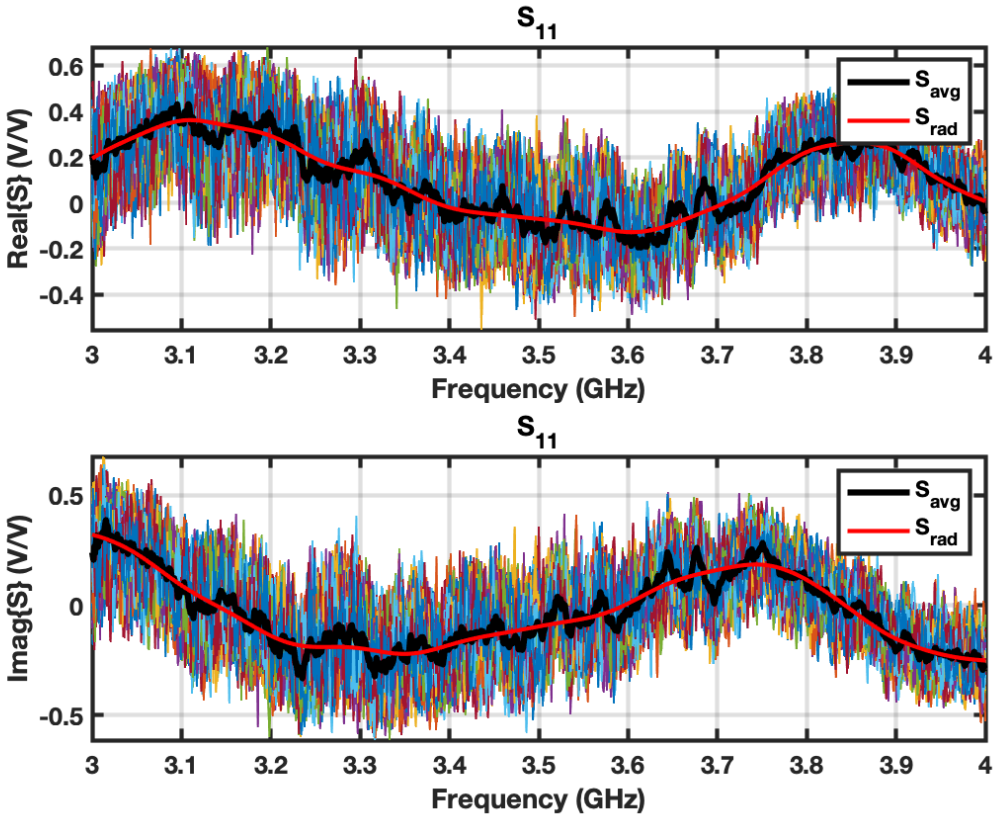

Figure B.3: **Example S_{11} Ensemble.** The solid black line is the ensemble average, S^{avg}, the solid red line is the radiation S-parameter, S^{rad}, and the colored lines are the individual realizations, S^{cav}.

S-parameters.

To demonstrate how to actually apply this in practice, several Matlab code listings follow. Listing B.1 provides code to generate the appropriate Kaiser-Bessel window with the parameters stated above. It requires defining the gate width (7 ns for the provided examples), the number of points present in the signal, and the overall frequency span of the signal.

Listing B.1: Computing the Kaiser-Bessel Window

```
%need to have the gate width (gateWidth) defined, along with ...
    signal size (N) and span in frequency (fSpan)
%get the time spacing
dt = (N-1)/(N*fSpan);
%first setup the parameters for the Kaiser-Bessel window
alpha = 68;
%kaiser beta parameter
if alpha > 50
    beta = 0.1102*(alpha - 8.7);
elseif alpha > 21
    beta = 0.5842*(alpha - 21)^0.4 + 0.07886*(alpha-21);
else
    beta = 0;
end
%get the window length
nWindowPoints = ceil((alpha - 7.95)*N/80.4160);
%make sure window length is odd
if mod(nWindowPoints,2) == 0
    nWindowPoints = nWindowPoints + 1;
end
%get the window
wf = window(@kaiser,nWindowPoints,beta);
```

Listing B.2 builds the time gating function, W, in the frequency domain. This assumes the window function from Listing B.1 was generated.

Listing B.2: Computing the Time Gating Function

```
%determine the number of samples to be set to one in the gate
nGateSamples = floor(gateWidth/dt);
%make sure it is an odd number
if (mod(nGateSamples,2) == 0)
    nGateSamples = nGateSamples - 1;
end
%initialize R in the time domain
Rt = zeros(nWindowPoints,1);
%construct R as a box with energy at the edges
%nGateSamples is odd, put (N-1)/2 at the front and (N-1)/2 + 1 ...
    at the back
Rt(1:(nGateSamples-1)/2) = 1;
Rt(end-(nGateSamples-1)/2:end) = 1;
%take the Fourier transfer and use fftshift to center the gate ...
    in the
%frequency domain
Rf = fftshift(fft(Rt));
%Muliply the gate and the window to get the windowed gate
W = Rf.*wf;
```

Listing B.3 then applies the time gating with renormalization. It assumes the input signal vector, Sf, contains the entire ensemble and is sized as number of frequencies $\times$ number of realizations.

Listing B.3: Frequency Domain Gating with Renormalization

```
%% Apply the gate in the frequency domain
%initialize the output signal
Sfc = zeros(size(Sf));
%convolve the original signal with the windowed gate
for count = 1:M
    Sfc(:,count) = conv(Sf(:,count),W,'same');
end
%for renormalization, build a unit vector across the frequencies
unitResponse = ones(size(Freq));
%convolve the unit vector with the windowed gate
unitResponsec = conv(unitResponse,W,'same');
%divide the signal by the gated unit response
Sfg = Sfc./unitResponsec;
```

Appendix C: Estimating Cavity Time Constants

This appendix presents a method to estimate the cavity time constant or reverberation time, τ_c, the basic aspects of which are provided in the literature [119, 130]. The time constant is critical to estimating the cavity Q and loss parameter, α, so it is important to have a good estimate of τ_c to start. The time constant is an inherent parameter of the cavity that is solely dependent on the losses [130]. It is independent of the geometric details and does not depend on the position or specific configuration of the antennas used for measurement; however, lossy or nonlinear elements in the antenna will impact the estimate.

To estimate the time constant, we use the Power Delay Profile (PDP), which is defined as [189, 200]

$$\mathrm{PDP} = \left\langle \left| \mathrm{IFT}\left\{S_{21}\right\} \right|^2 \right\rangle \tag{C.1}$$

Because the PDP is a function of the cavity, the cavity S-parameters are used in Eq. C.1 rather than the radiation S-parameters. Note that for a 2-port system, the PDP is defined by the transmission coefficients, S_{21} or S_{12}. The reflection coefficients, S_{11} or S_{22} can also be used provided the ports are well coupled (the reflection coefficients are small); if the ports are not well coupled, losses due to reflection dominate and

S_{11} (or S_{22}) does not accurately represent losses in the cavity. In this regard, the difference between estimating the time constant with S_{11} (or S_{22}) vs. S_{12} (or S_{21}) provides a measure of how well coupled the ports are.

The PDP will have an exponential decay as the energy is lost from the cavity, so we can estimate the time constant by either fitting an exponential curve in linear space or a straight line in log space. In log space, $\tau_c = 4.34/\nu$, where ν is the slope of the PDP in dB/s [119]. The fitting concept is straightforward; however, there is an art to selecting the region where the fit is applied.

The PDP will deviate from a decaying exponential at both early and late times. For the early time behavior, the cavity is still charging and is described by a double exponential model [189]. The measurements usually have a single sided spectrum without a DC component, so a direct IFFT induces a roll off in the response at long times (relative to the sampling time). Late time behavior is further impacted by the noise floor, which introduces additional uncertainty in the estimate. To account for these issues, the fitting should start late enough to avoid early time deviations and stop early enough to avoid late time deviations. The start and stop times can be determined by inspection, and are dependent on both the cavity under test and the measurement settings.

Figure C.1 shows the fit to the PDP with a starting time of 750 ns and a stopping time of 2.75 μs. The blue curve is the raw measured PDP, the red curve is the PDP after smoothing with a moving average filter, the black line is the fit over the specified region, and the yellow line is the fit extended to cover the full time period. The overall time constant here was found to be 180 ns.

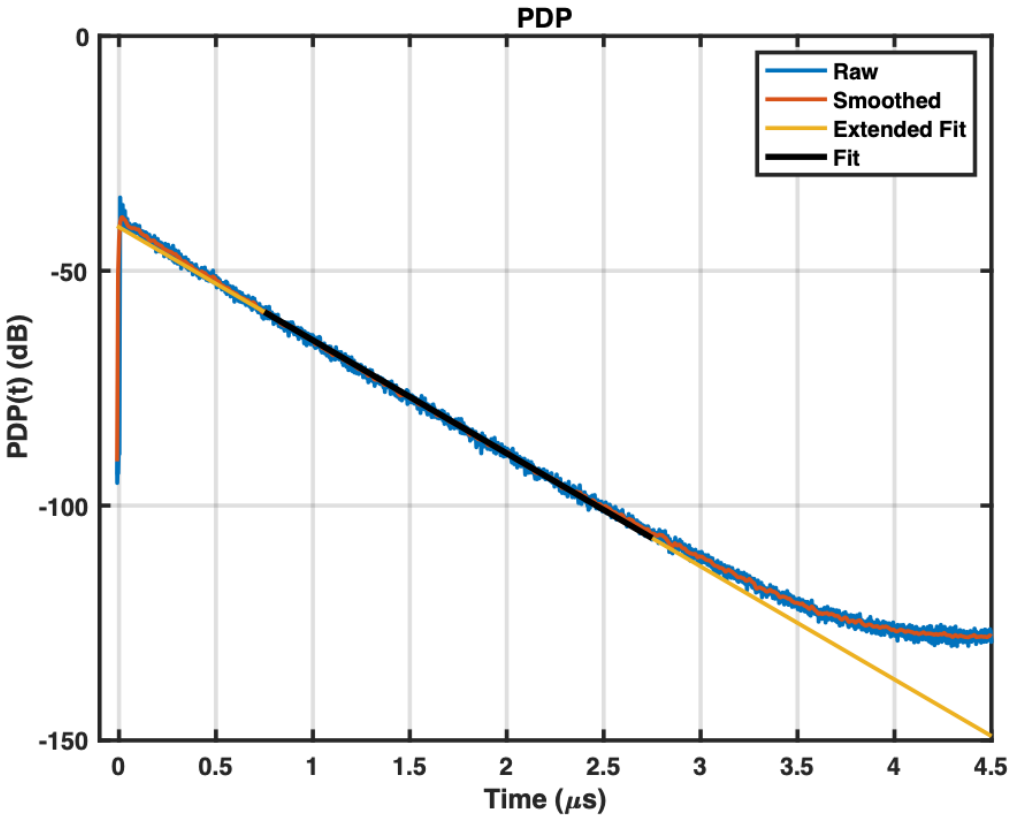

Figure C.1: **Estimating Cavity τ_c.** PDP of the measured realization along with the curve fits to estimate τ_c.

Appendix D: Ensemble Statistics

This appendix presents the Pearson correlation coefficient for complex data and discusses ensemble statistics from chaotic cavities. To compare the similarity between measured sets of S-parameters, we can use the Pearson correlation coefficient, ρ_c. For complex sequences, where the sequence is over N frequency points, ρ_c for realizations S^a and S^b is given as

$$\rho_c(S^a, S^b) = \frac{1}{N-1} \sum_{i=1}^{N} \left[\frac{S_i^a - \langle S^a \rangle}{\sigma_a} \right]^* \left[\frac{S_i^b - \langle S^b \rangle}{\sigma_b} \right] \tag{D.1}$$

Here, σ_a represents the standard deviation of sequence (realization) a. An example of the correlation coefficient from an ensemble of S_{21} measurements at 50 different mode stirrer positions is given in Fig. D.1. In this figure, each realization indicates that the mode stirrer was rotated by an increment of $360^\circ/50 = 7.2^\circ$. For realization r, the correlation coefficient is computed between position 1 and position r of the mode stirrer. The largest correlation coefficients are therefore when the mode stirrer is close to its original position, i.e., at realization 2 and 50.

For an ensemble containing M realizations, the number of distinct combinations of 2 realizations can be determined from the binomial coefficient, which tells us how many independent correlation coefficients can be determined from the ensemble.

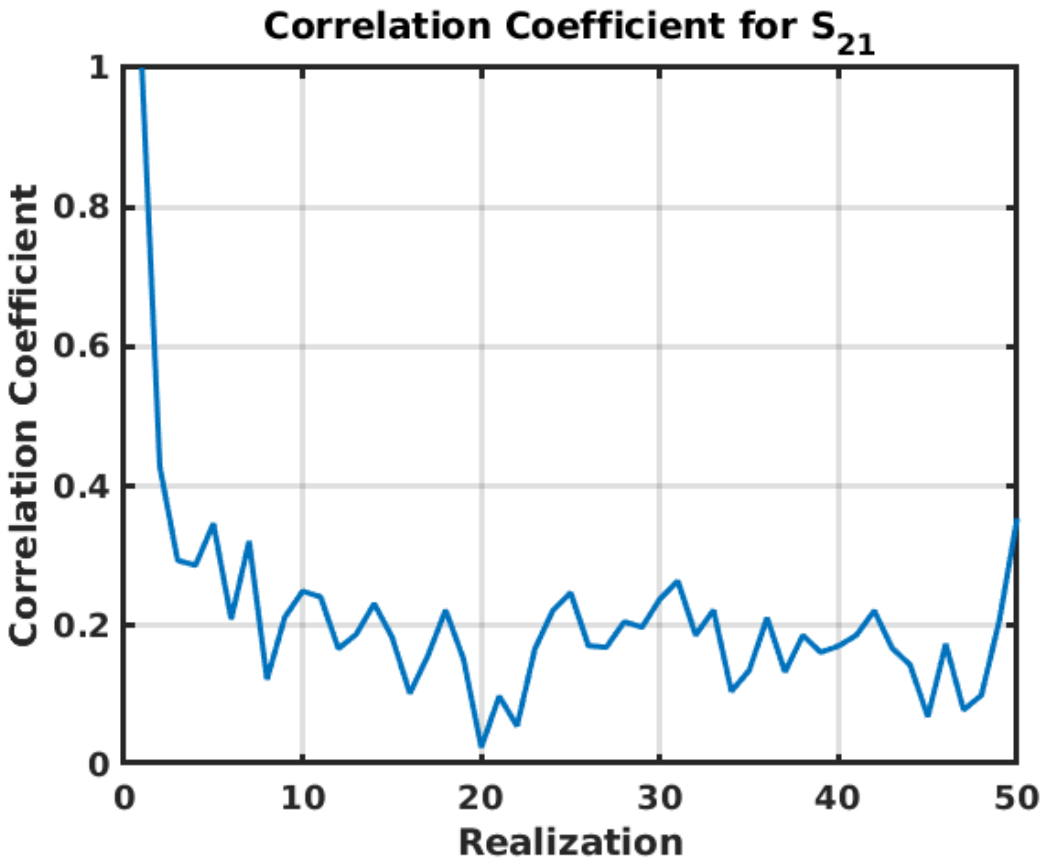

Figure D.1: **Correlation coefficient for S_{21} over an ensemble of 50 realizations.** Realizations were generated through a mechanical mode stirrer and the correlation coefficient of all the positions compared to the 1st position is shown here.

$$\binom{M}{2} = \frac{M!}{2\,(M-2)!} \tag{D.2}$$

An ensemble with $M = 10{,}000$ has 49.995×10^6 unique correlation coefficients, while an ensemble with $M = 50$ only has 1225 unique correlation coefficients. For computational efficiency, the correlation coefficients are often computed as a matrix. The diagonal of this matrix represents the correlation of each realization with itself and the correlation coefficients are symmetric about the diagonal. Therefore, the unique correlation coefficients are contained either above or below the diagonal. An example showing the matrix of correlation coefficients is given in Fig. D.2. Note that the largest correlations again occur between realizations separated by small angular differences of the mode stirrer, and are located near the diagonal.

The statistics of the correlation coefficients are useful for evaluating the ensemble. High correlation coefficients can indicate the presence of persistent short orbits or poor coupling between the port and the cavity. In addition, large reflection coefficients, $|S_{11}|$, are a hallmark of poor coupling and indicate that most of the energy is reflected back to the port rather than coupled into the cavity.

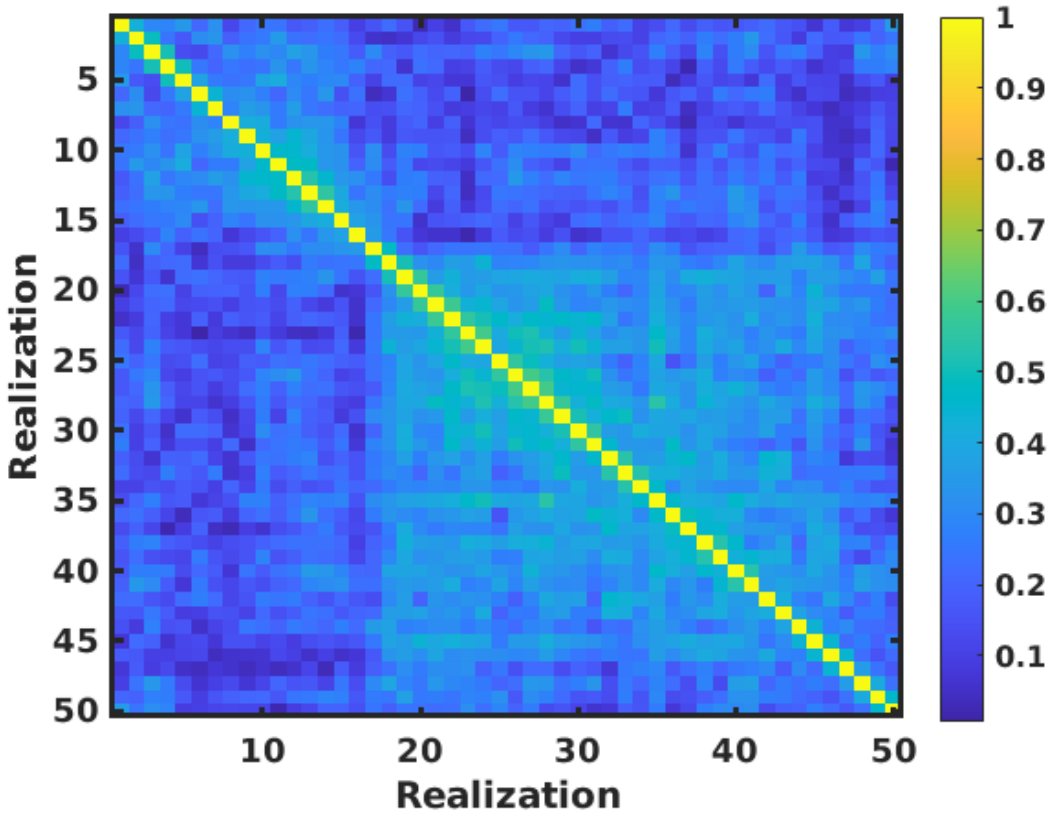

Figure D.2: **Matrix of correlation coefficients for S_{21} over an ensemble of 50 realizations.** Realizations were generated through a mechanical mode stirrer, the unique correlation coefficients are contained either above or below the main diagonal.

Appendix E: Stochastic Optimization Approach Supplemental Material

This appendix provides supplemental material for the stochastic optimization approach discussed in Chapter 3.

E.1 Experimental Setup

The metasurface used for the experimentation is a reflectarray fabricated by the Johns Hopkins University Applied Physics Laboratory (JHU/APL) and is shown in Figure E.1. The individual LC resonator unit cells can be seen on the front side and the GaAs FET amplifiers can be seen on the back side. It is designed to operate in the S-band from 3-3.75 GHz and provides 240 binary unit cells arranged in a rectangular grid of 10 rows and 24 columns. The unit cells have characteristic length $< \lambda/6$ [30].

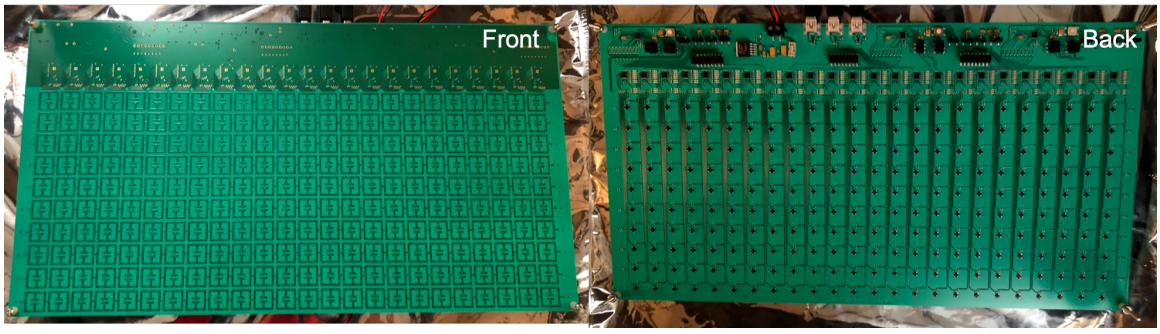

Figure E.1: **Metasurface device showing both the front and back of the board.**

A test cavity was constructed using aluminum angle brackets for the frame and 0.019 inch thick aluminum sheets for the sides. Each side length is 3ft (0.9144 m), so the total cavity volume is 0.76 m^3 and the total surface area is 5.02 m^2. This geometry ensures the cavity is overmoded with at least 9 wavelengths per side, but unfortunately means that the active area of the metasurface only covers a small portion ($\sim 1.5\%$) of the total surface area. All interior joints were lined with copper tape to minimize losses and hemispherical scatterers were installed in the corners of the cavity to force an irregular shape, after which the effective volume was reduced to 0.74 m^3. A higher loss version of the cavity was realized by distributing absorbing material in the bottom of the cavity. The metasurface was installed on a wall of the cavity as shown in Figure E.2, with a 1/4 inch gap between the metasurface and the wall. This figure also shows the power and 3 USB cables running out through the top of the cavity.

An overview of the experimental setup is shown in Figure E.3 and shows the cavity configuration, the locations of the 3 ports relative to the metasurface, and the overall connectivity. The cavity has 3 ports with port 2 used for scoring and ports 1 and 3 used for signal injection. The input ports can be driven either individually or collectively with a relative phase shift. When they are driven collectively, the underlying scattering matrix is 3 x 3; this extra dimensionality is hidden when using a 2-port network analyzer. To ensure that wave interaction with the metasurface dominates the results, a sheet of foil is used to block the direct line of sight path from the input ports to port 2. Ultra wide band (UWB) antennas designed for operation from 3-10 GHz are connected to each port. The antennas connected to

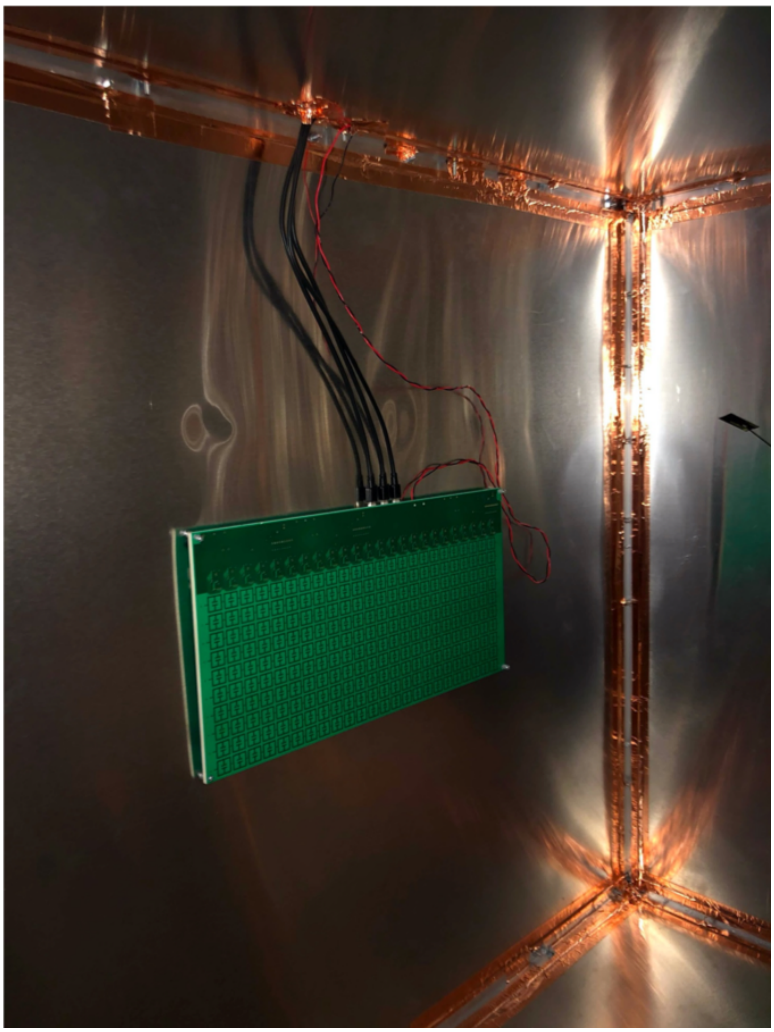

Figure E.2: **Metasurface installed inside the cavity.**

ports 1 and 3 are Taoglas FXUWB10 antennas and are mounted so they radiate outward into the cavity from the walls in a vertically polarized configuration. The antenna connected to port 3 is a PCB module mounted orthogonally so that it is horizontally polarized.

Finally, the metasurface is controlled through 3 USB interfaces using custom C++ software with a Python wrapper on a MacBook Pro laptop, which also controls the Agilent network analyzer through an ethernet connection. In order to prevent corruption from noise, multiple measurements are averaged.

E.2 Cavity Losses

The cavity time constant, τ_c, is an intrinsic aspect of the cavity that is dependent on losses but independent of the specifics of the underlying scattering system. This means τ_c is not dependent on the positioning or characteristics of the ports or antennas used to obtain it [130]. τ_c is an important characteristic of the cavity as it is related to the quality factor, Q, through $Q = \omega\tau_c$. One way to estimate τ_c is through Power Delay Profile (PDP), which is the ensemble average of the magnitude squared of the Inverse Fourier Transform (IFT) of the transmission coefficient, $\text{PDP} = \left\langle |\text{IFT}\left\{S_{21}\right\}|^2 \right\rangle$ [189,200]. Since the power in the cavity decays exponentially, we can perform a linear fit on the logarithmic magnitude and estimate τ_c as $4.34/\nu$ where ν is the slope of the PDP in dB/s [119]. See Appendix C for further details.

Figure E.4 shows the time constant estimated for the cavity under various configurations. There were 3 antenna pairs used in the PDP measurement: dipoles with both horizontal and vertical polarization, loops, and UWB. Measurements

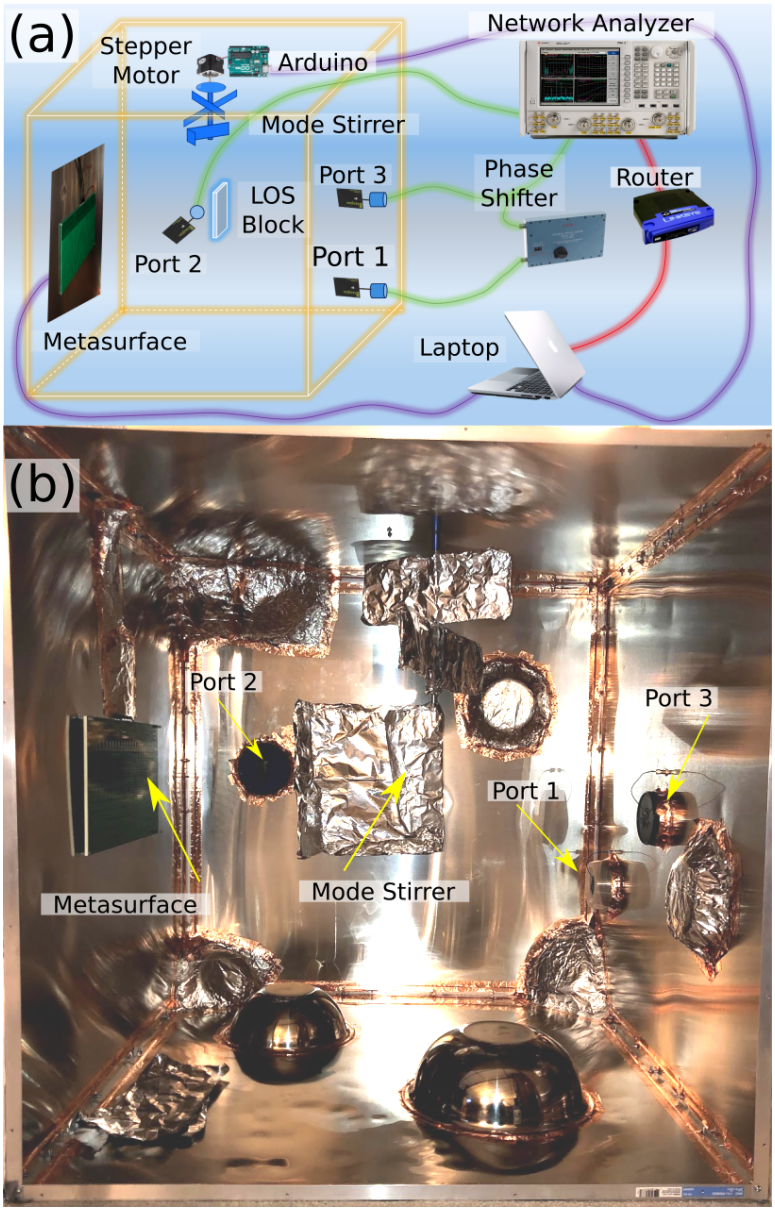

Figure E.3: **Experimental setup.**

were taken with the cavity empty before installing the metasurface and in low and high loss cases with the metasurface installed. Each data point in Figure E.4 came from a 100 MHz window centered at the corresponding frequency. Installing the metasurface in the cavity had a significant impact on the time constant, reducing it by a factor of 2. Adding absorbing material for the high loss configuration reduced the time constant by another factor of 2. Powering on and off the metasurface however, had little impact on the time constant in either configuration. The PDP was measured between ports 1 and 2 with 3 different antenna pairs: dipole (in both horizontally and vertically polarized orientations), loop and UWB. The On and Off curves indicate whether the metasurface was powered on or off.

E.3 Diverse Cavity Realizations

Attempts were made to decompose the input commands into a deterministic set of orthogonal basis functions. This included driving single elements, columns of elements and even Hadamard matrices. Unfortunately, these all produced a narrow range of responses that did not cover the full extent of possibilities. A Hadamard matrix provides an orthonormal basis and decomposes sequences similarly to spatial frequencies; it is less computationally intensive than 2-D Fourier transforms and has many applications in multi-input multi-output communication theory and synthetic aperture imaging [204, 205].

Figure E.5 presents the resulting ensembles from driving the metasurface with Hadamard basis functions and shows that the responses are narrow and do not cover the full extent. While the spatial frequency content and grouping of elements

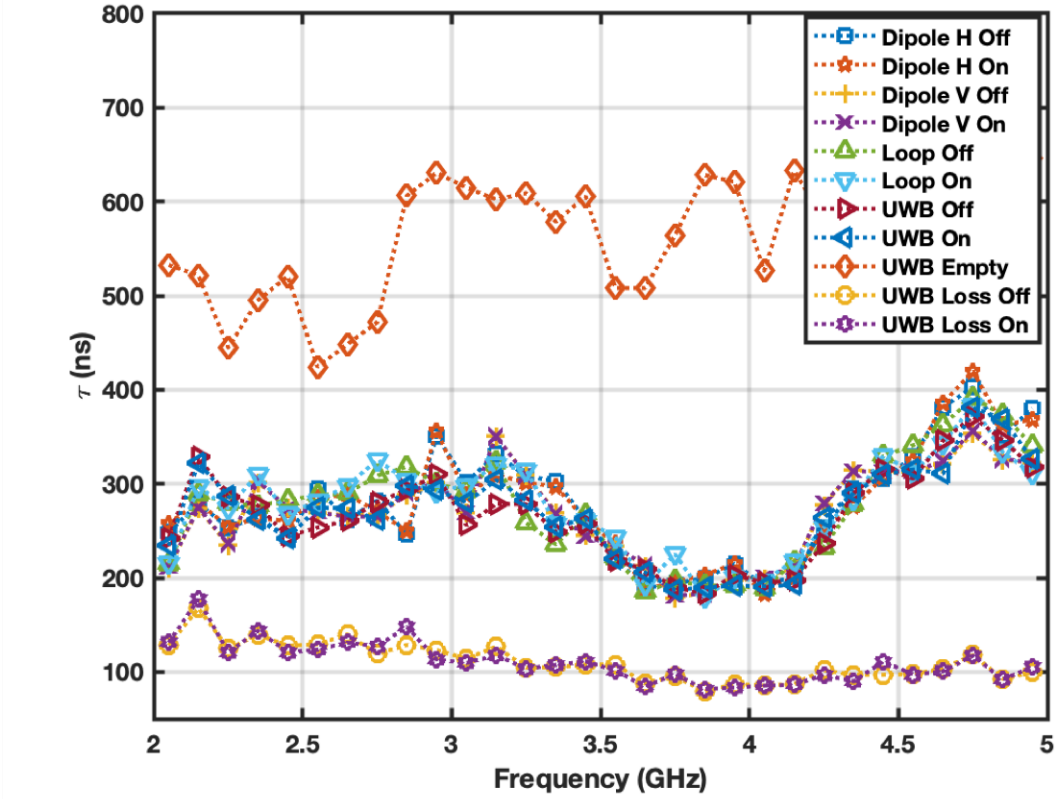

Figure E.4: **Estimated cavity time constant for various configurations.** The three primary groupings are the empty cavity without the metasurface (top), the low loss configuration with the metasurface (middle), and the high loss configuration with the metasurface (bottom).

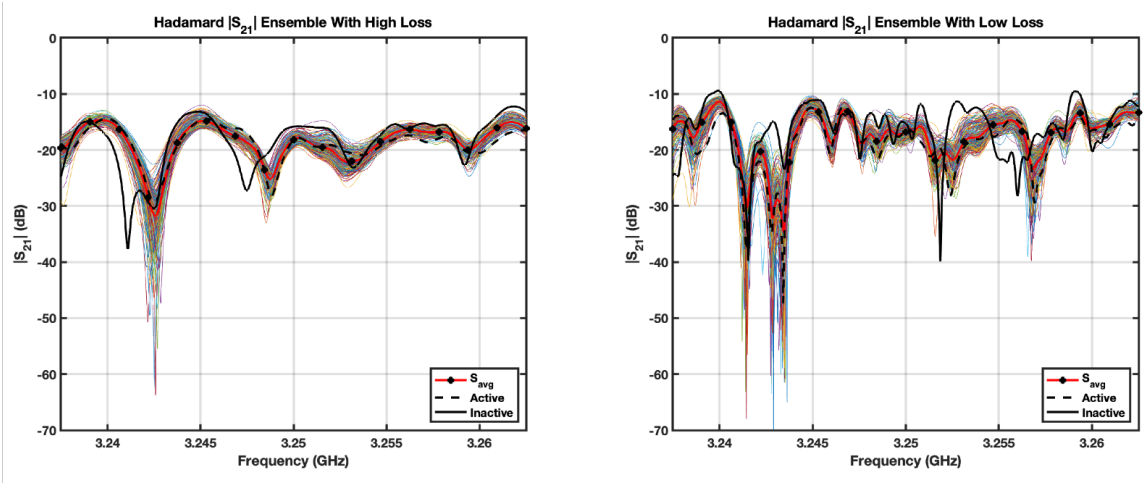

Figure E.5: **Hadamard sequence $|S_{21}|$ ensembles for both high loss and low loss cases.** The high loss case is shown on the left hand side and the low loss case is shown on the right hand side.

in the command sets changed, the number of active elements did not. An unbiased random coin toss approach likewise yielded a narrow distribution as roughly 1/2 the elements were active in each command set. The ensembles do not span the range covered by the inactive (all 0s) and active (all 1s) cases and do not cover the full extent. More variation is present in the low loss case because the ray trajectories survive longer and have more opportunities to interact with one another

The overall best approach to generating diverse ensembles was to use doubly random methods, in which compound probability distributions are used. This implies the statistics follow a Cox process, which is a generalization of a Poisson process with the underlying intensity or local mean itself a random process [206]. A random biased coin toss, where the bias itself was selected from a random draw for each command set worked well but only excited high spatial frequencies on the metasurface. To include lower spatial frequencies, we added a doubly random inverse power spectrum approach, where the power spectrum is just a power law on

spatial frequencies with the exponent a random draw.

$$C = \mathrm{Re}\left(\mathrm{IFFT}\left\{\left[\mathcal{N}(0,1) + j\mathcal{N}(0,1)\right]\sqrt{\mathbf{f_r}^{\gamma}}\right\}\right) \tag{E.1}$$

A small exponent will excite lower spatial frequencies, while a larger exponent will excite higher spatial frequencies. the number of active elements was allowed to change with each trial. We generally used a combination, with half the ensemble generated through an inverse power spectrum and the other half generated through a random biased coin toss. The ensemble for a set of 4000 realizations is shown in Figure E.6. The doubly random methods allow the number of active elements to change and provide a wider range of responses than the deterministic methods. This can be seen as the distribution from the biased coin toss covers the entire area between the inactive (all 0s) and active (all 1s).

From Figure E.6, we can see that the bandwidth of the narrowest null is $\sim$ 500 kHz and the closest spacing between nulls is $\sim$ 1 MHz. This matches the observed trends over the full 1 GHz measurement window, with typical bandwidths of 0.5-1 MHz and spacings of 1-2 MHz. The optimization metric was chosen to be the average power measured at port 2, P_2, when driving an input from some combination of ports 1 and 3. To isolate narrow band features, our initial metric bandwidth was chosen to be 500 kHz.

E.4 Coherent Perfect Absorption State Generation and Verification

A Coherent Perfect Absorption (CPA) state is not guaranteed at any specific frequency, so the approach needs to identify candidates. We repeated the iterative

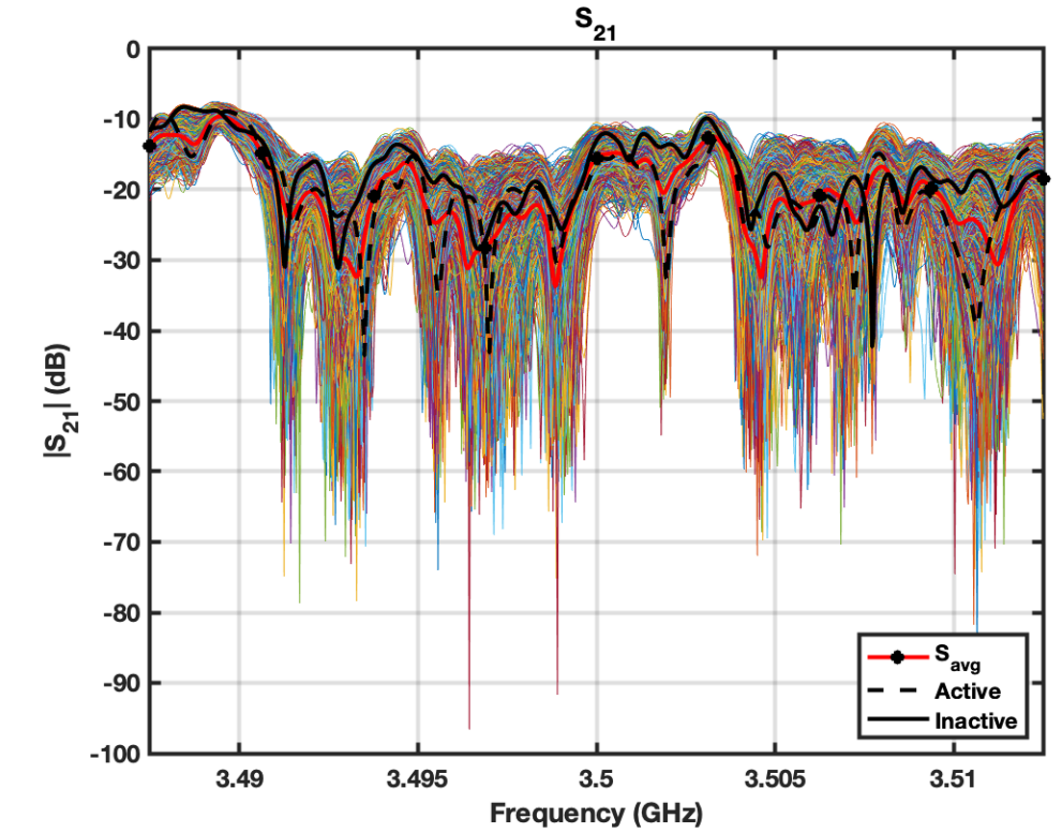

Figure E.6: **Ensemble of 4000 $|S_{21}|$ realizations from a combination of doubly random power spectrum and biased coin toss approaches.** The mean value is shown as the solid red line, the case with all elements active is shown as the solid black line and the case with all elements inactive is shown as the dashed black line.

optimization algorithm from the coldspot generation but initialized it by applying a random command set to the metasurface and then finding the eigenvalue with magnitude closest to a pre-selected value.

Verifying the CPA state requires exciting the scattering system with the corresponding eigenvalue. We used an Agilent N5242A PNA-X Network Analyzer configured for 2-independent source operation. This configuration requires selecting the appropriate signal path on the setup menu and making the jumper connections on the back of the instrument as per the network analyzer documentation. An external phase shifter was connected between port 1 and the cavity. The CPA state condition was tuned by adjusting the relative amplitude between the elements with the power control on the network analyzer and the relative phase between the elements with the external phase shifter. We can perform parameter sweeps to determine the sensitivity of the CPA to frequency, relative phase, relative amplitude, and metasurface commands.

Because there are 2 independent sources, S-parameter measurements are not available in this configuration and we need to make use of the network analyzer receivers. The network analyzer has reference and test port receivers to measure incoming and outgoing signals. R_1 measures the reference signal out of port 1 and R_2 measures the reference signal out of port 2. A measures the signal into port 1 and B measures the signal into port 2. The metric we are interested in is the power ratio of the total outgoing power from the cavity to the total incoming power from the network analyzer, $P_{\text{out}}/P_{\text{in}} = (A+B)/(R_1+R_2)$.

Appendix F: Deep Learning Approach Supplemental Material

This appendix provides supplemental material for the deep learning approach discussed in Chapter 4.

F.1 Cavity and Experimental Configuration

The cavity used has a volume of $\sim$0.76 m^3 and includes 3 ports, with ports 1 and 3 used to inject signals and port 2 used for scoring. The cavity configuration and experimental schematic are shown in Fig. F.1. Each port is connected to an ultra wide band antenna (UWB) and the nominal measurement window is 3-4 GHz. A line-of-sight block is used to obstruct the direct transmission path from port 2 to both ports 1 and 3, and an Arduino Uno [207] controlled mechanical mode stirrer is included to allow collection of an ensemble of cavity realizations. The stepper motor is a Stepper Online 17HS13 motor [208], and provides 200 distinct steps. The experimental setup is controlled by a MacBook Pro laptop, with an Agilent N5242A network analyzer used to measure cavity S-parameters. Ports 1 and 3 can be driven either independently or collectively with a relative phase shift provided by a NARDA phase shifter. To reduce the cavity symmetry, irregular scattering objects were installed on the walls.

In the frequency range of interest, 3-4 GHz, the Weyl formula [201] predicts

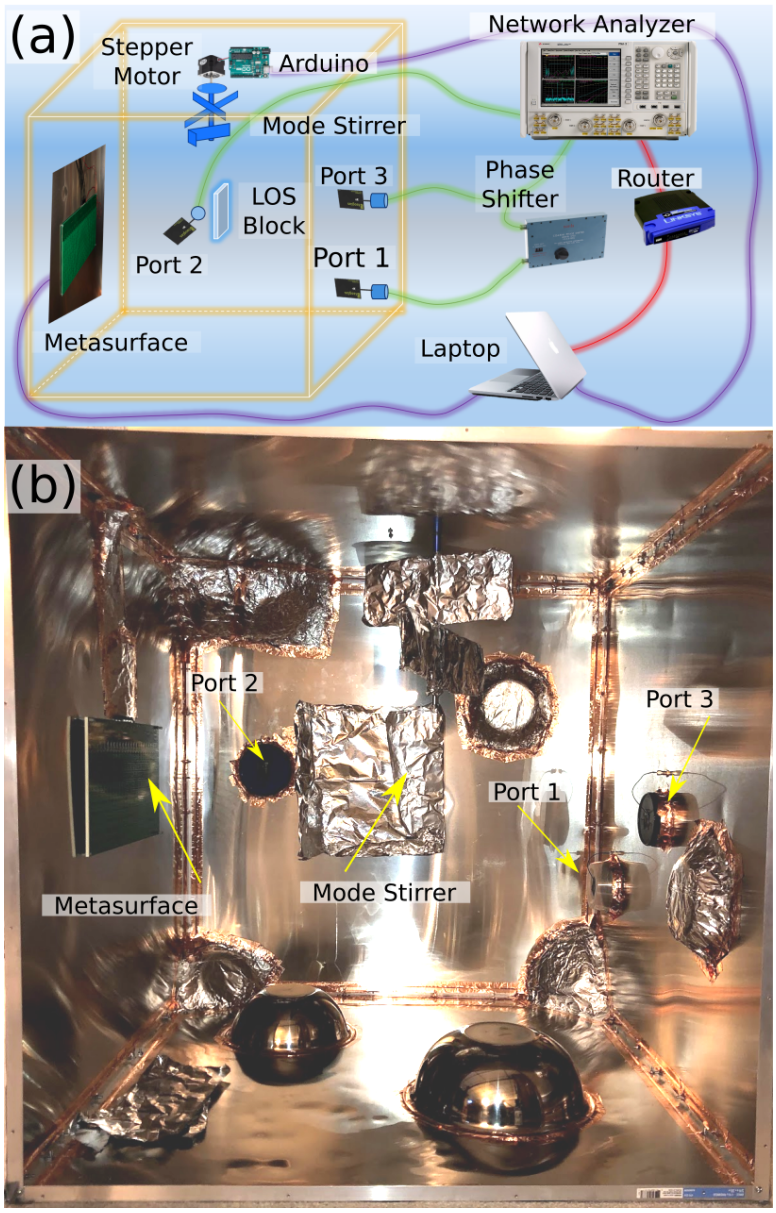

Figure F.1: **Cavity configuration.** **(a)** Experimental schematic of the cavity, showing the metasurface installed on the cavity walls, the locations of the 3 ports, the line-of-sight (LOS) block to prevent direct transmission between Port 2 and Ports 1 and 3, and the mode stirrer that is controlled by a stepper motor through an Arduino. Also shown are the network analyzer, phase shifter, control laptop and router. **(b)** Photograph of the interior of the cavity showing the components from the schematic as well as the irregular scatters that were installed on the cavity walls.

approximately 8524 resonant modes of the complex enclosure and the measured quality factor of the cavity is roughly 5.5×10^3 [24]. A resonance mode width is then ~ 5 times greater than the mean mode spacing, which means there is some local overlap between modes.

F.2 Metasurface Binning

The metasurface was binned to combine elements and start with a simpler configuration before moving up to more complicated configurations. Without binning, there are 240 elements arranged in a 10×24 grid, which allows 1.8×10^{72} possible combinations of commands. Binning elements into 2×2 groups results in 60 elements in a 5×12 grid and 1.2×10^{18} possible combinations, while binning elements into 3×3 groups results in 24 elements in a 3×8 grid and 1.7×10^7 possible combinations. In the 3×3 binning configuration, the bottom row consists of 4×3 groups to ensure all the elements are included. Finally, binning elements into 5×4 groups results in 12 elements in a 2×6 grid and 4096 possible combinations. These binning configurations are shown in Fig. 4.1.

Binning is an important capability that allows us to adapt the size of an effective element to the underlying scattering system. This is one of the major contributions of our work.

F.3 Data Preparation and Collection

An important step for deep learning is preparation of the measured data. The goal here is to represent the data in a basis set that can be ingested by the

neural network architecture. The raw data consists of M sets of complex two-port S-parameter values, each containing 32,001 points measured over a 3-4 GHz window. We are interested in the relationship between metasurface commands and transmission between the ports, so we select S_{21} as the primary variable of interest. The measured data contains local and global correlations, both of which must be captured by the deep learning network. We can exploit the local correlations with 1D convolutional neural network (CNN) layers, but would like the individual windows to cover a smaller bandwidth. Our previous work showed diminishing returns for optimization over bandwidths greater than 10 MHz [24], so 10 MHz provides a reasonable limit for the local window size. We therefore extract the complex S_{21} in 10 MHz frequency windows at 100 distributed center frequencies to provide 100 feature vectors containing 321 points each. A representative data set is shown in Fig. F.2 (a), with only 50 feature vectors used for illustration. The data are organized into a 3D structure of M sets of data $\times$ F local frequencies $\times$ N features, or $10,000 \times 321 \times 100$ for the 2×2 binning configuration. Each data set takes on a pseudo-2D format with a 321×100 pixel "image" as shown in Fig. F.2 (b). Local features in the 10 MHz frequency windows (over the F dimension) will be extracted by 1D CNN layers and global features (over the N dimension) will be extracted by the overall deep learning network architecture. The overall architecture then acts as a dense or fully connected layer from the perspective of the global features. The pseudo-2D format and its ability to capture both long-range and short-range correlations in frequency provides the second major novel aspect of our approach.

The output values of the deep learning network (equal in number to the num-

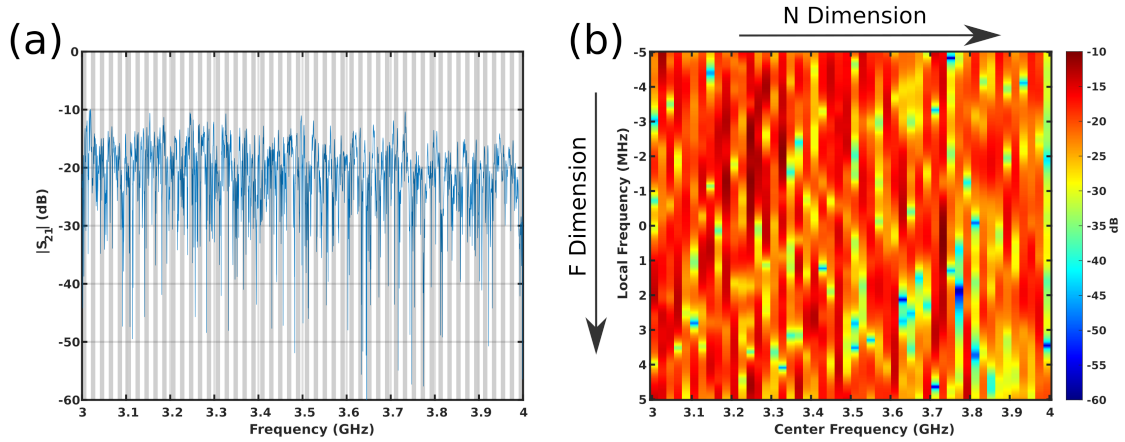

Figure F.2: **Data preparation example.** The deep learning networks use complex amplitudes, however, only the magnitude is shown here for illustrative purposes. **(a)** Raw $|S_{21}|$ data vs. frequency showing 50 local windows highlighted in gray. The actual data preparation uses 100 local windows, only 50 are shown here for clarity. The data was collected over a 1 GHz measurement window with 32,001 points and each local window (highlighted in gray) has a bandwidth of 10 MHz or 321 points. **(b)** Extracted $|S_{21}|$ data (in log scale) in a pseudo-2D format as a 321×50 pixel "image". The data is represented as center frequency (over the full 1 GHz measurement window or N dimension) vs. local frequency (over the 10 MHz local window or F dimension). The deep learning network will use 1D convolutions to extract features in the 10 MHz local frequency windows (y-axis) and use the relationships between convolutional filters to capture global correlations over the full 1 GHz measurement window (x-axis).

ber of binned metasurface elements) are floating point numbers rather than binary numbers and can be interpreted as the probability that a given element in the metasurface is active (set to 1). The determined commands are then found by rounding the outputs to either a 1 or a 0. Inspection of the raw (unrounded) outputs allows us to assess how correct the deep learning network was, or how confident the network was in the result. Details of the network training approach are provided in Section F.5.

A major concern with deep learning is the amount of data required for training, which grows with the complexity of the problem being solved. To work within the constraint of reasonable training time, we wish to limit the number of data sets that must be collected. Therefore, acquiring good training data is of critical importance to ensure we cover the full range of possible responses. As found in earlier work [24], a diverse set of measurements requires variations in the number of active elements, spatial frequencies of active elements, and local groupings of active elements. Therefore, we utilized a random biased coin toss approach with the bias itself a uniformly distributed random number to assign values to the elements for training data generation. To speed up operation as much as possible, the microwave network analyzer was configured to only provide S_{21} measurements vs. frequency. With averaging disabled, collecting 4,000 sets of data took a little under an hour and a half, while collecting 10,000 sets of data took roughly 3.5 hours. Training was performed on a computer running Ubuntu 20.04 equipped with an NVIDIA RTX 3080 GPU, and took roughly 30 minutes for 10,000 training sets.

For the initial experiment, we collected 4000 sets of data in each of the specified

binning configurations; the 5×4 configuration allows 4096 unique metasurface combinations, so we collected 4096 sets (covering all possible combinations) in that case. With the exception of the 5×4 binning configuration, the number of sets collected was far smaller than the number of possible configurations of the metasurface.

F.4 Deep Learning and Neural Network Layers

Neural networks come in many different shapes and sizes; no single type is optimal for all problems, a consequence of the "No Free Lunch" theorem [145]. The networks described in this thesis utilize 4 different types of layers as disscused in Section 2.7: 1) Dense, linear, or fully connected layers characterized by the number of neurons. The output is a linear combination of the inputs; 2) Convolutional layers characterized by the number of filters and the length of the kernel. The output is the result of convolving the inputs with the kernels; 3) Pooling layers characterized by the pool size. The output is either the maximum or average value over a sliding window of width given by the pool size. These layers serve to reduce the size of the feature map and help ensure the learning process is position invariant; and 4) Dropout layers characterized by the drop out rate. Dropout layers randomly set the specified percentage of inputs to 0 at each iteration in the training process, providing coarse regularization and simplifying the model.

Identifying the optimal deep learning network topology for a given binning configuration took a significant amount of time to iterate over many potential designs, e.g., for our computing resources, often several weeks, and was performed off-line. Once the deep learning network architecture was determined, we switched to an on-

line, closed loop configuration where the data collection and training processes were separated by a few hours rather than days or weeks. The determined metasurface commands were directly applied to the metasurface and the resulting S_{21} responses were measured by the network analyzer, closing the loop. The on-line configuration also serves as a "field-test" for the deep learning network, further validating it against data not seen during training, as well as testing performance against potential small variations in measurement noise and the scattering configuration of the cavity itself.

F.5 Network Training Setup

For training the networks, the data was split into 75% training data and 25% validation data. The validation data is used to score the performance after each training run and is not used during the training process itself, so that validation is unbiased. The data was randomly shuffled prior to splitting and each network was trained several times to ensure results were in family and that the training process was unbiased as well.

To score the performance, we need to define metrics for loss and accuracy. The loss function was selected as mean absolute error to emphasize outliers in the data and we define accuracy as the fraction of sets of commands that were predicted without error. To clarify the difference, the loss function is defined as the average of the sum of the absolute value of the true commands, T_j, subtracted from the predicted commands, P_j, for set j, computed over N sets of M elements.

$$L = \frac{1}{NM} \sum_{j} |P_j - T_j| \tag{F.1}$$

The loss function is then computed on a per element basis and tells us how close the prediction was on average for each element. The output of the network is floating point rather than binary, so the loss function does not necessarily provide an indication of the total number of incorrect predictions. The accuracy metric is defined as the percentage of sets that were predicted without a single error. It is evaluated on a per set basis and explicitly uses the rounded output (0 or 1) from the network. Because accuracy is computed on a per set basis, it is dependent on the number of elements in a command set and provides a more conservative estimate of performance for the various binning configurations. Accuracy is also more volatile, especially when the loss function is large. The loss function is continuous and more appropriate for training where we need to compute a gradient, while accuracy is a better metric for scoring the overall performance.

The networks were trained for 100-200 epochs using stochastic gradient descent (SGD) with momentum. The basic SGD algorithm has potential problems with pathological curvature, or narrow ravines, which are common around local optima, and the response tends to oscillate back and forth across the ravine. To address this, we can use momentum [209], effectively forgetting a portion of the previous gradient. Momentum can be thought of as a very coarse approximation of the curvature or 2nd derivative. To accelerate the training, we explicitly use Nesterov momentum [210].

The networks were trained in batches, meaning multiple data sets were eval-

uated at each iteration prior to updating the weights. This allows several samples of data to be processed simultaneously so that the effect of changes in the weights are observed over multiple sets of data, improving robustness and desensitizing the response to noise [211]. A batch size of 100 was used by default.

To prevent the networks from simply training on noise, we introduce an additional regularization step on the loss function. By enforcing an $\mathcal{L}_2$ regularization scheme, the regularized loss function L^* is computed from the loss function L, and the vector of weights for the current iteration, $\mathbf{w}_i$, as $L^* = L + \lambda||\mathbf{w}_i||^2$. The value λ is referred to as a weight decay. The learning rate, γ, is the step size along the gradient, so the weights are incremented at each iteration as

$$\mathbf{w}_{i+1} = \mathbf{w}_i - \gamma\nabla_{\mathbf{w}_i}L - 2\gamma\lambda\mathbf{w}_i \tag{F.2}$$

Finally, the learning rate is stepped down when the loss function plateaus, which allows the network to continue learning when it stalls due to the rate being too high.

F.6 Complex Network Layers and Existing Deep Learning Frameworks

With complex values, the mechanics of a network layer are the same as for the real-valued counterpart but they incur four times the computational cost due to having both real and imaginary components as well as the cross-terms. Our initial deep learning implementation leveraged Keras [144] and TensorFlow [143]. These provide an excellent, high level framework that is very easy to use. Unfortunately

this ease of use complicates things when attempting to develop custom complex-valued modules. Complex dense layers are straightforward to implement, but batch normalization, convolution, and recurrent layers are not. While there are repositories with complex deep networks containing some of these modules in Keras [212] and Caffe [213], they are not actively maintained and are not formally supported by the frameworks. In the case of Keras, changes to the way the backend is handled in the most recent version (v2.4) mean that the complex library [212] is no longer functional and would require significant modification to bring up to date.

This leads us to utilize PyTorch [214], another deep learning framework. The interface to PyTorch is lower level than Keras, which means it requires more knowledge of Python to use effectively, but that it is also easier to implement custom modules. In addition, there is an open source complex library written by Sebastien Popoff [185] that includes complex versions of dense, convolutional, and batch normalization layers. We were able to utilize this library with only minor modifications to the batch normalization implementation to handle our multiple feature data sets.

Figure F.3 shows the impact of using complex network layers. In each of the panels, the blue plots indicate results for a real-valued network while the red plots indicate results for a complex-valued network. In addition, the solid lines show results for the validation set while the dashed lines show the results for the training set. The complex-valued networks all converged faster than the real-valued networks and the complex-valued network achieved better accuracy on the training set than the real-valued network did for the 2×2 binning case. The only differences in training were for the real-valued network in the 2×2 binning case. The two differences here

were that 1) the "patience" parameter, or number of epochs to wait before reducing the learning rate had to be increased significantly because the training converged slowly even with an aggressive learning rate; and 2) the number of epochs had to be increased from 100 to 300 in order to capture the converged model. The increased patience parameter leads to a longer period of oscillations in the validation set loss function.

The acceleration in training comes with a caveat in that the overall computational time for the purely real-valued deep learning network is still less than that of the complex-valued deep learning network. Complex-valued layers increase the computation requirements for multiplication and convolution by a factor of 4 to handle the real and imaginary terms as well as the cross-terms. In addition, highly optimized and efficient implementations of purely real-valued layers are readily available through the NVIDA CUDA deep neural network library (cuDNN), but are not available for their complex-valued counterparts.

F.7 Network Architecture for Sequential Layers

Figure F.4 presents the generalized architecture used for sequential layers. The input consists of N_i feature vectors containing the local 10 MHz windows with F points in each vector. 1D CNN layers with N_l filters and a kernel length of k_l at the l^{th} layer perform the feature extraction. As shown in the lower inset, each CNN layer includes a 1D convolution followed by a batch normalization and a rectified linear unit (ReLU) activation function. The batch normalization is used to ensure the distribution of the data (mean and variance) remains relatively constant throughout

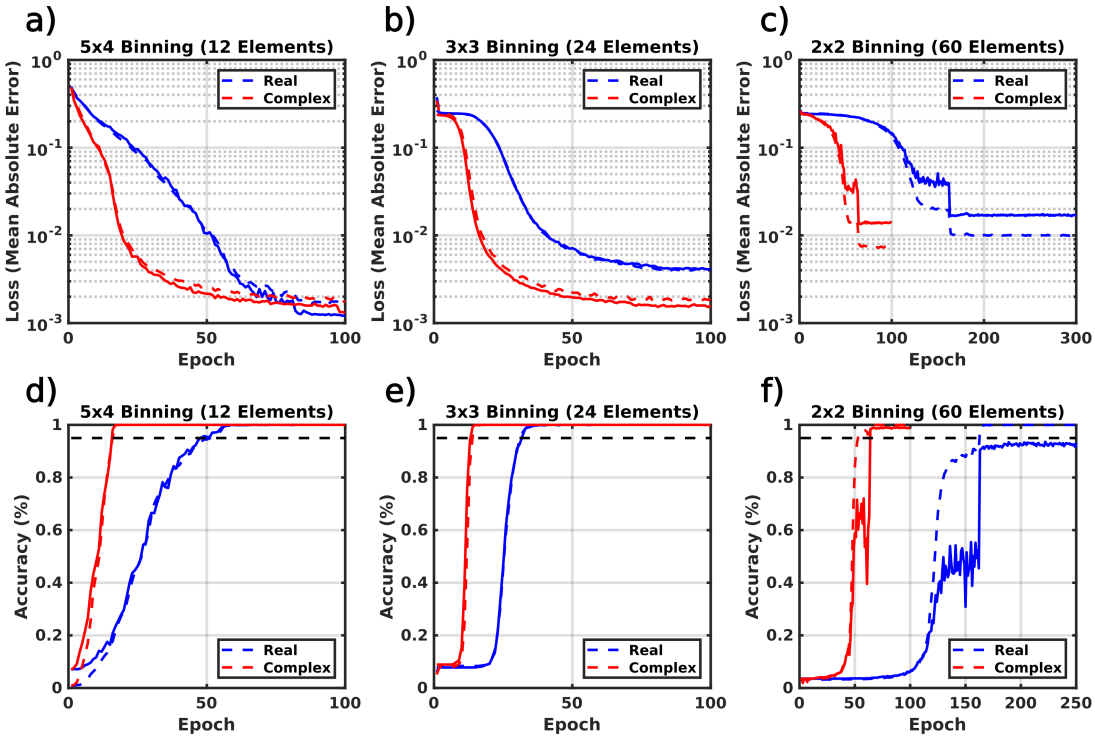

Figure F.3: **Impact of using complex network layers.** **a)** through **c)** Evolution of the loss function (mean absolute error) for the 5×4, 3×3, and 2×2 binning cases. **e)** through **f)** Evolution of the accuracy for the 5×4, 3×3, and 2×2 binning cases. The blue lines indicate real-valued network layers, the red lines indicate complex-valued network layers, the dashed lines indicate the training set, and the solid lines indicate the validation set. The dashed black line on the accuracy plots indicates 95% accuracy. Training was performed for 100 epochs in all cases except for the 3×3 binning case with real-valued network layers, which was trained for 300 epochs. In each case, training with the complex-valued layers converged faster than training with the real-valued layers. For the 2×2 binning case, the complex-valued network achieved higher accuracy for the training set (99.2%) than the real-valued network did (93.6%).

the network; changing distributions between layers induces internal covariate shift and leads to convergence issues during training [150]. The ReLU activation function is used in virtually all deep learning networks as it does not experience a vanishing gradient due to saturation, and leads to expedited convergence and generally better solutions than sigmoid like functions [215].

The Python code implementing the sequential network is provided in Listing F.1, and the code that implements the combined complex convolutional layer utilizing batch normalization and a ReLu activation function is provided in Listing F.2. Finally, the code that implements the hybrid 1D complex batch normalization is provided in Listing F.3. "Hybrid" here indicates batch normalization for a 1D signal with multiple features.

Listing F.1: Sequential Network Implementation

```
1
2 class SequentialComplexCNN(nn.Module):
3     def __init__(self,din,dout):
4         super(SequentialComplexCNN, self).__init__()
5         self.cnn11 =  complexConvBatchNormReLU1D(din, 32, 3)
6         self.cnn12 =  complexConvBatchNormReLU1D(32, 32, 5)
7         self.cnn13 =  complexConvBatchNormReLU1D(32, 64, 3)
8         self.cnn14 =  complexConvBatchNormReLU1D(64, 64, 5)
9
10        self.cnn21 =  complexConvBatchNormReLU1D(64, 64, 3)
11        self.cnn22 =  complexConvBatchNormReLU1D(64, 64, 5)
12        self.cnn23 =  complexConvBatchNormReLU1D(64, 128, 3)
13        self.cnn24 =  complexConvBatchNormReLU1D(128, 128, 5)
14
15        self.cnn31 =  complexConvBatchNormReLU1D(128, 128, 3)
16        self.cnn32 =  complexConvBatchNormReLU1D(128, 128, 5)
17        self.cnn33 =  complexConvBatchNormReLU1D(128, 64, 3)
18        self.cnn34 =  complexConvBatchNormReLU1D(64, 64, 5)
19
20        self.cnn41 =  complexConvBatchNormReLU1D(64, 64, 3)
21        self.cnn42 =  complexConvBatchNormReLU1D(64, 64, 5)
22        self.cnn43 =  complexConvBatchNormReLU1D(64, 32, 3)
23        self.cnn44 =  complexConvBatchNormReLU1D(32, 32, 5)
```

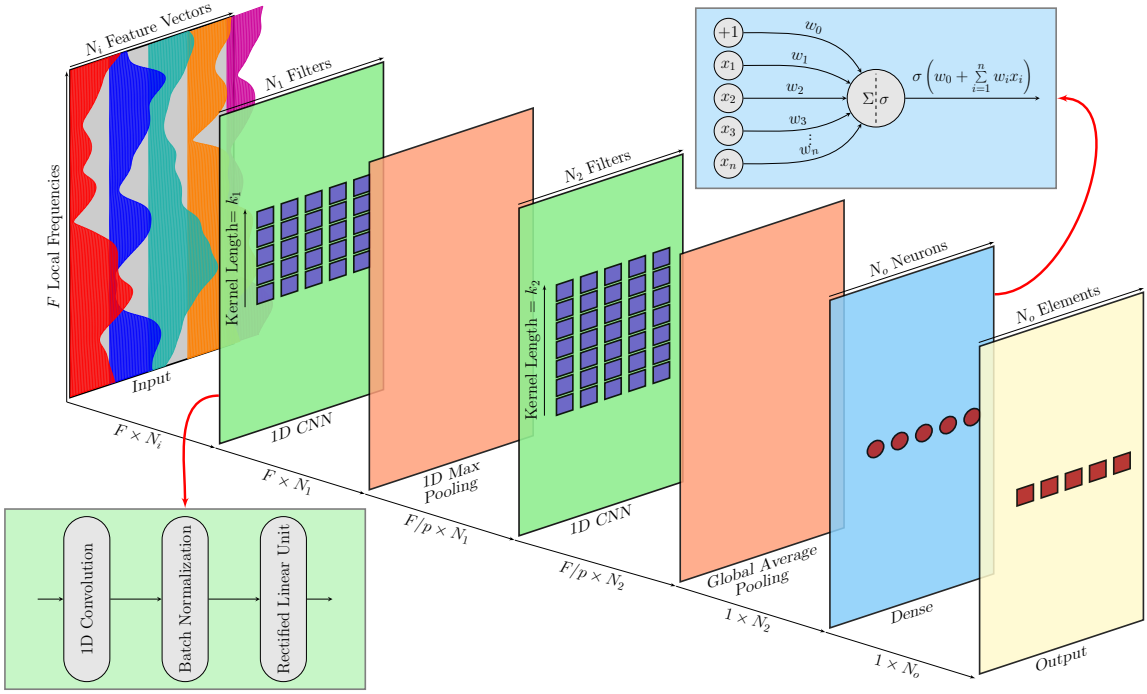

Figure F.4: **Sequential network layer architecture.** The input layer consists of N_i feature vectors containing S_{21} measurements in a local 10 MHz window of F points, for a total size of $F \times N_i$. This is followed by a series of 1D convolutional layers defined by the number of filters and the kernel length. Each convolutional layer includes a 1D convolution, a batch normalization to keep the distribution statistics constant throughout the network, and a rectified linear unit activation function. Interspersed with the convolutions are 1D max pooling layers defined by the pool size, p, that serve to reduce the dimensionality of the local frequency window. The output stage consists of a global average pooling layer to further reduce dimensionality and convert the complex-valued signals to magnitude, followed by a dense or fully-connected layer to ensure the correct number of outputs. The dense layer produces outputs that are linear combinations of the outputs from the global average pooling layer, and is followed by a sigmoid activation function to approximate binary values at the output.

```
24
25
26          self.denseOut = nn.Linear(32,dout)
27          self.sigmoid = nn.Sigmoid()
28
29
30
31      def forward(self,x):
32          xr = x.real
33          xi = x.imag
34          dr = 0.2
35
36          #stage 1
37          xr,xi = self.cnn11(xr,xi)
38          xr,xi = self.cnn12(xr,xi)
39          xr,xi = self.cnn13(xr,xi)
40          xr,xi = self.cnn14(xr,xi)
41
42          xr,xi = complex_max_pool1d(xr,xi, 2)
43          xr,xi = complex_dropout(xr,xi,dr)
44
45          #stage 2
46          xr,xi = self.cnn21(xr,xi)
47          xr,xi = self.cnn22(xr,xi)
48          xr,xi = self.cnn23(xr,xi)
49          xr,xi = self.cnn24(xr,xi)
50
51          xr,xi = complex_max_pool1d(xr,xi, 2)
52          xr,xi = complex_dropout(xr,xi,dr)
53
54          #stage 3
55          xr,xi = self.cnn31(xr,xi)
56          xr,xi = self.cnn32(xr,xi)
57          xr,xi = self.cnn33(xr,xi)
58          xr,xi = self.cnn34(xr,xi)
59
60          xr,xi = complex_max_pool1d(xr,xi, 2)
61          xr,xi = complex_dropout(xr,xi,dr)
62
63          #stage 4
64          xr,xi = self.cnn41(xr,xi)
65          xr,xi = self.cnn42(xr,xi)
66          xr,xi = self.cnn43(xr,xi)
67          xr,xi = self.cnn44(xr,xi)
68
69          xr,xi = complex_max_pool1d(xr,xi, 2)
70
71          x = torch.sqrt(torch.pow(xr,2)+torch.pow(xi,2))
72          x = x.mean([2])
73          x = self.denseOut(x)
74          x = self.sigmoid(x)
75          return x
```

Listing F.2: Complex Convolutional Batch Norm ReLu Layer Implementation

```
class complexConvBatchNormReLU1D(nn.Module):
        def __init__(self,din,Nf, k):
            super(complexConvBatchNormReLU1D,self).__init__()
            self.c1 = ...
               ComplexConv1d(din,Nf,k,1,int(np.floor((k-1)/2)))
            self.bn1 = ComplexBatchNormHybrid(Nf)


        def forward(self,xr,xi):

            xr,xi = self.c1(xr,xi)
            xr,xi = self.bn1(xr,xi)
            xr,xi = complex_relu(xr,xi)

            return xr,xi
```

Listing F.3: Hybrid Complex 1D Batch Normalization Implementation

```
class ComplexBatchNormHybrid(_ComplexBatchNorm):

    def forward(self, input_r, input_i):
        assert(input_r.size() == input_i.size())
        assert(len(input_r.shape) == 3)
        exponential_average_factor = 0.0

        if self.training and self.track_running_stats:
            if self.num_batches_tracked is not None:
                self.num_batches_tracked += 1
                if self.momentum is None:  # use cumulative ...
                   moving average
                    exponential_average_factor = 1.0 / ...
                       float(self.num_batches_tracked)
                else:  # use exponential moving average
                    exponential_average_factor = self.momentum


        if self.training:

            # calculate mean of real and imaginary part
            mean_r = input_r.mean([0, 2])
            mean_i = input_i.mean([0, 2])

            mean = torch.stack((mean_r,mean_i),dim=1)

            # update running mean
            with torch.no_grad():
                self.running_mean = exponential_average_factor * ...
                   mean\
                    + (1 - exponential_average_factor) * ...
                       self.running_mean
```

```
29
30              input_r = input_r-mean_r[None, :, None]
31              input_i = input_i-mean_i[None, :, None]
32
33              # Elements of the covariance matrix (biased for train)
34              n = input_r.numel() / input_r.size(1)
35              Crr = 1./n*input_r.pow(2).sum(dim=[0,2])+self.eps
36              Cii = 1./n*input_i.pow(2).sum(dim=[0,2])+self.eps
37              Cri = (input_r.mul(input_i)).mean(dim=[0,2])
38
39              with torch.no_grad():
40                  self.running_covar[:,0] = ...
                        exponential_average_factor * Crr * n / (n - 1)\
41                       + (1 - exponential_average_factor) * ...
                            self.running_covar[:,0]
42
43                  self.running_covar[:,1] = ...
                        exponential_average_factor * Cii * n / (n - 1)\
44                       + (1 - exponential_average_factor) * ...
                            self.running_covar[:,1]
45
46                  self.running_covar[:,2] = ...
                        exponential_average_factor * Cri * n / (n - 1)\
47                       + (1 - exponential_average_factor) * ...
                            self.running_covar[:,2]
48
49          else:
50              mean = self.running_mean
51              Crr = self.running_covar[:,0]+self.eps
52              Cii = self.running_covar[:,1]+self.eps
53              Cri = self.running_covar[:,2]
54
55              input_r = input_r-mean[None,:,0,None]
56              input_i = input_i-mean[None,:,1,None]
57
58          # calculate the inverse square root of the covariance matrix
59          det = Crr*Cii-Cri.pow(2)
60          s = torch.sqrt(det)
61          t = torch.sqrt(Cii+Crr + 2 * s)
62          inverse_st = 1.0 / (s * t)
63          Rrr = (Cii + s) * inverse_st
64          Rii = (Crr + s) * inverse_st
65          Rri = -Cri * inverse_st
66
67          input_r, input_i = Rrr[None,:,None]*input_r + ...
                Rri[None,:,None]*input_i, \
68                              Rii[None,:,None]*input_i + ...
                                    Rri[None,:,None]*input_r
69
70          if self.affine:
71              input_r, input_i = self.weight[None,:,0,None] *i ...
                    nput_r + self.weight[None,:,2,None] * input_i + ...
                    \ self.bias[None,:,0,None], \ ...
                    self.weight[None,:,2,None] * input_r + ...
```

```
                    self.weight[None,:,1,None] * input_i + \ ...
                    self.bias[None,:,1,None]
72
73          return input_r, input_i
```

The CNN layers are grouped together to form stages and are interspersed with 1D max pooling layers defined by the pool size, p. Also included are dropout layers for regularization, which are not explicitly shown in Fig. F.4. The output stage contains a global average pooling layer that averages along the F dimension, reducing the feature maps to a single dimension. It also converts the complex-valued signals to magnitude and is followed by a dense layer that provides the correct number of outputs, N_o. As shown in the upper inset, the output of the dense layer is a linear combination of the outputs from the global pooling layer, with a sigmoid activation function used to clip the output between 0 and 1.

F.8 Offline Training Results for 5 x 4 Binning

When using 5×4 binning, the metasurface is effectively partitioned into 12 elements, for 4096 possible sets of commands. We measured all 4096 combinations, and the 75%/25% split yielded 3072 sets for training and 1024 sets for validation. The batch size was set to 64 as a result. A purely sequential deep network was utilized, following the layout given in Fig. F.4. Four CNN layers, a max pooling layer, and a dropout layer were combined into a stage. Four stages were then used, with the output provided by a global average pooling layer and a dense layer with a sigmoid activation function.

Initial experiments used purely real-valued deep learning layers, in which case

the global average pooling layer only provided dimensionality reduction. We were able to establish excellent prediction performance, regularly achieving < 10 total prediction errors over the validation set or $> 99\%$ accuracy after training for 500 epochs. When switching to complex-valued layers with the same architecture, we were able to regularly achieve perfect prediction (100%) over both the training and validation sets after training for fewer than 100 epochs. This improvement demonstrates that the complex-valued deep learning network is able to exploit phase as well as amplitude to better fit the relationship between metasurface commands and transmitted power.

The training results for the 5×4 binning case with complex-valued layers are shown in Fig. F.5. Figure F.5 (a) shows the loss function evolution for the training and validation sets while Fig. F.5 (b) shows the accuracy evolution. The performance is excellent, achieving perfect prediction over both the training and validation data sets. Note that the loss function continues to improve even after the accuracy saturates at 100%. This is because the loss function is continuous and computed using the floating point predicted values rather than the rounded binary values; it shows that the network is still learning and continuing to increase its confidence in the prediction.

F.9 Inception and Terrapin Modules

An inception module is designed to promote sparse feature representation using available dense components [186] and works by optimizing the receptive field coverage of a convolutional network. The receptive field is the number of points in

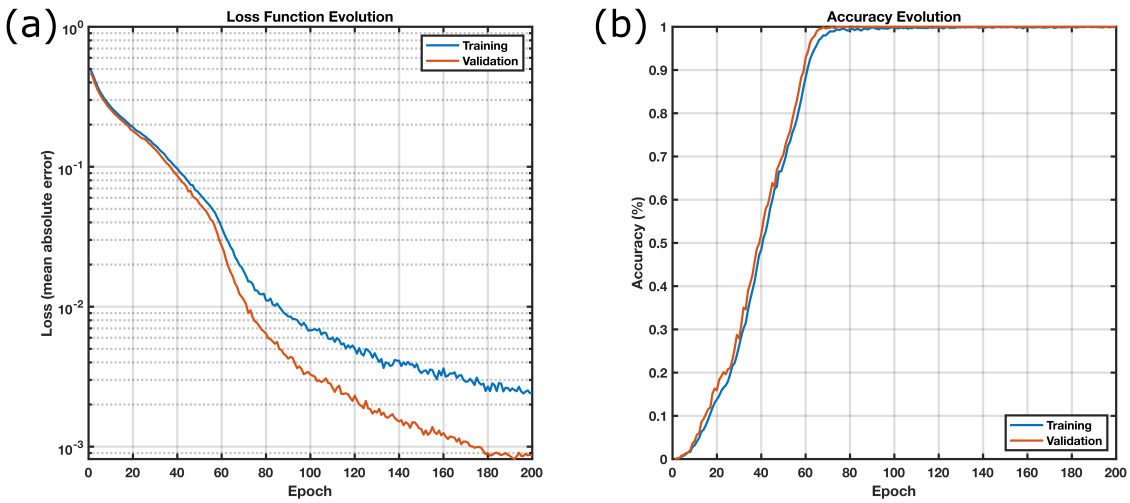

Figure F.5: **Deep learning performance with complex-valued layers for 5x4 Binning.** **(a)** Evolution of the loss function for the training and validation sets over 200 epochs. The loss function measures the average prediction error per element and provides an estimate of the confidence in the prediction. **(b)** Evolution of the accuracy for the training and validation sets over 200 epochs. Accuracy provides the relative number of sets of commands that were predicted without error, and shows that perfect prediction was achieved on both the training and validation sets in less than 100 epochs. The loss function continues decreasing after the accuracy saturates at 100% because it is continuous and evaluated on the floating point predicted values and the decrease indicates the network is still learning and improving its estimate.

input space that contribute to a point at a given layer of the deep learning network and is described in detail in Section S11 of the supplemental material. Through the use of parallelization and concatenation, the receptive field sizes at a layer are conserved for subsequent layers to utilize, extracting features through the width of the deep learning network as well as its depth.

The original inception module was developed for image processing and operates in a true 2D space, with full 2D convolutions. It uses 4 parallel paths with CNN layers containing unit length kernels for buffering and conditioning, along with 3 and 5 sample length kernels for feature extraction, and a max pool layer to improve performance [186]. There have been several variations of the inception module; however, none operate in the pseudo-2D space we desire. For our "images", the frequency spacing along columns is the resolution of the network analyzer (31.25 kHz), while the frequency spacing along rows is the separation between local windows (10 MHz). The difference in sampling means we need to treat the rows and columns accordingly and avoid traditional 2D convolutions that assume uniform sampling. Therefore, we modified the general architecture of the inception module to perform 1D convolutions over the 10 MHz local frequency windows. The 1D convolutional filters then extract local features over the 10 MHz windows, while the relationship between the filters acts as a dense or fully connected layer, extracting global features over the full 1 GHz measurement window.

From previous work with the cavity, we found that the mean mode spacing is $\sim$125 kHz and we demonstrated the ability to generate strong nulls over a 500 kHz bandwidth [24]. This suggests we should use a pooling window of 125 kHz

and allow the receptive field to increase by 125 kHz and 500 kHz at each stage, or layer in the deep learning network. After experimenting, we found that adding a 5th stage which increased the receptive field by 1 MHz helped to further improve performance. We refer to the final version of our module as a "Terrapin Module", a block diagram of which is shown in Fig. 4.2.

The Python code implementing the Terrapin Module is provided in Listing F.4.

Listing F.4: Terrapin Module Implementation

```
class TerrapinModule(nn.Module):
    def __init__(self,din,N11,N21,N22,N31,N32,N42,N51,N52):
        super(TerrapinModule, self).__init__()

        k1 = 1
        k2 = 5
        k3 = 17
        k4 = 33
        s = 1
        p1 = np.floor((k1-1)/2).astype(int)
        p2 = np.floor((k2-1)/2).astype(int)
        p3 = np.floor((k3-1)/2).astype(int)
        p4 = np.floor((k4-1)/2).astype(int)

        Nd = N11 + N22 + N32 + N42 + N52

        self.s1c1a =  ComplexConv1d(din, N11, k1,s,p1)
        self.bns1c1a = ComplexBatchNormHybrid(N11)

        self.s1c1b =  ComplexConv1d(din, N21, k1,s,p1)
        self.bns1c1b = ComplexBatchNormHybrid(N21)
        self.s1c2b =  ComplexConv1d(N21, N22, k2,s,p2)
        self.bns1c2b = ComplexBatchNormHybrid(N22)

        self.s1c1c =  ComplexConv1d(din, N31, k1,s,p1)
        self.bns1c1c = ComplexBatchNormHybrid(N31)
        self.s1c2c =  ComplexConv1d(N31, N32, k3,s,p3)
        self.bns1c2c = ComplexBatchNormHybrid(N32)

        self.s1c2d =  ComplexConv1d(din, N42, k1,s,p1)
        self.bns1c2d = ComplexBatchNormHybrid(N42)

        self.s1c1e =  ComplexConv1d(din, N51, k1,s,p1)
```

```
34          self.bns1c1e = ComplexBatchNormHybrid(N51)
35          self.s1c2e =  ComplexConv1d(N51, N52, k4,s,p4)
36          self.bns1c2e = ComplexBatchNormHybrid(N52)
37
38      def forward(self,xr,xi):
39          dr = 0.15
40          pool_size = 3
41          s = 1
42          padding = np.floor((pool_size-1)/2).astype(int)
43
44          #branch1
45          xr1,xi1 = self.s1c1a(xr,xi)
46          xr1,xi1 = self.bns1c1a(xr1,xi1)
47          xr1,xi1 = complex_relu(xr1,xi1)
48
49          #branch 2
50          xr2,xi2 = self.s1c1b(xr,xi)
51          xr2,xi2 = self.bns1c1b(xr2,xi2)
52          xr2,xi2 = complex_relu(xr2,xi2)
53          xr2,xi2 = complex_dropout(xr2,xi2,dr)
54          xr2,xi2 = self.s1c2b(xr2,xi2)
55          xr2,xi2 = self.bns1c2b(xr2,xi2)
56          xr2,xi2 = complex_relu(xr2,xi2)
57
58          #branch 3
59          xr3,xi3 = self.s1c1c(xr,xi)
60          xr3,xi3 = self.bns1c1c(xr3,xi3)
61          xr3,xi3 = complex_relu(xr3,xi3)
62          xr3,xi3 = complex_dropout(xr3,xi3,dr)
63          xr3,xi3 = self.s1c2c(xr3,xi3)
64          xr3,xi3 = self.bns1c2c(xr3,xi3)
65          xr3,xi3 = complex_relu(xr3,xi3)
66
67          #branch 4
68          xr4,xi4 = complex_max_pool1d(xr,xi, pool_size,s,padding)
69          xr4,xi4 = self.s1c2d(xr4,xi4)
70          xr4,xi4 = self.bns1c2d(xr4,xi4)
71          xr4,xi4 = complex_relu(xr4,xi4)
72
73          #branch 5
74          xr5,xi5 = self.s1c1e(xr,xi)
75          xr5,xi5 = self.bns1c1e(xr5,xi5)
76          xr5,xi5 = complex_relu(xr5,xi5)
77          xr5,xi5 = complex_dropout(xr5,xi5,dr)
78          xr5,xi5 = self.s1c2e(xr5,xi5)
79          xr5,xi5 = self.bns1c2e(xr5,xi5)
80          xr5,xi5 = complex_relu(xr5,xi5)
81
82          xr = torch.cat((xr1,xr2,xr3,xr4,xr5),1)
83          xi = torch.cat((xi1,xi2,xi3,xi4,xi5),1)
84
85          return xr,xi
```

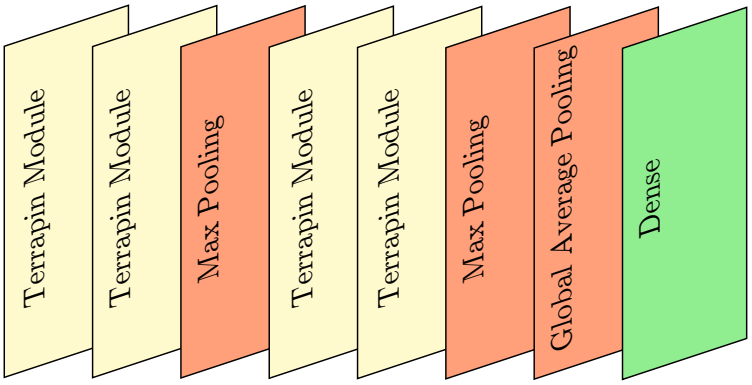

Figure F.6: **Terrapin Network.** Deep learning network utilized for the 3×3 and 2×2 binning configurations. There are 4 Terrapin Modules that are split between max pooling layers. The output stage consists of a global average pooling layer and a dense layer with a sigmoid activation function.

The deep learning network used for the 3×3 and 2×2 binning configurations utilized 4 Terrapin modules and is shown in Fig. F.6, and the code that implements the full network is provided in Listing F.5.

Listing F.5: Terrapin Network Implementation

```
class TerrapinNetwork(nn.Module):
    def __init__(self,din,dout):
        super(TerrapinNetwork, self).__init__()
        print('Building Model: TerrapinNetwork')
        N11a = 64
        N21a = 64
        N22a = 80
        N31a = 32
        N32a = 64
        N42a = 64
        N51a = 32
        N52a = 64
        ia = N11a + N22a + N32a + N42a + N52a

        N11b = 80
        N21b = 80
        N22b = 96
        N31b = 48
        N32b = 96
        N42b = 48
        N51b = 48
        N52b = 80
        ib = N11b + N22b + N32b + N42b + N52b

```

```
        N11c = 96
        N21c = 96
        N22c = 112
        N31c = 64
        N32c = 96
        N42c = 64
        N51c = 64
        N52c = 96
        ic = N11c + N22c + N32c + N42c + N52c

        N11d = 112
        N21d = 112
        N22d = 128
        N31d = 80
        N32d = 112
        N42d = 80
        N51d = 80
        N52d = 112
        id = N11d + N22d + N32d + N42d + N52d

        self.terrapin1 = TerrapinModule(din,N11a,N21a,N22a,N31a,
                                        N32a,N42a,N51a,N52a)
        self.terrapin2 = TerrapinModule(ia,N11b,N21b,N22b,N31b,
                                        N32b,N42b,N51b,N52b)

        self.terrapin3 = TerrapinModule(ib,N11c,N21c,N22c,N31c,
                                        N32c,N42c,N51c,N52c)
        self.terrapin4 = TerrapinModule(ic,N11d,N21d,N22d,N31d,
                                        N32d,N42d,N51d,N52d)

        self.denseOut = nn.Linear(id,dout)
        self.sigmoid = nn.Sigmoid()

    def forward(self,x):
        xr = x.real
        xi = x.imag
        dr = 0.15

        xr,xi = self.terrapin1(xr,xi)
        xr,xi = self.terrapin2(xr,xi)
        xr,xi = complex_max_pool1d(xr,xi, 4)
        xr,xi = complex_dropout(xr,xi,dr)

        xr,xi = self.terrapin3(xr,xi)
        xr,xi = self.terrapin4(xr,xi)

        x = torch.sqrt(torch.pow(xr,2)+torch.pow(xi,2))
        x = x.mean([2])
        x = self.denseOut(x)
        x = self.sigmoid(x)
        return x
```

F.10 Offline Training Results for 3 x 3 Binning

For the 3×3 binning configuration, the purely sequential network did not perform very well and was unable to learn the relationships for either the training or validation sets. This inspired the modified inception module that we defined as the Terrapin Module. The performance difference between the sequential CNN model and the Terrapin Module is shown in Fig. F.7, which presents training results for the 5×4, 3×3, and 2×2 binning cases. The sequential CNN is not able to train very well for the more complicated systems (3×3 and 2×2 binning cases), while the Terrapin Module is able to exploit the more complicated relationships and provide similar performance to the sequential CNN model on the 5×4 binning case.

The results for the 3×3 binning case with complex-valued layers are shown in Fig. F.8. The impact of reducing the learning rate on a plateau can be seen at Epoch 54, where a drop in the learning rate by a factor of 10 induces a drop in the loss function of approximately a factor of 2.

F.11 Offline Training Results for 2 x 2 Binning

For the 2×2 configuration with 4000 sets of data, we were able to achieve >98% accuracy on the training set, but were limited to ∼50% accuracy on the validation set. The discrepancy between training and validation results is a hallmark of overtraining. In this particular case, the validation results were improving but stalled as the training results approached 100% accuracy. The error landscape became extremely small with a negligible gradient, so there was no direction to take

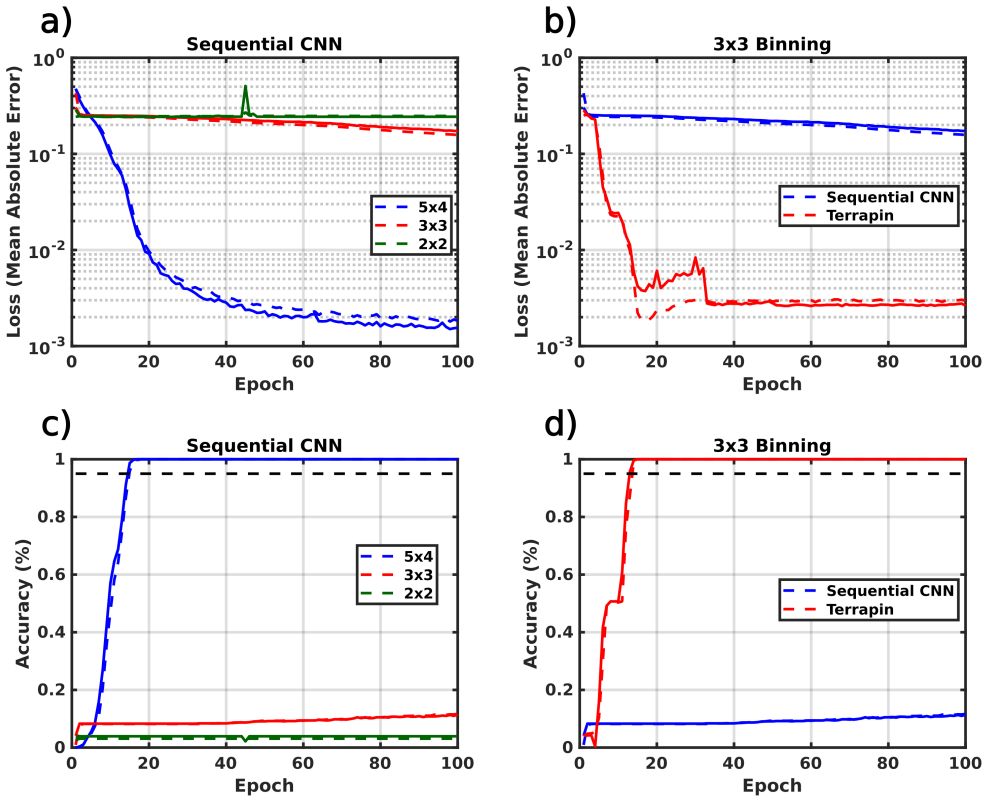

Figure F.7: **Sequential neural network performance with complex scattering systems.** The solid lines indicate results for the validation data while the dashed lines indicate results for the training data. **a)** and **c)** Evolution of the loss function and accuracy for the 5×4, 3×3, and 2×2 binning cases using the sequential CNN model. Only the 5×4 binning case is able to significantly reduce the loss function and provide reasonable accuracy. There is no separation between the validation results and the training results, indicating that there is not an issue with too little data. **b)** and **d)** Evolution of the loss function and accuracy for the 3×3 binning case using the sequential CNN and Terrapin Modules. The Terrapin Module provides similar loss and accuracy to the 5×4 binning case with the sequential CNN.

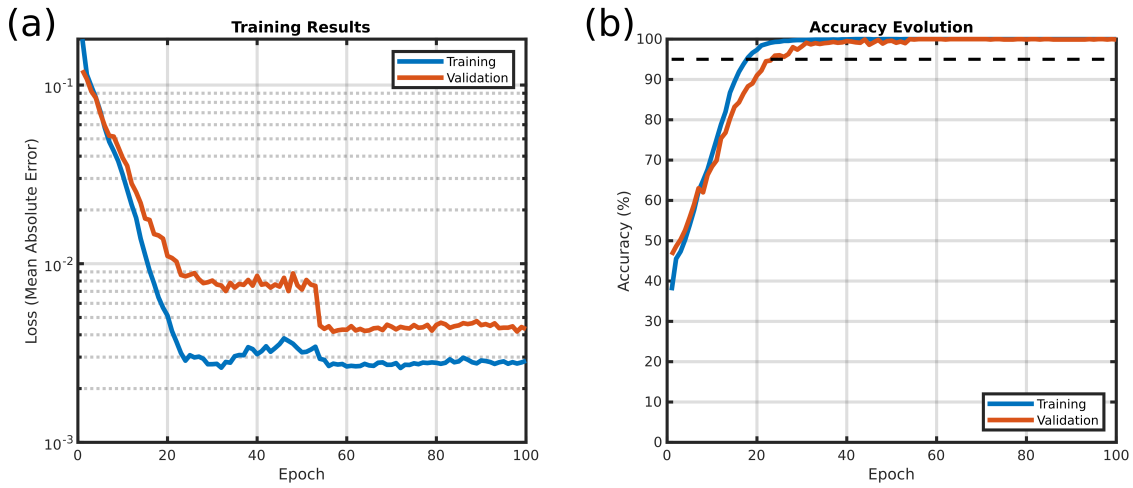

Figure F.8: **Deep learning performance with complex-valued layers for 3x3 Binning.** **(a)** Evolution of the loss function for the training and validation sets over 100 epochs. The loss function hits a plateau at approximately Epoch 33 but shows an additional drop at Epoch 54 when the learning rate is reduced. **(b)** Evolution of the accuracy for the training and validation sets over 100 epochs. Accuracy provides the relative number of sets of commands that were predicted without error, and shows that perfect prediction was achieved on both the training and validation sets in less than 100 epochs. The loss function continues decreasing after the accuracy saturates at 100% because it is continuous and evaluated on the floating point predicted values and the decrease indicates the network is still learning and improving its estimate.

and continue learning. The network therefore learned specific features of the training set rather than general features of the full range of possible responses. This suggests the overtraining is due to having a limited amount of data (only 4000 sets). We captured a larger amount of data (10,000 sets) and were able to achieve >95% accuracy on both the training and validation sets, as shown in Chapter 4, Fig. 4.3. Perfect accuracy for the validation set may be possible with the collection of an even larger amount of data.

F.12 Scattering Fidelity Loss

Figure F.9 shows the decay in scattering fidelity for online validation at the 4, 5, and 9 day marks. The accuracy is still >85% after 5 days, but the number of sets with more than 1 prediction error has increased. After 9 days, the accuracy drops to 65.5% and many cases with 2, 3, and even 4 prediction errors are found.

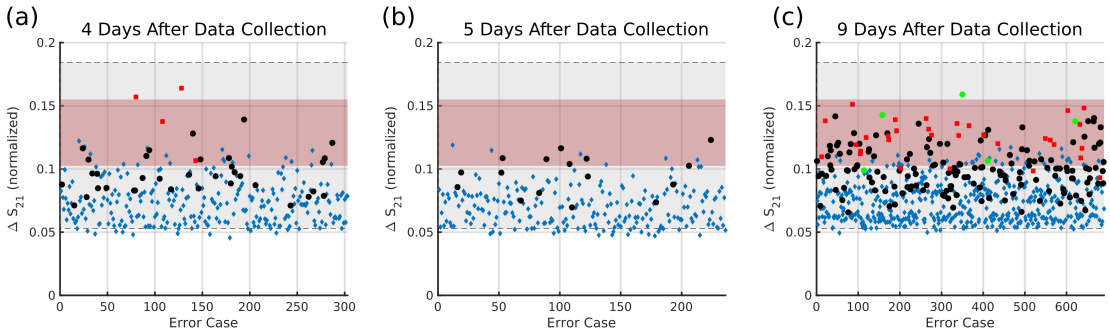

Figure F.9: **Scattering fidelity loss over time.** ΔS_{21} for online validation sets taken a specified time after the training data was collected. The shaded regions show the extent of the single element Hamming distance results from the training data. The grey region shows the full range from maximum to minimum, and the red region shows the 25^{th} and 75^{th} percentiles. The blue diamonds indicate cases with a single prediction error, the black circles indicate cases with 2 prediction errors, the red squares indicate cases with 3 prediction errors, and the green circles indicate cases with 4 prediction errors. These panels show that the ΔS_{21} for prediction errors is very small, and in the lower region of the statistics covered by observed cases with single element Hamming distances. **(a)** Validation 4 days after collecting training data, 2000 sets of commands were tested with 303 mispredictions for an accuracy of 84.9%. **(b)** Validation 5 days after collecting training data, 2000 sets of commands were tested with 236 mispredictions for an accuracy of 88.2%. **(c)** Validation 9 days after collecting training data, 2000 sets of commands were tested with 690 mispredictions for an accuracy of 65.5%.

Appendix G: Monte Carlo Simulations

This appendix presents the background theory for Monte Carlo simulations the produce impedance statistics following the RCM.

G.1 Normalized Impedance Realizations

The development of Monte Carlo simulations for the RCM has been well addressed by previous researchers in the Wave Chaos group at UMD [116]. We will address some specific issues here for performing Monte Carlo simulation with the frequency dependent RCM, and focus on a 2-port system. For a 3D cavity, the system configuration is defined by the loss parameter, α, and the cavity volume, V.

We first need to define the number of modes, M, to use. The mean mode spacing in frequency, Δf, is given by $\Delta f = \pi c^3 \left(2\omega^2 V\right)^{-1}$, so if we are interested in performing a simulation over a specified bandwidth, B_w, the requirement on the number of modes is $M \gg B_w/\Delta f$. A reasonable limit is to let $M \geq 10 B_w/\Delta f$.

We next need to find the eigenvalues following RMT. For the Gaussian orthogonal ensemble (GOE) case, a random matrix $\mathbf{A}$ with the appropriate statistics can be found from a matrix $\mathbf{B}$, whose elements are normally distributed random numbers with 0 mean and unit variance. This results in an $M \times M$ matrix, $\mathbf{A} = 1/2\left[\mathbf{B} + \mathbf{B}^T\right]$. The eigenvalues of $\mathbf{A}$, $\tilde{\lambda} = \mathrm{eig}(\mathbf{A})$, are then the eigenvalues of interest. Explicit eval-

uation of large matrices in this form is computationally expensive. Fortunately, we can generate eigenvalues with the same distribution with a symmetric, tri-diagonal matrix $\mathbf{H}$ [216].

$$\mathbf{H} = \frac{1}{\sqrt{2}} \begin{bmatrix} \mathcal{N}(0,2) & \chi_{M-1} & 0 & \dots & 0 \\ \chi_{M-1} & \mathcal{N}(0,2) & \chi_{M-2} & \ddots & \vdots \\ 0 & \chi_{M-2} & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \mathcal{N}(0,2) & \chi_1 \\ 0 & \dots & 0 & \chi_1 & \mathcal{N}(0,2) \end{bmatrix} \tag{G.1}$$

Here, χ_m is a Chi-distributed random variable with m degrees of freedom. M is again the number of modes, so $\mathbf{H}$ will be an $M \times M$ matrix. Computing the eigenvalues of $\mathbf{H}$ provides the same distribution as the eigenvalues of $\mathbf{A}$, but is significantly faster [216]. The eigenvalues will follow Wigner's semi-circle law, so we need to normalize them to provide eigenvalues with a uniform distribution [116]. Starting with the eigenvalues of $\mathbf{H}$, $\tilde{\lambda} = \text{eig}(\mathbf{H})$, the normalized distribution of eigenvalues, λ, is given by

$$\lambda = \frac{M}{2\pi} \left[\pi + 2\sin^{-1}\left(\frac{\tilde{\lambda}}{\sqrt{2M}} \right) + 2\frac{\tilde{\lambda}}{\sqrt{2M}} \frac{2M - \sqrt{\tilde{\lambda}}}{\sqrt{2M}} \right] - \frac{M}{2} \tag{G.2}$$

Listing G.1 presents Matlab code to determine the RMT eigenvalues. For this work, we are assuming time reversal symmetry holds, so we are interested in the GOE case and beta = 1 in line 7. In the Monte Carlo simulation, we compute the

eigenvalues for all the realizations before computing the normalized impedance.

Listing G.1: Computation of RMT Eigenvalues

```
%compute RMT eigenvalues, fast method from Ming-Jer Lee's ...
    Thesis, Appendix A
%M = number of modes to be included
for cnt=1:N
    %offdiagonal term
    d0= normrnd(0, sqrt (2) ,1 ,M)';
    %diagonal term
    d1= sqrt( chi2rnd( beta *((M-1): -1:1)))';
    %form tri-diagonal matrix
    H= spdiags([[d1 ;0] ,d0 ,[0; d1 ]] ,[ -1 ,0 ,1] ,M,M)/ sqrt (2);
    % now compute the eigenvalues
    eigen(:,cnt)=eig(H);
end
% Transform the eigenvalues to a uniform distribution, following ...
    Wigner's semicircle law
Elevel = (M/(2*pi))*(pi+2*asin(eigen./sqrt(2*2*M))+ ...
2.*(eigen./sqrt(2*2*M)).*sqrt(2*2*M-eigen.^2)/sqrt(2*2*M))-M/2;
```

The β parameter in Listing G.1 references the RMT type, and $\beta = 1$ will follow the GOE distribution. Once the eigenvalues have been found, we can compute the normalized impedance. We need to specify the frequency sampling over B_w, and generate a vector of spatial frequencies, $k = \omega/c$. We also need to find the center of the spatial frequencies, k_c, the mean eigenvalue spacing, $\Delta k_n^2 = 2\pi^2 (kV)^{-1}$ for a 3D cavity, and a pair of random coupling matrices, $\mathbf{W}_1$ and $\mathbf{W}_2$. The coupling matrices will have size $M \times N$ and, for the GOE case, contain normally distributed elements with 0 mean and unit variance. Letting $\mathbf{\Lambda}$ be a diagonal matrix containing the eigenvalues from Listing G.1, we can find the normalized fluctuating impedance, ξ, for each component from port x to port y as

$$\xi_{xy}(k) = -\frac{j}{\pi}\mathbf{W}_x \left[\frac{k^2 - k_c^2}{\Delta k_n^2} + \mathbf{\Lambda} - j\alpha\right]^{-1} \mathbf{W}_y^T \tag{G.3}$$

Listing G.2 presents Matlab code to generate the frequency dependent normalized impedance for the 2-port RCM.

Listing G.2: Frequency Dependent RCM

```
%setup the random coupling matrices for the 2 ports
%M = number of modes to be included
%N = number of realizations to be produced
%k is the wave number (spatial frequency)
%kc is the central wave number (spatial frequency)
%dkn2 is the mean eigenvalue spacing, dkn2 = 2*pi^2/(kc*V) in 3D
CoupleMatrix1=normrnd(0,1,M,N);%coupling matrix of port 1
CoupleMatrix2=normrnd(0,1,M,N);%coupling matrix of port 2
for p=1:length(k)
    green=(1i/pi)./(Elevel+1i*alpha-((k(p)^2-kc^2)/dkn2));% ...
        Setting up green's function
    z11(p,:)=sum(CoupleMatrix1.*green.*CoupleMatrix1);
    z12(p,:)=sum(CoupleMatrix1.*green.*CoupleMatrix2);
    z21(p,:)=sum(CoupleMatrix2.*green.*CoupleMatrix1);
    z22(p,:)=sum(CoupleMatrix2.*green.*CoupleMatrix2);
end
```

G.2 Simulated Cavity Impedance

We can provide a representative cavity impedance from the Monte Carlo results by using the RCM formulation [109] with the radiation resistance, $\mathbf{R}^{\text{rad}}$, and radiation reactance, $\mathbf{X}^{\text{rad}}$.

$$\mathbf{Z}^{\text{cav}} = j\mathbf{X}^{\text{rad}} + \xi\mathbf{R}^{\text{rad}} \tag{G.4}$$

G.3 Relation to Scattering Parameters

The impedance matrix can be converted to scattering parameters through a standard bilinear transformation [121].

$$\mathbf{S} = \left[\sqrt{\mathbf{Z}_0^{-1}}\mathbf{Z}\sqrt{\mathbf{Z}_0^{-1}} - \mathbf{I}\right]\left[\sqrt{\mathbf{Z}_0^{-1}}\mathbf{Z}\sqrt{\mathbf{Z}_0^{-1}} + \mathbf{I}\right]^{-1} \tag{G.5}$$

Generally, the characteristic impedance, $\mathbf{Z}_0$, is a scalar value (nominally 50 Ω) multiplied by the identity matrix, in which case we can simplify this equation.

$$\mathbf{S} = [\mathbf{Z} - \mathbf{Z}_0]\,[\mathbf{Z} + \mathbf{Z}_0]^{-1} \tag{G.6}$$

Bibliography

[1] Zachary B. Drikas, Jesus Gil Gil, Sun K. Hong, Tim D. Andreadis, Jen-Hao Yeh, Biniyam T. Taddese, and Steven M. Anlage. Application of the Random Coupling Model to Electromagnetic Statistics in Complex Enclosures. *IEEE Transactions on Electromagnetic Compatibility*, 56(6):1480–1487, December 2014.

[2] R. D. Leach and Alexander, M. B. Electronic systems failures and anomalies attributed to electromagnetic interference. Technical Report NASA-RP-1374, NASA, 1995.

[3] L. Subrt and P. Pechac. Intelligent walls as autonomous parts of smart indoor environments. *IET Communications*, 6:1004–1010(6), May 2012.

[4] Marci Di Renzo, Merouane Debbah, Dnh-Thuy Phan-Huy, Alessio Zappone, Mohamed-Slim Alouini, Chau Yuen, Vincenzo Sciancalepore, George C. Alexandropoulos, Jakob Hoydis, Haris Gacanin, Julien de Rosny, Ahcene Bounceur, Geoffroy Lerosey, and Mathias Fink. Smart radio environments empowered by reconfigurable ai meta-surfaces: an idea whose time has come. *EURASIP Journal on Wireless Communications and Networking*, 129(1):2450–2525, 2019.

[5] M. Di Renzo, A. Zappone, M. Debbah, M. S. Alouini, C. Yuen, J. de Rosny, and S. Tretyakov. Smart radio environments empowered by reconfigurable intelligent surfaces: How it works, state of research, and the road ahead. *IEEE Journal on Selected Areas in Communications*, 38(11):2450–2525, 2020.

[6] Michael Horodynski, Matthias Kühmayer, Andre Brandstötter, Kevin Pichler, Yan V. Fyodorov, Ulrich Kuhl, and Stefan Rotter. Optimal wave fields for micromanipulation in complex scattering environments. *Nature Photonics*, 14:149–153, May 2020.

[7] Matthew Frazier, Biniyam Taddese, Thomas Antonsen, and Steven M. Anlage. Nonlinear Time Reversal in a Wave Chaotic System. *Physical Review Letters*, 110(6):063902, February 2013.

[8] Bo Xiao, Thomas M. Antonsen, Edward Ott, and Steven M. Anlage. Focusing waves at arbitrary locations in a ray-chaotic enclosure using time-reversed synthetic sonas. *Physical Review E*, 93(5):052205, May 2016.

[9] I. M. Vellekoop and A. P. Mosk. Focusing coherent light through opaque strongly scattering media. *Optics Letters*, 32(16):2309–2311, August 2007.

[10] A.P. Mosk, A. Lagendijk, G. Lerosey, and M. Fink. Controlling waves in space and time for imaging and focusing in complex media. *Nature Photonics*, 6:283–292, 2012.

[11] Martin Booth, Débora Andrade, Daniel Burke, Brian Patton, and Mantas Zurauskas. Aberrations and adaptive optics in super-resolution microscopy. *Microscopy*, 64(4):251–261, August 2015.

[12] Maddalena Collini, Fabrizio Radaelli, Laura Sironi, Nicolo G. Ceffa, Laura D'Alfonso, Margaux Bouzin, and Giuseppe Chirico. Adaptive optics microspectrometer for cross-correlation measurement of microfluidic flows. *Journal of Biomedical Optics*, 24(02):1, February 2019.

[13] A. Epstein, J. P. S. Wong, and G. V. Eleftheriades. Design and applications of Huygens metasurfaces. In *2015 9th International Congress on Advanced Electromagnetic Materials in Microwaves and Optics (METAMATERIALS)*, pages 67–69, September 2015.

[14] Michael Chen, Minseok Kim, Alex M.H. Wong, and George V. Eleftheriades. Huygens' metasurfaces from microwaves to optics: a review. *Nanophotonics*, 7(6):1207–1231, June 2018.

[15] Qiong He, Shulin Sun, and Lei Zhou. Tunable/Reconfigurable Metasurfaces: Physics and Applications. *Research*, 2019:1–16, July 2019.

[16] Chuanlin Li, Peng Yu, Yongjun Huang, Qiang Zhou, Jiang Wu, Zhe Li, Xin Tong, Qiye Wen, Hao-Chung Kuo, and Zhiming M. Wang. Dielectric metasurfaces: From wavefront shaping to quantum platforms. *Progress in Surface Science*, 95(2):100584, May 2020.

[17] Matthieu Dupré, Philipp del Hougne, Mathias Fink, Fabrice Lemoult, and Geoffroy Lerosey. Wave-Field Shaping in Cavities: Waves Trapped in a Box with Controllable Boundaries. *Physical Review Letters*, 115(1):017701, July 2015.

[18] Nadège Kaina, Matthieu Dupré, Geoffroy Lerosey, and Mathias Fink. Shaping complex microwave fields in reverberating media with binary tunable metasurfaces. *Scientific Reports*, 4(1):6693, May 2015.

[19] Philipp del Hougne, Fabrice Lemoult, Mathias Fink, and Geoffroy Lerosey. Spatiotemporal Wave Front Shaping in a Microwave Cavity. *Physical Review Letters*, 117(13):134302, September 2016.

[20] Philipp del Hougne, Matthieu Davy, and Ulrich Kuhl. Optimal Multiplexing of Spatially Encoded Information across Custom-Tailored Configurations of a Metasurface-Tunable Chaotic Cavity. *Physical Review Applied*, 13(4):041004, April 2020.

[21] Jean-Baptiste Gros, Philipp del Hougne, and Geoffroy Lerosey. Tuning a regular cavity to wave chaos with metasurface-reconfigurable walls. *Physical Review A*, 101(6):061801, June 2020.

[22] Mohammadreza F. Imani, David R. Smith, and Philipp del Hougne. Perfect absorption in a disordered medium with programmable meta-atom inclusions. *Advanced Functional Materials*, 30(52):2005310, 2020.

[23] Philipp del Hougne, Dmitry V. Savin, Olivier Legrand, and Ulrich Kuhl. Implementing nonuniversal features with a random matrix theory approach: Application to space-to-configuration multiplexing. *Phys. Rev. E*, 102:010201, Jul 2020.

[24] Benjamin W. Frazier, Thomas M. Antonsen, Steven M. Anlage, and Edward Ott. Wavefront shaping with a tunable metasurface: Creating cold spots and coherent perfect absorption at arbitrary frequencies. *Phys. Rev. Research*, 2:043422, Dec 2020.

[25] Philipp del Hougne, K. Brahima Yeo, Philippe Besnier, and Matthieu Davy. On-demand coherent perfect absorption in complex scattering systems: Time delay divergence and enhanced sensitivity to perturbations. *Laser & Photonics Reviews*, 15(7):2000471, 2021.

[26] Y. D. Chong, Li Ge, Hui Cao, and A. D. Stone. Coherent Perfect Absorbers: Time-Reversed Lasers. *Physical Review Letters*, 105(5):053901, July 2010.

[27] Kevin Pichler, Matthias Kühmayer, Julian Böhm, Andre Brandstötter, Philipp Ambichl, Ulrich Kuhl, and Stefan Rotter. Random anti-lasing through coherent perfect absorption in a disordered medium. *Nature*, 567(7748):351–355, March 2019.

[28] Lei Chen, Tsampikos Kottos, and Steven M. Anlage. Perfect absorption in complex scattering systems with or without hidden symmetries. *Nature Communications*, 11(1):5826, December 2020.

[29] Benjamin W. Frazier, Thomas M. Antonsen Jr., Steven M. Anlage, and Edward Ott. Deep Learning Control of Complex Reverberant Wave Fields by a Programmable Metasurface. *arXiv:2103.13500 [cond-mat, physics:physics]*, 2021.

[30] R. L. Schmid, D. B. Shrekenhamer, O. F. Somerlock, A. C. Malone, T. A. Sleasman, and R. S. Awadallah. S-band GaAs FET Reconfigurable Reflectarray for Passive Communications. In *2020 IEEE Radio and Wireless Symposium (RWS)*, pages 91–93, San Antonio, TX, USA, January 2020. IEEE.

[31] Yun Bo Li, Lian Lin Li, Bai Bing Xu, Wei Wu, Rui Yuan Wu, Xiang Wan, Qiang Cheng, and Tie Jun Cui. Transmission-Type 2-Bit Programmable Metasurface for Single-Sensor and Single-Frequency Microwave Imaging. *Scientific Reports*, 6(1):23731, July 2016.

[32] Linda Shao, Weiren Zhu, Mikhail Yu Leonov, and Ivan D. Rukhlenko. Dielectric 2-bit coding metasurface for electromagnetic wave manipulation. *Journal of Applied Physics*, 125(20):203101, May 2019.

[33] Shahid Iqbal, Shuo Liu, Guo Dong Bai, Muhammad Furqan, Hamza Ahmad Madni, and Tie Jun Cui. Dual-band 2-bit coding metasurface for multifunctional control of both spatial waves and surface waves. *Journal of the Optical Society of America B*, 36(2):293, February 2019.

[34] Jun Luo and Tie Jun Cui. 2-Bit Ultrathin Amplitude-Modulated Coding Metasurfaces with Inserted Chip Resistors. In *2019 IEEE International Conference on Computational Electromagnetics (ICCEM)*, pages 1–3, Shanghai, China, March 2019. IEEE.

[35] Shuhui Li and Jian Wang. Adaptive free-space optical communications through turbulence using self-healing Bessel beams. *Scientific Reports*, 7(1):43233, February 2017.

[36] Weiming Hao, Ming Deng, Shuqi Chen, and Lin Chen. High-Efficiency Generation of Airy Beams with Huygens' Metasurface. *Physical Review Applied*, 11(5):054012, May 2019.

[37] Jiaqi Yang, Heng Zhou, and Tian Lan. All-dielectric reflective metasurface for orbital angular momentum beam generation. *Optical Materials Express*, 9(9):3594, September 2019.

[38] Hiroki Kishikawa, Haruya Kishimoto, Noriyuki Sakashita, Nobuo Goto, and Shien-Kuei Liaw. Pilot beam-assisted adaptive compensation for atmospheric turbulence in free-space optical transmission of beams carrying orbital angular momentum. *Japanese Journal of Applied Physics*, 59(SO):SOOD03, August 2020.

[39] João Sabino, Gonçalo Figueira, and Paulo André. Wavefront spatial-phase modulation in visible optical communications. *Microwave and Optical Technology Letters*, 59(7):1538–1541, 2017.

[40] Wankai Tang, Jun Yan Dai, Mingzheng Chen, Xiang Li, Qiang Cheng, Shi Jin, Kai-Kit Wong, and Tie Jun Cui. Programmable metasurface-based RF chain-free 8PSK wireless transmitter. *Electronics Letters*, 55(7):417–420, March 2019.

[41] Jun Yan Dai, Wan Kai Tang, Jie Zhao, Xiang Li, Qiang Cheng, Jun Chen Ke, Ming Zheng Chen, Shi Jin, and Tie Jun Cui. Wireless Communications through a Simplified Architecture Based on Time-Domain Digital Coding Metasurface. *Advanced Materials Technologies*, 4(7):1900044, July 2019.

[42] Jie Zhao, Xi Yang, Jun Yan Dai, Qiang Cheng, Xiang Li, Ning Hua Qi, Jun Chen Ke, Guo Dong Bai, Shuo Liu, Shi Jin, Andrea Alù, and Tie Jun Cui. Programmable time-domain digital-coding metasurface for non-linear harmonic manipulation and new wireless communication systems. *National Science Review*, 6(2):231–238, March 2019.

[43] Hanting Zhao, Ya Shuang, Menglin Wei, Tie Jun Cui, Philipp del Hougne, and Lianlin Li. Metasurface-assisted massive backscatter wireless communication with commodity Wi-Fi signals. *Nature Communications*, 11(1):3926, August 2020.

[44] Mohammadreza F. Imani, Jonah N. Gollub, Okan Yurduseven, Aaron V. Diebold, Michael Boyarsky, Thomas Fromenteze, Laura Pulido-Mancera, Timothy Sleasman, and David R. Smith. Review of Metasurface Antennas for Computational Microwave Imaging. *IEEE Transactions on Antennas and Propagation*, 68(3):1860–1875, March 2020.

[45] Lianlin Li, Tie Jun Cui, Wei Ji, Shuo Liu, Jun Ding, Xiang Wan, Yun Bo Li, Menghua Jiang, Cheng-Wei Qiu, and Shuang Zhang. Electromagnetic reprogrammable coding-metasurface holograms. *Nature Communications*, 8(1):197, December 2017.

[46] Majid Karimipour, Nader Komjani, and Iman Aryanian. Shaping Electromagnetic Waves with Flexible and Continuous Control of the Beam Directions Using Holography and Convolution Theorem. *Scientific Reports*, 9(1):11825, December 2019.

[47] Zhuochao Wang, Jian Liu, Xumin Ding, Weisong Zhao, Kuang Zhang, Haoyu Li, Badreddine Ratni, Shah Nawaz Burokur, and Qun Wu. Three-Dimensional Microwave Holography Based on Broadband Huygens' Metasurface. *Physical Review Applied*, 13(1):014033, January 2020.

[48] Timothy Sleasman, Mohammadreza F. Imani, Michael Boyarsky, Laura Pulido-Mancera, Matthew S. Reynolds, and David R. Smith. Reconfigurable metasurface aperture for security screening and microwave imaging. In David A. Wikner and Duncan A. Robertson, editors, *Proceedings of the SPIE*, volume 10189, page 101890G, Anaheim, California, United States, May 2017.

[49] Michael Boyarsky, Timothy Sleasman, Laura Pulido-Mancera, Aaron V. Diebold, Mohammadreza F. Imani, and David R. Smith. Single-frequency 3D synthetic aperture imaging with dynamic metasurface antennas. *Applied Optics*, 57(15):4123, May 2018.

[50] Zhenhua Wu, Lei Zhang, and Hongwei Liu. Generalized Three-Dimensional Imaging Algorithms for Synthetic Aperture Radar With Metamaterial Apertures-Based Antenna. *IEEE Access*, 7:59716–59727, 2019.

[51] Timothy Sleasman, Mohammadreza F. Imani, Michael Boyarsky, Kenneth P. Trofatter, and David R. Smith. Computational through-wall imaging using a dynamic metasurface antenna. *OSA Continuum*, 2(12):3499, December 2019.

[52] Timothy Sleasman, Mohammadreza F. Imani, Jonah N. Gollub, and David R. Smith. Microwave Imaging Using a Disordered Cavity with a Dynamically Tunable Impedance Surface. *Physical Review Applied*, 6(5):054019, November 2016.

[53] J. N. Gollub, O. Yurduseven, K. P. Trofatter, D. Arnitz, M. F. Imani, T. Sleasman, M. Boyarsky, A. Rose, A. Pedross-Engel, H. Odabasi, T. Zvolensky, G. Lipworth, D. Brady, D. L. Marks, M. S. Reynolds, and D. R. Smith. Large Metasurface Aperture for Millimeter Wave Computational Imaging at the Human-Scale. *Scientific Reports*, 7(1):42650, May 2017.

[54] Fabrice Lemoult, Geoffroy Lerosey, Julien de Rosny, and Mathias Fink. Manipulating Spatiotemporal Degrees of Freedom of Waves in Random Media. *Physical Review Letters*, 103(17):173902, October 2009.

[55] Mooseok Jang, Yu Horie, Atsushi Shibukawa, Joshua Brake, Yan Liu, Seyedeh Mahsa Kamali, Amir Arbabi, Haowen Ruan, Andrei Faraon, and Changhuei Yang. Wavefront shaping with disorder-engineered metasurfaces. *Nature Photonics*, 12(2):84–90, February 2018.

[56] Hengyi Sun, Zhuo Li, Changqing Gu, Qian Xu, Xinlei Chen, Yunhe Sun, Shengchen Lu, and Ferran Martin. Metasurfaced Reverberation Chamber. *Scientific Reports*, 8(1):1577, December 2018.

[57] Hengyi Sun, Changqing Gu, Zhuo Li, Qian Xu, Mengmeng Wei, Jiajia Song, Baijie Xu, Xiaohang Dong, Kuan Wang, and Ferran Martín. Parametric Testing of Metasurface Stirrers for Metasurfaced Reverberation Chambers. *Sensors*, 19(4):976, February 2019.

[58] Jean-Baptiste Gros, Geoffroy Lerosey, Fabrice Mortessagne, Ulrich Kuhl, and Olivier Legrand. Uncorrelated Configurations and Field Uniformity in Reverberation Chambers Stirred by Tunable Metasurfaces. *arXiv:1905.12757 [nlin, physics:physics]*, May 2019.

[59] Jichao Fu, Yi Jin, and Sailing He. Metasurface for Constructing a Stable High-Q Plano-Planar Open Cavity. *Advanced Optical Materials*, 7(5):1801339, March 2019.

[60] Edris Ameri, Seyed Hassan Esmaeli, and Seyed Hassan Sedighy. Ultra Wideband Radar Cross Section Reduction by Using Polarization Conversion Metasurfaces. *Scientific Reports*, 9(1):478, December 2019.

[61] He Wang, Jiyao Huang, Honglin Wang, Yongfeng Li, Sai Sui, Wei Li, Changxing Chen, Jieqiu Zhang, and Shaobo Qu. Chaos-based coding metasurface for radar cross-section reduction. *Journal of Physics D: Applied Physics*, 52(40):405304, October 2019.

[62] Imen Soltani, Takoua Soltani, and Taoufik Aguili. Low RCS Multi-Bit Coding Metasurface Modeling and Optimization: MoM-GEC Method in Conjunction with Genetic Algorithm. *Progress In Electromagnetics Research M*, 84:10, 2019.

[63] Chao Qian, Bin Zheng, Yichen Shen, Li Jing, Erping Li, Lian Shen, and Hongsheng Chen. Deep-learning-enabled self-adaptive microwave cloak without human intervention. *Nature Photonics*, 14(6):383–390, June 2020.

[64] Antonio Palermo and Alessandro Marzani. Control of Love waves by resonant metasurfaces. *Scientific Reports*, 8:7234, 2018.

[65] Yanbin He, Tianning Chen, and Xinpei Song. Manipulation of seismic rayleigh waves using a phase-gradient rubber metasurface. *International Journal of Modern Physics B*, 34(13):2050142, 2020.

[66] R. Bekenstein, I. Pikovski, H. Pichler, E. Shahmoon, S. F. Yelin, and M. D. Lukin. Quantum metasurfaces with atom arrays. *Nature Physics*, 16:676–681, December 2020.

[67] Horace W. Babcock. The possibility of compensating astronomical seeing. *Publications of the Astronomical Society of the Pacific*, 65(386):229–235, 1953.

[68] Robert A. Duffner. *The Adaptive Optics Revolution, A History*. University of New Mexico Press, Albuquerque, NM, 2009.

[69] John W. Hardy. *Adaptive Optics for Astronomical Telescopes*. Oxford University Press, New York, NY, 1998.

[70] Martin J Booth. Adaptive optics in microscopy. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 365(1861):2829–2843, 2007.

[71] Charles B. Hogge. Optical Distortion In High-Energy Laser Systems: Active Control. In Enrique Bernal G. and Harry V. Winsor, editors, *Optics in Adverse Environments I*, volume 0121, pages 140 – 145. International Society for Optics and Photonics, SPIE, 1978.

[72] Henry J. White, David W. Gough, Richard Merry, and Stephen Patrick. Demonstration of free-space optical communication link incorporating a closed-loop tracking system for mobile platforms. In Monte Ross and Andrew M. Scott, editors, *Advanced Free-Space Optical Communications Techniques and Technologies*, volume 5614, pages 119 – 128. International Society for Optics and Photonics, SPIE, 2004.

[73] Mark T. Gruneisen, Mark L. Eickhoff, Scott C. Newey, Kurt E. Stoltenberg, Jeffery F. Morris, Michael Bareian, Mark A. Harris, Denis W. Oesch, Michael D. Oliker, Michael B. Flanagan, Brian T. Kay, Johnathan D. Schiller, and R. Nicholas Lanning. Adaptive-optics-enabled quantum communication: A technique for daytime space-to-earth links. *Phys. Rev. Applied*, 16:014067, Jul 2021.

[74] Patrick S. Salter and Martin J. Booth. Adaptive optics in laser processing. *Light: Science and Applications*, 8:110, 2019.

[75] R. H. Freeman and J. E. Pearson. Deformable mirrors for all seasons and reasons. *Appl. Opt.*, 21(4):580–588, Feb 1982.

[76] Mark A. Ealey and John Wellman. Fundamentals Of Deformable Mirror Design And Analysis. In Daniel Vukobratovich, editor, *Precision Engineering and Optomechanics*, volume 1167, pages 66 – 84. International Society for Optics and Photonics, SPIE, 1989.

[77] Mark A. Ealey. Actuators: design fundamentals, key performance specifications, and parametric trades. In Mark A. Ealey, editor, *Active and Adaptive Optical Components*, volume 1543, pages 346 – 362. International Society for Optics and Photonics, SPIE, 1992.

[78] Allan Wirth, Jeffrey Cavaco, Theresa Bruno, and Kevin M. Ezzo. Deformable mirror technologies at AOA Xinetics. In Georg Korn, Luis Oliveira Silva, and Joachim Hein, editors, *High-Power, High-Energy, and High-Intensity Laser Technology; and Research Using Extreme Light: Entering New Frontiers with Petawatt-Class Lasers*, volume 8780, pages 122 – 133. International Society for Optics and Photonics, SPIE, 2013.

[79] M. Horenstein, T. Bifano, S. Pappas, J. Perreault, and R. Krishnamoorthy-Mali. Real time optical correction using electrostatically actuated mems devices. *Journal of Electrostatics*, 46(2):91–101, 1999. Selected Papers from the 1998 Joint ESA/IEJ Symposium on Electrostatics.

[80] Julie A. Perreault, Thomas G. Bifano, Bruce Martin Levine, and Mark N. Horenstein. Adaptive optic correction using micro-electro-mechanical deformable mirrors. *Optical Engineering*, 41(3):561 – 566, 2002.

[81] D. Casasent. Spatial light modulators. *Proceedings of the IEEE*, 65(1):143–157, 1977.

[82] G. Esposito, S. amd Brusa and D. Bonaccini. Liquid crystal wavefront correctors: Computer simulation results. In *Proceedings of the ICO-16 (International Commission for Optics) Satellite Conference on Active and adaptive optics*, pages 289–294. ESO, 1993.

[83] I. M. Vellekoop and A. P. Mosk. Universal optimal transmission of light through disordered materials. *Phys. Rev. Lett.*, 101:120601, Sep 2008.

[84] Dekel Veksler, Elhanan Maguid, Nir Shitrit, Dror Ozeri, Vladimir Kleiner, and Erez Hasman. Multiple Wavefront Shaping by Metasurface Based on Mixed Random Antenna Groups. *ACS Photonics*, 2(5):661–667, May 2015.

[85] Philipp del Hougne, Mathias Fink, and Geoffroy Lerosey. Shaping Microwave Fields using Non-Linear Unsolicited Feedback: Application to Enhanced Energy Harvesting. *Physical Review Applied*, 8(6):061001, December 2017.

[86] Lennart Ljung. *System Identification: Theory for the User*. Pearson, Upper Saddle River, NJ, 2nd edition, 1997.

[87] Robert K. Tyson and Benjamin W. Frazier. *Field Guide to Adaptive Optics, 2nd Edition*. Field Guide Series. SPIE Press, Bellingham, WA, 2012.

[88] Andreas Neubauer. On the ill-posedness and convergence of the shack–hartmann based wavefront reconstruction. *Journal of Inverse and Ill-Posed Problems*, 18(5):551–576, 2010.

[89] Curtis R. Vogel. *Computational Methods for Inverse Problems*. SIAM, Philadelphia, 2006.

[90] Per Christian Hansen, James G. Nagy, and Dianne P. O'Leary. *Deblurring Images: Matrices, Spectra, and Filtering*. SIAM, Philadelphia, 2006.

[91] Victoria Hutterer and Ronny Ramlau. Nonlinear wavefront reconstruction methods for pyramid sensors using landweber and landweber-kaczmarz iterations. *Appl. Opt.*, 57(30):8790–8804, Oct 2018.

[92] Victoria Hutterer, Ronny Ramlau, and Iuliia Shatokhina. Real-time adaptive optics with pyramid wavefront sensors: part II. accurate wavefront reconstruction using iterative methods. *Inverse Problems*, 35(4):045008, mar 2019.

[93] Sameer Hemmady, Xing Zheng, Thomas M. Antonsen, Edward Ott, and Steven M. Anlage. Universal statistics of the scattering coefficient of chaotic microwave cavities. *Physical Review E*, 71(5):056215, May 2005.

[94] Sameer Hemmady, Xing Zheng, James Hart, Thomas M. Antonsen, Edward Ott, and Steven M. Anlage. Universal properties of two-port scattering, impedance, and admittance matrices of wave-chaotic systems. *Physical Review E*, 74(3):036213, September 2006.

[95] J. A. Hart, T. M. Antonsen, and E. Ott. Effect of short ray trajectories on the scattering statistics of wave chaotic systems. *Physical Review E*, 80(4):041109, October 2009.

[96] I.M. Vellekoop and A.P. Mosk. Phase control algorithms for focusing light through turbid media. *Optics Communications*, 281(11):3071 – 3080, 2008.

[97] C. Dorrer and J. Qiao. Direct binary search for improved coherent beam shaping and optical differentiation wavefront sensing. *Applied Optics*, 57(29):8557–8565, October 2018.

[98] T. R. O'Meara. The multidither principle in adaptive optics. *J. Opt. Soc. Am.*, 67(3):306–315, Mar 1977.

[99] James C. Spall. *Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control*. Wiley, Hoboken, NJ, 2003.

[100] Mikhail A. Vorontsov and Valeriy Kolosov. Target-in-the-loop beam control: basic considerations for analysis and wave-front sensing. *J. Opt. Soc. Am. A*, 22(1):126–141, Jan 2005.

[101] M. A. Vorontsov, G. W. Carhart, and J. C. Ricklin. Adaptive phase-distortion correction based on parallel gradient-descent optimization. *Opt. Lett.*, 22(12):907–909, Jun 1997.

[102] M. A. Vorontsov and V. P. Sivokon. Stochastic parallel-gradient-descent technique for high-resolution wave-front phase-distortion correction. *J. Opt. Soc. Am. A*, 15(10):2745–2758, Oct 1998.

[103] Thomas Weyrauch and Mikhail A. Vorontsov. Free-space laser communications with adaptive optics: Atmospheric compensation experiments. *Journal of Optical and Fiber Communications Reports*, 1:355–379, 2004.

[104] V. Rodoplu and T. H. Meng. Bits-per-joule capacity of energy-limited wireless networks. *IEEE Transactions on Wireless Communications*, 6(3):857–865, 2007.

[105] C. Huang, A. Zappone, G. C. Alexandropoulos, M. Debbah, and C. Yuen. Reconfigurable intelligent surfaces for energy efficiency in wireless communication. *IEEE Transactions on Wireless Communications*, 18(8):4157–4170, 2019.

[106] Dimitri P Bertsekas. *Reinforcement learning and optimal control*. Athena Scientific, 2019.

[107] Yang Deng, Simiao Ren, Kebin Fan, Jordan M. Malof, and Willie J. Padilla. Neural-adjoint method for the inverse design of all-dielectric metasurfaces. *Opt. Express*, 29(5):7526–7534, Mar 2021.

[108] Fritz Haake. *Quantum Signatures of Chaos*. Springer, Berlin, Heidelberg, 2010.

[109] Sameer Hemmady, Xing Zheng, Edward Ott, Thomas M. Antonsen, and Steven M. Anlage. Universal Impedance Fluctuations in Wave Chaotic Systems. *Physical Review Letters*, 94(1):014102, January 2005.

[110] Jen-Hao Yeh, James A. Hart, Elliott Bradshaw, Thomas M. Antonsen, Edward Ott, and Steven M. Anlage. Universal and nonuniversal properties of wave-chaotic scattering systems. *Physical Review E*, 81(2):025201, February 2010.

[111] Sameer Hemmady, Thomas M. Antonsen, Edward Ott, and Steven M. Anlage. Statistical Prediction and Measurement of Induced Voltages on Components Within Complicated Enclosures: A Wave-Chaotic Approach. *IEEE Transactions on Electromagnetic Compatibility*, 54(4):758–771, August 2012.

[112] M V Berry. Regular and irregular semiclassical wavefunctions. *Journal of Physics A: Mathematical and General*, 10:2083–2091, 1977.

[113] Edward Ott. *Chaos in dynamical systems*. Cambridge University Press, Cambridge, U.K. ; New York, 2nd ed edition, 2002.

[114] Xing Zheng, Thomas M. Antonsen, and Edward Ott. Statistics of Impedance and Scattering Matrices in Chaotic Microwave Cavities: Single Channel Case. *Electromagnetics*, 26(1):3–35, January 2006.

[115] Xing Zheng, Thomas M. Antonsen, and Edward Ott. Statistics of Impedance and Scattering Matrices of Chaotic Microwave Cavities with Multiple Ports. *Electromagnetics*, 26(1):37–55, January 2006.

[116] Sameer D Hemmady. *A Wave-Chaotic Approach to Predicting and Measuring Electromagnetic Field Quantities in Complicated Enclosures*. PhD thesis, University of Maryland, 2006.

[117] Sameer Hemmady, James Hart, Xing Zheng, Thomas M. Antonsen, Edward Ott, and Steven M. Anlage. Experimental test of universal conductance fluctuations by means of wave-chaotic microwave cavities. *Physical Review B*, 74(19):195326, November 2006.

[118] Jen-Hao Yeh, James A. Hart, Elliott Bradshaw, Thomas M. Antonsen, Edward Ott, and Steven M. Anlage. Experimental examination of the effect of short ray trajectories in two-port wave-chaotic scattering systems. *Physical Review E*, 82(4):041114, October 2010.

[119] Bisrat Addissie, John Rodgers, and Thomas Antonsen. Extraction of the coupling impedance in overmoded cavities. *Wave Motion*, 87:123–131, April 2019.

[120] Jesus Gil Gil, Zachary B. Drikas, Tim D. Andreadis, and Steven M. Anlage. Prediction of Induced Voltages on Ports in Complex, Three-Dimensional Enclosures With Apertures, Using the Random Coupling Model. *IEEE Transactions on Electromagnetic Compatibility*, 58(5):1535–1540, October 2016.

[121] David M. Pozar. *MicrowaveEngineering, 4th Edition*. Wiley, Hoboken, NJ, 2011.

[122] J.H. Yeh, E. Ott, T.M. Antonsen, and S.M. Anlage. Fading Statistics in Communications - a Random Matrix Approach. *Acta Physica Polonica A*, 120(6A):A–85–A–88, December 2011.

[123] Jen-Hao Yeh, Thomas M. Antonsen, Edward Ott, and Steven M. Anlage. First-principles model of time-dependent variations in transmission through a fluctuating scattering environment. *Physical Review E*, 85(1):015202, January 2012.

[124] M. Simon and M.-S. Alouini. *Digital Communication Over Fading Channels, 2nd Edition*. Wiley-Interscience, Hoboken, NJ, 2005.

[125] Keith Ward, Robert Tough, and Simon Watts. *Sea Clutter, Scattering, the K Distribution and Radar Performance, 2nd Edition*. Insitution of Engineering and Technology, London, United Kingdom, 2013.

[126] R. Schäfer, H.-J. Stöckmann, T. Gorin, and T. H. Seligman. Experimental verification of fidelity decay: From perturbative to fermi golden rule regime. *Phys. Rev. Lett.*, 95:184102, Oct 2005.

[127] Biniyam Tesfaye Taddese, James Hart, Thomas M. Antonsen, Edward Ott, and Steven M. Anlage. Sensor based on extending the concept of fidelity to classical waves. *Applied Physics Letters*, 95(11):114103, September 2009.

[128] Biniyam Tesfaye Taddese, Thomas M. Antonsen, Edward Ott, and Steven M. Anlage. Sensing small changes in a wave chaotic scattering system. *Journal of Applied Physics*, 108:114911, 2010.

[129] Lei Zhang, Xiao Qing Chen, Shuo Liu, Qian Zhang, Jie Zhao, Jun Yan Dai, Guo Dong Bai, Xiang Wan, Qiang Cheng, Giuseppe Castaldi, Vincenzo Galdi, and Tie Jun Cui. Space-time-coding digital metasurfaces. *Nature Communications*, 9:4334, 2018.

[130] Christopher L. Holloway, Haider A. Shah, Ryan J. Pirkl, William F. Young, David A. Hill, and John Ladbury. Reverberation Chamber Techniques for Determining the Radiation and Total Efficiency of Antennas. *IEEE Transactions on Antennas and Propagation*, 60(4):1758–1770, April 2012.

[131] Tsampikos Kottos and Uzy Smilansky. Quantum Chaos on Graphs. *Physical Review Letters*, 79(24):4794–4797, December 1997.

[132] Tsampikos Kottos and Uzy Smilansky. Chaotic Scattering on Graphs. *Physical Review Letters*, 85(5):968–971, July 2000.

[133] Yan V Fyodorov, Suwun Suwunnarat, and Tsampikos Kottos. Distribution of zeros of the S -matrix of chaotic cavities with localized losses and coherent perfect absorption: non-perturbative results. *Journal of Physics A: Mathematical and Theoretical*, 50(30):30LT01, July 2017.

[134] Huanan Li, Suwun Suwunnarat, Ragnar Fleischmann, Holger Schanz, and Tsampikos Kottos. Random Matrix Theory Approach to Chaotic Coherent Perfect Absorbers. *Physical Review Letters*, 118(4):044101, January 2017.

[135] Lei Chen, Tsampikos Kottos, and Steven M. Anlage. Perfect Absorption in Complex Scattering Systems with or without Hidden Symmetries. *arXiv:2001.00956 [cond-mat, physics:nlin, physics:physics]*, March 2020.

[136] Philipp Ambichl, Andre Brandstötter, Julian Böhm, Matthias Kühmayer, Ulrich Kuhl, and Stefan Rotter. Focusing inside disordered media with the generalized wigner-smith operator. *Phys. Rev. Lett.*, 119:033903, Jul 2017.

[137] V. S. Asadchy, I. A. Faniayeu, Y. Ra'di, S. A. Khakhomov, I. V. Semchenko, and S. A. Tretyakov. Broadband Reflectionless Metasheets: Frequency-Selective Transmission and Perfect Absorption. *Physical Review X*, 5(3):031005, July 2015.

[138] Yoshua Bengio. Deep Learning of Representations: Looking Forward. *arXiv:1305.0445 [cs]*, June 2013.

[139] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, May 2015.

[140] Yoshua Bengio. Deep Learning of Representations for Unsupervised and Transfer Learning. *Proceedings of Machine Learning Research*, 27:17–36, 2012.

[141] Maithra Raghu, Ben Poole, Jon Kleinberg, Surya Ganguli, and Jascha Sohl-Dickstein. On the expressive power of deep neural networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2847–2854, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

[142] Guido F Montufar, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio. On the number of linear regions of deep neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27, pages 2924–2932. Curran Associates, Inc., 2014.

[143] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard,

Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mane, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viegas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *arXiv:1603.04467 [cs]*, March 2016.

[144] F. and others Chollet. Keras. `https://github.com/fchollet/keras`, 2015.

[145] David H Wolpert and William G Macready. No Free Lunch Theorems for Optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):16, 1997.

[146] Ramesh Kestur, Shariq Farooq, Rameen Abdal, Emad Mehraj, Omkar Subbaramajois Narasipura, and Meenavathi Mudigere. UFCN: a fully convolutional neural network for road extraction in RGB imagery acquired by remote sensing from an unmanned aerial vehicle. *Journal of Applied Remote Sensing*, 12(1):1 – 15, 2018.

[147] André Araujo, Wade Norris, and Jack Sim. Computing receptive fields of convolutional neural networks. *Distill*, 2019.

[148] Ayoosh Kathuria. Intro to optimization in deep learning: Momentum, rmsprop and adam. https://blog.paperspace.com/intro-to-optimization-momentum-rmsprop-adam/.

[149] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034, 2015.

[150] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *Proceedings of the 32nd International Conference on Machine Learning, PMLR*, 37:448–456, 2015.

[151] Gabriele Gradoni, Jen-Hao Yeh, Bo Xiao, Thomas M. Antonsen, Steven M. Anlage, and Edward Ott. Predicting the statistics of wave transport through chaotic cavities by the random coupling model: A review and recent progress. *Wave Motion*, 51(4):606–621, June 2014.

[152] S. M. Popoff, G. Lerosey, R. Carminati, M. Fink, A. C. Boccara, and S. Gigan. Measuring the Transmission Matrix in Optics: An Approach to the Study and Control of Light Propagation in Disordered Media. *Physical Review Letters*, 104(10):100601, March 2010.

[153] Angélique Drémeau, Antoine Liutkus, David Martina, Ori Katz, Christophe Schülke, Florent Krzakala, Sylvain Gigan, and Laurent Daudet. Reference-less measurement of the transmission matrix of a highly scattering material using a DMD and phase retrieval techniques. *Optics Express*, 23(9):11898, May 2015.

[154] Philipp del Hougne, Boshra Rajaei, Laurent Daudet, and Geoffroy Lerosey. Intensity-only measurement of partially uncontrollable transmission matrix: demonstration with wave-field shaping in a microwave cavity. *Optics Express*, 24(16):18631, August 2016.

[155] Claire M. Watts, David Shrekenhamer, John Montoya, Guy Lipworth, John Hunt, Timothy Sleasman, Sanjay Krishna, David R. Smith, and Willie J. Padilla. Terahertz compressive imaging with metamaterial spatial light modulators. *Nature Photonics*, 8(8):605–609, August 2014.

[156] Claire M. Watts, Christian C. Nadell, John Montoya, Sanjay Krishna, and Willie J. Padilla. Frequency-division-multiplexed single-pixel imaging with metamaterials. *Optica*, 3(2):133–138, February 2016.

[157] Michael Ossmann. HackRF One Software Defined Radio. `https://greatscottgadgets.com/hackrf/`, 2016.

[158] Nuand. bladeRF. `https://www.nuand.com`.

[159] Alex Krasnok, Denis G. Baranov, Andrey Generalov, Sergey Li, and Andrea Alù. Coherently Enhanced Wireless Power Transfer. *Physical Review Letters*, 120(14):143901, April 2018.

[160] Jen-Hao Yeh. *Wave Chaotic Experiments and Models for Complicated Wave Scattering Systems*. PhD thesis, University of Maryland, 2013.

[161] S. Ma, B. Xiao, R. Hong, B.D. Addissie, Z.B. Drikas, T.M. Antonsen, E. Ott, and S.M. Anlage. Classification and Prediction of Wave Chaotic Systems with Machine Learning Techniques. *Acta Physica Polonica A*, 136(5):757–764, November 2019.

[162] John Peurifoy, Yichen Shen, Li Jing, Yi Yang, Fidel Cano-Renteria, Brendan G. DeLacy, John D. Joannopoulos, Max Tegmark, and Marin Soljačić. Nanophotonic particle simulation and inverse design using artificial neural networks. *Science Advances*, 4(6), 2018.

[163] Dianjing Liu, Yixuan Tan, Erfan Khoram, and Zongfu Yu. Training Deep Neural Networks for the Inverse Design of Nanophotonic Structures. *ACS Photonics*, 5(4):1365–1369, April 2018.

[164] Christian C. Nadell, Bohao Huang, Jordan M. Malof, and Willie J. Padilla. Deep learning for accelerated all-dielectric metasurface design. *Optics Express*, 27(20):27523, September 2019.

[165] Sensong An, Clayton Fowler, Bowen Zheng, Mikhail Y. Shalaginov, Hong Tang, Hang Li, Li Zhou, Jun Ding, Anuradha Murthy Agarwal, Clara Rivero-Baleine, Kathleen A. Richardson, Tian Gu, Juejun Hu, and Hualiang Zhang. A Deep Learning Approach for Objective-Driven All-Dielectric Metasurface Design. *ACS Photonics*, 6(12):3196–3207, December 2019.

[166] Tianshuo Qiu, Xin Shi, Jiafu Wang, Yongfeng Li, Shaobo Qu, Qiang Cheng, Tiejun Cui, and Sai Sui. Deep Learning: A Rapid and Efficient Route to Automatic Metasurface Design. *Advanced Science*, 6(12):1900128, June 2019.

[167] Iman Sajedian, Heon Lee, and Junsuk Rho. Double-deep Q-learning to increase the efficiency of metasurface holograms. *Scientific Reports*, 9(1):10899, December 2019.

[168] Christopher Yeung, Ju-Ming Tsai, Brian King, Yusaku Kawagoe, David Ho, Mark W. Knight, and Aaswath P. Raman. Elucidating the Behavior of Nanophotonic Structures through Explainable Machine Learning Algorithms. *ACS Photonics*, 7(8):2309–2318, August 2020.

[169] Peter R. Wiecha and Otto L. Muskens. Deep Learning Meets Nanophotonics: A Generalized Accurate Predictor for Near Fields and Far Fields of Arbitrary 3D Nanostructures. *Nano Letters*, 20(1):329–338, January 2020.

[170] Jiaqi Jiang, Mingkun Chen, and Jonathan A. Fan. Deep neural networks for the evaluation and design of photonic devices. *Nature Reviews Materials*, 2020.

[171] Rohit Unni, Kan Yao, and Yuebing Zheng. Deep Convolutional Mixture Density Network for Inverse Design of Layered Photonic Structures. *ACS Photonics*, 7(10):2703–2712, October 2020.

[172] Abhishek Mall, Abhijeet Patil, Amit Sethi, and Anshuman Kumar. A cyclical deep learning based framework for simultaneous inverse and forward design of nanophotonic metasurfaces. *Scientific Reports*, 10(1):19427, December 2020.

[173] Sensong An, Bowen Zheng, Mikhail Y. Shalaginov, Hong Tang, Hang Li, Li Zhou, Jun Ding, Anuradha Murthy Agarwal, Clara Rivero-Baleine, Myungkoo Kang, Kathleen A. Richardson, Tian Gu, Juejun Hu, Clayton Fowler, and Hualiang Zhang. Deep learning modeling approach for metasurfaces with high degrees of freedom. *Optics Express*, 28(21):31932, October 2020.

[174] Lianlin Li, Ya Shuang, Qian Ma, Haoyang Li, Hanting Zhao, Menglin Wei, Che Liu, Chenglong Hao, Cheng-Wei Qiu, and Tie Jun Cui. Intelligent metasurface imager and recognizer. *Light: Science & Applications*, 8(1):97, December 2019.

[175] Lianlin Li, Hengxin Ruan, Che Liu, Ying Li, Ya Shuang, Andrea Alù, Cheng-Wei Qiu, and Tie Jun Cui. Machine-learning reprogrammable metasurface imager. *Nature Communications*, 10(1):1082, December 2019.

[176] Philipp del Hougne, Mohammadreza F. Imani, Aaron V. Diebold, Roarke Horstmeyer, and David R. Smith. Learned Integrated Sensing Pipeline: Reconfigurable Metasurface Transceivers as Trainable Physical Layer in an Artificial Neural Network. *Advanced Science*, 7(3):1901913, February 2020.

[177] Hao-Yang Li, Han-Ting Zhao, Meng-Lin Wei, Heng-Xin Ruan, Ya Shuang, Tie Jun Cui, Philipp Del Hougne, and Lianlin Li. Intelligent electromagnetic sensing with learnable data acquisition and processing. *Patterns*, 1(1):100006, 2020.

[178] Philipp del Hougne. Robust position sensing with wave fingerprints in dynamic complex propagation environments. *Phys. Rev. Research*, 2:043224, Nov 2020.

[179] Michael del Hougne, Sylvain Gigan, and Philipp del Hougne. Deeply Sub-Wavelength Localization with Reverberation-CodedAperture. *arXiv:2102.05642 [physics.app-ph]*, February 2021.

[180] Tao Shan, Xiaotian Pan, Maokun Li, Shenheng Xu, and Fan Yang. Coding Programmable Metasurfaces Based on Deep Learning Techniques. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 10(1):114–125, March 2020.

[181] R. Höhmann, U. Kuhl, H.-J. Stöckmann, L. Kaplan, and E. J. Heller. Freak waves in the linear regime: A microwave study. *Phys. Rev. Lett.*, 104:093901, Mar 2010.

[182] Chiheb Trabelsi, Olexa Bilaniuk, Ying Zhang, Dmitriy Serdyuk, Sandeep Subramanian, João Felipe Santos, Soroush Mehri, Negar Rostamzadeh, Yoshua Bengio, and Christopher J. Pal. Deep Complex Networks. *arXiv:1705.09792 [cs]*, February 2018.

[183] Oisín Moran, Piergiorgio Caramazza, Daniele Faccio, and Roderick Murray-Smith. Deep, complex, invertible networks for inversion of transmission effects in multimode optical fibres. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31, pages 3280–3291. Curran Associates, Inc., 2018.

[184] Piergiorgio Caramazza, Oisín Moran, Roderick Murray-Smith, and Daniele Faccio. Transmission of natural scene images through a multimode fibre. *Nature Communications*, 10(1):2029, December 2019.

[185] Sebastien Popoff. complexpytorch. `https://github.com/wavefrontshaping/complexPyTorch`, 2019.

[186] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, Boston, MA, USA, June 2015. IEEE.

[187] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision. In

2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 2818–2826, Las Vegas, NV, USA, June 2016. IEEE.

[188] Biniyam Tesfaye Taddese, Gabriele Gradoni, Franco Moglie, Thomas M Antonsen, Edward Ott, and Steven M Anlage. Quantifying volume changing perturbations in a wave chaotic system. *New Journal of Physics*, 15(2):023025, feb 2013.

[189] Christopher L. Holloway, Haider A. Shah, Ryan J. Pirkl, Kate A. Remley, David A. Hill, and John Ladbury. Early Time Behavior in Reverberation Chambers and Its Effect on the Relationships Between Coherence Bandwidth, Chamber Decay Time, RMS Delay Spread, and the Chamber Buildup Time. *IEEE Transactions on Electromagnetic Compatibility*, 54(4):714–725, August 2012.

[190] Christopher John Cornish Hellaby Watkins. *Learning from Delayed Rewards*. Ph.D., King's College, 1989.

[191] Marin Bukov, Alexandre G. R. Day, Dries Sels, Phillip Weinberg, Anatoli Polkovnikov, and Pankaj Mehta. Reinforcement Learning in Different Phases of Quantum Control. *Physical Review X*, 8(3):031086, September 2018.

[192] Jelena Mackeprang, Durga B. Rao Dasari, and Jörg Wrachtrup. A reinforcement learning approach for quantum state engineering. *Quantum Machine Intelligence*, 2(1):5, June 2020.

[193] Matteo M. Wauters, Emanuele Panizon, Glen B. Mbeng, and Giuseppe E. Santoro. Reinforcement-learning-assisted quantum optimization. *Physical Review Research*, 2(3):033446, September 2020.

[194] Hado van Hasselt, Arthur Guez, and David Silver. Deep Reinforcement Learning with Double Q-learning. *arXiv:1509.06461 [cs]*, December 2015.

[195] Matthew E Taylor and Peter Stone. Transfer Learning for Reinforcement Learning Domains: A Survey. *Journal of Machine Learning Research*, 10:1633–1685, 2009.

[196] Song Han, Huizi Mao, and William J. Dally. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. *arXiv:1510.00149 [cs]*, February 2016.

[197] NVIDIA. Jetson Embedded Systems. `https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/`.

[198] M Fink. Time-reversal waves and super resolution. *Journal of Physics: Conference Series*, 124:012004, jul 2008.

[199] James A. Hart, Thomas M. Antonsen, and Edward Ott. Scattering a pulse from a chaotic cavity: Transitioning from algebraic to exponential decay. *Physical Review E*, 79(1):016208, January 2009.

[200] Christopher L. Holloway, Haider A. Shah, Ryan Pirkl, William F. Young, David A. Hill, and John Ladbury. A three-antenna technique for determining the total and radiation efficiencies of antennas in reverberation chambers. *IEEE Antennas and Propagation Magazine*, 54(1):235–241, February 2012.

[201] David A. Hill. *Electromagnetic Fields in Cavities, Deterministic and Statistical Theories*. IEEE Press Series on Electromagnetic Wave Theory. Wiley, Hoboken, N.J, 2009.

[202] Joel Dunsmore. Gating effects in time domain transforms. In *2008 72nd ARFTG Microwave Measurement Symposium*, pages 1–8, Portland, OR, USA, December 2008. IEEE.

[203] Alan V. Oppenheim and Ronald W. Schafer. *Discrete-time signal processing*. Pearson, Upper Saddle River, 3rd ed edition, 2010.

[204] Jennifer Seberry, Beata J. Wysocki, and Tadeusz A. Wysocki. On some applications of Hadamard matrices. *Metrika*, 62(2-3):221–239, November 2005.

[205] Ihor Trots. Mutually Orthogonal Golay Complementary Sequences in Synthetic Aperture Imaging Systems. *Archives of Acoustics*, 40(2):283–289, June 2015.

[206] Noel A. C. Cressie and Christopher K. Wikle. *Statistics for spatio-temporal data*. Wiley series in probability and statistics. Wiley, Hoboken, N.J, 2011.

[207] Ardunio. UNO Getting Started Guide.

[208] StepperOnline. 17HS13-0404S1 Datasheet.

[209] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1139–1147, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR.

[210] Y. Nesterov. A method of solving a convex programming problem with convergence rate o(1/k2). *Soviet Mathematics Doklady*, 27(2):372–376, 1983.

[211] Leslie N. Smith. A disciplined approach to neural network hyper-parameters: Part 1 – learning rate, batch size, momentum, and weight decay. *arXiv:1803.09820 [cs, stat]*, April 2018.

[212] Chiheb Trabelsi. Deep complex networks. `https://github.com/ChihebTrabelsi/deep_complex_networks`, 2018.

[213] Patrick Virtue. Complex caffe. `https://github.com/pvirtue/caffe/tree/complex`, 2018.

[214] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[215] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034, 2015.

[216] Ming-Jer Lee. *Statistical Modeling of Wave Chaotic Transport and Tunneling*. Ph.D., University of Maryland, 2013.