Backscatter Gain and Array Modeling for a Large-Aperture High-Power Radar

A Major Qualifying Project
For the Silicon Valley Project Site
Submitted to the Faculty
Of the
WORCESTER POLYTECHNIC INSTITUTE
In Partial Fulfillment of the Requirements for the
Degree of Bachelor of Science

By:

Landon Airey

Patrick Sullivan

Jarred Velazquez

Presented to:

Professor John Orr, PhD

Michael Nicolls, PhD

SRI Geospace Division

Date: 3/6/2014

Abstract

A robust software package was developed for modeling the far field radiation pattern of phased array radars. The application of this package will assist SRI International's Geospace Division model the radiation pattern of their AMISR arrays. With this package, SRI will be able to calibrate their measurements of phenomena occurring in the upper atmosphere. Additionally, tools have been developed for beam analysis and radiation pattern characterization. Methods of validation for the computed radiation pattern's accuracy are included in the package.

Acknowledgements

The group would like to thank the following individuals for their continued guidance and support throughout the progression of this project:

PhD. Michael Nicolls, SRI International Geospace Division
Steven Chen, SRI International Geospace Division
Michael Greffen, SRI International Geospace Division

Professor John Orr, Worcester Polytechnic Institute
Professor David Finkel, Worcester Polytechnic Institute
Professor Sergey Makarov, Worcester Polytechnic Institute

Contents

Figures

Tables

Executive Summary

Research of the upper atmosphere has brought forth important information about space weather and its effects on modern technological systems. In order to collect data on the upper atmosphere, SRI International has developed the Advanced Modular Incoherent Scatter Radar (AMISR). SRI tasked senior WPI students to develop a software package to model the far-field radiation pattern of the AMISR system. The students have created a software package that added to the baseline model of the AMISR radiation pattern by incorporating authentic error factors and array status data, which has increased practicality of the model.

The radiation pattern of a phased array antenna (such as the AMISR) is dependent upon the radiation pattern of an individual element (assuming all antennas in the array are the same) and the geometry of the array (the locations of the antennas with respect to each other). The software package gives the user the ability to specify the array geometry and use data generated external simulators to compute the radiation pattern. Using this framework, the WPI students were able to model a phased array of all shapes and sizes.

The WPI team confirmed that Python would be an appropriate platform to model the radiation pattern of the AMISR by first comparing a two antenna array model from [16] with the model ported into Python. The students then created the model for a 4096 element array, which was compared to the baseline MATLAB model for 4096 elements. Once this accuracy was established, the team incorporated situational factors into the model including element gain, mutual coupling, phase error, and array status data. Using a satellite pass test, the model output was confirmed to be accurate. While the framework of the software package can be used for all AMISR systems, this characteristic was only confirmed using the RISR-N system due to time constraints.

The development of this software package enabled the Geospace Division at SRI to investigate the radiation pattern of phased array radars. Specifically, they can apply the package to model the radiation pattern of their AMISR. By including features which can pull data from externally generated sources, such as NEC modeling for element pattern, the package can be further refined with parameters of increasing accuracy. Additionally, the package offers the user the ability to pull real life status data about each antenna and incorporate and analyze this status's effect on the radiation pattern. To validate their model, a satellite pass test was created which compares computed data and measured data. By comparing the simulated data (white dots) and the measured data (red dots), the results show that the model is accurate (See Figure 1).

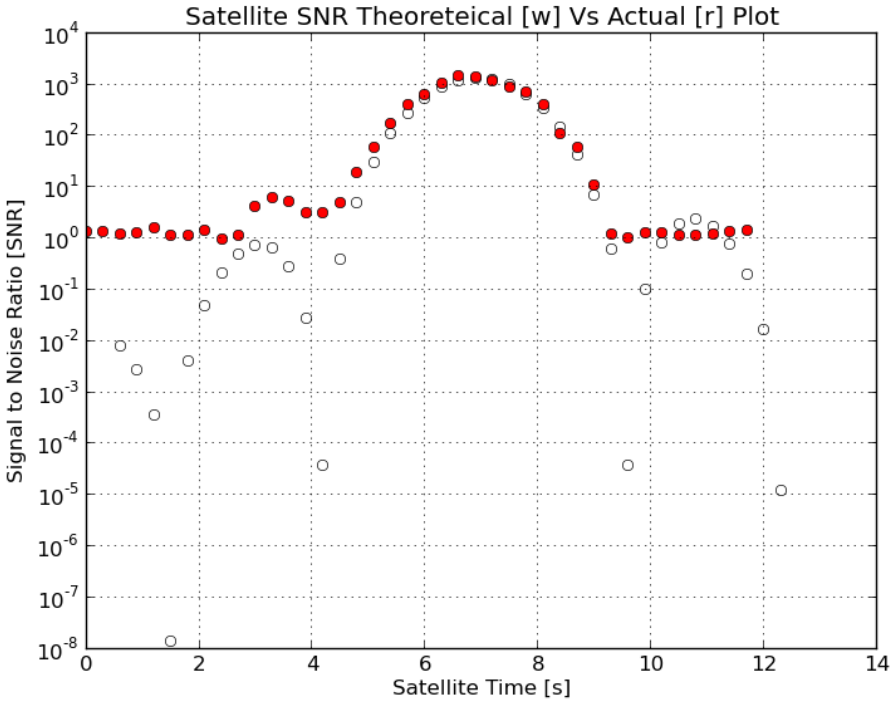

Figure 1-SNR Model Validation (Red dots- Recorded Data, White dots- Simulated Data)

Section 1: Introduction and Goals

This year's group of students from Worcester Polytechnic Institute chosen to work at SRI International assisted the Geospace Division's effort to analyze a radar system they have been developing. They will do so by creating a robust software package to simulate and analyze the far-field radiation pattern [7] of the Advanced Modular Incoherent Scatter Radar (AMISR). SRI International is a nonprofit research organization that works with private, government, and industrial organizations to promote science and discovery. It was established in 1946 by Stanford University, but split from the university in 1970, later becoming known as SRI International in 1977 [1]. SRI International has been using its various projects as opportunities to collaborate with academic engineering programs throughout the nation. Specifically, WPI has had involvement in the development of SRI's AMISR. The AMISR is an operational phased array radar system funded by the National Science Foundation capable of collecting data on space weather in the ionosphere. This information can help researchers develop a better understanding of weather and climatology of the space environment, and ultimately how space weather impacts the power grid, satellites, and communication systems [2].

A simple MATLAB model of the AMISR's radiation pattern exists. This model helped researchers predict the behavior of the AMISR and demonstrate the functionality of the system. Predicting the behavior of the radiation pattern with a model is a pivotal step in the process of understanding the data that is actually collected from the AMISR.

The main goal of this year's project was to develop SRI's model of the AMISR's beam by incorporating mutual coupling (the interaction between antennas) and the power measurements of the AMISR's AEUs. Doing so allows for researchers to draw significant findings from the data being collected by the AMISR. In order to accomplish this goal, the group needed to develop an understanding of radar, a topic incorporating many disciplines of study. The next step in accomplishing this goal was translating the preexisting MATLAB model into a Python environment. Accurately modeling the AMISR's radiation pattern requires an accurate model of the individual AEU's radiation pattern. Using modeling software, the group incorporated the effect that mutual coupling (cross-talk between antennas in an array) has on each antenna system. Doing this enhanced the accuracy of the AEU's radiation model and therefore the accuracy of the entire system's radiation pattern.

Section 2: Background Information

A functional understanding of waves in space, antennas, radar applications, phased array radars, the AMISR and modelling software is needed in order to develop and refine a model of the radiation pattern of a phased array radar. Section 2 introduces the concepts relevant to understanding these disciplines.

2.1: Waves in Space

Before delving into a scientific explanation of what a wave is, a simple analogy to oceanic waves can be useful to describe the concepts of amplitude, frequency, and wavelength. If one were to stand in the ocean, the amplitude of an oncoming wave is considered to be the maximum height that the wave reached on one's body as it flowed past to shore. The rate, with respect to time, at which the waves flowed passed the observer is the wave frequency. Additionally, if one were to take a picture of the waves, the distance between the peaks of two sequential waves would describe the wavelength. Now that the analogy has been set, the scientific explanations for these concepts will be introduced.

The following is the general equation used to define a time-continuous waveform in one-dimensional space:

$$y(x,t) = y_m \sin\left(\frac{2\pi}{\lambda}x - \frac{2\pi}{T}t + \phi_o\right) \qquad \text{[Eq. 1]}$$

For analysis, Equation 1 can be broken up into two main parts: amplitude (y_m), and oscillation ($\sin(\frac{2\pi x}{\lambda} - \frac{2\pi}{T}t + \Phi o)$). The amplitude of a waveform is "the magnitude of maximum displacement of elements from equilibrium position as a wave passes through a point" [3]. The part of the equation which describes the oscillation of a wave y(x,t) is characterized by both the sinusoid and its phase ($\frac{2\pi x}{\lambda} - \frac{2\pi}{T}t + \Phi o$). The phase can be characterized further into three components, the angular frequency ($2\pi f$), the angular wavenumber($\frac{2\pi}{\lambda}$), and the phase constant (ϕ_o). [3], [4].

The frequency (f) of a wave is the number of oscillations per unit of time. This value is equal to the inverse of the time period (T) that it takes for a wave to complete one cycle of oscillation. Angular frequency is equal to the frequency multiplied by 2π (the number of radians in a circle). Angular frequency has units of radians per second. Figure 2 below illustrates the concept of a time-continuous wave's frequency. [3], [4].

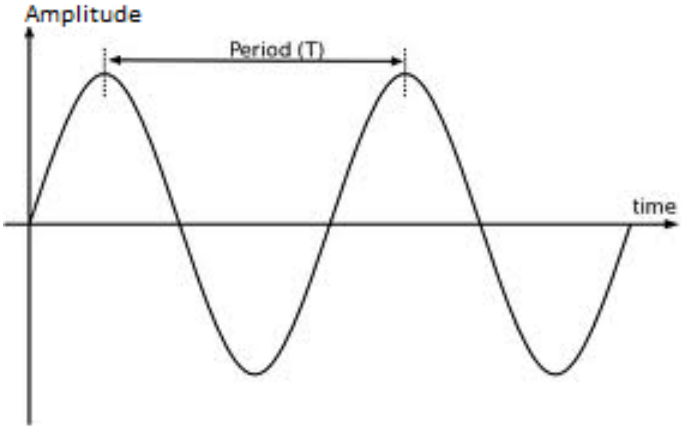

Figure 2- Demonstration of a Period of Wave Oscillation [s]

Angular wavenumber is analogous to angular frequency. Instead of looking at a waveform with respect to time, however, one can look at it with respect to space. To do so, one can look at the amount of space between distinct repetitions of a wave (such as peak values), rather than the amount of time between said repetitions. This length is classified as wavelength, represented by λ, and has units of meters (See Figure 3). Angular wavenumber is equal to 2π multiplied by the inverse of wavelength, which yields units of radian per meter. [5], [4].

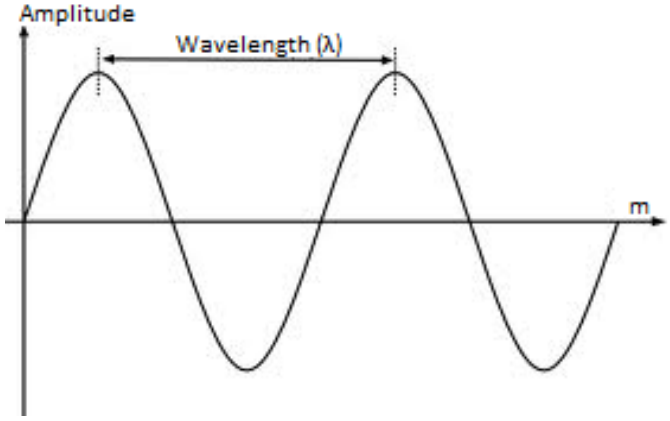

Figure 3-Demonstration of Wavelength [m]

The phase constant (ϕ_o) represents the displacement at time zero such that y(x,0) is not equal to zero. A sine wave is equal to a cosine wave when sine has a phase constant of $\frac{\pi}{2}$ radians. [5].

2.2 Antennas

As defined by IEEE, an antenna "is that part of a transmitting receiving system that is designed to radiate or to receive electromagnetic waves" [6]. There are many kinds of antennas, and each will demonstrate a characteristic behavior, known as the antenna pattern, which depends on its geometry (shape and size) and material of composition. The antenna pattern refers to the magnitude and direction of electromagnetic radiation of the antenna when a signal is being used for transmission or reception. If an antenna demonstrates the same pattern while receiving as it does while transmitting, it is a considered a reciprocal device. [4].

An isotropic antenna is a reciprocal antenna which radiates EM waves uniformly in all directions, as seen in Figure 4. Although this type of antenna does not exist, because no antenna can radiate uniformly in all directions, it does serve as a good starting point in conceptualizing the idea of antenna pattern. [5].

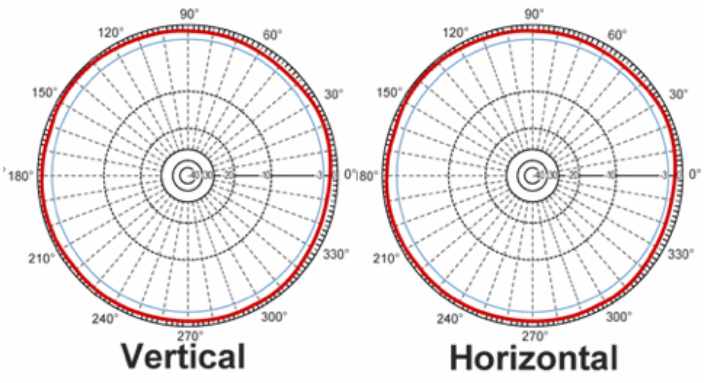

Figure 4- Antenna Pattern of an Isotropic Antenna

Although antennas cannot radiate in all directions uniformly, certain antennas and their respective orientation can be optimized to radiate EM waves in a desired direction. The directivity of an EM wave is described by the orientation of an antenna such that its radiation is at a maximum in a specific direction. The concept of directivity can be represented visually in a coordinate system classified by a zenith elevation plane (θ) and an azimuth plane (ϕ). This coordinate system is shown in Figure 5 below. [4], [7]. Gain is a term used to describe the efficiency (ability to transfer input power to output power) and directivity of an antenna. An antenna characterized by large gain signifies that most of its input power is converted into EM waves in a certain direction. [12].

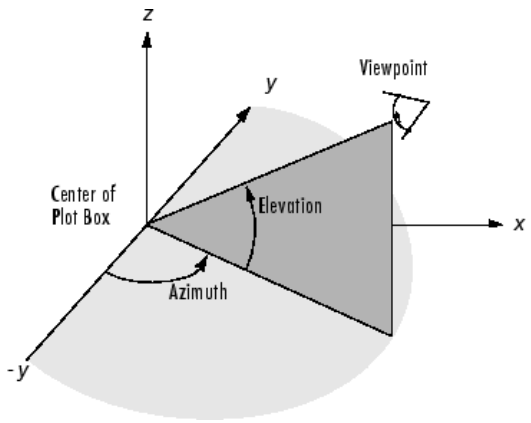

Figure 5-Specifying the Direction of a Radiated Waveform

Consider the simple dipole antenna consisting of two cylinders of specified length and radius shown in Figure 6. When a voltage is applied simultaneously to each of the cylinders, they become excited and radiate an electric field of differing magnitude in certain directions. Figure 6 demonstrates a simulation of a dipole antenna using COMSOL [8]. Knowing the orientation of the antenna in which the highest magnitude of radiation occurs is beneficial, thereby allowing the designer to direct the antenna where he or she needs the EM waves of highest magnitude to be sent. [7].

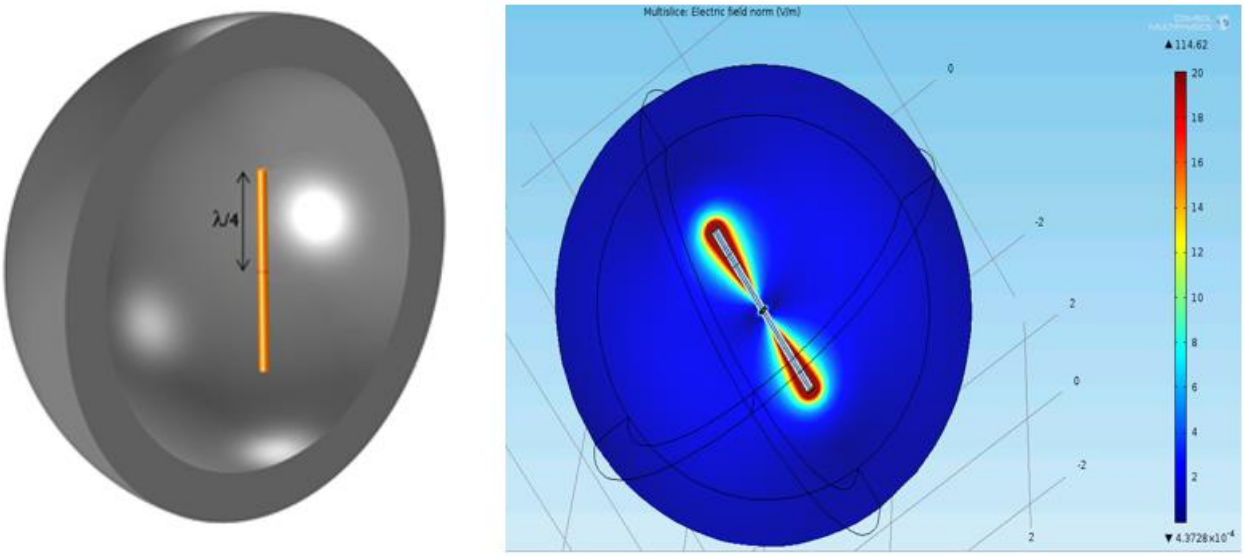

Figure 6-An Isolated Antenna and its Radiation Pattern

2.3 A Brief Description of Radar

Radar is an acronym standing for "Radio Detection And Ranging". The purpose of radar is to locate and track objects in a three dimensional space. Figure 7 shows a block diagram of a basic radar system. The "Transmitter" block represents a generator being used to produce a discontinuous electric signal, which is sent by transmission line to an antenna [9]. The antenna is the medium through which the electric signal is converted into an electromagnetic wave, or "Radar Pulse", through free space. This pulse can be oriented in a specific direction characterized relative to where an object is expected to be located. After the pulse is transmitted, the radar switches to its "Receiver" mode in which it is waiting for a return pulse. This return pulse, described as "Scattered Pulse", is the pulse that hits an object and returns back to the antenna [10]. The data received from the wave, such as phase and amplitude, can be used to describe physical features of the object it has detected, such as velocity (speed and direction), location, and shape. The accuracy of the data being used to describe the features is dependent on the antenna and processing capabilities of the radar. [9].

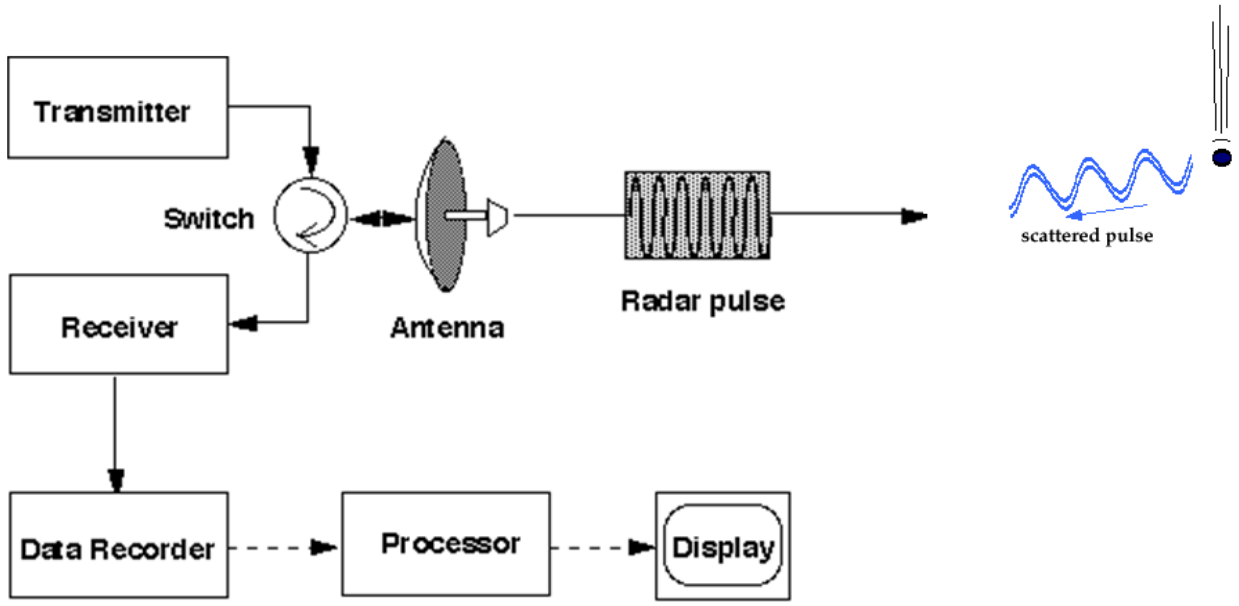

Figure 7-Block Diagram of a Basic Antenna System [10]

The radar's process of transmitting and receiving is periodic. This behavior is demonstrated in Figure 8 below. The time in which the radar is transmitting is referred to as the Pulse Width ("PW"). The time in which the radar is waiting for a return signal is known as the Rest Time ("RT"). Pulse Repetition Time ("PRT") is the time between the transmitting cycles. It is important to select an RT with long enough duration such that transmit pulses do not interfere with return pulses. RT can be selected if the general location of the object being searched for with respect to the radar system can be estimated. Knowing that electromagnetic waves travel through air at the speed of light of the medium, it is possible to estimate the amount of time it will take a transmitted signal to return to the antenna after reflecting off an object. [11].

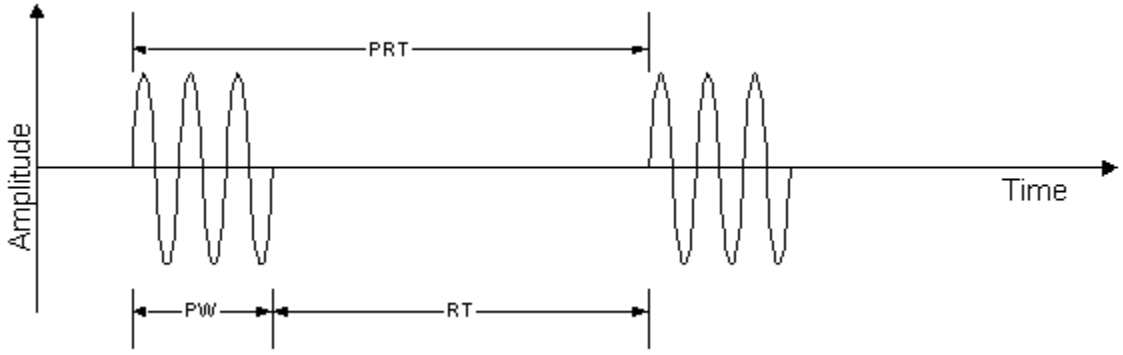

Figure 8-Transmit and Receive Periods of Radar [11]

There are many factors that affect a signal transmitted and received by the antenna during its time traveling through space. After the EM wave is propagated into free space by the antenna it is

subject to environmental factors which can attenuate (weaken the amplitude of) or scatter the beam. According to Dr. Robert O'Donnell of MIT's Lincoln Lab, the four main causes of attenuation and scatter of a signal propagated by a radar system are atmospheric attenuation, reflection off of the earth's surface, over the horizon diffraction, and atmospheric refraction. When EM waves propagate through the atmosphere, energy of the wave will scatter throughout the atmosphere and reduce the strength of the radar beam. Power is dissipated by water and oxygen in the air. Energy in lower angular portions of the beam will be reflected off of the earth's surface. Depending on the surface (water or land) these portions will be affected in various ways. Atmospheric refraction is caused by humidity and density, varying temperatures, pressure, fog and cloud water content and rain rate. All of these parameters are not homogeneous throughout the different layers of earth's atmosphere and attenuate the waveform as it propagates through space. [9].

2.4 Phased Array Radar

In order to build a radar system capable of transmitting higher power over longer distances, an array of antennas, referred to as a phased array, can be implemented. There are many benefits to using phased array radar. One such benefit is that it allows for remote control of the direction of the radar's beam by controlling the phase and amplitude that each antenna transmits with a stationary physical orientation of the radar. Another advantage is that each antenna can constructively interfere with the other antennas in the system, causing additive waves to increase the magnitude of a beam in a specified direction. This type of system allows for increased gain, increased Signal to Interference and Noise Ratio (SINR), and beam forming and determining of direction received signals.

The radiation pattern of a phased array radar system can be calculated as the sum of the product of each antenna's "element pattern" multiplied by its respective "array factor" [14]. The computation of radiation pattern is shown in Figure 9 below, where 'Y', '$X_{1,2..N}$' and '$w_{1,2,...N}$' represent the array's radiation pattern, the single element pattern and array factor for each element in an array. The element factor is synonymous with the concept of antenna pattern described in *Antennas* (Section 2.2). Element factor can be affected by mutual coupling, a phenomenon that occurs when one or more antennas cause the impedance of another antenna to change during transmit and receive. Array factor is a vector product representing the relative phase shift of an antenna element, its radial distance from the origin of the array face, and the excitation which is applied to achieve radiated electromagnetic waves from the element [15].

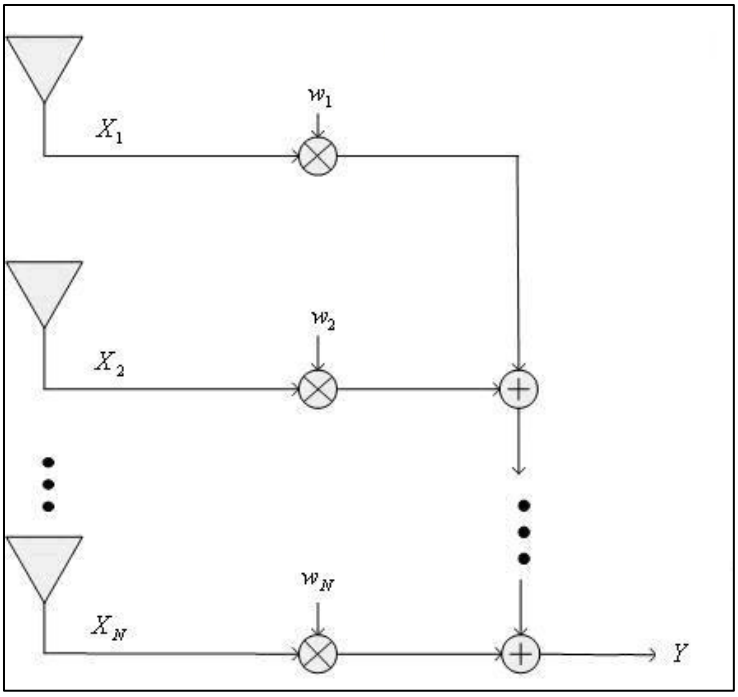

Figure 9-Schematic for the Input and Output Gain of an Antenna Array

An illustration of beam formation with four sources of transmitted waves is shown in Figure 10 below. Notice the sources on the Figure's left transmitted wave formations earlier than those on the right. As a result, the four sets of wave fronts constructively interfere with each other in common locations. The solid line drawn through these overlapping propagations represents the transmit direction of the main lobe. The main lobe is the beam formed in which the transmitted signal has highest amplitude due to phase shifting. Its width can be defined by the range of angles (in the Directional-Sine coordinate system, see [17] and Appendix C) with magnitudes greater than or equal to 3dB less than the maximum gain. Phase shifting is a method used to adjust the direction of the main lobe through constructive interference of transmitted signals. Increasing the number of transmitting elements in a phased array radar will create a narrower main lobe. It is also important to note that there are other directions in which some antenna sources add to each other to create side lobes. In these directions, the magnitude is less than the main beam formed. In Figure 10, the dashed line shows where the waves are combining to form a side lobe. In addition to the side lobes, a grating lobe may be present in the radiation pattern of a beam directed at a large angle off of boresight (perpendicular to the face). The grating lobe's magnitude is much higher than those of the side lobes. [7], [9], [10].

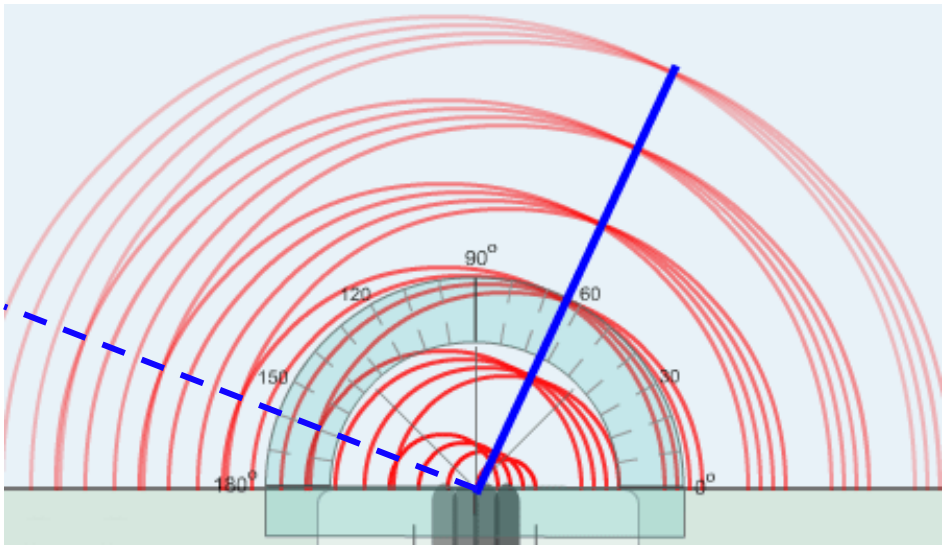

Figure 10-Phased Array Beam Formation

2.5 A Description of the AMISR

The AMISR is a robust system because of its ability to be controlled remotely, and the redundancies and organization incorporated into its design. SRI's AMISR project consists of 3 separate "faces". Each face encompasses the body on which all of the 4096 Antenna Element Units (AEUs) are housed. The AEU contains a solid-state power amplifier (SSPA) and cross dipole antenna. The face is divided into 8 groups of 16 panels (making 128 identical panels) which contain Panel Control Units (PCUs) used for AEU control and monitoring. There are 32 AEUs on each of the panels. Figure 11 shows the organization of AMISR's system previously described. One of the faces, Poker Flat Incoherent Scatter Radar (PFISR), is located in Poker Flat, Alaska. It has been in operation since 2007. The North Face and Canadian AMISR faces are both located in Resolute Bay, Nunavut Territory, Canada. They are referred to as RISR-N and RISR-C, respectively. [2].

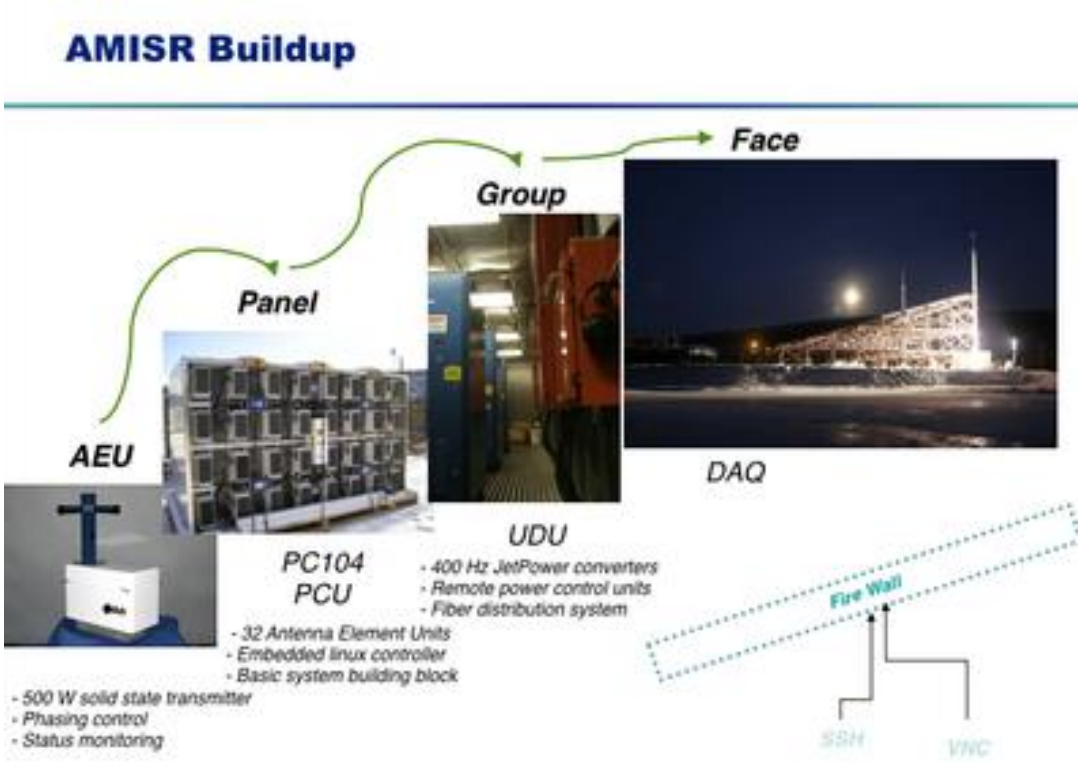

Figure 11-The AMISR System [2]

2.6 Modeling Software and Simulation Tools

There are many powerful software packages which simulate antenna radar systems. Advances in computational power of computers serve as the backbone for carrying out the complex solutions to the systems of equations used to model antennas. The group modeled antenna systems in the frequency domain using numerical analysis. There are two main types of computational electromagnetic software to choose from, Method of Moment (MoM) and Finite Element Method (FEM). Each type offers certain advantages in the solution of a computational model, however each must assume certain idealizations in order to make the simulation of a model feasible. Although these idealizations may simplify the solution of a model, they may decrease the accuracy of the results. The hierarchy of computational electromagnetics is organized in Figure 12 below.

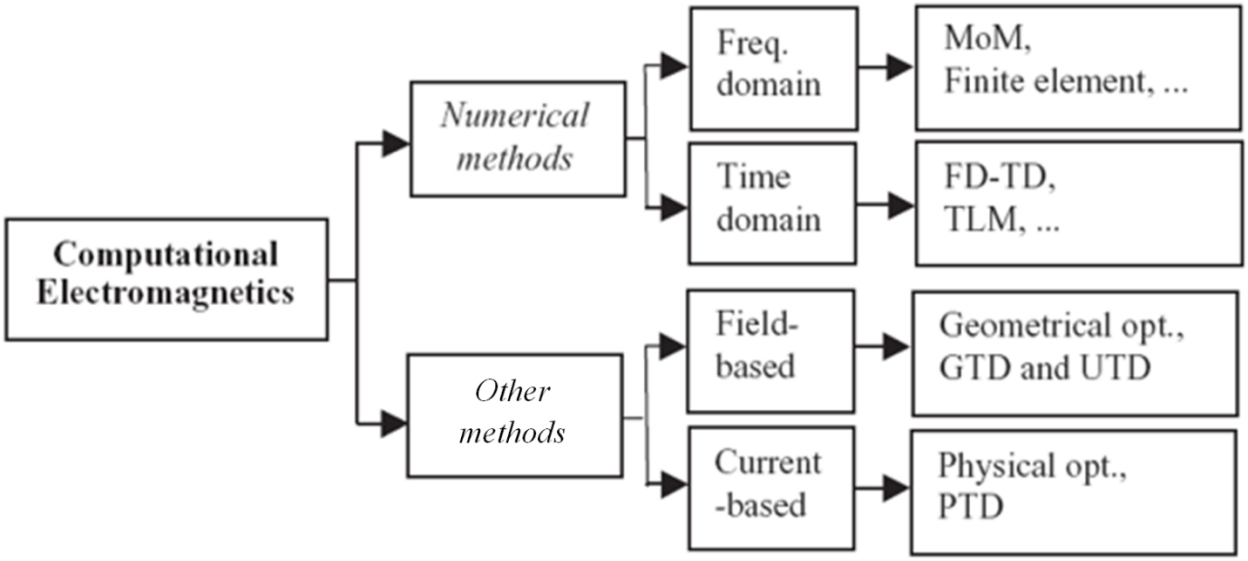

Figure 12-Variety of Computer Aided Antenna Design and Analysis

MoM is an approximation method implementing the computation and solution of linear partial differential equations. User defined or automatically set boundary conditions define the geometry of the model being simulated. The simulator calculates values for various equations according to the boundaries defined during its computation. Simulation tools available in market such as NEC2 and EZNEC implement MoM for solving antenna models. The group has implemented the modeling techniques as described in [14]. These techniques are interchangeable depending on the array being modeled and the desired accuracy of the user [14].

The second type of computational electromagnetic software of interest to the group is FEM. FEM was originally developed for structural analysis and is good for irregular constraints inconsistencies intrinsic in a material's properties. This type of electromagnetic simulation uses the following process: discretization of the total environment into polygon elements with an associated and approximate constant value, element characterization by potential equations and field estimates, and lastly solving the system of equations which relate the interactions that each element has on one another. Simulation tools like Ansoft HFSS, COMSOL Multiphysics, and FEKO are based on this modeling approach.

Section 3: Methodology

The ultimate goal of the group's work was to develop an accurate model of the AMISR's far-field radiation pattern. In order to do so, a software package, which served as a platform for radiation modeling and performance investigating, was created. Further iterations of the model enhanced the accuracy of the computed results. These iterations included an investigation of the array behavior. Phase errors were introduced to the array factor calculation. Element pattern was then introduced into the radiation pattern calculation. Finally, the group validated their model with an SNR test comparing calculated SNR with the measured SNR data from the PFISR.

3.1 Objective 1: Port Model from MATLAB to Python

The group was provided with a MATLAB model of the AMISR's radiation pattern which they translated into a Python environment. Using object oriented design allowed the group to produce a model of the AMISR with a more robust framework and additional functionality control. The hierarchy of the code emulates the modular design of the AMISR described in *Description of the AMISR* (Section 2.5). Working with one of their SRI mentors enabled the group to develop this hierarchy with Python's data structures. The use of control options are permitted by the modular design of the Python class structure of the AMISR model. Control options include having the ability to enable or disable JetPower status, individual panels, and individual AEUs.

To begin the transition between MATLAB and Python, the group needed to validate that Python can produce the same computation results as MATLAB. To do so, it was necessary to recreate a two-element array in Python and compare the computed radiation pattern results with those achieved by the MATLAB model. Additionally, the Python modeling results were compared with [16] which include contour plots and three dimensional representations of the two-element array's radiation pattern. The group had to develop tools to compare their simulated results against the resources provided by SRI. Such analysis tools include: robust contour plotting techniques, filtering through the computed radiation pattern matrix to see the radiation pattern in a specific direction, finding the local maxima of the main and side lobes, and calculating the beam width using the 3dB half power. By comparing the results and plots created by the Python model to those generated by MATLAB and shown in [16], the group was able to validate their results and move forward with modeling a larger array.

3.2 Objective 2: Investigate the Array Behavior

As described in *Phased Array Radar* (Section 2.4), the array pattern is defined by the sum of the product of the element pattern and its array factor for each element in the array. In order to understand the theoretical limits and behavior of the AMISR's radiation pattern, the group developed its modeling of the array factor and element pattern.

Developing the Array Factor Model

To achieve a higher understanding of the AMISR's array factor, the group assumed element pattern and excitation fields with normalized magnitudes. By calculating the array pattern under these conditions, the group was able to investigate the effect that changing the steering angle of the main lobe or disabling certain parts of the AMISR has on the array factor. When the beam is directed near the limits of the functional range of the AMISR, grating lobes are formed. The group was able to further investigate what angles these grating lobes occur at and confirmed the range of steering angles of safe operation. Next, the group simulated what happens to the array factor when JetPowers, panels, or AEUs

malfunction. This provided insight into how drastically the output power and directivity of the beam decrease when elements are malfunctioning.

Introducing Phase Error into the Array Factor Calculation

Phasing each element allows the user to direct the AMISR's main lobe. Phase error in the AMISR system is caused by PCU software and hardware limitations. By incorporating this feature in the model, the group was able to understand the effect on the behavior of the main lobe when the magnitude of that error increases. Understanding the array factor under various scenarios, such as those described above, assisted in understanding and predicting the AMISR's performance when a realistic element pattern and excitation is applied.

Developing the Element Pattern Model

The next iteration of our model introduced the element pattern into the radiation pattern calculation. The group used the Hertzian Dipole approximation in their first iteration of the cross-dipole element pattern [20]. This approximation changes with respect to each antenna's angular elevation, a magnitude represented in the azimuth-elevation coordinate system by radians away from the ground plane [17]. NEC software was used to calculate the element pattern specific to the geometry of the AMISR's cross dipole pair. In this stage of the modeling process, the group used the NEC data files produced by their mentor, Dr. Michael Nicolls, to investigate the effect that mutual coupling has on the individual pattern of an element. To do so, the group and their mentor ran simulations on the pattern of an isolated AEU, on each AEU independently in one panel, and then on each of the 288 AEUs in a 3x3 array. Because of computational, computer memory, and time limits, the group was not able to model the element pattern of each AEU on an entire AMISR face. After the group implements the NEC model of the element pattern, they compared the computed element pattern results with those produced by using the Hertzian dipole for each of the three cases.

3.3 Objective 3: Validating the Radiation Pattern Calculation

As mentioned in *Description of the AMISR* (Section 2.5), data has been collected on the status and health of the PFISR since 2007. The group populated the fields of their AMISR model's classes with the relevant information (power, and status of each AEU) taken from data sheets to calculate the far-field radiation pattern which the PFISR is exhibiting.

Satellite Pass Test

SRI's Geospace Division has been recording data (Signal to Noise Ratio data) defining the backscatter gain measured from spherical satellites which have passed through the PFISR's main lobe. Backscatter gain is defined as the total energy received by a radar which is reflected off of a conductive surface in space after being transmitted by the radar. Using known trajectory information about satellite passes, the Geospace Division can steer their beam multiple times to track the satellite passing through an AMISR's Grating Lobe Free Area (the star shape in Figure 13). The satellite's trajectory is shown as the orange line in Figure 13. The five red dots are the five successive steering angles of the AMISR's main lobe to track the satellite's passing.

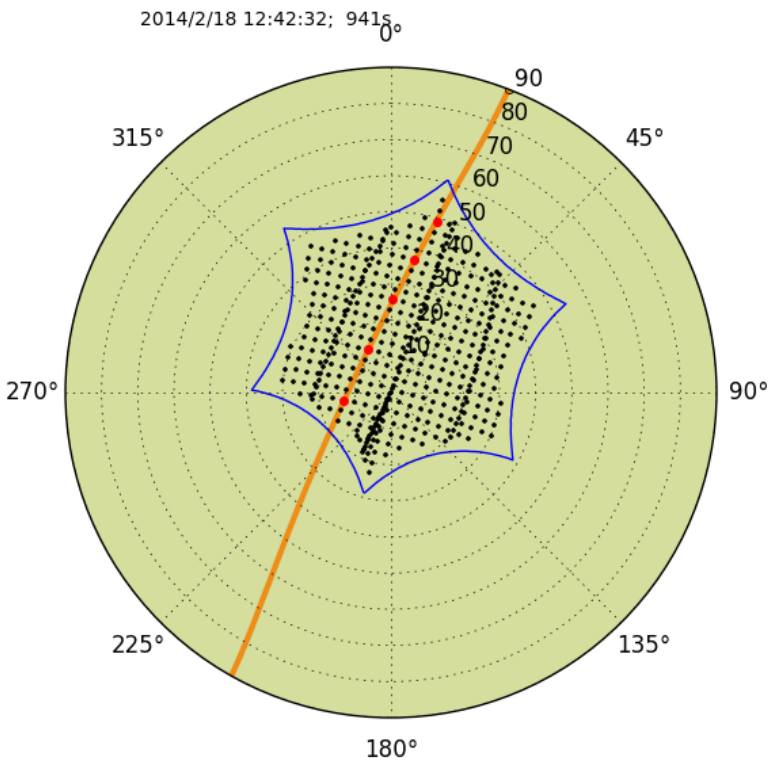

Figure 13-Satellite Trajectory through AMISR Grating Lobe Free Area

The group developed a method to model the passing of a satellite through the radiation pattern of an AMISR steering its main lobe in a specified direction. Knowing the beam's steering angle, one can produce its respective radiation pattern. By interpolating the satellite's known trajectory (defined by Directional-Sine angle pairs) against the calculated radiation pattern, one can find the corresponding "gain" for each point in the pass. These interpolated gain values are then used to calculate the theoretical power received by the AMISR. The following equations demonstrate how the theoretical SNR is calculated. The definition of each variable can be found in Table 1.

$$P_R = \frac{(P_T * G_{Tx} * G_{Rx} * c * \lambda^2)}{(4\pi)^3 * R_{Tx}^2 * R_{Rx}^2)} \quad [Eq.2]$$

$$P_N = k * T_{sys} * B \quad [Eq.3]$$

$$SNR = N_{Coh}\left(\frac{P_R}{P_N}\right) \quad [Eq.4]$$

Table 1-Definition for SNR Calculation Variables

Parameter	Description	Units
c	cross section	m^2
lam	wavelength	m
Rtx	range on Tx	m
Rrx	range on Rx	m
B	bandwidth	Hz
Tsys	system temperature	K
Gtx	transmit gain	
Grx	receive gain	
Pt	Total Power	W
Ncoh	nummber of coherent integrations	

Section 4: Results

4.1 Objective 1: Port Model from MATLAB to Python

The group implemented a structure for their software platform developed by one of their mentors. The platform's structure is set up with a class for each modular section of the AMISR: a Face class, an Array class, a Panel class, and finally an AEU class. Within each class there is a dictionary which contains the next sublevel of the modular design. The Face class contains a dictionary filled with an object of the Array class. The Array class has a dictionary which contains as many objects of the Panel class as there are panels on the specific AMISR. The Panel class has a dictionary which contains as many objects of the AEU class as there are AEU's per panel. Each modular class has an associated configuration file which stores user defined parameters. A more elaborate description of the code organization and implementation of the software is provided in the user manual attached in Appendix B. The hierarchical set up of the code is displayed in Figure 14 below.

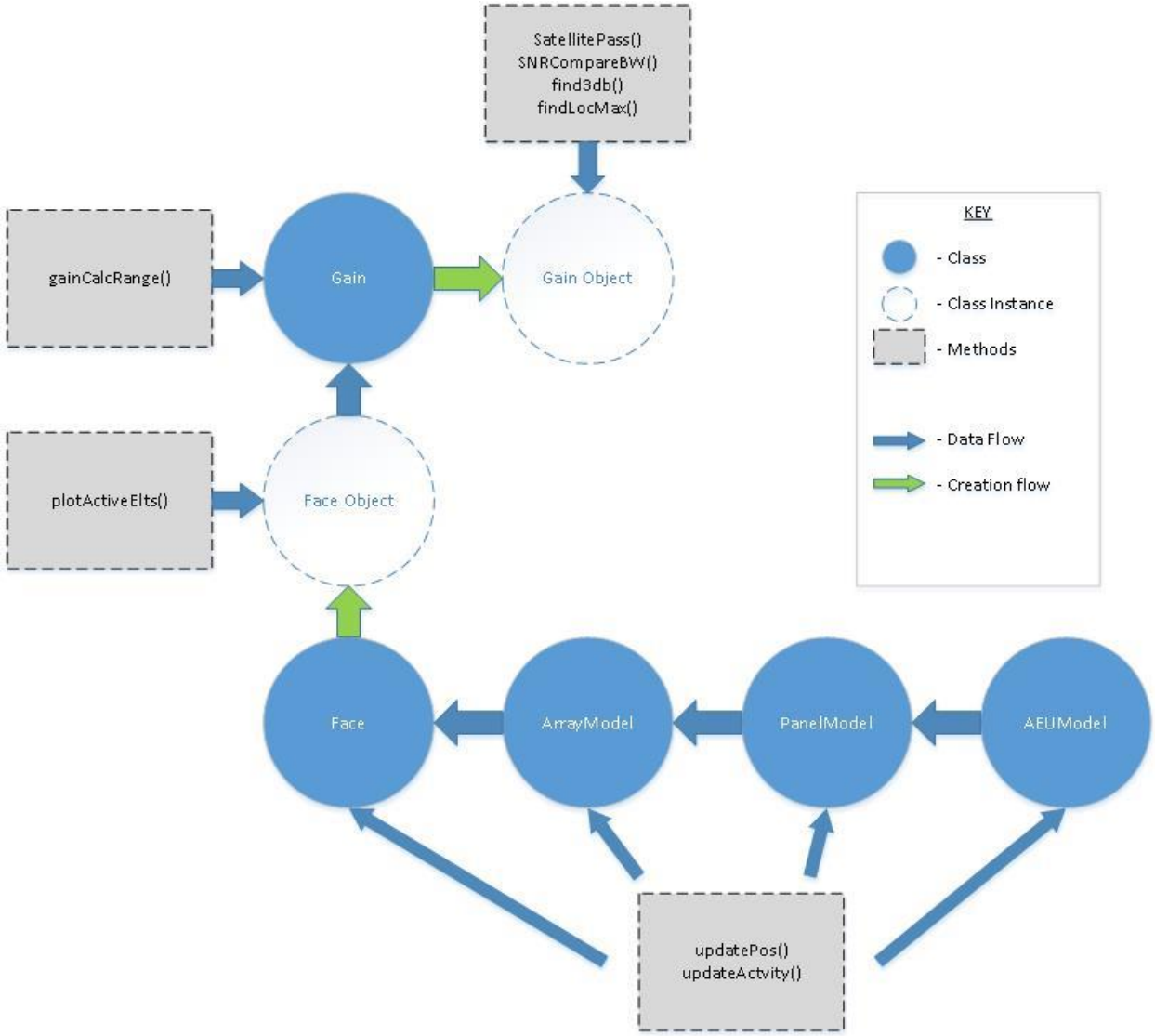

Figure 14-Code Flow Chart to Model the AMISR

To ensure that Python would yield results comparable in accuracy to the previous MATLAB model, the group researched the floating-point number's precision of Python (53 bits) [18] versus MATLAB's single-precision floating point (32 bits) and double-precision floating point (64 bits) [19]. According to Dr. Nicolls, this 53 bit level of accuracy would suffice for the precision needed in the model's computation and the implementation of a two element array was developed.

The first investigation is a comparison between an existing MATLAB model of a two element array and the group's initial Python model of a two element array. Plotting methods were created to produce three-dimensional images of the computed data, as the group was limited to using open source libraries for their code development. In Figure 15, the plots taken from [16] are provided. In Figure 16, the group's rendering is depicted. The leftmost three dimensional plot in Figure 15 is the radiation pattern of a two element array with a wave length spacing of one and an excitation of one [amp]. The rightmost three dimensional plot in Figure 15 is the radiation pattern of a two element array with one wave length spacing and an excitation of (1+i) and 1 in each element, respectively. The two-dimensional plots beneath the 3D plots demonstrate the radiation pattern at a specific azimuth slice through the 3D contour data. [See [17] and Appendix C for an explanation of the coordinate system being used in the following figures].

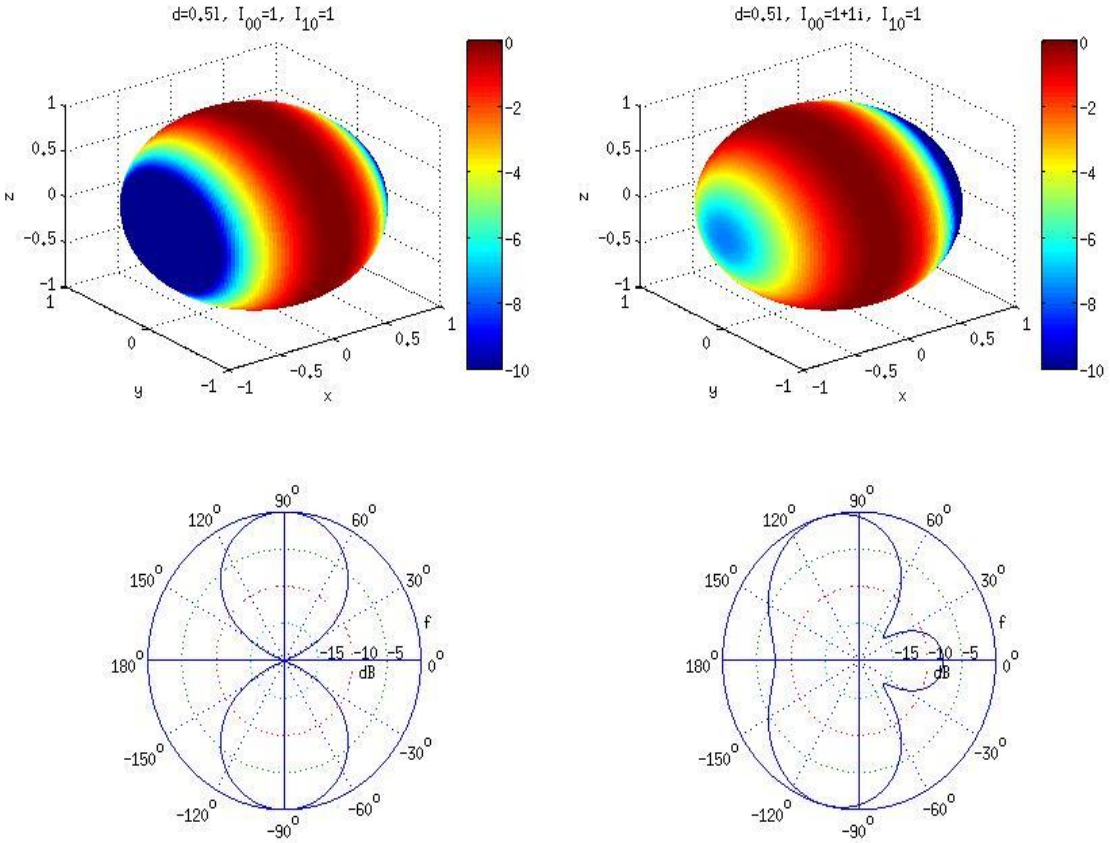

Figure 15-Contour Plots of a Two Element Array from PowerPoint [16]

For modeling the AMISR, it was only relevant to compute the radiation pattern excited north of the face. As such, Figure 16 demonstrates only the radiation pattern from 0° to 360° in azimuth and 0° to 90° in elevation for each discrete azimuth angle. It is important to note that the computed results are shown in a normalized magnitude whereas the MATLAB provided results are given in normalized decibels [dB]. It is clear from Figure 16 that Python produced comparable results to the MATLAB generated computations.

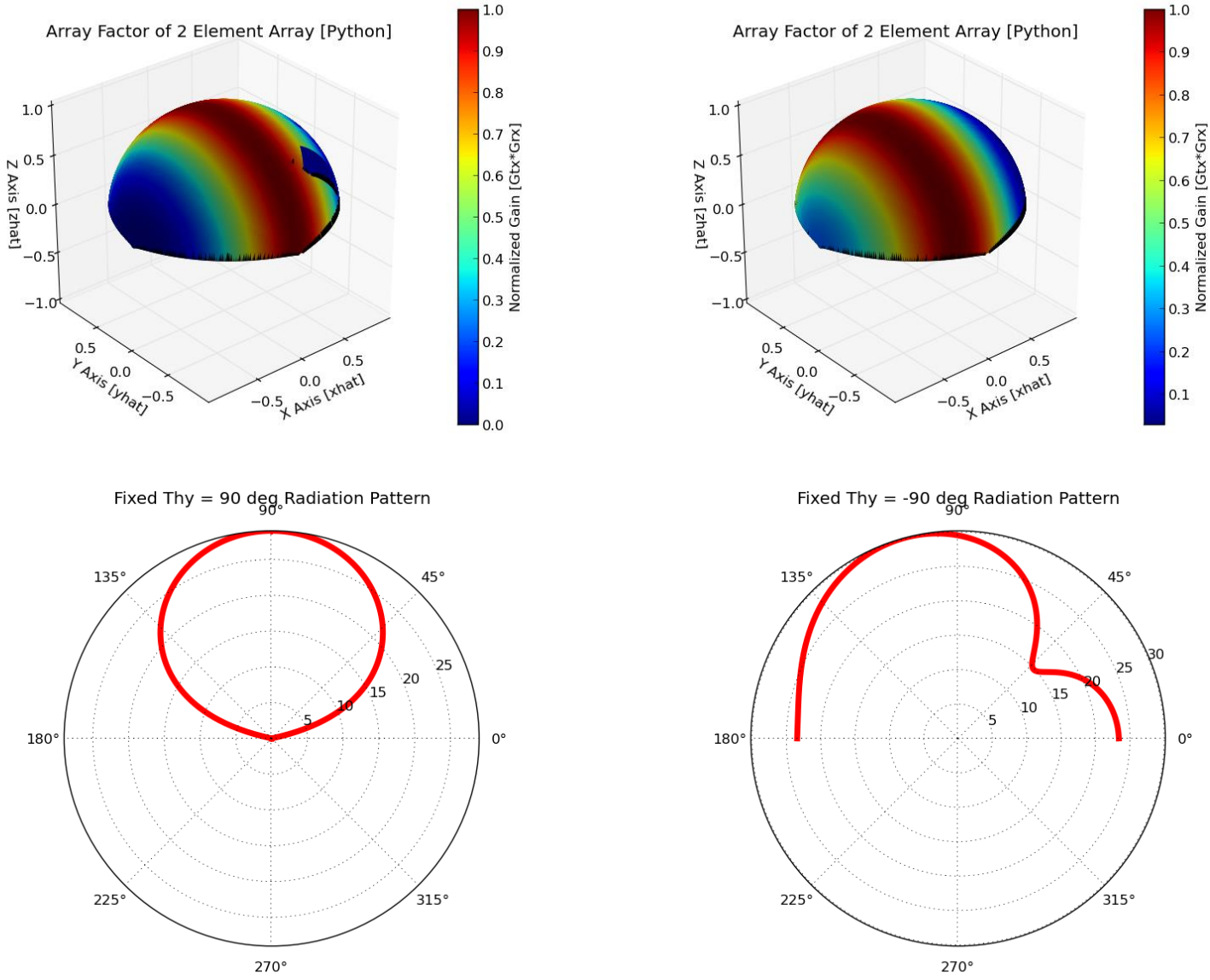

Figure 16-Contour Plots of a Two Element Array from the WPI Group

Aside from [16], SRI provided a MATLAB baseline model of the AMISR's radiation pattern. Using SRI's simple MATLAB model of the AMISR's radiation pattern, the group loaded in a two element array

and computed its corresponding radiation pattern. The results are shown below in Figure 17 represented in the Directional-Sine coordinate system (Theta-X, Theta-y) [17] [Appendix C]. The group developed a method to plot a two dimensional contour plot, also in the Directional-Sine coordinate system, shown in Figure 18 the using Python. In Figure 18, a masking circle which encapsulates all "real components" of the radiation pattern is demonstrated visually. All components of the pattern which are seemingly missing from the plot are ones which are not realizable in the Directional-Sine coordinate system [17].

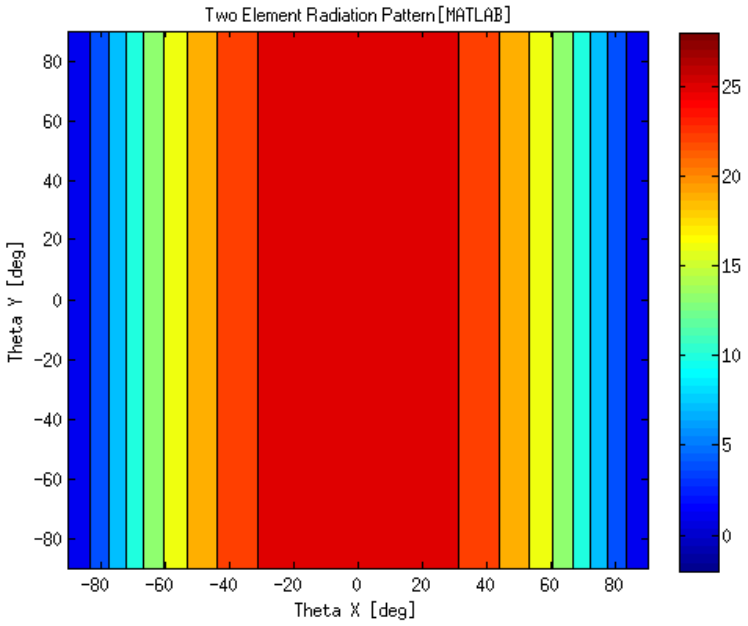

Figure 17- Two Element Radiation Pattern [SRI's MATLAB Baseline Model]

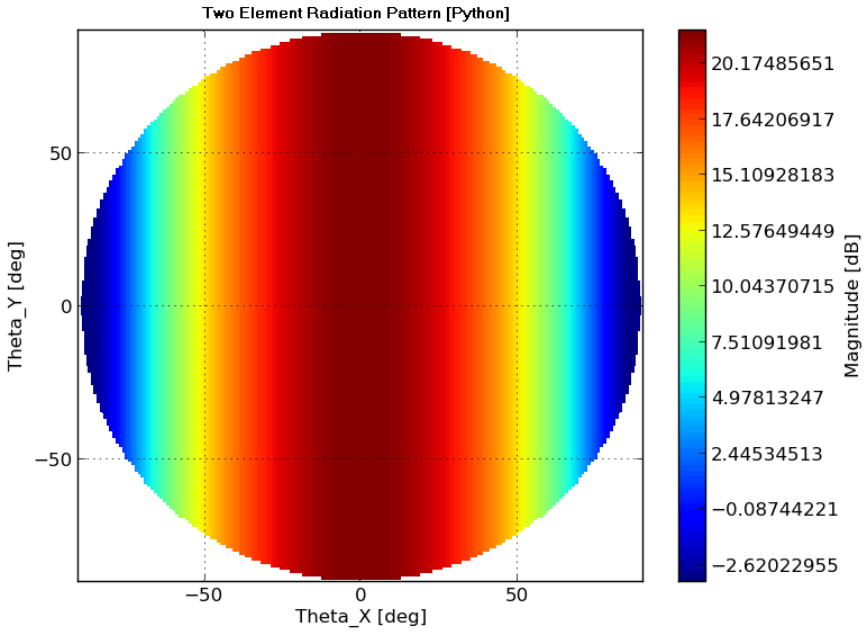

Figure 18-Two Element Radiation Pattern [Python]

It is clear from the plots that the values computed by Python and MATLAB produce comparable results in terms of magnitude and overall beam characteristics. As such, the group extended their model to include all 4096 AEUs. The relevant information for each AEU was initialized into the array object created by the code. Figure 19 is the radiation pattern of the AMISR using the MATLAB model. Looking at the axes of the plot, one can tell that the direction of the main lobe was steered at an angle (0°, 0°) from boresight. The magnitude of the main lobe is around 30dB, a value which was lower than expected from the theoretical model of a 64x64 element array [16].

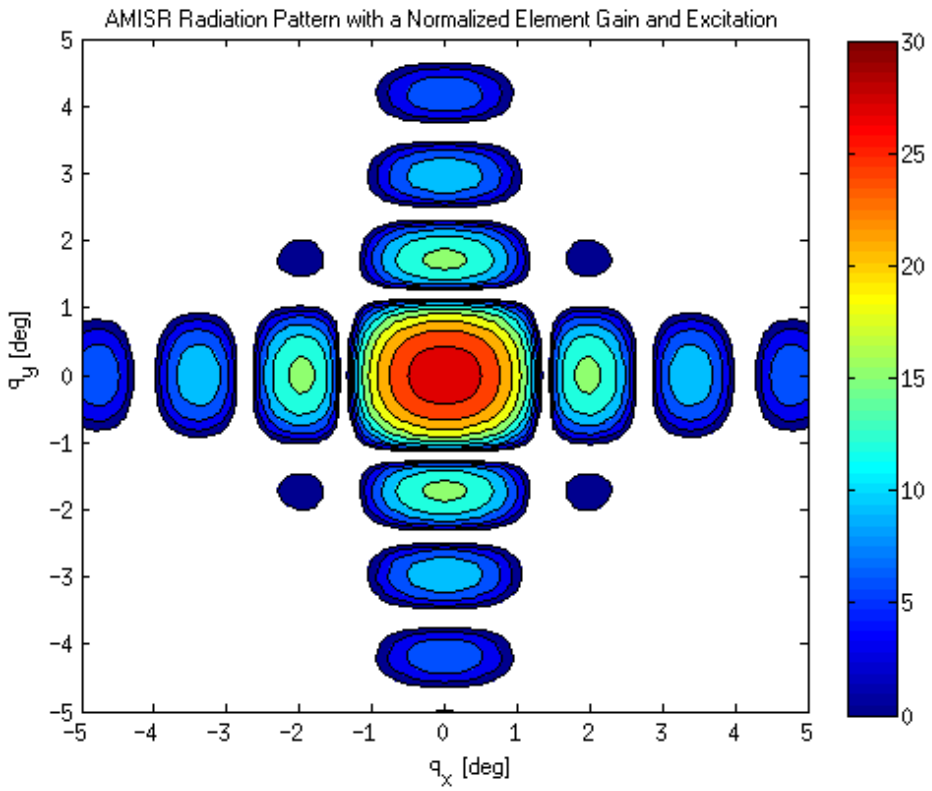

Figure 19-AMISR Radiation Pattern with a Normalized Element Gain and Excitation [MATLAB]

Figure 20 is the Python replication of the AMISR radiation pattern with normalized element gain and excitation. The value of maximum gain of the main lobe computed by the Python model at 43dB corresponds to what was expected using the theoretical model of a 64x64 element array at a steering angle of (0°, 0°) from boresight.

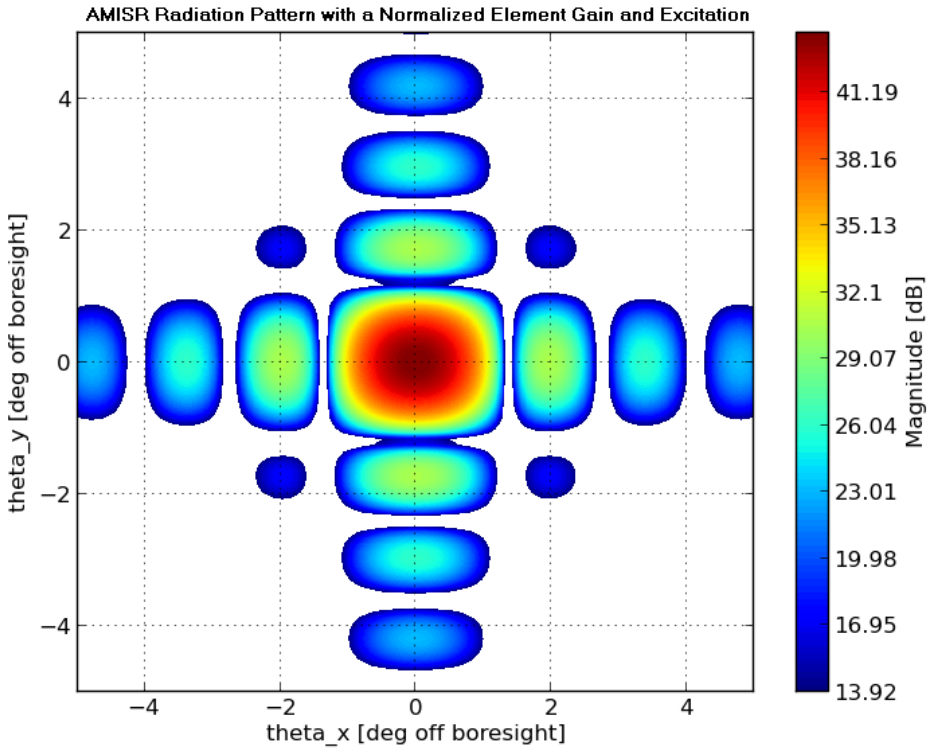

Figure 20-AMISR Radiation Pattern with a Normalized Element Gain and Excitation [Python]

At this point in the group's progress, the radiation pattern results were presented to Dr. Nicolls to confirm their validity. The maximum dB value of the main lobe for the 64x64 Python model versus the existing MATLAB model proved to be comparable. More importantly, the shape and side lobe formations of the Python model matches the expected pattern of the MATLAB models. This ideal radiation pattern shows the capability of the code structure and the robust plotting tools which will be used as a platform to add many more system parameters to the Python Model.

4.2 Objective 2: Investigate the Array Behavior:

The group began their investigation by normalizing the AEU excitation and element pattern. Doing so provided insight about the operational range of the array factor as a function of steering angle and active (transmitting and receiving) elements. Phase errors, which are a result of the computational limitations of the AMISR PCUs, and element pattern were then introduced into the array radiation pattern calculation.

Developing the Array Factor Model

In this section of the results, a description of the AMISR's array factor and an analysis of its behavior is provided. A description of the tools used to calculate the data and generate the visual representations of the array factor is included in the User Manual [Appendix B]. The purpose of these tests was to develop a full understanding of the pattern of the array factor. The array factor is can be characterized by the magnitude and radius of the main lobe, magnitude and location of the first side lobes, and the appearance of grating lobes. These characteristics were investigated as numerous

parameters of the array were varied. These parameters include size of the array and the steering angle of the main lobe.

In order to vary array size, the transmit and receive power fields of a number of AEUs were enabled. As the array size increased, the maximum magnitude of the radiation pattern's main lobe increased and its width decreased. Although data in Table 2 was collected for many iterations of increasing the array size, a visual representation has been provided for only a 1x2 panel array (64 active elements, Figure 21), to a 5x10 array (1600 active elements, Figure 22), to the full 8x16 AMISR array (4096 active elements, Figure 23). In each of the following three figures, the enabled AEUs are demonstrated in green and all disabled elements are red. The radiation pattern is shown to the right of the active elements plot. The maximum magnitude of the array factor (AF) and directivity of the main lobe increases substantially as the number of active elements increases. This behavior is characteristic of phased array radar systems and matches what was expected in both magnitude and performance [16]. The circumference of the main lobe is shown visually in Figure 21, Figure 22, and Figure 23 as the dotted white line. It is defined by a 3dB cutoff from maximum magnitude as described in Section 2.4. It is relevant to note that the magnitude of each of the three aforementioned plots is relative to 42.67 dB, the maximum measured magnitude for the AF of a 4096 element array.

Table 2-Radiation Pattern Magnitude and Beam Width as Array Area Increases

Enabled Elements	Max Magnitude of AF [db]	BW X [deg]	BW Y [deg]	RMS Radius [deg]
64	6.540176915	9.734734936	8.555127199	4.614809691
256	18.57929734	4.865069543	4.267055184	2.305940667
576	25.61948116	3.244529409	2.843756399	1.537366683
1024	30.61217487	2.434065754	2.135426734	1.151490532
1600	34.4823344	1.949588325	1.710055744	0.926040566
2304	37.64195566	1.623711165	1.422880294	0.769586942
3136	40.31080716	1.392515523	1.220167542	0.660376217
4096	42.62007577	1.218767984	1.070258493	0.579010462

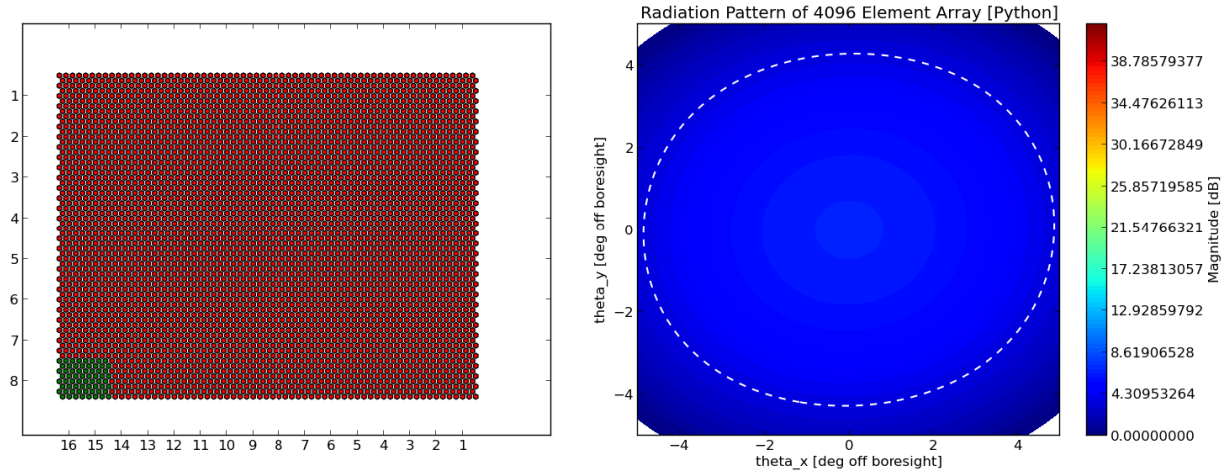

Figure 21-Active Elements and Radiation Pattern of a 1x2 Panel Array

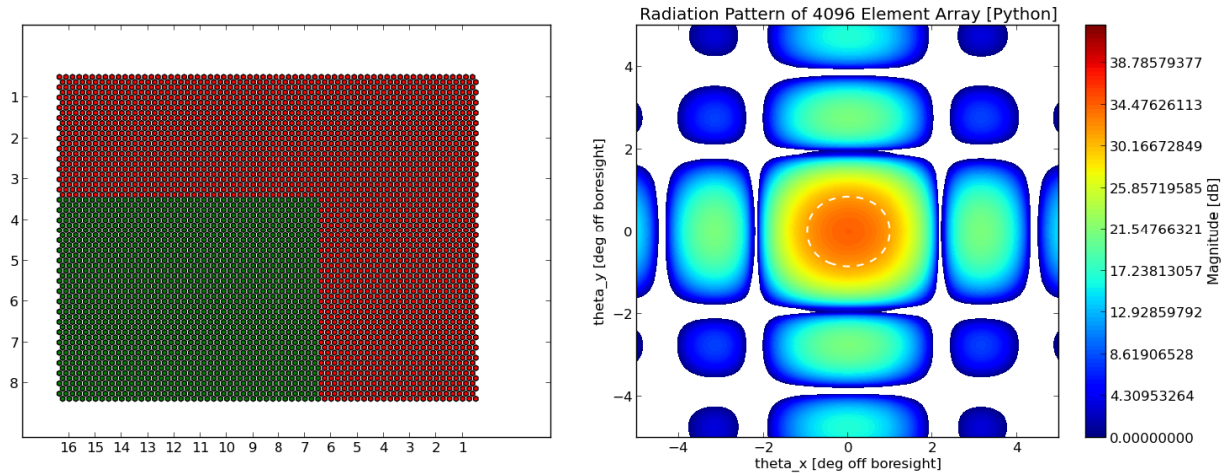

Figure 22-Active Elements and Radiation Pattern of a 5x10 Panel Array

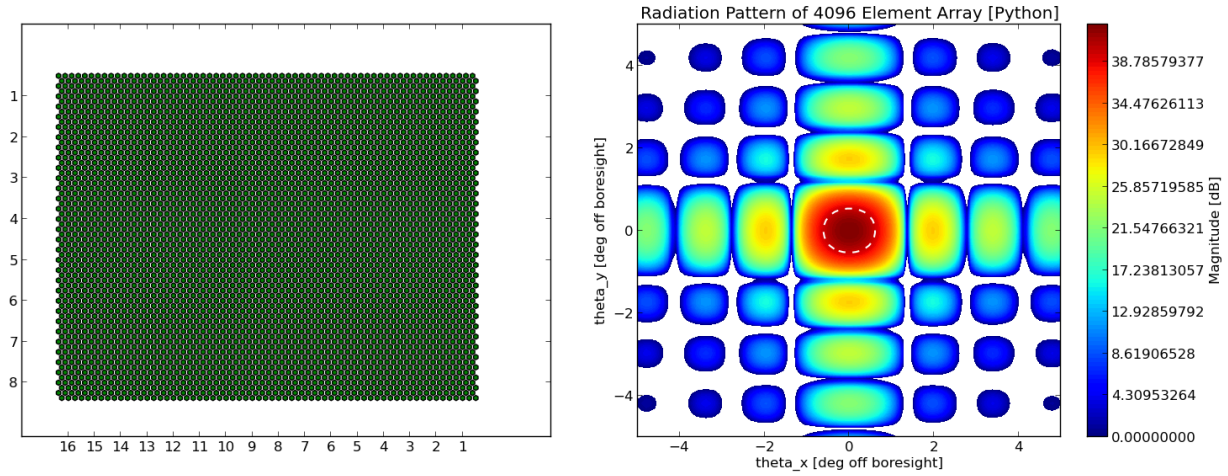

Figure 23-Active Elements and Radiation Pattern of an 8x16 Panel Array

The next iteration of the AF characteristics came with an investigation of the operation region of the AMISR's main lobe. The region of safe operation is limited by the directional angles at which grating lobes are formed. After producing AF plots at various steering angles, it was found that grating lobes occur around 30° to 40° away from boresight in both Theta-x and Theta-y directions. An example of the production of a grating lobe is shown below when the main lobe was steered at an angle (0°, 35°) in Figure 24. The AF calculated at the steering angle of the main lobe is 42.67 dB as expected, but a grating lobe appears at around (0°, -40°). Although others exist, this is one example of a steering angle which produced a grating lobe. From this experiment, it is now confirmed that the AMISR's steering operation range (also referred to as the "Grating-Lobe Free Zone") is limited to roughly 30° from boresight in Theta-x and Theta-y coordinates.

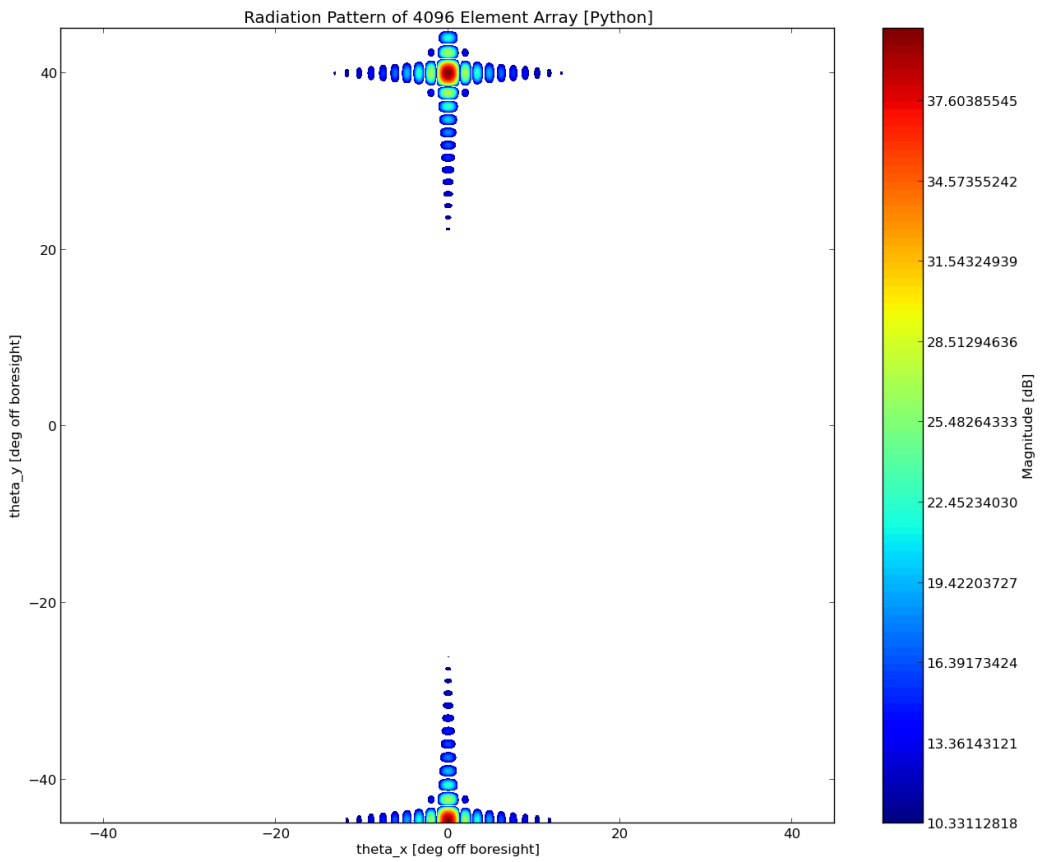

Figure 24- AF and Grating Lobe at a Steering Angle of (0°, 35°)

To summarize the results found in this section, investigating the array factor was made possible by normalizing the element gain and excitation. Doing so showed that the maximum magnitude and directivity of the main lobe increases as the number of elements in an array (and correspondingly array size) increases. Additionally, the 64x64 hexagonally-spaced element array configuration of the AMISR will produce grating lobes when the steering angle of the main lobe is directed more than 30° away from boresight (in either Theta-x or Theta-y).

Introducing Phase Error into the Array Factor Calculation

To compensate for the phase errors inherent in the AMISR system (due to software and hardware limitations), phase errors have been introduced into the model. The purpose of doing so was to characterize how the directivity of the main lobe deviates from its intended direction due to the phase errors and to gauge the overall effect that phase errors have on the radiation pattern performance.

The first test compared the model with no phase errors to one in which truncation (rounding errors) and phase offset were introduced. In this test, the phase offset was set to a maximum value of 1. This offset value was randomly selected from 0 to 1 and added to the phase which was calculated in the array factor equation in the model. The truncation of the sum of phase and phase offset was set to 5 decimal points. In Figure 25 and Figure 26, the array factor pattern (on the left) is shown with the magnitude of each lobe along a cut line of Theta-y equal to 0 (on the right). The results of having no phase error introduced into the model are shown in Figure 25, while the model with truncation and a phase offset is shown in Figure 26.

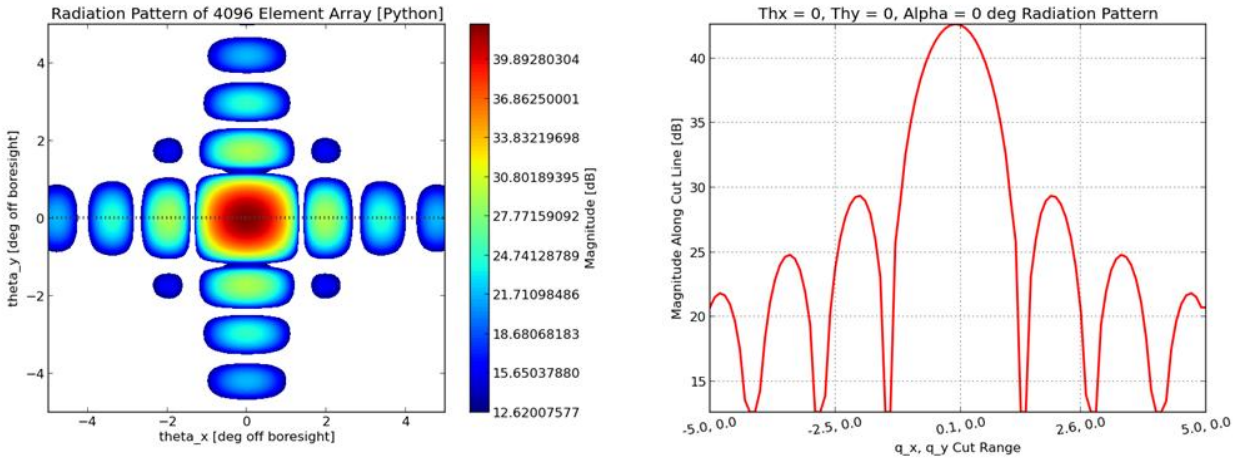

Figure 25-Array Factor Pattern and a Cutline Representing Magnitude of Side Lobes

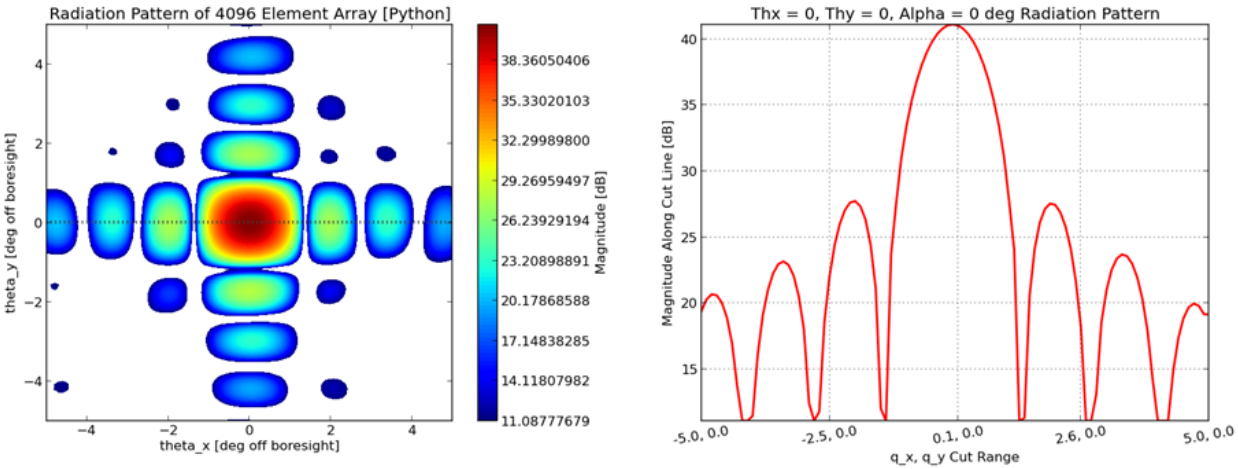

Figure 26-Array Factor Pattern and a Cutline Representing Magnitude of Side Lobes with Phase Error

It is clear that by introducing phase offset and truncation, there is significant performance degradation due to the emergence of new side lobes in the model. Although not depicted previously, there was extreme degradation in the uniformity of radiation pattern as the maximum phase offset was increased. Additionally, the magnitude of the main lobe decreased with the introduction of phase error. Introducing phase error into the model had no considerable effect on the direction of the main lobe. In other words, the center of the main lobe remained at its intended steer angle. This information is shown in **Error! Reference source not found.**.

Table 3-Phase Error Comparisons

Truncation	Maximum Phase Offset	Main Lobe [dB]	xCenter	yCenter	RMS Radius
0	0	42.62007577	49	49	0.579010462
5	0.1	42.60673119	50	50	0.578811063
5	1.0	41.08777679	50	50	0.579084076
5	2.0	35.69175923	49	50	0.577111369

To summarize the results found in this section, phase error does have an effect on the performance of the AMISR's array factor and therefore on its radiation pattern. Although it was anticipated that increasing the phase error would compromise the steering direction of the main lobe from its intended angle, the results proved otherwise. Additionally, the width of the main lobe did not increase due to an increasing phase error. Introduction of the phase error did however create more side lobes and eventually would decrease the maximum magnitude of the main if too large.

Developing the Element Pattern Model

When introducing element gain into the array pattern calculation, it was important to choose a model which accurately represented the element pattern of an AEU in the array. In the first iteration, the Hertzian dipole model, shown in Figure 27, was used to represent the element pattern of a theoretical cross dipole antenna. It is important to note that this approximation does not take into consideration the geometry or the material composition of the actual AEU. As such, it is not expected to accurately represent the AEU's radiation pattern in every direction. It does serve as a sufficient baseline for comparing future iterations, however.

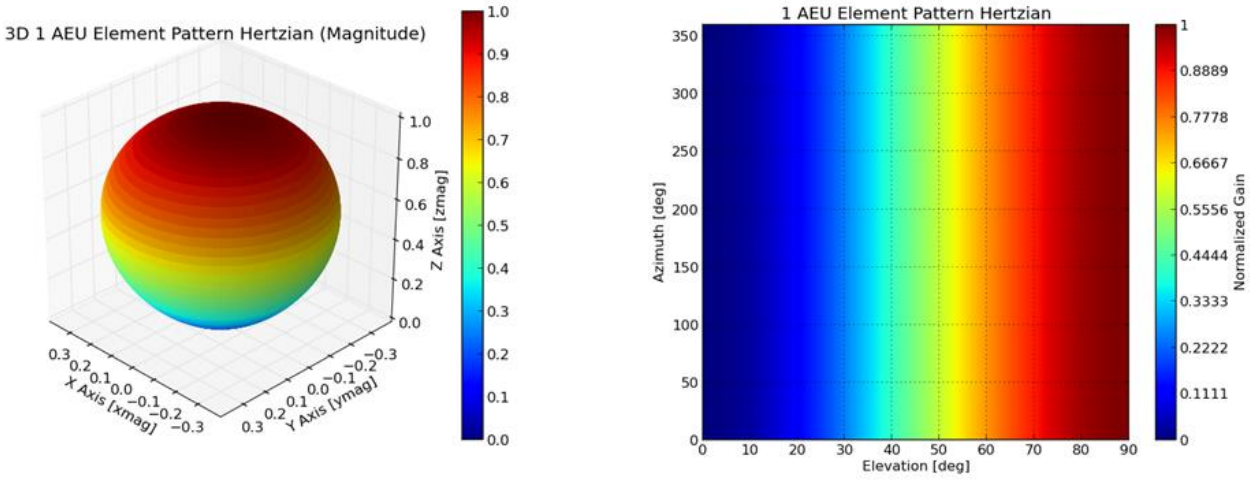

Figure 27-3D and 2D Representations of the Hertzian Dipole Model of a Cross Dipole Antenna

NEC modeling software was implemented to produce the element pattern specific to the composition and orientation of an AEU. The results of the calculated pattern are shown in Figure 28 below. The plot shows that the magnitude of highest radiation occurs at boresight for the isolated antenna. The 3D plot demonstrates this phenomenon. The plot in Figure 29 depicts a 2D slice of the element pattern of the isolated AEU at azimuth cut angles of 0°, 45°, and 90°.

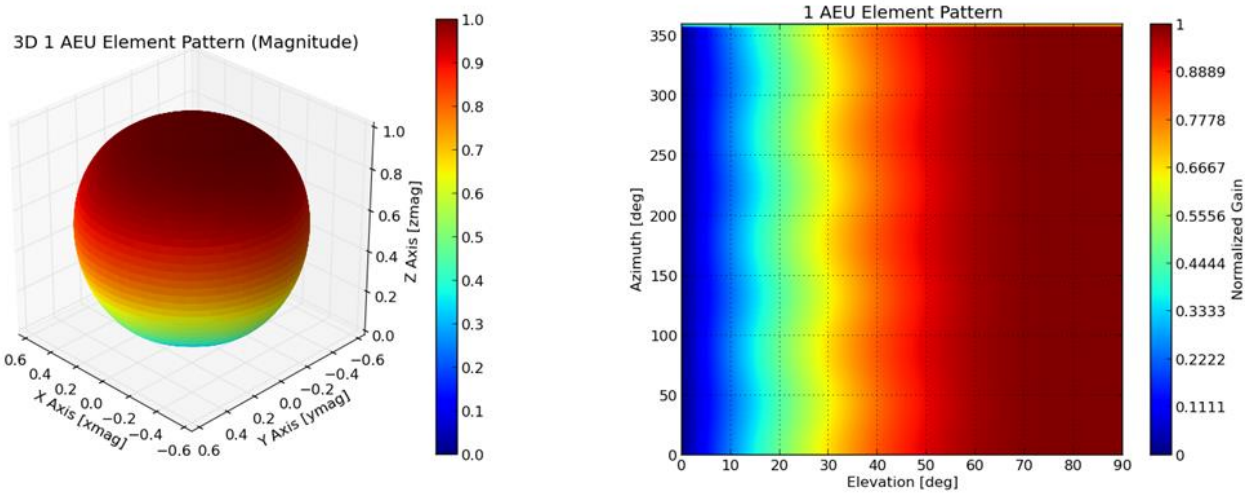

Figure 28-3D and 2D Representations of the NEC Model of an Isolated AEU

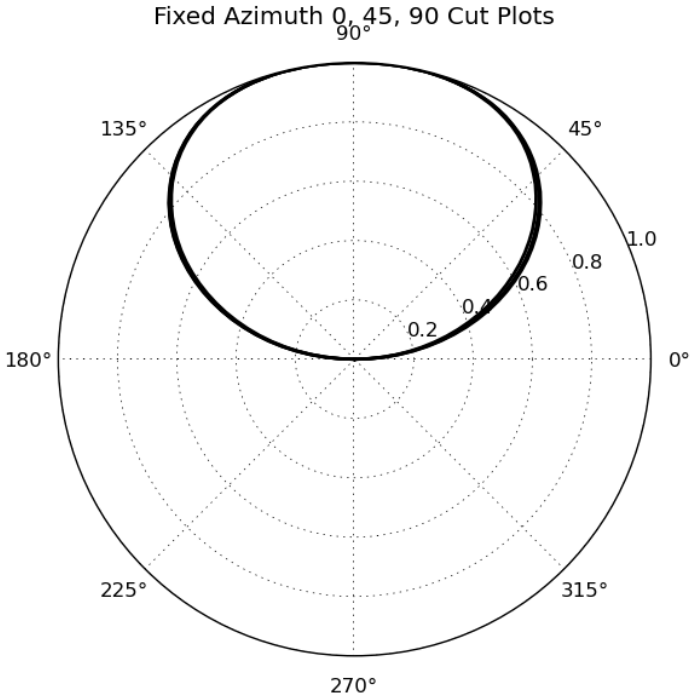

Figure 29-2D Slices of the Isolated AEU Element Pattern

The next iteration of the element pattern included expanding the NEC simulation to investigate mutual coupling's effect on an AEU pattern in the panel. If mutual coupling had an effect on the individual pattern of an AEU, it would be discernable in the differences computed between element patterns for each AEU. The method used to achieve this model was to recreate a panel with 32 AEUs in NEC. Each AEU would be excited individually and its respective element pattern would be computed. If the computed pattern differed amongst the AEUs, it would conclusively demonstrate that mutual coupling affects the AEU element pattern.

The results demonstrated that the element pattern varied amongst each AEU in the panel. Thus mutual coupling did affect AEU pattern. The 3D and 2D plots representing the average element pattern calculated from each of the 32 individually computed element patterns is demonstrated in Figure 30 and Figure 31, respectively. The highest magnitude of radiation for this model does not occur at boresight, as it did with the Hertzian and isolated AEU models, but instead at an elevation around 65°. The RF engineers at SRI designed an electrical system in the AEU which would compensate for not using quarter wavelength spacing. This creates a dimple at boresight which widens the width of the element pattern. This ultimately increases the operation range of steering the AMISR's main lobe at the cost of reducing the maximum magnitude of the element pattern [16]. Figure 32 shows a cut of the 3D element pattern demonstrating the dimple and widening of the pattern as compared to Figure 29.

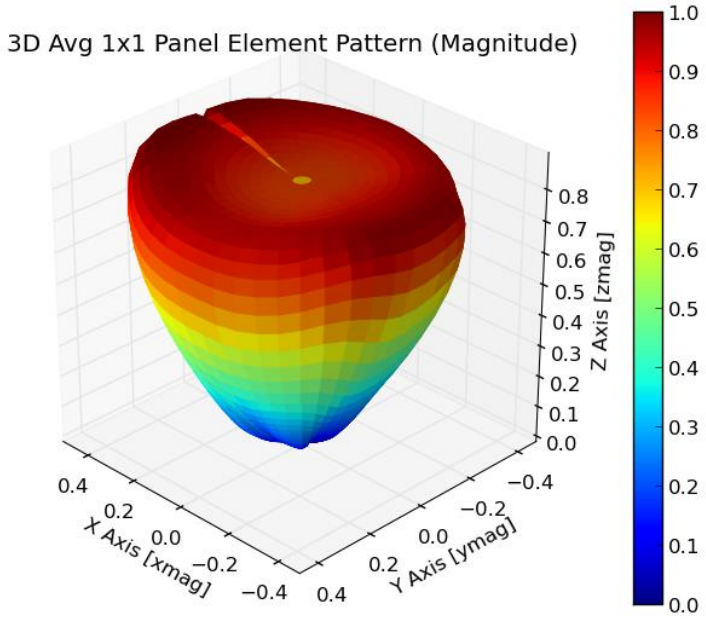

Figure 30-3D Representation of the Averaged Element Pattern of a 1Panel Array Using

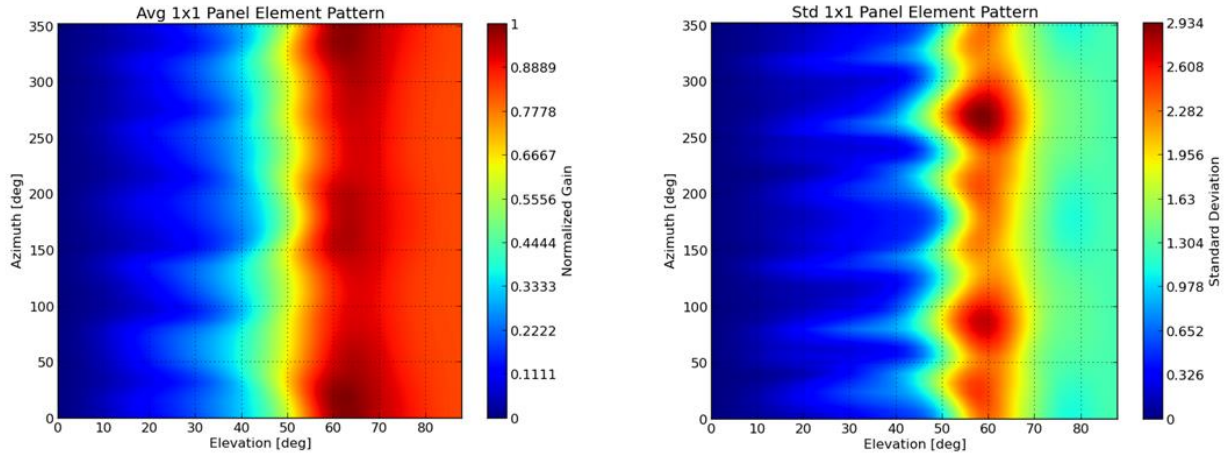

Figure 31-2D Representation of the Averaged Element Pattern of a 1x1 Panel Array and its Standard Deviation

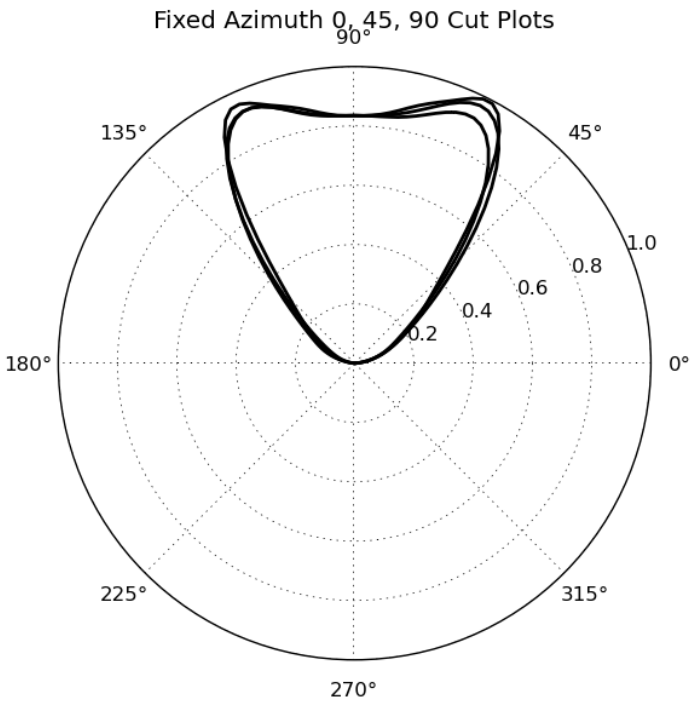

Figure 32-2D Slice of the 1x1 Panel Average Element Pattern

In order to see if the AEU average could be further refined, the 1 panel method for producing an average AEU element pattern was extended to a 1x3 and 3x3 panel model. Figure 33 is the 3D representation of the average element pattern computed for a 3x3 Panel. It is the average computed element pattern representing 288 individual AEU element patterns. The element pattern very closely represents that which is depicted in Figure 30. Its respective 2D average pattern and corresponding standard deviation are depicted in Figure 34.

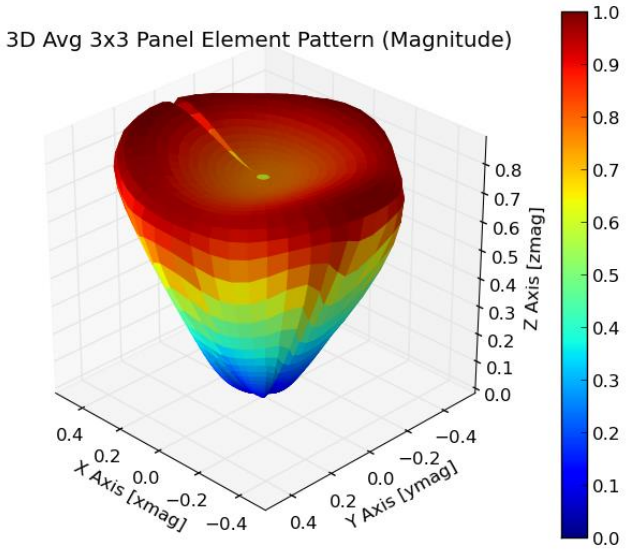

Figure 33-3D Representation of the Averaged Element Pattern of a 3x3Panel Array

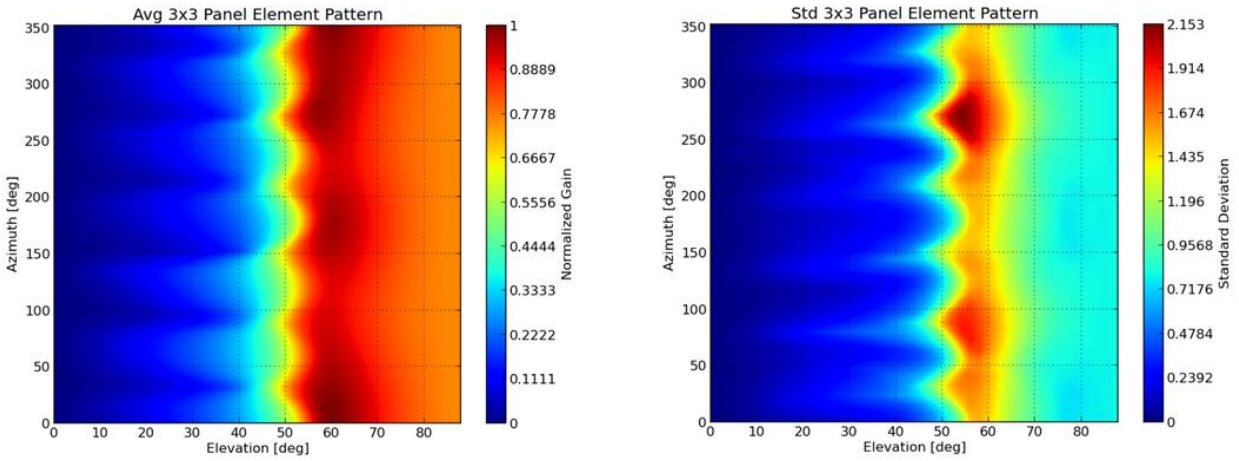

Figure 34-2D Representation of the Averaged Element Pattern of a 3x3 Panel Array and its Standard Deviation

The entire purpose of investigating the Hertzian dipole was to use it as a baseline for comparing the accuracy of each of the four NEC simulations previously described. The results are shown in Figure 35 below, as well as being numerically represented in **Error! Reference source not found.**. The isolated AEU has the lowest standard deviation when compared to the Hertzian and isotropic models. This was the result expected, as the effect of mutual coupling is not incorporated in either the Hertzian dipole or isotropic models. The purpose of the data represented in Figure 35 and Table 4 was to show that as the number of AEUs being modeled increased, the overall deviation from the baseline model(s) did not change significantly between the 1 panel, 1x3 panel, and 3x3 panel models. What did change, however, was the accuracy of the average models of each simulation compared to its respective samples. This information can be seen in the standard deviation contour plots shown in Figure 28, Figure 31, and Figure 34. As the number of AEUs being modeled increased, the standard deviation of the average model for each computed point decreased (with diminishing returns). As such, it was decided that the average element pattern representing the 3x3 panel model would suffice for an accurate representation of the element pattern for each AEU on the face of a full AMISR array.

The computed average element pattern representing the 3x3 panel array was used in the overall array radiation pattern calculation. An investigation of the array radiation pattern is presented in Section 4.3.

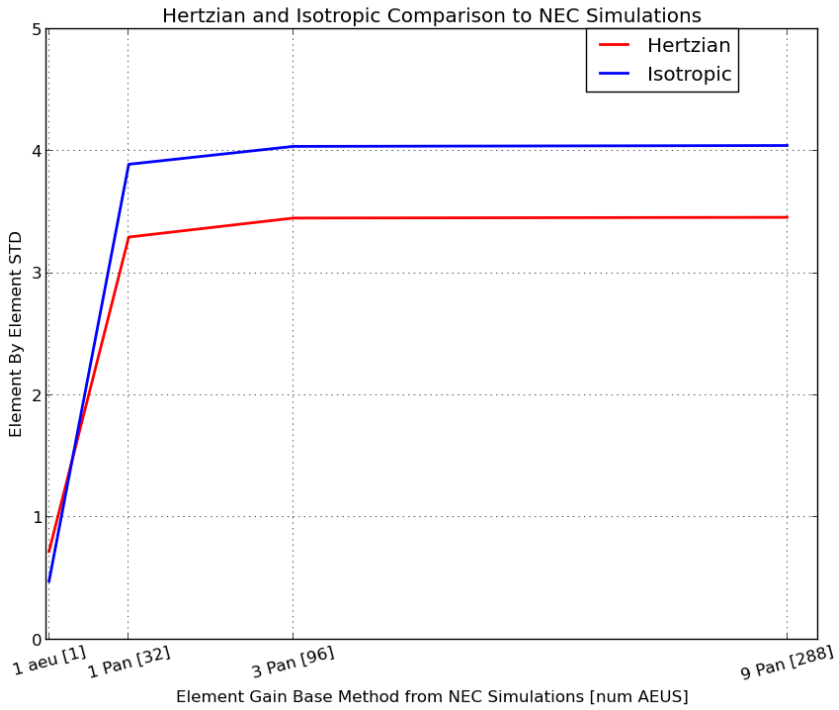

Figure 35-Comparison of the STD of the Various NEC Model Simulations Compared to the Hertzian and Isotropic Models

Table 4-Standard Deviation of the Various Average Models versus the Isotropic and Hertzian Models of a Cross Dipole

	num AEUs	Vs. Hertzian [Std]	Vs. Isotropic [std]
1 AEU	1	0.72234636	0.47818727
1 Pan	32	3.29700967	3.89287224
3 Pan	96	3.45248988	4.03932733
9 Pan	288	3.45847678	4.04661816

4.3 Objective 3: Validating the Radiation Pattern Calculation

Satellite Pass Test

Data files containing information about a satellite pass through PFISR's Grating Lobe Free Area were made available. These data files contain the measured SNR, ranges, and location of a satellite passing through the main lobe. The pass was traced successively with 5 different steering angles as the satellite passed through the Grating Lobe Free Area. Using the methodology described in Section 3.3, the theoretical SNR data was computed for each of the five steering angles. For each of the five beams, an average range value was used in [Eq. 4] to approximate the results. Furthermore, the system variables were approximated to the following values:

Table 5-Values Used to Calculate SNR

Parameter	Value	Units
c	$0.12^2\pi$	m^2
lam	0.67	m
Rtx	$\sim 1200*10^3$	m
Rrx	$\sim 1200*10^3$	m
B	$1.0*10^6$	Hz
Tsys	300	K
Gtx	transmit gain	dB
Grx	receive gain	dB
Pt	$200*10^6$	W
Ncoh	1000	

The passing of a satellite through one directed beam is shown in Figure 36 below. For each discrete point of its trajectory through the beam (shown as white dots), an interpolated magnitude representing gain is recorded. The corresponding "cut line" is shown in Figure 37. The range of this theoretical gain information was over the directional-sine angles (-9.2, 9.7) to (-3.3,8.5) each of which were associated with time as the object passed through the radiation pattern.

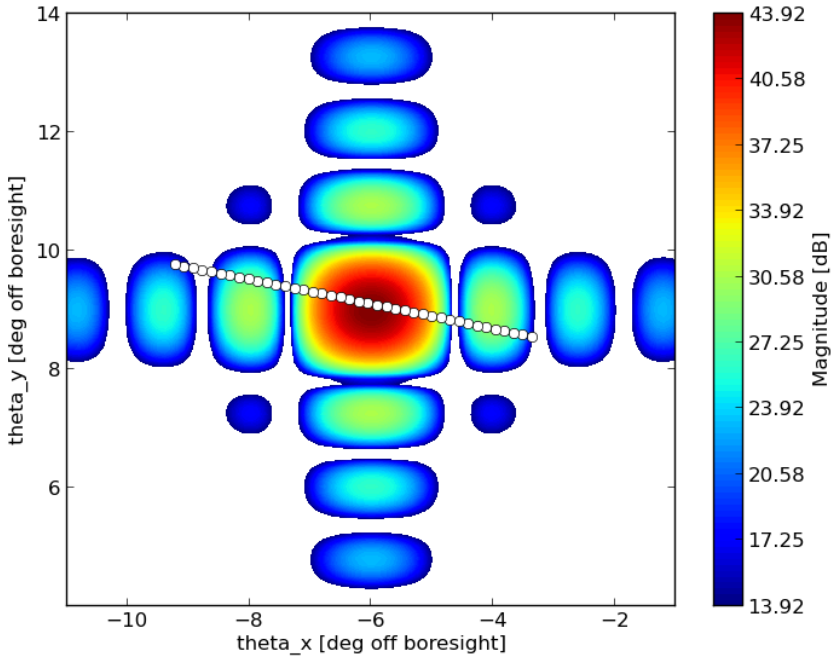

Figure 36-Satellite Passing through One of the Five Radiation Patterns

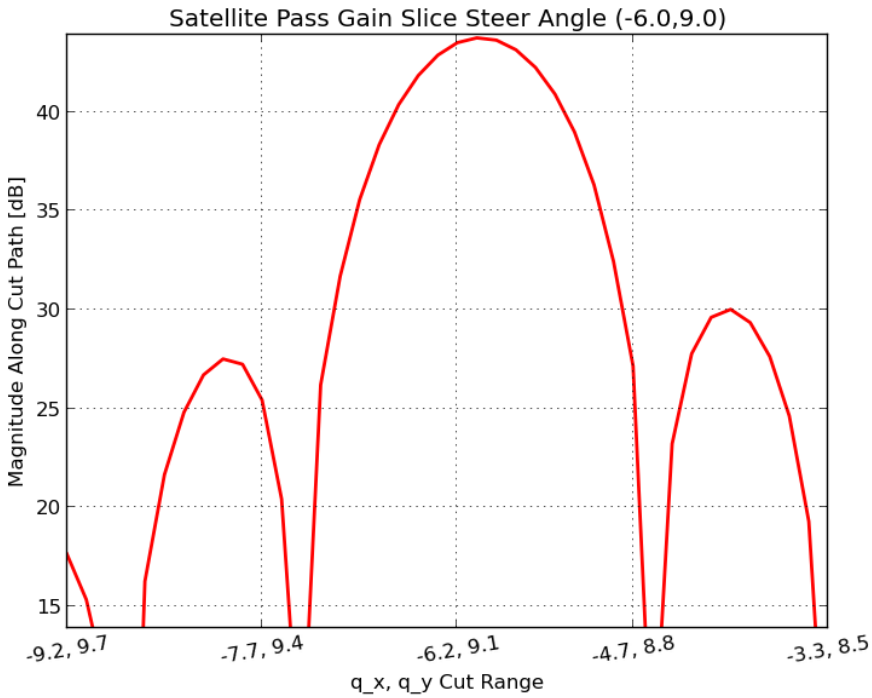

Figure 37-Cut Line Representing the Recorded Gain of a Satellite Pass

The plot in Figure 38 showed the first look at a comparison from the group's model to the actual received SNR data. Although there were many approximations used within the SNR equation and a difference in noise filtering, the plots line up within a reasonable degree of accuracy.

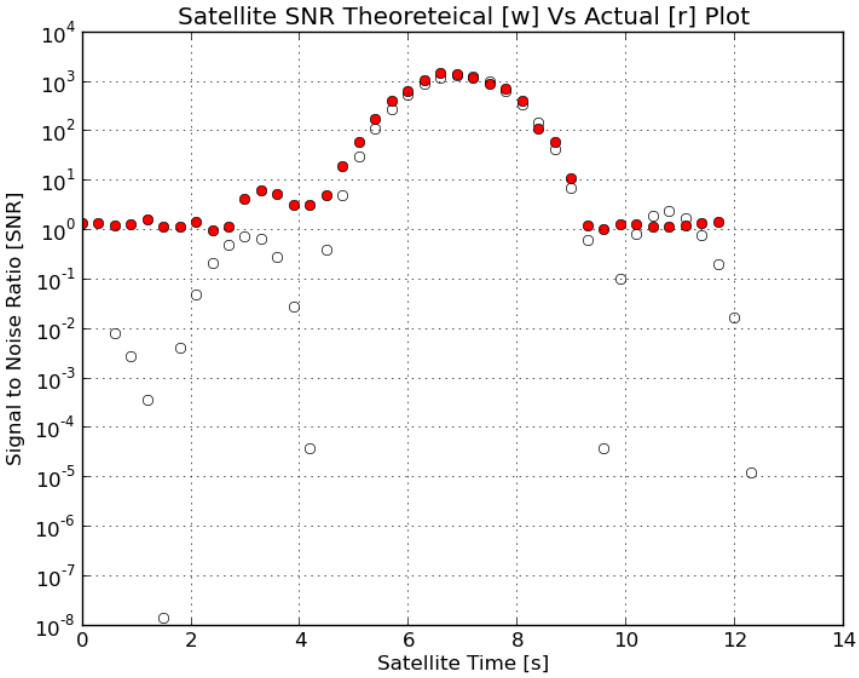

Figure 38-Calculated SNR for the Specified Pass

Figure 38 shows two sets of data overlaid on top of one another. The white markers represent the theoretical calculated SNR and the red markers show the actual measured SNR. Note that the measured SNR has a noise floor around 10 dB, meaning any signal below this line was lost in ambient noise in the backscatter gain. Furthermore, a small side lobe in the SNR data can be seen in the 2-4 second time frame. This lines up well with the position of the theoretical SNR side lobe in the same time frame.

An additional factor that was discovered was that the trajectory information and pass time information has a degree of inaccuracy as the satellite's orbit changes slightly year round. This introduced a time delay between the two peaks of the actual SNR and theoretical SNR data. In order to account for the discrepancy, which resulted in the above graph, a correlation factor was computed to quantify a time shift to a best fit. After reviewing the data plots it was necessary to create statistical comparisons between the two.

Since there were many sources of error in this beginning model it was most important to compare two characteristics of each data set; their signal width and maximum peak value. The signal width was determined as half power from the local peak of the signal, and the maximum peak value being the maximum value of the local signal. Table 6 summarizes the statistical fit properties of all the five passes. Note that the above satellite plot images correspond to Beam 5.

Table 6-Computed Values for the Satellite Pass

Beam	Calculated BW [s]	Measured BW [s]	Percent Error	Max Calculated Gain	Max Measured Gain
1	5.01819387108	4.73940515037	0.059	22793.7905893	14897.2851168
2	3.3818248113	3.50606862222	0.035	24868.8571144	29892.7612335
3	2.64545973142	2.96667391122	0.108	23603.0717693	40078.4086873
4	2.34545976465	2.60606681939	0.100	23738.2812379	25669.695111
5	2.50909560618	2.48182282785	0.011	24867.7022585	13145.2164142

Table 6 shows a method to compare the simulated and measured results of the satellite pass test. As a proof of concept for code functionality, SNR computation, and system modeling, these proved to be acceptable results. A maximum percent error of 10% was found on two of the beams which imply a slight inconsistency with the model. These inconsistencies may be inherent in the model, but they could come from calibration issues, or inaccurate measurements. In some cases, this error was as low as 3% and in others, 1% off.

After comparing the results of the ideal array's theoretical and measured SNR data, the same satellite pass test was conducted with sampled PFISR data. The model of the radiation pattern corresponding to the status of each AEU in the PFISR is depicted in Figure 39. This pattern will be confirmed if the theoretical SNR matches the measured SNR for its respective pass. The cut line corresponding to the satellite passing through the radiation pattern for this beam is shown in Figure 40.

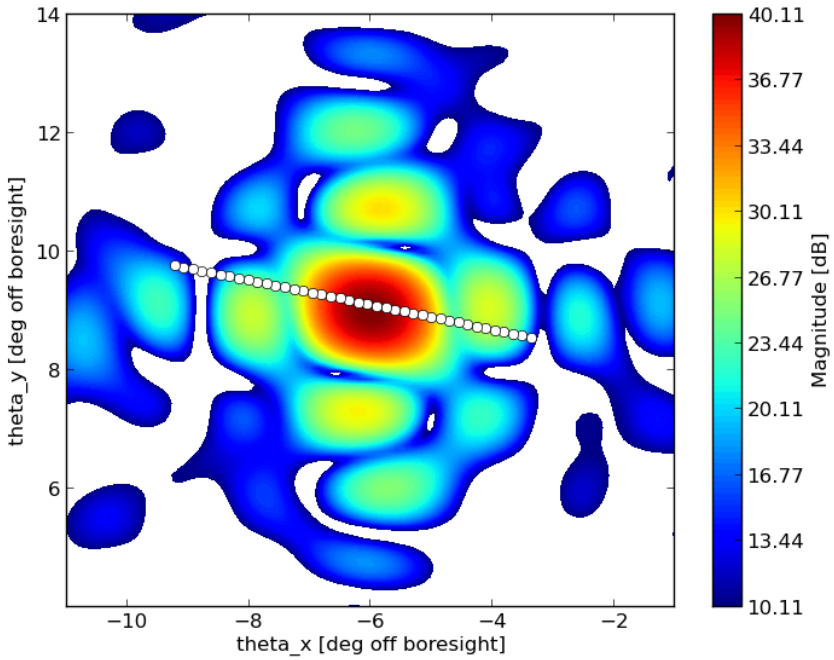

Figure 39-PFISR Radiation Pattern at the Time of the Satellite Pass

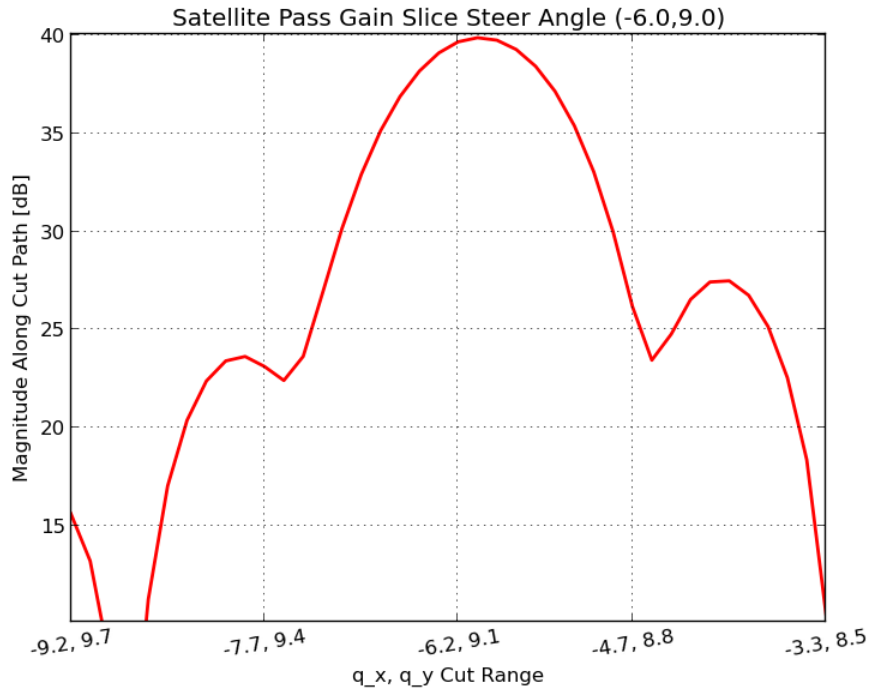

Figure 40-Cut Line Representing the Recorded Gain of a Satellite Pass

The calculated SNR corresponding to the recorded gain for this satellite pass, and the measured SNR data is displayed in Figure 41. Again, there is a nice visual representation of the noise floor at an SNR around 10^0. Additionally, the effect of a side lobe is seen at around 3 seconds for both SNR plots.

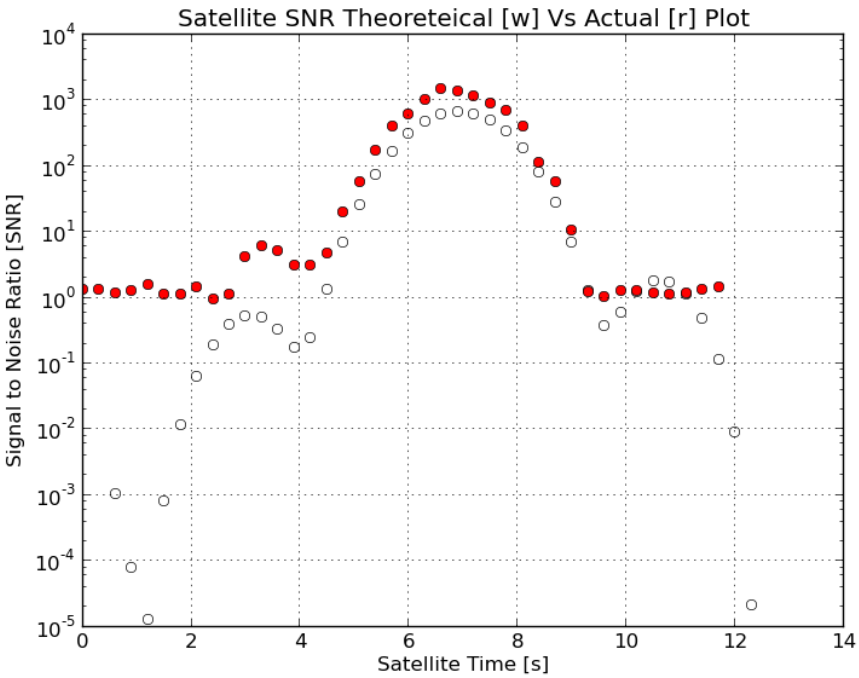

Figure 41-Satellite SNR Comparison Incorporating PFISR Status Data

Table 7-Computed Values for the Satellite Pass with PFISR Status Data

Beam	Calculated BW [s]	Measured BW [s]	Percent Err	Max Calculated Gain	Max Measured Gain
1	5.239404941	4.73940515	0.105	12077.31264	8588.616848
2	3.506067765	3.506068622	2.45E-007	11879.90388	17233.843
3	2.733339021	2.966673911	0.0786	10045.71439	23106.09573
4	2.424248011	2.606066819	0.0698	9797.306415	14799.15127
5	2.618186719	2.481822828	0.0549	11973.54901	7578.510197

When compared to the data to Table 6, the data shown in Table 7 demonstrates that by incorporating PFISR status data, the accuracy of the model increases. The results demonstrate that the unknown coefficients such as the number of coherent integrations, system temperature or range could be the potential sources of error in the SNR calculation. As such, the deviation between the calculated and measured data does not necessarily reflect errors in the array radiation pattern calculation.

Section 5: Deliverables

Final Report

The first deliverable of the work is this report. It encapsulates all relevant information needed for a successful completion of the project. Additionally, it provides the methodology used for achieving the results and the impact that those results had.

User Manual

The user manual can be found in Appendix B. In it, one can find an extensive description of the class structure used to replicate the modular design of the AMISR. Additionally, descriptions of each method, specific to a class or library, are included. These descriptions contain information about the input needed for the method to function and the output that is computed. Instructions for setting up the Python platform are included. This includes the interpreter used for execution and any additional packages needed for the code to be interpreted.

Software Package

The software package was handed off to SRI's Geospace Division for further iterations of the model and any bug corrections. The package was developed to be as robust as possible and include any method thought to be of interest to the Geospace Division.

Section 6: Conclusion and Recommendations

The developed software package is able to model the radiation pattern for any AMISR system, while incorporating realistic factors like phase error, mutual coupling, and the inconsistent transmit power antenna elements. While many concepts were proven in the creation of the AMISR radiation pattern model, there are many options for continuation of this project. One option involves NEC modeling considerations, and the other involves the usability of the software package.

More factors can be incorporated into the NEC model of the antenna. The Cross-Dipole radiates in all directions. It is possible that the ground plane (the caging that holds the panels on the array) may reflect radiation from the opposite direction back to the modeled direction. Including the conductivity of the ground plane in the NEC model may account for this reflection, and increase the practicality of the model.

It may be useful to investigate the difference between using an average radiation pattern for all elements and distinguishing between internal and external antenna elements. The effect of mutual coupling affects the radiation pattern of antennas differently depending on their location relative to other antennas. The one possible distinction is between internal elements (antennas completely surrounded by other antennas) and external elements (antennas at an edge of the panel). Depending on how drastic a difference there is between the two, incorporating the radiation pattern for external elements in combination with the radiation pattern for internal elements may augment the realism of the model.

In addition to modeling considerations, the usability of the software package can be improved by developing methods that add convenience for the user. One feature that would add convenience would be a saving function. This function would give the user the ability to save a specific AMISR configuration as well as its radiation pattern. In doing so, the user will be able to load the configuration and radiation pattern at a later time to compare them to other configurations. This makes the ability to compare the performance of multiple AMISR systems much easier.

A graphical user interface would make all the methods readily available, while also intuitively connecting the methods to the related data visualization technique. For instance, after plotting the active elements, the fields of each antenna element could be edited by clicking on the element in the plot. Overall, this graphical user interface would further simplify the use of the software package, while keeping all of the functionality intact. While there are many opportunities for further improvements, the software package is robust enough to be applied to all AMISR systems, and can easily be used by SRI International for future upper atmospheric research.

References

[1] Wikipedia, “SRI International”. [Accessed by web: 12/14/2013] http://en.wikipedia.org/wiki/SRI_International

[2] SRI International, “AMISR”. [Accessed by web 12/14/2013] www.amisr.com/amisr

[3] Halliday, David. Resnick, Robert. Walker, Jearl. 2013. *Fundamentals of Physics* 10th Edition.

[4] Ulaby, Fawwaz. Michielssen, Eric. Ravaioli, Umberto. 2010. *Fundamentals of Applied Electromagnetics* 6th Edition.

[5] Hughes, Eli. “What is wavenumber?” [Accessed by web 12/14/2013] http://www.youtube.com/watch?v=YMW8CcnDp7A)

[6] IEEE Standards Committee of the IEEE Antennas and Propagation Society. *IEEE Standard Definitions of Terms for Antennas*. [Accessed by web 12/14/2013] http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=286098

[7] Fusco, Vincent F. *Foundations of Antenna Theory and Techniques*. 2005.

[8] COMSOL. “Modeling a Dipole Antenna” [Accessed by web 12/3/2013] http://www.comsol.com/model/modeling-a-dipole-antenna-8715?utm_source=BOS+database+%2812%2F05+IEEE+Antenna+webinar%2C+follow-up+%28model+gallery%29&utm_campaign=us_webinars_dec13&utm_medium=Demail&utm_content=6

[9] MIT Lincoln Labs. “Introduction to Radar Systems”. [Accessed by web 11/10/2013] https://www.ll.mit.edu/workshops/education/videocourses/introradar/

[10] Institute for Geophysics, University of Texas. “How Radar Works”. [Accessed by web 12/13/2013] http://www.ig.utexas.edu/research/projects/mars/education/radar_works.htm

[11] Federation of American Scientists. “Radar Systems”. [Accessed by web 12/14/2013] http://www.fas.org/man/dod-101/navy/docs/es310/radarsys/radarsys.htm

[12] Nicolls, Michael. Email Correspondence (See Appendix B).

[13] Antenna-Theory. “Antenna Arrays (Phased Arrays)”. [Accessed by web 11/20/2013]. http://www.antenna-theory.com/arrays/main.php

[14] Kelley, David F. et al. *Array Antenna Pattern Modeling Methods That Include Mutual Coupling Effects*. 1993.

[15] Ofiandis, Sophocles J. *Electromagnetic Waves and Antennas Ch. 18*. Rutgers ©2008. [Accessed by web 2/3/2014]. http://eceweb1.rutgers.edu/~orfanidi/ewa/

[16] Nicolls, Michael. PhD., 2013. “Intro Phased Array Radar”, PowerPoint [Accessed by Web 3/3/2014]. https://www.dropbox.com/s/nb8g5g3dcbpgd2v/Intro%20Phased%20Array%20Radars.ppt

[17]Fleisch, Daniel. *A Student’s Guide to Maxwell’s Equations*. Cambridge Press © 2007. [Accessed by web 2/20/2014]. http://www4.wittenberg.edu/maxwell/CoordinateSystemReview.pdf

[18] Python.org, “Tutorial: Floating Point”. [Accessed by web 3/4/2014]. http://docs.python.org/2/tutorial/floatingpoint.html

[19] The MathWorks. “MATLAB Programming: Floating Point Numbers”. [Accessed by web 3/4/2014]. http://www.mathworks.com/help/matlab/matlab_prog/floating-point-numbers.html#f2-108458

[20] Fitzpatrick, Richard. “The Hertzian Dipole”. [Accessed by web 3/4/2014].
http://farside.ph.utexas.edu/teaching/em/lectures/node94.html
[21] Sparks, Jonathan. “Determination of physical and radiant meteor properties using PFISR Interferometry measurements of head echoes”. Journal of Atmospheric and Solar-Terrestrial Physics. August 11, 2010

Appendices

Appendix A: Email Correspondance with Dr. Nicolls

"Hi Landon, Patrick, and Jarred,

I wanted to get back to you with some additional information and responses to your questions.

1) When you say "health" of each of the 4096 elements, what specifically are you referring to? Is it the power out of each?

The AMISR system consists of 4096 antenna element units. We measure things like the current, forward and reflected power, voltage, etc. of each of the elements. The primary criteria here will be (a) is the element on (is it receiving E&M waves) (b) is it transmitting and (c) how much power is it transmitting. From this info, we can compute the gain on transmit and receive, and the total backscatter gain.

2) Is backscatter gain the phenomenon of a transmitted signal's strength when it is received after bouncing off of the ionosphere (or an object) and returning to the radar?

Gain refers to the directivity (and efficiency) of an antenna - a large gain means that the antenna is very directive, or converts most of its input E&M waves to power in a specific direction. Backscatter gain is useful for us because we detect volume scatter from all over the sky. The total received power is a weighted function of the power from individual volume elements, with the weighting determined by the backscatter gain. Backscatter gain is the average value of Grx times Gtx where Grx is the gain on receive and Gtx is the gain on transmit. For many applications, one can assume these are equal; for us, we may have a narrower beam on receive because of non-transmitting parts of the array. I've attached a paper that describes backscatter gain more rigorously. Take a look at the Introduction and let me know if you have any questions.

3) One of our objectives lists software development to predict beam patterns. We assume that there are currently software packages available to model parts of the AMISR project. To what extent will we be incorporating Python with the software packages?

We are working on putting together some basic software in Python to start you off. This code will allow you to turn on and off specific elements and predict the beam pattern of the array. We are hoping you will develop this software further, and compare to observations. We are also looking at more comprehensive electromagnetic modeling software to simulate the entire array, including mutual coupling effects.

4) What deliverables do you expect from our team throughout our time? We are unsure of how to prioritize our objectives. And overall, what is the expected result/ outcome of these 8 weeks that we will present to SRI International?

In the next couple of weeks, we'll put together a more detailed plan and schedule. The main components of this project will be:
(a) writing software to model the array gain
(b) data analysis of calibration spheres that fly through the beam pattern to compare model and observation
(c) analysis of many sphere passes to build up a picture of the far field radiation pattern of the array
(d) analysis of radio-star tracks to determine the receive gain of the array
** (e) potentially modeling the array in a more comprehensive electromagnetics code.
One goal of this project is to actually measure the far-field radiation pattern of our antenna, and show that we understand it.
Another is to provide a metric for calibration - when we lose portions of the array to failures, we'd like to predict in a systematic way the effect on the beam pattern and backscatter gain. We are hoping the results can provide us with this information.

Regarding resources and computing, you will be working in our offices in Menlo Park in the G building on SRI's campus. We will provide you with a computer to use while you are here, which we hope to set up with Python and

other tools that you need. Because we will be mainly using Python, it seems easiest to have you use a Linux machine - do you have any objections to that?

Does Friday the 6th, 3 PM PST still work for an initial telecon?

Thanks,
Mike”

Appendix B: AMISR Modeling Software Package User Manual

User Manual Presented by:
Landon Airey
Patrick Sullivan
Jarred Velazquez

To:
SRI International's Geospace Division

Set Up and Initialization

The software was written in Python and interpreted using Python 2.7.3.
The organization of the code represents the modular setup of the AMISR.
There is a class for: the Face, the Array, the Panel and the AEU. Object(s) of each class, depending on the configuration settings specific to the AMISR being modeled and established in the .ini files, are used as dictionaries. When a Face is finally initialized with all relevant data, it becomes the input of the Gain class. Figure 42**Error! Reference source not found.Error! Reference source not found.** shows the hierarchical flow of the code.

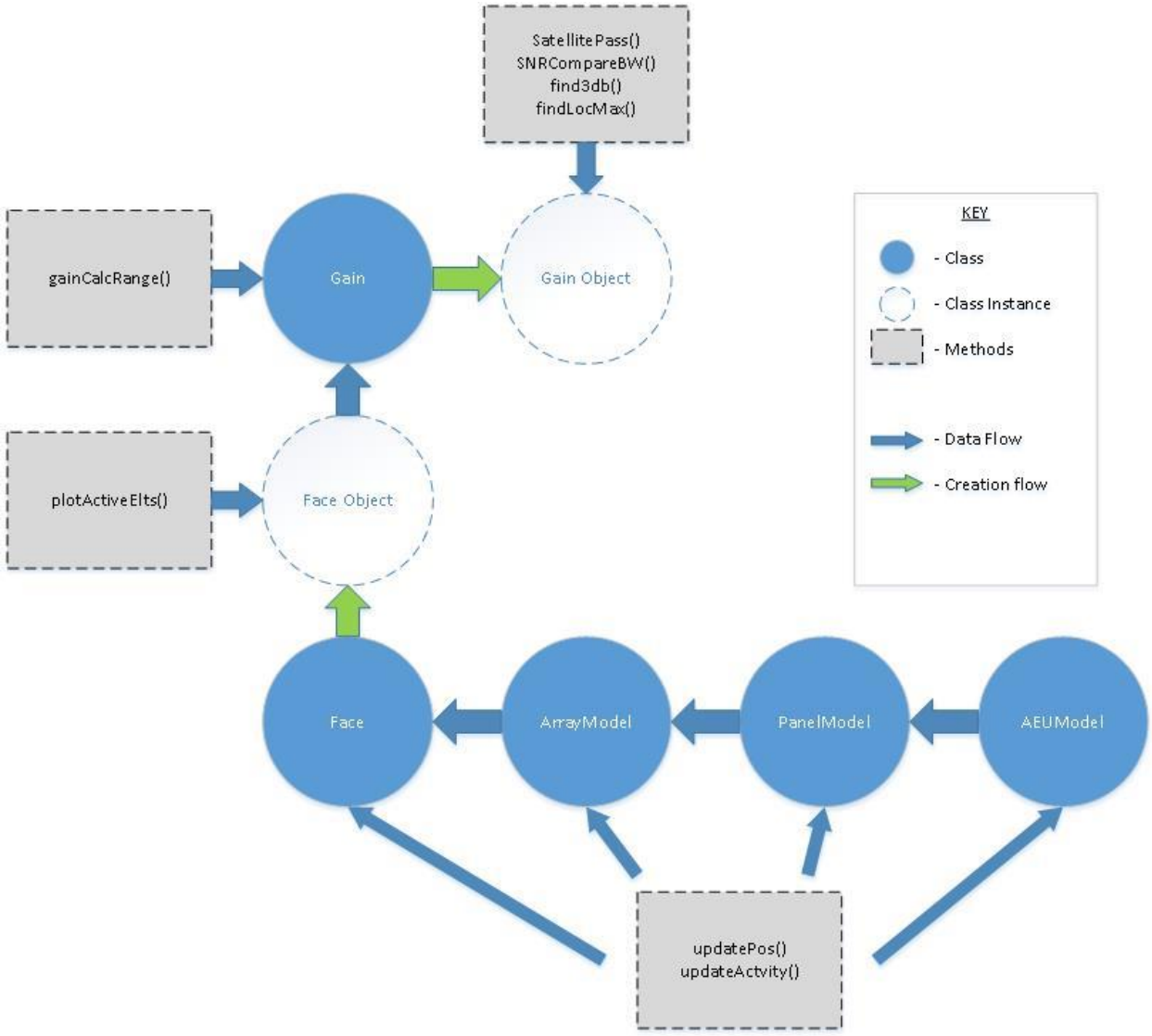

Figure 42-Flow Chart of the AMISR Modeling Package

Imported Packages

The following Python packages are imported into classes.

Package	Application
sys	This is used for path directing and system specific parameters and functions.
logging	Logs run time information.
LoggerInit.LoggerInit	Keeps track of code processes for the purpose of future debugging.
ConfigReader.ConfigReader	Reads in information from the configuration files.
matplotlib.pyplot	Contains plotting tools for visual representations.
numpy	Contains scientific computing tools.
scipy	Contains scientific computing tools.
time	Contains functions to record the time it takes for software processes.
re	Regular expression matching operations.
random	Generates random values.
tables	Pulls information from HDF to be stored into table format.
PlotTools*	Plotting techniques specific to AMISR modeling.
CoordConv*	Coordinate conversion techniques specific to AMISR modeling.

*Produced by the WPI team working with SRI.

Class-Specific Methods

All classes are initialized using the "Initialize" method. Generally, the Initialize() pulls data from the .ini files stored within the config folder of ArrayModeling. Initialize() is used to update various fields with other methods specific to that class. Additionally, Initialize() cascades through the hierarchy of the code to produce the dictionaries relative to each class as shown in Figure 42.

Gain Class

The gain class is used to calculate the gain of a given AMISR object.

Initialize()

Inputs: N/A
Outputs: N/A
Summary: The method will initialize the Gain class by pulling the element gain data from .npy files and the dy, dx element spacing from the Panel class.

gainCalcRange(steerAngles, elazFlag, degFlag, dth, tht, trunc, phaseErr)

Inputs: steerAngles (list of tuples, tupleOrder: (thx, thy))
elazFlag (True/False)
degFlag (True/False)
dth (float)
tht (float)
trunc (int)
phaseErr(int)
Outputs: gain (array)
txGain (array)
rxGain (array)
thxr (array)
thyr (array)
Summary: This method computes the radiation pattern of an array input into the gain class. It requires a either a tuple (or list of tuples) corresponding to the steer angle (or list of steer angles) which the user wishes to compute the gain for. The user denotes whether the angle is input as an Azimuth-Elevation pair (elazFlag) and if it is in degrees (degFlag). The computed angular resolution is defined by dth. The viewing window is 2*tht. The variables trunc and phaseErr are used for incorporating the effect of phase error due to the AMISR's hardware and software PCU limitations. For a n steering angles, arrays of length n are computed for the arrays: gain, txGain, rxGain, thxr, thyr. Gain is the array which contains the radiation pattern of a given Face object in a specified look angle. Thxr, thyr are the meshgrid arrays needed to visually represent the computed gain. This method calls gainCalc() n times.

gainCalc (anglePair, elazFlag, degFlag, dth, tht, trunc, phaseErr)

Inputs: steerAngles (tuple: (thx, thy))
elazFlag (boolean)
degFlag (boolean)
dth (float)
tht (float)
trunc (int)
phaseErr(int)
Outputs: gain (array)
txGain (array)

rxGain (array)
thxr (array)
thyr (array)

Summary: This method computes the radiation pattern [in dB] of an Face object input into the gain class. If elaz is true, the angle pair is converted to Directional Sine coordinate system. Gaindb, txGain, rxGain, thxr and thyr are returned.

find3dB(Garrdb, thxr, thyr, steerAngle, degFlag)

Inputs: Garrdb (array returned from either gainCalc ()or gainCalcRange())
Thxr (meshgrid array)
Thyr (meshgrid array)
steerAngle (tuple)
degFlag (Boolean)

Outputs: gain (array)
txGain (array)
rxGain (array)
thxr (array)
thyr (array)

Summary: Given a radiation pattern array (Garrdb), its meshgrid arrays (thxr, thyr), and steerAngle, the main lobe is found. The steerAngle can be input as either degrees or radians. If degFlag is true, the angle is converted from degrees to radians. The values returned are the RMS radius (int) of the main lobes half power beam width, distanceRange and angle. A plot is generated which plots a dotted line of the area encapsulating the half power beam width of the main lobe.

findLocalMax(G, thxr, thyr,magFlag)

Inputs: G (array returned from either gainCalc ()or gainCalcRange())
Thxr (meshgrid array)
Thyr (meshgrid array)
magFlag (boolean)

Outputs: maxGinfo (list)

Summary: The local maxima of the first side lobes of a radiation pattern array's (G) are found. Their locations and magnitude are passed to findLocMax_Contour() as well as magFlag which will center the magnitude's value on its respective lobe.

interpolateElementGainLookup(elementGainLookup, theta, phi)

Inputs: elementGainLookup (array)
theta (float)
phi (float)

Outputs: elementGain

Summary: The element gain for a given theta, phi combination is interpolated from the elementGainLookup table which is loaded in Initialize() as a numpy array from the NEC modeling simulation files.

satellitePass(dth, tht, trunk, phaseErr)

Inputs: dth (float)
tht (float)
trunk (int)
phaseErr (int)
Outputs: compareStats (array)
Summary: This method reads in orbit pass data using OrbitReading(). The relevant information is taken from the files produced to replicate a satellite pass and compute the SNR based off of gain calculated with gainCalc. Necessary substeps have been explained within the method. A visual representation of the satellite pass is produced for each direction the beam is pointed in in order to track the satellite. compareStats, an array, is returned which contains measured and calculated: time stamps, percent errors and maximum calculated SNR. compareStats is an array of length n. n corresponds to the number of discrete steerings of the beam to track the satellite. satellitePass calls SNRCompareBW() to calculate the difference in values between measured and calculated values.

SNRCompareBW(calcGain, MeasuredSNR, BeamTime, Range)

Inputs: calcGain (list)
MeasuredSNR(list)
BeamTime (list)
Range (list)
Outputs: theoreticalTime, actualTime, percentErr, maxCalc, maxMeas
Summary: This is a helper function used to calculate the difference between the calculated SNR data and the measured SNR data. This method calls smooth(). Smooth is used to filter an array so that it can be interpolated more smoothly.

smooth(x)

Inputs: x (list)
Outputs: x (list)
Summary: This method will refit a data sample by "smoothing" the data within it. The method was developed by another and further documentation can be found at wiki.scipy.org/Cookbook/SignalSmooth.

Face Class

The Face class selects the specific AMISR face being modeled (e.g. "PFISR", "RISR-N", "RISR-C") and creates a dictionary of type ArrayModel.

Initialize()

Inputs: N/A
Outputs: N/A
Summary: The method will initialize the Face class by creating the dictionary of ArrayModels. This calls ArrayModel's method Initialize(), self.updatePos(), self.updateActivity, self.updateJP_id().

updatePos()

Inputs: N/A
Outputs: N/A
Summary: The method will update the coordinate position of the Face "radar" based on the .ini configuration file. This method calls ArrayModel's method updatePos().

updateActivity ()

Inputs: stat (integer 1 or 0)
Outputs: N/A
Summary: The method will update the Tx and Rx fields in the AEU class based on the input status field. This method calls ArrayModel's method updateActivity().

updateJP_id()

Inputs: N/A
Outputs: N/A
Summary: The method will update the ID fields of the ArrayModel class.

plotPan (xcoord, ycoord)

Inputs: xcoord, ycoord (integers)
Outputs: N/A
Summary: The method will plot the position of the AEUs of a specific panel (selected by x/ycoord). This calls ArrayModel's method plotPan().

plotActiveElts ()

Inputs: N/A
Outputs: N/A
Summary: The method will plot the position of the AEUs of the entire face "radar". Each AEU will have a color associated to it relative to the Tx/Rx power set to it from either the .ini configuration or .hdf data (taken from the AMISR using StatusPullRISR()). This method calls on PlotTools* plotActiveEltsHelp() function.

printStatus ()

Inputs: N/A
Outputs: N/A
Summary: The method will print which Face has been created.

StatusPullRISR ()

Inputs: N/A
Outputs: N/A
Summary: The method will update the Tx and Rx fields in the AEU class based on the value associated to it in the .hdf data files. This method calls ArrayModel's StatusPullRISR().

ArrayModel Class

The ArrayModel class selects the array information specific to the AMISR face being modeled.

Initialize()

Inputs: N/A
Outputs: N/A
Summary: The method will initialize the ArrayModel class by creating dictionaries of PanelModel's. This calls PanelModel's method Initialize() for each Panel, specified by Row and Column location, of the array.

updatePos()

Inputs: N/A
Outputs: N/A
Summary: The method will update the coordinate position of the specific panel based on the .ini configuration file. This method calls PanelModel's method updatePos().

updateActivity(stat)

Inputs: stat (integer 1 or 0)
Outputs: N/A
Summary: The method will update the Tx and Rx fields in the AEU class based on the input status field and the jetStatus field in the .ini file corresponding to the specific panel. This method calls PanelModel's method updateActivity().

updateJP_id()

Inputs: N/A
Outputs: N/A
Summary: The method will update the ID fields of the PanelModel class. It selects the jet power associated, "ind", with the specific panel (classified by row and column) and calls on PanelModel's updateJP_id(ind).

plotPan(xcoord, ycoord)

Inputs: xcoord, ycoord (integers)
Outputs: N/A
Summary: The method will plot the position of the AEUs of a specific panel (selected by x/ycoord). This calls PanelModel's method plotPan().

randomOff(numPans, numAEUs)

Inputs: numPans, numAEUs (integers)
Outputs: N/A
Summary: This method will randomly turn off the number of panels and/or AEUs input by the user. This calls PanelModel's updateActivity() to disable the amount of panels and AEUs specified as input.

StatusPullRISR()

Inputs: N/A
Outputs: N/A
Summary: This method pulls data from the h5 file that contains status data (AEU tx/rx power levels/statuses) of an AMISR array.

printStatus ()

Inputs: N/A
Outputs: N/A
Summary: The method will print which ArrayModel has been created.

PanelModel Class

The PanelModel class selects the Panel level AEU information to successfully model a specific AMISR face.

Initialize(row, column)

Inputs: row, column (integers)
Outputs: N/A
Summary: The method will initialize the AEUModel class by creating dictionaries of AEUModels. The values for the number of AEUs and dy-dx element spacing are pulled from the .ini file. This calls AEUModel's method Initialize() for each AEU, specified by element number in the panel.

updatePos()

Inputs: N/A
Outputs: N/A
Summary: The method will update the coordinate position of the specific AEU based on the configuration specifications (dy-dx). This method calls AEUlModel's method updatePos(x,y).

updateActivity(stat)

Inputs: stat (integer 1 or 0)
Outputs: N/A
Summary: The method will update the Tx and Rx fields in the AEU class based on the input status field. This method calls AEUModel's method updateActivity(stat).

updateTx(stat)

Inputs: stat (integer 1 or 0)
Outputs: N/A
Summary: The method will update the Tx field in the AEU class based on the input status field. This method calls AEUModel's method updateActivity(stat).

updateRx(stat)

Inputs: stat (integer 1 or 0)
Outputs: N/A
Summary: The method will update the Rx field in the AEU class based on the input status field. This method calls AEUModel's method updateActivity(stat).

randomOff(numAEU)

Inputs: numPans
Outputs: N/A
Summary: This method will randomly turn off the number AEUs input by the user. This calls AEUModel's updateActivity(stat) to disable the amount of panels and AEUs specified as input.

updateJP_id(id)

Inputs: N/A
Outputs: N/A
Summary: The method will update the ID fields of the PanelModel class.

plotPos()

Inputs: N/A
Outputs: N/A
Summary: This method will plot the position of the AEUs on the entire face of the radar.

plotPan()

Inputs: N/A
Outputs: N/A
Summary: The method will plot the position of the AEUs of a specific panel (selected by x/ycoord).

printStatus ()

Inputs: N/A
Outputs: N/A
Summary: The method will print which Panel has been created and how many AEUs it has.

AEUModel Class

The AEUModel class is the lowest level in the AMISR hierarchy.

Initialize(element)

Inputs: element (integer)
Outputs: N/A
Summary: The method will initialize the AEUModel.

updatePos(x,y)

Inputs: x,y (floats)
Outputs: N/A
Summary: The method will update the coordinate position of the specific AEU based on the configuration specifications (dy-dx).

updateActivity(stat)

Inputs: stat (integer 1 or 0)
Outputs: N/A
Summary: The method will update the Tx and Rx fields in the AEU class based on the input status field.

updateTx(stat)

Inputs: stat (integer 1 or 0)
Outputs: N/A
Summary: The method will update the Tx field in the AEU class based on the input status field.

updateRx(stat)

Inputs: stat (integer 1 or 0)
Outputs: N/A
Summary: The method will update the Rx field in the AEU class based on the input status field.

updateTxPower(power)

Inputs: stat (integer 1 or 0)
Outputs: N/A
Summary: The method will update the Tx field in the AEU class based on the input power field based on the hdf data.

updateAEUonPan(x,y)

Inputs: x,y (float)
Outputs: N/A
Summary: The method will update the AEU position relative to a panel.

printStatus ()

Inputs: N/A
Outputs: N/A
Summary: The method will print which AEU has been created.

Libraries Containing Helper Functions

CoordConv Library

elaz2Dir (elazPair, degFlag)

Inputs: elazPair (tuple or list of tuples)
degFlag (Boolean)
Outputs: dirPair (tuple or list of tuples)
Summary: The function will take an elevation-azimuth angle (deg or rad) and return a Directional-Sine thetaX, thetaY tuple pair in radians.

dir2elaz (dirPair, degFlag)

Inputs: dirPair (tuple or list of tuples)
degFlag (Boolean)
Outputs: elazPair (tuple or list of tuples)
Summary: Converts Directional-Sine thetaX, thetaY tuple pair to elevation , azimuth Polar-Coordinate pair in radians.

Geo2faceElAz(geoElAzPair, degFlag)

Inputs: geoElAzPair (tuple or list of tuples)
Outputs: faceElAzPair (tuple or list of tuples)
Summary: This function will convert geodetic (earth) relative points to PFISR-face relative points. This calls on Rotate_PFISR().

Rotate_PFISR ()

Inputs: N/A
Outputs: N/A
Summary: The method will produce the coefficients needed in the conversion from geodetic to face-local points. These coefficients are specific to the PFISR face.

PlotTools Library

plotActiveEltsHelp(numRows, numCols, normalized, dx, dy, yadj, x, y, xRemoved, yRemoved, colorvals)

Inputs: numRows (int)
numCols (int)
normalize()
dx (float)
dy (float)
yadj(float)
x (list)
y (list)
xRemoved (list)
yRemoved (list)
colorvals (array)

Outputs: N/A

Summary: The method will generate a plot of all AEUs in an AMISR. Each AEU is represented by a hexagon filled with a color corresponding to the power that that AEU is transmitting at.

makeHexagon(x0,y0, dx, c)

Inputs: x0 (float)
y0 (float)
dx (float)
c (string)

Outputs: hexagon (patch object of type hexagon for plotting)

Summary: The function will produce a correctly spaced and shaped hexagon which is plotted. This is a helper function.

gainContour (Garr, thxr, thyr, minimum, maximum, maskFlag)

Inputs: Garr (array)
thxr (array)
thyr (array)
minimum (int)
maximum (int)
maskFlag (Boolean)

Outputs: N/A

Summary: The method will produce a contour plot to illustrate dB gain in the form of color, corresponding to thx and thy coordinates. The colorbar range will be set to the min and max of the Garr unless specified by user. The maskFlag, if set to True, will mask the imaginary parts of Garr and will show up as white space in the contour map.

contourCutLine(dth, tht, thxr, thyr, thx0, thy0, alpha, Garr)

Inputs: dth (float)
tht (float)
thxr (array)
thyr (array)
thx0, thy0 (float)
alpha (float)
Garr (array)

Outputs: N/A

Summary: dth is the angular resolution in degrees, tht is the viewing window of the contour plot (-tht,tht in width). Thxr, thyr are the meshgrid coordinates. thx0, thy0 are the steering angle of the main lobe. alpha is the cut angle, Garr is the calculated gain array.

The user defines alpha which is the cutting angle through the origin (thx0, thy0). The points of intersection between this line and the masking circle of an imaginary boundary in the gain matrix is found using the cutLineFind() function. Those points of intersection are used to define x,y indices along the line. The corresponding magnitude along the line is found by interpolating through the Garr matric. The resulting gain data is plotted verses the thx, thy pairs and the corresponding contour plot with a superimposed line representing the cut is also plotted.

contourCutCustom (dth, tht, thxr, thyr, thxylist, Garr)

Inputs: dth (float)
tht (float)
thxr (array)
thyr (array)
thxylist (list)
Garr (array)
Outputs: Ginterp (array)
Summary: The function produces an interpolated gain array of a satellite pass. This information is used to track a satellite through the gain of an array and to visually represent that trajectory.

lineIntersectSquare(x0,y0, alpha, boxLen)

Inputs: x0 (float)
y0 (float)
alpha (float)
boxLen (int)
Outputs: coords (list of tuples)
Summary: This helper function finds the intersection points between a line and a square, where the line is defined by a point and an angle. The coords returned are the indices of the positions where the cut line intersects with the window of the contour plot.

lineIntersectCircle(x, y, theta, r, xorigin, yorigin)

Inputs: x (float)
y (float)
theta (float)
r (float)
xOrigin (float)
yOrigin (float)
Outputs: coords (list of tuples)
Summary: The helper function finds the intersection points between a line and a circle, where the line is defined by a point and an angle and the circle is defined by an origin point and radius. The coords returned are the indices of the positions where the cut line intersects with the circle.

satellitePass_SNRplot(BeamSNR, SatSNR, BeamTime, SatTime)

Inputs: BeamSNR (array)
SatSNR (array)
BeamTime (array)
SatTime (array)

Outputs: N/A
Summary: This method overlays SNR data for both the calculated model and actual collected data versus time.

find3db_Contour(thxr, thyr, Garrdb)

Inputs: thxr (array)
thyr (array)
Garrdb (array)
Outputs: CS (plot handle)
Summary: The function produces a contour plot representing the gain pattern of the radar. The main lobe half power beam width is illustrated by the dotted line.

findLocMax_Contour(x, y maxGinfo, magFlag)

Inputs: x (list)
y (list)
maxGinfo (array of tuples)
magFlag (Boolean)

Outputs: N/A

Summary: Plots the location of the first side lobes maximum. If the user chooses a magFlag true, the magnitude will be displayed centered at its location. Otherwise, a circle marker will be displayed at the location of maximum gain.

Appendix C: Coordinate Systems

Coordinate System Conversion And Usage:

Radar systems and more specifically phased array radar system require engineers to use many different coordinate systems. It is important to establish a coordinate system that makes sense to use when representing data. For example, when referring to the direction of the main lobe of the radar system it would make sense to discuss it in terms of Azimuth and Elevation coordinates. It would not, however, be entirely useful to discuss the lobe with its normalized direction vector in terms of Cartesian coordinates X, Y, Z. This section will serve as a review about coordinate systems and their use within modeling the AMISR [17].

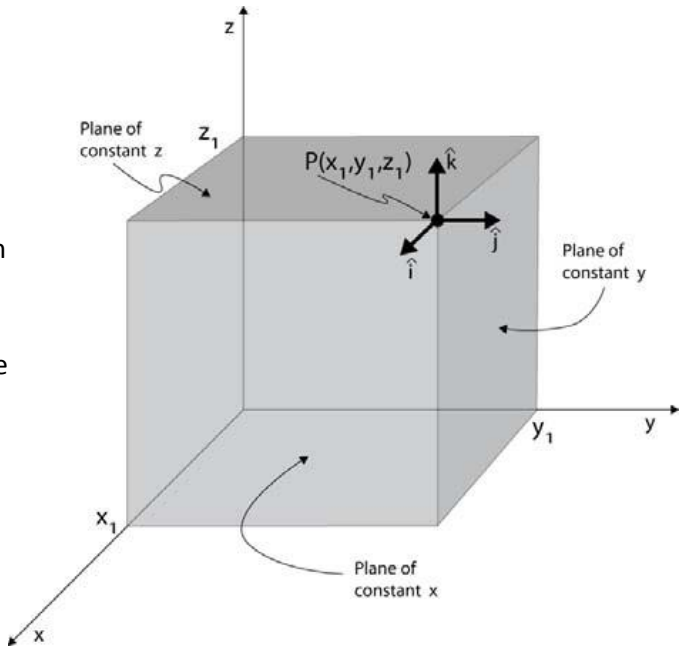

Figure 43-Cartesian Coordinate System

Rectangular, or Cartesian, coordinates simply expand the XY plane to include the Z dimension such that making one coordinate a constant establishes a 2d plane with the other two components. This is illustrated in the diagram which shows a three-dimensional surface where each face is a plane made up of two coordinate components. This coordinate system is the most basic of the three used in this project. However, most all computation and three-dimensional data plotting is done relative to X,Y,Z locations [17].

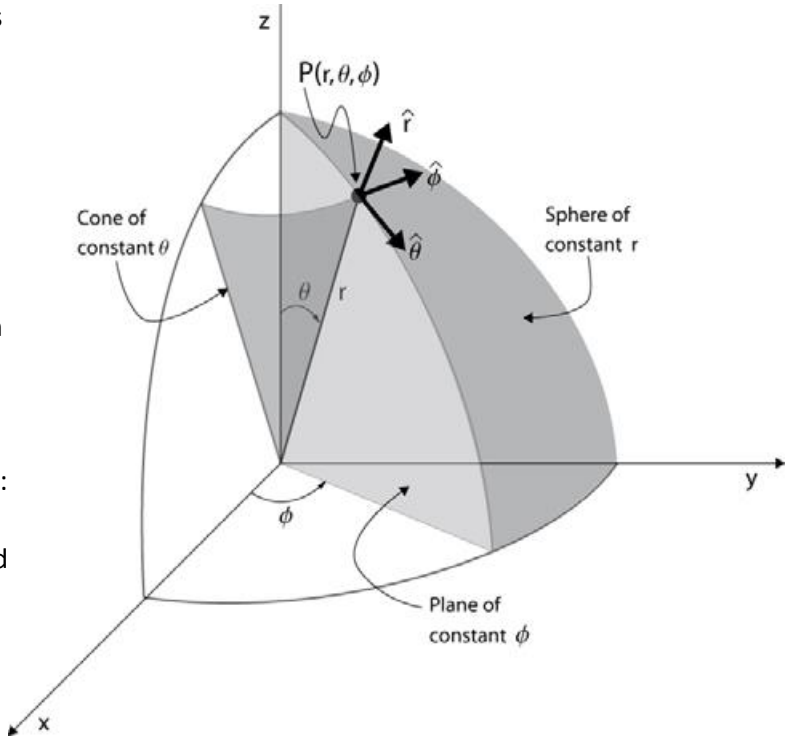

Figure 44-Polar Coordinate System (Azimuth-Elevation)

The next system widely used in this industry is spherical, or Azimuth & Elevation, coordinates. Points in space are determined by first its angle of direction at the ground plane (Azimuth: ϕ), shown in the diagram in the XY plane. Next the angle above the ground plane (Elevation: θ, sometimes called zenith) is determined. Lastly the magnitude of distance away from the origin is determined by the coordinate r. This coordinate system is useful for visualizing wave propagations from

origins outward into space and is well suited for demonstrating the phenomena of radar systems. More specifically, far field gain patterns of phased array radar systems, like the AMISR, may be described in terms of Azimuth, Elevation, and Transmit Gain [dBi]. Other uses include describing orbital patterns of satellites and target spheres. Known trajectory paths of these objects are then used for the AMISR to calibrate it's focal point from observing the Transmit Gain and Receive Gain [17].

The third coordinate system used in modeling the AMISR is called Directional-Sine. In a three-dimension space, with X, Y, Z axis for reference, there are two angles of importance. Angle Beta: β is defined as the angle between the y-axis and vector to the point of interest in space. The next angle is Alpha: α which determines the angle from the x-axis to the point vector. Lastly to define an absolute point in space, the coordinate r determines the distance the point is away from the origin. Note that these values by themselves can only represent positions in the upper hemisphere. The reason being that a set of angles α and β satisfies a line direction above the XY plane (positive Z value) and the angle pair can satisfy a mirrored line direction below the XY plane (negative Z value). To avoid this discrepancy usually an angle Gamma: γ defines the angle from the z-axis to the point vector. However, there is little use to model radiation in the lower hemisphere (for example, this region corresponds to the area below the AMISR face) other than determining if its healthy to walk underneath the phased array. Therefore, the angles used within the calculations for the AMISR with this coordinate system exclude the γ coordinate. Furthermore, when talking about the AMISR, the two coordinate angles describe position off bore sight as Theta-x, Theta-y. The relation between this convention and the classical Directional-Sine system is compared by: 90deg - Theta-x = angle α and 90deg - Theta-y = angle β. The r coordinate associated with that angle pair is the radius or radiation gain in dB referring to either the transmit or receive operation [17].

Now with all these coordinate systems defined it is important to be able to translate between them. By relating the Directional-Sine angles to Cartesian vectors we get the following:

xhat = sin(Theta-x)
yhat = sin(Theta-y)

(x^2 + y^2 + z^2 = 1) therefore:
zhat = sqrt(1 – (x^2 + y^2))

The following relates the Cartesian coordinates to Azimuth, Elevation: (ϕ, θ) coordinates. By using these relations one can translate between all three types of coordinate systems.

(tan(ϕ) = yhat/xhat + pi/2) therefore:
ϕ = arctan(yhat/xhat + pi/2)

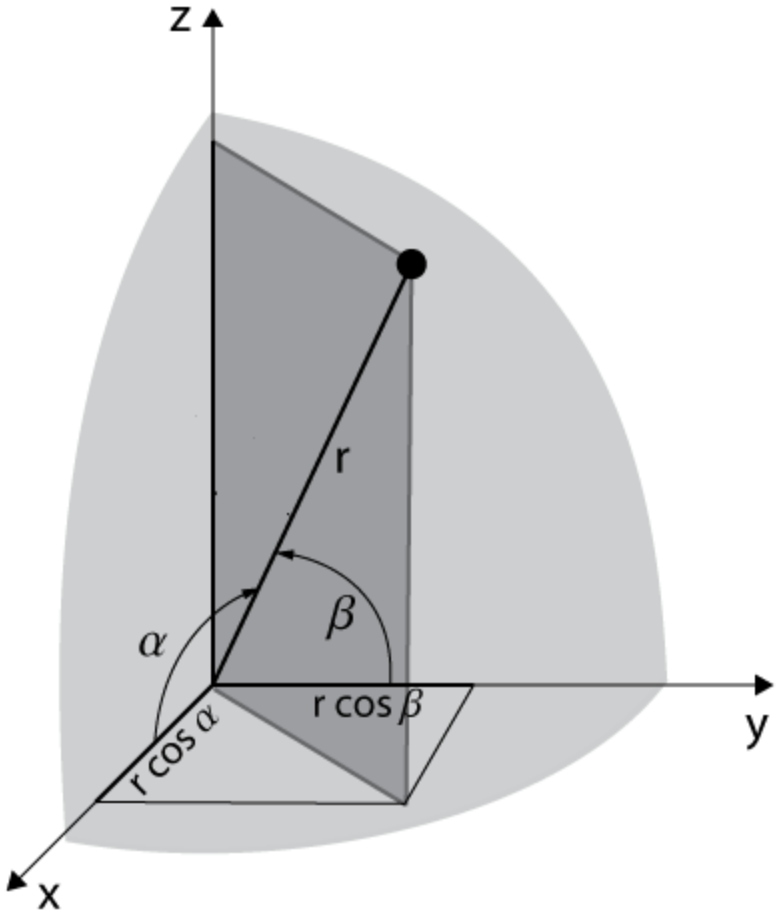

Figure 45-Directional Sine Coordinate System

(cos(θ) = zhat) therefore:
θ = arccos(zhat)

Pictures are taken from [17]

The fourth coordinate system used in modeling the AMISR is perhaps the most important. It's called the Geodetic Azimuth-Elevation coordinate system. The Geodetic system is used frequently in the scientific RADAR community as it describes locations of events relative to a point on the Earth's surface. The world is not a perfect sphere, and because of that non-ideal characteristic, drawing a line from the center of a tangential plane on the Earth's surface towards the center of the earth will most likely not actually hit the "center". The figure below illustrates this important characteristic of the Geodetic Coordinate system. First a point on the Earth's surface has been established as a relative position to work from Then Geodetic Coordinates use an Azimuth-Elevation system (described previously) with that point as a center of origin. Note thatn when representing the x,y,z axis for the Azimuth Elevation system, the x axis points in the direction towards the north pole (but tangent to the surface), y points west (but again tangent), and z points up (normal).

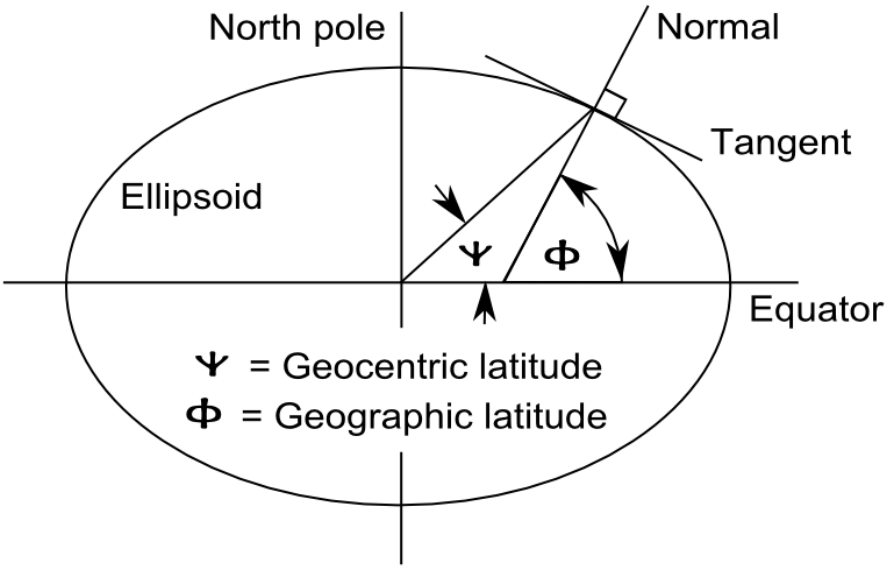

Figure 46-Geodetic Coordinate System

In the case of the AMISR, Satellite Pass information and steering angle information is stored as Geodetic relative coordinates. In order to compute Gain equations and other quantities relative to a particular AMISR Face, one needs to convert from Geodetic Azimuth-Elevation to Face Relative Azimuth-Elevation. Below is an image which describes the location of the PFISR Face in relation to a Geodetic Coordinate system. The axis of the PFISR relative system is rotated on the Azimuth plane by 15 degrees from Geodetic Coordinates and then tilted downwards in 16 degrees Elevation. Below is an image which shows the Geodetic Coordinate System (solid line), overlaid with the PFISR Local Coordinate System (dotted line). [21]

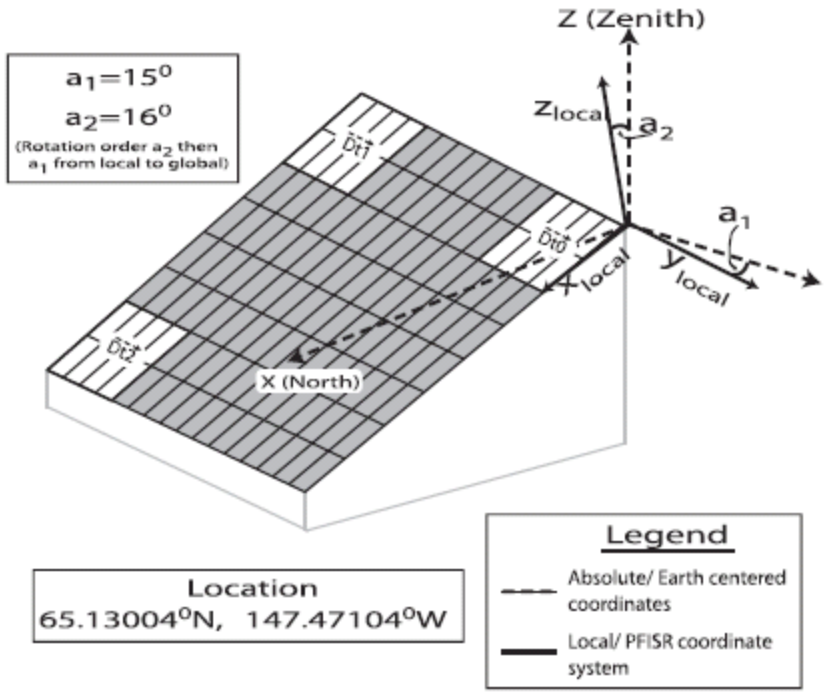

Figure 47-PFISR Coordinate System

In order to convert between the two systems one must use a transformation matrix that takes x,y,z Geodetic Coordinates and translates them to x,y,z PFISR Relative Coordinates. These x,y,z coordinates in either the Geodetic or Face Relative system can then be computed as Azimuth-Elevation points, or Directional Sine points but it is important to have a solid understanding of the x,y,z transformation from Geodetic to Face Relative. Below shows the transformation matrix from Face Relative to Geodetic Relative.

Step1) Rotate azimuth by +15 deg (clockwise) [z axis fixed here, x and y changes]
Step2) Tilt elevation by -16 deg [newer y axis fixed here z and newer x changes]

R_FG1 describes step 1
R_FG2 describes step 2

```
R_FG1 = scipy.array([[ cos(AZ_ROT),-sin(AZ_ROT),      0      ],
                     [ sin(AZ_ROT), cos(AZ_ROT),      0      ],
                     [      0,           0,           1      ]])


R_FG2 = scipy.array([[ cos(E_TILT),       0,     -sin(E_TILT)],
                     [      0,           1,           0      ],
                     [ sin(E_TILT),       0,      cos(E_TILT)]])
```

R_FG describes R_FG1 dotted with R_FG2
To convert xg,yg,zg (Geodetic x,y,z) to xf,yf,zf (Face x,y,z) use R_FG

```
xf = xg*R_GF[0,0] + yg*R_GF[0,1] + zg*R_GF[0,2]
yf = xg*R_GF[1,0] + yg*R_GF[1,1] + zg*R_GF[1,2]
```

```
zf = xg*R_GF[2,0] + yg*R_GF[2,1] + zg*R_GF[2,2]
```