CHALMERS

Simulation and Evaluation of an Active Electrically Scanned Array (AESA) in Simulink®

JENNY HILBERTSSON
JOSEFINA MAGNUSSON

Signal Processing Group
Department of Signals and Systems
Chalmers University of Technology
Göteborg, Sweden, 2009

EX033/2009

CHALMERS

Simulation and Evaluation of an Active Electrically Scanned Array (AESA) in Simulink®.

Jenny Hilbertsson
Josefina Magnusson

Antenna Group & Signal Processing Group
Department of Signals and Systems
CHALMERS UNIVERSITY OF TECHNOLOGY
GOTHENBURG 2009

Supervisors at Saab Microwave Systems:
Maria Lanne, Ph. D.
Peter Nilsson Drackner, M.Sc.E.E

Examiners at Chalmers:
Professor Mats Viberg
Professor Per-Simon Kildal

EX033/2009

ABSTRACT

An *Active Electronically Scanned Array* (AESA) is an active antenna consisting of a large number of radiating elements and is commonly used in today's radar systems. One of the major advantages with this kind of antenna system is the ability to steer the beam electronically from one direction to another without any mechanical motion.

The assignment of the master thesis project is to create a model of an AESA in Simulink® with the main purpose to evaluate whether Simulink® is a satisfying tool when simulating these large complex systems. Furthermore, the idea was to make a simulation using the Simulink® model with as many features of the Saab Microwave Systems antenna demonstrator (a 96 element receiver antenna) as possible.

A basic model is developed and then piece by piece upgraded to a more realistic antenna system model. The model is built using predefined Simulink® blocks where the block parameters are set by the user. One limitation with this model is the difficulty to implement a system with an arbitrary dimension since every box needs to be connected to the other boxes by hand.

To overcome this problem, a similar system model is developed using Embedded Matlab™ Function blocks. The two system models are designed to have the same functionality and since the first version is fixed to 96 elements, the comparison of the system models is performed with the same amount of elements. The difficulty with the Embedded Matlab™ version is the handling with large matrixes. The magnitude of the problem, that is, the number of array elements and the number of evaluation angles in the far-field, needs to be kept low enough to not run out of memory.

The two models both have strengths and weaknesses. A combination of the two models would probably result in the best outcome, the advantage of being able to study large and detailed system models combined with the possibility of having a flexible and arbitrary number of array elements.

PREFACE

This thesis is a part of a Master of Science program at Chalmers University of Technology and has been performed at the Department of Signals and Systems between September 2008 and April 2009. The work has taken place at, and in cooperation with, Saab Microwave Systems in Lackarebäck, Gothenburg.

Moreover, this master thesis is a part of the strategic research centres *Strategic Research Center on Microwave Antenna Systems* (Charmant) and *Chalmers Antenna Systems VINN Excellence Centre* (Chase), where the first is sponsored by the *Swedish Foundation for Strategic Research* (SSF) and the second is sponsored by the *Swedish Governmental Agency for Innovation Systems* (VINNOVA) and Chalmers.

First, we would like to thank our supervisors Maria Lanne and Peter Nilsson Drackner for their explanations and guidance throughout the difficulties and obstacles during the work. We are particularly grateful to Maria for the effort she has put on reviewing grammar and offering suggestions for improvement in our thesis. A special thank to Sören Johansson for his interest and support to our thesis and for given us the opportunity to do this work at DD/UX and Saab Microwave Systems.

We would like to express our gratitude to Andreas Wikström who helped us regarding simulations of the large system in Simulink®. A grateful thank to Ulf Carlberg at *SP Technical Research Institute of Sweden* (SP) for the help regarding Simulink® and for being indulgent towards spam. We would also like to thank Mats Viberg and Per-Simon Kildal for being our examiners.

Thank to all co-workers that has shown interest in our work and contributed to making this an enjoyable time, especially Lars Erhage, Magnus Eriksson, Hanna Isaksson, Johan Jönsson, Mikael Pedersen, Elin Rönnow, Jesper Skogsberg and Jonas Wingård.

Last, but not least, we would like to thank our families and friends for supporting and encouraging us during this thesis and the time of study.

Gothenburg, April 2009

Jenny Hilbertsson & Josefina Magnusson

ABBREVIATIONS AND ACRONYMS

AESA	Active Electrically Scanned Array
Charmant	Strategic Research Centre on Microwave Antenna Systems
Chase	Chalmers Antenna Systems VINN Excellence Centre
ESA	Electrically Scanned Array
GENA	Generic AESA Demonstrator
GUI	Graphical User Interface
HPA	High Power Amplifier
IFF	Identification Friend or Foe
LNA	Low Noise Amplifier
M-AESA	Multirole AESA
MSA	Mechanically Scanned Array
PA	Power Amplifier
PESA	Passive Electrically Scanned Array
RAM	Random Access Memory
RF	Radio Frequency
SLL	Side Lobe Level
SNR	Signal-to-Noise Ratio
SSF	Swedish Foundation for Strategic Research
T/R module	Transmit Receive module
VINNOVA	Swedish Governmental Agency for Innovation Systems

CONTENTS

TABLE OF FIGURES

1 INTRODUCTION

In this thesis, a specific type of antenna system, called *Active Electrically Scanned Array* (AESA), is studied. The basic principle as well as the structure of the system is presented where the latter is modelled and simulated in a program called Simulink®.

The assignment is to model an antenna system where all the fundamental parts are included such as antenna aperture, receiver and combining network. A simple and ideal *transmit receive module* (T/R module) is modelled as a first step and piece by piece developed into a larger and more complex planar array system.

1.1 BACKGROUND

Ever since Christian Hülsmeyer got his patent for the *Telemobilescope* 1904, where he calculated the distance for an electro-magnetic wave to travel back and forth between a metal object and his scope, the development of radar technology has progressed.[1] In 1935 Great Britain launched research activities regarding air defence. The radar technology in particular was a question of vital importance for the defence of Britain but it took some years before the technique was exploited to the utmost. The new radar technology was integrated with radiotelephony and direction finding, which made it possible to achieve both early warnings and the positions for hostile planes in the battle of Britain during World War II.[2]

As many times before, military purposes have precipitated the technical development which later on has proven useful also for civil applications. With more advanced antennas, the radar can perform more sophisticated search and track pattern but the technique is also used for non military applications, such as air traffic control, weather monitoring and road speed control.

The radar technology can briefly be described as follows. Electromagnetic radiation is transmitted and propagates through the air and when it reaches a target, or some other obstacle, the wave is reflected. The reflection returns as a weak signal, also known as an echo, and from this signal useful information of the target can be extracted, for example the distance, the position and the rate of speed (in case of a moving target).

Today, radar technology is a widely spread technique for different fields of applications. Several systems have been designed, for instance the Erieye and the Giraffe, two radars developed by Saab Microwave Systems. Both are surveillance radars, but the first one is air-borne radar and the second is ground-based radar. Among others, SAMPSON and APAR radars can be mentioned as phased array naval systems, the MESA and THAAD as surveillance systems and AMSAR and Gripen PS-05 as fighter radars.[3]

The next step in systems for radar and electronic warfare is the development of the *Multirole AESA* (M-AESA), which is a development program between Saab Microwave Systems, Selex Sistemi Integrati S.p.A and Elettronica S.p.A by order of the Swedish and Italian Armed Forces. The goal is to create a system for radar, electronic warfare and communication, used for land-, sea- and air-based applications. Radar systems of today usually have one main function, but the main idea of the M-AESA is a multifunction system where *Identification Friend or Foe* (IFF), missile warnings and jamming of hostile radars are some of the functions. The existing hardware cannot manage the requirements for the future

[1] Skolnik, Merrill I. *Introduction to Radar Systems*. 3rd ed. New York: McGraw-Hill. 2001. p. 15

[2] <http://www.foi.se > (17 December 2008)

[3] Nilsson Drackner, Peter. M.Sc.E.E. personal communication (8 October 2008)

but new semiconductor components and materials might be the solution to this problem. With multithreat capacity and multifunction, a system will, depending on the tactical situation, be used for different applications. [4]

1.2 PURPOSE

The main purpose of this thesis is to design a model, which consists of the essential parts of an AESA system, in Mathworks™ Simulink®. An AESA consists of a large quantity of components which together constitute a large complex system. To understand and predict its performance knowledge about the different parts is needed, as well as an overview and understanding of the entire system. By modelling the system in a simulation program, it is possible to increase the understanding for how the different parts act together and affect the system performance.

Furthermore, a goal is to evaluate whether Simulink® is a satisfying tool to use when modelling these types of large, complex systems and if it can be useful for similar types of systems. As the work proceeds, the model will become more and more realistic and aspects like maladjustments and interference from internal as well as external factors will be taken into consideration.

Another goal is to extract information from an authentic AESA model and implement the features into the Simulink® model to get a more realistic model.

1.3 LITERATURE SEARCH

Information concerning antennas, essential components and AESA principles are procured during the first weeks. For the main part the literature consists of articles from IEEE Xplore, previous thesis and books about array antennas and radar technology.

This thesis work has, as mentioned, been performed as part of the strategic research centres Chase and Charmant which implicate an opportunity to take advantage of the knowledge and experience of the centres, especially concerning programming in Simulink®. Since Saab Microwave Systems holds a large amount of information and knowledge itself, several rewarding discussions and conversations with co-workers have taken place and given a clearer vision of the subject.

1.4 LIMITATIONS AND SIMPLIFICATIONS

Since AESA systems are complex and contains a large amount of components, a true model is hard to achieve. In this thesis several limitations have been done, some due to complexity and others because of time limitation.

The antenna elements are modelled as isotropic elements where losses, mutual coupling and maladjustments are disregarded. In the model using Embedded Matlab™ Function blocks there is a possibility to take radiation losses into consideration.

In this thesis, the focus is on the AESA operating in receive mode, where the main purpose for this decision is to be able to compare the simulated results with existing measurement data as the model is developed.

[4] <etn.se> (2 February 2009)

The T/R module has been limited to contain the three most important components; a *low noise amplifier* (LNA), a phase shifter and an attenuator, where the LNA has been modelled as an ideal component, described by a constant. The combining network, which in reality is a complicated net due to mismatches are in this thesis represented by a summation net which add up all signals without any losses.

2 THEORY

This section contains some basic antenna theory that is used within the work. Along with informative text and pictures some equations are presented in order for the simulation work to appear more comprehensive. The theory is divided into some general antenna theory, a Simulink® introduction and a short presentation of a demonstrator. Together they contribute to give the reader a brief theoretical introduction to this thesis work as well as insight in what the reality can look like. The terms used, are considered as an antenna in either transmitting or receiving mode, depending on which is more convenient for explaining a particular antenna property.

2.1 ARRAY ANTENNAS

Some antenna sources can be seen as isotropic elements, which mean that they radiate equally well in all directions. This is just a theoretical assumption which is used to simplify calculations. Many antenna elements, especially small ones, are often radiating similar to isotropic elements with a wide radiation beam. This is not always a desired feature, several field of antenna applications require a more directive radiation characteristic.

The ability to focus energy in a certain direction is called *directivity* and for an antenna with physical area *A* this is calculated according to *Equation 1,* where lambda (λ) is the free space wavelength. One way to achieve higher directivity is to enlarge the electrical size of the antenna which can be done by combining several antennas into a so called array antenna.

The relation between the directivity and the *gain* of an antenna is given according to *Equation 2,* where *G* is the gain, *D* is the directivity and e_{rad} is the radiation efficiency. (The radiation efficiency is always less or equal to unity.) As can be seen in the two equations below, an array antenna has higher gain than an element of a single antenna, which makes it possible to transmit a signal longer distances. Therefore are arrays more suited for applications like tracking radar and search radar where transmitting over long distances is a desired property. [5]

$$D_{\max} = \frac{4\pi}{\lambda^2} A \quad [\text{-}]$$

Equation 1[6]

$$G = e_{rad} D \quad [\text{-}]$$

Equation 2[7]

An array antenna can be designed in many ways using different kinds of antenna elements, but in most cases the same type of antenna element is used over the whole antenna aperture. Depending on which properties that are desired, the elements can be located in different ways, which means the elements are placed in a row or in a grid of elements placed on a linear or a planar surface, respectively. Other configurations, such as circular arrays are also possible, where circular arrays are one specific case of conformal arrays, that is, arrays placed on non-

[5] Stimson, George W. *Introduction to Airborne Radar.* 2nd ed. Mendham, USA. SciTech Publishing. 1998 pp. 473-479.

[6] Kildal, Per-Simon. *Foundations of Antennas: A Unified Approach*. Lund, Studentlitteratur. 2000 p. 69

[7] Pozar, David M. *Microwave and RF design of wireless systems*. John Wiley & Sons. 2001 p. 118

planar surfaces. This provides the possibility to tailor-made array antennas with certain properties.

Array antennas can be divided into two main groups, *broadside* and *endfire*, depending on how their main beam is designed to radiate. An *endfire array* has its main beam in the direction along the array, while the *broadside array* radiate perpendicular to the arrays orientation, see *Figure 1*. If the array is phase steered, the direction of the beam can be changed without rotating the antenna platform.[8]

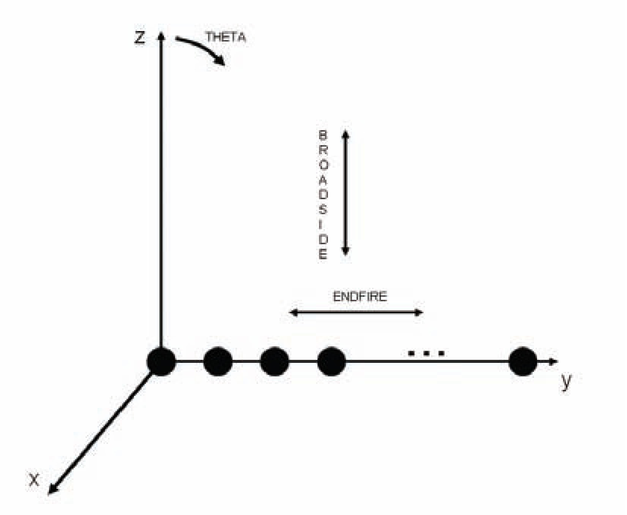


Figure 1. Topology of a linear array.[9]

Not all the radiated power from an array is concentrated to the main lobe direction as desired; some of the power will leak into so called *side lobes*. The amount of this misdirected power is measured relative to the main beam power and is called the *side lobe level* (SLL). Side lobes occur since the surface of the array is finite, which means radiation from less than -90° and more than 90° are received and affect the radiation pattern. In a radiation pattern where losses are neglected, the first side lobe has a relative level of -13.2 dB. To reduce the side lobes and decrease the SLL a method called *tapering* can be used. This method changes the radiation pattern by lowering the signal amplitude in the elements close to the edges. Using this technique the main beam will be less focused, meaning that the main beam will be wider.

The radiation field from an array antenna (or a single antenna element) has different properties depending on how far away from the array it is measured. The region surrounding the antenna can be divided into two different main parts, *near field* and *far field*, where the transition between them depend of the wavelength and the diameter of the antenna. The far field is defined as the area where the spherical wave front, radiated by the antenna or array, becomes a close approximation to the ideal planar phase front of a plane wave. The definition of the far field is described by the radius distance, *r*, from the antenna according to *Equation 3*, where *D* is the largest dimension of the antenna and λ is the wavelength.[10]

[8] <www.orbanmicrowave.com> (2 October 2008) pp. 1-15

[9] <www.orbanmicrowave.com> (2 October 2008) p. 1

[10] Kildal, pp. 37-42.

$$r \geq \frac{2D^2}{\lambda} \quad [\mathrm{m}]$$ *Equation 3*[11]

2.1.1 LINEAR ARRAYS

The elements in a linear array are distributed over a one dimensional straight line and are often equidistantly positioned to achieve a desired radiation pattern. If a plane wave is incident towards an equidistantly array where all elements are identical and the antenna elements are omnidirectional, the received wave will diverge only by a phase difference at each element. This phase difference can be calculated according to *Equation 4*, where *x* is the extra distance the wave has to travel from on element to the next, *dx* is the distance between two elements and θ is the direction of the incident wave. An illustration of these parameters can be seen in *Figure 2.*[12]

$$\Delta\varphi = \frac{x}{\lambda}2\pi = \frac{dx\sin\theta}{\lambda}2\pi \quad [\mathrm{rad}]$$ *Equation 4*[13]

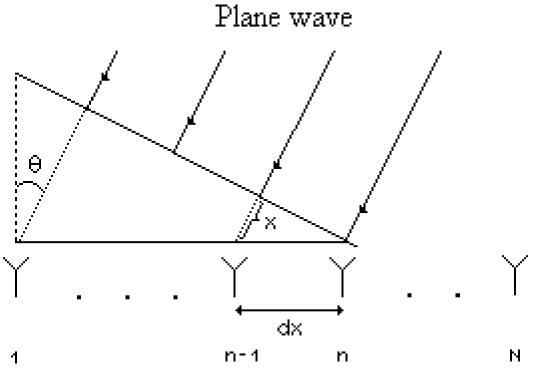


Figure 2. Geometry of a linear array antenna.

To achieve a more definite radiation pattern, the field from the elements need to interfere positively in the desired direction and have negative interference in all other directions. This is possible by phase steering the radiation from each antenna element.[14]

The far field radiation pattern from an N-element linear equally spaced array is a summation of the radiation of each element according to *Equation 5*, where *j* is the imaginary unit, *k* is the wave number (2π/λ) and V_n is the complex excitation of the element *n*. Further, *dx* represent the distance between two elements while θ are the angle of the incident wave. Assuming identical element patterns, the equation (for the radiation pattern) can be rewritten and divided into two parts, the *element pattern* and the *array factor,* according to *Equation 6.*

[11] Kildal, p. 38.
[12] Visser, Hubregt J. *Array and Phased Array Antenna Basics.* Chichester, England. John Wiley & Sons. 2005 pp. 123-135
[13] Visser, p. 124
[14] Visser, p. 124.

$$\boldsymbol{E}(\theta,\phi) = \sum_{n=1}^{N} V_n \boldsymbol{E}_{ele}(\theta,\phi) e^{jkndx\sin\theta} \quad \text{[V/m]}$$ *Equation 5*[15]

$$\boldsymbol{E}(\theta,\phi) = \underbrace{\boldsymbol{E}_{ele}(\theta,\phi)}_{Element\ pattern} \underbrace{\sum_{n=1}^{N} V_n e^{jkndx\sin\theta}}_{Array\ factor} \quad \text{[V/m]}$$ *Equation 6*

The element pattern, $\boldsymbol{E}_{ele}$, describes the radiation of a single antenna element and will therefore change appearance depending on what kind of antenna element is used. The array factor is dependent of the element spacing *dx,* the number of antenna elements in the array, *N*, and the amplitude and phase excitation of each element.[16] The bigger element spacing the larger directivity is obtained, as long as there are no visible *grating lobes*. Grating lobes are maxima in the radiation pattern in other directions than the main beam. To avoid these often undesired maxima (they can be mistaken for the response of a target in the main beam) the requirement on the element spacing for non-radiating grating lobes can be calculated according to *Equation 7,* where λ_{min} is the wavelength at the highest frequency. Furthermore, $\theta_{\max}$ is the position of the main lobe and *sin(*θ_{00}/2) corresponds to the half null depth beam width of the grating lobe. An illustration of the parameters can be seen in *Figure 3*.

Having small element spacing, approximately half a wavelength, makes it possible to theoretically design the antenna for full scan, which means the antenna can scan up to almost ± 90° from broadside. In reality, the radiation impedance changes with scan, which means the useful scan region becomes limited to ± 60°.[17]

$$d \leq \frac{\lambda_{\min}}{1+\sin\theta_{\max} + \sin(\theta_{00}/2)} \quad \text{[m]}$$ *Equation 7*[18]

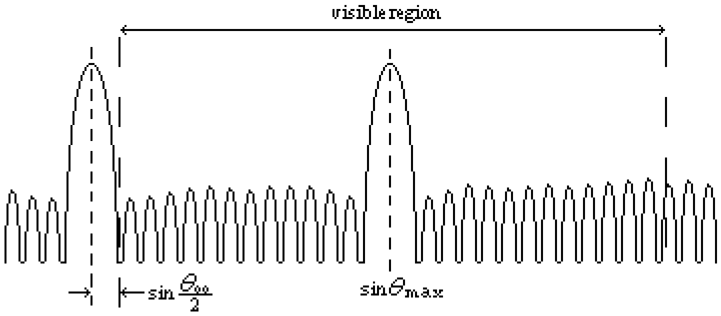


Figure 3. Criteria for non-radiating grating lobes.

[15] Lanne, p. 8.
[16] Lanne, Maria. *Antenna Arrays System: Electromagnetic and Signal Processing Aspects* Lic. Chalmers University of Technology. Gothenburg, Chalmers Reproservice. 2005 pp. 7-10.
[17] Kildal, pp. 341-378.
[18] Kildal, p. 354.

2.1.2 PLANAR ARRAYS

A planar array consists of antenna elements spread over two dimensions where the positions compose a planar lattice. A planar array can be described as a linear array of linear arrays if the grid on the planar array is of rectangular shape and all the elements are equally spaced, this does not mean they need to have same spacing in the two orthogonal directions, see *Figure 4*. The radiation field from a planar array is a summation of the radiation from each element, just like for the linear array. Since the planar array is distributed over two dimensions the equation will be described by both *theta* (θ) and *phi* (ϕ), according to *Equation 8*. V_{mn} describe the complex excitation for the element in the m^{th} row, n^{th} column. $\boldsymbol{E}_{ele}$ implicate the element pattern, *k* the wave number and *dx* respectively *dy* represent the distance between two elements in the different dimensions.

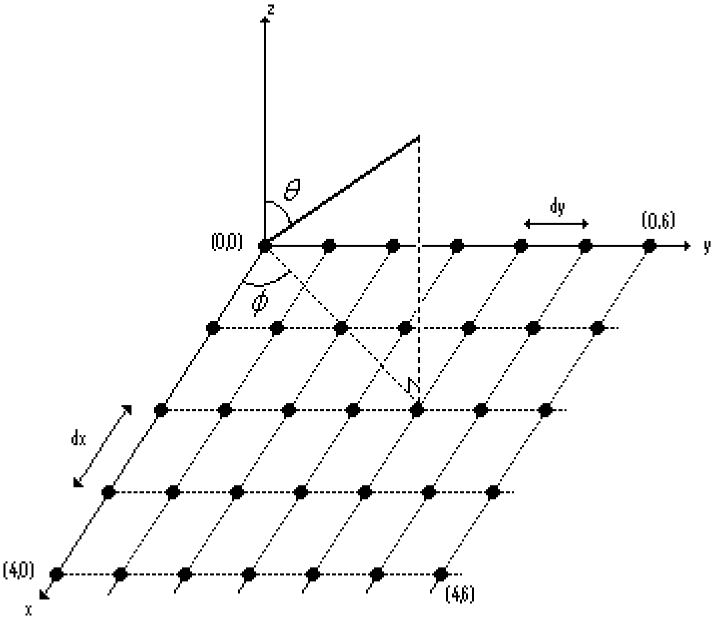


Figure 4. A periodic rectangular two dimensional array geometry.

$$\boldsymbol{E}(\theta,\phi)=\sum_{m=1}^{M}\sum_{n=1}^{N}V_{mn}\boldsymbol{E}_{ele}(\theta,\phi)e^{jk\sin\theta(mdx\cos\phi+ndy\sin\phi)} \quad [\text{V/m}] \quad \textit{Equation } 8^{19}$$

Using planar arrays makes it possible to shape the radiation pattern in two dimensions and to control the radiation pattern in both θ- and ϕ-direction. Further, the side lobes can be lowered in both directions as well and can be beam steered toward any point in space.[20]

Radiation diagrams are one way to present the result from an array antenna. The diagrams represent the radiation from the antenna and can be plotted in θ- and ϕ-plane or in *u*- and *v*-plane. The most common way is to plot the radiation in *u*- and *v*-plane which is a transformation from the (θ, ϕ)-coordinates using *Equation 9* and *Equation 10*.

[19] Lanne, p. 14.

[20] Balanis, Constantine A. *Antenna Theory: Analysis and Design.* 3nd ed. Hoboken, New Jersey. John Wiley & Sons. 2005 pp. 349-362

$$u = sin\theta \cdot cos\phi \quad [-]$$ *Equation 9*

$$v = sin\theta \cdot sin\phi \quad [-]$$ *Equation 10*

2.2 SCANNED ARRAYS

This section will introduce and briefly describe and compare different scanned arrays as well as a comparison between them, where advantages and disadvantages will be listed. The first section will introduce and compare the two main groups of scanned arrays, *Electronically Scanned Array* (ESA) and *Mechanically Scanned Array* (MSA). In the last section there will be a comparison between *Passive ESA* (PESA) and *Active ESA* (AESA) which are two subgroups to the ESA.

2.2.1 ESA VS MSA

Some of the technologies today require the array antenna to have a moveable main beam, for instance many radar applications. Scanned arrays are the solution to this requirement since they have the ability to scan large areas instead of only using a fixed beam.

As mentioned earlier, the MSA is a scanned array which uses mechanics to change the direction of the main beam. This is achieved by mounting the array on a turn-table and then tilts the entire antenna to point in the desired direction. The largest disadvantage of this method is the force of inertia which limits the speed of the mechanical movement. Therefore scanning between two far off angles is time consuming, but it is a good operative method considering the financial aspects, due to the fact that MSAs only uses one central transmitter and receiver for the entire system. These kind of mechanical arrays are commonly used in both land-based and air-based applications.

In contrast to the MSA, the ESA is mounted on a fixed plane and instead of mechanical movement the beam is steered using electronics. By adjusting the amplitude and phase of the element excitations, the direction of the main beam can be changed. Phase shifters are used to change the phase while the amplitude adjustment is done by amplifiers. Since this technique is based on electronic signals, the time to scan the main beam from one point into another far off point is identical with scanning to the nearest possible. This is a desirable advantage and one of the main reasons why many MSAs are replaced by ESAs even though the latter is more expensive due to its complexity.[21]

2.2.2 AESA VS PESA

The ESAs can be divided into the two groups, passive and active ESAs, both having the feature to steer the main beam using the combination of phase shifters and amplifiers. The main difference with the PESA is the use of only one central transmitter and receiver for the entire system, while the AESA has one power amplifier behind each antenna element. This makes the AESA more complex and therefore is it also more expensive than the PESA. Both the PESA and the AESA have one electronically controlled phase shifter placed behind each radiating element of the array.

[21] Stimson, pp. 473-479.

The AESA is commonly designed with one T/R module behind each element in the array. In addition to phase shifters and attenuators, T/R modules commonly include *high power amplifiers* (HPA), circulators or switches (to switch between reception and transmission), limiters and LNA for the received signals. All T/R modules are in transmit mode fed with a RF-signal via a RF feed network. The use of array antennas with a large number of elements and T/R modules makes it possible to operate the antenna even when some T/R modules are broken. This advantage is called *graceful degradation*. Since the T/R module makes it possible to individually steer the amplitude and phase excitation of each element, superior beam shape agility is also provided.[22]

2.3 SIMULINK®

Simulink® is a *Graphical User Interfac*e (GUI) extension to Matlab®, containing several tools for modelling and simulation of multidomain dynamic systems. It supports linear and nonlinear systems, modelled in continuous time, sampled time, or a combination of the two. Since Simulink® is an integrated part of Matlab®, data can easily be transferred between the programs which make it possible for the user to take full advantage of the features offered by both environments. In Simulink® the user can build hierarchical models as block diagram, with predefined blocks using drag-and-drop mouse operations. The program has an extensive block library containing sources, sinks, math functions and many other useful blocks which altogether is over two hundred pieces. Some of the boxes are underframes in which the user has the possibility to create own properties using operation code as for instance C++ and Embedded Matlab® code.[23]

Since a large amount of boxes might cause the system to be difficult to lucid, the model can be simplified by grouping blocks into subsystems. Besides the lucidness, subsystems make it easy to keep blocks with related functionality together and establish a hierarchical block diagram with different layers. Subsystems are either executed conditionally or unconditionally, where the latter are always executed. Depending on the input signal, the conditionally executed subsystem may or may not be executed.[24] As there are coding algorithms better suited for the textual language instead of the graphical, the *Embedded Matlab™ Function* block is a useful tool. Working with the Embedded Matlab™ subset makes it possible to perform embedded software design as well as implementation and testing completely within the Matlab® workspace.[25]

2.4 DRUTTEN, THE DEMONSTRATOR

Saab Microwave Systems, earlier Ericsson Microwave Systems, has, as part of the *Generic AESA Demonstrator* (GENA) program, worked with an active antenna demonstrator which consists of a rectangular phased array antenna. The demonstrator is only built to work in receive mode.

The demonstrator consists of a planar array with 200 dipole elements set up in a rectangular grid with 10×20 antenna elements. The elements are equidistantly spaced in *x*-direction as well as in *y*-direction, but do not have the same spacing in the two orthogonal dimensions. The 96 elements in the middle, placed 6×16, are connected to one receive module each whereas the

[22] Stimson, pp. 473-491.
[23] <www.mathworks.com> b (29 September 2008)
[24] <www.mathworks.com> c (9 February 2009) pp. 198-199
[25] <www.mathworks.com> a (9 February 2009) p.15

104 elements on the border are terminated into matched loads, see *Figure 5*. The receive path for each of the 96 elements consists of, as can be seen in *Figure 6*, a limiter, an LNA, an eight bit phase shifter and a nine bit attenuator.

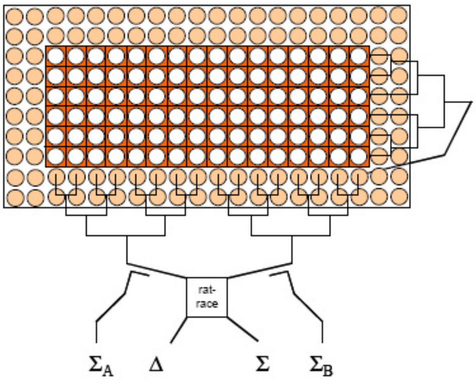


Figure 5. Scheme of the antenna elements in Drutten.[26]

The GENA program focus was on developing basic equipments necessary for next generation of prototypes and products within the field of active antennas for modern radar. It concentrated on technologies and antenna architectures that could be used for different radar applications at different frequencies, where the demonstrator works at S-band (2-4 GHz). The demonstrator has been used to get practical experience in areas like antenna system design, T/R module production, antenna system calibration and evaluation. Some of the goals have been to demonstrate a stable active antenna function with an active receive array and to evaluate a calibration scheme based on factory characterization.[27]

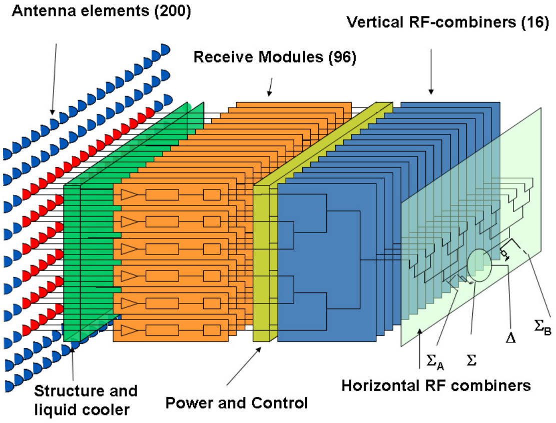


Figure 6. Simple diagram of the antenna unit.[28]

[26] Nilsson Drackner, Peter and Engström, Björn. *An Active Antenna Demonstrator for Future AESA- Systems*. In: IEEE International, *Proceedings of the International Radar Conference 2005*. 2005. p. 227.
[27] Nilsson Drackner and Engström, pp. 226-231
[28] Nilsson Drackner, Peter. M.Sc.E.E. E-mail. (25 March 2009)

3 METHOD

In this section the implementation of the AESA system is presented. Since the assignment of the thesis work is to evaluate whether Simulink® is a good simulation tool for large arrays, two different modelling approaches are evaluated and compared. This is done to explore as many of the features in Simulink® as possible.

The first modelling approach presented is almost pure Simulink® programming with boxes linked to each other where the idea is to avoid intervention of Matlab® to the greatest possible extent, whereas the second is built up with boxes of Embedded Matlab™ programming complemented with Matlab® scripts. The first modelling approach is called *Model Based on Drag-and-Drop Boxes*, while the second modelling approach is called *Model Composed by Embedded Matlab™ Function Blocks*.

Some of the predefined Simulink® blocks used have functionalities that are quite easy to understand, while others need an explanation. The boxes used in the models and a short summary about how they work can be found in *Appendix A*.

3.1 MODEL BASED ON DRAG-AND-DROP BOXES

In this modelling approach the focus is to model an AESA system based on the drag-and-drop boxes method that the Simulink® library offers. First the system is built in a way where each component is ideal and then the components are modified step by step into a more complex and realistic model. As a start an antenna element and a T/R module are modelled, these are modified and copied into a linear and finally a planar array.

3.1.1 ANTENNA ELEMENT AND T/R MODULE

The first step in the modelling of the AESA is to create a model of the T/R module, where all the parts in the module are ideal and linear. In receive mode the first component in the module is an LNA which in this stage corresponds to a multiplication factor. The phase shifter, which follows after the LNA, is modelled as an exponential function where the phase is manually change by the user. The last component in the T/R module is an attenuator and like the LNA this part is represented by a multiplication factor. In this initial model the antenna is lossless and therefore the received signal of the antenna is transmitted unchanged into the LNA. When simulating the model, it is assumed that a plane wave impinges on the antenna. Complex base band representation is used, which means that the signals received by the antenna elements are modelled as complex numbers. This signal is transported through the simulation model and then, using a *Scope block*, the user is able to watch the result, see *Figure 7*.

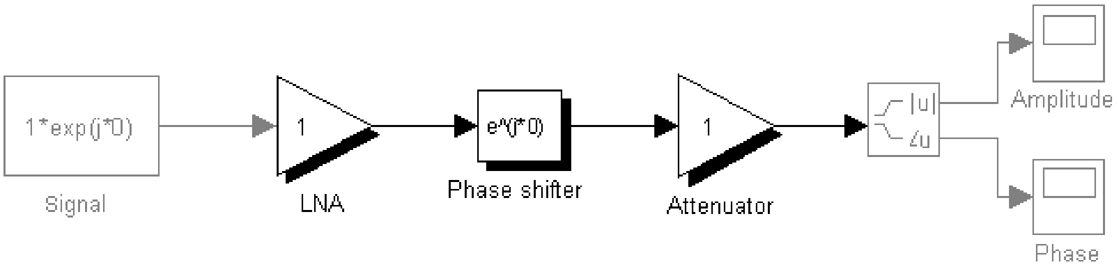


Figure 7. T/R module in receiver mode consisting of an LNA, a varying phase shifter and an attenuator.

As mentioned in the theory part (section 2.3), there is a possibility in Simulink® to group different boxes into so called subblocks. This provides the opportunity to divide the model into different subsystems and thereby make the model more apprehensible. The three components; the LNA, the phase shifter and the attenuator are representing a T/R module and are therefore placed in a subblock with the same name, see *Figure 8*.

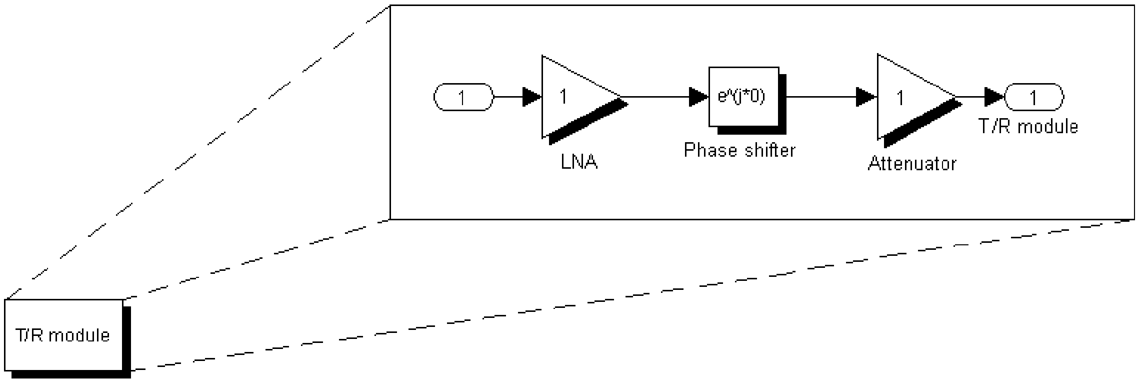


Figure 8. The three components, LNA, phase shifter and attenuator, grouped into a subblock called T/R module.

The input signal to the T/R module comes from the antenna element. The antenna element is modelled under the assumption that there is no mismatch, losses or coupling between the elements and consequently, everything received by the element is thereby transmitted unchanged into the T/R module.

3.1.2 MODELLING OF LINEAR ARRAY

To model an array of ten antenna elements and T/R modules, the model with an antenna element and a T/R module, is copied into ten identical versions and then their signals are added in a combining network, see *Figure 9*. The combining network is at this stage a summation network, where all signals are added disregarding losses and mismatches.

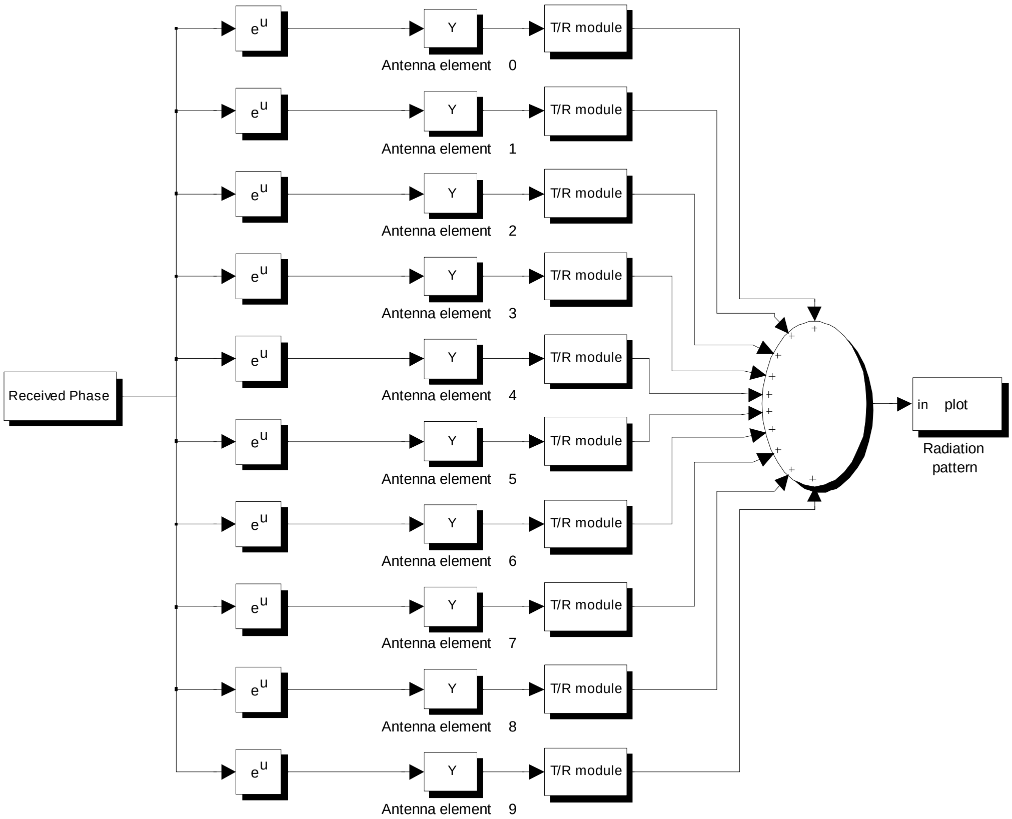


Figure 9. Ten identical antenna elements, their T/R modules and a combining network.

The signals received by the antenna elements are modelled as a vector of complex numbers representing the received plane wave. The received signals are a function of the direction of the incident plane wave relative to the normal of the array antenna aperture. To model how the array receives signals from all angles of incident in the θ-plane, the vector with plane wave directions contains angles from -π/2 – π/2 radians with 180 steps. This is placed in a subsystem called *Received Phase*, see *Figure 10*. Since the angles from -π/2 – π/2 radians equals the angles from -90° – 90°, there will be 181 points representing all integers in the interval.

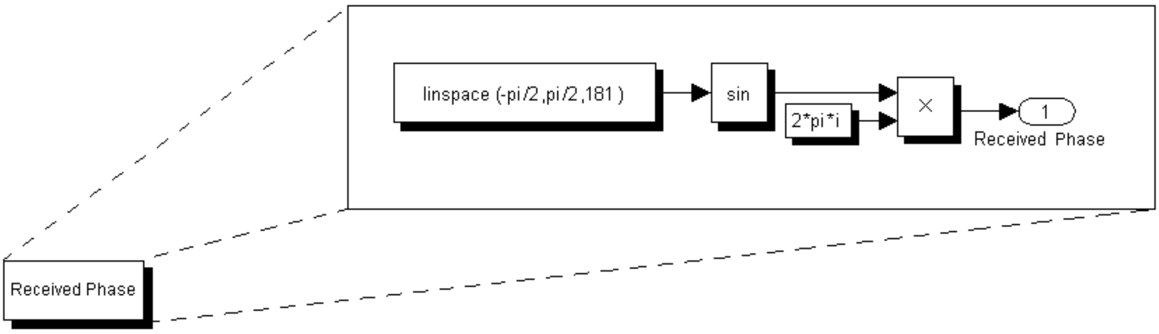


Figure 10. Content in the Receiver phase subbox.

In an array antenna there are physical distances between the antenna elements. Because of these distances an incoming signal will not be received at all elements on the same time, unless the signal is incident perpendicular to the array aperture surface. The incoming signal corresponds to a narrow band plane wave and the difference in the received signal between two elements can therefore be described as a phase shift instead of a time shift. When implementing this in the model, each element needs to have an index corresponding to its position in the array, see *Figure 11*. This index is multiplied with a phase constant which represents the physical distance between two adjacent elements, according to *Equation 11*. Where *k* represents the wave number, *n* describes the index of the element and *dx* the distance between two elements. The angle of the incident wave is described by θ.

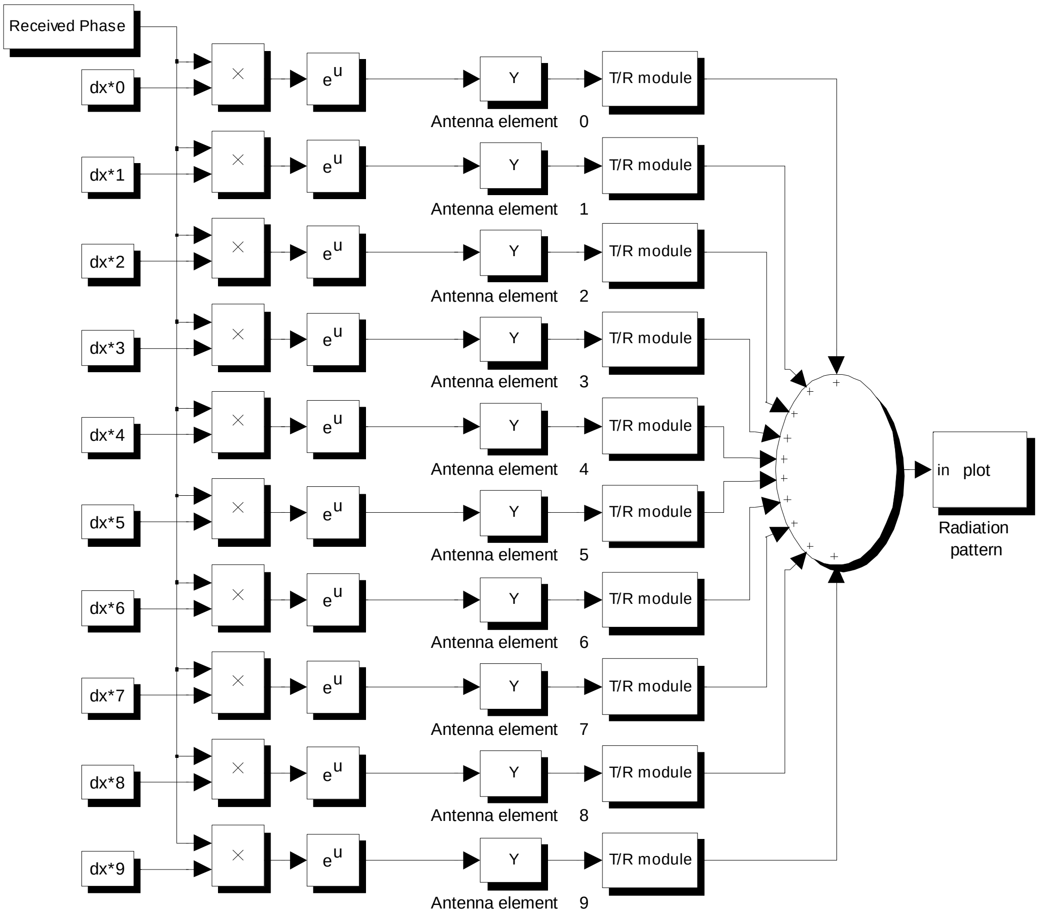


Figure 11. A linear array with ten indexed antenna elements.

$$signal\ into\ element\ n = e^{-jkndx\sin\theta} \quad [\text{-}] \qquad \textit{Equation 11}$$

Both phase shifters and attenuators are created with a predefined scale of values. This scale describes the values that the component can adopt and is limited by a maximum and a minimum value. Phase shifters and attenuators are manufactured with a certain number of bits which is a measure of how many states the scale of values are divided into. If the component for example is an eight bit attenuator the scale is divided equally into 256 different states, according to *Equation 12*. Since there is a limited amount of states in this type of component it is not always possible to set the desired attenuation value. This results in a small residual for each component setting. In this model the number of bits is defined by the user and can be changed from one simulation to another.

$$8-bit\ component \Rightarrow 2^8 = 256 \quad [\text{-}]$$ *Equation 12*

When these bit components are manufactured it is not possible to make them identical since there always will be small variations between the components. This is the reason why two components do not have identical states of scale even though they are designed and manufactured in the same way. The uneven states are described in this model using a *Gaussian normal distribution*, according to *Equation 13*, where *mu (μ)* and *sigma (σ)*, represent the mean value and standard deviation. These parameters determine the magnitude of the variation between the phase shifters and attenuators respectively and are set by the user.

The equation is implemented in a Matlab® script, see *Appendix B*, and calculated for each component when running the program. In the script, the function *randn* is used to return pseudorandom, scalar values drawn from a normal distribution with the specified mean value and standard deviation. This implies that a unique variation between each component will occur.

$$f(x \mid \mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad [\text{-}]$$ *Equation 13*

This description is used in this model for both phase shifters and attenuators with the difference that the states of the attenuators are divided between 0 – 1 while the phase shifters are divided between 0 – 2π. It is possible for the user to have different μ- and σ-values for the phase shifters and the attenuators.

The phase shifter as well as the attenuator is represented by one subsystem each. The phase shifter is a subblock containing an exponential to steer the main lobe into a, by the user predefined direction, see *Figure 12*. The desired phase shift is depending on where in the geometry of the array the element is placed. Since the incoming signal is received with a time shift between the elements this needs to be reversed by the phase shifters to make sure the signals adds up in phase (constructively). Consequently, the phase shifters must have knowledge about where in the array they are placed by knowing the positions of their interconnected antennas. The exponential that can be seen in the figure is varying dependent on the index of the antenna element.

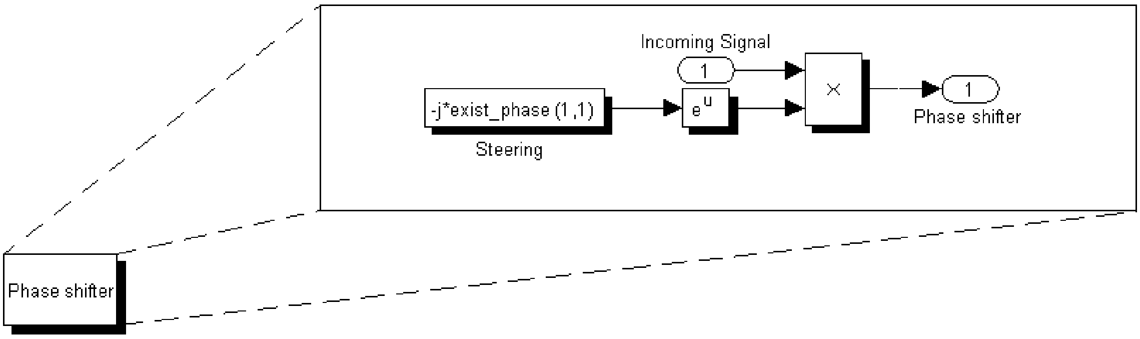


Figure 12. The steering value for element (1,1) are multiplied with the incoming signal in the subblock Phase shifter.

The attenuators are the components that are used when tapering an array. In this thesis a *Taylor tapering* is applied to the elements in the array. For example, a *30 dB Taylor* distribution corresponds to a beam pattern where the highest side lobes are 30 dB below the main lobe maximum. A Matlab® script, see *Appendix B*, is used to calculate the attenuation for each element to achieve a specific tapering. The script calculates the ideal values for a linear array with a specified number of elements. From these values the nearest existing value in the scale of the attenuator is chosen and a small error of attenuation arises (corresponding to the difference between the desired and actual attenuation value). The resulting tapering value is multiplied with the incoming signal in the attenuator, see *Figure 13*.

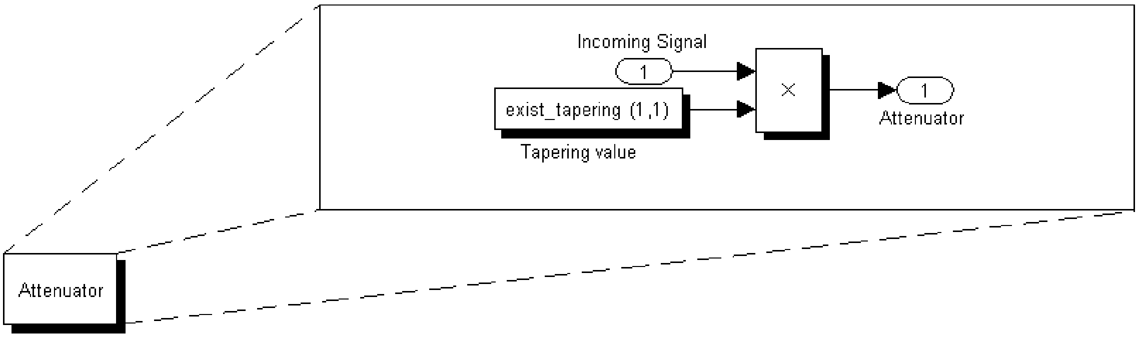


Figure 13. The tapering value for element (1,1) are multiplied with the incoming signal in the subblock Attenuator.

The simulation of the ten different antennas and their T/R modules is done for 181 different angles of incidence which in the end of the simulation is combined and presented in a radiation pattern. When the simulation of the linear array is done, an Embedded Matlab™ Function block is programmed to automatically present a radiation pattern which completes the simulation of the linear array.

3.1.3 MODELLING OF PLANAR ARRAY

To model the planar array the previously modelled linear array is used. Since the model should be comparable to *Drutten* (the antenna demonstrator) having a size of 6 × 16 elements, the linear array is increased to 16 elements. These are copied into six different subsystems, representing six linear arrays, see *Figure 14*. To let the six linear arrays represent a planar array, the indexation of the elements are increased to contain two numbers corresponding to the two dimensions. The signal received at the elements in a planar array are also dependent on two directions and the equation describing the radiation will be expanded as described in the theory, according to *Equation 8*.

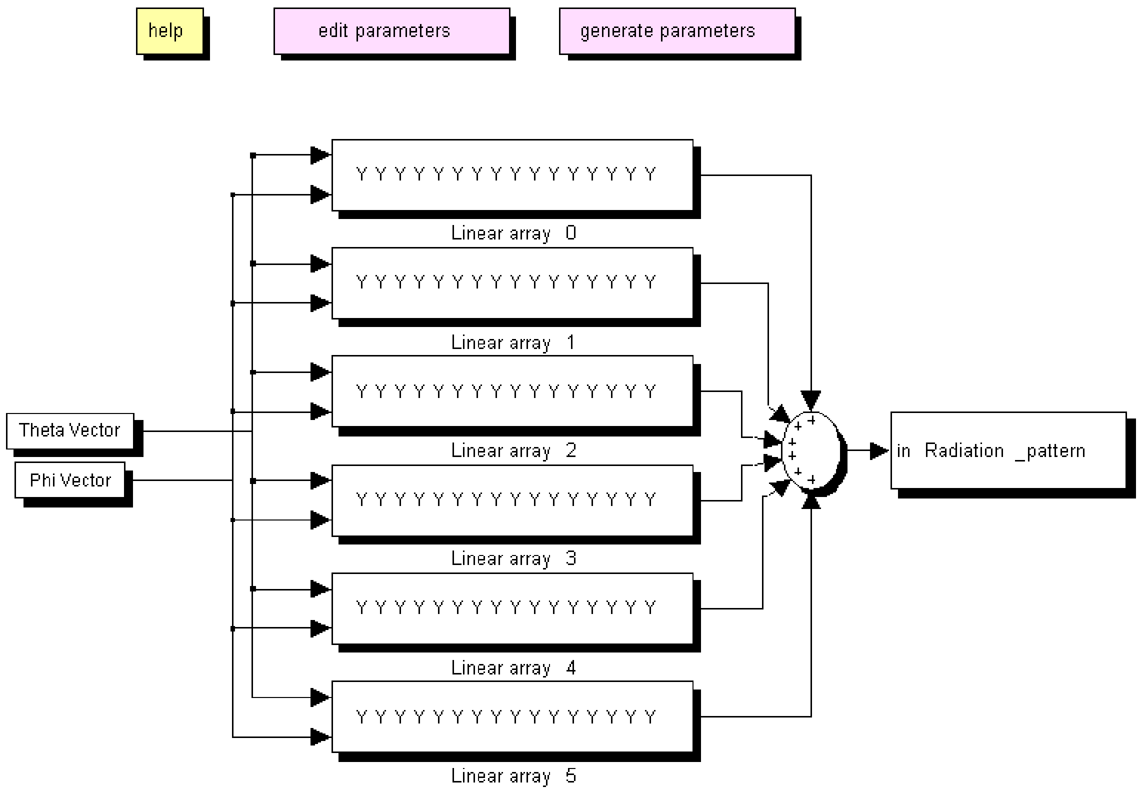


Figure 14. Structure of the planar array.

When having a planar array new aspects must be taken into consideration. The array is two dimensional which means, the beam can be scanned in two dimensions. Therefore is the antenna system modelled as a function of both θ and ϕ, which can be seen in *Figure 15*.

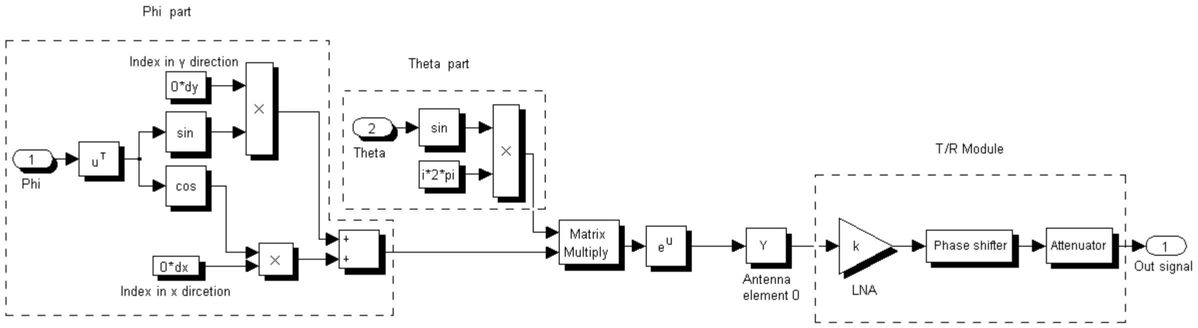


Figure 15. The first element in the planar array.

The simulation of the planar array is done using 181 different angles of incidence in both θ- and ϕ-plane, where θ goes from 0 – $\pi/2$ and ϕ goes between 0 – 2π. The signals from the different elements are added and presented in a three dimension radiation pattern.

The system is complemented with two Matlab® scripts, one containing the different adjustable parameters for the system and one containing calculations based on the modified parameters. The first script, see *Appendix C*, contains the parameters for the antenna geometry, the phase shifter, the attenuator and other variables needed for the simulation. The user should only get in touch with this first script and make changes suited for the wanted simulation since the second script is called upon automatically from the first. The second script, which has been mentioned before, contains calculations of values required for the phase shifters and the attenuators, see *Appendix B*.

3.2 Model Composed by Embedded Matlab™ Function Blocks

The previously designed model was created with a fix number of elements in the antenna system. It also used one box for each operation in the model, which lead to a substantially large amount of boxes. Knowing this, the second model is made with the intention of having an arbitrary number of elements and having all components with the same functionality modelled in same block. One way to create these blocks is to use Embedded Matlab™ Function blocks in which the properties for the blocks are programmed by the user. This makes it possible to create blocks containing many components of the same type, but where each of the components has unique performance.

The previous model was built as an ideal antenna element with a connected T/R module, which subsequently was extended into a large more complex planar array. The *Model Composed by Embedded Matlab™ Function Blocks* is intended to be equal to the final result of the *Model Based on Drag-and-Drop Boxes* system and therefore a planar array is designed without the intermediate steps of a single antenna element and a linear array. Since this AESA model is based on Embedded Matlab™ Function blocks, see *Figure 16*, the system consists of coding algorithms in textual language. The code for the different component blocks used in the system can be seen in *Appendix D*.

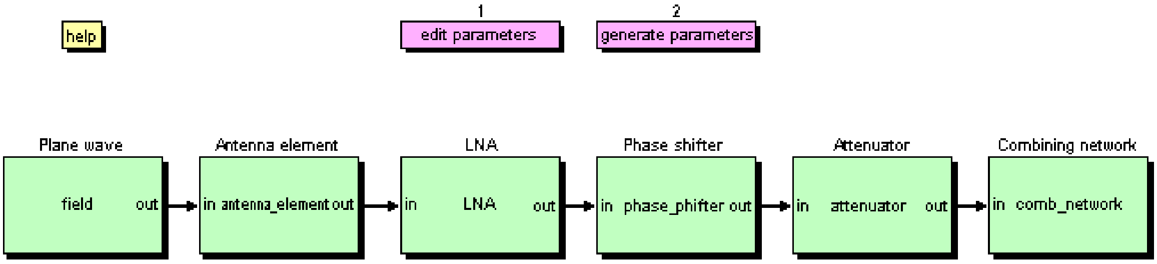


Figure 16. Model composed by Embedded Matlab™ Function blocks.

3.2.1 Information Matrix

The model is built using six Embedded Matlab™ Function blocks, where each of the blocks is programmed to represent a component or a function in the AESA system. Since each antenna element can receive signals from arbitrary directions specified by the two dimensions, θ and ϕ, the information at each element will be represented by a matrix. The matrix is sent throughout the whole system and modified in every box before it is summed up in the last block. This is done by storing the matrixes of each element after each other, in the order in which their antenna elements are positioned. Altogether, the columns in the large matrix contain the amount of elements in x-direction times the number of scanned θ-values and the rows contain the amount of elements in y-direction multiplied with the number of ϕ-values to be scanned. A visualization of a matrix can be seen in *Figure 17*.

The code used in the Embedded Matlab™ Function blocks are similar to the one used in Matlab®, but with a smaller number of functions. The command *cell*, which is a function that creates matrix arrays, is one example of a function that would facilitate the handling of these large matrixes. The cell array allows the programmer to store matrixes at each position in a matrix but this command is not supported by Simulink®.

$$\begin{array}{c|cccc|}
 & 1 & 2 & \cdots & 16 \\
1 & [181\times181] & [181\times181] & \cdots & [181\times181] \\
2 & [181\times181] & [181\times181] & \cdots & [181\times181] \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
6 & [181\times181] & [181\times181] & \cdots & [181\times181]
\end{array}$$

Figure 17. Two-dimensional matrix with 16 × 6 elements and 181 × 181 points in different directions.

One of the criterions for the Embedded Matlab™ Function block is the requirement of having the size of the output independent of the size of the input. For this reason, the size of the matrix is predefined in the script with the eligible variables, see *Appendix E*.

3.2.2 PLANE WAVE

The first block in this model is called *Plane wave* and contains the equation describing the appearance of the signal received at each element of the antenna, according to *Equation 14*. Where the equation is calculated from *Equation 8* for element (*x*,*y*) and where the amplitude of the received field is set to one. The signal depends on the position of each antenna element which is described by the integers *x* and *y* and where *dx* and *dy* represent the spacing between the elements expressed in wavelengths. Furthermore, θ and ϕ symbolize the angles from where the plane wave has been transmitted. If both angels are zero, which means that the signal is incident perpendicular towards the antenna aperture, the signal at each antenna element will be identical. As previously, the received signals are described by a matrix of received values, reflecting the position of the transmitted signal scanned. The received signals are stored together and then transmitted into the next block called *Antenna element.*

$$\textit{signal into element}\,(x,y) = 1\cdot e^{2\pi\cdot j\sin\theta((y-1)dy\cdot\sin\phi+(x-1)dx\cdot\cos\phi)} \quad [-] \qquad \textit{Equation 14}$$

3.2.3 ANTENNA ELEMENT

The antenna elements in the previous model had the features of an isotropic antenna, which means that the antenna was assumed to radiate equally in all directions. This assumption is a common simplification and requires that mutual coupling, losses and maladjustments are neglected.

In this model the antenna element is developed one step further as an element pattern is implemented. The element pattern describes the radiation of an antenna element and is in this model approximated using a cosine function, according to *Equation 15*, where the equation is expressed in normalized amplitude and *m* describes the value which the cosine factor is superscript to. This equation will attenuate the radiation in near edge angles of the antennas. If the *m*-factor is set to zero, the model will be identical to the previous model since the cosine term will become equal to one. Furthermore, the equation is independent of scanning angles in

ϕ-direction, which makes the appearance of the element diagram to be the same for all angles of ϕ. Since the appearance of the diagram is independent of the position of the elements in the antenna, they will have identical pattern for each element. The values for the element pattern are calculated and placed in a vector, which is multiplied with the received matrix using two for-loops and then transmitted into the next block.

$$element\ diagram = \sqrt{\cos^m \theta} \quad \text{[V]}$$

Equation 15[29]

3.2.4 TRANSMIT RECEIVE MODULE

As described in the theory part (section 2.2.2), the T/R module is one of the key elements for the AESA system and even though different architectures are possible the three most important parts; LNA, phase shifter and attenuator, are included in this model. In contrast to the previous model this one has no block named T/R module, since all the components in the T/R modules are placed directly in the model.

In real systems, the function of the LNA is to amplify a weak signal before sending it into the rest of the system. The LNA is the first component in receive channel in the T/R module and can therefore amplify the signal before noise has been added from the rest of the components. In this way the *Signal-to-Noise Ratio* (SNR) is kept high and the signal can more easily be distinguished. The function of the LNA is identical with the foregoing model and since the models are created with a high input signal where the noise is neglected the function of the LNA is not as important as in real systems.

The LNA block, transfers the incoming matrix without modification further to the next block. If the user wants to add maladjustments or losses, the block can be reprogrammed to include them, using two for-loops with a describing equation in the middle.

When the matrix is sent through the system and arrives at the phase shifter box, this block repeal the influence of the distances between the antenna elements. As been described in the drag-and-drop model, the phase shifter should reverse the phase shift that is applied on the received signal due to the distances between the antenna elements. It is the phase shifters that steer the antenna beam into a, by the user, predefined direction as well.

The phase shifter is one of the components which is most developed. In this model, both manufacturing variations and bits are taken into consideration, which makes each phase shifter unique and the result from one simulation to another can include having a small difference. Since there where problems to create an equivalent phase shifter in the drag-and-drop model, the script used in this model, see *Appendix F*, was also implemented in the first system model but with some modifications.

The attenuators are similar to the phase shifters since both have bits and manufacturing variations that are taken into consideration. A Matlab® script, see *Appendix G*, is used to calculate the ideal attenuation values for each antenna element using three parameters defined by the user; the number of elements in the particular array, the SLL and the desired number of side lobes having the same height. To create unique attenuators to this model the ideal attenuation values are modified using the same Matlab® script as for the phase shifters, see *Appendix F*. Each unique tapering value is multiplied into the received matrix on all places

[29] Mailloux Robert J. *Phased Array Antenna Handbook.* 2nd ed. Norwood, USA. Artech House. 2005. p. 305

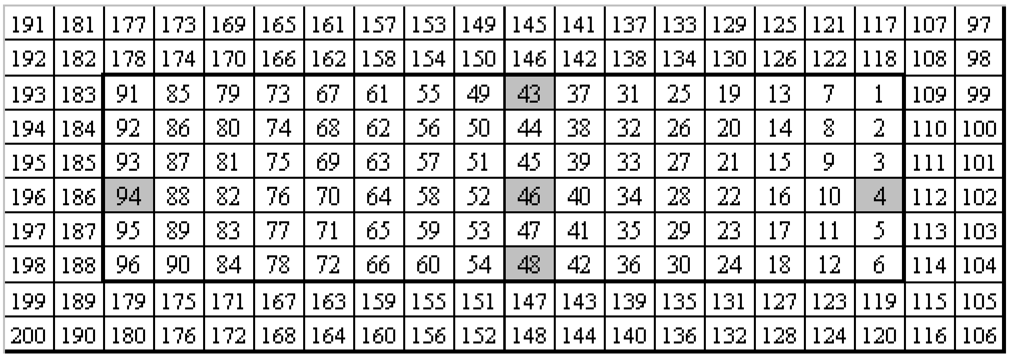
representing its position in the array, for example, the tapering value at element (*1,1*) is multiplied at the first 181×181 places in the incoming matrix. This is done for each antenna element in the array using a double for-loop and then the matrix is transmitted to the combining network block.

3.2.5 COMBINING NETWORK

In the previously described model the combining network corresponds to a summation block which adds up the signals from all elements and transmits the final signal into a plotting embedded box. The *Combining network* block is programmed to have the same functions as the previous system model, but with both the summation network as well as the automatic plotting function in the same box. The block is adding all signals having the same angel of incident, which results in a matrix having a size equal to the number of points in the (θ,ϕ)-directions. The measured data of the matrix is illustrated in three different ways; the 3D far field pattern and the far field cross sections for both *u* and *v*.

3.3 IMPLEMENTATION OF THE DEMONSTRATOR

Measurement data from the demonstrator has been extracted and implemented in the model to get one step closer to a true antenna system. The implementation is done using the *Model Composed by Embedded Matlab™ Function Blocks*, even though the *Model Based on Drag-and-Drop Boxes* could be developed to handle this implementation as well. The measurement data come from the 200 antenna elements in the demonstrator and their positions can be seen in *Figure 18*. As mentioned in section 2.4, the 96 elements in the middle are the ones not terminated into matched loads. Five of the element patterns, corresponding to the elements positioned at place 4, 43, 46, 48 and 94 in the figure, are used to create an average function corresponding to the diagram. The function is implemented in the antenna elements in the model and is therefore representing a model one step closer to a real antenna system.



191	181	177	173	169	165	161	157	153	149	145	141	137	133	129	125	121	117	107	97
192	182	178	174	170	166	162	158	154	150	146	142	138	134	130	126	122	118	108	98
193	183	91	85	79	73	67	61	55	49	43	37	31	25	19	13	7	1	109	99
194	184	92	86	80	74	68	62	56	50	44	38	32	26	20	14	8	2	110	100
195	185	93	87	81	75	69	63	57	51	45	39	33	27	21	15	9	3	111	101
196	186	94	88	82	76	70	64	58	52	46	40	34	28	22	16	10	4	112	102
197	187	95	89	83	77	71	65	59	53	47	41	35	29	23	17	11	5	113	103
198	188	96	90	84	78	72	66	60	54	48	42	36	30	24	18	12	6	114	104
199	189	179	175	171	167	163	159	155	151	147	143	139	135	131	127	123	119	115	105
200	190	180	176	172	168	164	160	156	152	148	144	140	136	132	128	124	120	116	106

Figure 18. Configuration of the elements in Drutten, seen from the front.

4 RESULTS

In this chapter, results from the simulated models are presented and analysed. The outcome from the different models, *Model Based on Drag-and-Drop Boxes* and *Model Composed by Embedded Matlab™ Function Blocks,* are compared and explained. As described in the theory (section 2.1.2), one way to present the performance of an array antenna is to present its radiation pattern. Most commonly, this pattern is presented in spherical coordinates and the results for the models are therefore plotted like this throughout the whole chapter. The simulations are done to show how the performance of the components influences the radiation pattern for the whole system. To distinguish characteristics easier, the cross-sections of the radiation pattern are presented as well.

During all simulations presented in this chapter, the parameter *regenerate_modules* is set to *true* which imply that the T/R modules are unique, not only compared with the other modules, but also from one simulation to another. As long as the standard deviations are set to relatively low values, the regeneration of modules should not affect the outcome considerably.

4.1 OUTCOME FROM THE EMBEDDED MATLAB™ FUNCTION BLOCK MODEL

To demonstrate how the different parameters and components influence the system, figures illustrating the outcome from each block in the model are plotted. The first figure, representing the outcome from the *Field* block, illustrates the radiation pattern for an ideal untapered (uniform) planar array consisting of 16 × 6 isotropic antenna elements, see *Figure 19*. The cross-section of this pattern is illustrated in *Figure 20* and shows a SLL of -13.2 dB, which is correct according to theory, see section 2.1. If counting the number of peaks between u = 0 and u = 1 (corresponding to half a period of the periodic array factor for this array with element separation $\lambda/2$), including the main beam as one peak and multiplying it with two, the product represents the number of elements in the *x*-direction. The same can be performed for the *v*-plane resulting in the number of elements in the *y*-direction.

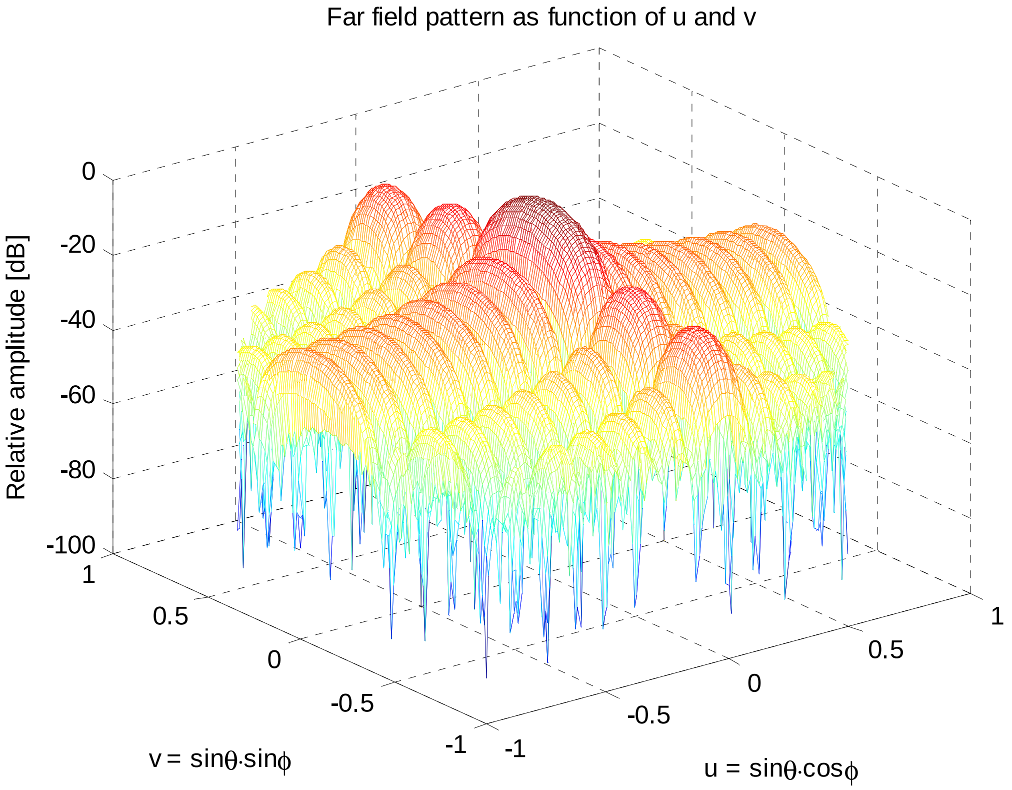


Figure 19. Radiation pattern for an ideal, uniform planar array with 16 × 6 isotropic antenna elements.

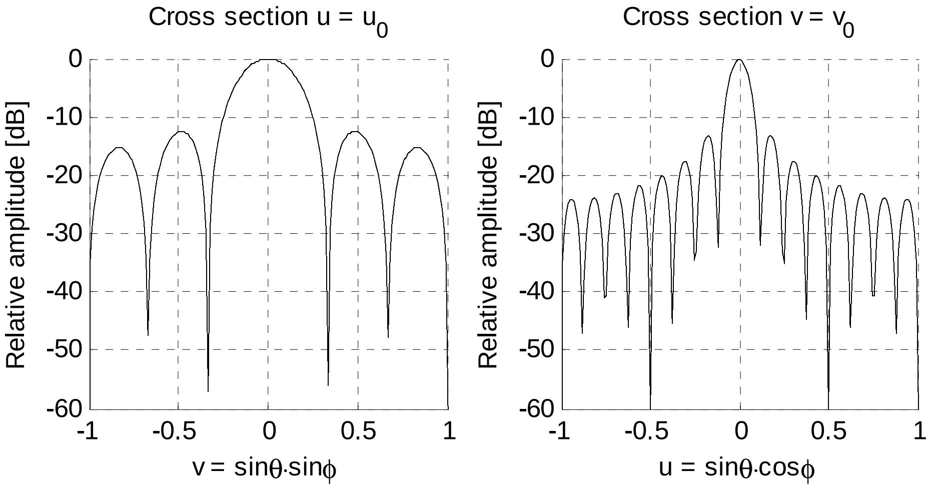


Figure 20. The cross-sections of the radiation pattern from the incoming field.
Left: $u=0$, $v=\sin\theta\sin\phi$ Right: $u=\sin\,\theta\cos\,\phi$, $v=0$

To give a more realistic description of how the antenna elements radiate an element diagram is applied using a cosine shape in both *u*- and *v*-direction, according to *Equation 15* in section 2.6.3. The factor which cosine is superscript to is in this case 1.4 and result in a decreasing radiation for angles further away from zero. If the cosine factor is superscript to zero, the elements become isotropic which represent the case of the *Model Based on Drag-and-Drop Boxes*. The outcome with isotropic element patterns would be the same as in previous figure, since there is no contribution from the elements.

When the field has been received by the antenna elements and weighted by the element pattern, the radiation pattern has changed and can be seen in *Figure 21*. The difference with the previous figure can be seen in the cross-section for *u*, where the level of the second side lobe has attenuated an extra 3 dB, at *v* = 0.81, compared to the left diagram in *Figure 20*. In the cross-section for *v*, every side lobe from the second one and further is attenuated, more or less, compared to the right diagram in *Figure 20*. The largest difference can be seen in the last side lobe where the attenuation has increased by more then 4 dB. The cosine factor is a projection and mismatch factor which increases with the angle and is the reason why the outer side lobes are affected the most.

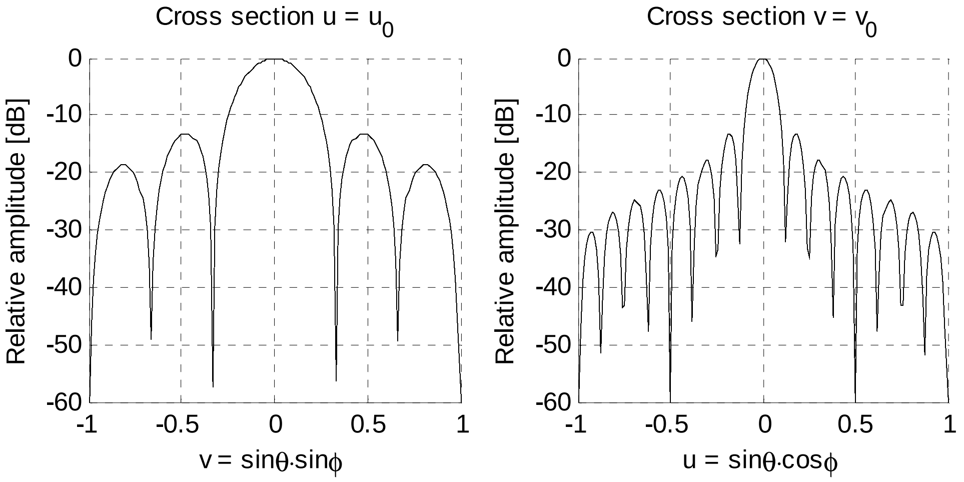


Figure 21. The cross-sections of the radiation pattern for an untapered planar array of 16 × *6 antenna elements with cosine-shaped element patterns.*
Left: u=0, v=sinθ sinϕ   Right: u=sin θ cos ϕ, v=0

If the antenna is electronically tilted, the phase shifters counteract the phase shifts which arise when the radiation is not incident perpendicular to the aperture. This means that the phase shifters are changed so the array is receiving in another direction than the main lobe. As can be seen in *Figure 22*, the main beam is pointed in *u* and *v* equal to 0.3536 which means the antenna is electronically tilted into θ_0 = 30° and ϕ_0 = 45°, according to *Equation 9* and *Equation 10* from the theory part (section 2.1.2). The cross-sections of *u* and *v* of the radiation pattern give a clearer view and can be seen in *Figure 23*.

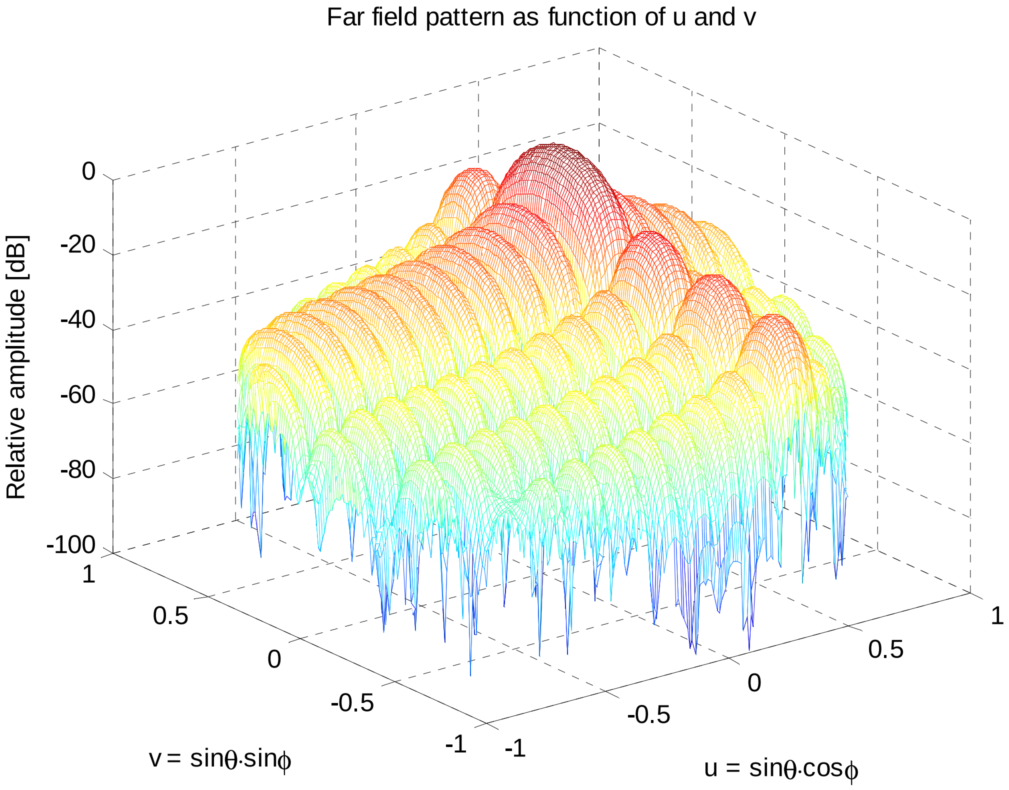


Figure 22. The radiation pattern when the beam is scanned to θ=30º and ϕ=45º.

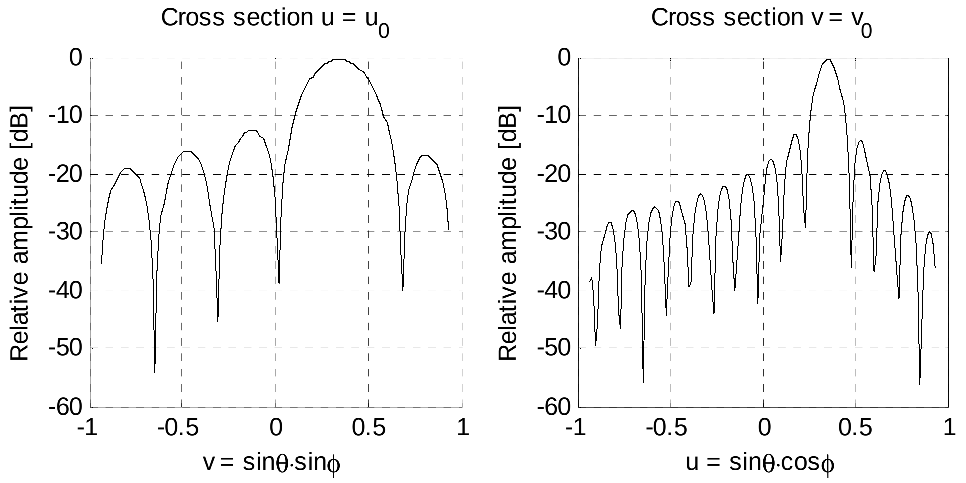


Figure 23. The cross-sections when the beam is scanned to θ=30º and ϕ=45º.
Left: u=0.3536, v=sinθsinϕ Right: u=sinθcosϕ, v=0.3536.

As mentioned in the theory (section 2.1), finite antennas cause side lobes and to decrease the side lobes a tapering can be applied to reduce the radiation from the elements near the edges. This result in lower side lobes but contributes to widen the main beam. To indicate the affect of tapering, a *Taylor 45 dB* tapering in *x*-direction and a *Taylor 30 dB tapering* in *y*-direction is applied and simulated. For both taperings, the number of side lobes with the same height is set to four and the result of the simulation can be seen in *Figure 24*. In the figure significant changes have occurred compared to *Figure 22*; the SLL has decreased and the main beam has become wider. In the left image in *Figure 25*, the cross-section of *u* is presented and since all the side lobes are lowered below -30 dB the desired tapering is obtained. In the right image the cross-section of *v* is presented, showing that the values for the first side lobes are adjusted to the desired tapering value. The outer side lobes, further away from the main beam, are not tapered below -45 dB since the combination of the low tapering level and letting four side lobes have the same height is too hard to achieve. This result in a combination of the two criterions where all side lobes have been attenuated but the ones closest to the main beam are affected the most.

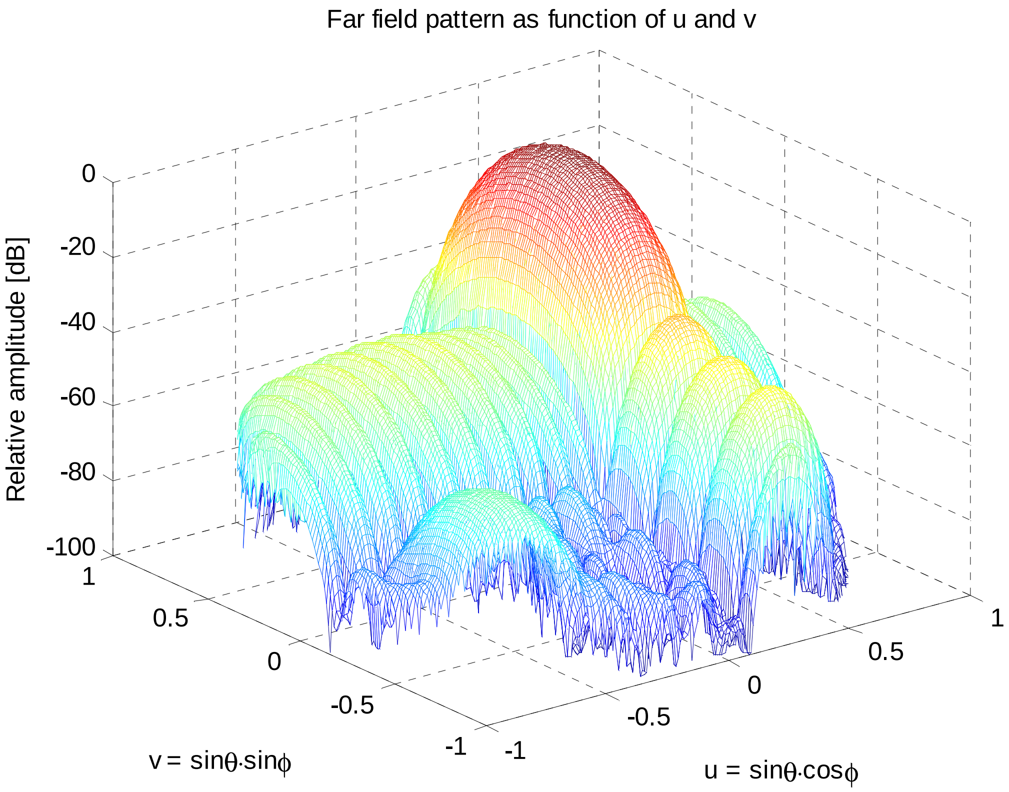


Figure 24. The radiation pattern using a Taylor tapering of -45 dB in x-direction and a -30 dB in y-direction.

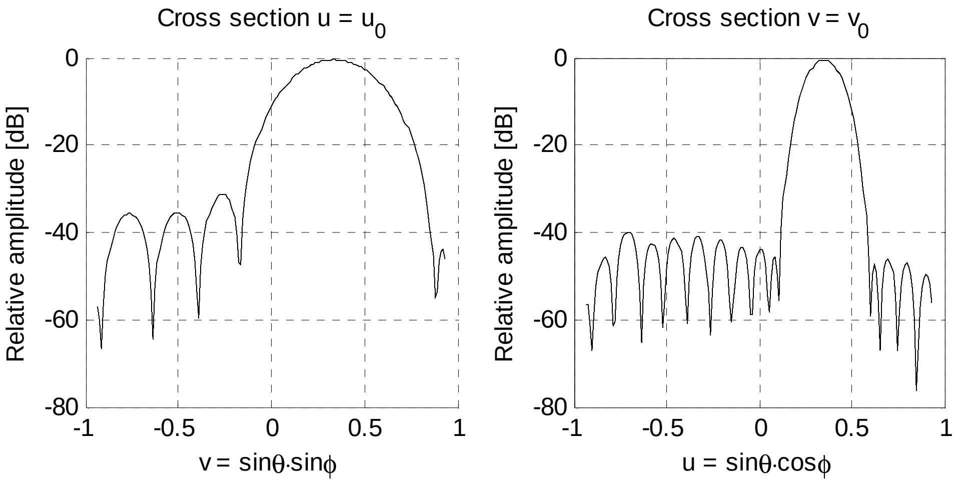


Figure 25. The cross-sections after a Taylor taper has been applied in the attenuator block.
Left: u=0.3536, v=sinθ sinϕ, with a Taylor tapering of -30 dB
Right: u=sinθ cosϕ, v= 0.3536, with a Taylor tapering of -45 dB

To establish the fact that the two models, *Model Based on Drag-and-Drop Boxes* and *Model Composed by Embedded Matlab™ Function Blocks*, can be equally comparable a simulation is done where the two models have the same values of their variables. If comparing *Figure 26* with *Figure 27* they are identical, which means the models are comparable.

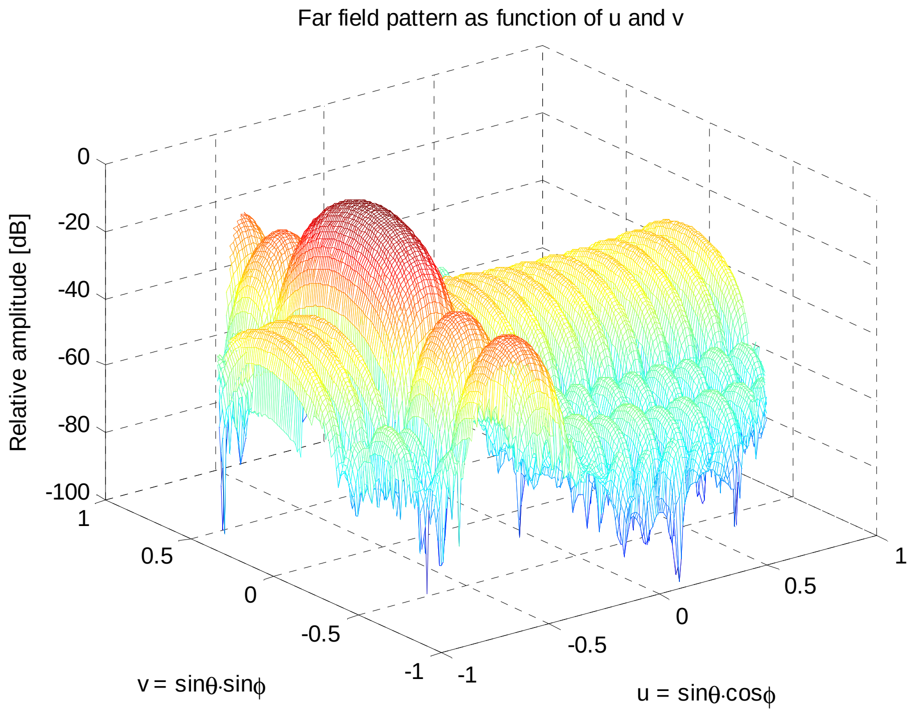


Figure 26. A 3D plot of the Embedded Matlab™ system when steering the main beam into (30°, 170°) having a Taylor tapering of 30 dB in x-direction and a Taylor tapering of 20 dB in y-direction.

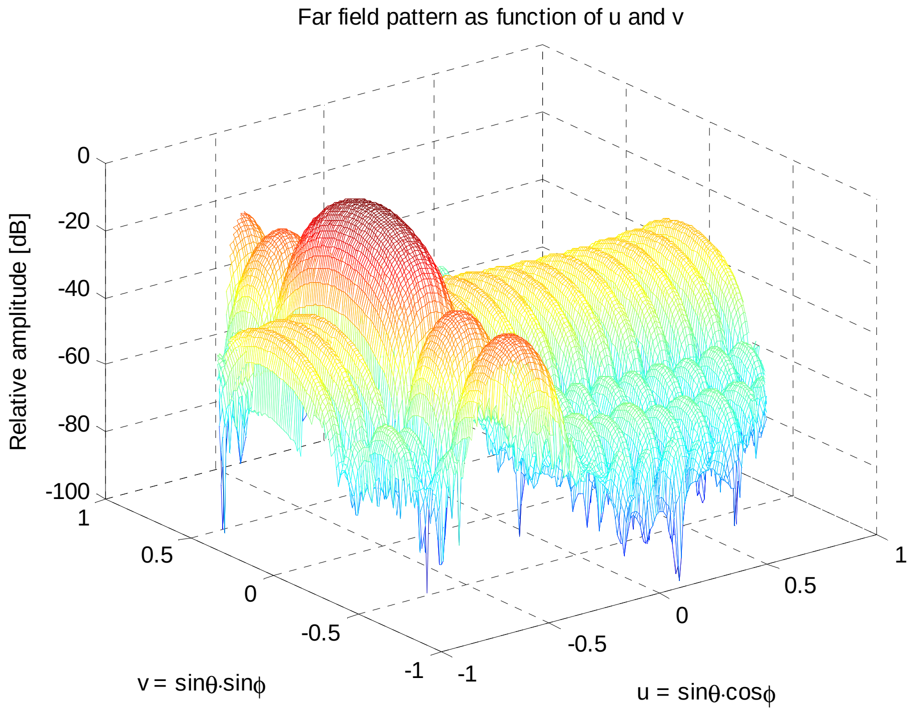


Figure 27. A 3D plot of the Drag-and-Drop system when steering the main beam into (30°, 170°) having a Taylor tapering of 30 dB in x-direction and a Taylor tapering of 20 dB in y-direction.

The performance and accuracy of phase shifters and attenuators depend on the number of bits which the components are manufactured for. In previously simulations, the number of bits was selected to nine, but to illustrate how a small number of bits affect the system, a simulation using five bits for both types of components is done. This low number of bits causes a larger space between each level that the phase shifter can adapt, according to *Equation 16*.

$$phase\ level\ step = \frac{360}{2^{number\ of\ bits}} \Rightarrow \frac{360}{2^5} = 11.25 \quad [°] \qquad \textit{Equation 16}$$

When the main beam is directed into θ_0= 30° and ϕ_0 = 45°, the phase at each antenna element is close to a multiple of the phase level step. Therefore will the influence on the radiation pattern not be so distinct, but as can be seen in *Figure 28*, the small amounts of bits have generated a high side lobe at approximately *u* and *v* equal to -0.5. Having low side lobes makes it easier to distinguish errors, since the magnitude of the errors is more comparable to the levels of the side lobes.

As can be seen in *Figure 29*, the performance of the attenuation of the side lobes is reduced. The tapering is set to attenuate 45 dB for the cross-section of *v*, but the outcome corresponds not even to *Taylor 30*. Since the figures are plotted in dB, errors at lower levels is much more prominent then for higher side lobes, therefore are the phenomena not as prominent in the cross-section of *u* compared to *v*.

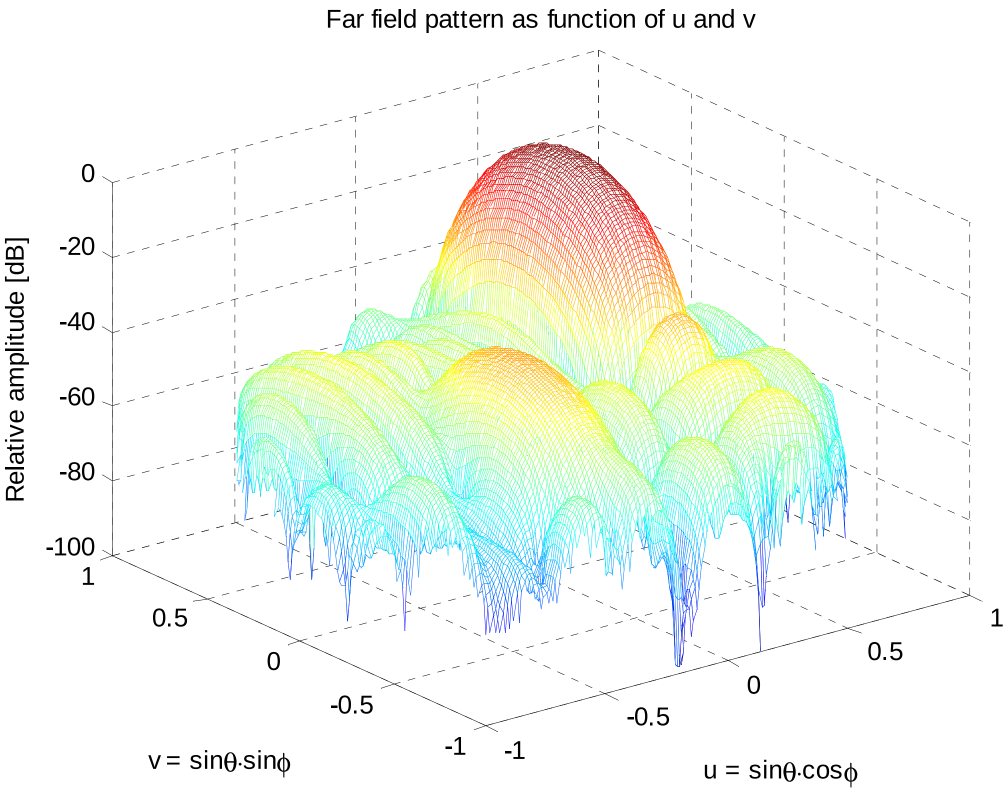


Figure 28. The radiation pattern when having a low number of bits in attenuators and phase shifters, a Taylor tapering of -45 dB in x-direction and -30 dB in y-direction is applied.

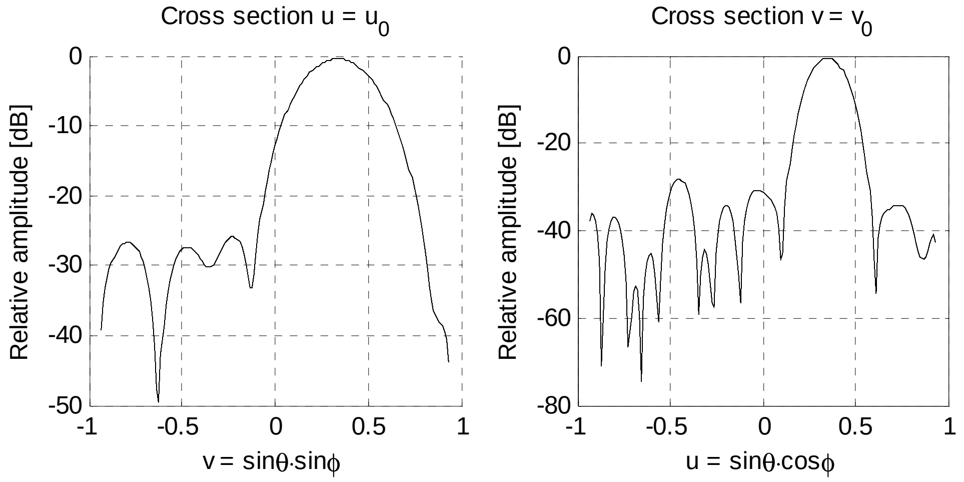


Figure 29. The cross-sections when having a low number of bits.
Left: u=0.3536, v=sinθ sinϕ, with a tapering of -30 dB
Right: u=sinθ cosϕ, v= 0.3536 , with a tapering of -45 dB.

Another issue that affects the performance of the phase shifters and the attenuators are the variation of the components when they are manufactured. To give a clear view of how these fluctuations affects the radiation results the number of bits is reset to nine and instead the standard deviation (*σ*) of the phase and attenuation for the phase shifters and attenuators, respectively, is selected to 0.07. This standard deviation corresponds to a phase of 4° and an attenuation of 0.59 dB. The result of the variations can be seen in *Figure 30* and apart from the *σ* parameter all the variables have the same values as in *Figure 25*.

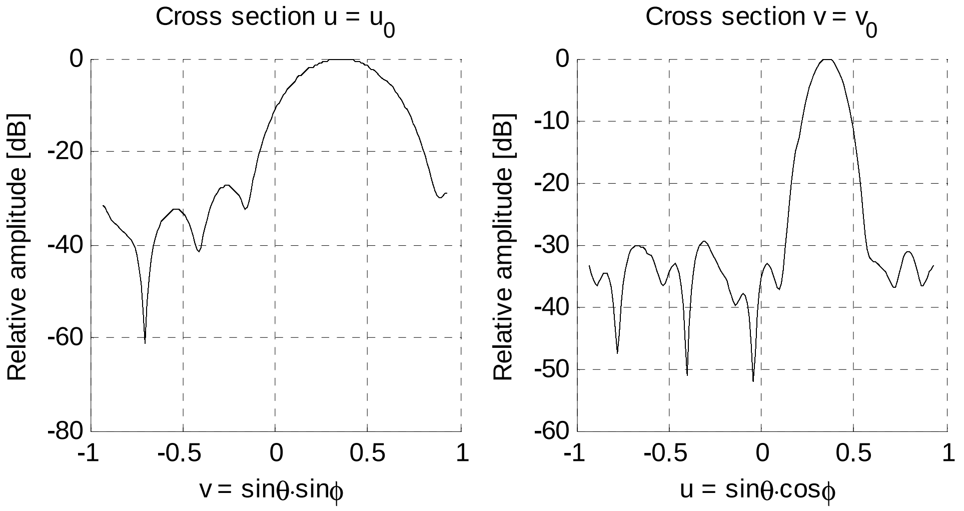


Figure 30. The cross-sections for σ=0.07.
Left: u=0.3536, v=sinθ sinϕ, with a Taylor tapering of -30 dB
Right: u=sinθ cosϕ, v= 0.3536 , with a Taylor tapering of -45 dB.

A simulation having an element spacing of 0.8 λ in both *x*- and *y*-direction is done to illustrate the importance of having a not too big distance between the elements in the array (see section 2.1.1). These large distances between the elements cause grating lobes and as can be seen in *Figure 31,* two have appeared in the visible region when changing the distance from 0.5 λ to 0.8 λ.

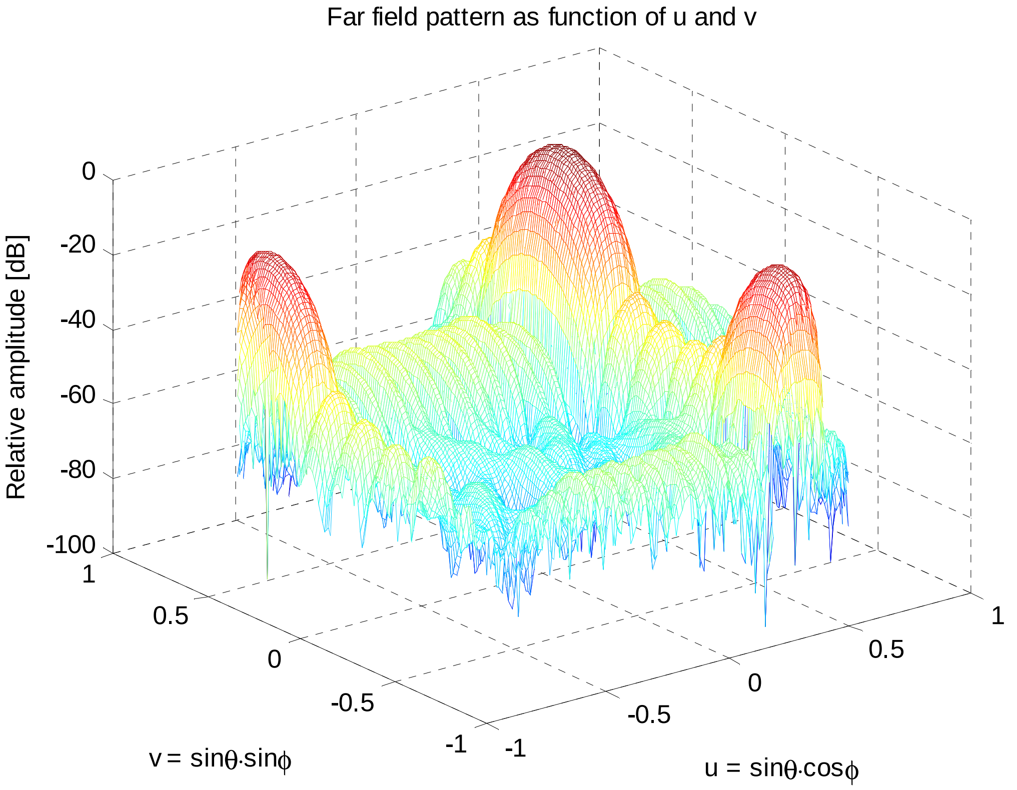


Figure 31. Grating lobes caused by a change in distance, from 0.5 λ to 0.8 λ in both x- and y-directions, between the elements in the array.

4.2 Implementation from Drutten

In this subchapter, the outcome of the model adapted to the receiver antenna demonstrator Drutten is presented. As mentioned earlier, the implementation is made using measurement data from the antenna elements in the demonstrator.

Measurement data from five of the active elements in Drutten, one in the middle and four at the edges, are used to estimate a common element pattern, see *Figure 18*. Since Drutten is built to have its best element pattern in the azimuth angle this is the one used for all of the five elements. The diagrams of the five elements and the estimated function are almost identical and can be seen in *Figure 32*. The patterns from the elements in the middle are very similar to the ones at the edges and therefore are a simplification of using the same diagram for all the elements a good approximation of the true antenna element performance.

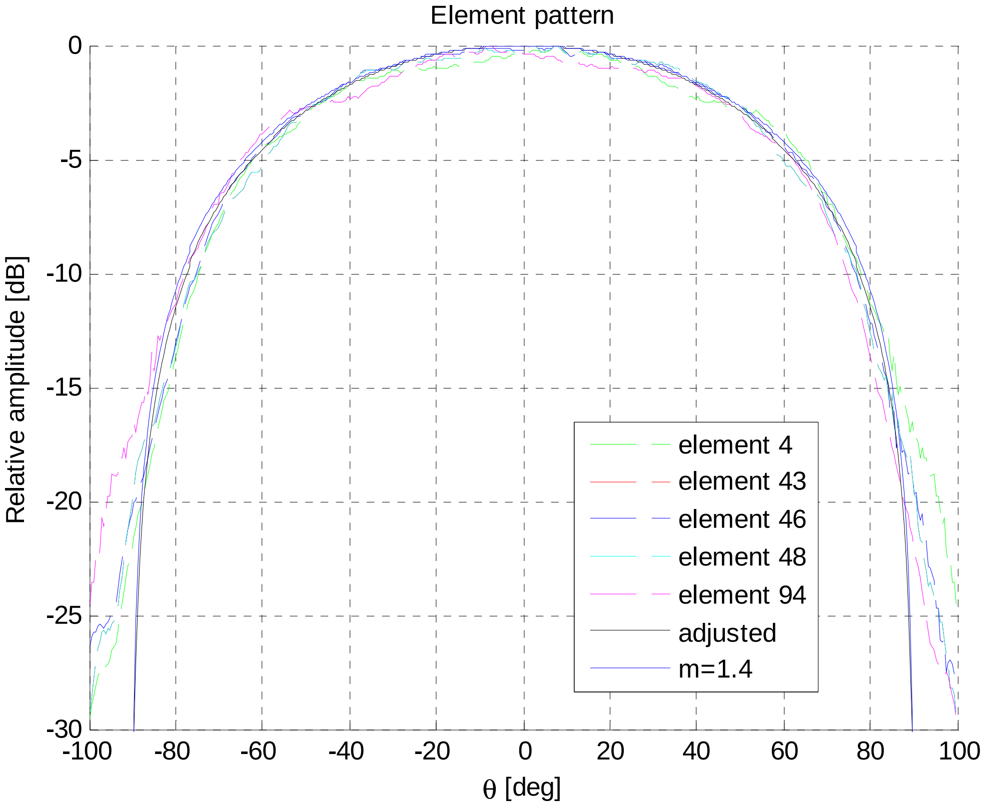


Figure 32. Element patterns from five elements in the demonstrator, the cosine shaped and the adjusted diagram.

The outcome of the *Antenna element* block is illustrated for three different element patterns; isotropic, cosine and adjusted. The latter is the one using the adapted diagram representing the element pattern from the elements in the demonstrator. It can be seen that the radiation pattern using the adjusted element is closer to the radiation pattern using the cosine shaped pattern than the isotropic one, see *Figure 33* and *Figure 34*. The larges difference, between the three different element patterns, can be distinguished at the last side lobes whereas the main beams of the three patterns are identical.

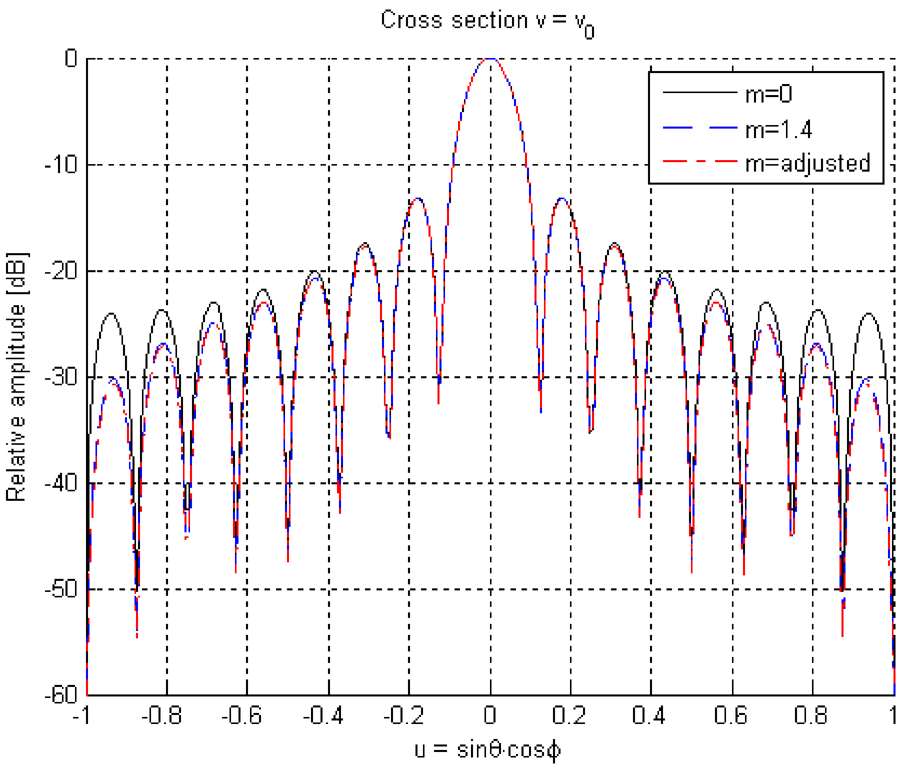


Figure 33. Radiation pattern for cross-section v after antenna element with isotropic, cosine and measured element diagram.

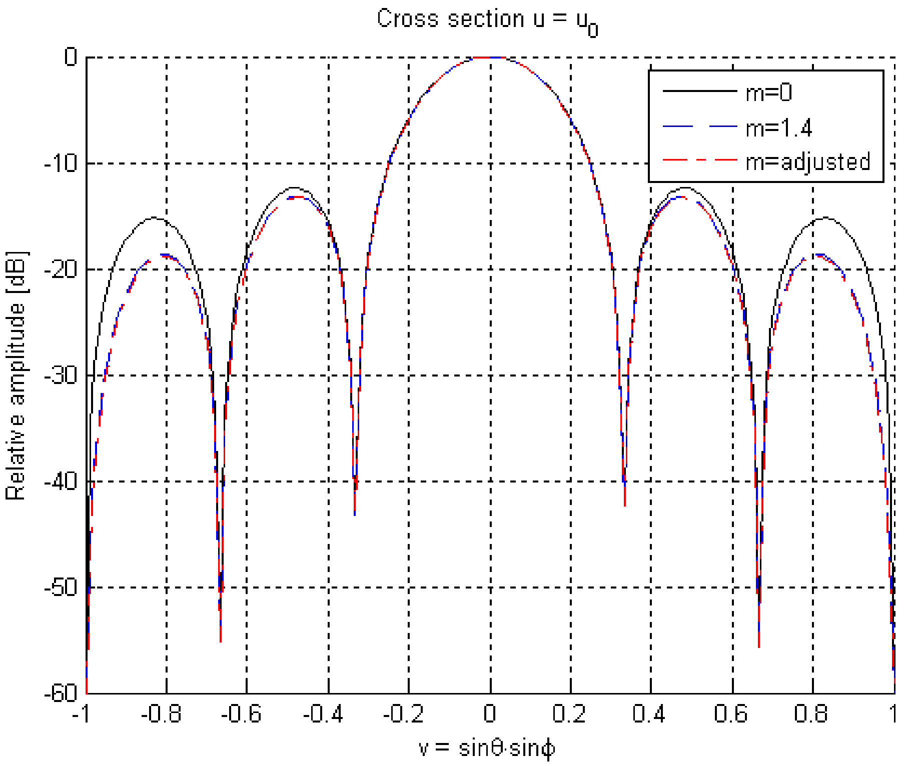


Figure 34. Radiation pattern for cross-section u using an antenna element with isotropic, cosine and measured element diagram.

5 DISCUSSION

In this work, an AESA system is implemented in two ways, both using Simulink® for simulations. The first, *Model Based on Drag-and-Drop Boxes*, is build with graphical code while the latter, *Model Composed by Embedded Matlab™ Function Blocks*, is implemented with textual scripts.

This chapter is made with the purpose to discuss the outcome of the simulations as well as present a comparison between the two different models. There is also a part that brings up the discussion about whether Simulink® is a good simulation tool for these kind of systems and how to continuing in a possible future work.

5.1 ADVANTAGES AND DISADVANTAGES

Even though Simulink® is a user-friendly tool and easy to understand for the first-time user, there are many functions and settings that might complicate things for the user before appreciating the benefits. For the simulations with drag-and-drop boxes, one limitation is the difficulty to implement an arbitrary size m×n system since every box has to be connected to another and the parameters in these boxes are changed by hand. This can be made relatively easy and quickly for a small system with few components, but as the system grows, it becomes more time consuming and the risk for mistakes becomes substantial.

The model with Embedded Matlab™ Function blocks on the other hand is less complicated in the sense of giving a better overview of the system and is easier to change the size of the antenna systems. At first sight, the Embedded Matlab™ Function block gives the impression of having a code identical to the one used in Matlab® and even though there are many resemblances there are several differences as well. The Embedded Matlab™ Function block requires for instance that the size of the output is independent of the input and once a variable is defined, it can not be redefined to any other type or size. Furthermore, a difference from the regular Matlab® program is that the subset does not support matrix deletion. This might be seen as a drawback, but it opens up to reuse of matrixes and the user becomes encouraged to write more efficient code. The feature, the new way of writing code, might seem inconvenient for the experienced Matlab® programmer, since a part of the capacity in Matlab® functions can not be utilized.

Due to the fact that the Embedded Matlab™ Function block does not support some of the common Matlab® functions, alternative solutions to problems have been necessary. The command *cell* is one of the functions not supported by Embedded Matlab™ Function. Instead an alternative solution was created, where a matrix was used to give the impression of functioning as the command cell. In each element position another matrix was positioned, containing the scanned angles in both directions.

Another problem with the drag-and-drop concept has been to create realistic phase shifters and attenuators equal to the ones created in the model composed by embedded blocks. The idea was to use the textual script created for the model with Embedded Matlab™ Function and translate it into graphical code. In this way the drag-and-drop system would be modelled without intervention of Matlab®. Difficulties came up when trying to determine the closest possible phase value compared to the correct phase value. So far, a solution to the graphical translation has not been conceived and therefore, the textual script is used in both models.

One advantage with Simulink® is the generation of error message windows that appears when something in the model is incorrect. The message gives a short description on what is wrong and sometimes even gives a solution to the problem. For the simulations belonging to this thesis there have been error messages often related to *running low on memory* in the Embedded Matlab™ Function version, which can be described as one of the larger problems with this model. The model uses one matrix that is modified throughout the whole system. The size of this matrix depends on the number of elements and the amount of scanned values, which means that it becomes large for arrays with many elements and for radiation patterns with fine resolution. The running low on memory issue is an essential problem since array antennas often are composed by a large number of elements and requires a large amount of values to give an accurate result. To solve the problem, the Embedded Matlab™ Function version was modelled on a computer with higher *Random Access Memory* (RAM).

One of the key features with Simulink® is the ability of sampling. This feature has so far not been taken into consideration, but if, for instance, the angles of θ are sampled and the outcome for each sample is saved, the result can be added together in the end. By using sampling this way, a possible result for the Embedded Matlab™ Function model is that a better and more refined model can be achieved since more values for the angles of θ as well as ϕ can be used without the problem of running low on memory. For the model based on the drag-and-drop concept, this use of sampling can result in, among other things, smaller matrixes.

5.2 FURTHER WORK

Since the model of the AESA is focused on the receive mode, one continuation of this work could be to develop the system such that a simulation in both receive and transmit mode are possible. Furthermore, another step could be to simulate the whole process of a radar; a signal from transmitting mode into receiving mode where the reflecting part is modelled using time sampling.

It is possible to continue to update the model in several ways. In particular all those parts, which have been modelled assuming ideal performance. The model of the antenna elements could for example be expanded to including mismatch, manufacturing differences and mutual coupling to give a more realistic result. The combining network is another part which could be further developed.

There is a possibility to expand the model and implement more components and their features, for example mixers and filters. Further more, aspects like noise temperature can be taken into consideration. Another possibility is to compare more of the measurement data from the demonstrator to see how close to reality the model can get. The model could also be expanded to take into account realistic noise figures, amplifier gain and system losses in order to evaluate the overall noise figure and S/N-ratio.

6 CONCLUSIONS

There are many possibilities when using Simulink® to build models. The program has several different ways to create models and the numerous supplies of tools to perform the modelling are various. During the work, when modelling this large and complex AESA system, there are parts that have been hard to solve. If this is due to restrictions in Simulink® as a program, or if the knowledge about the software is too limited, cannot be firmly declared.

Two models were simulated, one based on the drag-and-drop concept and one composed by Embedded Matlab™ Function blocks, with the intention of having the same functionality. The models have several features that demonstrate the concept of a true AESA system, even though the system could be further developed. The two most important drawbacks with the latter model, is the insufficient flexibility and the large memory requirements. These questions need to be addressed during further developments of the model. Since the first model had problem-free simulations regarding the memory issue, a combination of the two models is likely to result in the best outcome.

Simulink® makes it easy to focus on one component at a time and to develop and improve the functionality gradually. Furthermore, Simulink® is considerably easier to overview than a large Matlab® script because of the feature that the system can be divided into subboxes or Embedded Matlab™ Functions. The larger problems that have arisen so far have been solved, either in Simulink® or by contribution of Matlab®, so generally speaking, the software is a good tool for implementations of this kind.

REFERENCES

Axberg, Stefan. (2005). *Tekniken skriver krigshistorien*. (Electronic) <http://www.foi.se/FOI/templates/Page____3930.aspx> (17 December 2008)

Balanis, Constantine A. (2005). *Antenna Theory: Analysis and Design.* 3rd ed. Hoboken, New Jersey. John Wiley & Sons.

Kildal, Per-Simon (2000). *Foundations of Antennas: A Unified Approach*. Lund, Studentlitteratur.

Lanne, Maria. (2005). *Antenna Arrays System: Electromagnetic and Signal Processing Aspects* Lic. Chalmers University of Technology. Gothenburg, Chalmers Reproservice.

Mathworks™. (2008a). *Embedded MATLAB™ Getting Started Guide*. (Electronic) PDF: <http://www.mathworks.com/access/helpdesk/help/pdf_doc/eml/eml_gs.pdf> p. 15 (9 February 2009)

Mathworks™. (2008b). *Simulink® 7.2*. (Electronic) <http://www.mathworks.com/products/simulink/description1.html> (29 September 2008)

Mathworks™. (2008c). *Simulink® 7 User's Guide*. (Electronic) PDF: <http://www.mathworks.com/access/helpdesk/help/pdf_doc/simulink/sl_using.pdf > pp. 172-178. (9 February 2009)

Mailloux Robert J. (2005) *Phased Array Antenna Handbook*. 2nd ed. Norwood, USA. Artech House

Moernaut, Gerald J.K. & Orban, Daniel L. (2006). *The Basics of Antenna Arrays*. (Electronic) PDF: <http://www.orbanmicrowave.com/The_Basics_of_Antenna_Arrays.pdf > (2 October 2008)

Nilsson Drackner, Peter & Engström, Björn. (2005) An Active Antenna Demonstrator for Future AESA- Systems. In: IEEE International, *Proceedings of the International Radar Conference 2005*. Arlington, VA, USA. 9-12 May, 2005.

Pozar, David M. (2001). *Microwave and RF design of wireless systems*. John Wiley & Sons.

Skolnik, Merrill I. (2001). *Introduction to Radar Systems*. 3rd ed. New York. McGraw-Hill.

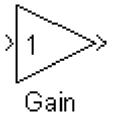
Stimson, George W. (1998). *Introduction to Airborne Radar*. 2nd ed. Mendham, USA. SciTech Publishing.

Visser, Hubregt J. (2005). *Array and Phased Array Antenna Basics.* Chichester, England. John Wiley & Sons.

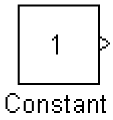
Wennberg, Anna. (2007). *Se, snacka och stör med samma antenn*. (Electronic) <http://etn.se/22546> (2 February 2009)

APPENDIX A: *SIMULINK$^{®}$ BLOCKS*

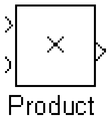
A short presentation of the different block used in the drag-and-drop model.



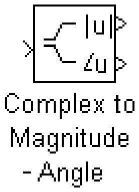
Gain: The gain block multiplies the input by a constant value and accepts a real or complex scalar, vector, or matrix of any data type supported by Simulink$^{®}$ software. If the input of the gain block is real and the gain is complex, the output will also be complex.



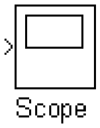
Constant: The constant block generates a real or complex constant value. This can be done with a scalar, vector, or matrix output, depending on the dimensionality of the constant value parameter and the setting of the interpret vector parameters as 1-D parameter. The output of the block has the same dimensions and elements as the constant value parameter.



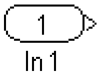
Product: The product block performs multiplication or division of its inputs. This block produces outputs using either element-wise or matrix multiplication, depending on the value of the multiplication parameter. The product block accepts real or complex signals of any data type supported by Simulink$^{®}$ software including fixed-point data types.



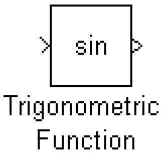
Complex to Magnitude - Angle: The complex to magnitude-angle block accepts a complex-valued signal of type double or single. Its outputs, the magnitude and/or phase angle of the input signal, depend on the setting of the output parameter and are real values of the same data type as the block input. The input can be an array of complex signals, in which case the output signals are arrays. The magnitude signal array contains the magnitudes of the corresponding complex input elements, while the angle output contains the angles of the input elements.



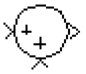
Scope: The scope block displays its input with respect to simulation time. The scope block can have multiple axes (one per port) and all axes have a common time range with independent y-axes. The scope block allows you to adjust the amount of time and the range of input values displayed.



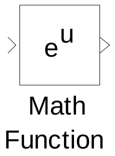
Inport: Inport blocks are the links from outside a system into the system are common in subboxes.



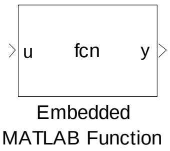
Trigonometric function: The trigonometric function block performs numerous common trigonometric functions as: sin, cos, tan, asin, acos, atan, atan2, sinh, cosh, tanh, asinh, acosh, and atanh. The output of the block is a result of the function using on one or more inputs. The trigonometric function block accepts input signals as real or complex of type double.



Sum block: The sum block performs addition or subtraction on its inputs. This block can add or subtract scalar, vector or matrix inputs, it can also collapse the elements of a signal.



Math function: The Math Function block performs numerous common mathematical functions using exponential, transpose, square root, reciprocal and many more. The block output is the result of the operation of the function on the input or inputs and both real and complex inputs are accepted, except for reciprocal and square root.



Embedded Matlab Function: An Embedded Matlab™ Function block lets the user compose a Matlab function within a Simulink model. The user can specify input and output data to the Embedded Matlab Function block in the function header as arguments and return values. The Embedded Matlab Function block supports a subset of the language for which it can generate efficient embeddable code.[30]

[30] <http://www.mathworks.com/access/helpdesk/help/toolbox/simulink/index.html?/access/helpdesk/help/toolbox/simulink/> (31 mars 2009)

APPENDIX B: *DRAG AND DROP ’GENERATE_PARAMETERS.M’*

```
%% Calculations based on edited parameters ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~

% Transformation from degrees to radians
theta_0_rad = theta_0*(2*pi/360);
phi_0_rad = phi_0*(2*pi/360);

% Input parameters to the simulink model
parameter_vec = [elem_x, elem_y, dx, dy, m, theta_0_rad, phi_0_rad];
theta_vec = linspace(0,pi/2,181);
phi_vec = linspace(0,2*pi,181);

% Taylor tapering
% Used in the attenuator
if sll_x > 0;
    tap_x_vec = taylortap_lin(elem_x,nn_x,sll_x);
else
    tap_x_vec = ones(1,elem_x);
end

if sll_y > 0;
    tap_y_vec = taylortap_lin(elem_y,nn_y,sll_y);
else
    tap_y_vec = ones(1,elem_y);
end

matrix_in = j*ones(elem_y*length(phi_vec),elem_x*length(theta_vec));

%% generate_receiver ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~

if regenerate_modules strcmp(regenerate_modules,'true');
  % LNA
  % Create one unique phase shifter for each antenna element using a normal
  % distributed random function.

  % Phase shifter
  % Create one unique phase shifter for each antenna element using a normal
  % distributed random function.

  exist_phase = zeros(length(elem_y),length(elem_x));
  for x = 1:elem_x
     for y = 1:elem_y
        wanted_phase(y,x) = 2*pi*sin(theta_0_rad)*
                         ((x-1)*dx*cos(phi_0_rad)+(y-1)*dy*sin(phi_0_rad));
        if wanted_phase(y,x) > 2*pi || wanted_phase(y,x) < -(2*pi)
           wanted_phase(y,x) = mod(wanted_phase(y,x),2*pi);
        end
        if wanted_phase(y,x) < 0
           wanted_phase(y,x) = wanted_phase(y,x) + 2*pi;
        end
     end
  end

    ideal_vec = linspace(0,2*pi,2^phase_bit+1);

    for x = 1:elem_x
```

```
           for y = 1:elem_y
               [rte,index] = min(abs(ideal_vec - wanted_phase(y,x)));
               error_vec = mu_phase_shifter + sigma_phase_shifter.*randn(1);
               exist_phase(y,x) = ideal_vec(index) + error_vec;
           end
    end

    pos = [find(exist_phase < 0)];
    for z = 1:length(pos)
        exist_phase(pos(z)) = 0;
    end
    pos = [find(exist_phase > 2*pi)];
    for z = 1:length(pos)
        exist_phase(pos(z)) = 2*pi;
    end

    clear ideal_vec error_vec x y pos;

    % Attenuator
    % Create one unique attenuator for each antenna element using a normal
    % distributed random function

    exist_tapering = zeros(length(elem_y),length(elem_x));
    wanted_tapering = tap_y_vec'*tap_x_vec;
    ideal_vec = linspace(0,1,2^attenuation_bit+1);

    for x = 1:elem_x
        for y = 1:elem_y
            [rte,index] = min(abs(ideal_vec - wanted_tapering(y,x)));
            error_vec = mu_attenuator + sigma_attenuator.*randn(1);
            exist_tapering(y,x) = ideal_vec(index) + error_vec;
        end
    end

    pos = [find(exist_tapering < 0)];
    for z = 1:length(pos)
        exist_tapering(pos(z)) = 0;
    end
    pos = [find(exist_tapering > 2*pi)];
    for z = 1:length(pos)
        exist_tapering(pos(z)) = 2*pi;
    end

    clear ideal_vec error_vec x y pos;
end

exist_tapering = exist_tapering';
exist_phase = exist_phase';
clear bits attenuator x y p circular_attenuator min_value max_value;
clear wanted_tapering;
```

Appendix C: *Drag and Drop ’Edit_Parameters.m’*

```
clc;

% Edit parameters for simulation
% Press SAVE before returning to the simulink model.

% Antenna geometry
dx = 0.5; % Element separation in x direction [lambda]
dy = 0.5; % Element separation in y direction [lambda]
p_theta = 181; % The number of points in theta direction
p_phi = 181; % The number of points in phi direction
m = 0; % Factor used to raise cosinus, in this model it has to be zero

% Phase shifter
theta_0 = 0; % Steering in theta direction, range: 0-90 [degrees]
phi_0 = 0; % Steering in phi direction, range: 0-360 [degrees]
mu_phase_shifter = 0; % Mean for normal distribution
sigma_phase_shifter = 0.00; % Standard deviation for normal distribution
phase_bit = 9; % Number of bits

% Attenuator
mu_attenuator = 0; % Mean for normal distribution
sigma_attenuator = 0.00; % Standard deviation for normal distribution
attenuation_bit = 9; % Number of bits
sll_x = 0; % Reduce side lobe level in x direction, set to zero if
untapered [dB]
sll_y = 0; % Reduce side lobe level in y direction, set to zero if
untapered [dB]
nn_x = 4; % Number of affected nulls
nn_y = 4; % Number of affected nulls



% Set 'true' if new receiver modules are wanted, 'false' if latest modules
% should be used. First time the parameter should be set to 'true'.
regenerate_modules = 'true';

% SAVE and return to the Simulink model

%%
% Fixt in the drag-and-drop model
elem_x = 16; % The total number of antenna elements in x direction
elem_y = 6;  % The total number of antenna elements in y direction
run('H:\Simulink\Slutgiltiga Simulinkmodeller\phase_attenuation.m');
```

APPENDIX D: *EMBEDDED MATLAB™ FUNCTION BLOCKS*

```
% ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
% Plane wave block
% The plane wave is distributed over the different antenna elements of the
array

function out  = field(matrix_in, theta_vec, phi_vec, parameter_vec)

% Making theta and phi- values to row vectors
theta = theta_vec;
phi = phi_vec;

% Parameter vector
tot_x = parameter_vec(1);
tot_y = parameter_vec(2);
dx = parameter_vec(3);
dy = parameter_vec(4);
% m = parameter_vec(5);
% theta_0 = parameter_vec(6);
% phi_0 = parameter_vec(7);

a = 0;
b = 0;
for y = 1:tot_y
    for x = 1:tot_x
        % Calculation for the incoming field for one element
        exponential = exp((((y-1)*dy*sin(phi))'+
                      ((x-1)*dx*cos(phi))')*(2j*pi*(sin(theta))));
        % Saving the features for the antenna element in the matrix
        matrix_in(b+(1:length(phi)),a+(1:length(theta))) = exponential;
        a = a + length(theta);
    end
    a = 0;
    b = b + length(phi);
end

% Parameters and matrixes out
out = matrix_in;
theta_out = theta;
phi_out = phi;
parameters_out = parameter_vec;



% ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
% Antenna element block
function out  = antenna_element(in, theta_vec, phi_vec, parameter_vec)

matrix_in = in;
theta = theta_vec;
phi = phi_vec;

% Parameter vector
tot_x = parameter_vec(1);
tot_y = parameter_vec(2);
% dx = parameter_vec(3);
% dy = parameter_vec(4);
```

```
m = parameter_vec(5);
% theta_0 = parameter_vec(6);
% phi_0 = parameter_vec(7);

% Two eligible element patterns where the second is the adjusted
% an = sqrt(cos(theta).^complex(m));
% approximated element pattern
an = 1.7*sqrt((cos(theta)).^(2.2));

antenna = ones(length(phi),1) * an;

a = 0;
b = 0;
for y = 1:tot_y
    for x = 1:tot_x
        matrix_in(b+(1:length(phi)),a+(1:length(theta))) =
matrix_in(b+(1:length(phi)),a+(1:length(theta))).*antenna;
        a = a + length(theta);
    end
    a = 0;
    b = b + length(phi);
end

out = matrix_in;



% ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
% LNA block
function out  =  LNA(in, theta_vec, phi_vec, parameter_vec)

matrix_in = in;
theta = theta_vec;
phi = phi_vec;

% Parameter vector
% tot_x = parameter_vec(1);
% tot_y = parameter_vec(2);
% dx = parameter_vec(3);
% dy = parameter_vec(4);
% m = parameter_vec(5);
% theta_0 = parameter_vec(6);
% phi_0 = parameter_vec(7);

% Parameters and matrixes out
out = matrix_in;



% ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
% Phase shifter block
function out  = phase_shifter(in, theta_vec, phi_vec, parameter_vec,
exist_phase)

matrix_in = in;
theta = theta_vec;
phi = phi_vec;

% Parameter vector
tot_x = parameter_vec(1);
tot_y = parameter_vec(2);
```

```
% dx = parameter_vec(3);
% dy = parameter_vec(4);
% m = parameter_vec(5);
% theta_0 = parameter_vec(6);
% phi_0 = parameter_vec(7);

a = 0;
b = 0;
for y = 1:tot_y
    for x = 1:tot_x
        % The phase shifts are multiplied with each element
        phase = exp(-1j*exist_phase(y,x));
        matrix_in(b+(1:length(phi)),a+(1:length(theta))) =
        phase.*matrix_in(b+(1:length(phi)),a+(1:length(theta)));
        a = a + length(theta);
    end
    a = 0;
    b = b + length(phi);
end

% Parameters and matrixes out
out = matrix_in;



% ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
% Attenuator block
function out = attenuator(in, theta_vec, phi_vec, parameter_vec,
exist_attenuator)

matrix_in = in;
theta = theta_vec;
phi = phi_vec;

% Parameter vector
tot_x = parameter_vec(1);
tot_y = parameter_vec(2);
% dx = parameter_vec(3);
% dy = parameter_vec(4);
% m = parameter_vec(5);
% theta_0 = parameter_vec(6);
% phi_0 = parameter_vec(7);

a = 0;
b = 0;

for y = 1:tot_y
    for x = 1:tot_x
        % The taperings are multiplied with each element
        matrix_in(b+(1:length(phi)),a+(1:length(theta))) =
  exist_attenuator(y,x).*matrix_in(b+(1:length(phi)),a+(1:length(theta)));
        a = a + length(theta);
    end
    a = 0;
    b = b + length(phi);
end

% Parameters and matrixes out
out = matrix_in;
```

```
% ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
% Combining network block
function comb_network (in, theta_vec, phi_vec, parameter_vec)

eml.extrinsic('figure','subplot','mesh','plot','ylabel','xlabel','zlabel');
eml.extrinsic('title','axis','plot','surf','clear','hold','griddata');

matrix_in = in;
theta = theta_vec';
phi = phi_vec';

% Parameter vector
tot_x = parameter_vec(1);
tot_y = parameter_vec(2);
% dx = parameter_vec(3);
% dy = parameter_vec(4);
% m = parameter_vec(5);
theta_0 = parameter_vec(6);
phi_0 = parameter_vec(7);

t = complex(zeros(length(phi),length(theta)));

a = 0;
b = 0;
for y = 1:tot_y
    for x = 1:tot_x
        % Summing up the signals from each element
        t = t + matrix_in(b+(1:length(phi)),a+(1:length(theta)));
        a = a + length(theta);
    end
    a = 0;
    b = b + length(phi);
end


abs_input = abs(t);
norm_abs_input = abs_input./max(max(abs_input));
output = 20*log10(norm_abs_input);

matrix_theta = repmat(theta,1,length(phi))';
matrix_phi = repmat(phi,1,length(theta));
% (theta,phi)
theta_deg = matrix_theta.*(360/(2*pi));
phi_deg = matrix_phi.*(360/(2*pi));

% (u,v)
u = sin(matrix_theta).*cos(matrix_phi);
v = sin(matrix_theta).*sin(matrix_phi);

[u_new v_new] = meshgrid(linspace(-1,1,length(theta)),linspace(-
1,1,length(phi)));
output_new = griddata(u,v,output,u_new,v_new,'cubic');

u_0 = sin(theta_0)*cos(phi_0);
v_0 = sin(theta_0)*sin(phi_0);

cross_u = round(((u_0+1)/2)*length(theta));
cross_v = round(((v_0+1)/2)*length(phi));
```

```
% 3D plot with theta, phi and u,v (u=sin(theta)*cos(phi),
v=sin(theta)*sin(phi))
figure(1);
subplot(2,2,1);
mesh(theta_deg,phi_deg,output);
title('Beam function as a function of \theta and \phi');
ylabel('\phi [deg]');
xlabel('\theta [deg]');
zlabel('Relative amplitude [dB]');

subplot(2,2,2);
mesh(u_new,v_new,output_new);
title('Far field pattern as function of u and v');
xlabel('u = sin\theta\cdotcos\phi');
ylabel('v = sin\theta\cdotsin\phi');
zlabel('Relative amplitude [dB]');

output = output_new;
subplot(2,2,3);
plot(u_new',output(cross_v,:),'k');
title('Cross section v = v_0');
xlabel('u = sin\theta\cdotcos\phi');
ylabel('Relative amplitude [dB]');

subplot(2,2,4);
plot(v_new,output(:,cross_u),'b')
title('Cross section u = u_0')
xlabel('v = sin\theta\cdotsin\phi');
ylabel('Relative amplitude [dB]');
```

APPENDIX E: *EMBEDDED MATLAB™ ‘EDIT_PARAMETERS.M’*

```
%clearvars -except phase_shifter exist_phase exist_attenuator, clc, clf;
close(figure(1))

%% Edit parameters
% Edit parameters for simulation
% Press SAVE before returning to the simulink model.

% Antenna geometry
tot_x = 16; % The total number of antenna elements in x direction
tot_y = 6; % The total umber of antenna elements in y direction
dx = 0.5; % Element separation in x direction [lambda]
dy = 0.5; % Element separation in y direction [lambda]
m = 1.4; % Factor used to raise cosinus, i.e. sqrt(cos^m)

% Phase shifter
theta_0_deg = 0; % Steering in theta direction, range: 0-90 [degrees]
phi_0_deg = 0; % Steering in phi direction, range: 0-360 [degrees]
mu_phase_shifter = 0; % Mean for normal distribution
sigma_phase_shifter = 0.0; % Standard deviation for normal distribution
phase_bit = 9; % Number of bits

% Attenuator
mu_attenuator = 0; % Mean for normal distribution
sigma_attenuator = 0.0; % Standard deviation for normal distribution
attenuation_bit = 9; % Number of bits
sll_x = 0; % Reduce side lobe level in x direction, set to zero if
untapered [dB]
sll_y = 0; % Reduce side lobe level in y direction, set to zero if
untapered [dB]
nn_x = 4; % Number of affected nulls
nn_y = 4; % Number of affected nulls

% Simulation parameters
% If a cross section is wanted, set one of the variables equal to one.
points_theta = 181; % Number of points in theta vector
points_phi = 181; % Number of points in phi vector

% Set 'true' if new receiver modules are wanted, 'false' if latest modules
% should be used.
regenerate_modules = 'true';

% SAVE and return to the Simulink model

run('/home/emwawik/matlab/jenny_josefina/generate_parameters.m');
```

APPENDIX F: *EMBEDDED MATLAB™ ‘GENERATE_PARAMETERS.M’*

```
%% Calculations based on edited parameters ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
% Transformation from degrees to radians
theta_0 = theta_0_deg*(2*pi/360);
phi_0 = phi_0_deg*(2*pi/360);

% Input parameters to the simulink model
parameter_vec = [tot_x, tot_y, dx, dy, m, theta_0, phi_0];
theta_vec = linspace(0,pi/2,points_theta);
phi_vec = linspace(0,2*pi,points_phi);

% Taylor tapering
% Used in the attenuator
if sll_x > 0;
    tap_x_vec = taylortap_lin(tot_x,nn_x,sll_x);
else
    tap_x_vec = ones(1,tot_x);
end

if sll_y > 0;
    tap_y_vec = taylortap_lin(tot_y,nn_y,sll_y);
else
    tap_y_vec = ones(1,tot_y);
end

matrix_in = j*ones(tot_y*length(phi_vec),tot_x*length(theta_vec));

%
clear nn_x nn_y phi_0_deg theta_0_deg points_phi points_theta sll_x sll_y;

%% generate_receiver ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~

if regenerate_modules strcmp(regenerate_modules,'true');
  % LNA
  % Create one unique phase shifter for each antenna element using a normal
  % distributed random function.

  % Phase shifter
  % Create one unique phase shifter for each antenna element using a normal
  % distributed random function.

  exist_phase = zeros(length(tot_y),length(tot_x));
  for x = 1:tot_x
     for y = 1:tot_y
         wanted_phase(y,x) = 2*pi*sin(theta_0)*((x-1)*dx*cos(phi_0)+
                             (y-1)*dy*sin(phi_0));
         if wanted_phase(y,x) > 2*pi || wanted_phase(y,x) < -(2*pi)
             wanted_phase(y,x) = mod(wanted_phase(y,x),2*pi);
         end
         if wanted_phase(y,x) < 0
             wanted_phase(y,x) = wanted_phase(y,x) + 2*pi;
         end
      end
  end

  ideal_vec = linspace(0,2*pi,2^phase_bit+1);
  for x = 1:tot_x
      for y = 1:tot_y
          [rte,index] = min(abs(ideal_vec - wanted_phase(y,x)));
          error_vec = mu_phase_shifter + sigma_phase_shifter.*randn(1);
          exist_phase(y,x) = ideal_vec(index) + error_vec;
      end
  end

  pos = [find(exist_phase < 0)];
  for z = 1:length(pos)
      exist_phase(pos(z)) = 0;
  end
  pos = [find(exist_phase > 2*pi)];
  for z = 1:length(pos)
      exist_phase(pos(z)) = 2*pi;
  end

  clear ideal_vec error_vec x y pos;

  % Attenuator
  % Create one unique attenuator for each antenna element using a normal
  % distributed random function

  exist_attenuator = zeros(length(tot_y),length(tot_x));
  wanted_tapering = tap_y_vec'*tap_x_vec;
  ideal_vec = linspace(0,1,2^attenuation_bit+1);

  for x = 1:tot_x
      for y = 1:tot_y
          [rte,index] = min(abs(ideal_vec - wanted_tapering(y,x)));
          error_vec = mu_attenuator + sigma_attenuator.*randn(1);
          exist_attenuator(y,x) = ideal_vec(index) + error_vec;
      end
  end

  pos = [find(exist_attenuator < 0)];
  for z = 1:length(pos)
      exist_attenuator(pos(z)) = 0;
  end
  pos = [find(exist_attenuator > 2*pi)];
  for z = 1:length(pos)
      exist_attenuator(pos(z)) = 2*pi;
  end

  clear ideal_vec error_vec x y pos;
end

% exist_attenuator = exist_attenuator';
% exist_phase = exist_phase';
clear bits attenuator x y p circular_attenuator min_value max_value;
clear wanted_tapering;
```

APPENDIX G: *MATLAB SCRIPT ‘TAYLORTAP_LIN’*

```
function excitering=taylortap_lin(length,nn,sl)
%TAYLORTAP_LIN  Beräknar exciteringar för linjär gruppantenn med lika
%avstånd mellan antennelementen
%
%   TAYLORTAP_LIN(length,nn,sl)
%       length: antal element
%       nn: antal lika stora sidlober innan dämpningen ökas
%       sl:hur många dB första sidloben ska vara dämpad
%   EXEMPEL:EX=taylortap_lin(41,6,25)
%       beräknar exciteringen för 41 element
%       där de 6 första sidloberna på varje sida är dämpade 25 dB i
%       förhållande till huvudloben. Sedan dämpas efterföljande
%       sidlober ytterligare med sin(x)/x. Resultatet ges som
%       utparameter i vektorn EX.
```