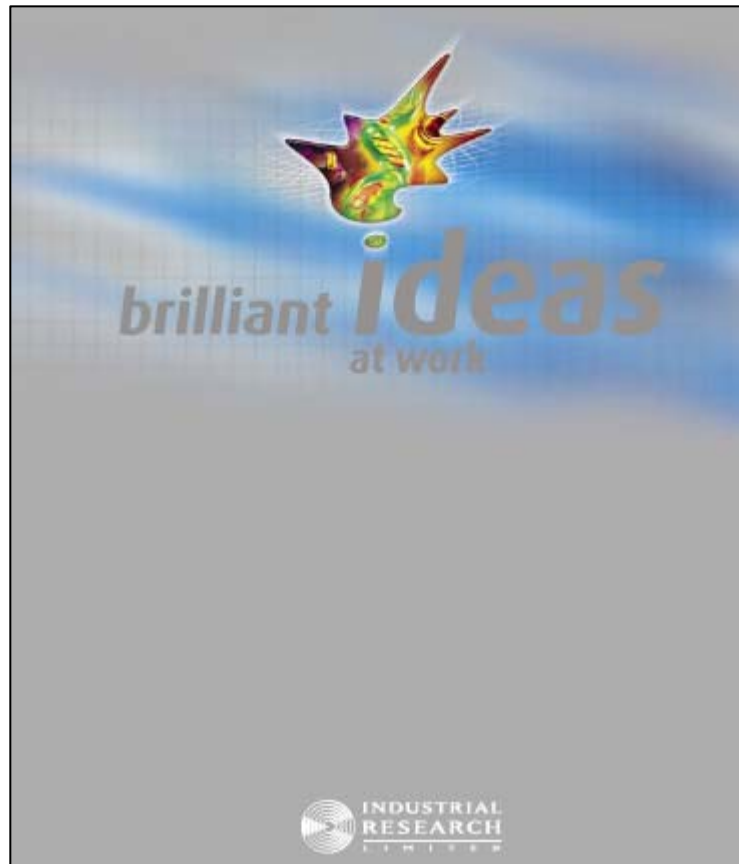


Work Report for Industrial Research Limited



Employer Details:

Name of Company : Industrial Research Limited
Address : Gracefield Research Centre, Gracefield Road, PO Box 31310, Lower Hutt,
Wellington, New Zealand.
Telephone : (04) 9313000
Supervisor : Mr. Neil Scott – Team Leader, Communications and Sensors
Department : Communication, Computing and Electronic Technologies –
Communications and Sensors Team

Employee Details:

Name : Thusitha Dananjaya De Silva Mabotuwana
Student ID : 9790416
Department : Electrical and Electronic / Computer Systems

Work Period : 01st December 2003 – 14th February 2004
Number of hours completed: Sub Professional - 430 hours
Date of Report : 14th February 2004

Information provided in this report contains confidential information of Industrial Research Limited and therefore should not be used for any purpose other than what it is intended for. Any part of this report should not be reproduced without the written consent of the author and Industrial Research Limited.

Summary

In order to gain the required practical work experience for the completion of my degree, I worked for Industrial Research Limited (IRL) in Wellington during the university summer vacation in 2003. The main objective of my involvement with IRL was to develop multiple-input-multiple-output (MIMO) radio channel simulator modules which take various scenarios, such as scattering environment, antenna configuration and so on into account and produce appropriate channel matrices.

Fourteen narrowband channel scenarios based on the *Kronecker Model* and two wideband channel models, with one of them only partly implemented due to the time constraints, have successfully been implemented in the final solution. All these modules have been coded in MATLAB and integrated into one main module thereby showing how the individual modules can be used later-on, as the team decides to take the project further.

This report gives a detailed overview of IRL, company activities, its people, commercial relationships, research interests, working atmosphere and facilities and a description of the work I was involved in. It concludes with general comments and conclusions.

Table of Contents

Summary	I
Table of Contents	II
Acknowledgement	III
Glossary of Terms.....	IV
List of Figures.....	V
1.0 Introduction	1
1.1 Introduction to Industrial Research Limited.....	1
1.2 What is Covered in this Report	1
2.0 Industrial Research Limited	1
2.1 Departments and Research Interests.....	1
2.1.1 Advanced Materials and Performance	1
2.1.2 Intelligent Devices and Systems	2
2.1.3 Communication, Computing and Electronic Technologies	2
2.1.4 Sustainable and Distributed Energy Technologies	2
2.1.5 Biochemical Technologies.....	3
2.1.6 Complex Measurements and Analysis.....	3
2.2 Company Mission, Business Approach and Values	4
2.3 Production Facilities and Structure	4
2.4 Management structure	5
2.5 Employee Relations, Work Conditions and Company Culture	5
3.0 Work I was Involved in.....	6
3.1 Introduction to MIMO Systems and Nature and Background of the Project	6
3.2 Development of the MIMO Simulator	6
3.3 MATLAB Command Line Interface and Introduction to the Simulator	7
3.4 Generation of Transmit and Receive Correlation Matrices for Different Scenarios	10
3.4.1 LOS model with ULA or user specified antenna placements	10
3.4.2 Rayleigh Models with Uniform, Gaussian or Laplacian Distributed Angular Spectra	11
3.4.3 Rician Models with Uniform, Gaussian and Laplacian Distributed Angular Spectrums.....	12
3.4.4 Wideband Channel Models.....	13
3.5 Known Bugs and Drawbacks	13
3.6 Suggested Future Developments	14
4.0 Conclusions	14
References	15
<u>Appendices</u>	
Appendix 1: Mission Statement.....	II
Appendix 1I: Professional Code of Ethics.....	III
Appendix 1II: Building Layout at Gracefield Research Centre.....	IV
Appendix IV: MATLAB Code Written for main.m	V
Appendix V: MATLAB Code Written to Implement Various MIMO Channel Models.....	XII

Acknowledgement

Working for IRL during the university summer break has been a great privilege and I would like to thank Mr. Neil Scott, Team Leader of Communications and Sensors for giving me this great opportunity. It has not only helped me gain knowledge on MIMO and wireless communication systems, but also given me an insight to some of the real engineering activities. I would like to take this opportunity to thank all IRL staff, especially Neil Scott, Miles Leonard-Taylor, Sudhir Singh and Mrs. Dale Randall for all the help and encouragement they gave me throughout my involvement with IRL.

Special thanks to Neil and Sudhir for their valuable time spent explaining certain concepts to me and all the patience and tolerance they had when this process had to be repeated quite a few times whenever I found certain concepts hard to come to grips with.

I wish the company all the best in all its future endeavours.

Glossary of Terms

MIMO – Multiple Input Multiple Output

PDP – Power Delay Profile

FIR – Finite Impulse Response

LOS – Line Of Sight

AoA – Angle of Arrival

ULA – Uniform Linear Array

AS – Angular Spectrum

i.i.d – Independent and Identically Distributed

GUI – Graphical User Interface

List of Figures

Figure 01: A community level integrated distributed energy system funded by IRL [1].	3
Figure 02: MATLAB command line interface.....	7
Figure 03: Flowchart showing how instances of H can be generated [modified from 3].	8
Figure 04: Different multipath scenarios for the wideband models.....	9
Figure 05: Flowchart showing the basic information flow in the simulator	9
Figure 06: Required user inputs for LOS, ULA scenario.....	10
Figure 07: Required user inputs for a Rayleigh, ULA, uniform distribution scenario with two clusters.....	12

1.0 Introduction

1.1 Introduction to Industrial Research Limited

Founded in 1940, Industrial Research limited (IRL) is an innovation-focused commercial company with the New Zealand Government as the main shareholder. The company focuses mainly on six frontline areas of technology advancement, namely Advanced Materials and Performance; Communication, Computing and Electronic Technologies; Intelligent Devices and Systems; Biochemical Technologies; Sustained and Distributed Energy Technologies and Complex Measurements and Analysis. The main campus is based in Gracefield Research Centre in Wellington and employees over 400 personnel, while the other campuses are located in Auckland, Christchurch, Australia, Singapore and United Kingdom. The annual earnings of the company which include a mix of onshore and offshore revenues are around NZ\$60 million [1].

1.2 What is Covered in this Report

This report gives a detailed overview of the company activities, its people, commercial relationships, research interests, working atmosphere and facilities and a summary of the work I was involved in during the summer vacation in 2003, followed by general comments about the work from a subjective point of view and conclusions. It contains some of the confidential information pertaining to IRL and therefore should not be used for any purpose other than what it is intended for. Any part of this report should not be reproduced without the written consent of the author and IRL.

2.0 Industrial Research Limited

2.1 Departments and Research Interests

As mentioned in Section 1.1, IRL's main research interests are six fold and the research activities are carried out by the individual departments. This section gives a brief overview of these departments and their main interests along with some examples of the commercialised developments.

2.1.1 Advanced Materials and Performance

The main objective is to innovate lighter, smaller, more efficient and robust materials with longer lives. IRL is currently in the process of developing high-temperature superconducting coils and wires for generators, motors, current limiters and transformers. Company is also interested in developing specialised air-conditioning units which can withstand the long term vibration of trains and severe shock loads of military vessels [1]. Some of the other related areas of interest are:

- Advanced ceramics and electroceramics
- Polymers, ultra violet degradation of materials and composites
- Cement, concrete and building materials
- Plant life extension
- Condition monitoring, risk based inspection and failure analysis
- Engineering dynamics and shock and vibration testing
- Composite engineering and biomechanics

2.1.2 Intelligent Devices and Systems

This department focuses mainly on combining artificial intelligence and automation with knowledge-based systems and sensors. IRL has been able to develop a low cost melanoma imaging system to help earlier detection of melanomas which has turned out to be a huge success. Last year, a unique card inspection device based on machine vision technology was developed for casinos which can automatically check eight decks of cards every two minutes [1]. Some other interests of this department are:

- Microwave sensing and processing
- X-Ray inspection
- Integrated manufacturing technologies
- Quality improvement in short-run production
- Automated inspection systems and cutting technologies

2.1.3 Communication, Computing and Electronic Technologies

The Virtuoso 3D Audio System developed by the Audio and Acoustics Team in this department is one of the department's great many successes. This system effectively creates a virtual room over headphones or normal speakers. I worked for the Communications and Sensors Team in this department and hence had the privilege of watching a demonstration of this system by its developers who believe that home theatre systems will become extinct in the near future due to similar systems.

Developing a new generation of optical devices for use in communication, information storage and optical processing systems using light, designing a portable electronic scale for weighing farm stock like cattle and deer, studying MIMO systems to increase data rates in wireless communication and developing an assisted reverberation system which improves sound quality in performance venues have been some of the other projects of interest in this department [1,2]. Main fields of ongoing projects include:

- Sonic sensing in wood
- Sensor signal conditioning and processing
- Sound system theory, audio processing and surround sound systems
- Capacity increasing from applying antenna array systems
- Microprocessor hardware and software, system design and integration
- Digital and optical signal processing
- Microwave communication and antenna design
- Database development

2.1.4 Sustainable and Distributed Energy Technologies

Future-focused research on distributed generation systems, sustainable energy and hydrogen-based technologies is the main focus of this section of IRL. Prototyping hydrogen fuel cell systems which convert surplus electricity into hydrogen to generate electricity when needed and studying community level integrated distributed energy systems (Figure 01) to develop viable combinations of renewable small-scale energy systems are two of the current projects of interest [1,2]. Other fields of interest of this department are:

- Hydrogen energy storage
- CO₂ sequestration

- Energy efficiency, energy management and renewable energy technologies
- Demand side management
- Power electronics, electrical engineering research and testing



Figure 01: A community level integrated distributed energy system funded by IRL [1]

2.1.5 Biochemical Technologies

This section of IRL focuses mainly on creating innovative new products, by-product usage and process efficiencies. Developing a dietary supplement made from a concentrated extract of shark muscle lipid that is capable of preventing blood vessel formation and developing Immucillin H, which has the ability to block T-cell cancer are two of the key successes this department has had recently [1]. The main research areas are:

- Biopharmaceuticals
- Carbohydrate, polyphenolic and plant pigment chemistry
- Chemical synthesis
- Molecular structure analysis
- Plant response to ultra violet radiation
- Pharmaceuticals and agrochemicals

2.1.6 Complex Measurements and Analysis

Modelling, designing and developing instruments with more accurate measurements with closer tolerances, in shorter timeframes is the main focus of this department. The Measurements Standards Laboratory in this department is New Zealand's national metrology organisation responsible for developing accurate measurements for new technologies, improving critical measurements in processes and providing testing and calibration services [1,2]. Key fields of research interests are:

- Measurement standards – includes electrical, length, mass and pressure, time and frequency and temperature
- Radiometry and photometry
- Calibration
- Advice, consultancy and training
- Complex measurements – such as difficult process measurements, metering and process validation
- Applied mathematics – in areas such as data analysis, industrial modelling using ANSYS, fluid and particulate material flow, optimisation and simulation, geothermal energy, visualisation and statistics

2.2

Company Mission, Business Approach and Values

In general IRL's focus is to maintain their position as a leading innovation-focused business utilising world-class science to create globally competitive, market viable technologies while adding value to New Zealand industry [2]. The company tries to create value by commercialising technology and working with key business partners to establish high value industries. Their vision is simply captured in the company motto, 'brilliant ideas at work'.

The business approach focuses on building long-term partnerships, adding value to industry, developing open, trust-based relationships with clients and partners and working closely with clients to develop a full understanding of their needs and opportunities. The company offers a variety of ways of doing business with them including contract research and development, licensing, partnerships, joint ventures, small scale manufacture, pilot scale production and consultancy services [1]. For more details of IRL's business approach please refer to Appendix I.

IRL has a strong commitment towards the environment and the society. They also strive to have good relationships with their staff, customers and suppliers. The professional code of ethics at IRL involve acting with integrity; sharing, collaborating and networking; thinking creatively and innovatively; operating with excellence; achieving results; professionalism, integrity and competence; behaviour towards colleagues; behaviour towards clients and funders and interactions with the community [10]. A full copy of IRL's Professional Code of Ethics can be found in Appendix II.

2.3 Production Facilities and Structure

IRL has its main research centre in Lower Hutt, Wellington, which is where I worked during the university summer break. There were approximately 400 employees, with around 50 percent with PhDs, in mid February but this number is expected to increase to about 450 by the end of 2004 as the Company is planning to invest on some new research projects [2].

Although the Gracefield Research Centre has four companies onsite, the Institute of Geological and Nuclear Sciences, Eyecom (NZ) Ltd, IRL and Robinson Seismic Ltd, with a total of 23 buildings, IRL occupies the largest portion of it. The main buildings belonging to IRL, the Robertson building, Blocks A,B,C,D and G and the separate large building near the main entrance are shown in the building layout in Appendix III. However the floor plans of any of these buildings cannot be given due to security reasons.

Given below is a brief description of the main buildings of IRL:

Robertson building comprises of the Complex Measurements and Analysis Department and the Communication, Computing and Electronic Technologies Department. Blocks A,B,D and G along with the Processing Block are shared by the other four departments along with some common laboratories and workshops. The Finance Division, HR Group, Glassblower Lab, Mailroom and the Health Unit are in Block C while the Main Reception, Library, Computing Services and Web, Business Development and the Information Services Departments are located in the building near the main entrance.

2.4

Management structure

Basically IRL has a fairly flat structure where any person would only have one superior (or two at most) to report to. Senior management has very well defined roles and clearly knows whom they have to manage. Almost all the employees belong to a team within a department and report to their individual Team Leaders about day-to-day activities and other project related matters but may report to a senior manager for any other issues if necessary. A complete staff organisation of the company cannot be included due to confidential reasons hence is not included in the report.

2.5 Employee Relations, Work Conditions and Company Culture

IRL has quite a relaxed environment to work in. All employees are expected to work eight hours a day including a thirty-minute lunch break and two paid fifteen-minute coffee breaks which effectively gives the wage earners seven and half hours of pay per day. However these hours can vary depending on the nature of the project, time constraints and other issues. For example the summer scholarship students in the Communications and Sensors Team were approved eight paid hours of work per day by the Team Leader.

According to company tradition morning tea is at 10.30am, lunch at 12 noon and afternoon tea at 3pm but the employees may take these breaks at anytime they wish. The staff is provided with a very spacious, clean cafe with microwaves, refrigerators and clean cutlery and crockery where lunch is served between 11.30am and 2.30pm at subsidised rates. The lunch menu consists of main lunch, light lunch and vegetarian lunch hence providing a wider variety of food. This common room is always well stocked with tea, coffee, sugar, drinking cups, boiling water, milk and also has two free coffee machines and a vending machine.

The company offers its employees a wide range of opportunities. Most employees have the opportunity to enjoy benefits such as world-wide travel through international projects, support through scholarships, overseas study and work experience, internal training opportunities and attend conferences, seminars and other training courses. It also offers forty five summer scholarships to deserving tertiary students each year which help the students get their hands on real industry work [1]. Most employees in IRL are on salary basis but the students were paid on an hourly basis depending on the number of years they have studied. For a fourth year engineering student (i.e with three years of course completion), the standard rate was \$14.83 this year.

There are very strong company cultures and subcultures. For example, whenever an employee is leaving the job, gets promoted or else is completing a number of years (generally at least 30 years) at IRL, a collection is made and a gift is given to that person as a token of appreciation. There are sport events arranged between departments as well, which include sports activities such as rugby, soccer and fishing. All in all the company has a very friendly environment where people help each other, which has been one of the key issues to IRL's steady new innovations.

3.0 Work I was Involved in

The main objective of my involvement with IRL was to develop various MIMO radio channel simulator modules that take different scattering environments, channel models and antenna placements into account and produce appropriate outputs. This was considered the first step of an ongoing project, hence the deliverables were the individual channel modules, one *main* module which would integrate these modules together thus showing how they could be used later-on by the team, and documentation.

This section contains a detailed description of the work I was involved in, including channel models implemented and some of the results obtained. It also outlines the current drawbacks of the simulator modules and suggested future developments.

3.1 Introduction to MIMO Systems and Nature and Background of the Project

IRL has been doing extensive research on wireless communication for a number of years and MIMO systems has been one of the key areas of interest lately. These systems help increase capacity via the potential decorrelation in the MIMO radio channel, which can be exploited to create many parallel subchannels. However, the potential capacity gain is highly dependent on the multipath richness in the radio channel since a fully correlated MIMO channel only offers one subchannel, while a completely decorrelated channel offers multiple subchannels depending on the antenna configuration [3]. For a narrowband system, the complex channel response between a single element at the transmitter and receiver can be represented as a single complex number, hence the full channel response of a system comprising of N_{tx} transmit elements and N_{rx} receive elements can be described by an N_{rx} -by- N_{tx} two-dimensional complex matrix [4], which will be referred to as the *H matrix* in the following sections. Also the notations N_{tx} and N_{rx} will always be used to represent the number of receive and transmit elements respectively.

Amid a lot of ongoing MIMO projects, it was Mr. Neil Scott's idea to recruit three students under the IRL Summer Research Scholarship Scheme and get them to work on one of his research projects. The selected team consisted of two other students from University of Auckland and myself. It was briefed out that the project was to consist of three main parts, namely getting measurements from antennas using a multi-port vector network analyser; analysing the data and producing appropriate outputs; and developing modules for a MIMO radio channel simulator by taking various scattering models and antenna placements into account. My task was to develop the simulator modules while the other two were to work on the first two tasks.

3.2 Development of the MIMO Simulator

A lot of research had to be done in the first few weeks since the concept of MIMO and other technical details involved in the project were totally unfamiliar to me. I was given many related text books, conference papers and journals by the supervisor and was asked to note down any terms that needed more explanation. After reading many papers and some chapters of the given text books along with a substantial amount of internet research, I was able to get a better understanding of the whole MIMO concept and a feel for the actual work I had to do along with what keywords to look for while researching.

After approximately two weeks of research and many discussions with the supervisor, it was decided to implement a uniform-linear-array (ULA), line-of-sight (LOS) scenario based on the *Kronecker Model* [3] as the first attempt since this model is widely used in the industry. Also due to the limited knowledge I had in the field, it was considered one of the simplest ones to implement compared to other models such as purely statistical models and wideband channel models. It was also decided to use MATLAB in the implementation process since this package provides many built-in mathematical functions that are required.

Fourteen narrowband channel scenarios based on Kronecker Model and two wideband channel models, with one of them only partly implemented due to the time constraints, have successfully been implemented in the final solution. Outputs of some of the modules have even been compared with the actual measurements taken by other group members, and analysis of these values has produced quite similar results. Unfortunately the outputs of all the modules could not be analysed against real values since the required environments for the measurements (distribution of scatterers, number of clusters and so on) could not be realised within the laboratory environment we were working in.

3.3 MATLAB Command Line Interface and Introduction to the Simulator

All the simulator modules are written in MATLAB. A basic command line interface has been created which can be considered as the top level module where the user can select the desired channel model to simulate and enter the required input parameters. This module is called *main.m* and the MATLAB code for this can be found in Appendix IV. The MATLAB code for all the other modules is included in Appendix V along with some general comments on each module.

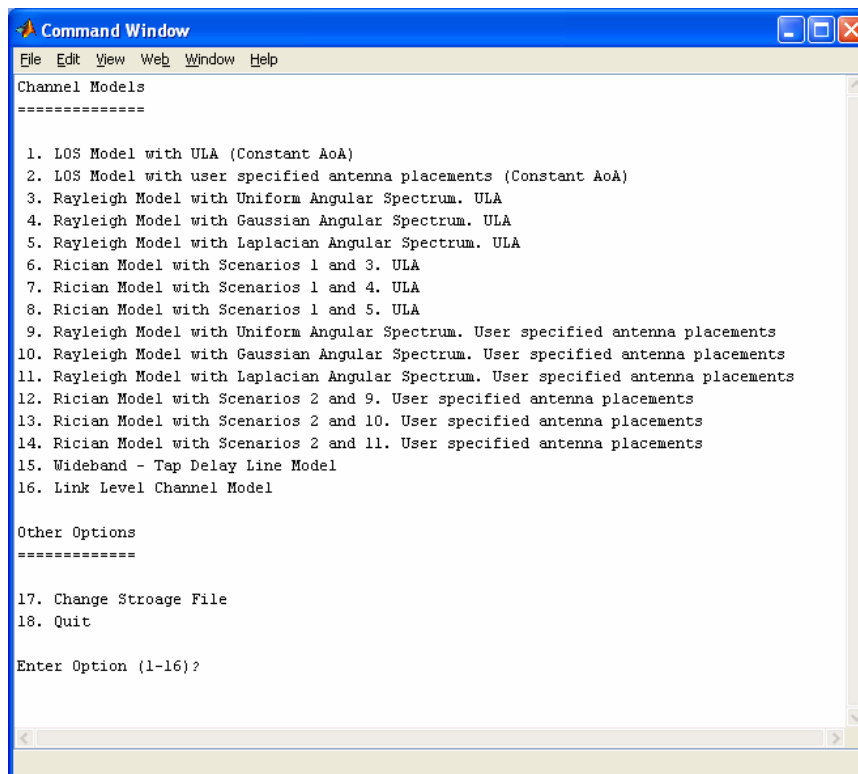


Figure 02: MATLAB command line interface

Once the scenario for a narrowband model (i.e. one of the first fourteen models in the above list) is selected and the required parameters entered, the corresponding channel module that generates the antenna correlations by taking the frequency into account will be called and the $[C]$ matrix in Figure 03 generated. Time is taken into account by creating independent and ideally distributed (i.i.d.) random channels ($[a]$ matrix in Figure 03) for the required number of time samples for a certain frequency. Repeating this process for the required number of frequency samples effectively creates a 4-dimensional matrix of size; Number of time samples required-by-frequency points-by- N_{rx} -by- N_{tx} , which is later used for analysing purposes. A simple flowchart of this process is shown below which mimics how an instance of the H matrix (of size N_{rx} -by- N_{tx}) can be created for a given frequency and time instant.

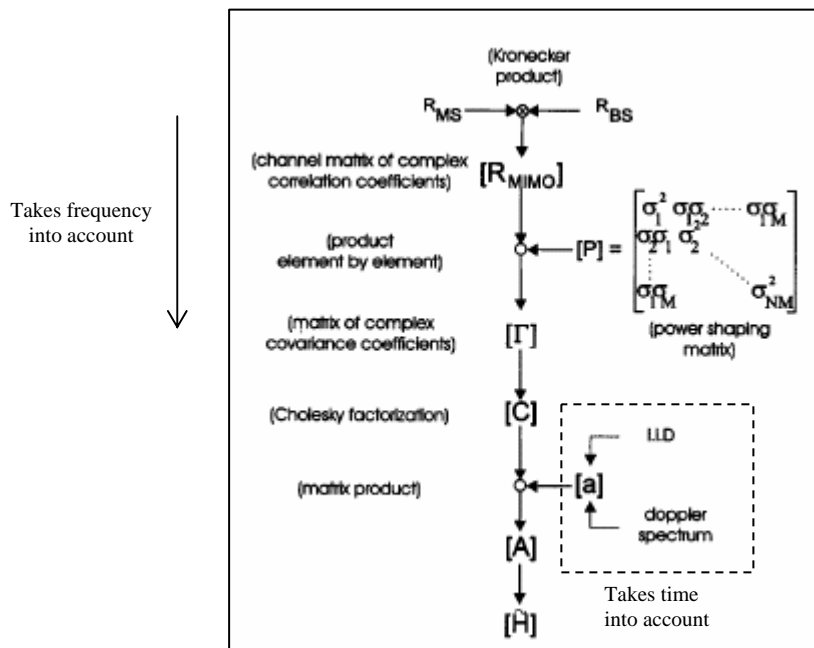


Figure 03: Flowchart showing how instances of H can be generated [modified from 3]

The $[P]$ matrix is assumed to contain all ones which will need to be modified to account for the real power shaping values in a future version of the simulator. Also the Doppler spectrum was considered irrelevant at this stage, hence has been ignored.

If one of the two wideband models (i.e. model 15 and 16 of the list) is selected, the above process needs a slight modification to account for the power delay profile (PDP) which cannot be seen in the narrow band models since the signals cannot be resolved at the receiver due to the narrow bandwidth. Upon the selection of option 15 or 16, the screen shown below in Figure 04 is brought up with the environment selections. The PDPs for these scenarios were taken from [5] but are widely available in many other literatures. The PDP values are stored in the file and the corresponding values are chosen at run time based on the environment selected. The random paths (denoted by $[a]$ in Figure 03) are then normalised by these PDP values and after accounting for the average power and delay spread by means of using finite impulse response (FIR) filters, the received signals are generated for a particular frequency. The matrix $[C]$ is generated the same way as for the narrowband models based on the distribution of scatterers. However this has been implemented only for the LOS and Rayleigh models but not the Rician models, hence only options 1-5 and 9-11 of the main menu (Figure 02) will be valid inputs.

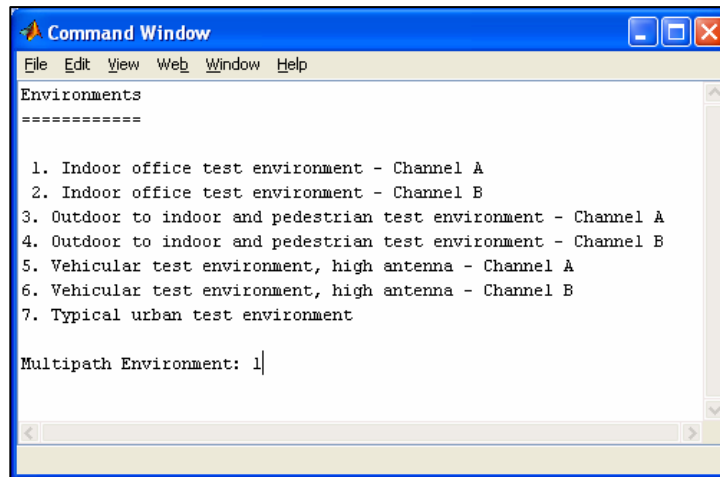


Figure 04: Different multipath scenarios for the wideband models

These generated H matrices are stored in a MATLAB .mat file with other relevant details (such as the antenna placements, N_{rx} , N_{tx} , channel model and so on) and are then used by the modules written by the other team members to analyse the data. Shown below in Figure 05 is a flowchart of the basic information flow in the simulator.

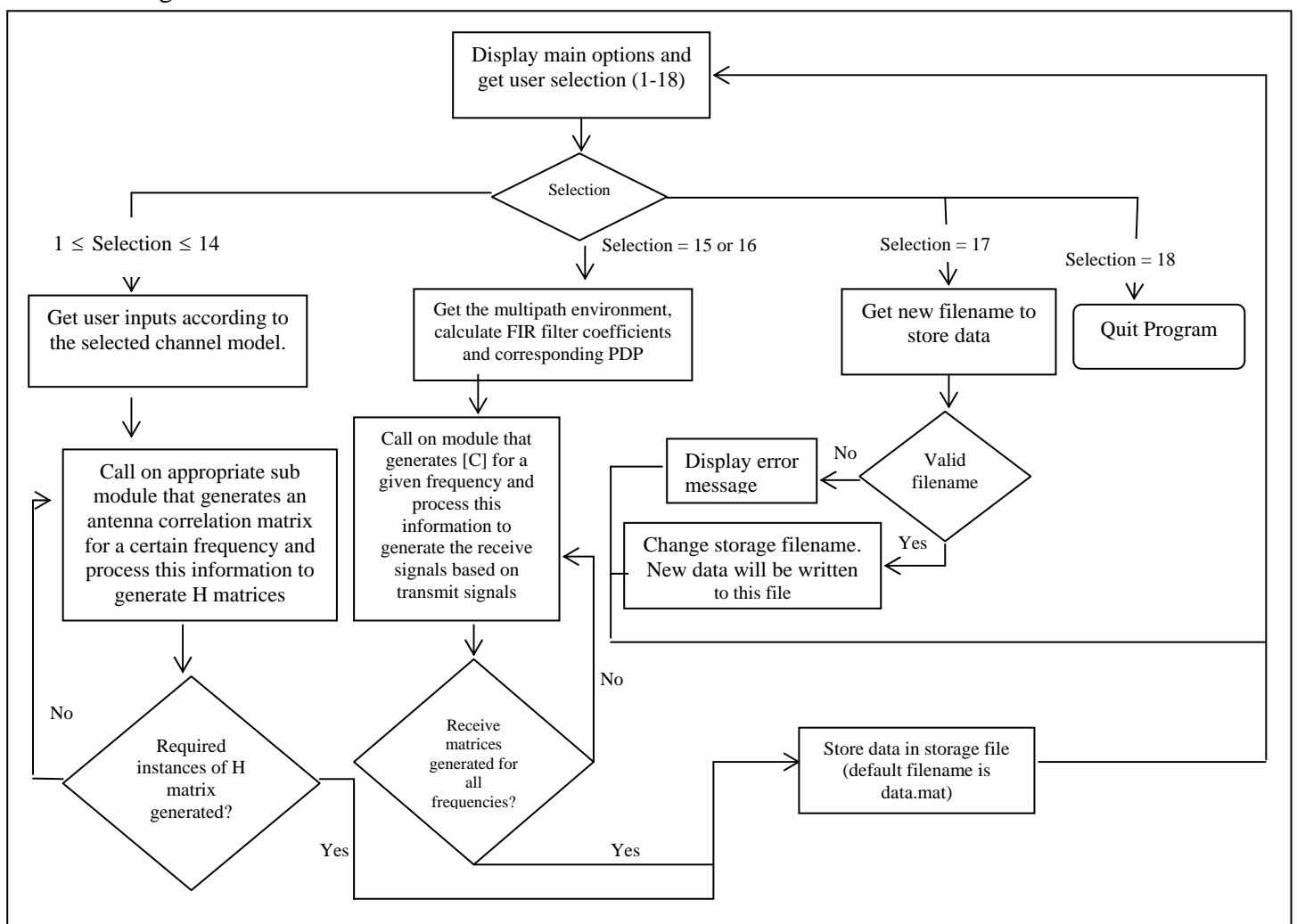


Figure 05: Flowchart showing the basic information flow in the simulator

3.4 Generation of Transmit and Receive Correlation Matrices for Different Scenarios

As shown in Figure 02, sixteen different scenarios have been implemented in the final solution. Given below is a brief description of these scenarios. Appropriate equations are included where necessary. All these routines have been implemented in MATLAB and the code can be found in Appendix V.

3.4.1 LOS model with ULA or user specified antenna placements

This is by far the simplest of all the scenarios. No scatterers are considered since the receivers are expected to receive signals from the transmitters directly. As outlined in [6], the transmit and receive correlation matrices are calculated using the equation:

$$\theta_{mk} = \begin{cases} e^{-j2\pi \frac{d_{mk} \sin \alpha}{\lambda}} & m \neq k \\ 1 & m = k \end{cases} \text{-----(1)}$$

Where θ_{mk} = Correlation between transmit/receive antenna elements m and k
 d_{mk} = Distance between elements m and k
 λ = Wavelength
 α = Angle of arrival

```

Enter Option (1-16)? 1
Transmit Antenna Spacing (in metres): .04
Receive Antenna Spacing (in metres): .04
Number of Transmit Antennas: 10
Number of Receive Antennas: 14
Angle of Arrival (in degrees): 30
Start Frequency (in GHz): 2.4
End Frequency (in GHz): 2.5
Frequency Samples Required: 10
Time Samples Required: 30

Processing Data....
Approximate time remaining (in seconds): 19.827, Frequency: 2.4
Approximate time remaining (in seconds): 16.504, Frequency: 2.4111
Approximate time remaining (in seconds): 14.231, Frequency: 2.4222
Approximate time remaining (in seconds): 12.378, Frequency: 2.4333
Approximate time remaining (in seconds): 10.315, Frequency: 2.4444
Approximate time remaining (in seconds): 8.172, Frequency: 2.4556
Approximate time remaining (in seconds): 6.099, Frequency: 2.4667
Approximate time remaining (in seconds): 4.106, Frequency: 2.4778
Approximate time remaining (in seconds): 2.042, Frequency: 2.4889
Approximate time remaining (in seconds): 0, Frequency: 2.5

sizeOf_H =

    30    10    14    10

Data successfully saved to data.mat
Simulate another channel (Y/N)?
    
```

Figure 06: Required user inputs for LOS, ULA scenario

Shown above in Figure 06 is the interface used to get the required inputs for LOS, ULA scenario after option 1 has been entered at the Figure 02 prompt. For this case of LOS, element spacing can be directly calculated since the distance between two elements is known. For the second case of LOS (where the antennas are at arbitrary positions) the individual distances are calculated using the formula

$$\mathbf{r} = \mathbf{x} - \mathbf{y} \quad \text{-----(2)}$$

where \mathbf{x} and \mathbf{y} antenna placement vectors are known in the form of $a+jb$ with respect to some coordinate system.

As explained in Section 3.3, the instances of the H_matrix are generated for required number of time samples-by-required number of frequency samples. [C] in Figure 03 is calculated once for each frequency, hence this process has been implemented using two *for* loops with the outer loop representing frequency and the inner loop representing time, in order to minimise execution time. The *Approximate time remaining* is calculated by taking the time for the most recent iteration of the outer loop and then multiplying this value by the remaining iterations of the same loop.

3.4.2 Rayleigh Models with Uniform, Gaussian or Laplacian Distributed Angular Spectra

As explained in more detail in [6], the spatial correlation coefficients for a Rayleigh fading MIMO channel can be modelled by the equation:

$$\theta_{mk} = \begin{cases} \int_{-\alpha_r/2}^{\alpha_r/2} e^{-j 2 \pi \frac{d_{mk} \sin \alpha}{\lambda}} p(\alpha) & m \neq k \\ 1 & m = k \end{cases} \quad \text{-----(3)}$$

where $p(\alpha)$ is the probability distribution of the direction of arrival or the angular spectrum (AS), and α_r is the transmit/receive antenna angular spread. The other variables have the same meaning as in equation (1).

However, the solutions to this equation depend on the distribution of the AS which can be considered mainly three fold, namely uniformly distributed AS, i.i.d. Gaussian AS and Laplacian distributed AS. The solutions to equation (3) with these three distributions can be found in [7].

Figure 07 below shows some sample values entered for a uniformly distributed AS, Rayleigh ULA scenario. All required user inputs are the same for all these modules except for the following:

- For the uniformly distributed AS case, the angular spread is required but is not for the other two
- For i.i.d. Gaussian and Laplacian, the deviation from the mean angle of arrival (or Delta Phi) and standard deviation are required but not for the uniform scenario.

```

Enter Option (1-16)? 3
Transmit Antenna Spacing (in metres): .04
Receive Antenna Spacing (in metres): .08
Number of Transmit Antennas: 12
Number of Receive Antennas: 20
Number of Clusters: 2
Transmit Phi-Zero matrix(in degrees as a 1 x 2 row matrix): [30 45]
Receive Phi-Zero matrix(in degrees as a 1 x 2 row matrix): [20 60]
Transmit angular spread matrix(in degrees as a 1 x 2 row matrix): [30 40]
Receive angular spread matrix(in degrees as a 1 x 2 row matrix): [40 40]
Cluster Linear Power at Tx (as a 1 x 2 row matrix): [1 1.2]
Cluster Linear Power at Rx (as a 1 x 2 row matrix): [1 0.8]
Terms to add up: 15
Start Frequency (in GHz): 2.4
End Frequency (in GHz): 2.5
Frequency Samples Required: 8
Time Samples Required: 15

Processing Data....
Approximate time remaining (in seconds): 77.602, Frequency: 2.4
Approximate time remaining (in seconds): 51.132, Frequency: 2.4143
Approximate time remaining (in seconds): 42.46, Frequency: 2.4286
Approximate time remaining (in seconds): 33.972, Frequency: 2.4429
Approximate time remaining (in seconds): 25.596, Frequency: 2.4571
Approximate time remaining (in seconds): 16.384, Frequency: 2.4714
Approximate time remaining (in seconds): 7.911, Frequency: 2.4857
Approximate time remaining (in seconds): 0, Frequency: 2.5

sizeof_H =

    15     8    20    12

Data successfully saved to data.mat
Simulate another channel (Y/N)? n
    
```

Figure 07: Required user inputs for a Rayleigh, ULA, uniform distribution scenario with two clusters

Similar to the LOS case, with the user specified antenna placement scenarios, the antenna placements will need to be entered and the relevant distances will be calculated according to formula (2).

3.4.3 Rician Models with Uniform, Gaussian and Laplacian Distributed Angular Spectrums

The Rician model is a combination of LOS and Rayleigh models and the Rician channel, H_{Rician} is given by [6]:

$$H_{\text{Rician}} = DH_{\text{LOS}} + \sqrt{2}\sigma_r H_{\text{Ray}} \text{ -----(4)}$$

Where $H_{\text{LOS}} = H$ matrix for LOS

$H_{\text{Ray}} = H$ matrix generated by the Rayleigh model

And the Rician-K factor, K is given by

$$K = \frac{D^2}{2\sigma_r^2} \text{-----}(5)$$

Where the powers are normalised by

$$D^2 + 2\sigma_r^2 = 1 \text{-----}(6)$$

Since this is a combination of LOS and Rayleigh models, user inputs required by both these models will be required for the Rician model.

3.4.4 Wideband Channel Models

As briefed out in Section 3.3, these channel models are more complicated to implement since they take the PDPs into account. In simple terms this means that the MIMO channel cannot be modelled as a memory-less channel anymore. The basic equation used [8] in these models is:

$$H(\tau) = \sum_{l=1}^L A_l \delta(\tau - \tau_l) \text{-----}(7)$$

Where

$$A_l = \sqrt{PC} a_l \text{-----}(8)$$

Following the notation in Figure 03 the [C] matrix is calculated based on the required scenario for a certain frequency. The received signal is then calculated using the *filter* function in MATLAB to implement equation (7). As explained in [8], the [P] in equation (8) is calculated using the average power and time delays from [9] according to the number of filter taps required.

3.5 Known Bugs and Drawbacks

Although all the modules have been tested with various values, there still could be unknown bugs apart from the ones mentioned below. The known reasons for errors generated are:

- For the Cholesky's factorisation, the *cholesky* module written by Mr. Neil Scott (included in Appendix V) was used since MATLAB's *chol* routine has been generating errors for some of the values of R_{MIMO} . The complaint was that the matrix was not positive-definite, which is the case when numbers greater than one are generated as correlation coefficients, but the *chol* routine has failed even for number less than one. The exact reason and interpretation for this is not known.
- Error checking has been done in most of the places, but the program will still generate errors if invalid inputs are entered. For example, after specifying a two cluster scattering environment, if the user enters an AS with just one value, an appropriate error message will be displayed and the *1 x no of clusters AS* will be prompted for, but if an invalid input such as *34sd* or a variable name in the base workspace is entered MATLAB will generate error messages and terminate the program.
- The last model or the Link Level Model has only been partially implemented. The details given out in [5] were not very comprehensive hence there were problems with my understanding of the model. It was agreed that the available time was better spent on other modules.

3.6

Suggested Future Developments

- It would be more attractive and more user-friendly if a graphical user interface (GUI) can be designed where the user can select channel models and enter the inputs with greater ease.
- Implementing more channel models, such as statistical based narrow or wide band channel models will provide the user with more options to model the MIMO channel.
- Providing a feel for the actual antenna placements in the *user specified antenna placement* scenarios would be helpful. For example an approximate distance between the antennas can be prompted for and based on this, a co-ordinate system can be generated on the GUI where the user can click on the desired positions where the antennas need to be placed during the modelling process.
- At the moment all the scenarios assume that the transmitter array is parallel to the receiver array in general, hence ignoring possible orientations between the two sides. Having a mechanism to account for this drawback would make the simulator more realistic.

4.0 Conclusions

The first stage of implementing a MIMO simulator has successfully been completed. Sixteen different channel models have been implemented for various antenna placements and scattering environments. These modules, implemented using MATLAB can now be used as sub-modules in later versions of the simulator. Also the involved personnel are more aware of the problems involved with the development of various channel simulator models, hence will be more familiar with tackling problems that arise in the process of taking the project further.

Working for IRL during the University summer break has been a great privilege and I would like to thank Mr. Neil Scott and all other IRL staff for all the support given to me during the course of work at IRL.

I wish the company all the best in all its future endeavours.

References

1. 2003 Company Profile – Industrial Research Limited.
2. *Communications and Sensors Team*. Retrieved 10 February, 2004 from http://technology/comms_sensors.html
3. Jean Philippe Kermaol, Laurent Schumacher, Klaus I. Pedersen and Preben E. Mogensen, "[A Stochastic MIMO Radio Channel with Experimental Validation](#)", *IEEE Journal on Selected Areas in Communications*, vol. 20, n. 6, August 2002, pp. 1211-1226
4. Beach, M.A., McNamara, D.P., Fletcher, P.N., "MIMO—A Solution for Advanced Wireless Access", *11th International Conference Antennas and Propagation*, April 2001, Conf. Pub No. 480, pg. 231-235.
5. TR 101 112 V3.2.0 (1998-04) Technical Report: Universal Mobile Telecommunications System (UMTS); Selection procedures for the choice of radio transmission technologies of the UMTS (UMTS 30.03 version 3.2.0)
6. Branka Vucetic and Jinhong Yuan, "Effect of System Parameters and Antenna Correlation on the Capacity of MIMO Channels", *Performance Limits of Multiple-Input Multiple-Output Wireless Communication System*, John Wiley & Sons Ltd, 2003. pp. 25 – 36
7. Laurent Schumacher, Klaus I. Pedersen and Preben E. Mogensen, "[From Antenna Spacings to Theoretical Capacities - Guidelines for Simulating MIMO Systems](#)", Proceedings of 13th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications PIMRC 2002, Lisbon, Portugal, 15-18 September 2002, Session TAO1.3.
8. K.I. Pedersen, J.B. Andersen, J.P. Kermaol, and P.E. Mogensen, "A Stochastic Multiple-Input-Multiple-Output Radio Channel Model for Evaluation of Space-Time Coding Algorithms", *Proceedings of VTC 2000 Fall*, pp. 893-897, Boston, United States, September 2000.
9. IEEE C802.20-03/92, "Draft 802.20 Permanent Document, Channel Models for IEEE 802.20 MBWA System, Simulations – Rev 03", 2003-11-06. pp 13-14.
10. *Principles*. Retrieved 10 February, 2004 from <http://www.irl.cri.nz/employment/principles.html>

Appendices

Appendix 1: Mission Statement	II
Appendix 1I: Professional Code of Ethics	III
Appendix 1II: Building Layout at Gracefield Research Centre	IV
Appendix IV: MATLAB Code Written for main.m.....	V-XI
Appendix V: MATLAB Code Written to Implement Various MIMO Channel Models	XII-XXIV

Appendix 1: Mission Statement

The following is an extract from [1]:

We strive to -

- build long-term partnerships
- add value to NZ and industry
- develop open, trust-based relationships with clients and partners
- work closely with clients to develop a full understanding of their needs and opportunities

We capture value from our technology platforms through -

Contract R&D - We have a multi-disciplinary pool of staff that undertakes world-class contract research and development for clients.

Licensing - Our technologies have exciting commercial potential and represent a significant technological advance in their field. We have licensed various technologies to New Zealand and offshore companies. We also have a portfolio of technologies which offer investors opportunities to participate in and benefit from further development and commercialisation.

Partnerships - We have entered into strategic business alliances with selected companies and are their preferred research and development supplier. We also have valuable partnerships with some of the world's leading research institutes.

Joint ventures - We have formed numerous joint ventures, usually to take Industrial Research's technology or intellectual property through commercialisation stages to full manufacturing, marketing and sales positions. Our current portfolio includes nine internationally focused joint ventures and subsidiaries.

The creation of new businesses - We back our own technologies, particularly high value technologies, with capital investment leading to full commercially operated businesses.

Small scale manufacture - Industrial Research's fine chemical production capabilities are used by researchers and manufacturers internationally. With expertise in the production to order of enzyme inhibitors and other bioactives, Industrial Research operates an online ordering service - www.fine-chemicals-online.com - for clients throughout the world.

Pilot scale production - New and improved products and manufacturing processes are developed for clients starting from discovery, through process development to multi-kilogram GMP production. Our pilot scale production expertise includes biopharmaceuticals, fine chemicals and glycotherapeutics.

Consultancy services - We provide a range of specialist consultancy services including feasibility studies, appraisals, technical advice, testing, verification and system reviews.

Technology services - Our technology services include:

- Materials selection
- Information technology
- Remanent life assessment
- Materials testing/analysis
- Literature searches
- System development and analysis
- Modelling
- Product testing
- Calibration of equipment
- Access to standards information

Appendix 1I: Professional Code of Ethics

The following is an extract from [10]:

At Industrial Research, we are committed to upholding the highest standards of ethics and business conduct.

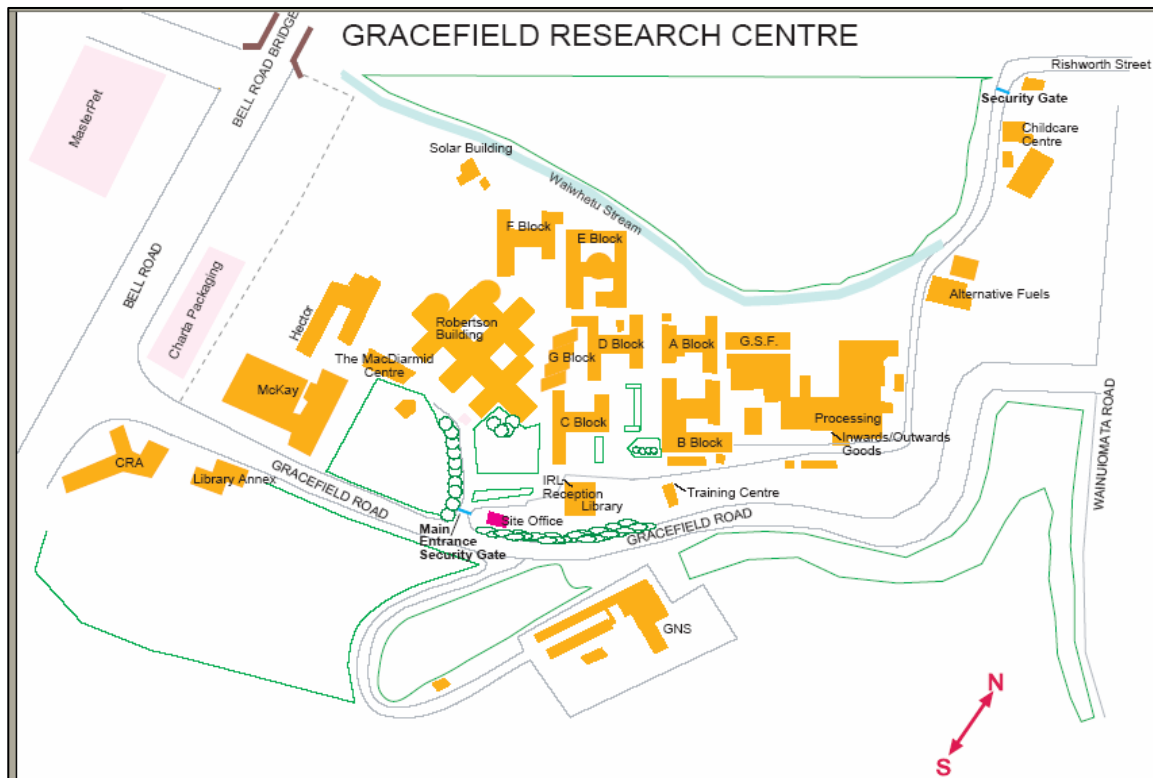
We have in place a code for professional conduct and ethics with which all employees must comply. The code covers:

- Professionalism, integrity and competence
- Behaviour towards colleagues
- Behaviour towards clients and funders
- Interactions with the community

It encompasses our relationship with our shareholders (ie the government), with each other as employees, our customers, our suppliers, the communities in which we operate, and our competitors.

The code covers principles such as fair competition, honest transactions, equal opportunities for all employees, and being a responsible corporate citizen of the communities in which the company operates.

Appendix III: Building Layout at Gracefield Research Centre



Appendix IV: MATLAB Code Written for *main.m*

Code:

```

%function main
%This is the main module used to integrate all the different channel models
%implemented. Top level routine.
%Written by Thusitha De Silva on 10.02.2004
%Variables:
%Dtx,Drx - transmitter/receiver element spacing
%Ntx,Nrx - Number of transmitters/receivers
%AoA - Angle of Arrival
%timeSmpls - required number of time samples. Used to model randomness of channel
%stFreq - start frequency (generally 2.4Ghz)
%endFreq - end frequency (usually 2.5)
%freqSmpls - frequency samples required
%frequencies - Array of frequencies between start & end freq (with reqd no
%of freq samples)
%Tx_matrix,Rx_matrix - antenna placement of transmitters/receivers (in the form a+jb)
%noOfClusters - number of clusters in a multicluster environment
%phiZeroTx,phiZeroRx - mean AoA (sort of) for transmitter/receiver
%ASTx,ASRx - angular spread for transmitter/receiver
%delPhiTx,delPhiRx - deviation between mean and one end of AS for transmitter/receiver
%sigmaTx,sigmaRx - standard deviation for transmitter/receiver
%clstr_LnrPwrTx,clstr_LnrPwrRx - cluster linear power
%terms - number of terms to add up in the infinite summation solutions
%QTx,QRx - scaling factor for the cluster
%All of above should be entered in the form [1 x noOfClusters]

function main
clc;
fileName = 'data.mat'; %This is the default storage file. Can be changed later if reqd
%Main loop
while(1)
    clc;
    %Different scenarios implemented
    disp(sprintf('Channel Models\n===== \n'));
    disp(' 1. LOS Model with ULA (Constant AoA)');
    disp(' 2. LOS Model with user specified antenna placements (Constant AoA)');
    disp(' 3. Rayleigh Model with Uniform Angular Spectrum. ULA');
    disp(' 4. Rayleigh Model with Gaussian Angular Spectrum. ULA');
    disp(' 5. Rayleigh Model with Laplacian Angular Spectrum. ULA');
    disp(' 6. Rician Model with Scenarios 1 and 3. ULA');
    disp(' 7. Rician Model with Scenarios 1 and 4. ULA');
    disp(' 8. Rician Model with Scenarios 1 and 5. ULA');
    disp(' 9. Rayleigh Model with Uniform Angular Spectrum. User specified antenna placements');
    disp('10. Rayleigh Model with Gaussian Angular Spectrum. User specified antenna placements');
    disp('11. Rayleigh Model with Laplacian Angular Spectrum. User specified antenna placements');
    disp('12. Rician Model with Scenarios 2 and 9. User specified antenna placements');
    disp('13. Rician Model with Scenarios 2 and 10. User specified antenna placements');
    disp('14. Rician Model with Scenarios 2 and 11. User specified antenna placements');
    disp('15. Wideband - Tap Delay Line Model');
    disp('16. Link Level Channel Model');
    disp(sprintf('\nOther Options\n===== \n\n17. Change Storage File'));
    disp(sprintf('18. Quit\n'));
    model = getInput('Enter Option (1-16)? ',1,1);

    %LOS Model with ULA (Constant AoA)
    if model == 1
        [Dtx,Drx,Ntx,Nrx] = getCommon4ULA;
        AoA = getInput('Angle of Arrival (in degrees): ',1,1);
        [frequencies,timeSmpls,stFreq,endFreq,freqSmpls] = getFrequencies;
        disp(sprintf('\nProcessing Data...'));
        processLOSdata(model,Dtx,Drx,AoA,Ntx,Nrx,0,0,frequencies,timeSmpls,stFreq,endFreq,fileName,0);

    %LOS Model with user specified antenna placements (Constant AoA)
    elseif model == 2
        [Tx_matrix,Rx_matrix,Ntx,Nrx] = getCommon4NULA;
        AoA = getInput('Angle of Arrival (in degrees): ',1,1);
        [frequencies,timeSmpls,stFreq,endFreq,freqSmpls] = getFrequencies;
        disp(sprintf('\nProcessing Data...'));
        processLOSdata(model,0,0,AoA,Ntx,Nrx,Tx_matrix,Rx_matrix,frequencies,timeSmpls,stFreq,endFreq,fileName,0);
    end
end
    
```

```

%All the Rayleigh models
%Rayleigh Model with Uniform/gaussian/Laplacian Angular Spectrum.
%ULA/arbitrary placement
elseif model == 3 | model == 4 | model == 5 | model == 9 | model == 10 | model == 11
    if model==3|model==4|model==5 %All the ULA cases
        [Dtx,Drx,Ntx,Nrx] = getCommon4ULA;
        Tx_matrix=0;Rx_matrix=0;
    else %All the NULA cases
        [Tx_matrix,Rx_matrix,Ntx,Nrx] = getCommon4NULA;
        Dtx=0;Drx=0;
    end
    [noOfClusters,phiZeroTx,phiZeroRx,ASTx,ASRx,delPhiTx,delPhiRx,sigmaTx,sigmaRx,clstr_LnrPwrTx,...
    clstr_LnrPwrRx,terms,QTx,QRx] = getAllRayleighParameters(model);
    [frequencies,timeSmpls,stFreq,endFreq,freqSmpls] = getFrequencies;
    disp(sprintf('\nProcessing Data...'));
    processRayleighData(model,Dtx,Drx,Ntx,Nrx,noOfClusters,phiZeroTx,phiZeroRx,ASTx,ASRx,delPhiTx,delPhiRx,sigmaTx,sigmaRx,...
    clstr_LnrPwrTx,clstr_LnrPwrRx,terms,QTx,QRx,Tx_matrix,Rx_matrix,frequencies,timeSmpls,stFreq,endFreq,fileName,0);

%All the Rician models
elseif model == 6 | model == 7 | model == 8 | model == 12 | model == 13 | model == 14
    if model==6|model==7|model==8 %All the ULA cases
        [Dtx,Drx,Ntx,Nrx] = getCommon4ULA;
        Tx_matrix=0;Rx_matrix=0;
    else %All the NULA cases
        [Tx_matrix,Rx_matrix,Ntx,Nrx] = getCommon4NULA;
        Dtx=0;Drx=0;
    end
    AoA = getInput('Angle of Arrival (in degrees): ',1,1);
    K = getInput('Rician K-factor: ',1,1);

    [noOfClusters,phiZeroTx,phiZeroRx,ASTx,ASRx,delPhiTx,delPhiRx,sigmaTx,sigmaRx,clstr_LnrPwrTx,...
    clstr_LnrPwrRx,terms,QTx,QRx] = getAllRayleighParameters(model);
    [frequencies,timeSmpls,stFreq,endFreq,freqSmpls] = getFrequencies;
    disp(sprintf('\nProcessing Data...'));
    processRicianData(model,Dtx,Drx,Ntx,Nrx,AoA,noOfClusters,phiZeroTx,phiZeroRx,ASTx,ASRx,delPhiTx,delPhiRx,sigmaTx,sigmaRx,...
    clstr_LnrPwrTx,clstr_LnrPwrRx,terms,QTx,QRx,Tx_matrix,Rx_matrix,frequencies,timeSmpls,stFreq,endFreq,fileName,K);

%Wideband - Tap Delay Line Model
elseif model == 15
    disp(sprintf('\nMultipath Environments\n=====\n'));
    disp('1. Indoor office test environment - Channel A');
    disp('2. Indoor office test environment - Channel B');
    disp('3. Outdoor to indoor and pedestrian test environment - Channel A');
    disp('4. Outdoor to indoor and pedestrian test environment - Channel B');
    disp('5. Vehicular test environment, high antenna - Channel A');
    disp('6. Vehicular test environment, high antenna - Channel B');
    disp('7. Typical urban test environment');

    chnlModel = getInput('\nChannel Model or Distribution of Scatterers (1-11 except 6-8): ',1,1);
    env = getInput('Multipath Environment: ',1,1);
    %LOS
    if chnlModel == 1
        [Dtx,Drx,Ntx,Nrx] = getCommon4ULA;
        AoA = getInput('Angle of Arrival (in degrees): ',1,1);
        Tx_matrix=0;Rx_matrix=0;
    elseif chnlModel == 2
        [Tx_matrix,Rx_matrix,Ntx,Nrx] = getCommon4NULA;
        AoA = getInput('Angle of Arrival (in degrees): ',1,1);
        Dtx=0;Drx=0;
    %Rayleigh - ULA
    elseif chnlModel==3 | chnlModel==4 | chnlModel==5
        [Dtx,Drx,Ntx,Nrx] = getCommon4ULA;
        [noOfClusters,phiZeroTx,phiZeroRx,ASTx,ASRx,delPhiTx,delPhiRx,sigmaTx,sigmaRx,clstr_LnrPwrTx,...
        clstr_LnrPwrRx,terms,QTx,QRx] = getAllRayleighParameters(chnlModel);
        Tx_matrix=0;Rx_matrix=0;
    %Rayleigh - NULA
    elseif chnlModel==9 | chnlModel==10 | chnlModel==11
        [Tx_matrix,Rx_matrix,Ntx,Nrx] = getCommon4NULA;
        [noOfClusters,phiZeroTx,phiZeroRx,ASTx,ASRx,delPhiTx,delPhiRx,sigmaTx,sigmaRx,clstr_LnrPwrTx,...
        clstr_LnrPwrRx,terms,QTx,QRx] = getAllRayleighParameters(chnlModel);
        Dtx=0;Drx=0;
    end

```

```

[frequencies,timeSmpls,stFreq,endFreq,freqSmpls] = getFrequencies; %we don't care about timeSmpls returned by this function for this
timeSmpls = getInput('Length of Transmission Array: ',1,1);
%Here Tx_matrix is the transmit signal (in the form Ntx x time
%samples)
Tx_matrix = getInput(['Transmit Signal (as ',num2str(Ntx),' x ',num2str(timeSmpls),' Matrix): '],Ntx,timeSmpls);
noOfTaps = getInput('Number of Filter Taps Required (0-11): ',1,1);
samplFreq = getInput('Sampling Frequency: ',1,1);

disp(sprintf('\nProcessing Data...'));
for f = 1:length(frequencies)
    startTime = clock;
    wavelength = 3*1e8/(frequencies(f)*1e9);
    %LOS
    if chnlModel==1 | chnlModel==2
        C = processLOSdata(chnlModel,Dtx,Drx,AoA,Ntx,Nrx,0,0,frequencies(f),timeSmpls,stFreq,endFreq,fileName,1);
    %Rayleigh
    elseif chnlModel==3 | chnlModel==4 | chnlModel==5 | chnlModel==9 | chnlModel==10 | chnlModel==11
        C = processRayleighData(chnlModel,Dtx,Drx,Ntx,Nrx,noOfClusters,phiZeroTx,phiZeroRx,ASTx,ASRx,delPhiTx,delPhiRx,...
            ,sigmaTx,sigmaRx,clstr_LnrPwrTx,clstr_LnrPwrRx,terms,QTx,QRx,Tx_matrix,Rx_matrix,...
            frequencies(f),timeSmpls,stFreq,endFreq,fileName,1);
    end
    [Rx_matrix] = TapDelayLineModel(Nrx,Tx_matrix,noOfTaps,samplFreq,env,C);
    %Recv4_DiffFreq is the huge f x Nrx x Ntx matrix with all the
    %receive signals
    Recv4_DiffFreq(f,:) = Rx_matrix;
    %Approx end time = time for 1 iteration * no of remaining iterations
    endTime=etime(clock,startTime)*(length(frequencies)-f);
    %just to have a feel for how long the simulation's gona take
    disp(['Approximate time remaining (in seconds): ',num2str(endTime),' , Frequency: ',num2str(frequencies(f))]);
end
sizeOf_Recv4_DiffFreq = size(Recv4_DiffFreq)
%Format the .mat file and save variables so that Vinod's modules can use data from this module
array = Recv4_DiffFreq; startFreq = stFreq*1e9; stopFreq = endFreq*1e9;
dimensions = 'Number of frequency points x Number of Rxs x Number of Tx';
save(fileName,'array','dimensions','startFreq','stopFreq');
disp(['Data successfully saved to ',fileName]);
if chnlModel == 1
    model2='LOS, ULA';
    save(fileName,'Dtx','Drx','AoA','model2','-append');
elseif chnlModel == 2
    model2='LOS, Arbitrary Antenna Placements';
    save(fileName,'Tx_matrix','Rx_matrix','AoA','model2','-append');
end
if chnlModel==3|chnlModel==9 distString='Uniform';elseif chnlModel==4|chnlModel==10 distString='Gaussian';...
    else distString='Laplacian';end
if chnlModel==3|chnlModel==4|chnlModel==5 st='ULA';,else st='Arbitrary Antenna Placements';,end
model2 = strcat('Tap Delay Line Model, ',st,distString,' Distribution');
save(fileName,'env','Tx_matrix','noOfTaps','samplFreq','model2','-append');
disp(['Data successfully saved to ',fileName]);
beep

%Link Level Channel Model
elseif model == 16
    disp('This model hasn't been integrated into the main module yet!');

%change the default file name
elseif model == 17
    fN = input('Enter new file name:)? ','s');
    [pathstr,name,ext,versn] = fileparts(fN);
    if strcmp(ext, '.mat')
        disp(['Storage file changed from ',fileName,' to ',pathstr,name,ext]);
        fileName = strcat(pathstr,name,ext);
    else disp('Couldn't change filename. Invalid extension');,end
    disp(sprintf('\nPress any key to continue...')); pause;

%quit the program
elseif model == 18
    break
end
if model~=17, reply = input('Simulate another channel (Y/N)? ','s');,end
if ~isempty(reply) & lower(reply)=='n' ,break,end
end

```

```

%Get all the common inputs for ULA
function [Dtx,Drx,Ntx,Nrx] = getCommon4ULA
    Dtx = getInput('Transmit Antenna Spacing (in metres): ',1,1);
    Drx = getInput('Receive Antenna Spacing (in metres): ',1,1);
    Ntx = getInput('Number of Transmit Antennas: ',1,1);
    Nrx = getInput('Number of Receive Antennas: ',1,1);

%Get all the common inputs for NULA
function [Tx_matrix,Rx_matrix,Ntx,Nrx] = getCommon4NULA
    Ntx = getInput('Number of Transmit Antennas: ',1,1);
    Nrx = getInput('Number of Receive Antennas: ',1,1);
    Rx_matrix = getInput(['Receive Antenna Placements (as a ',num2str(Nrx),' x 1 matrix in the form a+jb): '],Nrx,1);
    Tx_matrix = getInput(['Transmit Antenna Placements (as a ',num2str(Ntx),' x 1 matrix in the form a+jb): '],Ntx,1);

%Get the frequency details
function [frequencies,timeSmpls,stFreq,endFreq,freq$smpls] = getFrequencies
    stFreq = getInput('Start Frequency (in GHz): ',1,1);
    endFreq = getInput('End Frequency (in GHz): ',1,1);
    freq$smpls = getInput('Frequency Samples Required: ',1,1);
    timeSmpls = getInput('Time Samples Required: ',1,1);
    if freq$smpls==1, frequencies=stFreq;else frequencies = linspace(stFreq,endFreq,freq$smpls);end

%Make sure input is in the required form
function out=getInput(st,rows,cols)
    out=input(st);
    while(size(out,1)~=rows | size(out,2)~=cols)
        disp(['Error: Input should be a ',num2str(rows),' x ',num2str(cols)];disp(' '));
        out = input(st);
    end;

%Get all the Raileigh parameters reqd for selected model
function [noOfClusters,phiZeroTx,phiZeroRx,ASTx,ASRx,delPhiTx,delPhiRx,sigmaTx,sigmaRx,clstr_LnrPwrTx,...
    clstr_LnrPwrRx,terms,QTx,QRx] = getAllRayleighParameters(model)

    noOfClusters = getInput('Number of Clusters: ',1,1);
    phiZeroTx = getInput(['Transmit Phi-Zero matrix(in degrees as a 1 x ',num2str(noOfClusters),' row matrix): '],1,noOfClusters);
    phiZeroRx = getInput(['Receive Phi-Zero matrix(in degrees as a 1 x ',num2str(noOfClusters),' row matrix): '],1,noOfClusters);
    if model==4 | model==5 | model==7 | model==8 | model==10 | model==11 | model==13 | model==14
        delPhiTx = getInput(['Transmit Delta Phi matrix(in degrees as a 1 x ',num2str(noOfClusters),' row matrix): '],1,noOfClusters);
        delPhiRx = getInput(['Receive Delta Phi matrix(in degrees as a 1 x ',num2str(noOfClusters),' row matrix): '],1,noOfClusters);
        sigmaTx = getInput(['Transmit Standard Deviation matrix(in degrees as a 1 x ',num2str(noOfClusters),' row matrix): '],1,...
            noOfClusters);
        sigmaRx = getInput(['Receive Standard Deviation matrix(in degrees as a 1 x ',num2str(noOfClusters),' row matrix): '],1,...
            noOfClusters);
        ASTx=0;ASRx=0;
    else %reqd only for the uniform case. delPhi and Q are calculated using AS for uniform but not for Gaussian and Laplacian
        ASTx = getInput(['Transmit angular spread matrix(in degrees as a 1 x ',num2str(noOfClusters),' row matrix): '],1,noOfClusters);
        ASRx = getInput(['Receive angular spread matrix(in degrees as a 1 x ',num2str(noOfClusters),' row matrix): '],1,noOfClusters);
        delPhiTx=0;delPhiRx=0;sigmaTx=0;sigmaRx=0;
    end
    clstr_LnrPwrTx = getInput(['Cluster Linear Power at Tx (as a 1 x ',num2str(noOfClusters),' row matrix): '],1,noOfClusters);
    clstr_LnrPwrRx = getInput(['Cluster Linear Power at Rx (as a 1 x ',num2str(noOfClusters),' row matrix): '],1,noOfClusters);
    terms = getInput('Terms to add up: ',1,1);

%Call on [modified] Laurent Schumacher's code to get the corresponding Qs, delta-phis
if model == 3 | model == 6 | model == 9 | model == 12
    [delPhiTx_rad,QTx] = normalisation_uniform(noOfClusters, clstr_LnrPwrTx, ASTx);
    [delPhiRx_rad,QRx] = normalisation_uniform(noOfClusters, clstr_LnrPwrRx, ASRx);
    delPhiTx=delPhiTx_rad*180/pi;
    delPhiRx=delPhiRx_rad*180/pi;
elseif model == 4 | model == 7 | model == 10 | model == 13
    QTx = normalisation_gaussian(noOfClusters,clstr_LnrPwrTx,delPhiTx,sigmaTx);
    QRx = normalisation_gaussian(noOfClusters,clstr_LnrPwrRx,delPhiRx,sigmaRx);

elseif model == 5 | model == 8 | model == 11 | model == 14
    QTx = normalisation_laplacian(noOfClusters,clstr_LnrPwrTx,delPhiTx,sigmaTx);
    QRx = normalisation_laplacian(noOfClusters,clstr_LnrPwrRx,delPhiRx,sigmaRx);
end

%Format the .mat file so that Vinod's modules can use data from this module
function saveData(H,stFreq,endFreq,model,fileName)
    array = H; startFreq = stFreq*1e9; stopFreq = endFreq*1e9;
    dimensions = 'time samples x frequency samples x No. of Receivers x No. of Transmitters';
    save(fileName,'array','model','dimensions','startFreq','stopFreq');
    disp(['Data successfully saved to ',fileName]);
    beep

```

```

%Processes all the LOS data. Calls on the appropriate LOS module based on
%selection. optionalRtrn is used only when ndOptnlRtrn==1 which is true only
%for the tap delay model since we dont need the time loop
function optionalRtrn = processLOSdata(model,Dtx,Drx,AoA,Ntx,Nrx,Tx_matrix,Rx_matrix,frequencies,timeSmpls,...
    stFreq,endFreq,fileName,ndOptnlRtrn)

for f = 1:length(frequencies)
    startTime = clock;
    waveLength = 3*1e8/(frequencies(f)*1e9);
    if model == 1
        R_MIMO = LOS_Kron_constAlphal(Dtx,Drx,AoA,Ntx,Nrx,waveLength);
    elseif model == 2
        R_MIMO = LOS_Kron_constAlpha2_diffPos(Rx_matrix,Tx_matrix,waveLength,AoA);
    end
    tmp_powerShapingMat = ones(Ntx*Nrx,Ntx*Nrx); %Assumed to be all ones
    gamma = R_MIMO.*tmp_powerShapingMat;
    C = cholesky(gamma);
    if ndOptnlRtrn==1, optionalRtrn=C;return,else optionalRtrn=0;end
    %dont need this time loop for the tap delay model. Need only C
    for t = 1:timeSmpls
        a = (randn(Ntx*Nrx,1)+randn(Ntx*Nrx,1)*j)/sqrt(2);
        H(t,f,,:) = reshape(C*a,Nrx,Ntx);
    end
    endTime=etime(clock,startTime)*(length(frequencies)-f);
    disp(['Approximate time remaining (in seconds): ',num2str(endTime),' , Frequency: ',...
        num2str(frequencies(f))]); %just to have a feel for how long the simulation's gonna take
end
sizeOf_H = size(H)
if model == 1
    saveData(H,stFreq,endFreq,'LOS, ULA',fileName);
    save(fileName, 'Dtx', 'Drx', 'AoA', '-append');
else
    saveData(H,stFreq,endFreq,'LOS, Arbitrary Antenna Placements',fileName);
    save(fileName, 'Tx_matrix', 'Rx_matrix', 'AoA', '-append');
end

%Processes all the Rayleigh data. Calls on the appropriate Rayleigh module based on
%selection. optionalRtrn is used only when ndOptnlRtrn==1 which is true only
%for the tap delay model since we dont need the time loop
function optionalRtrn = processRayleighData(model,Dtx,Drx,Ntx,Nrx,noOfClusters,phiZeroTx,phiZeroRx,ASTx,ASRx,...
    delPhiTx,delPhiRx,sigmaTx,sigmaRx,clstr_LnrPwrTx,clstr_LnrPwrRx,terms,QTx,QRx,Tx_matrix,Rx_matrix,...
    frequencies,timeSmpls,stFreq,endFreq,fileName,ndOptnlRtrn);

for f = 1:length(frequencies)
    startTime = clock;
    waveLength = 3*1e8/(frequencies(f)*1e9);
    if model == 3 %ULA, Uniform
        R_MIMO = Ray_Kron_uniformAS3(Dtx,Drx,ASTx,ASRx,Ntx,Nrx,waveLength,...
            phiZeroTx,phiZeroRx,delPhiTx,delPhiRx,QTx,QRx,terms,noOfClusters);
    elseif model == 4 %ULA, Gaussian
        R_MIMO = Ray_Kron_gausAS4(Dtx,Drx,Ntx,Nrx,waveLength,...
            phiZeroTx,phiZeroRx,delPhiTx,delPhiRx,QTx,QRx,terms,noOfClusters,sigmaTx,sigmaRx);
    elseif model == 5 %ULA, Laplacian
        R_MIMO = Ray_Kron_lapAS5(Dtx,Drx,Ntx,Nrx,waveLength,...
            phiZeroTx,phiZeroRx,delPhiTx,delPhiRx,QTx,QRx,terms,noOfClusters,sigmaTx,sigmaRx);
    elseif model == 9 %NULA, Uniform
        R_MIMO = Ray_Kron_uniformAS9_diffPos(Tx_matrix,Rx_matrix,ASTx,ASRx,waveLength,...
            phiZeroTx,phiZeroRx,delPhiTx,delPhiRx,QTx,QRx,terms,noOfClusters);
    elseif model == 10 %NULA, Gaussian
        R_MIMO = Ray_Kron_gausAS10_diffPos(Tx_matrix,Rx_matrix,waveLength,...
            phiZeroTx,phiZeroRx,delPhiTx,delPhiRx,QTx,QRx,terms,noOfClusters,sigmaTx,sigmaRx);
    elseif model == 11 %NULA, Laplacian
        R_MIMO = Ray_Kron_lapAS11_diffPos(Tx_matrix,Rx_matrix,waveLength,...
            phiZeroTx,phiZeroRx,delPhiTx,delPhiRx,QTx,QRx,terms,noOfClusters,sigmaTx,sigmaRx);
    end
    tmp_powerShapingMat = ones(Ntx*Nrx,Ntx*Nrx); %Assumed to be all ones
    gamma = R_MIMO.*tmp_powerShapingMat;
    C = cholesky(gamma);
    if ndOptnlRtrn==1, optionalRtrn=C;return,else optionalRtrn=0;end
    for t = 1:timeSmpls
        a = (randn(Ntx*Nrx,1)+randn(Ntx*Nrx,1)*j)/sqrt(2);
        H(t,f,,:) = reshape(C*a,Nrx,Ntx);
    end
end

```

```

        endTime=etime(clock,startTime)*(length(frequencies)-f);
        disp(['Approximate time remaining (in seconds): ',num2str(endTime),' , Frequency: ',...
            num2str(frequencies(f))]); %just to have a feel for how long the simulation's gona take
    end
    sizeOf_H = size(H)
    if model==3|model==9 distString='Uniform';elseif model==4|model==10 distString='Gaussian';...
        else distString='Laplacian';end
    if model==3|model==4|model==5 st='ULA, ';else st='Arbitrary Antenna Placements, ';end
    %save data and other relevant variables
    saveData(H,stFreq,endFreq,['Rayleigh, ',st,distString,' Distribution'],fileName);
    save(fileName,'phiZeroTx','phiZeroRx','delPhiTx','delPhiRx','-append');
    if model==3
        save(fileName,'Dtx','Drx','ASTx','ASRx','-append');
    elseif model==4|model==5
        save(fileName,'Dtx','Drx','delPhiTx','delPhiRx','sigmaTx','sigmaRx','-append');
    else
        save(fileName,'Tx_matrix','Rx_matrix','-append');
    end

%Processes all the Rician data. Calls on the appropriate LOS and Rayleigh module based on
%selection. Cannot be used with models 15 or 16
function processRicianData(model,Dtx,Drx,Ntx,Nrx,AoA,noOfClusters,phiZeroTx,phiZeroRx,ASTx,ASRx,delPhiTx,...
    delPhiRx,sigmaTx,sigmaRx,clstr_LnrPwrTx,clstr_LnrPwrRx,terms,QTx,QRx,Tx_matrix,Rx_matrix,frequencies,...
    timeSmpls,stFreq,endFreq,fileName,K)

    sigmaR = sqrt(0.5/(K+1));
    D = sqrt(K/(K+1));

    for f = 1:length(frequencies)
        startTime = clock;
        wavelength = 3*1e8/(frequencies(f)*1e9);
        %get the Rayleigh data
        if model == 6
            R_MIMO = Ray_Kron_uniformAS3(Dtx,Drx,ASTx,ASRx,Ntx,Nrx,wavelength,...
                phiZeroTx,phiZeroRx,delPhiTx,delPhiRx,QTx,QRx,terms,noOfClusters);
        elseif model == 7
            R_MIMO = Ray_Kron_gausAS4(Dtx,Drx,Ntx,Nrx,wavelength,...
                phiZeroTx,phiZeroRx,delPhiTx,delPhiRx,QTx,QRx,terms,noOfClusters,sigmaTx,sigmaRx);
        elseif model == 8
            R_MIMO = Ray_Kron_lapAS5(Dtx,Drx,Ntx,Nrx,wavelength,...
                phiZeroTx,phiZeroRx,delPhiTx,delPhiRx,QTx,QRx,terms,noOfClusters,sigmaTx,sigmaRx);
        elseif model == 12
            R_MIMO = Ray_Kron_uniformAS9_diffPos(Tx_matrix,Rx_matrix,ASTx,ASRx,wavelength,...
                phiZeroTx,phiZeroRx,delPhiTx,delPhiRx,QTx,QRx,terms,noOfClusters);
        elseif model == 13
            R_MIMO = Ray_Kron_gausAS10_diffPos(Tx_matrix,Rx_matrix,wavelength,...
                phiZeroTx,phiZeroRx,delPhiTx,delPhiRx,QTx,QRx,terms,noOfClusters,sigmaTx,sigmaRx);
        elseif model == 14
            R_MIMO = Ray_Kron_lapAS11_diffPos(Tx_matrix,Rx_matrix,wavelength,...
                phiZeroTx,phiZeroRx,delPhiTx,delPhiRx,QTx,QRx,terms,noOfClusters,sigmaTx,sigmaRx);
        end
        tmp_powerShapingMat = ones(Ntx*Nrx,Ntx*Nrx);
        gamma = R_MIMO.*tmp_powerShapingMat;
        C_Ray = cholesky(gamma);
        %Get the LOS component
        if model==6|model==7|model==8
            R_MIMO = LOS_Kron_constAlpha(Dtx,Drx,AoA,Ntx,Nrx,wavelength);
        else
            R_MIMO = LOS_Kron_constAlpha2_diffPos(Rx_matrix,Tx_matrix,wavelength,AoA);
        end
        tmp_powerShapingMat = ones(Ntx*Nrx,Ntx*Nrx);
        gamma = R_MIMO.*tmp_powerShapingMat;
        C_LOS = cholesky(gamma);
    end

```

```
for t = 1:timeSmpIs
    a = (randn(Ntx*Nrx,1)+randn(Ntx*Nrx,1)*j)/sqrt(2);
    H_Ray = reshape(C_Ray*a,Nrx,Ntx); %Rayleigh component

    a = (randn(Ntx*Nrx,1)+randn(Nrx*Nrx,1)*j)/sqrt(2);
    H_LOS = reshape(C_LOS*a,Nrx,Ntx);
    H(t,f,,:) = D*H_LOS + sqrt(2)*sigmaR*H_Ray; %eqn from book chapter
end
endTime=etime(clock,startTime)*(length(frequencies)-f);
disp(['Approximate time remaining (in seconds): ',num2str(endTime),' , Frequency: ',...
      num2str(frequencies(f))]); %just to have a feel for how long the simulation's gona take
end
sizeOf_H = size(H)
if model==6|model==12 distString='Uniform';elseif model==7|model==13 distString='Gaussian';else distString='Laplacian';end
if model==6|model==7|model==8 st='ULA, ';else st='Arbitrary Antenna Placements, ';end
saveData(H,stFreq,endFreq,['Rician, ',st,distString,' Distribution'],fileName);
```

Appendix V: MATLAB Code Written to Implement Various MIMO Channel Models

Kronecker Model 1: *LOS_Kron_constAlpha*

Code:

```
%Kronecker model 1
%function [R_MIMO] = LOS_Kron_constAlpha(Dtx,Drx,AoA,Ntx,Nrx,waveLength)
%=====
%ULA
%Interelement spacings (Dtx & Drx) can be different for Tx and Rx.
%Constant AoA considered
%Ntx = #of transmitters, Nrx = #of receivers
%Written on 04.02.2004 by Thusitha De Silva
%Ref: Branka Vucetic and Jinhong Yuan, "Effect of System Parameters and Antenna Correlation
%on the Capacity of MIMO Channels", Performance Limits of Multiple-Input Multiple-Output
%Wireless Communication System, John Wiley & Sons Ltd, 2003. pp. 25 - 36 eq:1.94

function [R_MIMO] = LOS_Kron_constAlpha(Dtx,Drx,AoA,Ntx,Nrx,waveLength)

%Convert to radians
AoA = AoA*pi/180;

%Generate the receive correlation matrix
distMatrixRx = toeplitz(0:Nrx-1)*Drx; %To account for the phase
distMatrixRx = distMatrixRx.*(-tril(ones(Nrx,Nrx),-1)+triu(ones(Nrx,Nrx),1));
Rrx = exp(-2*pi*distMatrixRx*j*sin(AoA)/waveLength); %eq:1.94

%Generate the transmit correlation matrix
distMatrixTx = toeplitz(0:Ntx-1)*Dtx;
distMatrixTx = distMatrixTx.*(-tril(ones(Ntx,Ntx),-1)+triu(ones(Ntx,Ntx),1));
Rtx = exp(-2*pi*distMatrixTx*j*sin(AoA)/waveLength);

R_MIMO = kron(Rtx,Rrx);
cholesky(kron(Rtx,Rrx));
```

Bugs:

- No known bugs as such
- Cholesky's decomposition might fail (will complain about Rrx or Rtx not being positive definite) at times if we use MATLAB's *chol* routine although the numbers generated are less than one. The *cholesky* routine written by Mr. Neil Scott has been used instead for this decomposition.

The *cholesky* routine written by Mr. Neil Scott

```
function g=cholesky(a);
% Cholesky decomposition
%
% g*gT=a
% Column version from Golub & Van Loan, 1983, paperback edition, p89
%
% Currently no checking nor reported errors
% This version programmed as Matlab's checks for positive definite first,
% which fails on a fully correlated channel matrix [1, 1 ; 1 ,1]
%
[r,c]=size(a);
if (r~=c)
    disp('Array parameter must be square')
end
%
%
%
g=a;
for kr=1:r
    g(kr,kr)=sqrt(g(kr,kr)-sum(g(kr,1:kr-1).^2));
    for k2=kr+1:r
        g(k2,kr)=(g(k2,kr)-sum(g(k2,1:kr-1).*g(kr,1:kr-1)))/g(kr,kr);
    end
end
for kr=1:r
    for kc=kr+1:r
        g(kr,kc)=0;
    end
end
end
```

Kronecker Model 2: *LOS_Kron_constAlpha2_diffPos*

Code:

```
%Kronecker model 2
%function [R_MIMO] = LOS_Kron_constAlpha2_diffPos(Rx_matrix,Tx_matrix,waveLength,AoA)
%=====
%Tx & Rx antenna placements should be given as TxMatrix=Ntx x 1 & RxMatrix=Nrx x 1
%in the form a+jb
%Constant AoA considered
%Ntx = #of transmitters, Nrx = #of receivers
%Written on 04.02.2004 by Thusitha De Silva
%Ref: Branka Vucetic and Jinhong Yuan, "Effect of System Parameters and Antenna Correlator
%on the Capacity of MIMO Channels", Performance Limits of Multiple-Input Multiple-Output
%Wireless Communication System, John Wiley & Sons Ltd, 2003. pp. 25 - 36 eq:1.94

function [R_MIMO] = LOS_Kron_constAlpha2_diffPos(Rx_matrix,Tx_matrix,waveLength,AoA)

Ntx = size(Tx_matrix,1);
Nrx = size(Rx_matrix,1);
AoA = AoA*pi/180;

%Generate the receive correlation matrix
for x = 1:Nrx
    distMatrixRx(:,x) = abs(Rx_matrix(:)-Rx_matrix(x));
end
distMatrixRx = distMatrixRx.*(-tril(ones(Nrx,Nrx),-1)+triu(ones(Nrx,Nrx),1));
Rrx = exp(-2*pi*distMatrixRx*j*sin(AoA)/waveLength); %eq:1.94

%Generate the transmit correlation matrix
for x = 1:Ntx
    distMatrixTx(:,x) = abs(Tx_matrix(:)-Tx_matrix(x));
end
distMatrixTx = distMatrixTx.*(-tril(ones(Ntx,Ntx),-1)+triu(ones(Ntx,Ntx),1));
Rtx = exp(-2*pi*distMatrixTx*j*sin(AoA)/waveLength);

R_MIMO = kron(Rtx,Rrx);
cholesky(kron(Rtx,Rrx));
```

Comments:

Same as for Model 1. Haven't tested as much since the antenna placements have to be entered everytime we need to call this routine.

Dr. Laurent Schumacher's [modified] *normalisation_uniform* Module

```
function [delta_phi_rad,Q] = normalisation_uniform(number_clusters, power_lin, AS_deg)
% Q = normalisation_uniform(number_clusters, power_lin, AS_deg)
%
% Computes the power normalising coefficients Q_k such that the
% Power Azimuth Spectrum (PAS) can be regarded as a probability
% distribution function (pdf), that is to say  $\int_{-\pi}^{\pi} \text{PAS}(\phi) d\phi = 1$ .
%
%
% STANDARD DISCLAIMER
%
% CSys is furnishing this item "as is". CSys does not provide any
% warranty of the item whatsoever, whether express, implied, or
% statutory, including, but not limited to, any warranty of
% merchantability or fitness for a particular purpose or any
% warranty that the contents of the item will be error-free.
%
% In no respect shall CSys incur any liability for any damages,
% including, but limited to, direct, indirect, special, or
% consequential damages arising out of, resulting from, or any way
% connected to the use of the item, whether or not based upon
% warranty, contract, tort, or otherwise; whether or not injury was
% sustained by persons or property or otherwise; and whether or not
% loss was sustained from, or arose out of, the results of, the
% item, or any services that may be provided by CSys.
%
% (c) Laurent Schumacher, AAU-TKN/IES/KOM/CPK/CSys - July 2001
% Modified by Thusitha De Silva on 04.02.2004 to return delta_phi_rad as well
%
% Computation
%
delta_phi_rad = (AS_deg.*pi./180).*sqrt(3);
if (number_clusters == 1)
    Q = 1/(2*delta_phi_rad);
else
    A = zeros(number_clusters);
    A(1:number_clusters-1,1) = 1/power_lin(1);
    for k=2:number_clusters
        A(k-1,k) = (-1)/power_lin(k);
    end;
    A(number_clusters,:) = delta_phi_rad;
    b = zeros(number_clusters,1);
    b(number_clusters,1) = .5;
    Q = (inv(A)*b).';
end;
%
% Validation
%
if ((sum(delta_phi_rad*Q.') - .5) > 1e-9)
    disp('Normalisation of uniform distribution failed!');
end;
```


Comments:

Uses Schumacher's *normalisation_gaussian* module to calculate Q . The original code has been modified since this was working only for values of 90^0 and 180^0 of *deltaPhi*. *Sigma* has to be entered manually in the modified version. Although any number can be entered for the *terms* term that is used to calculate the infinite summation, only about 10 terms will be used in reality due to the *erf* function in the formula.

Dr. Laurent Schumacher's [modified] *normalisation_gaussian* Module

```
%function [Q] = normalisation_gaussian(number_clusters,power_lin,delta_phi_deg,sigma_deg)
%=====
%Modified code from Laurent Schumacher with the extra input sigma_deg to calculate Q
%Can use any delta phi value instead of just 90 & 180 degrees which was a limitation of
%the original code
%Modified by Thusitha De Silva on 10.02.2004

function [Q] = normalisation_gaussian(number_clusters,power_lin, delta_phi_deg,sigma_deg)

% Conversion degree -> rad
sigma_rad = sigma_deg*pi/180;
delta_phi_rad = delta_phi_deg*pi/180;

% Computation of Q
if (number_clusters == 1)
    Q = 1/erf(delta_phi_rad/(sqrt(2)*sigma_rad));
else
    A = zeros(number_clusters);
    A(1:number_clusters-1,1) = 1/(sigma_rad(1)*power_lin(1));
    for k=2:number_clusters
        A(k-1,k) = (-1)/(sigma_rad(k)*power_lin(k));
    end;
    A(number_clusters,:) = erf(delta_phi_rad./(sqrt(2).*sigma_rad));
    b = zeros(number_clusters,1);
    b(number_clusters,1) = 1;
    Q = (inv(A)*b).';
end;

% Validation
if ~(sum(erf(delta_phi_rad./(sqrt(2).*sigma_rad))*Q.')==1 < 1e-15)
    disp('Normalisation of gaussian distribution failed!');
end;
```


Dr. Laurent Schumacher's [modified] *normalisation_laplacian* Module

```
%function [Q] = normalisation_laplacian(number_clusters,power_lin,delta_phi_deg,sigma_deg)
%=====
%Modified code from Laurent Schumacher with the extra input sigma_deg to calculate Q
%Can use any delta phi value instead of just 90 & 180 degrees which was a limitation of the
%original code
%Modified by Thusitha De Silva on 11.02.2004

function [Q] = normalisation_laplacian(number_clusters,power_lin,delta_phi_deg,sigma_deg)
%Convert to radians
sigma_rad = sigma_deg.*(pi/180);
delta_phi_rad = delta_phi_deg.*(pi/180);

% Computation of Q
if (number_clusters == 1)
    Q = 1/(1-exp((-1)*sqrt(2)*delta_phi_rad)/sigma_rad));
else
    A = zeros(number_clusters);
    A(1:number_clusters-1,1) = 1/(sigma_rad(1)*power_lin(1));
    for k=2:number_clusters
        A(k-1,k) = (-1)/(sigma_rad(k)*power_lin(k));
    end;
    A(number_clusters,:) = ones(1,number_clusters)-exp((-1).* ...|
        sqrt(2).* ...
        delta_phi_rad)./ ...
        sigma_rad);
    b = zeros(number_clusters,1);
    b(number_clusters,1) = 1;
    Q = (inv(A)*b).';
end;
% Validation
if ~(sum((ones(1,number_clusters)-exp((-1).*sqrt(2).* ...
        delta_phi_rad)./sigma_rad))*Q.')==1 ...
    < 1e-15)
    disp('Normalisation of laplacian distribution failed!');
end;
```

Kronecker Model 6,7,8, 12,13,14: *Rician Channel Models*

These models have been implemented in the main module since these ones use both LOS and Rayleigh modules.

Kronecker Model 9,10,11: *Rayleigh Models with user specified antenna placements*

Use the same formulas as models 3,4,5 to calculate Rrx or Rtx based on scatterer distribution. Difference between models 3,4,5 and 9,10,11 is that in the latter models, equation (2) is used when calculating the inter element distances.

The source code is same except the following few lines in the getCorrelation function, where inMatrix is the transmit or receive antenna placement

```
function [corrMatrix] = getCorrMatrix(waveLength,noOfClusters,inMatrix,sigma,deltaPhi,phiZero,Q,N,terms)
for x = 1:N
    distMatrix(:,x) = abs(inMatrix(:)-inMatrix(x));
end
distMatrix = distMatrix.*(-tril(ones(N,N),-1)+triu(ones(N,N),1));
```

Model 15: Tap Delay Line Model**Code:**

```

%function [Rx_matrix] = TapDelayLineModel(Nrx,Tx_matrix,noOfTaps,samplFreq,env,C)
%-----
%Written by Thusitha De Silva on 10.02.2004
%Ref: K.I. Pedersen, J.B. Andersen, J.P. Kermoal, and P.E. Mogensen, "A Stochastic Multiple-Input-Multiple-Output
%Radio Channel Model for Evaluation of Space-Time Coding Algorithms", Proceedings of VTC 2000 Fall, pp. 893-897,
%Boston, United States, September 2000. - eqn:01
%The PDPs have been taken from:
%IEEE C802.20-03/92, "Draft 802.20 Permanent Document, Channel Models for IEEE 802.20 MBWA System, Simulations -
% Rev 03", 2003-11-06. pp 13-14.
%The Tx_Matrix has to be given as an input in the form a+jb as a Ntx x number of time samples matrix.
%C is equal to R_MIMO is a power shaping matrix of all ones is assumed

function [Rx_matrix] = TapDelayLineModel(Nrx,Tx_matrix,noOfTaps,samplFreq,env,C)
Ntx = size(Tx_matrix,1);
%A bit of error checking
if noOfTaps <= 0 | noOfTaps>11
    disp('Error: Number of taps cannot be zero, negative or >11');
    Rx_matrix=0;
    return
elseif samplFreq <= 0
    disp('Error: Sampling frequency cannot be zero or negative');
    Rx_matrix=0;
    return
elseif env <=0 | env>7
    disp('Error: Environment selection should be between 1-7');
    Rx_matrix=0;
    return
end

relDelayTable = [0 50 110 170 290 310 310 310 310 310 310;...
                0 100 200 300 500 700 700 700 700 700 700;...
                0 110 190 410 410 410 410 410 410 410 410;...
                0 200 800 1200 2300 3700 3700 3700 3700 3700 3700;...
                0 310 710 1090 1730 2510 2510 2510 2510 2510 2510;...
                0 300 8900 12900 17100 20000 20000 20000 20000 20000 20000;...
                0 100 300 500 800 1100 1300 1700 2300 3100 3200];

avgPowerdBTable= [0 -3 -10 -18 -26 -32 0 0 0 0 0;...
                 0 -3.6 -7.2 -10.8 -18 -25.2 0 0 0 0 0;...
                 0 -9.7 -19.2 -22.8 0 0 0 0 0 0 0;...
                 0 -0.9 -4.9 -8 -7.8 -23.9 0 0 0 0 0;...
                 0 -1 -9 -10 -15 -20 0 0 0 0 0;...
                 -2.5 0 -12.8 -10 -25.2 -16 0 0 0 0 0;...
                 -4 -3 0 -2.6 -3 -5 -7 -5 -6.5 -8.6 -11];

%Environments - just uncomment if need to be displayed
% if env == 1 disp('Indoor office test environment - Channel A');
% elseif env == 2 disp('Indoor office test environment - Channel B');
% elseif env == 3 disp('Outdoor to indoor and pedestrian test environment - Channel A');
% elseif env == 4 disp('Outdoor to indoor and pedestrian test environment - Channel B');
% elseif env == 5 disp('Vehicular test environment, high antenna - Channel A');
% elseif env == 6 disp('Vehicular test environment, high antenna - Channel B');
% elseif env == 7 disp('Typical urban test environment');
% end

relDelays = relDelayTable(env,:); %Get the proper delay spread based on environment
realTaps = round(relDelays/samplFreq); %relDelays/samplFreq gives the taps. But some of these could be the same. Eg. [1 2 2 4 5]
avgPowerdB = avgPowerdBTable(env,:); %Avg. powers from table based on environment
[taps,PDP] = getPDP(realTaps(1:noOfTaps),avgPowerdB(1:noOfTaps)); %Get rid of repeating taps
for x = 1:length(taps) %Create the A array
    A_array(:,taps(x)+1) = C*PDP(x)*(randn(Nrx*Ntx,1) + randn(Nrx*Ntx,1)*j)/sqrt(2);
end
A_array;

%Filter and generate the receive signals
Rx_matrix = zeros(Nrx,size(Tx_matrix,2));
for rx = 1:Nrx
    for tx = 1:Ntx
        fltrCoefcntVec = A_array((rx-1)*Ntx+tx,:);
        Rx_matrix(rx,:) = Rx_matrix(rx,:) + filter(fltrCoefcntVec,1,Tx_matrix(tx,:));
    end
end
end
Rx_matrix;

```

```
%This function gets rid of any repeating taps and adds the powers. Eg. [1 2 2 4 4 5] -> [1 2 4 5] with powers added up
function [tps,pdp] = getPDP(realTaps,avgPowerdB)
x = 1;
while x<length(realTaps) & x~=0
    y = x;
    cntr = 0;
    while y<length(realTaps)
        if realTaps(y) == realTaps(y+1)
            cntr = cntr+1;
            y = y+1;
        else
            break
        end
    end
    if cntr ~= 0
        realTaps(x:x+cntr-1) = [];
        avg = sum(avgPowerdB(x:x+cntr));
        avgPowerdB(x:x+cntr-1) = [];
        avgPowerdB(x) = avg;
        x = x-1;
    else
        x = x+1;
    end
end
tps = realTaps;
avgPowerdB;
pdp = sqrt(10.^(avgPowerdB./10));
```

Comments:

The *T_x Matrix* has to be given as an input in the form a+jb as a *N_{tx} x number of time samples* matrix. No known bugs or errors but hasn't been tested against measured data hence the validity of the generated data is doubtful.

Model 16: Link Level Channel Model

Code:

```

%function [A_array] = linkLevelModel(Nrx,Ntx,alpha,noOfTaps,samplFreq,env,wavelength)
%-----
%Written by Thusitha De Silva on 10.02.2004
%Ref: K.I. Pedersen, J.B. Andersen, J.P. Kermoal, and P.E. Mogensen, "A Stochastic Multiple-Input-Multiple-Output
%Radio Channel Model for Evaluation of Space-Time Coding Algorithms", Proceedings of VTC 2000 Fall, pp. 893-897,
%Boston, United States, September 2000. - eqn:01

%The PDPs have been taken from:
%IEEE C802.20-03/92, "Draft 802.20 Permanent Document, Channel Models for IEEE 802.20 MBWA System, Simulations -
% Rev 03", 2003-11-06. pp 13-14.
%Attempted to account for the PAS as detailed on page 13 and 14 but has only been partially implemented.
%Uses AoA = randi(1,67,5,35) for MS; and AoA = randi(1,50,2) for BS; but the 67.5 degrees and the 50 degrees
%should be normalised for all the scenarios. Hasn't been completed due to time constraints and lack of knowledge
%of the whole model. Not included as a function call from the main module (main.m)

function [A_array] = linkLevelModel(Nrx,Ntx,alpha,noOfTaps,samplFreq,env,wavelength)

if noOfTaps <= 0 | noOfTaps>11
    disp('Error: Number of taps cannot be zero, negative or >11');
    Rx_matrix=0;
    return
elseif samplFreq <= 0
    disp('Error: Sampling frequency cannot be zero or negative');
    Rx_matrix=0;
    return
elseif env <=0 | env>4
    disp('Error: Environment selection should be between 1-4');
    Rx_matrix=0;
    return
end

%PDPs from, pp 13-14 of IEEE reference
relDelayTable = [0 200 800 1200 2300 3700 3700 3700 3700 3700;...
                0 310 710 1090 1730 2510 2510 2510 2510 2510;...
                0 300 8900 12900 17100 20000 20000 20000 20000 20000;...
                0 100 300 500 800 1100 1300 1700 2300 3100 3200];
avgPowerdBTable = [0 -0.9 -4.9 -8 -7.8 -23.9 0 0 0 0;...
                  0 -1 -9 -10 -15 -20 0 0 0 0;...
                  -2.5 0 -12.8 -10 -25.2 -16 0 0 0 0;...
                  -4 -3 0 -2.6 -3 -5 -7 -5 -6.5 -8.6 -11];

%Environments
if env == 1 disp('Outdoor to indoor and pedestrian test environment - Channel B');
elseif env == 2 disp('Vehicular test environment, high antenna - Channel A');
elseif env == 3 disp('Vehicular test environment, high antenna - Channel B');
elseif env == 4 disp('Typical urban test environment');
end

relDelays = relDelayTable(env,:); %Get the proper delay spread based on environment
realTaps = round(relDelays/samplFreq); %relDelays/samplFreq gives the taps. But some of these could be the same. Eg. [1 2 2 4 5]
avgPowerdB = avgPowerdBTable(env,:); %Avg. powers from table based on environment
[taps,PDP] = getPDP(realTaps(1:noOfTaps),avgPowerdB(1:noOfTaps)); %Get rid of repeating taps

distMatrixTx = (0:Ntx-1)*0.5*wavelength;
distMatrixRx = (0:Nrx-1)*0.5*wavelength;
for x = 1:length(taps)
    AoA = randi(1,67,5,35);
    thetaRx = [exp(2*pi*distMatrixRx*sin(AoA)/wavelength)'];
    AoA = randi(1,50,2);
    thetaTx = [exp(2*pi*distMatrixRx*sin(AoA)/wavelength)'];
    for tx = 1:Ntx
        A_array(:,tx,taps(x)+1) = PDP(x)*thetaTx(tx)*thetaRx(:);
    end
end
A_array

%This function gets rid of any repeating taps and adds the powers. Eg. [1 2 2 4 4 5] -> [1 2 4 5] with powers added up
function [tps,pdp] = getPDP(realTaps,avgPowerdB)
x = 1;
while x<length(realTaps) & x~=0
    y = x;
    cntx = 0;
    while y<length(realTaps)
        if realTaps(y) == realTaps(y+1)
            cntx = cntx+1;
            y = y+1;
        else
            break
        end
    end
    if cntx ~= 0
        realTaps(x:x+cntx-1) = [];
        avg = sum(avgPowerdB(x:x+cntx));
        avgPowerdB(x:x+cntx-1) = [];
        avgPowerdB(x) = avg;
        x = x-1;
    else
        x = x+1;
    end
end
tps = realTaps;
avgPowerdB;
pdp = sqrt(10.^(avgPowerdB./10));

```

Comments:

This module is not complete. Comments are included at the top of the module.