Institutionen för systemteknik

Department of Electrical Engineering

"Wireless Channel Models for the DTV Frequency Spectrum"

Master Thesis in Communication Systems

by

María Hiniesto

Departement of Electrical Engineering

Linköping universitet

SE-581 83 Linköping, Sweden

Acknowledgments

This project put an end to my university studies, in which has been crucial the support from many people whose words of encouragement and confidence have helped me to finally finish this period.

First, thanks to my tutor Danyo Danev, for the time and supervision of this work, without his knowledge and guidance this would not have been possible.

Secondly, I thank my parents for the possibility that I may have to spend this last year in Linköping, where I have lived a great experience that has enriched me as a person and made me grow inside. Thanks for your patience and understanding in the difficult times that have taken place during this academic period.

Thanks also to my sisters (Loli and Ana) and brothers (Luis and Geovani), each of them has known what to say me every moment in my life.

Thanks to my friends from the University Carlos III, particularly Bea and Juan, who shared many good times with me and always helped in every way possible, making it more bearable these years.

Thanks to all the Erasmus people, Javier, Carlos, Nacho, Miguel, Amin, David, Camilla, Helene, Nacima, Vesta and many others, who I will never forget.

And last but not least, thanks to Samu, for give me so many moments of happiness which no doubt have made easier this year.

Thank you very much!

Contents

1. Background

The present project is a study about the parameters and methods associated with the spatial channel modelling that are common to the needs of the 3GPP and 3GPP2 organizations.

We will work on physical parameters such as power delay profiles, angles spread, dependencies between parameters and a system evaluating methodology as the same way the IEEE 802.20 Working Group has done it before, but with an important difference; we’ll work with a different frequency in order to study the characteristics of the wireless channel models at low frequencies. We’ll study the behaviour for the DTV frequency band, seeing the benefits of allocating portions of the spectrum between 470 MHz and 862 MHz for broadband wireless services.

Link level simulation alone will not be used for algorithm comparision because they reflect only one snapshot of the channel behaviour. This is one of the reasons link level simulations will be used only for calibration (the comparision of performance results from different implementations of a given algorithm) and we’ll also study the system level simulation for the final algorithm.

In channel modelling we have to consider also different scenarios as a basic factor, because different test environments have different channel parameters. Urban and rural areas will not have the same channel characteristics; neither will tow different places within a building as we’ll see.

1.1. Terminology

3GPP	3rd Generation Partnership Project
3GPP2	3rd Generation Partnership Project 2
AAS	Adaptative Antenna Systems
AoA	Angle of Arrival
AoD	Angle of Departure
AS	Angle Spread = Azimuth Spread = σ_{AS}
AWGN	Additive White Gaussian Noise
BS	Base Station = Node-B = BTS
DoT	Direction of Travel
DS	Delay Spread= σ_{DS}
ELG	European Launching Group
GSL	GNU Scientific Library
HDTV	High Definition Television
ISI	Inter-Symbol Interference
LOS	Line of Sight
MF	Multi Frequency Network

MIMO	Multiple Input Multiple Output
MS	Mobile Station = UE = Terminal = Subscriber Unit
NLOS	Near Line of Sight
PAS	Power Azimuth Spectrum
PDP	Power Delay Profile
PL	Path Loss
SCM	Spatial Channel Model
SF	Lognormal Shadow Fading random variable = σ_{SF}
SFN	Single Frequency Network
SH	Lognormal Shadow Fading constant = σ_{SH}
SISO	Single Input Single Output
UE	User Equipment = MS

1.2. List of Symbols

σ_{AS}	Angle Spread or Azimuth Spread
σ_{DS}	Delay Spread
σ_{SF}	Lognormal Shadow Fading Random Variable
σ_{SH}	Lognormal Shadow Fading Constant
$\eta(a,b)$	Represents a Random Normal (Gaussian) Distribution with mean a and variance b

2. Introduction

2.1. Wireless Networks

Cellular networks, such as GSM and 3G, are today available all over the world. Properly set, these networks have the capability of providing high performance wireless connectivity. For the design and testing of wireless communication systems, channel model is an absolute necessity and that’s why any wireless communication system needs to specify a propagation channel model that can act as a basis for performance evaluation and comparision. Communication technologies are advancing quickly so we need to refined these models as further characteristics of the channel can be exploited and thus needs to be modelled.

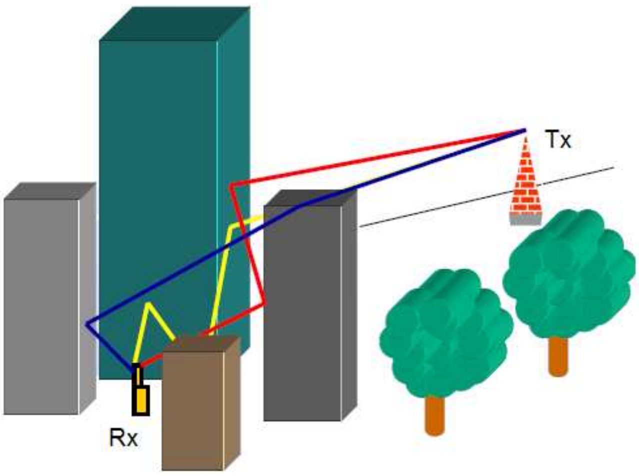

Figure 2.1: Example of wireless channels

Instead of the fixed networks where electrons moving in copper cables, mobile networks make use of electro-magnetic waves which propagate in air, see Fig 2.1 for a simple example of mobile channel. So we can say wireless channels carry out Maxwell’s equation that electromagnetic fields propagate in free space like light.

When a voltage is applied to an antenna, it creates an electromagnetic field that propagates in all directions (although antenna geometry affects how much power flows in any given direction) that induces electric currents in the receiver's antenna.

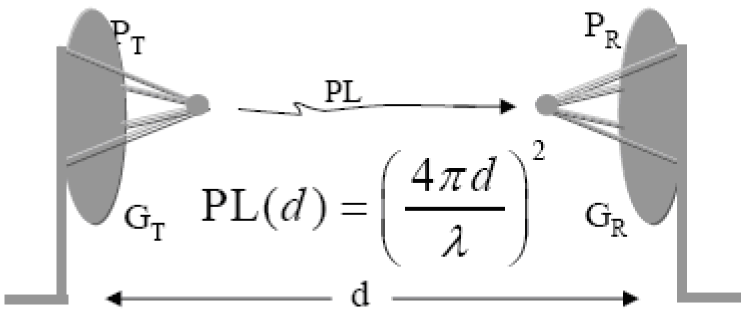

Figure 2.2: Free space propagation

In general terms, the dominant factor is the relation of the antenna's size to the field's wavelength. Frequency and wavelength are related, the wavelength (in metres) is equal to the speed of light (in metres/sec) divided by the frequency (in Hertz - Hz):

$$\lambda = c/f$$

Thus, wavelength and frequency are inversely related: High frequency corresponds to small wavelengths. For example, a 10 MHz electromagnetic field has a wavelength of 30 m.

$$\lambda = \frac{c}{f} = \frac{3.10^8\, m/s}{10\, MHz} = 30\, m$$

Antennas having a size or distance from the ground comparable to the wavelength radiate fields most efficiently. Consequently, the lower the frequency the bigger the antenna must be. Because most information signals are baseband signals, having spectral energy at low frequencies, they must be modulated to higher frequencies to be transmitted over wireless channels.

On the other hand, in mobile networks, the radio channel characteristics differ depending on location of base stations and information technology, movement of these handheld devices and also obstacles in the surrounding. Different channels could affect the performance for the end user. Therefore, information and knowledge about the radio channel characteristics in different scenarios would be very helpful and make it possible to better understand new 3G services.

2.2. MIMO Wireless Systems

Most wireless communication systems being used at present are Single Input Single Output (SISO) systems where a single transmit (Tx) antenna is used for transmission to a single receive (Rx) antenna.

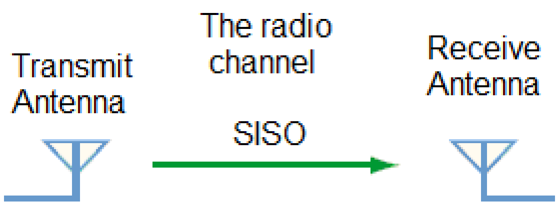

Figure 2.3: Single Input Single Output system

Additional Rx antennas can be used to provide diversity at the receiver. In the days to come, this scenario is likely to change with the advent of Multiple Input Multiple Output (MIMO) communication systems and this will be the scenario we will consider because of the benefit from multipath propagation and multiply transfer rates by taking advantage of random fading and multipath delay spread [2.1]. In addition, MIMO provides spatial diversity both at the transmitter and the receiver, thus improving the transmission quality in terms of the bit-error rate (BER).

The MIMO approach requires two or more transmit antennas and two or more receive antennas and must have at least as many receivers as data streams transmitted. But we don't have to confuse the number of these transmit streams with the number of transmit antennas.

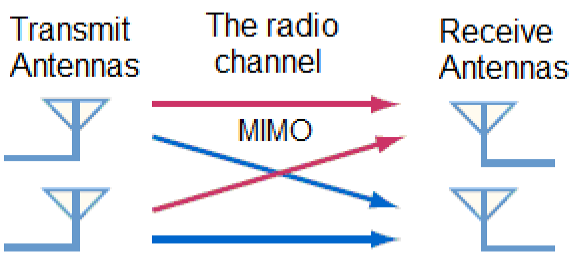

Figure 2.4: Multiple Input Multiple Output system

For MIMO operation, the transmissions from each antenna must be uniquely identifiable so that each receiver can determine what combination of transmissions it has received. A basic form of MIMO assigns one data stream to each antenna. The channel then mixes up the two transmissions such that at the receivers each antenna sees a combination of each stream. [2.2]

The MIMO channel has to be described for all transmit and receive antenna pairs. Therefore from a system level perspective, a linear time-variant MIMO channel is represented by an N x M channel matrix, where N and M are the number of receive and transmit antennas respectively.

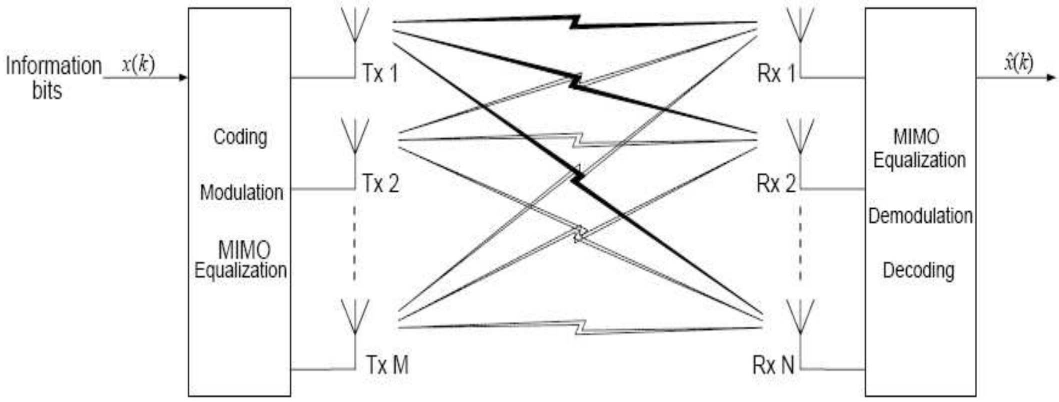

Figure 2.5: Block diagram of an M x N MIMO wireless system.

There are differents ways to classify MIMO channel models, some of them based on measurements, the type of channel, etc.

A potential way of distinguishing the individual models is according to the type of channel that is being considered, that is, *narrowband* (flat fading) versus *wideband* (frequency selective) models, *time-varying* versus *time-invariant* models, and so forth. Narrowband MIMO channels are completely characterized in terms of their spatial structure. In contrast, wideband (frequency-selectivity) channels require additional modeling of the multipath channel characteristics. [2.3]

With time-varying channels, one additionally requires a model for the temporal channel evolution according to certain Doppler characteristics.

2.3. Physical and Analytical Wireless Channel Models

The fundamental distinction for the individual models is between *physical models* and *analytical models*.

1) *Analytical channel models:* capture physical wave propagation and antenna configuration simultaneously by describing the impulse response (equivalently, the transfer function) between the antenna arrays at both link ends. Each entry in that matrix gives the transfer function from the M-th transmit to the N-th receive antenna element.

$$H(t,\tau) = \begin{pmatrix} h_{11}(t,\tau) & h_{12}(t,\tau) & \cdots & h_{1m}(t,\tau) \\ h_{21}(t,\tau) & h_{22}(t,\tau) & \cdots & h_{2m}(t,\tau) \\ \vdots & \vdots & \ddots & \vdots \\ h_{n1}(t,\tau) & h_{n2}(t,\tau) & \cdots & h_{nm}(t,\tau) \end{pmatrix}, \qquad (1)$$

$h_{ij}(t,\tau)$ denotes the time-variant impulse response between the *j*th transmit antenna and the *i*th receive antenna.

The channel matrix (1) includes the effects of antennas (type, configuration, etc.) and frequency filtering (bandwidth-dependent). It can be used to formulate an overall MIMO input-output relation between the length-*m* transmit signal vector **s**(*t*) and the length-*n* receive signal vector y(t) as:

$$y(t) = \int_{\tau} H(t,\tau)s(t-\tau)d\tau + n(t)$$

where n(t) models noise and interference.

So in contrast to physical models, as we'll see later, analytical channel models characterize the impulse response (equivalently, the transfer function) of the channel between the individual transmit and receive antennas in a mathematical/analytical way without explicitly accounting for wave propagation. The individual impulse responses are subsumed in a (MIMO) channel matrix. Analytical models are very popular for synthesizing MIMO matrices in the context of system and algorithm development and verification.

All of the analytical models are based on the assumption that the entries of the transfer function matrix are zero-mean complex Gaussian, with the possible addition of a line-of-sight component.

2) *Physical channel models*: focus on of the double-directional propagation mechanisms between the location transmitter and receiver describing the parameters of multipath components DOD, DOA, delay, and complex amplitudes. Such models are highly useful because they are independent of antenna configurations and describe the physical propagation alone.

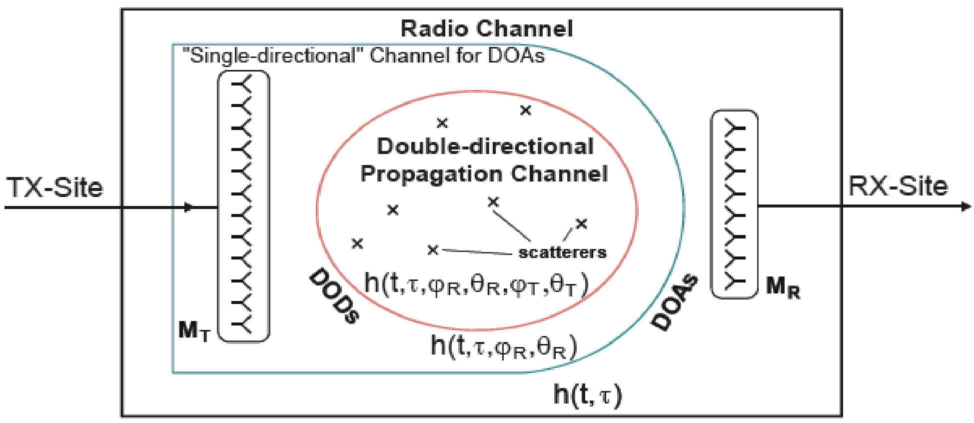

Figure 2.6: Double-Directional Propagation Channel [2.4]

Note that it is easy to obtain an analytical channel model from a double-directional model but not the other way around.

Among the *physical* models, we can distinguish between:

- deterministic models (e.g. ray tracing, stored measurements)
- geometry-based stochastic models (e.g. the GSCM of COST 259/273)
- stochastic models (without geometrical input).

3. The SCM Channel Models

3.1. Description

The Spatial Channel Model (SCM) is a geometric or ray-based model based on stochastic modelling of scatters.

It was developed by 3GPP/3GPP2 to be a common reference for evaluating different MIMO concepts in outdoor environments at a center frequency of 2 GHz and a system bandwidth of 5 MHz but we have introduced an important change using a different frequency of 700 MHz in order to study the behaviour at low frequencies.

Let see the proceeding the 3GPP/3GPP2 have used which has been the same methodology we have follow.

It defines three environments:

- Suburban Macro
- Urban Macro
- Urban Micro

and a fixed number of 6 “paths” in every scenario, each representing a Dirac function in delay domain, but made up of 20 spatially separated “sub-paths” according to the sum-of-sinusoids method.

3.2. Parameters

Each resolvable path is characterized by its own spatial channels parameters:

- Angle spread
- Angle of arrival
- Power azimuth spectrum (PAS)

We have modelled the path powers, path delays, and angular properties for both sides of the link as random variables defined through probability density functions (PDFs) and cross-correlations. All parameters, except for fast-fading, have been drawn independently in time, in what is termed “drops”. [3.1]

3.2.1. Calibration Model

Link level simulation alone will not be used for the algorithm comparision because they reflect only one snapshot of the channel behaviour; this is one of the reasons link level simulations will be used only for calibration. [3.2]

The calibration model is *not* intended for performance assessment of algorithms or systems, but a simplified channel model whose purpose is to check the correctness of simulation implementations comparing the performance of the algorithm in the "calibration" channels allows in order to see if two implementations are equivalent. [2.3]

It is a spatial extension of the ITU-R channel models, which describe the wideband characteristics of the channel as a tapped delay line. Taps with different delays are independently fading, and each tap is characterized by its own power azimuth spectrum (which exhibits an uniform or Laplacian distribution), angular spread (AS), and mean direction, at both the MS and the BS. The parameters (i.e., angular spread, mean direction, etc.), are fixed; thus the model represents stationary channel conditions. The Doppler spectrum is defined implicitly by introducing speed and direction of travel of the MS.

The model also defines a number of antenna configurations. Given those, the physical model can be transformed into an equivalent analytical model as discussed in Section 2.3.

The next table shows a summary of the parameters we are going to work on and we'll explain later and we'll see the way they are connect them.

MODEL		CASE I		CASE II		CASE III		CASE IV	
Corresponding 3GPP Designator*		**Case B**		**Case C**		**Case D**		**Case A**	
Corresponding 3GPP2 Designator*		**Model A, D, E**		**Model C**		**Model B**		**Model F**	
PDP		**Modified Pedestrian A**		**Vehicular A**		**Pedestrian B**		**Single Path**	
# of Paths		1) 4+1 (LOS on, K = 6dB) 2) 4 (LOS off)		6		6		1	
Relative Path Power (dB)	**Delay (ns)**	1) 0.0 2) -∞	0	0,0	0	0.0	0	0	0
		1) -6.51 2) 0.0	0	-1.0	310	-0.9	200		
		1) -16.21 2) -9.7	110	-9.0	710	-4.9	800		
		1) -25.71 2) –19.2	190	-10.0	1090	-8.0	1200		
		1) -29.31 2) -22.8	410	-15.0	1730	-7.8	2300		
				-20.0	2510	-23.9	3700		
Speed (km/h)		1) 3 2) 30, 120		3, 30, 120		3, 30, 120		3	
UE/Mobile Station	**Topology**	Reference 0.5λ		Reference 0.5λ		Reference 0.5λ		N/A	
	PAS	1) **LOS on:** Fixed AoA for LOS component, remaining power has 360 degree uniform PAS. 2) **LOS off:** PAS with a Lapacian distribution, RMS angle spread of 35 degrees per path		RMS angle spread of 35 degrees per path with a Lapacian distribution Or 360 degree uniform PAS.		RMS angle spread of 35 degrees per path with a Lapacian distribution		N/A	
	DoT (degrees)	0		22.5		-22.5		N/A	
	AoA (degrees)	22.5 (LOS component) 67.5 (all other paths)		67.5 (all paths)		22.5 (odd numbered paths), -67.5 (even numbered paths)		N/A	
Node B/ Base	**Topology**	Reference: ULA with 0.5λ-spacing or 4λ-spacing or 10λ-spacing						N/A	

MODEL		CASE I	CASE II	CASE III	CASE IV
	PAS	Lapacian distribution with RMS angle spread of 2 degrees or 5 degrees, per path depending on AoA/AoD			N/A
	AoD/AoA (degrees)	50^o for 2^o RMS angle spread per path 20^o for 5^o RMS angle spread per path			N/A

Table 3.1 Summary of Suggested SCM Link Level Parameters for Calibration Purposes [3.2]

a) Spatial parameters for the BS

- **BS antenna pattern**

We’ll consider three different values for reference antenna element spacing:

- o 0.5λ
- o 4λ
- o 10λ

The 3-sector antenna pattern is used for BS, which is plotted in Figure 3.1 and is specified by

$$A(\theta) = -\min\left[12\left(\frac{\theta}{\theta_{3dB}}\right)^2,\ A_m\right] \quad \text{where} \quad -180 \le \theta \le 180$$

- o θ: angle between the direction of interest and the broadside[1] of the antenna array.
- o θ_{3dB}: 3dB beam-width in degrees
- o A_m: maximum attenuation

For a 3 sector scenario θ_{3dB} is 70 degrees, and A_m = 20dB. The antenna broadside pointing direction is illustrated by Figure 3.2 for a 3-sector scenario. The antenna gain, as specified by 3GPP document [3.2], is 14 dBi for a 3-sector scenario.

[1] *Broadside:* refers to the direction from which the signal is coming perpendicularly to the Multi-Element Array. Antenna array shows the maximum gain at its broadside direction.

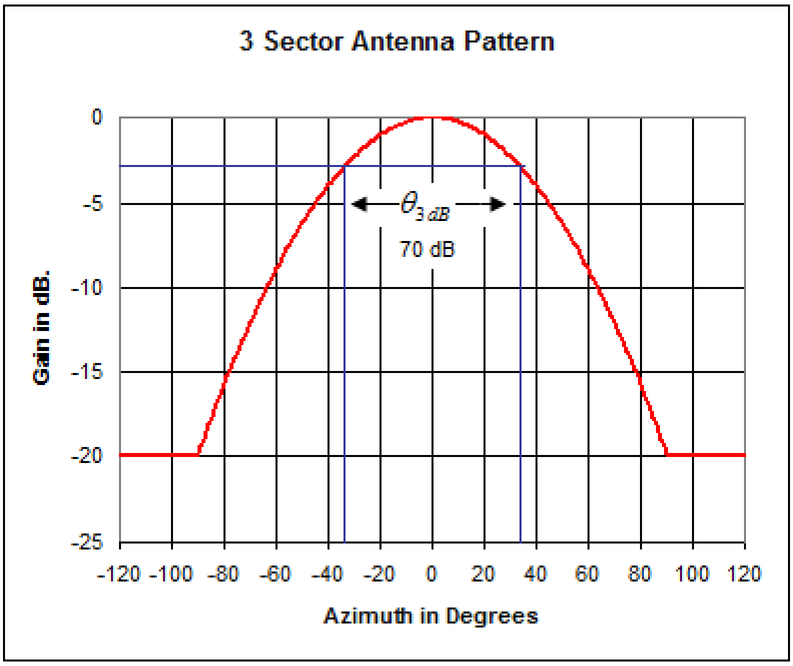

Figure 3.1: Antenna pattern for 3-sector cells

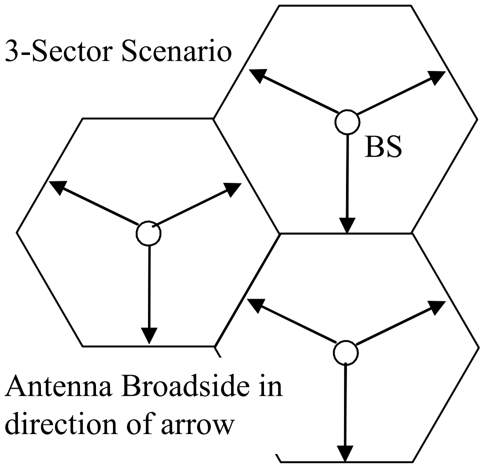

Figure 3.2: Broadside pointing direction of antenna array for 3-sector cells

For a 6 sector scenario, θ_{3dB} is 35 degree, A_m =23dB, which results in the antenna pattern shown in the figure below, and the broadside pointing direction illustrated by Figure 3.4 . The gain specified by 3GPP document [3.1] is 17dBi for a 6 sector scenario.

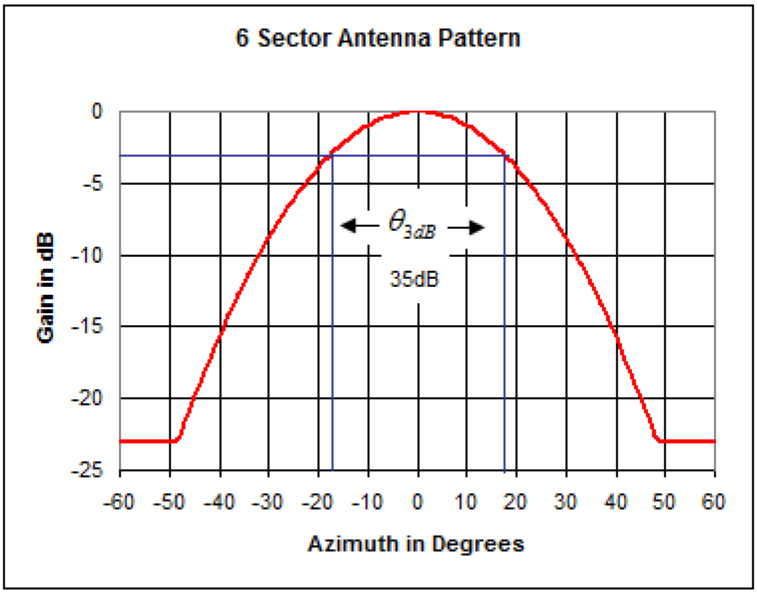

Figure 3.3: Antenna Pattern for 6-sector cells

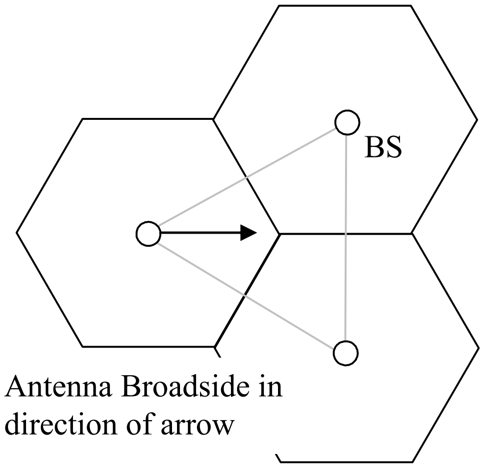

Figure 3.4: Broadside Pointing Direction for 6-sector cells

- **Per-path BS angle spread (AS) or Narrowband Angle Spread**

Is defined as the root mean square (RMS) of angles with which an arriving ray's power is received by the base station Multi-Element Array. The individual path powers are defined in the temporal ITU SISO channel models. Two values of BS angle spread (each associated with a corresponding mean angle of departure, AoD) will be considered in this project:

- AS: 2 degrees at AoD = 50^o
- AS: 5 degrees at AoD = 20^o

Attention should be paid when comparing the link level performance between the two angles spread values since the BS antenna gains for the two corresponding AoDs are different.

- **Per-path BS angle of departure**

The Angle of Departure (AoD) is defined to be the mean angle with which an arriving or departing ray's power is received or transmitted by the BS array with respect to the broadside. The two values considered are:

- AoD: 50 degrees (associated with the RMS Angle Spread of 2^o)
- AoD: 20 degrees (associated with the RMS Angle Spread of 5^o)

- **Per-path BS power azimuth spectrum**

The Power Azimuth Spectrum (PAS) of a ray arriving at the base station Multi-Element Array exhibits Laplacian distribution. For an AoD $\overline{\theta}$ and RMS angle-spread σ, the BS per path PAS value at an angle θ is given by:

$$P(\theta,\sigma,\overline{\theta}) = N_o \exp\left[\frac{-\sqrt{2}\left|\theta-\overline{\theta}\right|}{\sigma}\right] G(\theta)$$

where both angles $\overline{\theta}$ and θ are given with respect to the broadside of the MEA. It is assumed that all antenna elements' orientations are aligned. Also, P is the average received power and G is the numeric base station antenna gain given by

$$G(\theta) = 10^{0.1A(\theta)}$$

Finally, N_o is the normalization constant:

$$N_o^{-1} = \int_{-\pi+\overline{\theta}}^{\pi+\overline{\theta}} \exp\left[\frac{-\sqrt{2}\left|\theta-\overline{\theta}\right|}{\sigma}\right] G(\theta)\, d\theta$$

In the above equation, θ represents path components (sub-rays) of the path power arriving at an AoD.

b) Spatial parameters for the MS

- **MS antenna pattern**

At the MS, the Multi-Element Array element spacing is 0.5λ, instead of the three different values we had for the BS.

For each antenna element at the MS, the antenna pattern will be assumed omni directional with an antenna gain of -1 dBi.

- **Per-path MS angle spread (AS)**

The MS per-path AS is defined as the root mean square (RMS) of angles of an incident path's power at the MS array. Two values of the path's angle spread are considered:

- AS: 104^o (results from the PAS with a uniform distribution over 360 degree)
- AS: 35^o for a Laplacian PAS with a certain path specific Angle of Arrival (AoA).

- **Per-path MS angle of arrival**

The per-path Angle of Arrival (AoA) is defined as the mean angle of an incident ray at the MS MEA with respect to the broadside as shown in the figure below,

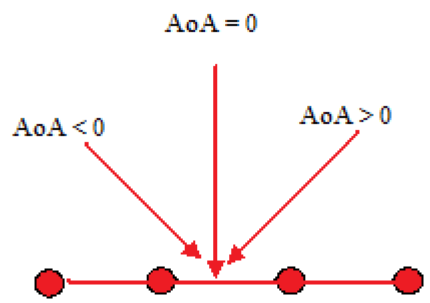

Figure 3.5: Angle of arrival orientation at the MS

The AoA analysis can provide an illustration of the PAS at MS Multi-Element Array. The IEEE Working Group suggest three different per-path AoA values at the MS are suggested for the cases of a non-uniform PAS,

- AoA: -67.5 degrees (associated with an RMS Angle Spread of 35^o)
- AoA: +67.5 degrees (associated with an RMS Angle Spread of 35^o)
- AoA: +22.5 degrees (associated with an RMS Angle Spread of 35^o or with an LOS component)

- **Per-path MS power azimuth spectrum**

The Power Azimuth Spectrum (PAS) of a ray arriving at the MS is modeled as either Laplacian distribution or uniform distribution over 360^o. Since an omni antenna is assumed at MS, the received per path PAS will remain either Laplacian or uniform. For an incoming AoA $\bar{\theta}$ and RMS angle spread σ , the MS per-path Laplacian PAS value at an angle θ is given by:

$$P(\theta, \sigma, \bar{\theta}) = N_o \exp\left[\frac{-\sqrt{2}\left|\theta - \bar{\theta}\right|}{\sigma}\right],$$

where both angles $\bar{\theta}$ and θ are given with respect to the broadside of the Multi-Element Array. It is assumed that all antenna elements' orientations are aligned. Also, P is the average received power and N_o is the normalization constant:

$$N_o^{-1} = \int_{-\pi+\bar{\theta}}^{\pi+\bar{\theta}} \exp\left[\frac{-\sqrt{2}\left|\theta - \bar{\theta}\right|}{\sigma}\right] d\theta \, .$$

In the above equation, θ represents path components (sub-rays) of the path power arriving at an incoming AoA $\bar{\theta}$.

- **MS direction of travel**

The mobile station direction of travel is defined with respect to the broadside of the mobile antenna array as shown in the figure below,

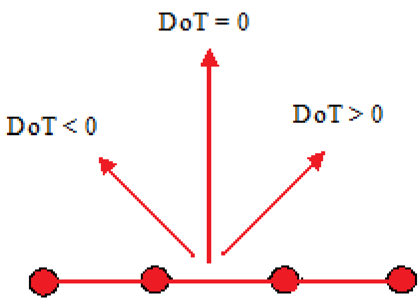

Figure3.6: Direction of Travel for MS

- **Per-path Doppler spectrum**

The per-path Doppler Spectrum is defined as a function of DoT, per-path PAS, and AoA at MS. Doppler spectrum is affected by the PAS and the Angle of Arrival. Doppler spectrum affects the time-domain behaviour of the channel.

3.2.2. System-Simulation Model

Instead of consider a single BS transmitting to a single MS, the system simulations consist of multiple cells, sectors, BSs and MSs.

The received signal at the MS consists of N time-delayed multipath replicas of the transmitted signal and each path consists of M subpaths.

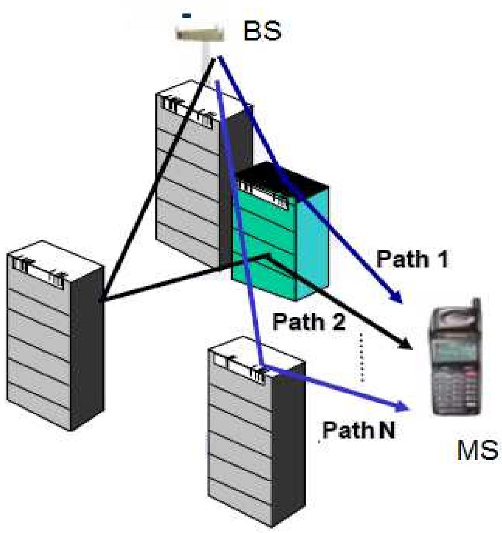

Figure 3.7: System Simulation Model

The overall procedure for generating the channel matrices consists of three basic steps:

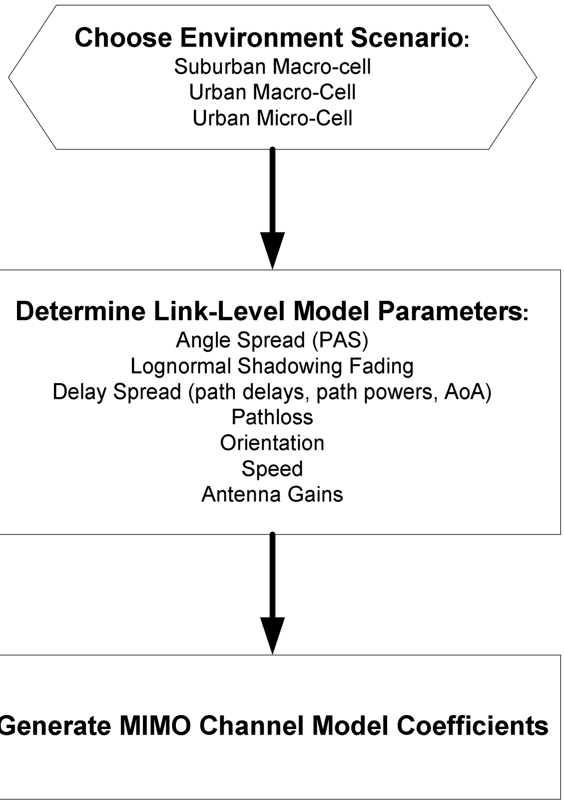

Figure 3.8: The flow chart for the generation of MIMO channel model coefficients [3.3]

As we have said before, the received signal at MS consists of ***N*** time-delayed multi-path replicas of the transmitted signal. These ***N*** paths are defined by the channel Power Delay Profile, and are chosen randomly according to the channel generation procedure. Each path consists of M sub-paths. Figure 3.9 shows the angular parameters used in the model.

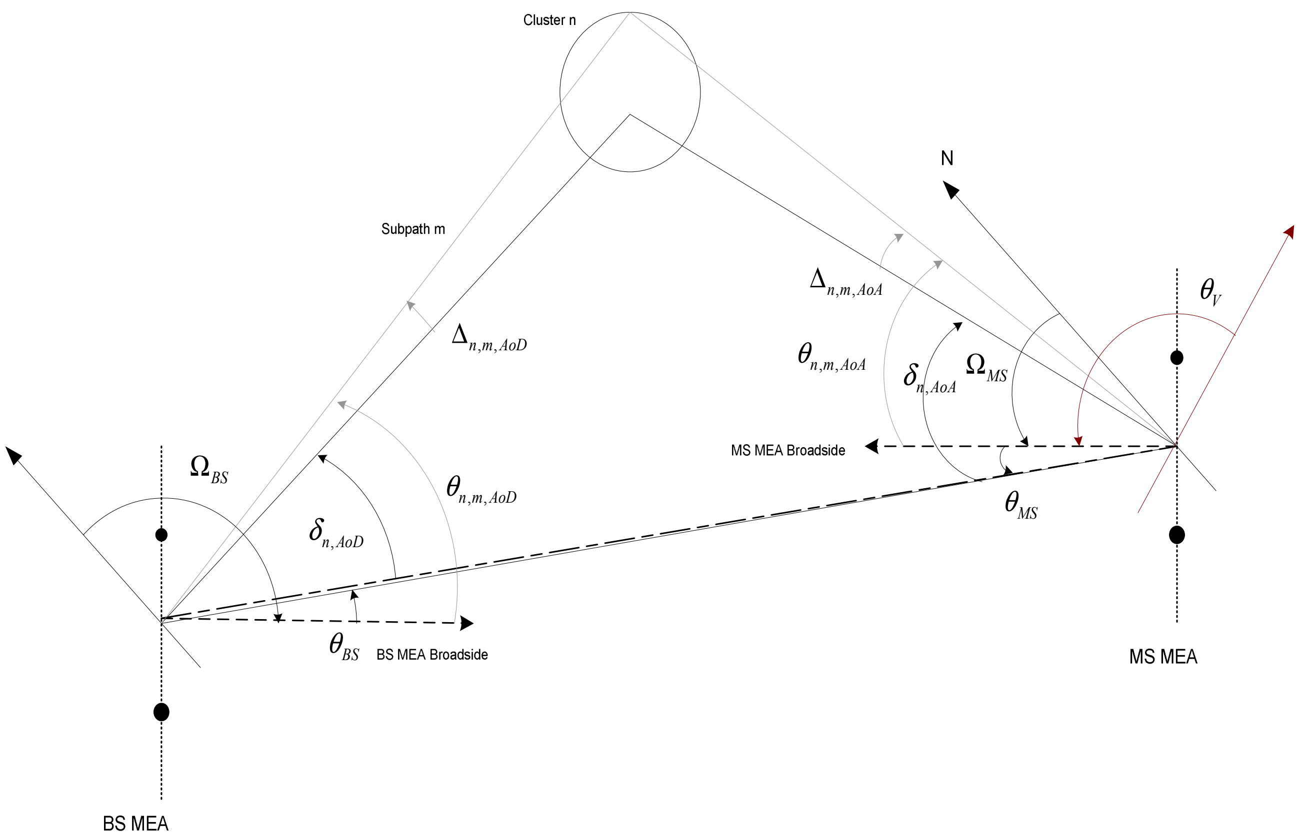

Figure 3.9: The MIMO channel model angle parameters at both BS and MS

Ω_{BS}	BS antenna array orientation, defined as the angle between the broadside of BS MEA and the absolute North (N) reference direction.
θ_{BS}	The angle between LOS direction and the broadside of BS array.
$\delta_{n,AoD}$	AoD for the ***n***th ray with respect to the LOS, where (n = 1 … N). In our case, N=6.
$\Delta_{n,m,AoD}$	Offset for the ***m***th subpath of the ***n***th ray with respect to, where (m = 1 … M). In our case, M=20.
$\theta_{n,m,AoD}$	Absolute AoD for the ***m***th subpath of the nth ray at the BS with respect to the BS broadside.
Ω_{MS}	MS antenna array orientation, defined as the angle between the broadside of the MS MEA and the absolute North reference direction.
θ_{MS}	Angle between the BS-MS LOS and the MS broadside
$\delta_{n,AoA}$	AoA of the ***n***th ray with respect to LOS
$\Delta_{n,m,AoA}$	Offset for the ***m***th subpath of the ***n***th ray with respect to $\delta_{n,AoA}$.
$\theta_{n,m,AoA}$	Absolute AoA for the ***m***th subpath of the ***n***th ray at the MS w.r.t. the MS broadside
V	MS velocity vector
θ_v	Angle of the velocity vector with respect to the MS broadside: $\theta_v = \arg(V)$

Note: The angle measured in a clockwise direction is assumed to be negative value.

3.3. Scenarios

The scenarios that we have studied cover some typical cases.

The following channel environments will be considered for system simulations:

1. **Suburban macro-cell**
   - Large cell radius (approximately 1-6 km distance BS to BS)
   - High BS antenna positions (above rooftop heights, between 10-80m)
   - Low delay and angle spreads
   - High range of mobility (0 – 250 km/h)
2. **Urban macro-cell**
   - Large cell radius (approximately 1-6 km distance BS to BS)
   - High BS antenna positions (above rooftop heights, between 10-80m)
   - Moderate (to high) delay and angle spread
   - High range of mobility (0 – 250 km/h)
3. **Urban micro-cell**
   - Small cell radius (approximately 300m – 500 m distance BS to BS)
   - BS antenna positions (at rooftop heights or lower)
   - High angle spread and moderate delay spread
   - Medium range of mobility (0 – 120 km/h)
   - The model is sensitive to antenna height and scattering environment (such as street layout, LOS)

The environment urban micro is differentiated in line-of-sight (LOS[2]) and non-LOS (NLOS) propagation.

[2] LOS: is defined as a path free of obstructions within the 1st Fresnel zone to minimize the simultaneous reception of reflected out–of-phase signals and excess losses due to signal diffraction. Although in practice it is common to tolerate obstructions in 30-40% of the 1st Fresnel zone it would still require higher base station heights at 700 MHz to achieve the same Fresnel zone clearance that can be achieved at 2500 MHz, for instance.

4. Adaptation to the DTV frequency bands

4.1. Description

At present there is huge interest in digital television (DTV) because it can deliver vast amounts of information at very low cost to the maximum number of viewers, it can now be fully integrated into completely digital transmission networks and it can be packaged as never before.

Digital television can deliver more programs than traditional analog television over any transmission medium. This is because digital information can be manipulated and treated in ways never possible with analog television.

In Europe many projects were undertaken in the early 1990s to specify a HDTV standard and with the help of the German government, a European Launching Group (ELG) was formed in 1992 that invited participation from interested organizations in Europe. With the success of ELG in 1993 approximately 84 broadcasters, standards bodies, telecommunications companies, manufacturers, and other organizations formed the Digital Video Broadcasting project (DVB) by signing a memorandum of understanding. [4.1]

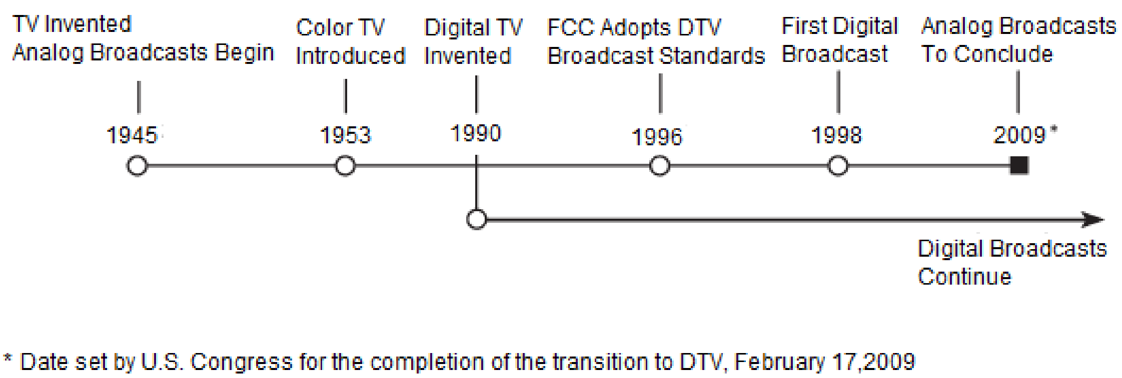

Figure 4.1: A timeline of television broadcasting [4.2]

The DVB project has specified MPEG-2 as the source encoding standard for audio, video, as well as system information, and multiplexing. The DVB project has specified Coded Orthogonal Frequency Division Multiplexing (COFDM) as the terrestrial broadcast channel modulation standard, and it's referred to as the DVB-T standard.

The greatest advantage of the digital system is the effective use of the frequency spectrum and its lower radiated power in comparison with the analogue transmission, while the covered area remains the same. Another key feature is the possibility of designing a Single Frequency Network (SFN), which means that the neighbouring broadcast stations use the same frequency and the adjacent signals don't get interfered. The diagram below shows how with overlapping signals *on the same frequency*, a DVB-T SFN allows the broadcaster to achieve universal service coverage and a better spectrum efficiency.

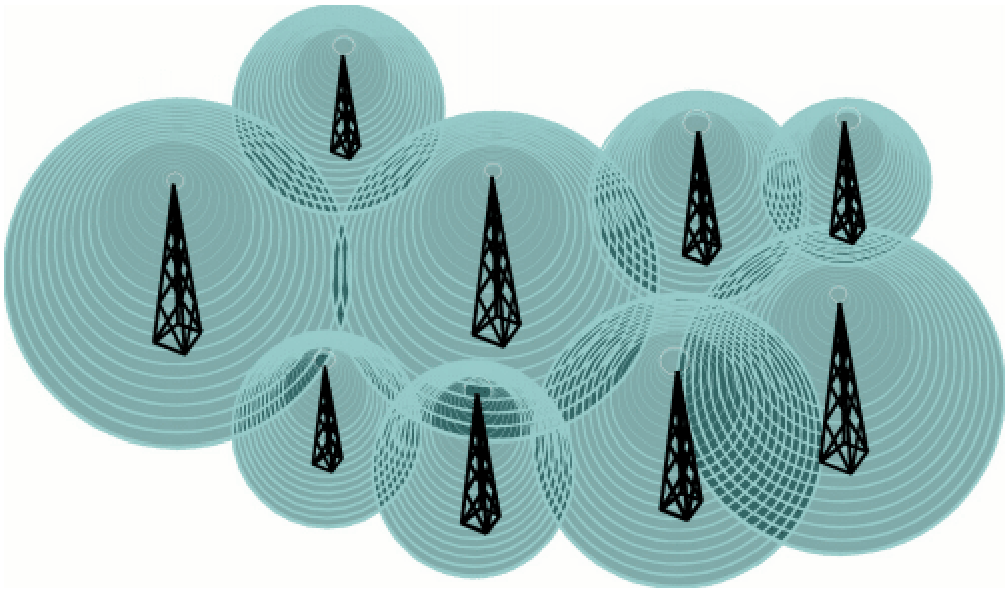

Figure 4.2: A wide-area Single Frequency Network (SFN) [4.3]

The technology of Single Frequency Network provided in DVB-T fulfill a large area with sufficient field strength for reception. Several transmitting stations at different distant locations broadcast the same TV program simultaneously with the same carrier frequency. SFN is able to establish a large area coverage in which a mobile receiver can appreciate the TV service when it is moving around. [4.3]

Thanks to COFDM, a receiver receiving two or more signals on the same frequency actually uses the "interference" to build a stronger signal. The dark areas of overlap in the diagram above thus do not indicate destructive interference but rather network gain. [4.4]

Orthogonal frequency division multiplex is a multi-carrier method with up of thousands of subcarriers, none of which interfere with each other because they are orthogonal to one another. Each of these subcarriers is vector modulated, i.e. QPSK, 16-QAM and often up to 64-QAM modulated. [4.5]

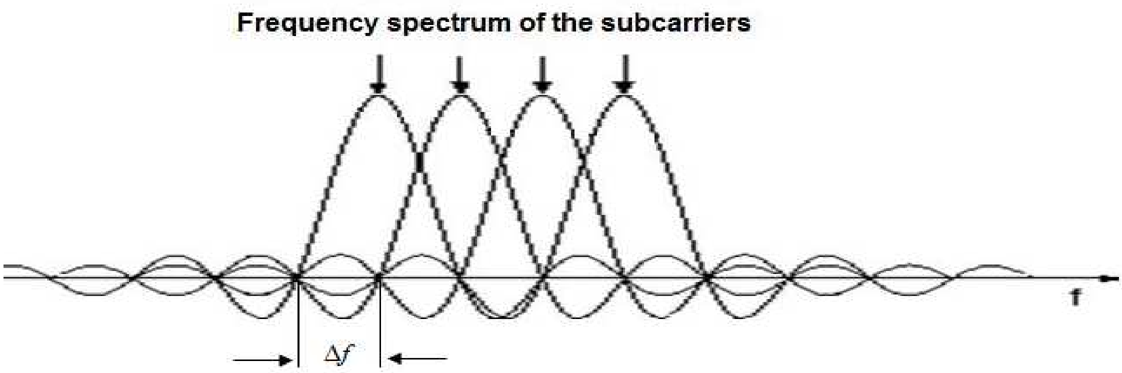

Figure 4.3: Orthogonality in OFDM

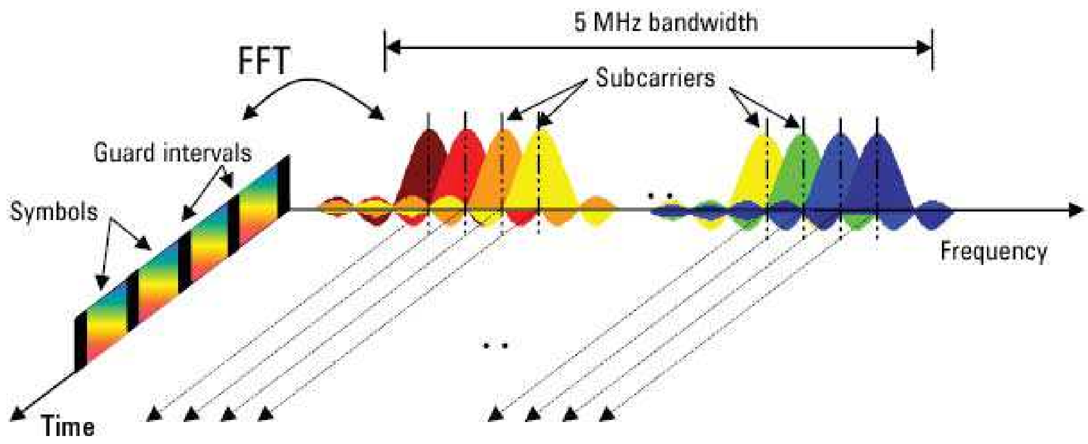

Figure 4.4: OFDM signal represented in frequency and time [4.6]

Figure 4.5 shows the probability of modulation and coding scheme usage from QPSK ½ to 64QAM ¾. It shows that the higher order modulation scheme would be more likely in the 700 MHz systems thus resulting in higher spectral efficiency. [4.7]

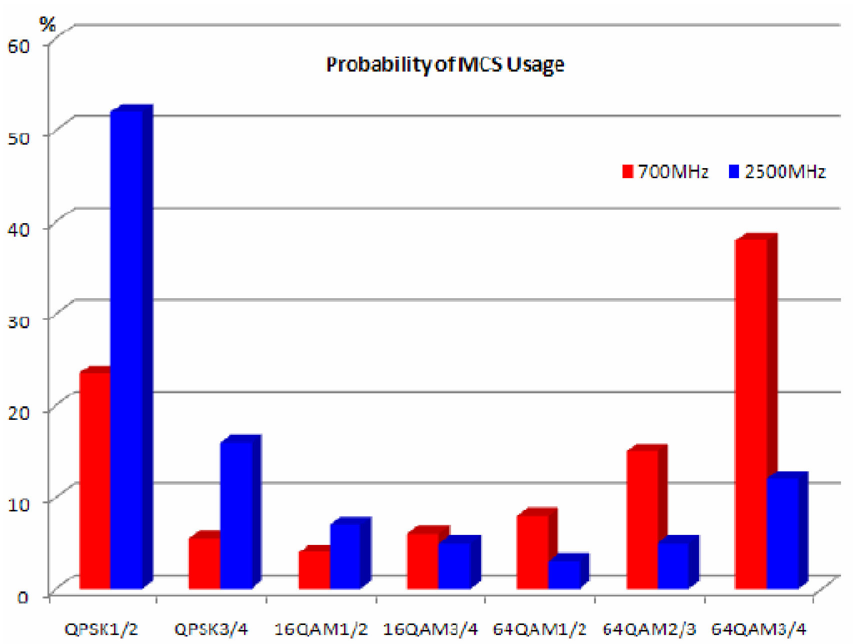

Figure 4.5: Probability of Channel Modulation Scheme

COFDM is a multi-carrier method that belongs to the most complicated transmission method of all and is in no way inferior to the code division multiple access (CDMA) methods. This complexity is due to the transmission medium which is an extremely difficult medium to deal with. [4.5]

The terrestrial transmission paths, exhibit the following characteristics features:

- Multipath reception via various echo paths caused by reflections from buildings, mountains, trees, vehicles;
- Additive white Gaussian noise (AWGN);
- Narrow-band or wide-band interference sources caused by internal combustion engines, streetcars or other radio sources;
- Doppler effect, i.e. frequency shift in mobile reception

This modulation is particularly suitable to operate on the terrestrial multipath propagation channel because of the narrow-band characteristic of each data carrier and of the presence of a "guard interval" (with duration T_g) which separates adjacent symbols and avoids inter-symbol interference (ISI) in the presence of echoes. [4.7] This type of modulation and the additional Digital Video Broadcasting-Terrestrial (DVB-T) characteristics reduce the impact of the frequency selective fading caused by multipath propagation in the terrestrial VHF/UHF channel. However, a spatial fading related to the location of the receiving antenna can still be present in the DVB-T channel [4.9]

COFDM is characterised by two operational modes, the first one with 2K carriers for conventional multi-frequency networks (MFN), the second one with 8K carriers for SFNs.

Mode	2k	8k
Number of subcarriers	2048	8192
Approx. subcarrier spacing Δf	4KHz	1KHz
Approx. symbol duration Δt=1/ Δf	250 µs	1 ms

The bandwidth of operation of DVB-T is mainly between 450 and 900 MHz, and may even work around 200 MHz, and as we can see later, this band is especially interesting due to the improved propagation conditions as compared to 2.5 and 3.5 GHz.

4.2. DVB-T System Parameters of the 8-/7-/6-MHz Channel

The DVB-T system offers a bit rate capacity ranging from 5 Mbps to 31.5 Mbps, depending on the chosen level of m-QAM modulation (m = 4, 16 or 64), the inner code rate (1/2, 2/3, 3/4, 5/6 or 7/8) and the guard interval duration (T_g / T_u =1/4, 1/8, 1/16 or 1/32; T_u = useful symbol duration = 224 µs for 2K mode and 896 µs for 8K mode). The system is optimised for 8 MHz channels (European UHF channellisation), but it can be easily adapted to 7 MHz and 6 MHz channels by adjusting the receiver sampling frequency.

Table 4.1: Main DVB-T System Parameters

Parameter		2k mode	8k mode
FFT length	N_{FFT}	2048	8192
Used carriers	N_C	1705	6817
Payload carriers	N_C	1512	6048
Subcarriers spacing	Δf_{SC}	4464 Hz	1116 Hz
Guard interv. Lengths	N_G/N_{FFT}	1/4, 1/8,1/16, 1/32	
Inner conv. Code rates	R	1/2, 2/3, 3/4,5/6,7/8	
Modulation		4-, 16-, 64-QAM	

The basic system parameter in DVB-T is the IFFT sampling frequency of the 8-MHz channel which is defined as:

- $F_{sample\ IFFT\ 8\ MHz}$ = 64/7 MHz = 9.142857 MHz

From this basic parameter, all other system parameters can be derived:

- $F_{sample\ IFFT\ 7\ MHz}$ = 64/7 MHz * 7/8 = 8 MHz
- $F_{sample\ IFFT\ 6\ MHz}$ = 64/7 MHz * 6/8 = 6.857142857 MHz

All 2048 or 8192 IFFT carriers in the 8-/7- and 6-MHz channel can be found within these IFFT bandwidths. From these bandwidth or sampling frequencies, the respective subcarrier spacing can be easily derived by dividing the bandwidth $F_{sample\ IFFT}$ by the number of subcarriers:

- $\Delta f = f_{sample\ IFFT} / N_{total\ carriers}$;
- $\Delta f_{2k} = f_{sample\ IFFT} /2048$;
- $\Delta f_{8k} = f_{sample\ IFFT} /8192$;

Table 4.2: DVB-T Subcarrier Spacing

Channel bandwidth	Δf of 2K Mode	Δf of 8K Mode
8 MHz	4.464285714 KHz	1.116071429 KHz
7 MHz	3.90625 KHz	0.9765625 KHz
6 MHz	3.348214275 KHz	0.8370535714

From the subcarrier spacing, the symbol length Δtsymbol can be determined directly. Due to the orthogonality condition, it is:

$$\Delta f_{symbol} = 1 / \Delta f$$

Therefore, the symbol lengths in the various modes and channel bandwidth in DVB-T are:

Table 4.3: DVB-T Symbol Durations

Channel bandwidth	Δt_{symbol} of 2K Mode	Δt_{symbol} of 8K Mode
8 MHz	224 μs	896 ms
7 MHz	256 μs	1.024 ms
6 MHz	298.7 μs	1.1947 ms

The DVB-T signal bandwidths are obtained from the subcarrier spacing Δf of the respective channel (8,7,6 MHz) and the number of carriers actually used in 2k and 8k mode (1705 and 6817).

$$f_{signal\ DVB-T} = Ncarriers\_used \cdot \Delta f$$

Table 4.4: DVB-T Signal Bandwidth

Channel bandwidth	$f_{signal\ DVB-T}$ of 2K Mode	$f_{signal\ DVB-T}$ of 8K Mode
8 MHz	7.612 MHz	7.608 MHz
7 MHz	6.661 MHz	6.657 MHz
6 MHz	5.709 MHz	5.706 MHz

4.3. The Transmission Path

In order to do simulations as close to the reality as possible, it is important to have a good channel model. The channel model used in this thesis has to support both an urban area and mobility of the client terminals.

In the ideal case, exactly one signal path arrives at the receiving antenna. The signal is then only attenuated to a greater or lesser extent and is merely subjected to additive white Gaussian noise (AWGN). This channel with a direct view of the transmitter is called a Gaussian channel and provides the best conditions of reception for the receiver. [4.10]

If multiple echoes are added to this direct signal path, the conditions of reception become much more difficult. This channel with a direct line of sight and a defined number of multiple echoes, which a direct line of sight and a defined number of multiple echoes, which can be simulated as a mathematical channel model, is called a Rice channel. [4.1]

If then the direct line of sight to the transmitter, i.e. the direct signal path, is also blocked, the channel is called a Rayleigh channel. This represents the worst conditions of stationary reception.

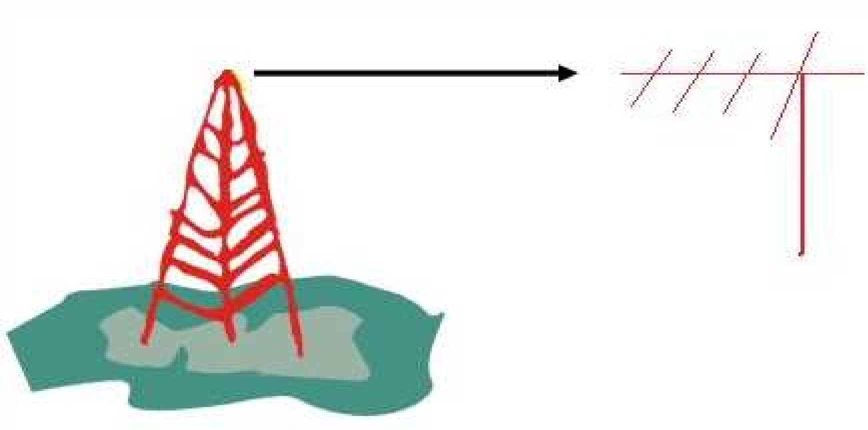

Figure 4.6: Gaussian Channel. Direct view no echoes

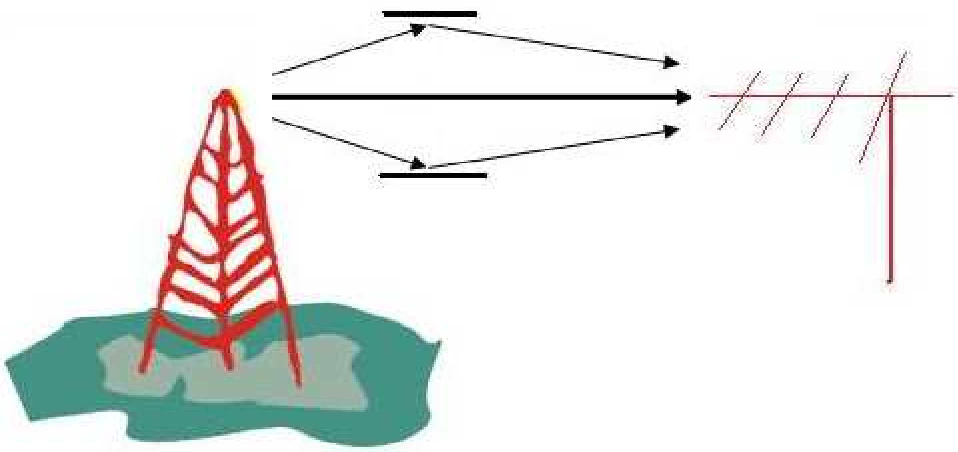

Figure 4.7: Rice Channel. Direct view and multiple echos

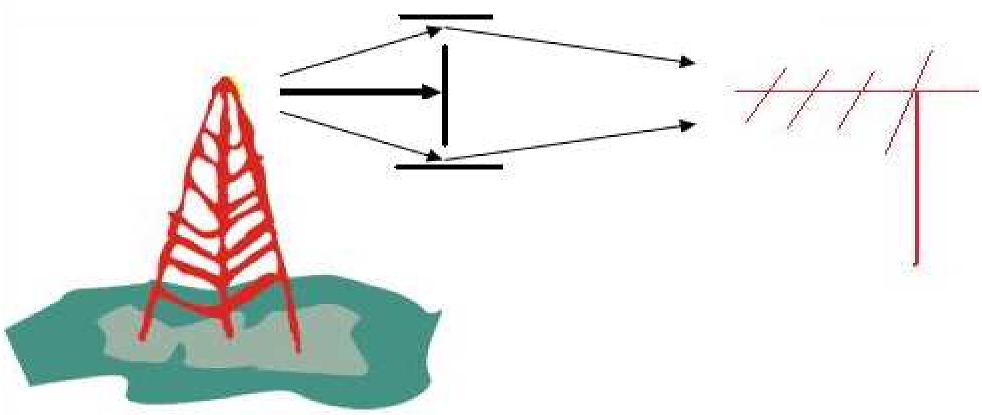

Figure 4.8: Rayleigh Channel. No direct view only multiple echos.

If, for instance, the receiver is moving at a certain speed away from the transmitter or towards the transmitter, a negative or positive frequency shift Δf will ocurr due the Doppler effect (fig.4.9). This frequency can be calculated from the speed of movement, the transmitting frequency and the velocity of light.

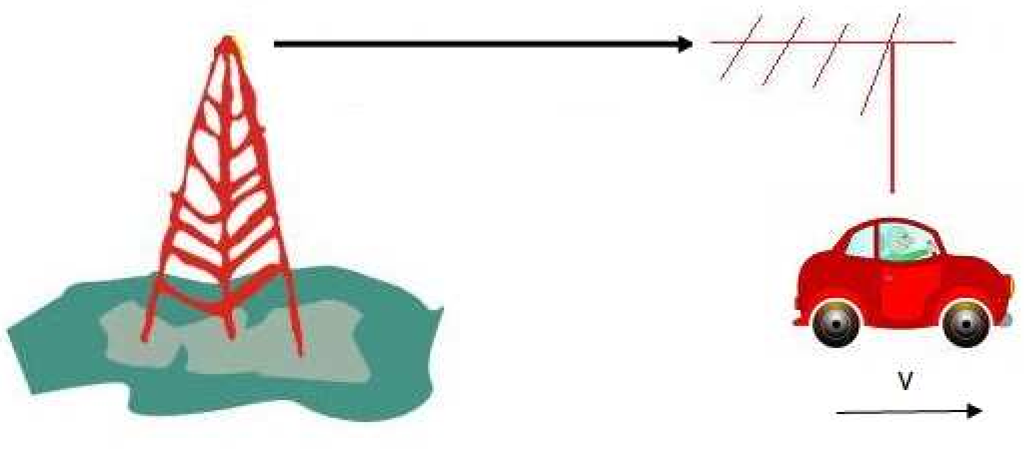

Figure 4.9: Doppler Effect

The following applies:

$$\Delta f = v \cdot \left(f/c \right) \cdot \cos(\varphi)$$

where v is the speed, f the transmitting frequency, c the velocity of light and φ the angle of incidence of the echo in relation to the direction of movement.

If, however, multiple echoes are added, the OFDM spectrum becomes smeared (Fig.4.10). This smearing is due to the fact the mobile receiver is both moving towards signal paths and moving away from other sources.

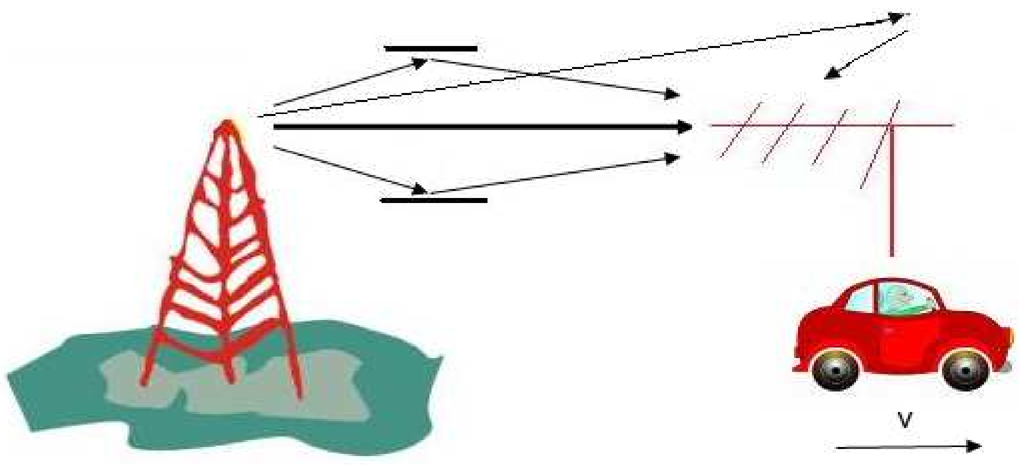

Figure 4.10: Doppler effect in Combination with Multipath Reception

More or less noise in the DVB-T channel leads to more or fewer bit errors during the reception. To determine the bit error rate in DVB-T, only the power in the actual payload carriers can be used as the signal power. In DVB-T, the difference between the overall carrier power and the power in the pure payload carriers is 0.857 dB in the 2k mode and 0.854 dB in the 8k mode but the noise bandwidth of the pure payload carrier is reduced with respect to the overall signal.

The reduce noise bandwidth of the payload carriers is:

$10\log\left(1512/1705\right) = -0.522\ dB$ in 2k mode;

$10\log\left(6048/6917\right) = -0.520\ dB$ in 8k mode;

Thus, the difference between C/N and S/N in DVB-T is:

$C/N - S/N = 0.34\ dB$ in 2k mode,

$C/N - S/N = 0.33\ dB$ in 8k mode.

Table 4.5: Minimum C/N Ratios for Non-Hierarchical Modulation

MODULATION TYPE	CODE RATE	GAUSSIAN CHANNEL	RICE CHANNEL	RAYLEIGH CHANNEL
	1/2	3.1 dB	3.6 dB	5.4 dB
	2/3	4.9 dB	5.7 dB	8.4 dB
QPSK	3/4	5.9 dB	6.8 dB	10.7 dB
	5/6	6.9 dB	8.0 dB	13.1 dB
	7/8	7.7 dB	8.7 dB	16.3 dB
	1/2	8.8 dB	9.6 dB	11.2 dB
	2/3	11.1 dB	11.6 dB	14.2 dB
16QAM	3/4	12.5 dB	13.0 dB	16.7 dB
	5/6	13.5 dB	14.4 dB	19.3 dB
	7/8	13.9 dB	15.0 dB	22.8 dB
	1/2	14.4 dB	14.7 dB	16.0 dB
	2/3	16.5 dB	17.1 dB	19.3 dB
64QAM	3/4	18.0 dB	18.6 dB	21.7 dB
	5/6	19.3 dB	20.0 dB	25.3 dB
	7/8	20.1 dB	21.0 dB	27.9 dB

Table 4.6: Minimum C/N Ratios for Hierarchical Modulation

MODULATION TYPE	CODE RATE	GAUSSIAN CHANNEL	RICE CHANNEL	RAYLEIGH CHANNEL
	1/2	6.5 dB	7.1 dB	8.7 dB
	2/3	9.0 dB	9.9 dB	11.7 dB
QPSK	3/4	10.8 dB	11.5 dB	14.5 dB
	1/2	16.3 dB	16.7 dB	18.2 dB
	2/3	18.9 dB	19.5 dB	21.7 dB
64QAM	3/4	21.0 dB	21.6 dB	24.5 dB
	5/6	21.9 dB	22.7 dB	27.3 dB
	7/8	22.9 dB	23.8 dB	29.6 dB

4.4. Channel Models

A number of channel models can be considered for 700 MHz and higher bands that include provision for mobile communication and the usage of multiple antenna concepts. Both for MIMO and Beamforming[3], a channel model that can consider the effect of direction of incoming and outgoing signals is desirable. [4.7]

With these requirements, a suitable channel model is the COST 273, Directional Channel Model, that can be considered as a parametric stochastic model. The model can be used for 13 different environments and can cover macro, micro, and pico cells.

Another channel model that is available for simulations is the COST 259 model (Wireless Flexible Personalized Communication). This channel model presents some limitations that we must consider:

- The usage for BW's higher than 6 MHz is not guaranteed to yield accurate results.
- The channel model assumes only the mobile station moving while the objects between the BS and MS are stationary.

After these considerations, we are going to base our path loss simulations on the Hata model which is compared to the COST 259 and COST 273 models in the following table.

Table 4.7: Path Loss Models

	COST-259	COST-273	Hata	COST-231
Frequency	**>500 MHz**	**X, >2GHz**	**<1500 MHz**	**>1500 MHz**
Broadband	X	√	X	X
Directional	√	√	X	X
MIMO	√	√	X	X
Beamforming	√	√	X	X
Mobility	√	√	X	X
Multipath	√	√	√	√
Urban Models	√	√	√	√
Suburban/Rural Models	√	√	√	√
Building Penetration Loss	+	+	+	+
Vehicle Penetration Loss	+	+	+	+
Channel Types	13	13	1	1

+: Can be added

There are different techniques that are propose for maximizing spectral efficiency and one of them is to avoid the use of multiple antenna techniques –MIMO and AAS (Adaptative Antenna

[3] Beamforming: is a signal processing technique used in sensor arrays for directional signal transmission or reception.

Systems)- due to those techniques exhibits poor performance in 700 MHz because of the LOS or NLOS, typical to such deployments. [4.12] On top of that, we have given priority to another factors and not just to the spectral efficiency, even if in many situations the antenna arrays requires for MIMO in the 700 MHz band are too large to be a realistic option.

The table shown below contains some relevant parameters for determining the path loss differences between our frequency of interest (700 MHz) and 2,5GHz assuming a (1x2) SIMO antenna configuration at both the Base Station and the Mobile Station.

Table 4.8: Parameters for Path Loss Comparision

Parameter	700 MHz	2.5 GHz
Propagation Model	Hata	COST 231
Region	Suburban	
BS Antenna Height	32 meters	
Height Above Average Building Height	10 meters	
Mobile Terminal Antenna Height	1.5 meters	
Path Loss at 1 Km	113.9 dB	140.6 dB

Table 4.9 provides a summary of the parameters used to estimate the range and coverage for the frequency bands of interest and reflects the fact that the use of antenna arrays would not be readily adaptable to the 700 MHz band due to the size and antenna spacing requirements, but will be quite common in the higher bands.

Table 4.9: Parameters for Range Estimation

Parameter	700 MHz	2.5 GHz
Duplex	TDD	
Channel BW	10 MHz	
BS Antennas	Tx=1, Rx=2 (1x2 SIMO)	Beamforming Array
BS Antenna Gain	15 dBi	21 dBi
BS Tx Power (at antenna)	10 Watts (+40 dBm)	
BS Antenna Height	32 meters	
MS Antennas	Tx=1, Rx=2 (1x2 SIMO)	
MS Antenna Gain	-1 dBi	
MS Tx Power	200 mW (+23 dBm)	
MS Antenna Height	1.5 meters	
BS Noise Figure	4 dB	
MS Noise Figure	7 dB	
Building Penetration Loss	8 dB	10 dB
Propagation Model	Hata	COST 231

The measured patterns show in the Fig. 4.11 is carried out at the resonant frequency of 700MHz. [4.13]

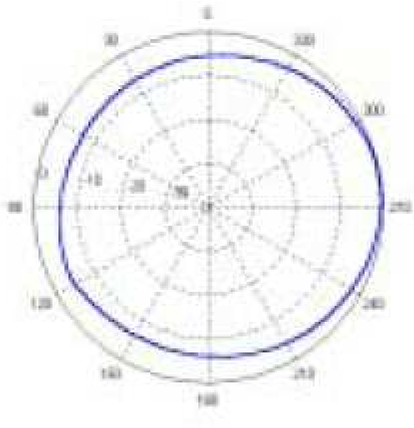

Figure 4.11: The measured radiation pattern of the compact DTV receiving antenna

4.5. Antenna Requirements

For MIMO and receive/transmit diversity systems the level of correlation between antenna elements has a direct impact on the resulting performance of these multi-antenna techniques. Two means of achieving low correlation are common, namely polarization and space diversity.

- *Polarization diversity* provides very low levels of correlation and is realized by using any two orthogonal polarizations, e.g. vertical and horizontal or ±45°. Such techniques can be used at both MS and BS. This approach will be equally effective at both 700 MHz and higher frequencies. For 2-branch MIMO systems, which is our case, using polarization diversity the BS antenna sizes are therefore likely to be the same as a conventional dual polar sector antenna e.g. ~300 mm wide at 700 MHz or ~150 mm wide at 2.5 GHz.

The difference between the frequency bands is more pronounced however when both space and polarization diversity is implemented. Here the antenna spacing is dictated by the level of decorrelation required for effective MIMO operation which in turn is determined by the degree of scattering present in the environment. In an urban channel, for instance, where there is a high degree of scattering about the mobile a relatively modest antenna separation might be adequate - perhaps 1 to 2 λ. In a suburban channel however the angle spread subtended at the BS by the multipath is much reduced and far greater antenna separation will be required to achieve the same level of de-correlation, perhaps 5 λ, while still greater separations would be required in rural deployments. The necessary physical spacing is then determined by the wavelength, with those at 700 MHz being roughly 3.5 times as great as those required at 2500 MHz. Similar correlation requirements exist at the mobile station (MS) where physical space is likely to be limited regardless of the frequency.

The practicality of such spatial techniques at 700 MHz will then depend on the details of the deployments involved and the performance benefits offered.

- *Beamforming schemes* can be realized using a variety of antenna configurations including the combinations of the polarization and space diverse arrangements discussed above.

Here again the antenna width will be dictated by the wavelength with a 4 column antenna being approximately 900 mm wide at 700 MHz but only 250 mm wide at 2500 MHz.

4.6. Per-Path Doppler Spectrum

As we have commented before, multiple independent frequency offsets exist within the received signal due to the variable Doppler shift that occurs with multi-path propagation and a moving MS. This is known as Doppler Spread since only the main path offset can be tracked. The actual impacts of the other paths’ offsets depend on the relative frequency offsets and their relative powers. Echoes moving towards the receiver will shift the spectrum into a different direction from moving away from the receiver and, as a result, the signal noise.

4.7. Scenarios

- **Indoor office**

Although we haven’t considered this scenario in our implementation (next chapter) it’s important to have some knowledge about this specific environment.

Since the introduction of European Digital TV transmission standard – DVB-T COFDM, the indoor reception of DTV has become a new promising broadcasting service. It means, the viewer does not need a traditional directional antenna, for example, a Yagi-Uda antenna, which is usually installed on the roof of a house, instead, the viewer may only need a compact antenna used in a room for receiving the broadcasting wave of digital TV.

When the broadcasting waves come into a room, it results a very complex electromagnetic environment, because the reflection, diffraction and interference do happen naturally and there is no LOS (Line-Of-Sight) reception at all. The room is to be fulfilled with multi-path signals.

Because of the characteristics of OFDM modulation plus the guard interval technique, it is suggested to use omni-directional instead of directional antennas for DVB-T reception inside a room.

As for the statistical model for the radio channel of indoor reception, the lognormal is usually referred. The expression for lognormal model is,

$$P(R) = \frac{1}{\sqrt{2\pi\sigma_R}} \exp\left[-\frac{1}{2}\left(\frac{R - M_R}{\sigma_R} \right)^2 \right]$$

in which, M_R and σ_R are its mean value and standard deviation, respectively.

- **Typical Urban Reception**

The most suitable version of the Hata model would be the path loss model in open urban areas given by:

Parameter	Comment
L_{50}	50% value (median) propagation path loss (urban)
f_c	Frequency from 150 MHz-1.5 GHz
h_{te}, h_{re}	Base Station and Mobile antenna height
$\alpha(h_{re})$	Correction factor for h_{re} , affected by coverage area
d	Tx-Rx separation

$$L_{50}(urban)(dB) = A + B\log_{10} d$$

$$A = 69.55 + 26.16\log_{10}(f_c) - 13.82\log_{10}(h_{te}) - \alpha(h_{re})$$

- Represents fixed loss – approximately 2.6 power law dependence on f_c
- Dependence on antenna heights is proportional to $h_{re}^{1.382}$

$$B = 44.9 - 6.55\log_{10}(h_{te})$$

- Represents path loss exponent, worst case ≈ 4.5

$$L_{50}(urban)(dB) = 69.55 + 26.16\log_{10}(f_c) - 13.82\log_{10}(h_{te}) - \alpha(h_{re}) + (44.9 - 6.55h_{te})\log_{10} d$$

- **Typical Rural Area Reception**

In this environment, the correction factor is given by the expression:

$$a(h_r) = (1.1\log_{10} f_c - 0.7)h_r - (1.56\log_{10} f_c - 0.8)$$

- **Suburban**

The Hata Suburban model applies:

$$L_B = L_{50}(Urb) - 2(\log_{10}\frac{f_c}{28})^2 - 5.4$$

A well-known model for the attenuation due to foliage is the COST-235 model. However, these models are only valid in order to calculate local path loss.

$$L_B(dB) = 26.6 f_c^{-0.2} d^{0.5}$$

$$L_B(dB) = 15.6 f_c^{-0.009} d^{0.26}$$

Where f_c is in MHz and d in metres.

5. Implementations

To implement the channel model we have taken the Multiple-Input Multiple-Output (MIMO) radio link parameters (so we have used 2 elements in the BS array and 2 elements in the MS), model configuration parameters, and antenna parameters as inputs, and the MIMO channel matrices as output. Channel matrices can be generated for multiple BS-MS links with one function call. The output is a multi-dimensional array which contains the channel impulse responses for a pre-defined number of radio links.

5.1. Description of files included in the package

Table 5.1 lists the MATLAB and C-files included in the package. A short description of each file is also given.

Table 5.1: MATLAB and ANSI-C files included in the distributed package [5.1]

FILENAME	DESCRIPTION
scm.m	3GPP Spatial Channel Model (3GPP TR 25.996). This the main function called by the user to generate channel matrices. It calls the auxiliary functions that generate bulk parameters, do antenna pattern interpolation and compute the actual channel matrices.
scmparset.m	Model parameter configuration for SCM. Helper function for setting the default parameter for the first input struct.
linkparset.m	Link parameter configuration for SCM. Helper function for setting the default parameter for the second input struct.
antparset.m	Antenna parameter configuration for SCM. Helper function for setting the default parameter for the third input struct.
pathloss.m	Pathloss models for 700MHz.
interp_gain.m	Antenna field pattern interpolation (requires GSL). Function for antenna pattern interpolation using MATLAB's interpolating functions
scm_core.m	Channel coefficient computation for a geometric channel model. Function for computing "the big for loop" that generates the channel matrices
generate_bulk_par.m	Function for generating the random "bulk" parameters for macro and micro cells
cas.m	A utility function for computing the circular angle spread. This function is not necessary for channel matrix generation.
ds.m	A utility function for computing RMS delay spread. This function is not necessary for channel matrix generation.
dipole.m	A utility function that generates the pattern of a slanted dipole for vertical and horizontal polarizations. This is useful for creating some simple MIMO antenna configurations.

5.2. Model Input/Output Interface

The full syntax for the SCM function is ([.] indicates optional arguments):

[CHAN, [DELAYS], [FULLOUTPUT]] = SCM(SCMPAR, LINKPAR, ANTPAR, [INITVALUES]).

• All input arguments are MATLAB structs. The first three input arguments are mandatory. A helper function will be supplied so that their default values can be set easily.

• The fourth input argument is optional. When given, SCM does not generate the channel parameters randomly, but uses the user-supplied initial channel values.

• The first output argument is a 5D-array containing the MIMO channel matrices for all links over a specified number of time samples.

• The second output argument includes multipath delays for all links. The path delays are given in seconds.

• The third output argument is a MATLAB struct containing the randomly generated link parameters and the final phases of the complex sinusoids. This MATLAB struct can be used as INITVALUES in subsequent function calls.

5.2.1. Input Parameters

Table 5.2: [5.1] General channel model parameters. Common for all links, MATLAB struct SCMPAR

PARAMETER NAME	DEFINITION	DEFAULT VALUE	UNIT
NumBsElements	The number of elements in the BS array. This parameter is ignored if antenna patterns are defined in the input struct ANTPAR. In this case the number of BS elements is extracted from the antenna definition.	2	
NumMsElements	The number of elements in the MS array. This parameter is ignored if antenna patterns are defined in the input struct ANTPAR. In this case the number of BS elements is extracted from the antenna definition.	2	
Scenario	Selected SCM channel,scenario (‘suburban_macro’, ‘urban_macro’ or ‘urban_micro’)	’urban_micro’	-
SampleDensity	Time sampling interval of the channel. A value greater than one should be selected if Doppler analysis is to be done.	2	$\frac{samples}{\lambda/2}$
NumTimeSamples	Number of channel samples (impulse response matrices) to generate per link.	100	-
UniformTimeSampling	If this parameter has value ‘yes’, the time sampling interval of the channel for each user will be equal. Sampling interval will be calculated from the SampleDensity and the highest velocity found in the input parameter vector MsVelocity. If this parameter	‘no’	

	has value ’no’, then the time sampling interval for each link will be different, if MSs have different speeds (see linkpar.MsVelocity). Setting this parameters ‘yes’ may be useful in some system-level simulations where all simulated links need to be sampled at equal time intervals, regardless of MS speeds		
BsUrbanMacroAS	BS mean angle spread for urban macro environment. Possible values are ‘eight’ and ‘fifteen’ degrees. This variable is ignored if ‘Scenario’ is not ‘urban_macro’.	‘eight’	-
NumPaths	Number of paths (channel taps). Path delays are drawn from the delay distribution specified in [3GPP03] regardless of the number of paths set.	6	-
NumSubPathsPerPath	Number of sub-paths per path. The only value supported in the SCM specification is 20 subpaths, see [3GPP03, Table 5.2].	20	-
CenterFrequency	Carrier center frequency. Affects path loss and time sampling interval.	7E8	Hz
ScmOptions	SCM options ('none','polarized','los','urban_canyon '). The options are mutually exclusive. i.e. one cannot, for instance, choose 'polarized' and 'los' simultaneously.	‘none’	-
DelaySamplingInterval	Delay sampling interval (delay resolution). $Tc=1/F_{sample\ IFFT}=1/6.85714$ MHz	1.4583e-7	sec
XpdIndependentPower	With this set to ‘yes’ the power of the elements of the channel matrix (without pathloss) is normalized to a constant, that does not depend on the XPD ratios.	‘no’	-
PathLossModelUsed	Path loss included in the channel matrices yes/no (if ‘no’, PL is calculated and returned in the third output argument, but not multiplied with the channel matrices)	‘no’	-
ShadowingModelUsed	Shadow fading included in the channel matrices yes/no (if ‘no’ shadow fading is still computed and returned in the third output argument, but not multiplied with the channel matrices). Note that if both path loss and shadowing are switched off the average power of the channel matrix elements will be one (with azimuthally uniform unit gain antennas).	‘no’	-
PathLossModel	The name of the path loss function. Function ‘pathloss’ implements the default SCM path loss model. If the	‘pathloss’	-

	default is used, center frequency is taken from the parameter CenterFrequency. One can define his/her own path loss function. For syntax, see PATHLOSS.		
AnsiC_core	Use optimized computation yes/no. With ‘yes’ faster C-function is used instead of m-function. Note the C-function SCM_MEX_CORE.C must be compiled before usage. For more information, see SCM_MEX_CORE.M.	‘no’	-
LookUpTable	If optimized computation is used, complex exponentials are taken from a look-up table to speed up computation or calculated explicitly. This parameter defines the lookup table size. Value 0 indicates that lookup table is not used, value –1 uses the default table size 2^{14} =16384. The size of the table must be a power-of-two. If AnsiC_core = ‘no’ this parameter is ignored.	0	integer
RandomSeed	Random seed for fully repeatable channel generation (if empty, seed is generated randomly). Note that even if RandomSeed is fixed, two channel realizations may still not be the same due to potential differences between random number generators in different MATLAB versions. Note also that one must also use the same link and antenna parameters.	<empty>	integer

When scmpar.ScmOptions is 'none' or **'urban_canyon'**:

- delays: path delays in seconds [KxN]
- path_powers: relative path powers [KxN]
- aods: angles of departure in degrees over (-180,180) [KxNxM]
- aoas: angles of arrival in degrees over (-180,180) [KxNxM]
- subpath_phases: final phases for subpaths in degrees over (0,360) [KxNxM]
- path_losses: path losses in linear scale [Kx1]
- shadow_fading: shadow fading losses in linear scale [Kx1]
- delta_t: time sampling intervals for all links [Kx1]

In addition, when scmpar.ScmOptions is **'los'** (in addition to the above):

- K_factors: K factors for all links [Kx1]
- Phi_LOS: final phases for LOS paths in degrees over (-180,180) [Kx1]

When scmpar.ScmOptions is **'polarized'** (in addition to scmpar.ScmOptions='none'):

- subpath_phases: final phases for subpaths in degrees over (0,360) [Kx4xNxM], where the second dimension are the [VV VH HV HH] components (iid).
- xpd: cross-polarization ratios in linear scale [Kx2xN] where the (:,1,:)th dimension is the V-to-H power coupling, and (:,2,:)th dimension is the H-to-V power coupling.

Table 5.3: Link-dependent parameters, MATLAB struct LINKPAR [5.1]

PARAMETER NAME	DEFINITION	DEFAULT VALUE	UNIT
MsBsDistance	Distance between BS and MS	Users are approximately uniformly distributed in a circular cell over distances of [35,500] meters	m
ThetaBs	θ_{BS} (see Figure 3.9)	U(-180,180)	deg
ThetaMs	θ_{MS} (see Figure 3.9)	U(-180,180)	deg
OmegaBs	Ω_{MS} (see Figure 3.9), this parameter is not currently used.	NaN	deg
OmegaMs	Ω_{MS} (see Figure 3.9), this parameter is not currently used. OmegaBs and OmegaMs define the orientation of antenna broadside with % respect to north. This parameter is currently redundant.	NaN	deg
MsVelocity	MS velocity.10 meters per second	10*ONES(1,K)	m/s
MsDirection	θ_v (see Figure 3.9)	U(-180,180)	deg
MsHeight	Height of MS. Possibly needed for path loss computation. 1.5 meters	1.5*ONES(1,K)	m
BsHeight	Height of BS. Possibly needed for path loss computation. 32 meters	32*ONES(1,K)	m
MsNumber	Index number (positive integer) of the MS for each simulated link. This parameter is needed for generating shadow fading values with inter-site correlation. Shadow fading is correlated for links between a single MS and multiple BSs (inter-site correlation). There is no correlation between shadow fading between different MSs.	[1:K]	-

Table 5.4: Antenna parameters, MATLAB struct ANTPAR [5.1]

PARAMETER NAME	DEFINITION	DEFAULT VALUE	UNIT
BsGainPattern	BS antenna field pattern values in a 4D array. The dimensions are [ELNUM POL EL AZ] = SIZE(BsGainPattern), where ELNUM is the number of antenna elements in the array. The elements may be dual-polarized. POL – polarization. The first dimension is vertical polarization, the second is horizontal. If the polarization option is not used, vertical polarization is assumed (if both are given). EL – elevation. This value is ignored. Only the first element of this dimension is used. AZ – complex-valued field pattern in the azimuth dimension given at azimuth angles defined in BsGainAnglesAz. If NUMEL(BsGainPattern)=1, all elements are assumed to have uniform gain defined by the value of BsGainPattern over the full azimuth angle, and the number of BS antenna elements is defined by the value of BsGainPattern over the full azimuth angle, and the number of BS antenna elements is defined by NumBsElements. This speeds up computation since field pattern interpolation is not required.	1	
BsGainAnglesAz	Vector containing the azimuth angles for the BS antenna field pattern values. These values are assumed to be the same for both polarizations. This value is given in degrees over the range (-180,180) degrees. If NUMEL(BsGainPattern)=1, this variable is ignored.	linspace(-180,176,90)	deg
BsGainAnglesEl	Vector of elevation angles for definition of BS antenna gain values. This parameter is for future needs only; its value is ignored in this implementation (SCM does not support elevation).	-	-
BsElementPosition	Element positions for BS linear array in wavelengths. This parameter can be either scalar or vector. If scalar, uniform spacing equal to the scalar is applied. If vector, it defines antenna element positions on a line. Note that one can place two elements in the same position and, by defining the antenna patterns	0.5	λ

	properly, create dual-polarized arrays.		
MsGainPattern	MS antenna field pattern values in a 4D array. The dimensions are [ELNUM POL EL AZ] = SIZE(MsGainPattern), where ELNUM – the number of antenna elements in the array. The elements may be dual-polarized. POL – polarization. The first dimension is vertical polarization, the second is horizontal. If the polarization option is not used, vertical polarization is assumed (if both are given). EL – elevation. This value is ignored. Only the first element of this dimension is used. AZ – complex-valued field pattern in the azimuth dimension given at azimuth angles defined in MsGainAnglesAz. If NUMEL(MsGainPattern)=1, all elements are assumed to have uniform gain defined by the value of MsGainPattern over the full azimuth angle, and the number of MS antenna elements is defined by scmpar.NumMsElements. This speeds up computation since field pattern interpolation is not needed.	1	compl ex
MsGainAnglesAz	Vector containing the azimuth angles for the MS antenna field pattern values. These values are assumed to be the same for both polarizations. This value is given in degrees over the range (-180,180) degrees. If NUMEL(BsGainPattern)=1, this variable is ignored.	linspace(-180,176,90)	deg
MsGainAnglesEl	Vector of elevation angles for definition of MS antenna gain values. This parameter is for future needs only; its value is ignored in this implementation (SCM does not support elevation).	-	-
MsElementPosition	Element positions for MS linear array in wavelengths. This parameter can be either scalar or vector. If scalar, uniform spacing is applied. If vector, it defines antenna element positions on a line. Note that one can place two elements in the same position and, by defining the antenna patterns properly, create dual-polarized arrays.	0.5	λ
InterpFunction	The name of the interpolating function. One can replace this with his own function. For syntax, see interp_gain.m, which is the default function. For faster computation, see interp_gain_c.m	'interp_gain'	
InterpMethod	The interpolation method used by the interpolating function. Available methods depend on the function. The default function is based on MATLAB's interp1.m function and supports e.g. 'linear' and	'cubic'	

	‘cubic’ (default) methods. Note that some methods, such as ‘linear’, cannot extrapolate values falling outside the field pattern definition.		

Table 5.5: Initial values, fourth optional input argument, a MATLAB struct INITVALUES [5.1]

PARAMETER NAME	DEFINITION	UNIT
delays	A K x N matrix of path delays.	Sec
path_powers	A K x N array of powers of paths.	linear scale
aods	A K x N x M array of angles of departure of subpaths	Degrees
aoas	A K x N x M array of angles of arrival of subpaths	Degrees
subpath_phases	A K x N x M array of initial subpath phases. When polarization option is used, this is a K x P x N x M array, where P=4. In this case the second dimension includes the phases for [VV VH HV HH] polarized components.	Degrees
path_losses	A K x 1 vector of path losses	linear scale
shadow_fading	A K x 1 vector of shadow fading losses	linear scale
xpd	A K x 2 x N array of cross-polarization coupling power ratios. The second dimension is the [V-to-H H-to-V] coupling ratios. This is needed only when ‘polarized’ option is used.	linear scale

5.2.2. Output Parameters

There are three output arguments: CHAN, DELAYS, FULLOUTPUT. The last two are optional output parameters.

K denotes the number of links,
N is the number of paths,
T the number of time samples,
U the number of receiver elements
S denotes the number of transmitter elements.

Table 5.6: Output parameter of the SCM function [5.1]

PARAMETER NAME	DEFINITION	UNIT
CHAN	A 5D-array with dimensions U x S x N x T x K	
DELAYS	A K x N vector of path delay values. Note that delays are, for compatibility with the INITVALUES, also included in	sec

	FULLOUTPUT.	
FULLOUTPUT	A MATLAB struct with the following elements:	
delays	A K x N matrix of path delays. This is identical to the second output argument.	sec
path_powers	A K x N array of powers of paths.	linear scale
aods	A K x N x M array of subpath angles of departure	degrees
aoas	A K x N x M array of subpath angles of arrival	degrees
subpath_phases	A K x N x M array giving the final phases of all subpaths. When polarization option is used, a K x P x N x M array, where P=4. In this case the second dimension includes the phases for [VV VH HV HH] polarized components.	degrees
path_losses	A K x 1 vector	linear scale
shadow_fading	A K x 1 vector	linear scale
delta_t	A K x 1 vector defining time sampling interval for all links.	sec
xpd	A K x 2 x N array of cross-polarization coupling power ratios. The second dimension is the [V-to-H H-to-V] coupling ratios.	linear scale

5.3. Simulations

```
%Matrix generation for 1 MS-BS link
%Setting and modifying default input parameters
>> scmpar=scmparset;
>> linkpar=linkparset;
>> antpar=antparset;
>> scmpar.NumTimeSamples=100; % 10 time samples per link
>> scmpar.ScmOptions='urban_canyon' % urban canyon option
>> scmpar.Scenario=’urban_micro’; %urban micro scenario
>> [H1 delays out]= scm(scmpar, linkpar,antpar);
%using final conditions as initial conditions in next function call
>> [H2 delays out]= scm(scmpar, linkpar,antpar,out);
```

>> scmpar=scmparset

scmpar =

NumBsElements: 2

NumMsElements: 2

Scenario: 'urban_micro'

SampleDensity: 2

NumTimeSamples: 100

UniformTimeSampling: 'no'

BsUrbanMacroAS: 'eight'

NumPaths: 6

NumSubPathsPerPath: 20

CenterFrequency: 700000000

ScmOptions: 'none'

DelaySamplingInterval: 1.6276e-008

XpdIndependentPower: 'no'

PathLossModelUsed: 'no'

ShadowingModelUsed: 'no'

PathLossModel: 'pathloss'

AnsiC_core: 'no'

LookUpTable: 0

RandomSeed: []

>> linkpar=linkparset

linkpar =

MsBsDistance: 466.8793

ThetaBs: -20.8655

ThetaMs: 28.6995

OmegaBs: NaN

OmegaMs: NaN

MsVelocity: 10

MsDirection: 83.1876

MsHeight: 1.5000

BsHeight: 32

MsNumber: 1

>> antpar=antparset

antpar =

BsGainPattern: 1

BsGainAnglesAz: [1x90 double]

BSGainAnglesEl: 0

BsElementPosition: 0.5000

MsGainPattern: 1

MsGainAnglesAz: [1x90 double]

MsGainAnglesEl: 0

MsElementPosition: 0.5000

InterpFunction: 'interp_gain'

InterpMethod: 'cubic'

bulkpar =

delays: [0 1.1393e-007 1.4648e-007 1.4648e-007 2.7669e-007 8.6263e-007] path_powers: [0.2581 0.0802 0.0911 0.1995 0.3498 0.0213]

aods: [1x6x20 double]

aoas: [1x6x20 double]

subpath_phases: [1x6x20 double]

path_losses: 3.6794e-019

shadow_fading: 0.6320

delta_t: 0.0107

We have proceeded in the same way for the rest of the cases and the results we have obtained are shown in the next table:

Table 5.7: Table of results

SCM OPTIONS	SCENARIO	RESULTS: Field	RESULTS: Value
None	**Urban micro**	**delays**	[0 3.9062e-007 3.9062e-007 6.6732e-007 7.8125e-007 1.123e-006]
		path_powers	[0.38312 0.19013 0.08349 0.10853 0.20184 0.032902]
		aods	*<1x6x20 double>*
		aoas	*<1x6x20 double>*
		subpath_phases	*<1x6x20 double>*
		path_losses	1.9447e-019
		shadow_fading	1.042
		delta_t	0.010707
	Urban macro	**delays**	[0 1.6276e-007 3.9062e-007 1.237e-006 1.3021e-006 1.5788e-006]
		path_powers	[0.59655 0.057372 0.12572 0.15479 0.044063 0.021508]
		aods	*<1x6x20 double>*
		aoas	*<1x6x20 double>*
		subpath_phases	*<1x6x20 double>*
		path_losses	2.1917e-022
		shadow_fading	6.1853
		delta_t	0.010707
	Suburban	**delays**	[0 1.6276e-008 3.2552e-008 8.138e-008 1.7904e-007 3.418e-007]
		path_powers	[0.4183 0.19064 0.13991 0.10894 0.047241 0.094959]
		aods	*<1x6x20 double>*
		aoas	*<1x6x20 double>*
		subpath_phases	*<1x6x20 double>*
		path_losses	5.9871e-020
		shadow_fading	0.066525
		delta_t	0.010707

SCM option	Scenario	Field	Value
Los	Urban Micro	delays	[0 3.2552e-008 4.8828e-008 3.2552e-007 8.3008e-007 1.0742e-006]
		path_powers	[0.18666 0.2877 0.28138 0.19974 0.031278 0.013241]
		aods	*<1x6x20 double>*
		aoas	*<1x6x20 double>*
		subpath_phases	*<1x6x20 double>*
		K_factors	0
		Phi_LOS	44.384
		path_losses	4.1758e-019
		shadow_fading	1.9951
		delta_t	0.010707
Urban Canyon	Urban Micro	delays	[0 1.3021e-007 5.8594e-007 6.8359e-007 7.9752e-007 8.789e-007]
		path_powers	[0.21663 0.27598 0.16264 0.082702 0.23835 0.023691]
		aods	*<1x6x20 double>*
		aoas	*<1x6x20 double>*
		subpath_phases	*<1x6x20 double>*
		path_losses	2.8785e-019
		shadow_fading	0.32607
		delta_t	0.010707
	Urban Macro	delays	[0 3.2552e-008 2.1159e-007 2.2786e-007 3.0924e-007 7.6497e-007]
		path_powers	[0.2209 0.22542 0.042582 0.12043 0.2433 0.14738]
		aods	*<1x6x20 double>*
		aoas	*<1x6x20 double>*
		subpath_phases	*<1x6x20 double>*
		path_losses	3.0624e-022
		shadow_fading	3.0188
		delta_t	0.010707

The most important values from the table shown above are the delays and path powers; with those values we can get the RMS delay spread:

Table 5.8: RMS Delay Spread

SCM OPTIONS	SCENARIO	RMS Delay Spread [μs]
None	**Urban Micro**	0.53497
	Urban Macro	1.1127
	Suburban	0.58643
Los	**Urban Micro**	0.33100
Urban Canyon	**Urban Micro**	0.26067
	Urban Macro	0.59005

- **RMS delay Spread**

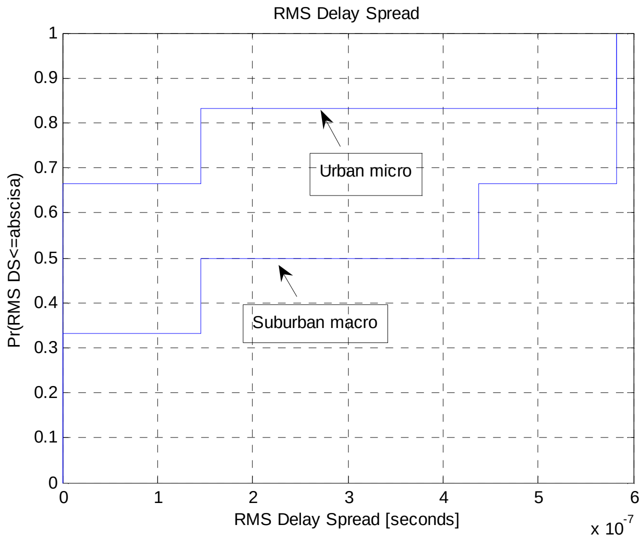

Figure 5.1: RMS delay spread

- **BS Angle Spread**

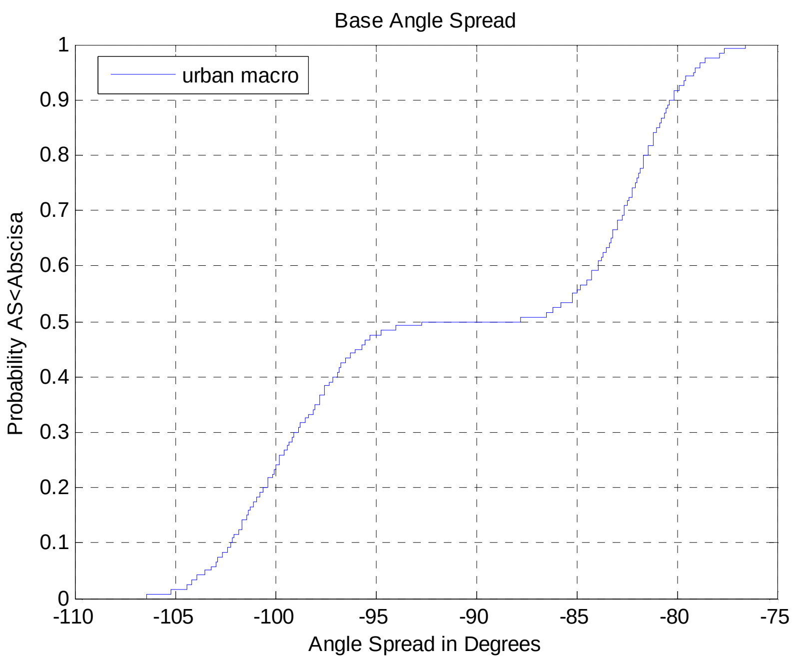

Figure 5.2: BS Angle Spread

- **MS Angle Spread**

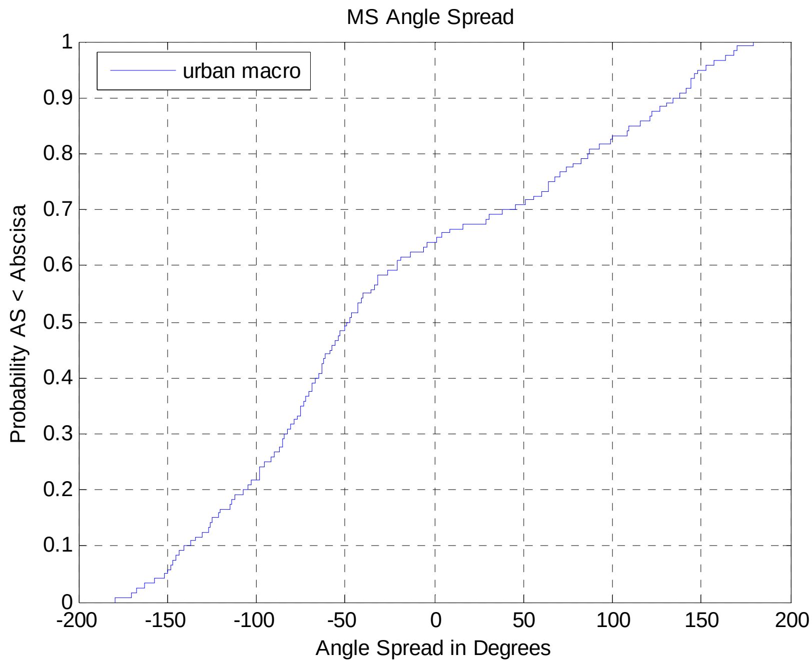

Figure 5.3: MS Angle Spread

- **Angle Spread and Lognormal Shadow Fading**

Table 5.9: Angle Spread and Lognormal Shadow Fading

	SCENARIO	**Angle Spread** σ_{AS}	**Lognormal Shadow Fading** σ_{SF}
8º RMS angle spread per path	**Suburban Macro**	2.6932	7.9156
15º RMS angle spread per path		5.2733	0.4960
8º RMS angle spread per path	**Urban Macro**	9.9199	0.4210
15º RMS angle spread per path		18.4310	0.4215

- **CDF of all path powers**

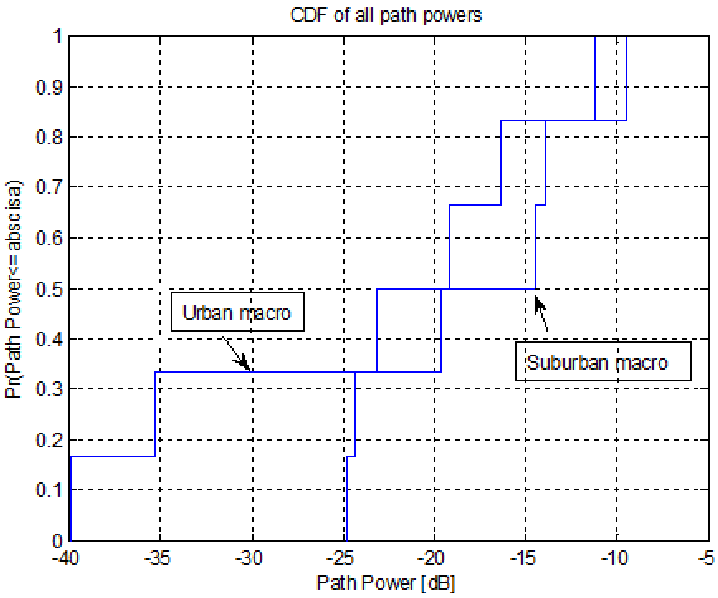

Figure 5.4: CDF of all path powers

6. Conclusions

It is clear the use of Multiple Input Multiple Output (MIMO) communication systems provides an important benefit from multipath propagation and multiply transfer rates by taking advantage of random fading and multipath delay spread.

There are different techniques that are propose for maximizing spectral efficiency and one of them is to avoid the use of multiple antenna techniques –MIMO and AAS (Adaptative Antenna Systems)- due to those techniques exhibits poor performance in 700 MHz because of the LOS or NLOS, typical to such deployments. On top of that, we have used a MIMO systems because all of its benefits even if in many situations the antenna arrays requires for MIMO in the 700 MHz band are too large to be a realistic option.

At present there is huge interest in digital television (DTV) because it can deliver vast amounts of information at very low cost to the maximum number of viewers, it can now be fully integrated into completely digital transmission networks and it can be packaged as never before.

The DVB-T SFN allows the broadcaster to achieve universal service coverage and a better spectrum efficiency.

7. List of Figures and Tables

8. Appendix

8.1. Generating Channel Model Parameters

[For a given scenario and a set of parameters, realizations of each channel model parameters such as the path delays, powers, and sub-path AoD and AoA can be derived.]

8.1.1. Generating Model Parameters for Urban and Suburban Macrocell Environments

Step 1: *Choose either an urban macrocell or suburban macrocell environment.*

Step 2: *Determine various distance and orientation parameters.*

Step 3: *Determine the DS, AS, and SF.*

Step 4: *Determine random delays for each of the **N** multipath components*.

Step 5: *Determine random average powers for each of the N multipath components*.

Step 6: *Determine AoDs for each of the N multipath components*.

Step 7: *Associate the multipath delays with AoDs.*

Step 8: *Determine the powers, phases and offset AoDs of the **M** = 20 sub-paths for each of the **N** paths at the BS.*

Step 9: *Determine the AoAs for each of the multipath components.*

Step 10: *Determine the offset AoAs at the UE of the M = 20 sub-paths for each of the N paths at the MS.*

Step 11: *Associate the BS and MS paths and sub-paths.*

Step 12: *Determine the antenna gains of the BS and MS sub-paths as a function of their respective sub-path AoDs and AoAs.*

Step 13: *Apply the path loss based on the BS to MS distance from Step 2, and the log normal shadow fading determined in step 3 as bulk parameters to each of the sub-path powers of the channel model.*

8.1.2. Generating Model Parameters for Urban Microcell Environments

[Urban microcell environments differ from the macrocell environments in that the individual multipaths are independently shadowed.]

Step 1: *Choose the urban microcell environment.*

Step 2: *Determine various distance and orientation parameters.*

Step 3: *Determine the bulk path loss and log normal shadow fading parameters.*

Step 4: *Determine the random delays for each of the N multipath components*.

Step 5: *Determine random average powers for each of the N multipath components*.

Step 6: *Determine AoDs for each of the N multipath components*.

Step 7: *Randomly associate the multipath delays with AoDs.*

Step 8: *Determine the powers, phases, and offset AoDs of the M = 20 sub-paths for each of the N paths at the BS.*

Step 9: *Determine the AoAs for each of the multipath components.*

Step 10: *Determine the offset AoAs of the M = 20 sub-paths for each of the N paths at the MS.*

Step 11: *Associate the BS and MS paths and sub-paths. Sub-paths are randomly paired for each path, and the sub-path phases defined at the BS and MS are maintained.*

Step 12: *Determine the antenna gains of the BS and MS sub-paths as a function of their respective sub-path AoDs and AoAs.*

Step 13: *Apply the path loss based on the BS to MS distance and the log normal shadow fading determined in Step 3 as bulk parameters to each of the sub-path powers of the channel model.*

8.2. MATLAB Code

scm.m

```
function [H, delays, full_output]=scm(scmpar,linkpar,antpar,initvalues)
%SCM 3GPP Spatial Channel Model (3GPP TR 25.996)
%   H=SCM(SCMPAR,LINKPAR,ANTPAR) is a 5D-array of channel coefficients. For
%   explanation of the input parameter structs, see SCMPARSET, LINKPARSET,
%   and ANTPARSET. SIZE(H)=[U S N T K], where U is the number of MS (RX)
%   elements, S is the number of BS (TX) elements, N is the number of
%   paths,  T is the number of time samples, and K is the number of links.
%   If K=1, the final dimension will be dropped, i.e. H is a 4D-array.
%
%   [H DELAYS]=SCM(...) outputs also a [KxN] matrix of path delays (in seconds).
%
%   [H DELAYS BULKPAR]=SCM(...) outputs also the struct BULKPAR, whose fields
%   are as follows:
%
%   When scmpar.ScmOptions is 'none' or 'urban_canyon':
%
%   delays           - path delays in seconds [KxN]
%   path_powers      - relative path powers [KxN]
%   aods             - angles of departure in degrees over (-180,180) [KxNxM]
%   aoas             - angles of arrival in degrees over (-180,180) [KxNxM]
%   subpath_phases   - final phases for subpaths in degrees over (0,360) [KxNxM]
%   path_losses      - path losses in linear scale [Kx1]
%   shadow_fading    - shadow fading losses in linear scale [Kx1]
%   delta_t          - time sampling intervals for all links [Kx1]
%
%   In addition, when scmpar.ScmOptions is 'los' (in addition to the above):
%
%   K_factors        - K factors for all links [Kx1]
%   Phi_LOS          - final phases for LOS paths in degrees over (-180,180) [Kx1]
%
%   When scmpar.ScmOptions is 'polarized' (in addition to scmpar.ScmOptions='none'):
%
%   subpath_phases   - final phases for subpaths in degrees over (0,360)
%                      [Kx4xNxM], where the second dimension are the [VV VH HV HH]
%                      components (iid).
%   xpd              - cross-polarization ratios in linear scale [Kx2xN],
%                      where the (:,1,:)th dimension is the V-to-H power coupling,
```

```
%                      and (:,2,:)th dimension is the H-to-V power coupling.
%
%
%   [H ...]=SCM(...,INIT_VALUES) uses initial values given in the struct
%   INIT_VALUES, instead of random parameter generation. INIT_VALUES has
%   the same format as BULKPAR, except that SUBPATH_PHASES are now the
%   initial phases. Also, time sampling intervals (delta_t) are not used
%   (they are recalculated for every call of SCM).
%
%   The 'far scatterer clusters' option [1, Sec. 5.5.2] is not currently
%   supported. The SCM options are mutually exclusive, i.e. one cannot, for
%   instance, choose 'polarized' and 'los' simultaneously.
%
%   Examples:
%       % to generate matrices for 10 links with default parameters
%       H=scm(scmparset,linkparset(10),antparset);
%       % to generate matrices for 'urban_macro' scenario
%       scmpar=scmparset;scmpar.Scenario='urban_macro';
%       H=scm(scmpar,linkparset(10),antparset);
%
%   Ref. [1]: 3GPP TR 25.996 v6.1.0 (2003-09)
%
%   See also SCMPARSET, LINKPARSET, ANTPARSET

%   Authors: Jari Salo (HUT), Giovanni Del Galdo (TUI), Pekka Kyösti (EBIT),
%   Daniela Laselva (EBIT), Marko Milojevic (TUI), Christian Schneider (TUI)
%   $Revision: 0.34$  $Date: Dec 12, 2004$


% Note: all units are in degrees, meters, Hertz (1/s) and meters/second (m/s)


ni=nargin;
if (ni<3 || ni>4)
    error('SCM requires three or four input arguments !')
end



% SCM parameters, common to all links
Scenario=scmpar.Scenario;
SampleDensity=scmpar.SampleDensity;
NumTimeSamples=scmpar.NumTimeSamples;
N=scmpar.NumPaths;
M=scmpar.NumSubPathsPerPath;
CenterFrequency=scmpar.CenterFrequency;
ScmOptions=scmpar.ScmOptions;
DelaySamplingInterval=scmpar.DelaySamplingInterval;
PathLossModel=scmpar.PathLossModel;
RandomSeed=scmpar.RandomSeed;
UniformTimeSampling=scmpar.UniformTimeSampling;
PathLossModelUsed=scmpar.PathLossModelUsed;
ShadowingModelUsed=scmpar.ShadowingModelUsed;
AnsiC_core=scmpar.AnsiC_core;
LookUpTable=scmpar.LookUpTable;

% antenna parameters
BsGainPattern=antpar.BsGainPattern;
BsGainAnglesAz=antpar.BsGainAnglesAz;
BsElementPosition=antpar.BsElementPosition;
MsGainPattern=antpar.MsGainPattern;
MsGainAnglesAz=antpar.MsGainAnglesAz;
MsElementPosition=antpar.MsElementPosition;
InterpFunction=antpar.InterpFunction;
InterpMethod=antpar.InterpMethod;

% link parameters
MsBsDistance=linkpar.MsBsDistance;
ThetaBs=linkpar.ThetaBs;
ThetaMs=linkpar.ThetaMs;
OmegaMs=linkpar.OmegaMs;
MsVelocity=linkpar.MsVelocity;
MsDirection=linkpar.MsDirection;
MsHeight=linkpar.MsHeight;
BsHeight=linkpar.BsHeight;
MsNumber=linkpar.MsNumber;
```

```
% check that the scenario is a valid string
if(any(strcmpi(Scenario,{'suburban_macro','urban_macro','urban_micro'}))==0)
    error('scmpar.Scenario must be ''suburban_macro'', ''urban_macro'', or
''urban_micro''')
end

% check that the ScmOptions is a valid string
if(any(strcmpi(ScmOptions,{'none','polarized','los','urban_canyon'}))==0)
    error('scmpar.Scmoptions must be ''none'', ''polarized'', ''los'', or
''urban_canyon'' ')
end


% check that SCM options comply with the selected scenario
if (strcmpi(ScmOptions,'urban_canyon')==1 && strcmpi(Scenario,'suburban_macro')==1 )
    scmpar.Scenario='urban_macro';
    warning('MATLAB:UrbanCanyonWrongScenario','Urban canyon option cannot be selected
with "suburban_macro" -> scenario changed to "urban_macro"')
end

if (strcmp(ScmOptions,'los')==1 && strcmp(Scenario,'urban_micro')==0 )
    scmpar.Scenario='urban_micro';
    warning('MATLAB:LineOfSightWrongScenario','LOS option can only be selected with
"urban_micro" -> scenario changed to "urban_micro"')
end


% extract the number of links
NumLinks=length(MsBsDistance);

% Check that the struct linkpar has the same number of parameters in
% each of its fields. This is also the number of links/users.
if (    NumLinks ~= length(ThetaBs)     ||...
        NumLinks ~= length(ThetaMs)     ||...
        NumLinks ~= length(OmegaMs)     ||...
        NumLinks ~= length(MsVelocity)  ||...
        NumLinks ~= length(MsDirection) ||...
        NumLinks ~= length(MsHeight)    ||...
        NumLinks ~= length(BsHeight)    ||...
        NumLinks ~= length(MsNumber))
    error('All fields in input struct LINKPAR must be of same size!')
end



% Set random seeds if given
if (isempty(RandomSeed)==0)
    rand('state',RandomSeed);
    randn('state',RandomSeed);
end



% determine the size of the MIMO system
% S - number of BS array antenna elements
if (numel(BsGainPattern)==1)
    S=scmpar.NumBsElements;
else
    S=size(BsGainPattern,1);
end

% U - number of MS array antenna elements
if (numel(MsGainPattern)==1)
    U=scmpar.NumMsElements;
else
    U=size(MsGainPattern,1);
end

% check that element displacement vector is of right size
if (length(BsElementPosition)~=S && length(BsElementPosition)~=1)
    error('antpar.BsElementPosition has wrong size!')
end

if (length(MsElementPosition)~=U && length(MsElementPosition)~=1)
```

```
    error('antpar.MsElementPosition has wrong size!')
end


% check that LUT size is a power-of-two
if (strcmpi(AnsiC_core,'yes')==1)
    if (LookUpTable>0)
        if (2^nextpow2(LookUpTable)-LookUpTable~=0)
            scmpar.LookUpTable=2^nextpow2(LookUpTable);
            warning('MATLAB:LUTSizeChanged',['scmpar.LookUpTable is not a power-of-2:
size changed to ' num2str(scmpar.LookUpTable) '.'])
        end
    end
end


% These features are not included in this version, so they are fixed
FixedPdpUsed='no'; FixedAnglesUsed='no';
if (strcmpi(FixedPdpUsed,'yes')==1 && N~=6)
    scmpar.NumPaths=6; N=6;
    warning('MATLAB:NumPathsChangedPdp',['Using fixed PDP, scmpar.NumPaths changed to '
num2str(scmpar.NumPaths) '.'])
elseif (strcmpi(FixedAnglesUsed,'yes')==1 && N~=6)  % if fixed AoD/AoAs are used,
NumPaths must be six
    scmpar.NumPaths=6; N=6;
    warning('MATLAB:NumPathsChangedAoa',['Using fixed AoD/AoAs, scmpar.NumPaths changed
to ' num2str(scmpar.NumPaths) '.'])
end



% GENERATION OF RANDOM "BULK" PARAMETERS FOR ALL LINKS
switch (ni)

    case (3)    % do the basic thing

        % check that M=20
        if (M ~= 20)
            scmpar.NumSubPathsPerPath=20; M=20;
            warning('MATLAB:NumSubPathsChanged','NumSubPathsPerPath is not 20! Using
NumSubPathsPerPath=20 instead.')
        end


        % generate bulk parameters for all links
        bulkpar=generate_bulk_par(scmpar,linkpar,antpar);

        % for interpolation
        aods=bulkpar.aods;
        aoas=bulkpar.aoas;


    case (4)    % do not generate random link parameters, use initial values

        % take bulk parameters from input struct
        bulkpar=initvalues;

        % for interpolation
        aods=bulkpar.aods;
        aoas=bulkpar.aoas;


end



% ANTENNA FIELD PATTERN INTERPOLATION
% Interpolation is computationally intensive, so avoid it if possible.
% Since SCM does not support elevation, dismiss the elevation dimension (for now)
% NOTE: aods/aoas should be given in degrees.
BsGainIsScalar=0;
MsGainIsScalar=0;
if numel(BsGainPattern)>1
    if (strcmp(ScmOptions,'polarized')==1)
        BsGainPatternInterpolated = zeros([2 S size(aods)]); % [polarizations(2)
elements links N(6) M(20)]
```

```
BsGainPatternInterpolated(1,:,:,:,:)=feval(InterpFunction,squeeze(BsGainPattern(:,1,1,:)
),BsGainAnglesAz,aods, InterpMethod); % V

BsGainPatternInterpolated(2,:,:,:,:)=feval(InterpFunction,squeeze(BsGainPattern(:,2,1,:)
),BsGainAnglesAz,aods, InterpMethod); % H
        BsGainPatternInterpolated=permute(BsGainPatternInterpolated,[3 2 1 4 5]); %
[link rx_element polarization path subpath]
    else

BsGainPatternInterpolated=feval(InterpFunction,squeeze(BsGainPattern(:,1,1,:)),BsGainAng
lesAz,aods, InterpMethod); % V only
        BsGainPatternInterpolated=permute(BsGainPatternInterpolated,[2 1 3 4]);
    end
else    % if BsGainPattern is scalar
    if (strcmp(ScmOptions,'polarized')==1)
        BsGainPatternInterpolated=repmat(BsGainPattern, [NumLinks S 2 N M]);    % [link
rx_element polarization path subpath]
        BsGainIsScalar=1;
    else
        BsGainPatternInterpolated=repmat(BsGainPattern, [NumLinks S N M]);
        BsGainIsScalar=1;
    end
end

if numel(MsGainPattern)>1
    if (strcmp(ScmOptions,'polarized')==1)
        MsGainPatternInterpolated=zeros([2 U size(aoas)]);% [polarizations(2) elements
links N(6) M(20)]

MsGainPatternInterpolated(1,:,:,:,:)=feval(InterpFunction,squeeze(MsGainPattern(:,1,1,:)
),MsGainAnglesAz,aoas, InterpMethod); % V

MsGainPatternInterpolated(2,:,:,:,:)=feval(InterpFunction,squeeze(MsGainPattern(:,2,1,:)
),MsGainAnglesAz,aoas, InterpMethod); % H
        MsGainPatternInterpolated=permute(MsGainPatternInterpolated,[3 2 1 4 5]); %
[link Ms_element polarization path subpath]
    else

MsGainPatternInterpolated=feval(InterpFunction,squeeze(MsGainPattern(:,1,1,:)),MsGainAng
lesAz,aoas, InterpMethod); % V only
        MsGainPatternInterpolated=permute(MsGainPatternInterpolated,[2 1 3 4]);
    end
else    % if MsGainPattern is scalar
    if (strcmp(ScmOptions,'polarized')==1)
        MsGainPatternInterpolated=repmat(MsGainPattern, [NumLinks U 2 N M]);    % [link
rx_element polarization path subpath]
        MsGainIsScalar=1;
    else
        MsGainPatternInterpolated=repmat(MsGainPattern, [NumLinks U N M]);
        MsGainIsScalar=1;
    end
end

% Note: The gain patterns at this point have size(MsGainPatternInterpolated) = [link
rx_element path subpath]
%  OR
% size(MsGainPatternInterpolated) = [link rx_element polarization path subpath]
% (the same for BsGainPatternInterpolated)

% Do antenna field pattern interpolation for the LOS path
if (strcmpi(ScmOptions,'los')==1)
    if numel(BsGainPattern)>1
        BsGain_Theta_BS=
feval(InterpFunction,squeeze(BsGainPattern(:,1,1,:)),BsGainAnglesAz,ThetaBs(:),
InterpMethod); % V only
        BsGain_Theta_BS= BsGain_Theta_BS.'; % size()= [NumLinks S]
    else
        BsGain_Theta_BS=repmat(BsGainPattern,[NumLinks S]);
    end

    if numel(MsGainPattern)>1
        MsGain_Theta_MS=
feval(InterpFunction,squeeze(MsGainPattern(:,1,1,:)),MsGainAnglesAz,ThetaMs(:),
InterpMethod); % V only
        MsGain_Theta_MS= MsGain_Theta_MS.'; % size()= [NumLinks U]
```

```
    else
        MsGain_Theta_MS= repmat(MsGainPattern,[NumLinks U]);
    end
else
    % Set dummy values in case LOS option is not used
    BsGain_Theta_BS=NaN;
    MsGain_Theta_MS=NaN;

end


% CHANNEL MATRIX GENERATION
[H delta_t FinalPhases FinalPhases_LOS] = scm_core( scmpar,...
                                                    linkpar,...
                                                    antpar,...
                                                    bulkpar,...
                                                    BsGainPatternInterpolated,...
                                                    BsGain_Theta_BS,...             %
gain of LOS path
                                                    MsGainPatternInterpolated,...
                                                    MsGain_Theta_MS,...             %
gain of LOS path
                                                    0,...                            %
offset time (not used typically)
                                                    BsGainIsScalar,...
                                                    MsGainIsScalar);

% final phases
bulkpar.subpath_phases=FinalPhases

% time sampling grid
bulkpar.delta_t=delta_t;


% If path loss and shadowing are to be multiplied into the output
if ( (strcmpi(PathLossModelUsed,'yes')==1) || strcmpi(ShadowingModelUsed,'yes')==1 )

    if (size(H,5)==1) % only one link
        if (strcmpi(PathLossModelUsed,'yes')==1)
            H=sqrt(bulkpar.path_losses).*H;   % path loss in linear scale
        end

        if (strcmpi(ShadowingModelUsed,'yes')==1)
            H=H*sqrt(bulkpar.shadow_fading);          % shadow fading in linear scale
        end
    else    % if more than one link

        siz_H=size(H);
        Hmat=reshape(H,prod(siz_H(1:end-1)),siz_H(end));  % a matrix with NumLinks cols
        if (strcmpi(PathLossModelUsed,'yes')==1)
            pl_mat=diag(sparse(sqrt(bulkpar.path_losses)));
            Hmat=Hmat*pl_mat;           % multiply path loss into each link
        end

        if (strcmpi(ShadowingModelUsed,'yes')==1)
            sf_mat=diag(sparse(sqrt(bulkpar.shadow_fading)));    % shadow fading is in
linear scale
            Hmat=Hmat*sf_mat;           % multiply shadow fading into each link
        end

        H=reshape(Hmat,siz_H);      % put back to original size

    end
end


% GENERATE OUTPUT
no=nargout;
if (no>1)
    delays=bulkpar.delays;
end

if (no>2)
    switch lower(ScmOptions)
        case {'none','urban_canyon','polarized'}
            full_output=bulkpar;
```

```
        case ('los')
            bulkpar.Phi_LOS=FinalPhases_LOS;
            full_output=bulkpar;

    end
end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% A function that maps inputs from (-inf,inf) to (-180,180)
function y=prin_value(x)
y=mod(x,360);
y=y-360*floor(y/180);
```

scmparset.m

```
%SCMPARSET Model parameter configuration for SCM
%   SCMPAR=SCMPARSET sets default parameters for the input struct SCMPAR
%   (see SCM).
%
%   SCMPARSET parameters [ {default} ]:
%
%   NumBsElements           - Number of BS array antenna elements [ {2} ]
%   NumMsElements           - Number of MS array antenna elements [ {2} ]
%   Scenario                - SCM scenario [ suburban_macro | urban_macro |
{urban_micro} ]
%   SampleDensity           - number of time samples per half wavelength [ {2} ]
%   NumTimeSamples          - number of time samples [ {100} ]
%   UniformTimeSampling     - Use same time sampling grid for all links [ yes | {no} ]
%   BsUrbanMacroAs          - BS angle spread for urban macro in degrees [ {eight} |
fifteen ]
%   NumPaths                - number of paths [ {6} ]
%   NumSubPathsPerPath      - number of subpaths per path [ {20} ] (cannot be changed)
%   CenterFrequency         - carrier frequency in Herz [ {2e9} ]
%   ScmOptions              - SCM optional features [ {none} | polarized | los |
urban_canyon ]
%   DelaySamplingInterval   - delay sampling grid [ {1.6276e-008} ]
%   XpdIndependentPower     - power normalization with polarization [ yes | {no} ]
%   PathLossModelUsed       - usage of path loss model [ yes | {no} ]
%   ShadowingModelUsed      - usage of shadow fading model [ yes | {no} ]
%   PathLossModel           - path loss model function name [ {pathloss} ]
%   AnsiC_core              - use optimized computation [ yes | {no} ]
%   LookUpTable             - look up EXP(j*THETA) from a table [{0}]
%   RandomSeed              - sets random seed [ {[empty]} ]
%
%   Notes about parameters:
%   - The number of BS and MS elements is normally extracted from ANTPAR.
%     The values of NumBsElements and NumMsElements are used only if a single
%     scalar is given as the antenna field pattern in ANTPAR (see ANTPARSET).
%   - For successful Doppler analysis, one should select SampleDensity > 1.
%     The time sample interval is calculated from CenterFrequency and
%     MsVelocity (see LINKPARSET) according to wavelength/(2*MsVelocity*SampleDensity).
%     The calculated time sample interval for each link is included in the optional
%     output argument (delta_t) of SCM.
%   - If UniformTimeSampling is 'yes' all links will be sampled at
%     simultaneous time instants. In this case, the time sample interval is
%     the same for all links it is calculated by replacing MsVelocity with
%     MAX(MsVelocity), where the maximum is over all links.
%   - Number of paths can be changed at will, although [1] supports only
%     six paths. The delays and mean AoD/AoAs are generated according to
%     [1]. Subpath AoD/AoAs are always taken from [1, Table 5.2].
%   - Number of subpaths is fixed to 20. This is because the AoD/AoAs for
%     subpaths in SCM have fixed angle spread given in [1, Table 5.2].
%   - CenterFrequency affects path loss and time sampling interval.
%   - DelaySamplingInterval determines the sampling grid in delay domain.
%     All path delays are rounded to the nearest grid point. It can also
%     be set to zero. Default value is 1/(16*3.84e6) seconds [1].
%   - With XpdIndependentPower='yes' the power of the channel matrices
%     is normalized to a constant. Otherwise, it depends on the random XPD
%     ratios, and incorporates the polarization dependent part of the path
%     loss. The default value 'no' complies with [1].
%   - When PathLossModelUsed is 'no' the path losses are still computed for
%     each link but they are not multiplied into the channel matrices. If
```

```
%     ShadowingModelUsed is also 'no', each channel matrix element has unit
%     mean power (summed over all paths). In other words,
%     MEAN(MEAN(ABS(SUM(H,3)).^2,4),5) is a matrix of (approximately) ones
%     when isotropic unit-gain antennas are used. Exception: with
%     'polarized' option the power normalization is by default random
%     (depends on XPDs). See parameter scmpar.XpdIndependentPower.
%   - Path loss model is implemented in a separate function, whose name is
%     defined in PathLossModel. For syntax, see PATHLOSS.
%   - The C-function must be compiled before usage. For more information
%     of the ANSI-C core function, see SCM_MEX_CORE.
%   - The LookUpTable parameter defines the number of points used in the
%     cosine look-up table; a power-of-2 should be given. The look-up table
%     is used only in the ANSI-C optimized core function. Value 0 indicates
%     that look-up table is not used. Value -1 uses the default number of
%     points, which is 2^14=16384. Since a large part of computation in SCM
%     involves repeated evaluation of a complex exponential, the look-up
%     table can speed up computation on certain platforms and C compilers.
%   - Even fixing the random seed may not result in fully repeatable
%     simulations due to differences in e.g. MATLAB versions.
%
%   Ref. [1]: 3GPP TR 25.996 v6.1.0 (2003-09)
%
%   See also SCM, LINKPARSET, ANTPARSET.

%   Authors: Jari Salo (HUT), Pekka Kyösti (EBIT), Daniela Laselva (EBIT),
%   Giovanni Del Galdo (TUI), Marko Milojevic (TUI), Christian Schneider (TUI)
%   $Revision: 0.41 $  $Date: Jan 14, 2005$




if length(varargin)>0
    error('No such functionality. Try ''scmpar=scmparset'' instead.')
end

% Set the default values
scmpar=struct(  'NumBsElements',2,...
                'NumMsElements',2,...
                'Scenario','urban_micro',...
                'SampleDensity', 2,...                    % in samples/half-wavelength
                'NumTimeSamples',100,...
                'UniformTimeSampling','no',...
                'BsUrbanMacroAS','fifteen',...            % choices: 'eight' and
'fifteen'.
                'NumPaths',6,...
                'NumSubPathsPerPath',20,...               % only value supported is 20.
                'CenterFrequency',7e8,...                 % New frequency of 700 MHz [in
Herz]
                'ScmOptions','none',...                   %
'none','polarized','los','urban_canyon'
                'DelaySamplingInterval',1.6276e-008,... % default=1/(16*3.84e6), see [1]
                'XpdIndependentPower','no',...            % with 'yes' normalizes matrix
power with polarized arrays
                'PathLossModelUsed','no',...
                'ShadowingModelUsed','no',...
                'PathLossModel','pathloss',...
                'AnsiC_core','no',...
                'LookUpTable',0,...                       % number of points in Ansi-C
core look-up table for cosine, 0 if not used
                'RandomSeed',[]);                         % if empty, seed is not set.
```

linkparset.m

```
function linkpar=linkparset(varargin)
%LINKPARSET Link parameter configuration for SCM
%   LINKPAR=LINKPARSET(K) is a struct consisting of randomly generated link
%   parameters for K links. LINKPAR=LINKPARSET(K,RMAX) uses cell radius
%   RMAX for generation of MS-BS distances (default: 500 meters).
%
%   LINKPAR=LINKPARSET(...,SEED) sets the random seed used in link
%   parameter generation.
%
%   The parameters and their defaults are:
%
%   MsBsDistance     - see below
```

```
%   ThetaBs          - U(-180,180) degrees, U denotes uniform pdf
%   ThetaMs          - U(-180,180) degrees
%   OmegaBs          - NaN, this parameter is not currently used
%   OmegaMs          - NaN, this parameter is not currently used
%   MsVelocity       - 10 meters per second
%   MsDirection      - U(-180,180) degrees with respect to broadside
%   MsHeight         - 1.5 meters
%   BsHeight         - 32 meters
%   MsNumber         - [1:K], i.e. all simulated links are different MSs
%
%   See [1, Fig. 5.2].
%
%   The pdf of the random variable (RV) MsBsDistance is R+RMIN, where R is
%   an RV with pdf p(r)=2*r/r0^2, where r0 defaults to (RMAX-RMIN) meters.
%   Hence, MsBsDistance is an RV such that users are approximately
%   uniformly distributed in a circular disk over [RMIN,RMAX] meters. RMIN
%   is fixed to 35 meters because some path loss models do not support
%   distances smaller than this. RMAX defaults to 500 meters as this is the
%   radius of microcell given in [1]. For usability, the same default is
%   used for all scenarios.
%
%   Some further notes about the parameters:
%
%   - OmegaBs and OmegaMs define the orientation of antenna broadside with
%     respect to north. This parameter is currently redundant.
%   - MsHeight and BsHeight defaults are based on [1, Table 5.1].
%   - MsNumber is a positive integer defining the index number of MS for
%     each link. This parameter is used in generation of inter-site
%     correlated shadow fading values; shadow fading is correlated for
%     links between a single MS and multiple BSs. There is no correlation
%     in shadow fading between different MSs. Examples: The default value
%     is the case where all links in a call to the SCM function correspond
%     to different MSs. Setting MsNumber=ones(1,K) corresponds to the case
%     where the links from a single MS to K different BSs are simulated.
%
%
%   Ref. [1]: 3GPP TR 25.996 v6.1.0 (2003-09)
%
%   See also SCM, SCMPARSET, ANTPARSET.

%   Authors: Jari Salo (HUT), Pekka Kyösti (EBIT), Daniela Laselva (EBIT),
%   Giovanni Del Galdo (TUI), Marko Milojevic (TUI), Christian Schneider (TUI)
%   $Revision: 0.31$  $Date: Jan 14, 2005$


% defaults
num=1;      % number of links
rmax=500;   % cell radius

rmin=35;    % to prevent warnings from path loss models.

ni=length(varargin);
if ni>0, if (~isempty(varargin{1})), num=varargin{1}; end, end
if ni>1, if (~isempty(varargin{2})), rmax=varargin{2}; end, end
if ni>2, if (~isempty(varargin{3})), seed=varargin{3}; rand('state',floor(seed)); end,
end
if ni>3, error('Too many input arguments!'), end



linkpar=struct( 'MsBsDistance',distrnd(num,rmax-rmin)+rmin,...
                'ThetaBs',360*(rand(1,num)-0.5),...
                'ThetaMs',360*(rand(1,num)-0.5),...
                'OmegaBs',repmat(NaN,1,num),...
                'OmegaMs',repmat(NaN,1,num),...
                'MsVelocity',repmat(10,1,num),...
                'MsDirection',360*(rand(1,num)-0.5),...
                'MsHeight',repmat(1.5,1,num),...
                'BsHeight',repmat(32,1,num),...
                'MsNumber',1:num);




function d=distrnd(num,rmax)
```

```
% DISTRND Distance from BS in a circular cell
%   D=DISTRND(K,RMAX) generates K random variables from the pdf
%   p(r)=2*r/RMAX^2. This is the pdf of distance from base station when
%   users are uniformly (in area) distributed in a cell with radius RMAX.

%   Authors: Jari Salo (HUT), Marko Milojevic (TUI)
%   $Revision: 0.1$  $Date: Sep 30, 2004$

% create random variables from triangular pdf whose width is 2*rmax
a=sum(rmax*rand(2,num));

% fold the random variables about the rmax
inds=find(a>rmax);
a(inds)=-a(inds)+2*rmax;

d=a(:).';
```

antparset.m

```
function antpar=antparset(varargin)
%ANTPARSET Antenna parameter configuration for SCM
%   ANTPAR=ANTPARSET sets default parameters for the input struct ANTPAR.
%
%   Default parameters are [ {default} ]:
%
%   BsGainPattern       - complex BS array element field patterns [ {1} | 4D-array]
%   BsGainAnglesAz      - azimuth angles (degrees) for BsGainPattern [ {linspace(-
180,180,90)} ]
%   BsGainAnglesEl      - elevation angles (not used currently)
%   BsElementPosition   - element positions for BS linear array in wavelenghts [ {0.5} ]
%   MsGainPattern       - complex MS array element field patterns [ {1} | 4D-array]
%   MsGainAnglesAz      - azimuth angles (degrees) for MsGainPattern [ {linspace(-
180,180,90)} ]
%   MsGainAnglesEl      - elevation angles (not used currently)
%   MsElementPosition   - element positions for MS linear array in wavelenghts [ {0.5} ]
%   InterpFunction      - name of the interpolation function [{'interp_gain'}]
%   InterpMethod        - interpolation method used  [{'cubic'}]
%
%   Some notes about the antenna parameters:
%
%   - The complex field patterns are given in linear scale. The antenna gain
%     is 20*log10(abs(BsGainPattern)).
%   - Field patterns should be defined over the full 360 degree azimuth
%     angle. Unless BsGainPattern is a scalar (see below), the intermediate
%     values will be interpolated.
%   - Only linear arrays are supported currently. The element positions can
%     be given (in wavelengths) in the vectors BsElementPosition and
%     MsElementPosition. When a scalar is given (default), uniform spacing
%     is assumed.
%   - If BsGainPattern and/or MsGainPattern field is a scalar, the antenna
%     field pattern is assumed constant (equal to the scalar) over the whole
%     azimuth angle. For example, setting BsGainPattern=SQRT(1.64) (2.15 dB)
%     would correspond to a BS dipole array with NumBsElements (see below).
%   - When BsGainPattern (MsGainPattern) is a scalar, the number of the
%     BS (MS) antenna elements is determined from parameters NumBsElements
%     (NumMsElements) in the input struct SCMPAR (see SCMPARSET). Otherwise,
%     the number of elements in the link end is deduced from the dimensions
%     of the 4D-array BsGainPattern (MsGainPattern).
%   - If BsGainPattern (MsGainPattern) is not a scalar it must be a complex
%     4D-array with dimensions NUM_ELxPOLxELxAZ, where NUM_EL is the
%     number of array elements, POL is 1 or 2, EL is arbitrary, and AZ
%     is LENGTH(BsGainAnglesAz). If 'polarized' option is used, the
%     (:,1,1,:)th dimension is assumed the vertical polarization and (:,2,1,:)
%     is assumed the horizontal polarization. Otherwise, only the (:,1,1,:)th
%     dimensions are used. The size of the third dimension is unimportant
%     as elevation is not used in the current implementation.
%   - SIZE(BsGainPattern,4) must equal LENGTH(BsAnglesAz). In other words,
%     all element patterns are defined over the same azimuth grid. The same
%     applies for MsGainPattern and MsAnglesAz.
%   - InterpFunction defines the name of the interpolating function. One
%     can also use his own function. For syntax, see INTERP_GAIN.
%   - InterpMethod depends on the interpolating function used. INTERP_GAIN
%     uses the MATLAB's INTERP1 function to do the dirty work. Recommended
%     methods are: 'cubic' or 'linear'. For faster computation, see
%     INTERP_GAIN_C.
```

```
%
%   See also DIPOLE, INTERP_GAIN, INTERP_GAIN_C.

%   Authors: Jari Salo (HUT), Pekka Kyösti (EBIT), Daniela Laselva (EBIT),
%   Giovanni Del Galdo (TUI), Marko Milojevic (TUI), Christian Schneider (TUI)
%   $Revision: 0.11 $  $Date: Dec 4, 2004$



if (length(varargin)>2)
    error('No such functionality yet. Try ''antpar=antparset'' instead.')
end


antpar=struct(  'BsGainPattern',{1},...                         % in general:
[Number_of_antennas, 2, Elevation_points, Azimuth_points]
                'BsGainAnglesAz',{linspace(-180,176,90)},...    % size [1
Azimuth_points]
                'BSGainAnglesEl',{0},...                        % size [1
Elevation_points] (parameter ignored)
                'BsElementPosition',[0.5],...                   % in wavelengths. When
scalar, uniform spacing assumed
                'MsGainPattern',{1},...
                'MsGainAnglesAz',{linspace(-180,176,90)},...
                'MsGainAnglesEl',{0},...
                'MsElementPosition',[0.5],...                   % in wavelengths. When
scalar, uniform spacing assumed
                'InterpFunction','interp_gain',...              % name of the
interpolation function
                'InterpMethod','cubic');                        % interpolation method,
depends on the function used
```

pathloss.m

```
function loss=pathloss(scmpar,linkpar)
%PATHLOSS Pathloss models for 700MHz
%   PATH_LOSSES=PATHLOSS(SCMPAR,LINKPAR) returns path losses in dB scale
%   for all links defined in SCM input struct LINKPAR for the center
%   frequency and scenario given in SCMPAR. The output is a column vector
%   whose length is equal to the number of links defined in LINKPAR, e.g.
%   LENGTH(LINKPAR.MsBsDistance). The center frequencies and distances  in
%   SCMPAR must be specified in Herzes and meters, respectively.
%
%   PATHLOSS supports 700 MHz center frequency and the SCM scenarios:
%   suburban macro, urban macro, and urban micro [1].  MS and BS heights
%   are not currently supported.
%
%   Refs.   [1]: 3GPP TR 25.996 v6.1.0 (2003-09)
%
%   See also SCMPARSET and LINKPARSET.




% extract required parameters from the input structs
NumUsers=length(linkpar.MsBsDistance);
MsBsDistance=linkpar.MsBsDistance;
Scenario=scmpar.Scenario;
CenterFrequency=scmpar.CenterFrequency;
Options=scmpar.ScmOptions;

% print out a warning if center freqency is not within a tolerance
tol=7e7;    % Hz
if (abs(CenterFrequency - 7e8)>tol)
    CenterFrequency=7e8;
    warning('MATLAB:CenterFrequencyChanged','Center frequency of 700 MHz used for path
loss computation.')
else
    CenterFrequency=7e8;
end
```

```
switch lower(Scenario)

    case {'suburban_macro'}

        switch (CenterFrequency)            % other frequencies may be added, if
necessary

            case (7e8)                      % SCM suburban macro [1, Section 5.2 and
Table 5.1]
                if (min(MsBsDistance)<35)
                    warning('MATLAB:TooSmallMsBsDistance','MsBsDistance less than 35
meters encountered. Path loss computation may be unreliable.')
                end
                loss=113.9+35*log10(MsBsDistance);
        end


    case {'urban_macro'}

        switch (CenterFrequency)            % other frequencies may be added, if
necessary

            case (7e8)                      % SCM urban_macro model [1, Section 5.2 and
Table 5.1]
                if (min(MsBsDistance)<35)
                    warning('MATLAB:TooSmallMsBsDistance','MsBsDistance less than 35
meters encountered. Path loss computation may be unreliable.')
                end
                loss=123.17+35*log10(MsBsDistance);

        end


    case {'urban_micro'}

        % options for urban micro
        switch lower(Options)

            case {'none','polarized','urban_canyon'}

                switch (CenterFrequency)    % other frequencies may be added, if
necessary

                    case (7e8)              % SCM urban micro model [1, Section 5.2 and
Table 5.1]
                        if (min(MsBsDistance)<20)
                            warning('MATLAB:TooSmallMsBsDistance','MsBsDistance less
than 20 meters encountered. Path loss computation may be unreliable.')
                        end
                        loss=95.68+35*log10(MsBsDistance);

                end


            case ('los')

                switch (CenterFrequency)    % other frequencies may be added, if
necessary

                    case (7e8)              % SCM urban micro model [1, Section 5.2 and
Table 5.1]
                        if (min(MsBsDistance)<20)
                            warning('MATLAB:TooSmallMsBsDistance','MsBsDistance less
than 20 meters encountered. Path loss computation may be unreliable.')
                        end
                        loss=95.68+35*log10(MsBsDistance);

                end


        end % end options for urban micro
```

```
end     % end switch Scenario


% output
loss=loss(:);
```

interp_gain.m

```
function gains = interp_gain(field_patterns, angles, at_values, interp_method)
%INTERP_GAIN Antenna field pattern interpolation
%   G = INTERP_GAIN(PAT, ANGLES, DATA, METHOD) are the complex antenna
%   field patterns interpolated at azimuth angles given in DATA,  given in
%   degrees. SIZE(G)=[NUM_EL SIZE(DATA)], where NUM_EL is the  number of
%   rows in matrix PAT. PAT has LENGTH(ANGLES) columns; ANGLES  is a vector
%   defining the angles (in degrees) at which the antenna  element patterns
%   in the rows of PAT have been defined. Note: LENGTH(ANGLES) must equal
%   SIZE(PAT,2), i.e. all field patterns have to be specified  over the
%   same azimuth grid.
%
%   METHOD is a string defining interpolation method. For a list of
%   methods, see INTERP1. It is recommended that the antenna field patterns
%   in PAT  are given so that there are no duplicate points in ANGLES and
%   that the  support of the interpolated function spans over the entire
%   azimuth angle, i.e. 360 degrees. (Note that e.g. linear interpolation
%   cannot  extrapolate values falling outside the support of the
%   interpolated  function.)
%
%   Phase and magnitude are interpolated separately.
%
%   Elevation interpolation is not supported currently.
%
%   See also INTERP_GAIN_C.

%   Authors: Jari Salo (HUT), Jussi Salmi (HUT), Giovanni Del Galdo (TUI)
%   $Revision: 0.1 $  $Date: July 22, 2004$


% if it's a vector, make sure that it's a row vector
if (min(size(field_patterns,2)==1))
    field_patterns=field_patterns(:).';
end

if (size(field_patterns,2) ~= length(angles(:)))
    error('Size mismatch in antenna parameters! ')
end

siz_at_values    = size(at_values);
nd               = ndims(at_values);
num_elements     = size(field_patterns,1);

at_values=prin_value(at_values);

% interpolation
% Note that extrapolation is not possible with e.g. linear interpolation
% Note also that interpolated values are in degrees
if (isreal(field_patterns)==0)  % if complex-valued do amplitude and phase separately
    int_gain = interp1(angles(:), abs(field_patterns.'), at_values(:),interp_method);
    int_phase = interp1(angles(:), unwrap(angle(field_patterns.'),1),
at_values(:),interp_method);
else    % otherwise interpolate only the real part
    int_gain = interp1(angles(:), field_patterns.', at_values(:),interp_method);
    int_phase= -pi*(-0.5+0.5*sign(int_gain));   % take into account the sign of real
part
end

% back to complex values, this has size [PROD(siz_at_values) num_elements]
abs_gain=abs(int_gain);
gains=complex(abs_gain.*cos(int_phase), abs_gain.*sin(int_phase));

% make the output size [num_elements siz_at_values]
gains=reshape(gains,[siz_at_values num_elements]);
gains=permute(gains,[nd+1 1:nd]);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% A function that maps inputs from (-inf,inf) to (-180,180)
function y=prin_value(x)
y=mod(x,360);
y=y-360*floor(y/180);
```

generate_bulk_par.m

```
function [bulk_parameters]=generate_bulk_par(scmpar,linkpar,antpar)
%GENERATE_BULK_PAR Generation of SCM bulk parameters
%   [BULK_PAR]=GENERATE_BULK_PAR(SCMPAR,LINKPAR,ANTPAR) generates the
%   "bulk" parameters according to 3GPP TR 25.996. For explanation of
%   the input structs, see SCMPARSET, LINKPARSET, and ANTPARSET.
%   Denoting with K the number of links, N the number of paths,
%   M the number of subpaths, the fields BULK_PAR are as follows:
%
%   When scmpar.ScmOptions is 'none' or 'urban_canyon':
%   delays           - path delays in seconds [KxN]
%   path_powers      - relative path powers [KxN]
%   aods             - angles of departure in degrees over (-180,180) [KxNxM]
%   aoas             - angles of arrival in degrees over (-180,180) [KxNxM]
%   subpath_phases   - random phases for subpaths in degrees over (0,360) [KxNxM]
%   path_losses      - path losses in linear scale [Kx1]
%   shadow_fading    - shadow fading losses in linear scale [Kx1]
%
%   In addition, when scmpar.ScmOptions is 'los' (in addition to the above):
%   K_factors        - K factors for all links [Kx1]
%   Phi_LOS          - random phases for LOS paths in degrees over (-180,180) [Kx1]
%
%   When scmpar.ScmOptions is 'polarized' (in addition to scmpar.ScmOptions='none'):
%   subpath_phases   - random phases for subpaths in degrees over (0,360)
%                      [Kx4xNxM], where the second dimension are the [VV VH HV HH]
%                      components (iid).
%   xpd              - cross-polarization ratios in linear scale [Kx2xN],
%                      where the (:,1,:)th dimension is the V-to-H power coupling,
%                      and (:,2,:)th dimension is the H-to-V power coupling.
%
%   Ref. [1]: 3GPP TR 25.996 v6.1.0 (2003-09)
%
%   See also SCM.

%   Authors: Jari Salo (HUT), Daniela Laselva (EBIT), Giovanni Del Galdo (TUI),
%   Marko Milojevic (TUI), Pekka Kyösti (EBIT), Christian Schneider (TUI)
%   $Revision: 0.21 $  $Date: Dec 14, 2004$


% Input parameter validity checking is done in the main function.


% extract certain parameters from the input structs
Scenario=scmpar.Scenario;


switch lower(Scenario)

    % SUBURBAN MACRO AND URBAN MACRO, [1, Sec. 5.3.1]
    case {'suburban_macro','urban_macro'}

        bulk_parameters=macro(scmpar,linkpar,antpar);


        % URBAN MICRO, [1, Sec. 5.3.2]
    case {'urban_micro'}

        bulk_parameters=micro(scmpar,linkpar,antpar);

end     % end of user parameter generation main program
```

```
% FUNCTION DEFINITIONS
%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% A function to generate sigma_as, sigma_ds and sigma_sf for all links
% Step 3 in [1, Sec 5.3.1], see also [1, Sec 5.6].
% Here Section 5.6 in [1] is interpreted so that it describes the channel matrix
% generation for a single MS only. Hence, there is inter-site correlation
% only between radio links between a single MS and multiple BSs.
function sigmas=step3(scmpar,linkpar)

% extract certain parameters from the input structs
MsNumber=linkpar.MsNumber(:);
Scenario=scmpar.Scenario;
ScmOptions=scmpar.ScmOptions;

NumLinks=length(MsNumber);

% matrices from [1,Sec. 5.6]
%alfa_beta=0.5;
%gamma_beta=-0.6;
%gamma_alfa=-0.6;
%A=[1 alfa_beta gamma_alfa; alfa_beta 1 gamma_beta;gamma_alfa gamma_beta 1];
%B=[0 0 0; 0 0 0; 0 0 0.5];
Bsq=[0 0 0; 0 0 0; 0 0 0.7071];
bsq=0.7071;     % Bsq(3,3)
% pre-computed value: C=sqrtm(A-B)
C = [0.8997 0.1926 -0.3917; 0.1926 0.8997 -0.3917; -0.3917 -0.3917  0.4395];

% the number of different MS
NumOfMs= max(MsNumber); % MsNumber is a vector!
if (NumOfMs>10*NumLinks)
    warning('MATLAB:SparseMsNumberVector','Max index of linkpar.MsNumber is large
compared to number of links!')
end

switch lower(Scenario)

    case {'suburban_macro'}

        % general environment parameters for suburban macro [1, Table 5.1]
        mu_as      =  0.69 ;
        epsilon_as =  0.13 ;
        mu_ds      = -6.80 ;
        epsilon_ds =  0.288;
        sigma_sf_ave   =  8    ; % in dB

        % generate alphas, betas and gammas for all links
        abc = C*randn(3,NumLinks);

        % inter-site correlation terms for all different MSs
        gamma= bsq*randn(1,NumOfMs);    % bsq*ksi_3 for all different MSs
        gammas=gamma(MsNumber); gammas=gammas(:).';  % so that works also when
NumOfMs==1
        abc(3,:)=abc(3,:) + gammas;    % add inter-site correlation term

        sigma_ds = 10.^(epsilon_ds*abc(1,:).' + mu_ds);
        sigma_as = 10.^(epsilon_as*abc(2,:).' + mu_as);
        sigma_sf = 10.^(0.1*sigma_sf_ave*abc(3,:).');

        % output
        sigmas=[sigma_ds sigma_as sigma_sf];


    case {'urban_macro'}

        % general environment parameters for urban macro [1, Table 5.1]
        if strcmp(scmpar.BsUrbanMacroAS,'fifteen')
            mu_as      =  1.18 ;
            epsilon_as =  0.210;
        else      % Note: 8 degree angle spread is set automatically if no match to
'fifteen'
            mu_as      =  0.810;
            epsilon_as =  0.34 ;
```

```
        end
            mu_ds      = -6.18 ;
            sigma_sf_ave   =  8    ; % in dB
            epsilon_ds =  0.18 ;

        % generate alphas, betas and gammas for all links
        abc = C*randn(3,NumLinks);

        % inter-site correlation terms for all different MSs
        gamma= bsq*randn(1,NumOfMs);    % bsq*ksi_3 for all different MSs
        gammas=gamma(MsNumber); gammas=gammas(:).';     % so that works also when
NumOfMs==1
        abc(3,:)=abc(3,:) + gammas;    % add inter-site correlation term

        sigma_ds = 10.^(epsilon_ds*abc(1,:).' + mu_ds);
        sigma_as = 10.^(epsilon_as*abc(2,:).' + mu_as);
        sigma_sf = 10.^(0.1*sigma_sf_ave*abc(3,:).');

        % output
        sigmas=[sigma_ds sigma_as sigma_sf];



    case {'urban_micro'}

        if strcmpi(ScmOptions,'los')
            sigma_sf_ave = 4;    % in dB
        else    % NLOS
            sigma_sf_ave = 10;    % in dB
        end

        % inter-site correlation terms for all different MSs
        gamma_intersite= bsq*randn(1,NumOfMs);    % bsq*ksi_3 for all different MSs
        gamma_intersites=gamma_intersite(MsNumber);
gamma_intersites=gamma_intersites(:).';
        gamma= C(3,:)*randn(3,NumLinks)+ gamma_intersites;    % add inter-site
correlation term

        sigma_sf = 10.^(0.1*sigma_sf_ave*gamma);

        % output
        sigmas=[sigma_sf];

end




%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% A function that maps inputs from (-inf,inf) to (-180,180)
function y=prin_value(x)
y=mod(x,360);
y=y-360*floor(y/180);





%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% A function to generate bulk parameters for suburban and urban macro cells
% See [1, Sec.5.3.1].
function bulk_parameters=macro(scmpar,linkpar,antpar)

% extract the number of users from the first field of linkpar struct
NumLinks = length(linkpar.MsBsDistance);

% extract certain parameters from the input structs
Scenario              = scmpar.Scenario;
ScmOptions            = scmpar.ScmOptions;
N                     = scmpar.NumPaths;
M                     = scmpar.NumSubPathsPerPath;
DelaySamplingInterval = scmpar.DelaySamplingInterval;

% check that M=20
if (M ~= 20)
```

```
    M=20;
    warning('MATLAB:NumSubPathsChanged','NumSubPathsPerPath is not 20! Using
NumSubPathsPerPath=20 instead.')
end

% make sure that user-specific parameters are row vectors
ThetaBs      = linkpar.ThetaBs(:).';
ThetaMs      = linkpar.ThetaMs(:).';



% general environment parameters for suburban macro [1, Table 5.1]
switch lower(Scenario)
    case {'suburban_macro'}
        r_as      = 1.2;
        r_ds      = 1.4;
        sigma_rnd = 3;    % per-path shadowing std in dB, needed in step 5

    case {'urban_macro'}
        r_as      = 1.3;
        r_ds      = 1.7;
        sigma_rnd = 3;    % per-path shadowing std in dB, needed in step 5

end



% step 3: determine DS, AS and SF for all users
% This step takes into account channel scenario automatically
% Note: path loss is computed in step 13
sigmas   = step3(scmpar,linkpar);   % a (NumLinks x 3) matrix
sigma_ds = sigmas(:,1);
sigma_as = sigmas(:,2);
sigma_sf = sigmas(:,3);


% step 4: generate delays in a (NumLinks x N) matrix
% The unit of taus is seconds
sigma_ds     = repmat(sigma_ds,1,N);                % delay spreads for all users
taus         = sort(-r_ds*sigma_ds.*log(rand(NumLinks,N)),2);
taus_sorted  = taus - repmat(taus(:,1),1,N );       % normalize min. delay to zero
if (DelaySamplingInterval>0)
    taus_rounded=DelaySamplingInterval*floor( taus_sorted/DelaySamplingInterval + 0.5);
else
    taus_rounded=taus_sorted;
end



% step 5: determine random average powers in a (NumLinks x N) matrix
ksi    = randn(NumLinks,N)*sigma_rnd;           % per-path shadowing
Pprime = exp((1-r_ds)/r_ds*taus_sorted./sigma_ds ).*10.^(-ksi/10);
P      = Pprime./repmat(sum(Pprime,2),1,N);     % power normalization
%Psub   = repmat(P,[1 1 M])/M;                   % a (NumLinks x N x M) array



% step 6: determine AoDs
sigma_aod     = r_as*sigma_as;                             % AoD angle spreads for all
users
deltas        = abs(randn(NumLinks,N).*repmat(sigma_aod,1,N));
deltas_sorted = sort(deltas,2);
delta_aod     = sign(rand(NumLinks,N)-0.5).*deltas_sorted;  % a (NumLinks x N) matrix of
path AoDs
delta_aod     = reshape(delta_aod.',NumLinks*N,1);
delta_aod     = repmat(delta_aod,1,M).';                    % a (M x (NumLinks*N))
matrix


% step 7: associate the path delays with AoDs (a dummy step)



% step 8: determine the powers, phases and offset AoDs at the BS
% The phases are computed in step 13
```

```
aod_2deg     = [0.0894 0.2826 0.4984 0.7431 1.0257 1.3594 1.7688 2.2961 3.0389 4.3101];
% [1, Table 5.2]
delta_nm_aod = [aod_2deg; -aod_2deg];
delta_nm_aod = delta_nm_aod(:);   % this (M x 1) vector is the same for all users and
paths
delta_nm_aod = repmat(delta_nm_aod,1,N*NumLinks);  % a (NumLinks x N) matrix




% step 9: determine the AoAs
% If urban_canyon option is selected steps 9a-9d replace step 9
if (strcmp(ScmOptions,'urban_canyon')==1)

    % STEPS 9a-9d  [1, Sec. 5.5.4]
    % Note that MsDirection below effectively overrides the one given in
    % the input struct LINKPAR. Note also that MS array orientation is not
    % actually needed anywhere.
    MsDirection = 360*(rand(NumLinks,1)-0.5);     % step 9a: generate MS DoT = Street
Orientation
    %MsArrayDir  = 360*(rand(NumLinks,1)-0.5);     % step 9b: generate MS Array
orientation
    alpha       = 0.9;                            % percentage of links experiencing
urban canyon effect
    p_DoT       = 0.5;
    offset      = 180;


    % Select the links that experience urban canyon
    beta        = rand(1,NumLinks);         % step 9c
    uc_links    = find(beta<=alpha);
    nuc_links   = find(beta>alpha);         % these links do not experience the canyon

    % Generate the AoAs for the urban canyon links
    delta_aoa              = MsDirection;
    offset_links           = uc_links(find(rand(length(uc_links),1)>p_DoT));
    delta_aoa(offset_links) = MsDirection(offset_links)+offset;
    delta_aoa              = repmat(delta_aoa,1,N);

    % Generate the AoAs for non-canyon links
    P_nuc_links            = P(nuc_links,:);
    sigma_aoa_nuc_links    = 104.12*(1-exp(-0.2175*abs(10*log10(P_nuc_links))));
    delta_aoa(nuc_links,:)  = randn(length(nuc_links),N).*sigma_aoa_nuc_links;     % a
(NumLinks x N) matrix of path AoAs
    delta_aoa              = reshape(delta_aoa.',NumLinks*N,1);
    delta_aoa              = repmat(delta_aoa,1,M).';           % a (M x (NumLinks*N))
matrix


else    % Urban Canyon option not used
    % step 9
    sigma_aoa = 104.12*(1-exp(-0.2175*abs(10*log10(P))));
    delta_aoa = randn(NumLinks,N).*sigma_aoa;     % a (NumLinks x N) matrix of path AoAs
    delta_aoa = reshape(delta_aoa.',NumLinks*N,1);
    delta_aoa = repmat(delta_aoa,1,M).';          % a (M x (NumLinks*N)) matrix


end






% step 10: determine the offset AoAs at the MS
aoa_35deg    = [1.5679 4.9447 8.7224 13.0045 17.9492 23.7899 30.9538 40.1824 53.1816
75.4274];      % [1, Table 5.2]
delta_nm_aoa = [aoa_35deg; -aoa_35deg];
delta_nm_aoa = delta_nm_aoa(:);       % these are the same for all users and paths
delta_nm_aoa = repmat(delta_nm_aoa,1,N*NumLinks); % a (M x N*NumLinks) matrix





% step 11: pair AoA subpaths randomly with AoD subpaths (within a path)
```

```
[dummy h]           = sort(rand(M,N*NumLinks),1);       % create N*NumLinks random
permutations of integers [1:M]
inds                = h+repmat([1:M:M*N*NumLinks],M,1)-1;
delta_nm_aoa_paired = delta_nm_aoa(inds);    % random permutation of columns, a (M x
N*NumLinks) matrix



% step 12: determine angles depending on array orientation
ThetaBs      = repmat(ThetaBs,N,1); ThetaBs=ThetaBs(:).';
ThetaBs      = repmat(ThetaBs,M,1);      % a (M x N*NumLinks) matrix
theta_nm_aod = ThetaBs+delta_aod+delta_nm_aod;
ThetaMs      = repmat(ThetaMs,N,1); ThetaMs=ThetaMs(:).';
ThetaMs      = repmat(ThetaMs,M,1);      % a (M x N*NumLinks) matrix
theta_nm_aoa = ThetaMs + delta_aoa+delta_nm_aoa_paired;

% Values of theta_nm_aoa and theta_nm_aod may be outside (-180,180).
% This is corrected in the following.
theta_nm_aoa=prin_value(theta_nm_aoa);
theta_nm_aod=prin_value(theta_nm_aod);


% put AoDs, AoAs, and power gains into a 3D-array with dims [NumLinks N M]
theta_nm_aod=reshape(theta_nm_aod,M,N,NumLinks);
theta_nm_aod=permute(theta_nm_aod,[3 2 1]);
theta_nm_aoa=reshape(theta_nm_aoa,M,N,NumLinks);
theta_nm_aoa=permute(theta_nm_aoa,[3 2 1]);



% step 13: Path loss and shadowing
% employ the user-defined path loss model
path_losses=feval(scmpar.PathLossModel,scmpar,linkpar);
path_losses=10.^(-path_losses(:)/10);    % a (NumLinks x 1) vector


% optional steps for polarized arrays and output generation
if (strcmp(lower(ScmOptions),'polarized')==1)   % [1, Sec. 5.5.1]
    % Step 13 - dummy step

    % Step 14 - dummy step

    % Step 15 - generates random phases
    phi = 360*rand(NumLinks,4,N,M);      % random phases for all users: [NumLinks pol
path subpath]

    % Step 16 - dummy step

    % Step 17 - generate XPD ratios
    A           = 0.34*10*log10(P)+7.2; % in dB
    B           = 5.5;  % in dB
    xpd_in_db   = zeros(NumLinks,2,N);

    xpd_in_db(:,1,:)=A+B*randn(size(A));   % V-to-H coupling
    xpd_in_db(:,2,:)=A+B*randn(size(A));   % H-to-V coupling

    xpd=10.^(xpd_in_db/10);                % size()=[NumLinks pol N], xpd(:,1,:) is V-
to-H coupling

    % output
    bulk_parameters=struct( 'delays',taus_rounded,...
                            'path_powers',P,...            % before:
'subpath_powers',Psub,...
                            'aods',theta_nm_aod,...         % in degrees
                            'aoas',theta_nm_aoa,...         % in degrees
                            'subpath_phases',phi,...        % in degrees
                            'xpd',xpd,...                   % in linear scale
                            'path_losses',path_losses,...   % in linear scale
                            'shadow_fading',sigma_sf);      % in linear scale

else
    phi= 360*rand(NumLinks,N,M);        % random phases for all users

    % output
    bulk_parameters=struct( 'delays',taus_rounded,...
```

```
                            'path_powers',P,...             % before:
'subpath_powers',Psub,...
                            'aods',theta_nm_aod,...         % in degrees
                            'aoas',theta_nm_aoa,...         % in degrees
                            'subpath_phases',phi,...        % in degrees
                            'path_losses',path_losses,...   % in linear scale
                            'shadow_fading',sigma_sf);      % in linear scale

end




% end of function 'macro'






%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% A function to generate urban micro cell parameters
function bulk_parameters=micro(scmpar,linkpar,antpar)


% extract the number of users from the first field of linkpar struct
MsBsDistance    =linkpar.MsBsDistance;
NumLinks        =length(MsBsDistance);

% extract certain parameters from the input structs
ScmOptions              =scmpar.ScmOptions;
N                       =scmpar.NumPaths;
M                       =scmpar.NumSubPathsPerPath;
DelaySamplingInterval   =scmpar.DelaySamplingInterval;


% check that M=20
if (M ~= 20)
    M=20;
    warning('MATLAB:NumSubPathsChanged','NumSubPaths is not 20! Using NumSubPaths=20
instead.')
end


% make sure that user-specific parameters are row vectors
MsBsDistance=linkpar.MsBsDistance(:).';
ThetaBs     =linkpar.ThetaBs(:).';
ThetaMs     =linkpar.ThetaMs(:).';


% general environment parameters for urban micro [1, Table 5.1]
max_ds      =1.2e-6;  % maximum excess delay in seconds
max_aod     =40;     % maximum AoD angle in degrees
sigma_rnd   =3;    % per-path shadowing std in dB, needed in step 6


% step 3: determine DS, AS and SF for all users
% This step takes into account channel scenario automatically
% path loss is computed in step 13
sigma_sf=step3(scmpar,linkpar);   % a (NumLinks x 1) matrix
sigma_sf=sigma_sf(:);


% step 4: generate delays in a (NumLinks x N) matrix
% The unit of taus is seconds
taus=max_ds*rand(NumLinks,N);  % path delays for all users



% step 5
taus=sort(taus,2);
taus=taus-repmat(taus(:,1),1,N );       % normalize min. delay to zero
if (DelaySamplingInterval>0)
    taus_rounded=DelaySamplingInterval*floor( taus/DelaySamplingInterval + 0.5);
```

```
else
    taus_rounded=taus;
end


% step 6: determine random average powers in a (NumLinks x N) matrix
z=randn(NumLinks,N)*sigma_rnd;       % per-path shadowing
Pprime=10.^( -(taus/1e-6 + z/10) );
P=Pprime./repmat(sum(Pprime,2),1,N);    % power normalization
%Psub=repmat(P,[1 1 M])/M;       % a (NumLinks x N x M) array of subpath powers for all
users


% step 7: determine AoDs
delta_aod=2*max_aod*(rand(NumLinks,N)-0.5);
delta_aod=reshape(delta_aod.',NumLinks*N,1);
delta_aod=repmat(delta_aod,1,M).';      % a (M x (NumLinks*N)) matrix


% step 8: associate the path delays with AoDs (a dummy step)


% step 9: determine the powers, phases and offset AoDs at the BS
aod_5deg=[0.2236 0.7064 1.2461 1.8578 2.5642 3.3986 4.4220 5.7403 7.5974 10.7753]; % [1,
Table 5.2]
delta_nm_aod = [aod_5deg; -aod_5deg];
delta_nm_aod=delta_nm_aod(:);   % this (M x 1) vector is the same for all users and
paths
delta_nm_aod=repmat(delta_nm_aod,1,N*NumLinks);  % a (NumLinks x N) matrix




% step 10: determine the AoAs
% If urban_canyon option is selected steps 10a-10d replace step 10
if (strcmp(ScmOptions,'urban_canyon')==1)

    % STEPS 10a-10d  [1, Sec. 5.5.4]
    % Note that MsDirection below effectively overrides the one given in
    % the input struct LINKPAR. Note also that MS array orientation is not
    % actually needed anywhere.
    MsDirection = 360*(rand(NumLinks,1)-0.5);     % step 10a: generate MS DoT = Street
Orientation
    %MsArrayDir  = 360*(rand(NumLinks,1)-0.5);     % step 10b: generate MS Array
orientation
    alpha       = 0.9;                            % percentage of links experiencing
urban canyon effect
    p_DoT       = 0.5;
    offset      = 180;


    % Select the links that experience urban canyon
    beta        = rand(1,NumLinks);         % step 10c
    uc_links    = find(beta<=alpha);
    nuc_links   = find(beta>alpha);         % these links do not experience the canyon

    % Generate the AoAs for the urban canyon links
    delta_aoa              =MsDirection;
    offset_links           =uc_links(find(rand(length(uc_links),1)>p_DoT));
    delta_aoa(offset_links) =MsDirection(offset_links)+offset;
    delta_aoa              =repmat(delta_aoa,1,N);

    % Generate the AoAs for non-canyon links
    P_nuc_links            = P(nuc_links,:);
    sigma_aoa_nuc_links     = 104.12*(1-exp(-0.265*abs(10*log10(P_nuc_links))));
    delta_aoa(nuc_links,:)  = randn(length(nuc_links),N).*sigma_aoa_nuc_links;     % a
(NumLinks x N) matrix of path AoAs
    delta_aoa              = reshape(delta_aoa.',NumLinks*N,1);
    delta_aoa              = repmat(delta_aoa,1,M).';          % a (M x (NumLinks*N))
matrix
```

```
else    % Urban Canyon option not used
    % step 10
    sigma_aoa = 104.12*(1-exp(-0.265*abs(10*log10(P))));
    delta_aoa = randn(NumLinks,N).*sigma_aoa;     % a (NumLinks x N) matrix of path AoAs
    delta_aoa = reshape(delta_aoa.',NumLinks*N,1);
    delta_aoa = repmat(delta_aoa,1,M).';          % a (M x (NumLinks*N)) matrix


end







% step 11: determine the offset AoAs at the MS
aoa_35deg       =[1.5679 4.9447 8.7224 13.0045 17.9492 23.7899 30.9538 40.1824 53.1816
75.4274];      % [1, Table 5.2]
delta_nm_aoa    = [aoa_35deg; -aoa_35deg];
delta_nm_aoa    =delta_nm_aoa(:);       % these are the same for all users and paths
delta_nm_aoa    =repmat(delta_nm_aoa,1,N*NumLinks); % a (M x N*NumLinks) matrix



% step 12: pair AoA subpaths randomly with AoD subpaths (within a path)
[dummy h]           = sort(rand(M,N*NumLinks));       % create N*NumLinks random
permutations of integers [1:M]
inds                =h+repmat([1:M:M*N*NumLinks],M,1)-1;
delta_nm_aoa_paired =delta_nm_aoa(inds);    % random permutation of columns, a (M x
N*NumLinks) matrix




% step 13: determine the antenna gains of BS and MS
% determine angles depending on array orientation
ThetaBs     =repmat(ThetaBs,N,1); ThetaBs=ThetaBs(:).';
ThetaBs     =repmat(ThetaBs,M,1);      % a (M x N*NumLinks) matrix
theta_nm_aod=ThetaBs+delta_aod+delta_nm_aod;
ThetaMs     =repmat(ThetaMs,N,1); ThetaMs=ThetaMs(:).';
ThetaMs     =repmat(ThetaMs,M,1);      % a (M x N*NumLinks) matrix
theta_nm_aoa= ThetaMs + delta_aoa+delta_nm_aoa_paired;

% Values of theta_nm_aoa and theta_nm_aod may be outside (-180,180).
% This is corrected in the following.
theta_nm_aoa=prin_value(theta_nm_aoa);
theta_nm_aod=prin_value(theta_nm_aod);

% put AoDs, AoAs, and power gains into a 3D-array with dims [NumLinks N M]
theta_nm_aod=reshape(theta_nm_aod,M,N,NumLinks);
theta_nm_aod=permute(theta_nm_aod,[3 2 1]);
theta_nm_aoa=reshape(theta_nm_aoa,M,N,NumLinks);
theta_nm_aoa=permute(theta_nm_aoa,[3 2 1]);


% employ the user-defined path loss model
path_losses=feval(scmpar.PathLossModel,scmpar,linkpar);
path_losses=10.^(-path_losses(:)/10);    % a (NumLinks x 1) vector



% optional steps
switch (lower(ScmOptions))   % [1, Sec. 5.5.1]

    case ('los')
        LOS_probability=max([zeros(size(MsBsDistance)); (300-MsBsDistance)./300]);
        prob=rand(size(LOS_probability));
        LOS_probability=LOS_probability.*(prob<=LOS_probability);

        % calculate K factors of the links --  K factor > 0 only if LOS_probability>0
        K_factors=10.^((13-0.03*MsBsDistance)/10); % [1, Sec. 5.5.3]
        K_factors=K_factors.*(LOS_probability~=0);  % in linear scale
        K_factors=K_factors(:);
        % set the LOS phase randomly
```

```
        Phi_LOS=360*(rand(NumLinks,1)-0.5);

        phi= 360*rand(NumLinks,N,M);        % random phases for all users


        % output
        bulk_parameters=struct( 'delays',taus_rounded,...
                                'path_powers',P,...                 % before:
'subpath_powers',Psub,...
                                'aods',theta_nm_aod,...
                                'aoas',theta_nm_aoa,...
                                'subpath_phases',phi,...
                                'K_factors',K_factors,...           % in linear scale
                                'Phi_LOS',Phi_LOS,...               % phases for LOS
paths, in degrees
                                'path_losses',path_losses,...       % in linear scale
                                'shadow_fading',sigma_sf);          % in linear scale



    case ('polarized')
        % Step 13 - dummy step

        % Step 14 - dummy step

        % Step 15 - generates random phases
        phi= 360*rand(NumLinks,4,N,M);      % random phases for all users: [NumLinks pol
path subpath]

        % Step 16 - dummy step

        % Step 17 - generate XPD ratios
        A=8;        % in dB
        B=8;        % in dB
        xpd_in_db=zeros(NumLinks,2,N);

        xpd_in_db(:,1,:)=A+B*randn(NumLinks,N);    % V-to-H coupling
        xpd_in_db(:,2,:)=A+B*randn(NumLinks,N);    % H-to-V coupling

        xpd=10.^(xpd_in_db/10);                     % size()=[NumLinks pol N],
xpd(:,1,:) is V-to-H coupling

        % output
        bulk_parameters=struct( 'delays',taus_rounded,...
                                'path_powers',P,...             % before:
'subpath_powers',Psub,...
                                'aods',theta_nm_aod,...         % in degrees
                                'aoas',theta_nm_aoa,...         % in degrees
                                'subpath_phases',phi,...        % in degrees
                                'xpd',xpd,...                   % in linear scale
                                'path_losses',path_losses,...   % in linear scale
                                'shadow_fading',sigma_sf);      % in linear scale




    case {'none','urban_canyon'}
        phi= 360*rand(NumLinks,N,M);       % random phases for all users

        % output
        bulk_parameters=struct( 'delays',taus_rounded,...
                                'path_powers',P,...             % before:
'subpath_powers',Psub,...
                                'aods',theta_nm_aod,...
                                'aoas',theta_nm_aoa,...
                                'subpath_phases',phi,...
                                'path_losses',path_losses,...   % in dB
                                'shadow_fading',sigma_sf);      % in linear scale

end



% end of function 'micro'
```

ds.m

```
function [sigma_DS, excess_delay]=ds(tau,P)
%DS RMS delay spread
%   SIGMA_DS=DS(TAU,P) returns the rms delay spread SIGMA_DS. TAU are the
%   delays of paths and P are the powers of the corresponding paths. If P
%   is a matrix, SIGMA_DS is computed for each column; in this case TAU can
%   be either a matrix with SIZE(TAU)=SIZE(P) or a column vector with the
%   same number of rows as P. If TAU is a column vector the same delays are
%   used for each column of P.
%
%   [SIGMA_DS, ED]=DS(TAU,P) returns also the excess delay in ED.
%
%   Note that if P is an impulse response, SIGMA_DS is its sample rms delay
%   spread.

%   Author: Jari Salo (HUT)
%   $Revision: 0.2 $  $Date: July 5, 2006$


if (ndims(tau) > 2 || ndims(P) > 2)
    error('Input arguments must be vectors or matrices!')
end


% if P is a vector
if ( min(size(P))==1 )
    P=P(:);     % make it column vector
end

if (min(size(tau))==1)  % if tau is a vector
    tau=tau(:);
    if length(tau) ~= size(P,1)
        error('Number of delays is not equal to number of powers!')
    end
    tau=repmat(tau,1,size(P,2))-min(tau);
else    % tau is a matrix
    if any( size(P)-size(tau) )
        error('Input argument size mismatch!')
    end
    tau=tau-repmat(min(tau),size(tau,1),1);
end


Dvec=sum(tau.*P)./(sum(P)+realmin);     % + realmin to avoid division by zero
D=repmat(Dvec,size(tau,1),1);

% compute std of delay spread
sigma_DS=sqrt( sum((tau-D).^2.*P)./(sum(P)+realmin ));


if (nargout>1)
    excess_delay=max(tau);
end
```

cas.m

```
function [sigma_AS]=cas(theta,P,units)
%CAS Circular angle spread (3GPP TR 25.996)
%   SIGMA_AS=CAS(THETA,P) returns the circular angle spread SIGMA_AS as
%   defined in Annex A of 3GPP TR 25.996 v6.1.0. THETA are the angles (in
%   radians) of paths and P are powers  of the paths. THETA and P must be
%   of same size and the (i,j)th element of P must be the power
%   corresponding to the (i,j)th  angle. In 3GPP notation both THETA and P
%   are N X M matrices,  where N is the number of paths and M is the number
%   of subpaths.
%
%   With SIGMA_AS=CAS(THETA,P,'deg') input and output angles are given in
%   degrees.

%   Author: Jari Salo (HUT)
%   $Revision: 0.2 $  $Date: July 5, 2006$
```

```
% check that input args have same size
if (any(size(theta)-size(P))==1)
    error('cas: Input argument size mismatch!')
end

% check if both inputs are scalar
if isscalar(theta) && isscalar(P)
    sigma_AS=0;
    return
end


deg_flag=0; % unit is radians
if (nargin>2)
    if (strcmp(lower(units),'deg')==1)
        theta=theta/180*pi;     % computation is in radians
        deg_flag=1; % unit is degrees
    end
end


% vectorize inputs
P=P(:);
theta=theta(:);

len_theta=length(theta);

delta=linspace(-pi,pi);     % a 100-point grid for minimization
delta_mat=repmat(delta,len_theta,1);
theta_mat=repmat(theta,1,length(delta));
theta_mat=prin_value(theta_mat+delta_mat);
P_mat=repmat(P,1,length(delta));

% mean values over the grid
mu_thetas=sum( theta_mat.*P_mat )./(sum(P_mat)+realmin);

% demeaned angles
theta_nm_mus=theta_mat-repmat(mu_thetas,len_theta,1);
theta_nm_mus=prin_value(theta_nm_mus);
cas_vec= sqrt(sum(theta_nm_mus.^2.*P_mat)./( sum(P_mat)+realmin) );

sigma_AS=min(cas_vec);

if (deg_flag==1)    % map back to degrees
    sigma_AS=sigma_AS/pi*180;
end



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% A function to map inputs from (-inf,inf) to (-pi,pi)
function y=prin_value(x)
y=mod(x,2*pi);
y=y-2*pi*floor(y/pi);
```

dipole.m

```
function pattern=dipole(varargin)
%DIPOLE Field pattern of half wavelength dipole
%   PAT=DIPOLE(AZ) returns the azimuth field pattern
%   at angles given in AZ (degrees).
%
%   PAT is a 3D-array with dimensions [2 1 LENGTH(AZ)].
%   The first two dimensions are the V and H
%   polarizations, respectively.
%
%   PAT=DIPOLE(AZ,SLANT) gives the pattern of a
%   slanted dipole. The slant angle is defined as
%   the counter clock-wise angle (in degrees) seen
%   from the front of the dipole.
%
%   Currently elevation is not supported.
%
```

```
%   Example: To create a 2-element BS array with
%   45 degrees slanted dipoles:
%       g=zeros(2,2,1,100); antpar=antparset;
%       az=linspace(-180,180);
%       g(1,:,:,:)=dipole(az,45);
%       g(2,:,:,:)=dipole(az,-45);
%       antpar.BsGainPattern=g;
%       antpar.BsGainAnglesAz=az;
%
%   See also ANTPARSET.

%   Author: Jari Salo (HUT)
%   $Revision: 0.1 $  $Date: July 22, 2004$


az=varargin{1};

if (nargin>1)
    slant=varargin{2};
    slant=-slant/180*pi;    % change sign
else
    slant=0;
end

% put all angles to radians
az=az/180*pi;

siz_az=size(az);        % elevation has same size
az_vec=az(:);

% assume elevation is zero
[X Y Z]=sph2cart(az_vec, zeros(size(az_vec)), repmat(1,size(az_vec)));

% rotation matrix in cartesian coordinates
R = [1 0 0; 0 cos(slant) -sin(slant); 0 sin(slant) cos(slant) ];
XYZr=R*[X.'; Y.'; Z.'];
[az el r]=cart2sph(XYZr(1,:), XYZr(2,:), XYZr(3,:));
el=reshape(el(:),siz_az);

% our coordinate system has elevation 90 deg to the zenith
% while the standard dipole formula has zero angle in zenith
offset=-pi/2;
el=-(el+offset);   % elevation is now from -90 to 90 (directly below to zenith)


% ideal pattern of a slanted dipole
% the dipole pattern becomes singular at {0,180} degrees elevation
tol=1e6*eps;
I1=find(abs(el)<tol);
I2=find(abs(el-pi)<tol);
I=[I1(:); I2(:)];   % set these indices to zero
patternV=zeros(size(el));
patternH=zeros(size(el));
patternV(I)=0;
patternH(I)=0;
Inot=setdiff([1:numel(el)],I);
patternV(Inot)=sqrt(1.64)*abs(cos(pi/2*cos(el(Inot)))./sin(el(Inot)))*cos(slant);
patternH(Inot)=sqrt(1.64)*abs(cos(pi/2*cos(el(Inot)))./sin(el(Inot)))*sin(slant);

pattern=zeros(2,1,numel(el));
pattern(1,1,:)=patternV;
pattern(2,1,:)=patternH;
```

9. References

[2.1] David Gesbert, Mansoor Shafi, Da-shan Shiu, Peter J. Smith, and Ayman Naguib, “From theory to practice: an overview of MIMO space-time coded wireless systems,” IEEE Journal on Selected Areas in Communications, vol. 21, no. 3, pp. 281-302, Apr. 2003.

[2.2] Moray Rummey, Janine Whiteacre. “Analyze Antenna Approaches for LTE Wireless Systems”. August 2008.

[2.3] P. Almers, E. Bonek, A. Burr et al. “Survey of Channel and Radio Propagation Models for Wireless MIMO Systems” , Nov. 2006

[2.4] Ernst Bonek . “On MIMO Channel Models for LTE” Institut für Nachrichtentechnik und Hochfrequenztechnik Technische Universität Wien and Forschungszentrum Telekommunikation Wien. 16 January 2009.

[3.1] Baum, D.S.; Hansen, J.; Salo, J. “An interim channel model for beyond-3G systems: extending the 3GPP spatial channel model (SCM)”. Vehicular Technology Conference, 2005. VTC 2005-Spring. 2005 IEEE 61st

[3.2] 3GPP & 3GPP2 Spatial Channel Model AHG, “Spatial Channel Model Text Description”, SCM Text V6.0. December 2002.

[3.3] IEEE Working Group 802.20. “Channel Models for IEEE 802.20 MBWA System Simulations”. November, 2003.

[4.1] Walter Fisher. “Digital Television: a practical guide for engineers”. 2004

[4.2] LG HDTV Guide. “The Future of TV” 2005

[4.3] Chi-Fang Huang, Yi-Min Tsai et al. “Model analysis for the radio channel of DVB-T indoor reception in a Single Frequency Network” *Graduate Institute of Communication Engineering.* Tatung University. Taipei, Taiwan.

[4.4] Martin Jacklin and Peter MacAvock (DVB Project Office) with Andrew Oliphant (BBC Research & Development). “DVB comments on the DVB-T Digital Terrestrial Broadcasting Standard”

[4.5] Springer Berlin Heidelberg. “Coded Orthogonal Frequency Division Multiplex (COFDM)” 2008

[4.6] 3GPP. “Agilent 3GPP Long Term Evolution: System Overview, Product Development, and Test Challenges”

[4.7] MWG/AWG; WIMAX Forum. "A Comparative Analysis of Spectrum Alternatives for WiMAX™ Networks with Deployment Scenarios Based on the U.S. 700 MHz Band" June 2008

[4.8] Mignone, V.; Morello, A.; Visintin, M. "An advanced algorithm for improving DVB-T coverage in SFN" Broadcasting Convention, 1997.

[4.9] Mercedes Sánchez Varela, Manuel García Sánchez, Lloyd Lukama, and David J. Edwards. "Spatial Diversity Analysis for Digital TV Systems", IEEE transactions on broadcasting, vol. 47, no. 3, september 2001

[4.10] Daniel Larson. "Analysis of channel estimation methods for OFDMA". Master of Science Thesis. Stockholm, Sweden 2006-12-19

[4.11] Cristina Rico García, Andreas Lehner, Thomas Strang and Korbinian Frank. "Channel Model for Train to Train Communication using the 400 MHz Band". German Aerospace Center. Institute of Communications and Navigation

[4.12] "Overcoming the Challenges of WiMAX Deployment in 700MHz Band"

[4.13] Chi-Fang Huang and Li Chen. "A compact omni-directional UHF antenna for DTV broadcasting reception". *Graduate Institute of Communication Engineering.* Tatung University. Taipei, Taiwan.

[5.1] J. Salo, G. Del Galdo, J. Salmi, P. Kyösti, M. Milojevic, D. Laselva, and C. Schneider. (2005, Jan). MATLAB implementation of the 3GPP Spatial Channel Model (3GPP TR 25.996) [Online].

[5.2] "Spatial channel model for Multiple Input Multiple Output (MIMO) simulations", 3GPP TR 25.996 V6.1.0 (2003-09)