M-Sequences Related to the Multifocal Electroretinogram: Identification of Appropriate Primitive Polynomials to Avoid Cross-Contamination in Multifocal Electroretinogram Responses

Jillian M. Ireland

Department of Clinical Physics & BioEngineering
ElectroDiagnostic Imaging Unit
Tennent Institute of Ophthalmology
Gartnavel General Hospital
Glasgow G12 OYN

&

Department of Mathematics
University of Glasgow
University Gardens
Glasgow G12 8QW

Thesis submitted to The Faculty of Medicine for the
Degree of MSc (Med. Sci.) by Research

October 2001

To: Rhona O'Reilly

The research from this thesis has been presented both verbally and in writing:

PUBLICATIONS:

"Identification of Appropriate Primitive Polynomials to Avoid Cross-Contamination in Multifocal Electroretinogram Responses"

To Appear in IEE Medical & Biological Computation & Engineering, May 2002

PRESENTATIONS:

15th Feb 2001, *"The Senses Looking Into Sight"*
2nd Year Postgraduate Seminar,
University of Glasgow.

30th Mar 2001, *"Identification of Appropriate Primitive Polynomials to Avoid Cross Contamination in Multifocal Electroretinogran Responses"*
Research Seminar, Gartnavel General Hospital

5th Apr 2001, *"Functional Imaging of the Visual System"*
Postgraduate Combinatorics Conference,
University of Reading.

21st June 2001, *"M-Sequences related to Multifocal Electroretinography"*
EMS Postgraduate Students Meeting,
The Burn, near Montrose.

16th Oct 2001, "*Insight into Pseudo-Random Binary Sequences*"
Research Seminar, University of Stirling.

ADDITIONAL CONFERENCES ATTENDED:

Dec 1999 "Cryptography and Coding"
7th IMA International Conference, Cirencester, UK.

Apr 2001 "British Mathematical Colloquium"
University of Glasgow, UK.

Abstract

The basis of multifocal ERG is the use of a decimated m-sequence for simultaneous and independent stimulation of many areas of the visual pathway. The purpose of this thesis is to investigate the effects of cross contamination from higher orders of the response.

To examine the effects of cross contamination a series of primitive polynomials were found by constructing finite fields. The first order ERG response is formed by cross correlating the m-sequence with the physiological response. A second order response is formed by investigating particular flash sequences of the stimulation sequence and is formed by cross correlation of a second order m-sequence with the physiological response. Zech Logarithms were used to identify cross contamination between the various first and second order sequences. Tables of good and bad primitive polynomials were constructed for degree 12 to degree 16 and the effects of window length and decimation length examined. If we decimate the sequence into 128 areas, and look at a window of length 16, cross-contamination occurs in all sequences generated from primitive polynomials of degree less than or equal 12, but only 26% in the case of degree 14, and 5.6% for degree 16. Finally, selected good and bad primitive polynomials were used to generate decimated m-sequences for a multifocal electrophysiological experiment to demonstrate the practical effects of cross – contamination.

Trace arrays showing uncontaminated discreet physiological responses from 61 individual elements were recorded using the example good primitive polynomial whereas additional waveforms were present on the trace array when the same experiment was repeated with a bad primitive polynomial.

The use of finite field theory to generate primitive polynomials and zech logorithm analysis enables us to predict which primitive polynomials are suitable for m-sequence generation for multifocal electroretinography. Practical investigations support the theoretical analysis. This has important implications for developers of multifocal electrophysiology systems.

Acknowledgements

I am grateful to The W.H. Ross Foundation for financially supporting this research. I extend my thanks to Professor A. T. Elliott, Head of Clinical Physics & BioEngineering and to Professor D. R. Fearn, Head of Mathematics for allowing me to be a research student in their respective departments at the University of Glasgow.

I would like to express my sincere thanks to my Mathematical Advisor Dr Ian Anderson - for his valuable advice, many comments and diligent reading of this thesis, for which I am very grateful.

I also extend my thanks to Dr Mohan Nair for his support and encouragement.

Dr Kenny Paterson and Dr Neil Sloane who were always on hand for advice if needed.

Dr D Keating at Gartnavel Hospital and Dr S Hoggar at Glasgow University. Dr Aled Evans at the Southern General Hospital for his advice and to the research workers in the Electrodiagnostic Imaging Unit - Joanne McDonagh, Sarah Muscat and David Nichol.

Also, to Hataikan, Stephen, Joe, Martin, Halis, Sophie, Tom, Alistair, Gennaro, Mohammad and all my fellow Postgraduate Students in the Mathematics Department at Glasgow University, past and present.

Finally, and most importantly, my thanks go to my family for their unfailing support and encouragement.

Jillian M Ireland
Glasgow 2001

"Nil Satis Nisi Optimum"

There are certain privileges of a writer,
the benefit whereof, I hope, there will be no reason to doubt;
Particularly, that where I am not understood, it shall be concluded,
that something very useful and profound is couched underneath.

- Jonathan Swift
(Tale of a Tub, preface1704)

Contents

List of Figure Captions:

Chapter 1

EYE ANATOMY AND PHYSIOLOGY

In this chapter, the primate visual system will be introduced.

1.1 ANATOMY OF THE EYE

"EYE -the organ of sight of animals; the ability to see; sense of vision"

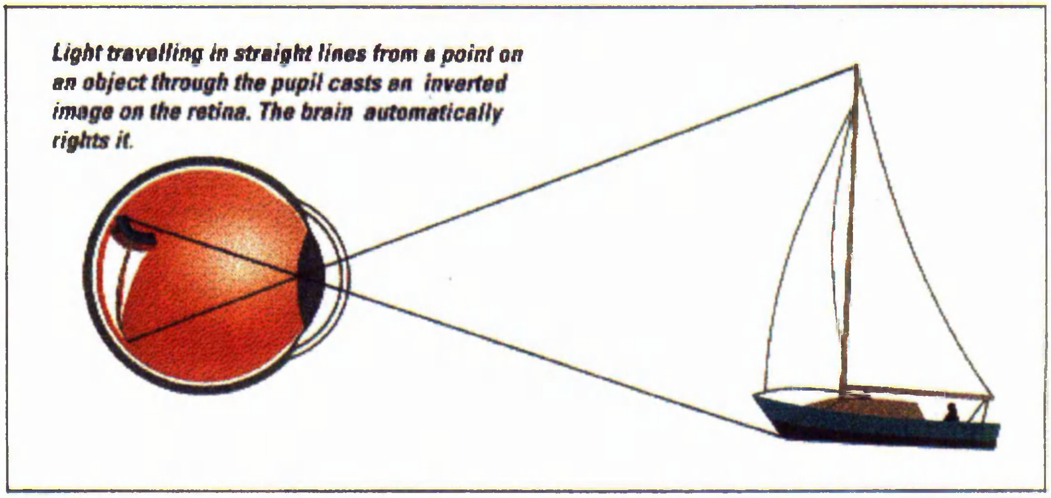
- extract from Collins English Dictionary

A large amount of information about our surroundings comes to us through our eyes. We gain a panoramic view of the world due to their position and mobility. The structure of the eye is designed to supply us with information on depth, distance, dimension, and movement. The collective function of the non-retinal parts of the eye is to keep a focused, clear image of the outside world anchored on two retinae. The retina is the sense organ of the eye. It allows us to see under a wide range of conditions. It discriminates wavelength allowing us to see colour, and provides a precision sufficient for us to detect a human hair or speck of dust from a few yards away.

It has often been said that looking at a camera can help give an understanding of how the eye works. The front part of the eye works as a lens, like the glass lens at the front of the camera; the pupil (dark part in middle of eye), acts like the aperture behind the camera lens, opening wider or narrowing down to control the amount of light that enters the eye; and the retina (inner lining of the eye), is like the film inside the camera, where the image or picture is focused.

The eye is much more complicated than the camera. Humans and animals have the ability to make sense of, and act upon, the ever-changing information that comes from the light acting on their retinas - this is because the eyes are connected to the brain. On the other hand, cameras just allow images to be recorded on film.

Each eye is connected to the brain by the Optic Nerve, and messages are transmitted along this nerve from the retina to the brain. The function of the retina is to convert light photons into electrical impulses, and these impulses are then interpreted by the brain to provide visual perception. Our brain enables us to see the right way up. As light passes through the lens, the image is inverted. With a camera, the image is actually upside down, although we see it the correct way round through the viewfinder. Figure 1.1 shows the inverted image on the retina. The brain then 'reads' the image, and transforms it.



Light travelling in straight lines from a point on an object through the pupil casts an inverted image on the retina. The brain automatically rights it.

Figure 1.1 *Inverted Image on Retina*

The human eye is spherical (approximately 1 inch in diameter). Figure 1.2, overleaf, shows the structure of the eye.

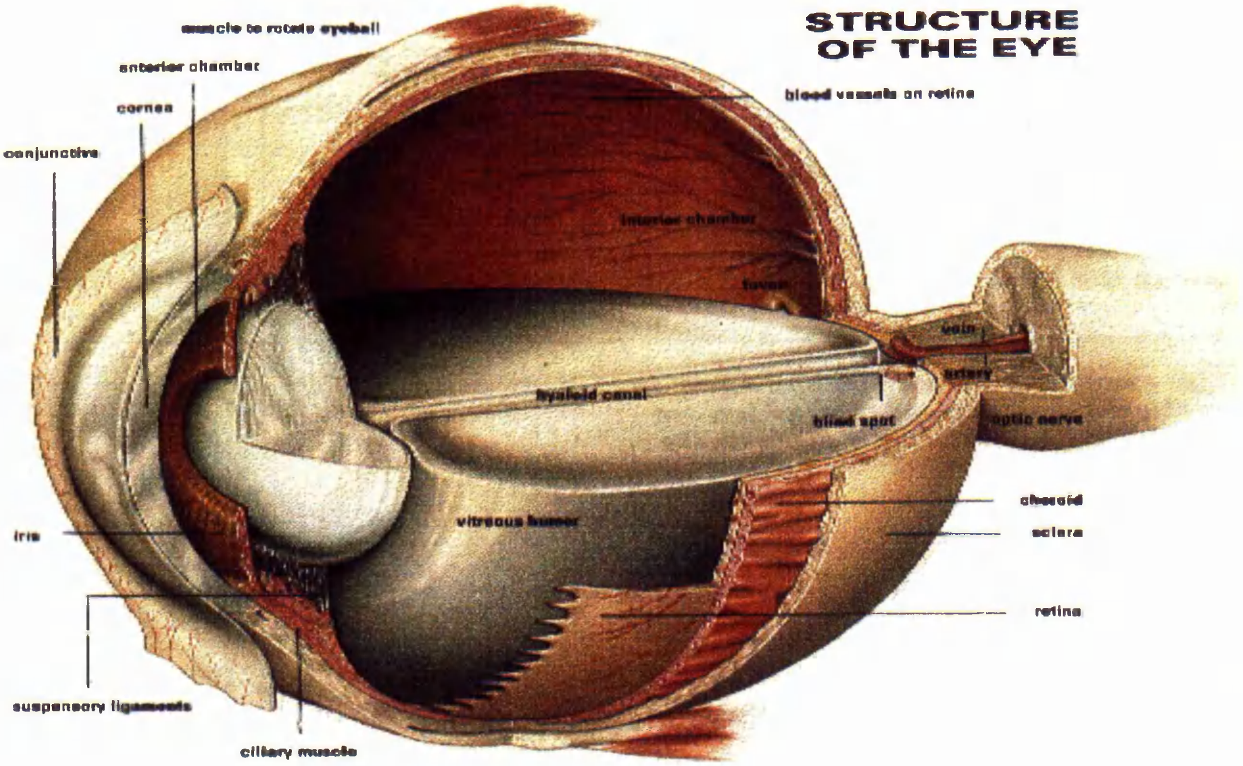
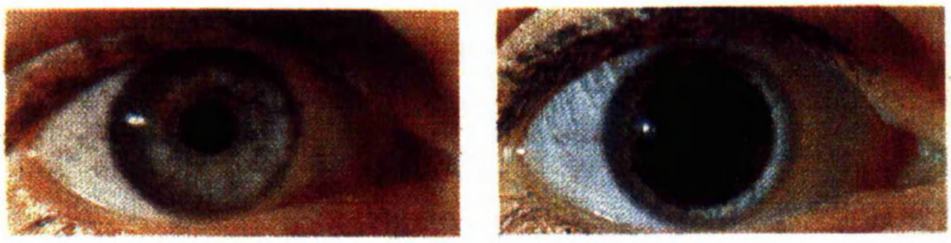


Figure 1.2 *Structure of the Eye*

Light is focused on the retina by the cornea and lens. The **cornea** is the transparent window on the front of the eye. It has a small radius of curvature, therefore, it "bulges" forward. The substance of the cornea is stroma, which is formed by a latice of cologen fibres. Its function is to be the principal refracted medium of the eye, and the cornea accounts for seventy percent of total refracted power. It does not contain any blood vessels, which isolates it from the immune system, thus, damaged corneas can be transplanted without rejection. The cornea is actually a transparent part of the **sclera.** The sclera extends all the way round the eye, and is lined with microscopically thin layers of tissue within which is a layer of fibrous tissue or stroma. These layers are known as the **choroid**, and this has a network of tiny blood vessels that supply nutrients to the eye. At the front of the eye, the choroid becomes thicker, and more complicated, and has its own name, the **ciliary body**. Round the rest of the eye, it is lined on the inside by the retina.

The coloured part of the eye is known as the **iris**, and this is a muscular disc with a hole at its centre, see Figure 1.3. The hole is the apparently black **pupil**, and this is made smaller or larger by the action of the iris.

Figure 1.3 *The Iris*



THE IRIS

The pupil, the hole in the middle of the iris, allows Light to enter the eye. The amount of light admitted is governed by the iris. In bright light the iris contracts and the pupil becomes very small, allowing only a little light to enter.

The iris relaxes in dim light, allowing the pupil to expand and let as much light as possible into the eye. The individual is usually unaware of this process.

Behind the iris is the **lens**, set within the ciliary body of the choroid, and held in place by a network of fibres, known as the suspensory ligaments. The **lens** is bi-convex. It consists of transparent Epithelial cells in concentric layers (like an onion). The lens is held in tension by a circular ligament, called zonula, which connects it to a circular ring of muscle, called the ciliary muscle. The function of the lens is to vary the power of the eye. When a near object is viewed, the ciliary muscle contracts to relax the zonule, and allows the lens to bulge forward because of its natural elasticity.

Between the cornea and iris, exists a chamber of fluid, the **anterior (outer) chamber**. The **posterior chamber** is located between the iris, zonule fibres, and lens. Both of these chambers are filled with **aqueous (water-like) humour**. Fresh aqueous humour is constantly being produced, and the excess drains into the cornea, helping to keep it clear and infection-free. The **vitreous chamber**, between the lens and retina, is filled with a more viscous humour, the **vitreous (glass-like) humour**. This is a clear, jelly-like substance, through which the light which has been bent, or refracted, by the cornea and the lens, passes before reaching the retina.

The function of the **retina** is to convert light photons into electrical impulses, which are then transmitted to the brain along the optic nerve. It is the sense organ of the eye. The retina is a circular disk, which measures approximately 42mm in diameter

Figure 1.4, is a fundus[1] image of the human retina, which can be viewed through the pupil using an opthalmoscope. The most notable feature is the optic disc (pinky, yellow disk on nasal side of fundus). This is where the optic nerve fibres leave, on the way to the brain. Fanning out from the optic disc, are the retinal arteries, which supply the retina. Two and a half disc diameters to the left of the optic disc, the blood vessel-free redish spot known as the fovea can be seen. This is in the centre of the area known as the macula.

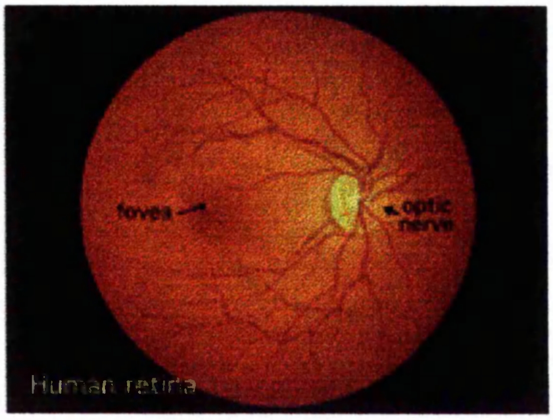


Figure 1.4 *Fundus Image of Human Retina*

[1] Fundus - *Anat.* the base of an organ, or the part farthest away from its opening.

The central retina is the circular field 6mm around the fovea, and beyond this is the peripheral retina stretching to the ora serrata, 21mm from the center of the optic disc. The retina is held in place by the jelly-like mass of vitreous humor, and any change in this vitreous humor leads to the detachment of the retina. The retina is a very complex structure, as can be seen from Figure 1.5.

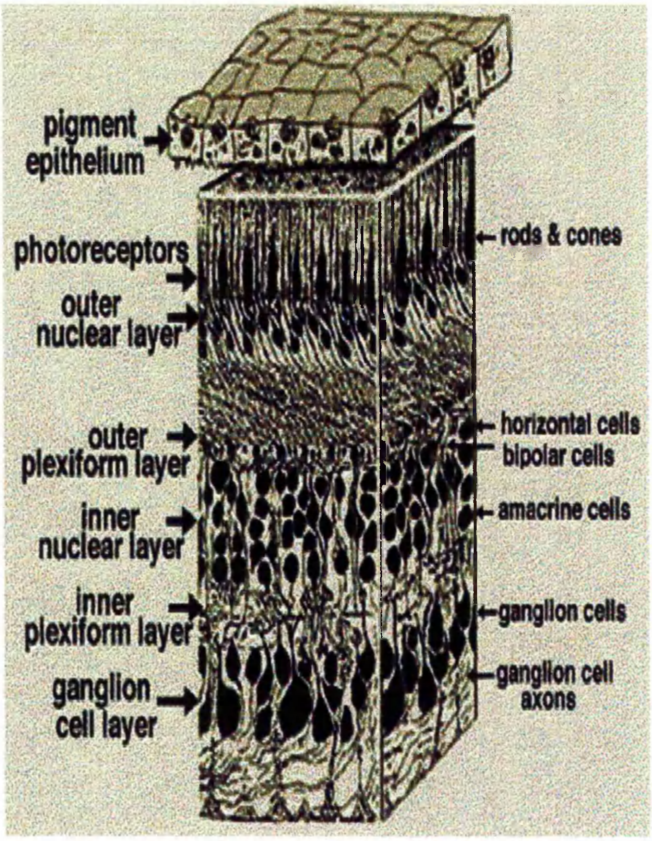


Figure 1.5: *Cross-Section of the Human Retina.*

Figure 1.6 illustrates a simplistic view of the passage of light through the retina.

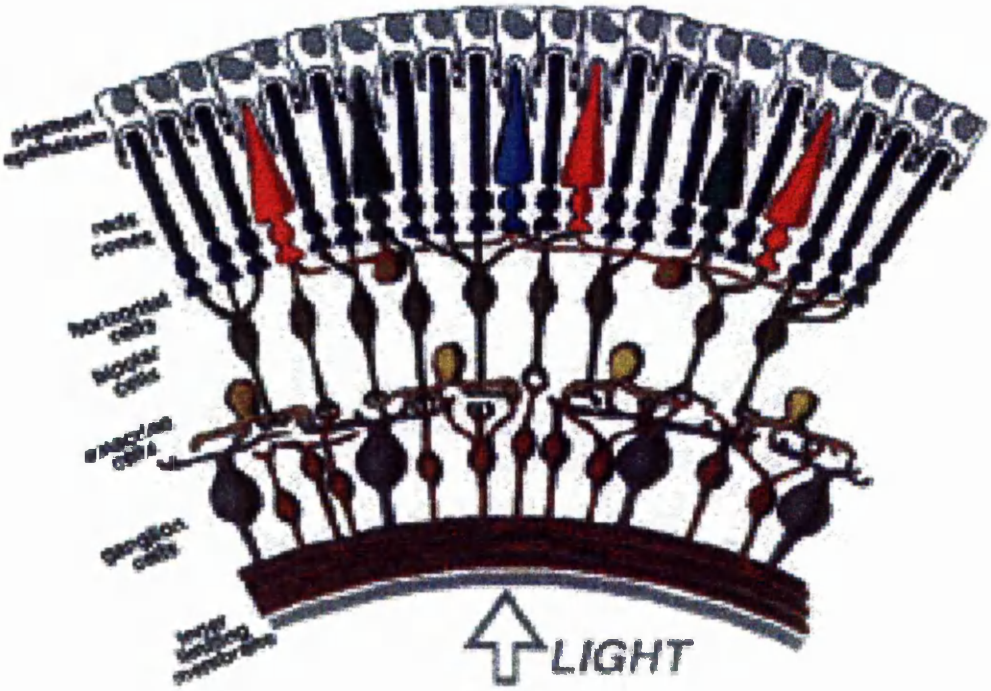


Figure 1.6 *Passage of light through the retina*: Light enters the eye through the cornea and lens, and then, has to pass through the complete thickness of the retina before striking the photosensitive elements, the rods and cones. The later are situated at the back of the retina against the pigment epithelium and choroid layers.

The tier of cells at the back of the retina contain the photoreceptors. They are the retinal cells, which convert light photons to electrical impulses. There are two basic types of photoreceptors - **rods** and **cones** (Figure 1.7). Rods are highly sensitive to light intensity, and cones to colour. Cones also help to give a clear image.

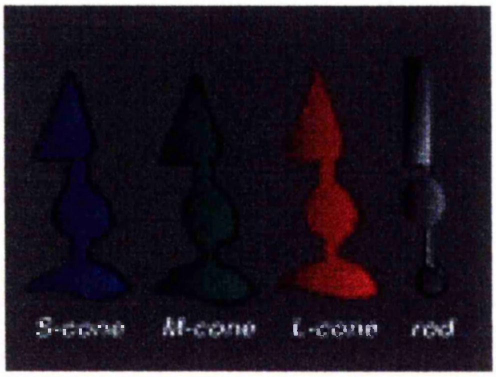


Figure 1.7: *Rod & Cones*

Rods contain the visual pigment **Rhodopsin**, which, when it absorbs light, splits into retinene and scotopsin. This structural change is thought to trigger off nerve impulses. Rods are highly sensitive to blue-green light (Figure 1.8), with a peak sensitivity at the 500nm wavelength of light, and are used for vision under dim-dark conditions at night.

Cones contain **cone opsins** as their visual pigment. They are maximally sensitive to either long wavelengths of light (red light), medium wavelengths of light (green light), or short wavelengths of light (blue light), (see Figure 1.8), depending on the exact structure of the opsin molecule. These cones, and the consequent pathways of connectivity to the brain, are the basis of colour perception in our visual image.

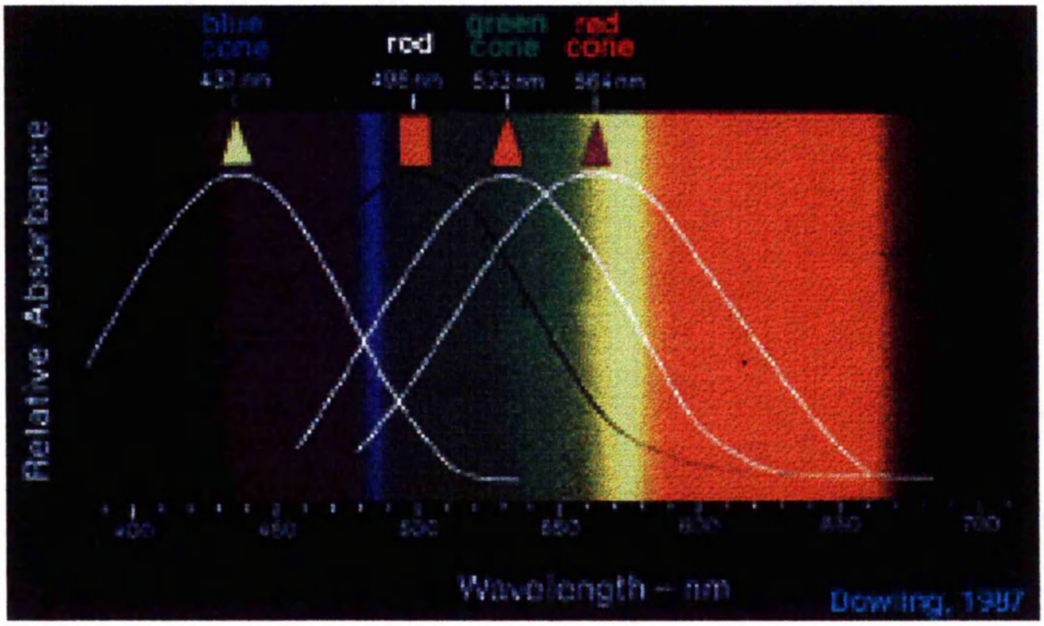
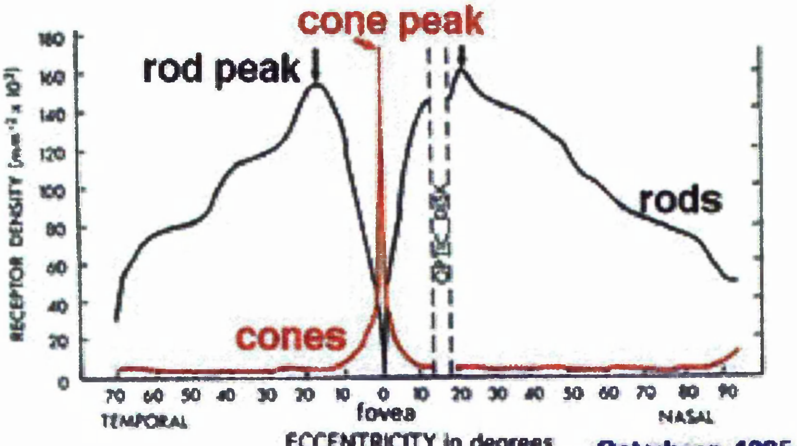


Figure 1.8: Relative absorption spectrum of cone receptors
(http://insight .med.utah.edu/Webvision)

There are 75-150 million rods, and 5-7 million cones in the average human retina, which are organized in a fairly exact mosaic. The fovea (which only accounts for 1 degree of our central vision) is rod free, and contains a peak cone density of 199,900 cones/mm^2, resulting in the highest visual acuity at the central focusing point. The mosaic here, is a hexagonal packing of cones. With increasing eccentricity, cone density falls steeply and the close hexagonal packing of the cones, is broken up by the rods outside the fovea. It is still a fairly organized mosaic, with the cones evenly spaced, surrounded by a ring of rods. The highest rod densities are located in a ring around the fovea, approximately 18 degrees (4.4mm) from the foveal pit. This is illustrated in Figure 1.9.



Osterberg, 1935

Figure 1.9 *Distribution of rods and cones along a horizontal line through the fovea*

Therefore, to recap, the function of the photoreceptor cells in the retina, is to catch quanta of light, and pass a message, concerning numbers of quanta of light and sensitivities to the different wavelengths, onto the next stage for processing. Information is transmitted laterally across the retina, due to the **horizontal cells**.

The first stage of processing for the evoked signal is the **bipolar cells**, which fall into two main categories - the on-bipolar and the off-bipolar. The on-bipolar cells hyperpolarise (increase the polarity of their transmembrane potential) in response to the receptor membrane potential generated. The off-bipolar cells depolarise (decrease this transmembrane potential) in response to the same signal (Hubel 1988).

Amacrine cells are analogous to horizontal cells. They are also laterally transmitting cells.

The **ganglion cells** further process the signal. They are categorised as on-centre, off-centre or on-off centre. They convert the amplitude modulated (analogue) signal generated by the bipolar cells to a pulse frequency modulated signal or digital spike varying in frequency of discharge in accordance with the magnitude change of the stimulating signal.

The axons of the ganglion cells sweep across the inner surface of the retina to the optic nerve head. A certain amount of scatter is caused at the receptor layer, due to the fact that the ganglion cell fibres lie at the front of the receptors. To avoid scatter at the foveal region (where the acuity of the eye is at its highest), the ganglion cells travel in arcs round the fovea, and converge at the optic disc. It is here that the ganglion cell axons pass through the sclera and form the optic nerve.

The terminal stage for physiological processing, and the start of more complex visual processing, is the visual cortex. This is located at the posterior part of the brain.

1.2 BASIC VISUAL PATHWAYS

1.2.1 The Pathway

Vision is generated by photoreceptors in the retina (Figure 1.10).

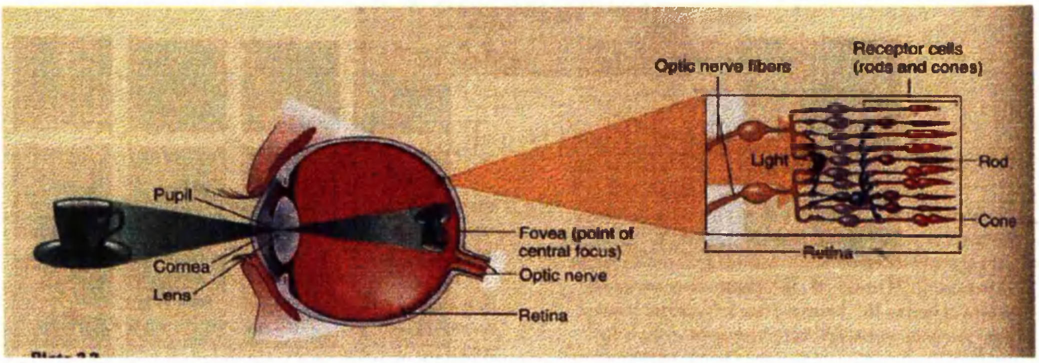


Figure 1.10 *Visual Fields in each eye*

The information leaves the eye via the optic nerve, and there is a partial crossing of axons at the optic chiasm. The axons are called the optic tract after the chiasm. This wraps around the mid brain to get to the lateral geniculate nucleus (LGN), where all the axons must synapse. The LGN axons fan out through the deep white matter of the brain as the optic radiations, which will ultimately travel to primary visual cortex at the back of the brain (Figure 1.11)

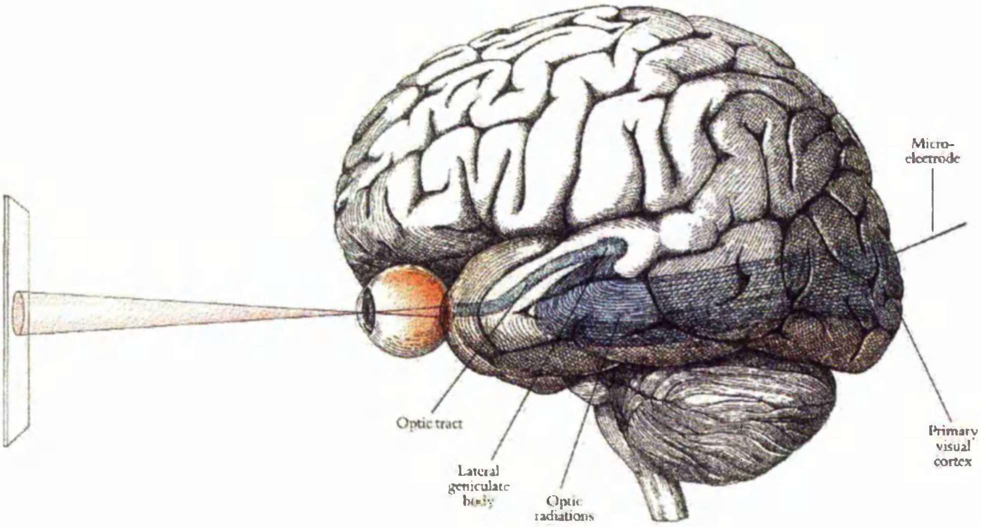


Figure 1.11: *Human Brain viewed from below*

Information about the world enters both eyes with a great deal of overlap. You can cut the image which is projected onto the retina down the middle, with the fovea defining the centre. Essentially, there are two halves of the retina, a temporal half (next to your temple) and a nasal half (next to your nose).

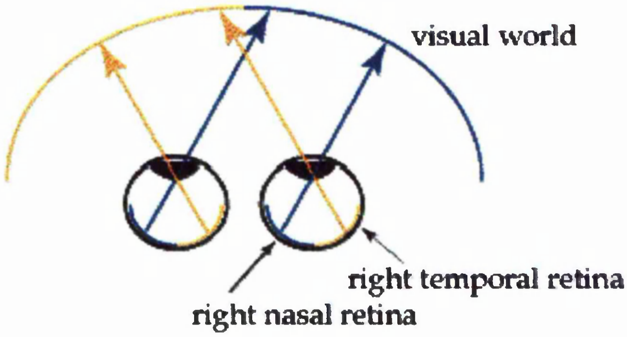


Figure 1.12 *Projection onto the retina*

As the visual image passes through the lens, it is inverted. Take for example your right eye, the nasal retina sees the right half of the world, and the temporal retina sees the left half. The right nasal retina and left temporal retina see pretty much the same thing (Figure 1.12). Also note, from figure 1.12, if you drew a line through the world at your nose, they would see everything to the right of that line. That field of view is referred to as the right hemifield. The image you see is divided into left and right hemifields, with each eye obtaining information from both right and left hemifields. For any object that you can see, both eyes are actually seeing it, which is important for depth perception, but the image will be falling on one nasal retina and one temporal retina.

The brain works on a crossed wires system - the left half of the brain controls the right side of the body, and vice versa. Thus, the left half of the brain is only concerned with input from the right side of the world. The fibres of the retina sort themselves out in order to separate left and right hemifields. Figure 1.13 illustrates the nerve fibres from the nasal retina crossing over at the optic chiasm. Whereas, the temporal retinae, are already positioned to see the opposite sides of the world, so they do not cross.

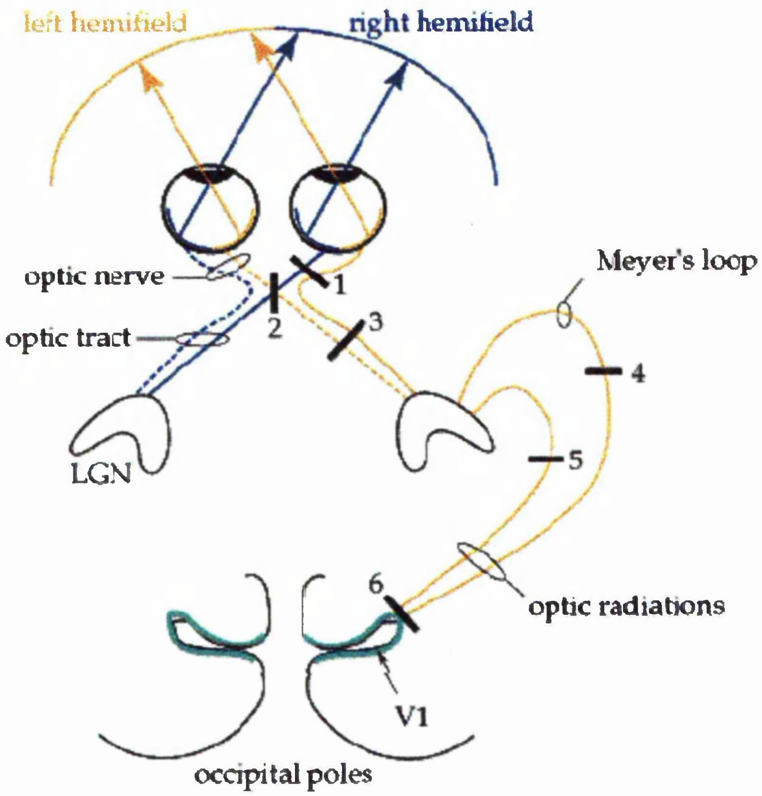


Figure 1.13 *Visual Fields*

Chapter 2

MEASUREMENT OF VISUAL FUNCTION

Retinal diseases are the most common cause of blindness in western Countries; the prevalence of these conditions is increasing rapidly as the population ages. In diagnosing these conditions, and in evaluating treatment, the electrical responses of the retina are very important. Vision relies on more than one area of the retina - the central area is specialized for high resolution vision (reading, etc.), and the periphery is needed for orientation, and control of position and movement in the world.

The Multifocal Electroretinogram (MFERG) will be introduced in this chapter. It can evaluate the retina at up to 103 locations. It describes the way in which the response of the retina changes its characteristics after it receives a flash of light, and this response contains significant diagnostic information.

2.1 PERIMETRY & VISUAL FIELDS

Non-invasive electrodiagnostic and psychophysical testing enable assessment of the entire length of the visual pathway.

Perimetry is a common psychophysical test of visual function - it yields important diagnostic information on a wide variety of opthalmic disorders. Traditionally visual function has been tested as **visual acuity** - the ability to discriminate fine details of objects and **visual field** - the portion of space in which objects are visible at the same moment during steady fixation of gaze in one direction.

The two most common types of perimetry are Goldmann kinetic perimetry and threshold static automated perimetry. With Goldmann or "kinetic" perimetry, a trained perimetrist moves the stimulus; stimulus brightness is held constant. The limits of the visual field are mapped to lights of different sizes and brightness.

With threshold static automated perimetry, the patient sits in front of a concave dome, and visually fixates on a central object within the dome. A computer-driven programme flashes small lights at different locations within the domes surface, and the patient presses a button to acknowledge whether they saw the stimulus. The most commonly used computer programme tests the central 30° of the visual field using a six degree spaced grid. This is accomplished by keeping the size and location of a target constant and varying the brightness until the dimmest target the patient can see at each location is found. These tests produce a visual field map, which is very important in diagnosing diseases of the visual system. Scotomas, or non-seeing areas, are plotted on this visual field map. Shape, location and sensitivity loss of these scotomas are important in the diagnostic process.

There are a number of drawbacks to conventional Perimetry. It is a subjective test, which relies on the patient fixating on a central fixation mark. The patient responds to an external stimuli, by pressing a button, or giving a verbal response. Perimetry gives a field map of the entire pathway, and cannot localise a defect to retina, optic nerve, or visual cortex. However, this can usually be achieved in conjunction with other clinical information.

Electrodiagnostic testing complements the information obtained from subjective measures of visual function. In general, electrodiagnosis gives an objective evaluation of retinal function. A variety of techniques are used to stimulate the retina, and give global information on a particular level or layer of the visual pathway.

2.2 VISUAL ELECTROPHYSIOLOGY

Traditionally, the main tests carried out in the electrophysiology clinics are, **Electrooculograms (EOG), Visual Evoked Cortical Potential (VECP),** and

Electroretinogram (ERG). The **EOG** is a test of the outer retinal layer, called the Retinal Pigment Epithelium. It measures a constantly present standing potential between the front (cornea), and back (retina) of the eye. The main use of the **EOG** is in the diagnosis of Best's disease and other forms of juvenile macular degeneration. The **VECP** is a test of optic nerve tract and optic nerve function. Two young Scotsmen Dewar and McKendrick, independently discovered the **ERG** in 1873, but it did not find widespread clinical application until the contact lens electrode was developed by Riggs in 1941. The ERG gives functional information on a number of retinal cells, such as the photoreceptors, bipolar cells, ganglion cells and the Retinal Pigment Epithelium.

When light strikes the retina, a series of fast and brief changes in electrical potential can be detected by recording electrical potential at an electrode placed on or near the eye. A plot of the changing electrical potential with time, on a timescale of milliseconds, is called an electroretinogram, see Figure 2.1

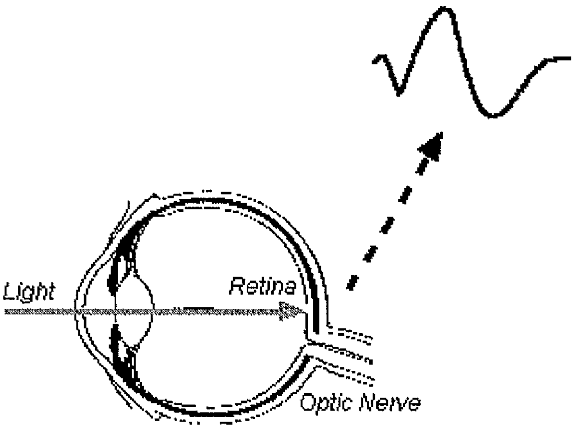


Figure 2.1 *Basic ERG Principle*

Typically, the ERG consists of three components, the "a", "b" and "c" waves, and it is the "a" and "b" waves which are of most interest in routine clinical investigations. The negative component or "a-wave" is measured from the baseline to the trough, and The positive component or "b-wave" is measured from the preceding trough to the positive peak. Less important waveforms are sometimes observed in ERG waveforms known as the "c-wave". The "a", "b" and "c" waves are illustrated in Figure 2.2.

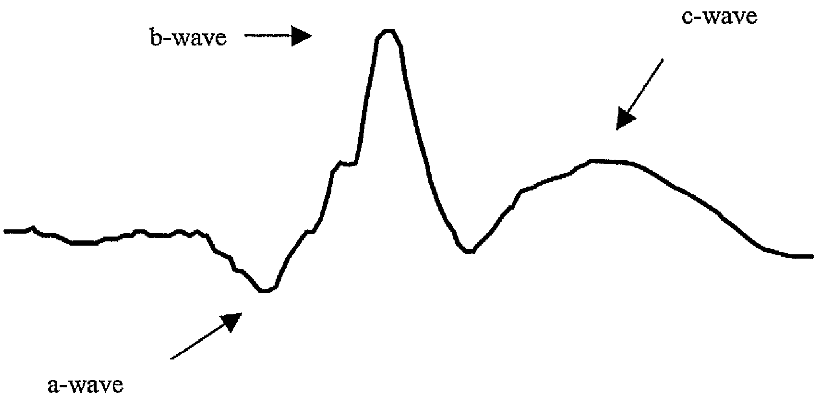


Figure 2.2 *"a", "b" and "c" Waves*

The height, duration or shape of a peak can change with a retinal disorder. Scientists and ophthalmologists have learned to interpret such changes. For more information on the origin of the "a" and "b" waves, see for example Carr & Siegel (1990).

The International Society for Clinical Electrophysiology of Vision (ISCEV) published Standards in 1989, to ensure that electrophysiology is carried out in safe standard conditions throughout the world. It also allows comparisons to be made between data produced at different centres.

A full standard ERG is designed to measure the following responses:

1. Rod dominated response in the dark adapted eye (primarily rods stimulated)
2. Maximal response in the dark-adapted eye (both rods and cones stimulated)
3. Oscillatory potentials (thought to be produced by horizontal and amacrine cells)
4. Cone mediated response
5. Flicker Response (from cone photoreceptors, since stimulus is driven at a frequency too high for rods to cope)

Figure 2.3 illustrates the five basic responses:

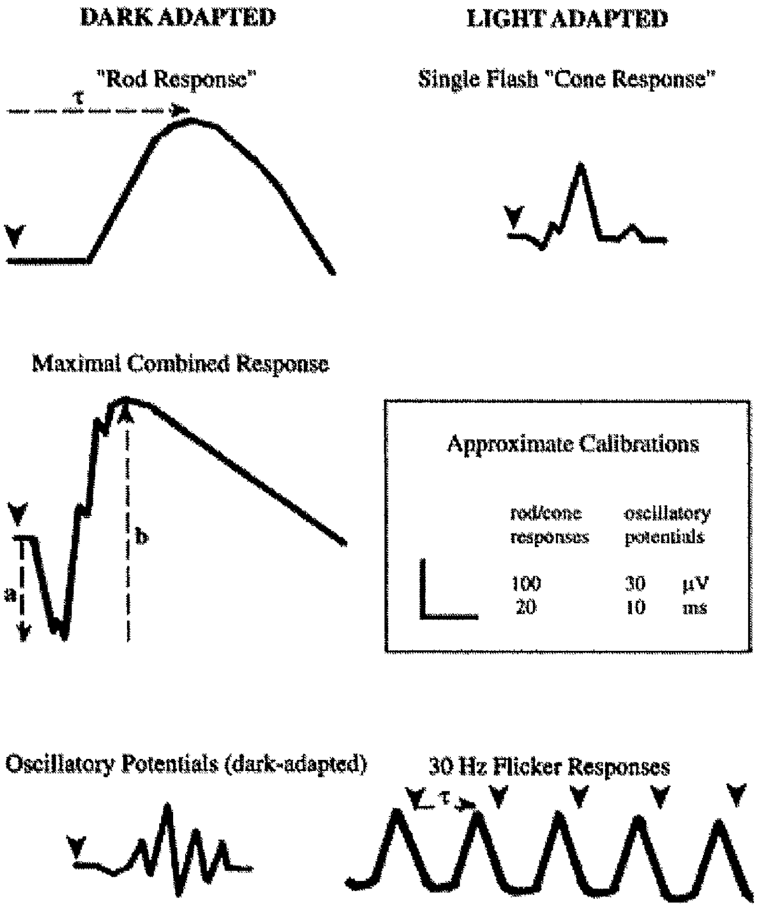


Figure 2.3: *Five Basic Responses*

To recap, for more than 100 years, it has been known that a flash of light will elicit a distinctive response from the human eye – the electroretinogram (ERG). The ERG records the evoked potentials and gives functional information on a number of retinal cells. In particular, it shows marked differences in the response of rod and cone membrane potentials to a flash of light (Carr&Siegel 1990). The response will be either normal or abnormal, depending on the amplitude and latency of different components of this ERG waveform. It is a mass response from the retina, and has been commonplace in Electrophysiology Clinics for around thirty years.

The problem with the global ERG, however, is that approximately thirty percent of the photoreceptors have to be malfunctioning before any abnormalities are detected in the resulting waveform, and although Electrophysiology plays a prominent role as a diagnostic tool, its ability to detect localised pathology leading to specific visual dysfunction is necessarily limited, due to the fact that a large number of retinal pathologies originate in small localized regions of the retina. Take for example the macula, which only contains about seven percent of the total cone population, therefore the combined cone and rod responses from the macula contribute less than ten percent of the full photopic ERG. Because of this, disease limited to the macular region is typically not detected.

Several elaborate and sophisticated techniques have been designed to refine the ERG, in order to allow small areas to be examined. A **Focal ERG** is an ERG evoked by a small (10° or less) focal stimulus. It has not been feasible to obtain ERG responses from a sufficient number of retinal locations to permit response topography maps, due to two main reasons, signal detection and stray light from the retina. Sanderberg and Ariel (1977) developed a hand-held stimulator opthalmoscope as an alternative approach to field topography mapping. This permits placement of a single small ERG stimulus to generate a response from a few selected areas. However, it does not

allow recording from a large number of locations in the same session.
Recent advances in the application of Pseudo Random Binary Sequences (PRBS) to signal averaging has yielded higher signal to noise ratios in shorter periods of acquisition than was previously practical (Fricker & Sanders 1974, Srebro & Weldon 1980, Sutter & Tran 1992). The enhancement of signal quality has made it is possible to recover functional information from localised regions of the retina by stimulating multiple areas of the visual field with independent modulators (flashing lights), controlled by a special case of PRBS called an m-sequence (Sutter & Tran 1992).

Electronic engineering textbooks (see for example Horowitz & Hill, 1989), describe the use of shift-registers to create pseudo-random binary sequences.

2.3 MULTIFOCAL ELECTRORETINOGRAPHY

The **Multifocal ERG** technique, which was first described by Sutter and Tran (1991), enables derivation of responses from hundreds of locations in approximately the same amount of time it would take to derive one single local response. Multifocal ERG is an advanced technology based on the standard Electroretinogram. The current generated by the neural activity in the retina changes when the retina is exposed to changing light levels. This current can be measured using an electrode (Gold Foil) resting against the cornea at the front of each eye.

A description of the methodology behind the multifocal system can be found in Sutter & Tran (1991). Linear systems are described by their impulse responses, which can be measured directly using pulsed inputs. If white noise is used as a test input, much better signal-to-noise ratios can be achieved. The impulse response **I(t)** is derived from the response as the cross-correlation function between input **x(t)** and the response **r(t)**:

$$I(t') = \int_0^{\tau} x(t - t')r(t)dt$$

For correlational shifts larger than the duration τ, the impulse response, the recorded signal no longer shows any correlation with the input except for background noise. Now, if the same input sequence, delayed relative to the first input by a time interval T >τ, is used to stimulate the second input, then its impulse response would appear on the cross-correlation function starting at the correlational shift T. Therefore, a single cross-correlation can be used to extract the impulse response of both inputs. Figure 2.4, illustrates this process for multiple inputs.

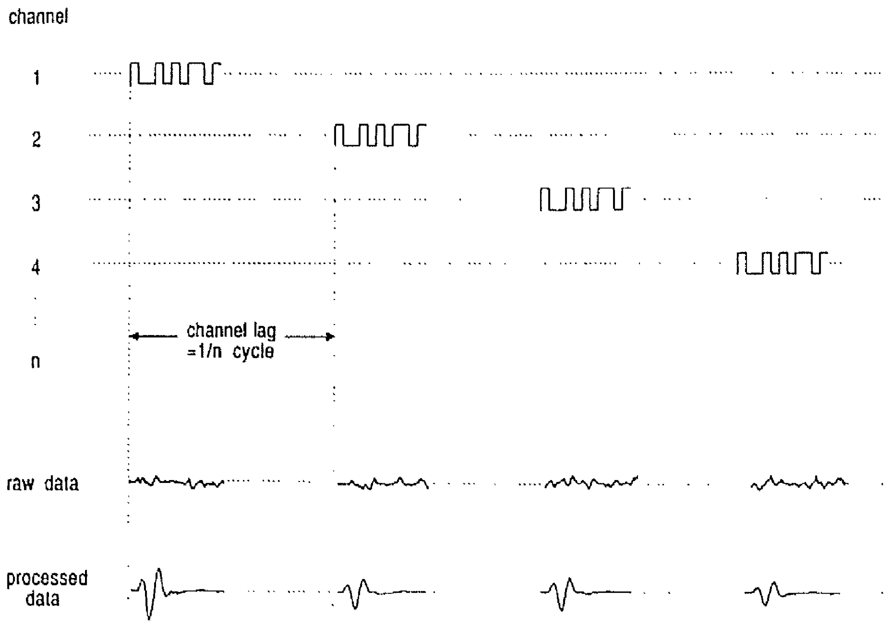


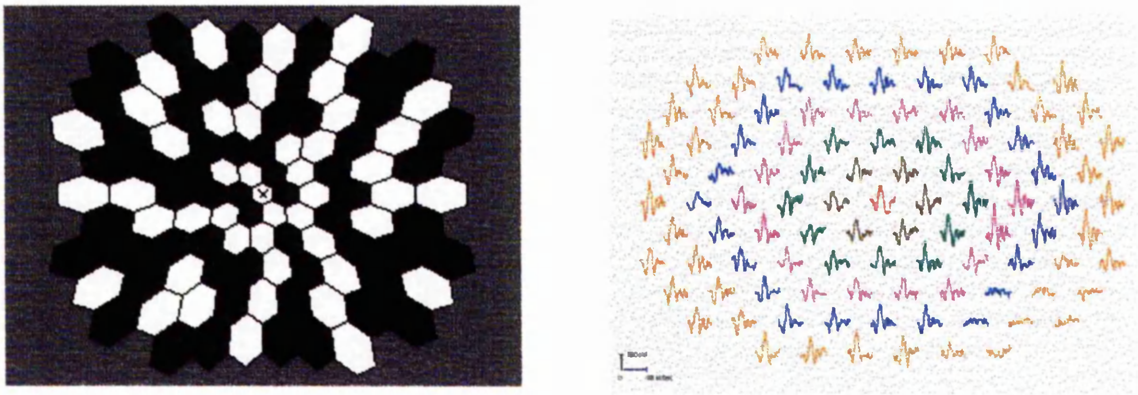
Figure 2.4: Cross-Correlation Function. Each channel is stimulated according to the same binary m-sequence. A delay in the stimulus between channels renders the response from different channels uncorrelated. They are extracted from the raw data by computation of cross-correlation between the m-sequence and the response cycle. The responses of the individual channels are found distributed along the cross-correlation cycle at intervals equal to the channel lag.

Simultaneous stimulation of multiple inputs with this technique does not increase the noise - if noise contamination is overall additive. The input responses of all the inputs are obtained with the same signal-to-noise ratio, as if each one were individually stimulated for the same period of time. Fortunately, noise in the ERG signal is overall additive, and signal contributions from different inputs have zero correlation, and therefore, do not contaminate each other with apparent noise.

The basic principle used in this study is to stimulate all inputs with the same white sequence. This is to obtain responses from a large number of inputs. Note that the stimulation of all inputs is exactly equivalent. White noise test inputs with a binary amplitude distribution, such as m-sequences, are the most efficient, especially in the analysis of nonlinear systems. More information on the generation and properties of m-sequences follows in Chapters 3 and 4 respectively. The most important feature is that, if the m-sequence is selected properly, then the nonlinear interactions between the sequences, will fall between the first order responses mentioned above.

2.3.1 Multifocal Stimulus

Multifocal Electroretinography projects a pattern, which changes pseudo-randomly depending on the m-sequence chosen (Figure 2.5a). The algorithm has been chosen to guarantee that during an examination, no stimulus sequence is repeated, and all stimulus patterns appear once and only once, thus allowing separate signals to be extracted for each test location (Figure 2.5b).



2.5a Patient sits looking at a screen on which this hexagonal stimulus is presented

2.5b Whilst the patient sits there, a real time map of their visual function is produced which can be used instantly for diagnosis

Figure 2.5: *Multifocal ERG Stimulus and Trace Array*

The stimulus pattern usually consists of 61-241 hexagonal areas covering up to +/- 25 degrees of the visual field.

An ERG response consists of: large cone and rod response, small contribution from ganglion cell and optic nerve head component. Figure 2.6 shows a graph of the averaged ERG signals over concentric rings of the retina.

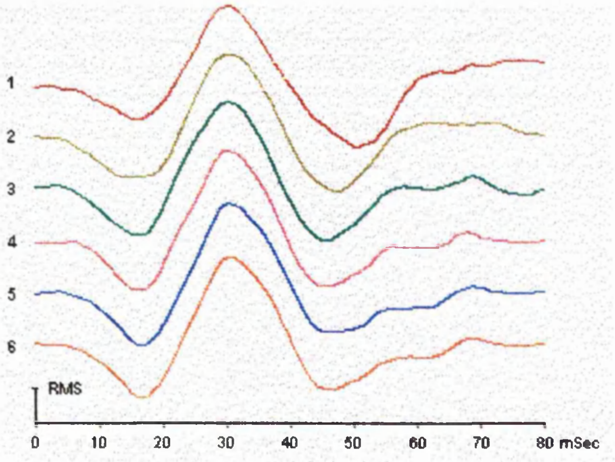


Figure 2.6 *Averaged ERG Signals over concentric rings of the retina.*

A wide range of variables can influence the quality of the multifocal response. These variables can be placed into four categories:

1. The method of stimulus delivery will determine the field of view, inference levels and the duration of on-state stimulation.
2. Data acquisition variables such as electrode type and placement, amplifier specifications and filter bandwidth settings will have a direct inpact on waveform shape and on the topographic distribution of signal amplitudes.
3. Patient variables such as fixation, pupil dilation and refractive error.
4. There are a variety of measurements which can be taken from multifocal recordings.

For more information, see for example, "Technical aspects of multifocal ERG recording" Keating et al (2000).

2.4 THE MULTIFOCAL LED STIMULATOR

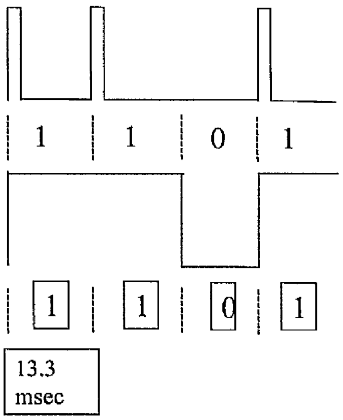
The multifocal LED System was developed to improve the understanding of the Multifocal response. It shows that significant multifocal responses can be obtained at high driving frequencies with short recording techniques. The LED Stimulator is a useful tool for investigating and optimising the multifocal technique.

2.4.1 CRT v LCD Stimulus Delivery

The standard output device for both computers and the domestic television set is the cathode ray tube (CRT). The LCD is the Liquid Crystal Device. Figure 2.7 compares the duration of a white state (i.e. a '1' in the m-sequence) for both the CRT and LCD

Stimulus.

CRT v LCD stimulus delivery



CRT

m-sequence ‘1’ = 2 msec On + 11 msec Off

m-sequence ‘0’ = 13 msec Off

LCD

m-sequence ‘1’ = 13 msec On

m-sequence ‘0’ = 13 msec Off

Figure 2.7: *CRT v LCD Stimulus Delivery*

Recently a Multifocal LED Stimulator has been developed in collaboration with the Department of Electronics at the Southern General Hospital in Glasgow. Several Light Emitting Diodes (LED's) are responsible for illuminating each hexagonal area in the multifocal display. The LED's can be controlled to a much finer degree than the CRT or LCD systems. Theoretically, we can vary the temporal frequency of the LED's (i.e. the LED's can be switched off or on for any length of time). They have the ability to project higher luminance levels, and there is no raster update from the screen.

2.4.2 LED Stimulator Specifications and Construction

This is beyond the scope of this thesis, but a summary has been included for completeness.

2.4.2.1 LED Stimulator Specifications

- Fixed Stimulus Geometry - 61 element array
- Temporal resolution of 1 msec
- High stimulus intensity (up to 4,200 Cd m $^{-2}$)
- PC Control

2.4.2.2 LED Device Construction

Figure 2.8 (a) - (d) illustrates diagrammatically the construction of the LED System.

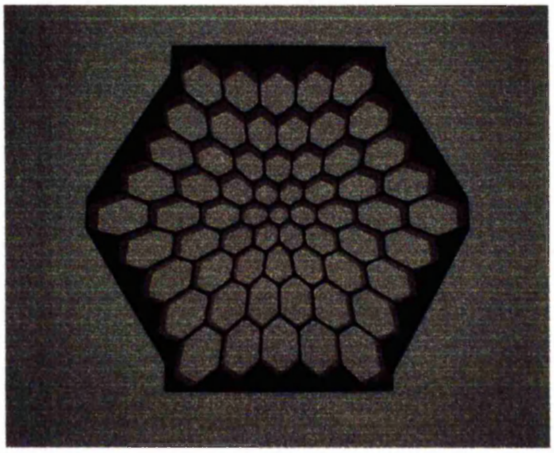


Figure 2.8(a): *Black Polycarbonate honeycomb LED housing*

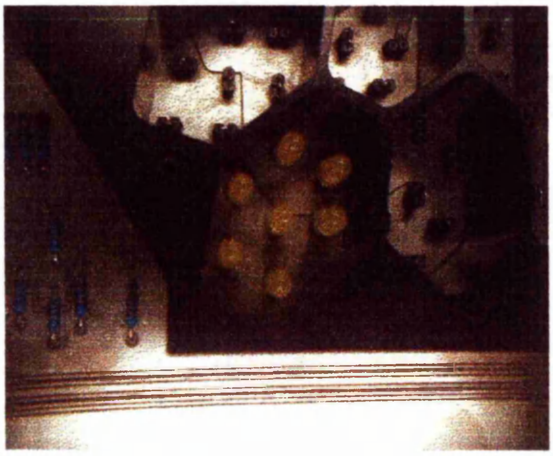


Figure 2.8(b):*Build circuitry & populate with LEDs*

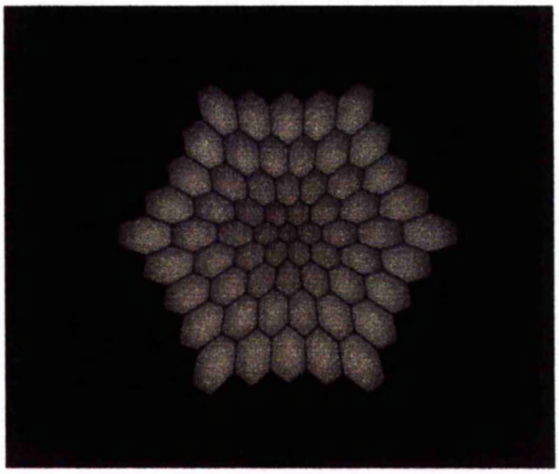


Figure 2.8 (c): *Add Diffuser*

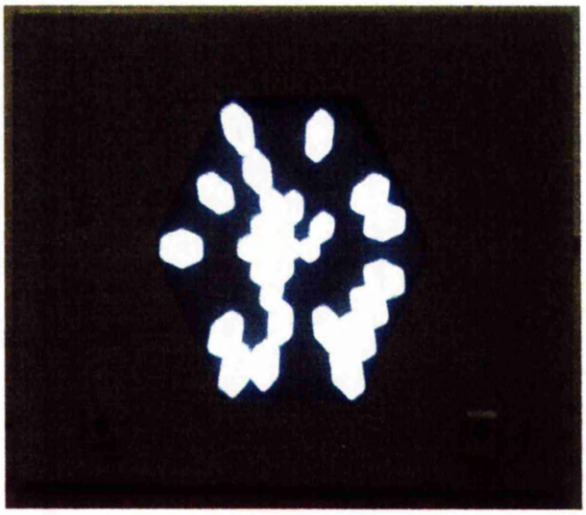


Figure 2.8(d): *Integration with control electronics*

2.4.3 Varying the Temporal Driving Frequency

The Multifocal LED System allows the temporal driving frequency to be altered. The standard CRT stimulation rate is 77Hz (1 / 13.3 msecs), see Figure 2.9. As mentioned previously this is when a 1 in the m-sequence results in the stimulus being in an "On" state for 1 msec and "Off" state for 12 msecs. Whereas, a 0 in the m-sequence results in the stimulus being in an "Off" state for 13 msecs.

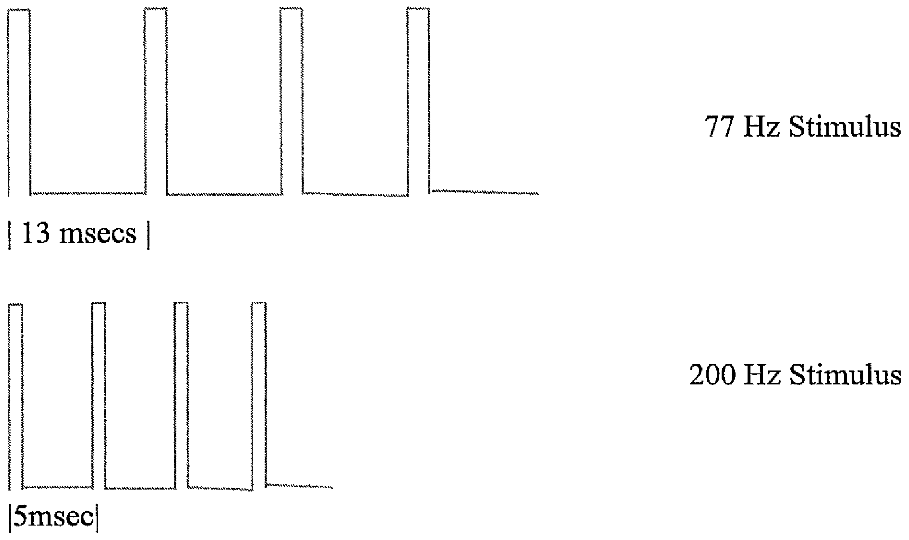
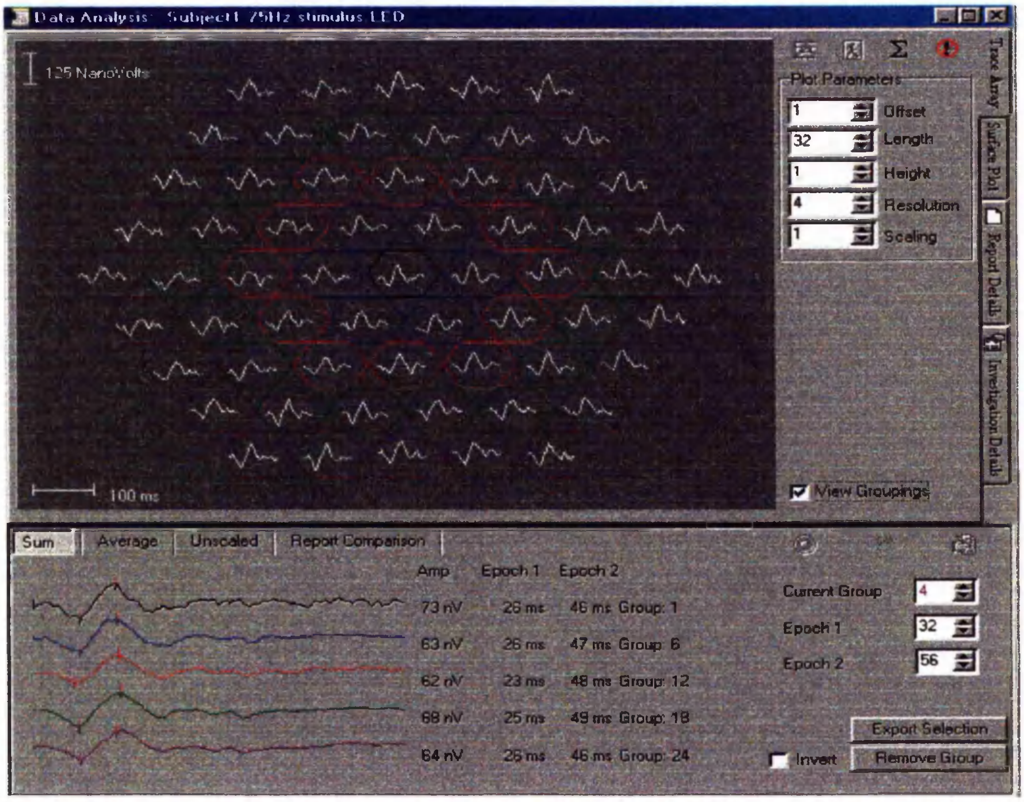


Figure 2.9: *Stimulation Rate of 77Hz and 200 Hz.*

Similarly if we stimulate at 200 Hz (1/5msec), then if a 1 in the m-sequence results in the stimulus being in an "On" state for 1 msec and "Off" state for 4 msecs. Whereas, a 0 in the m-sequence results in the stimulus being in an "Off" state for 5 msecs.

Figure 2.10(a), (b) and (c) is the results of a test using a 15 bit m-sequence of length 32,767, to stimulate each hexagonal area. If the stimulus is driven at 77Hz, the test has a recording time of 7 minutes 6 seconds, whereas if we drive the stimulus at 200 Hz, then the recording time decreases to 2 minutes 44 seconds. The test time can decreased even more, to 1 minute 5 seconds if we stimulate at 500Hz.



Plot Parameters

1	Offset
32	Length
1	Height
4	Resolution
1	Scaling

☑ View Groupings

Amp	Epoch 1	Epoch 2	
73 nV	26 ms	46 ms	Group: 1
63 nV	26 ms	47 ms	Group: 6
62 nV	23 ms	48 ms	Group: 12
68 nV	25 ms	49 ms	Group: 18
64 nV	26 ms	46 ms	Group: 24

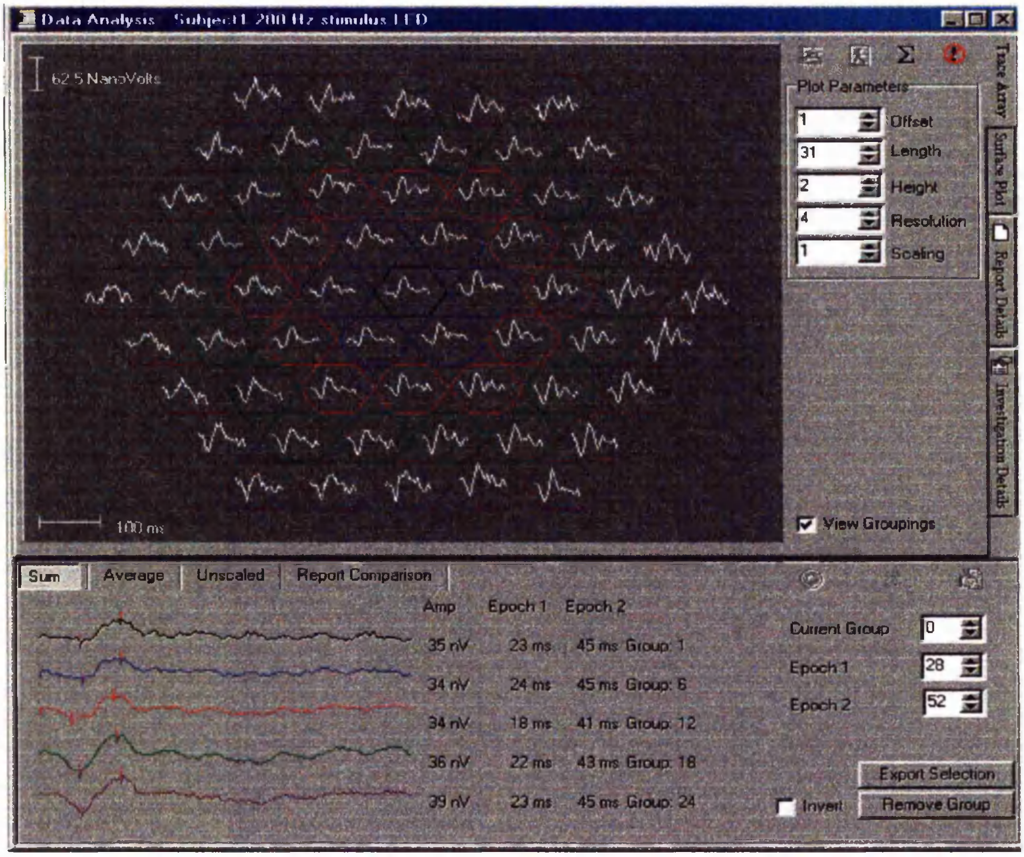
Current Group 4

Epoch 1 32

Epoch 2 56

□ Invert

Figure 2.10(a): *13 msec base period - Stimulation Rate 77Hz*



Plot Parameters

1	Offset
31	Length
2	Height
4	Resolution
1	Scaling

☑ View Groupings

Amp	Epoch 1	Epoch 2	
35 nV	23 ms	45 ms	Group: 1
34 nV	24 ms	45 ms	Group: 6
34 nV	18 ms	41 ms	Group: 12
36 nV	22 ms	43 ms	Group: 18
39 nV	23 ms	45 ms	Group: 24

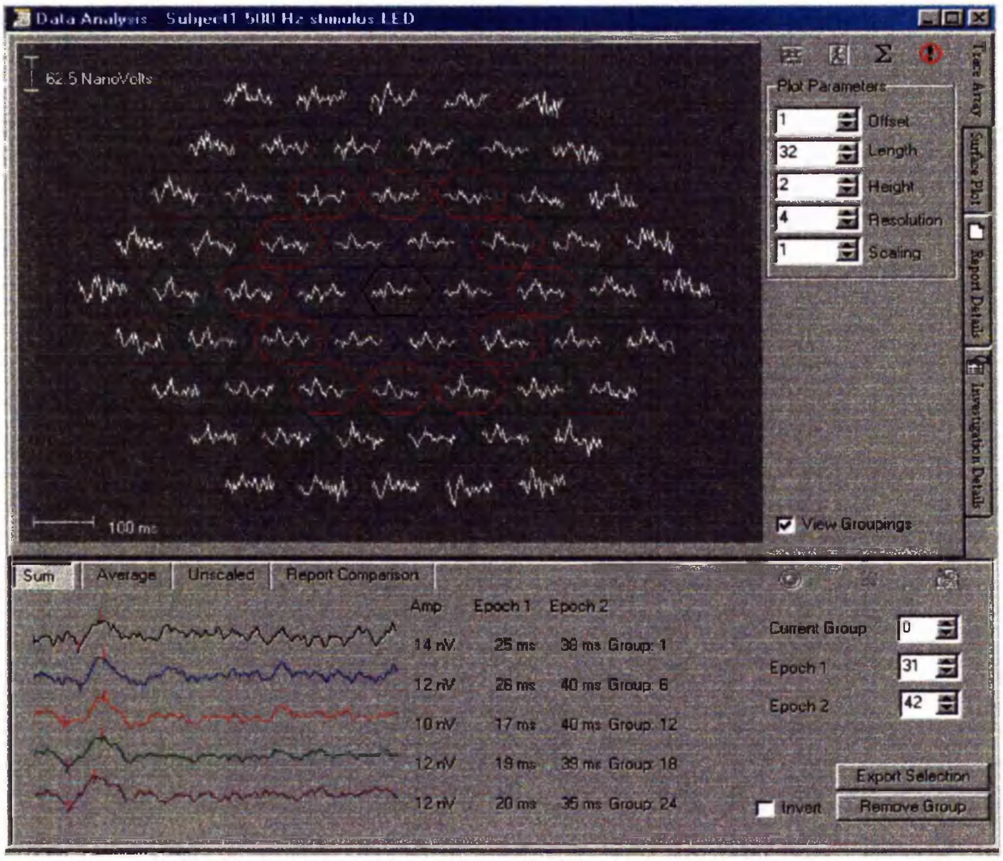
Current Group 0

Epoch 1 28

Epoch 2 52

□ Invert

Figure 2.10(b): *5 msec base period - Stimulation Rate 200 Hz.*



Data Analysis   Subject1 500 Hz stimulus LED

Plot Parameters

- Offset: 1
- Length: 32
- Height: 2
- Resolution: 4
- Scaling: 1

☑ View Groupings

Amp	Epoch 1	Epoch 2	
14 nV	25 ms	38 ms	Group: 1
12 nV	26 ms	40 ms	Group: 6
10 nV	17 ms	40 ms	Group: 12
12 nV	19 ms	39 ms	Group: 18
12 nV	20 ms	35 ms	Group: 24

- Current Group: 0
- Epoch 1: 31
- Epoch 2: 42

□ Invert

Figure 2.10(c): *2 msec base period - Stimulation Rate 500 Hz.*

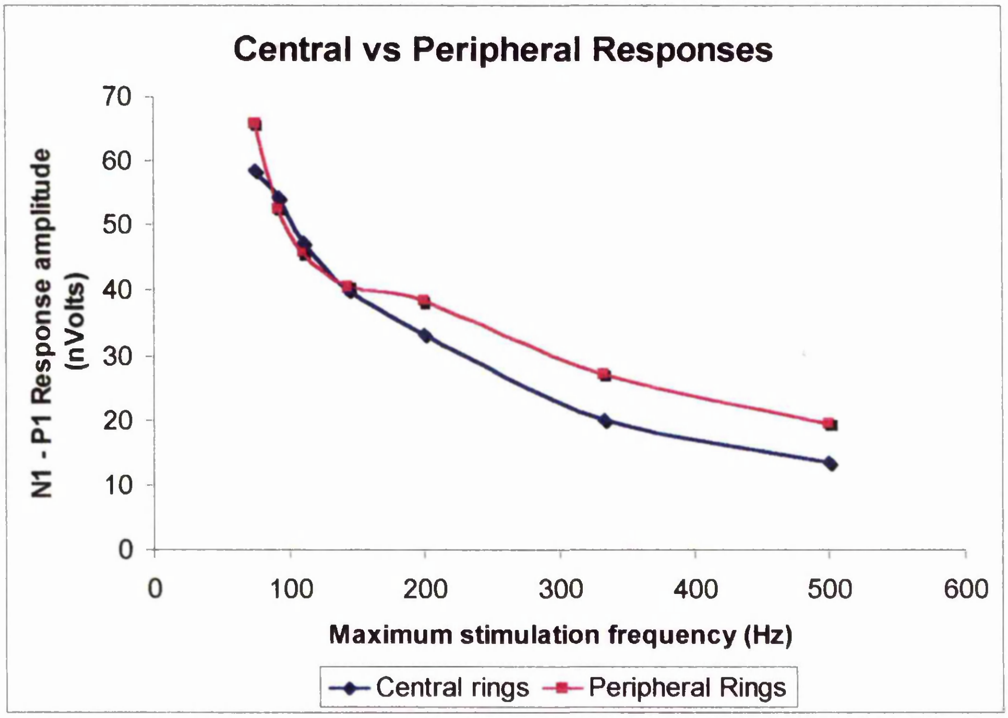


Figure 2.11: *Response amplitude as a function of driving frequency.*

The amplitude of 500Hz responses illustrated in Figure 2.11, are 27% that of the 75Hz responses. Which leads to the question, Do these experiments imply that the retina can respond at frequencies as high as 500Hz? This is being investigated in more detail by Dr David Keating at Gartnavel General Hospital.

2.4.4 Understanding the Multifocal First Order Response

Consider the standard 75Hz Stimulation. The first order or impulse response illustrates how the eye responds to a flash of light, and defined as the sum of transitions to a white stimulus minus the sum of transitions to a black stimulus in the schematic diagrams of Figure 2.12 (a) and (b).

2.4.5 Understanding the Multifocal Second Order Response

The second order shows how the eye reacts to the interaction between flashes of light, and is represented by the difference between a change of state and no change of state, in the schematic diagram illustrated in Figure 2.13 (a) and (b).

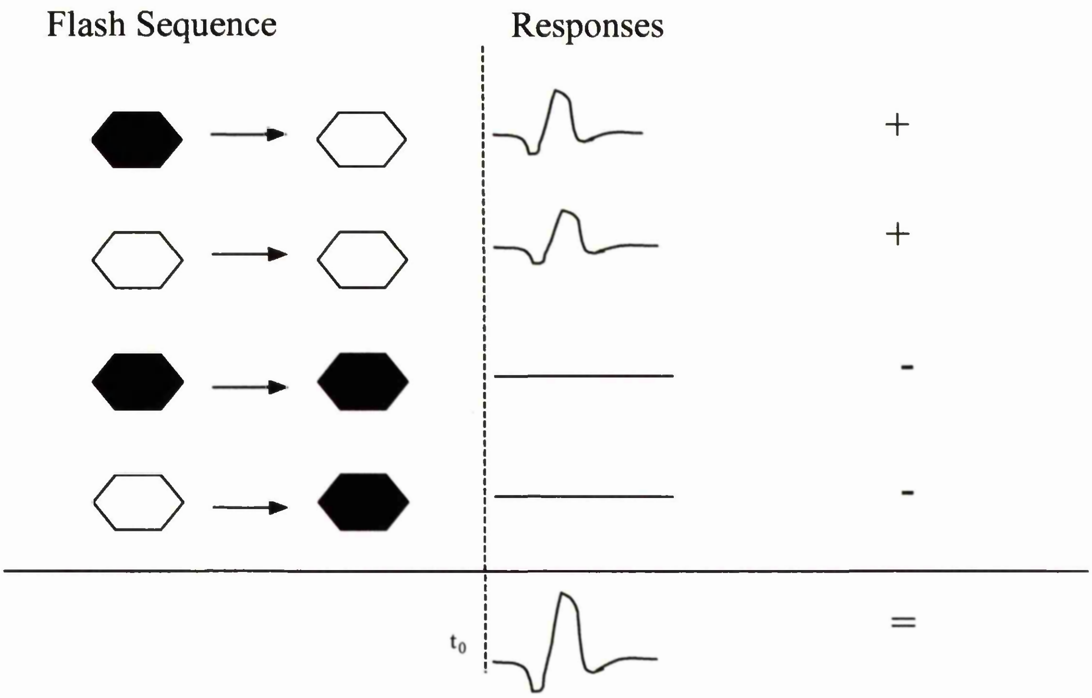


Fig 2.12(a): *First order CRT Response*

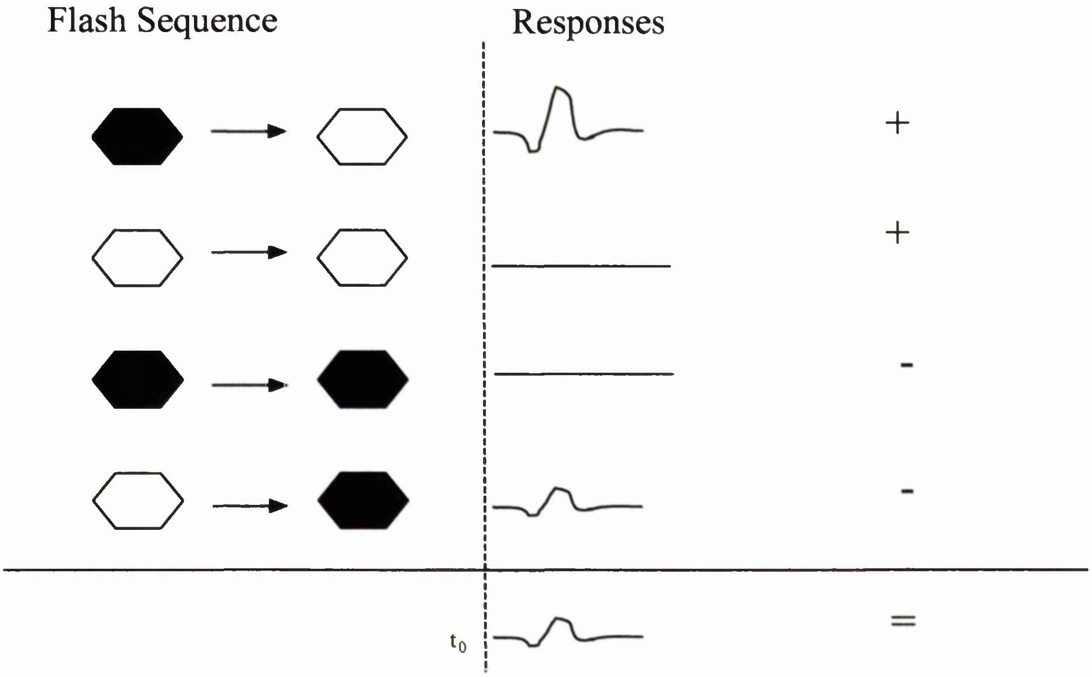


Fig 2.12(b): *First order LCD Response*

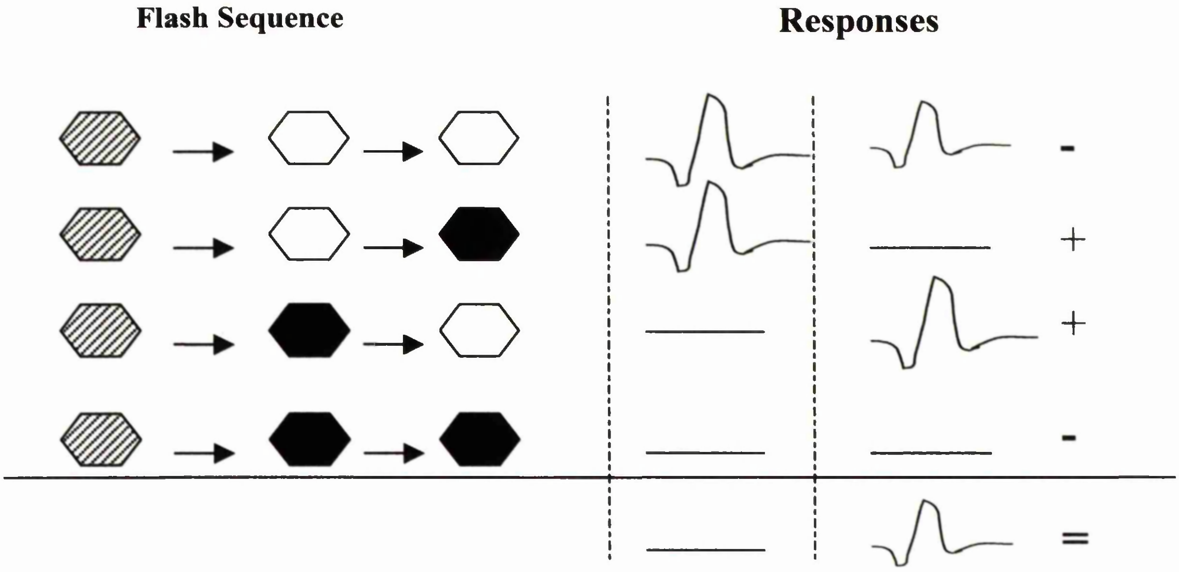


Fig 2.13(a): *2nd order CRT Response*

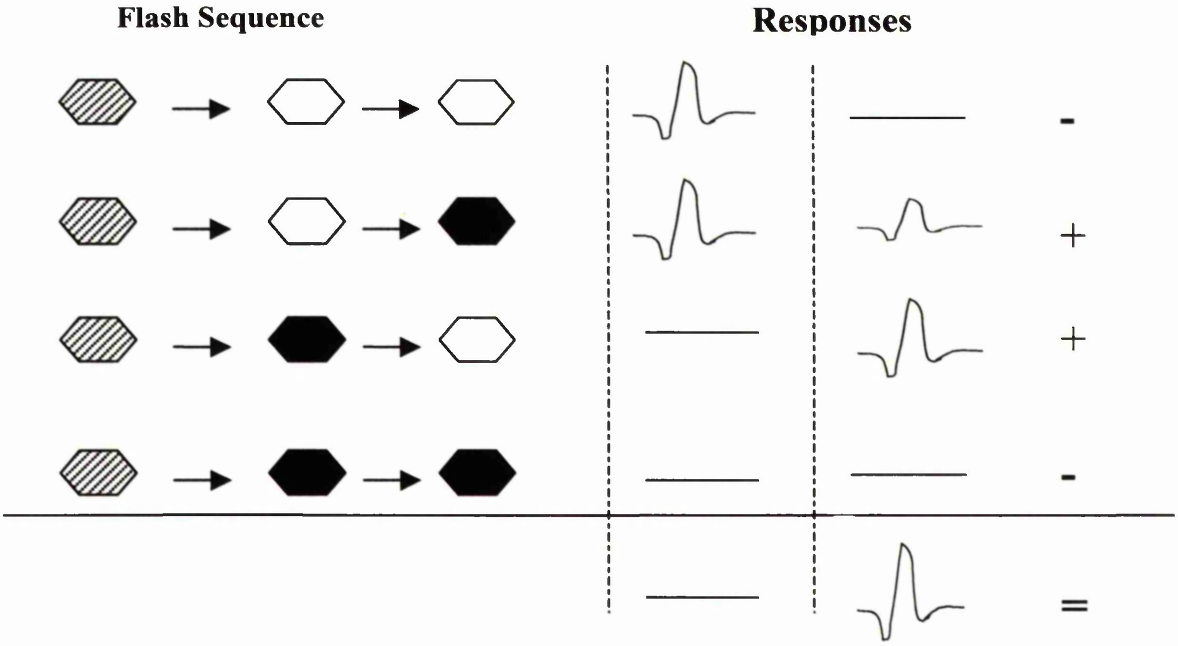


Fig 2.13(b): *2nd order LCD Response*

Chapter 3

GENERATION OF PSEUDO RANDOM BINARY SEQUENCES

In this chapter pseudo-random binary sequences will be constructed and the mathematical foundations for the remainder of the thesis will be introduced.

3.1 THE SHIFT REGISTER

The most popular (and the simplest) Pseudo-Random Binary Sequence (PRBS) generator is the feedback shift register. PRBS (which are also called PN (pseudo noise) sequences, m- sequences, or maximal length shift register sequences) are binary sequences of length $2^m - 1$, which satisfy a linear recurrence whose characteristic polynomial of degree m, is primitive (McEliece, 1987). Primitive polynomials h(x), will be defined below, but for the moment, take an example where m = 4:

$$h(x) = x^4 + x + 1$$

This corresponds to a feedback shift register as shown in Figure 3.1

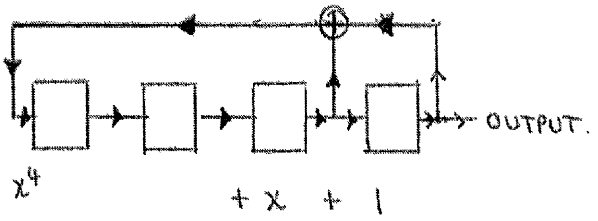
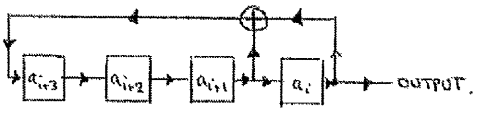


FIGURE 3.1 *Feedback shift register corresponding to* $x^4 + x + 1$

Generally, this polynomial specifies a shift register with m boxes, each containing a 0 or 1 (since binary). At each time unit, the contents of the boxes are shifted one place to the right, and the boxes corresponding to the terms in the primitive polynomial are added together modulo 2, and fed back into the left hand box (MacWilliams & Sloane, 1976). Therefore, if the register contains $\mathbf{a_{i+3}, a_{i+2}, a_{i+1}, a_i}$ (Figure 3.2), at time i, then at time i+1, it contains $\mathbf{a_{i+4} = a_{i+1} + a_i, \quad a_{i+3}, \quad a_{i+2}, \quad a_{i+1}}$.



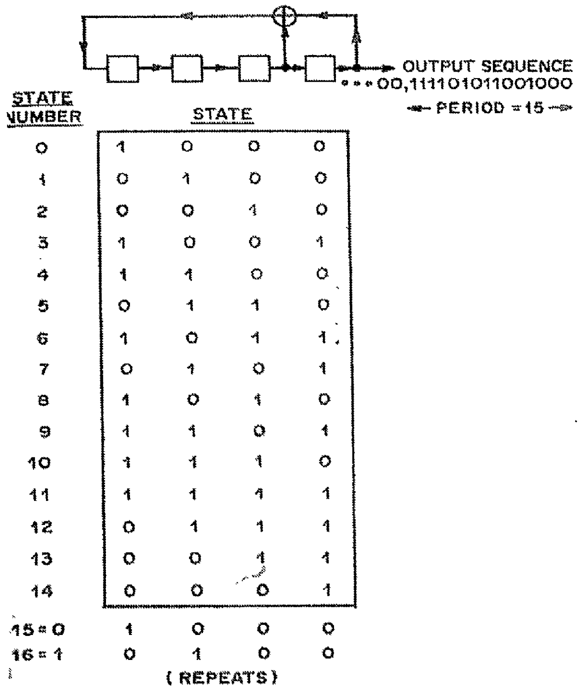
The Feedback Shift Register Specifies a Recurrence Relation

FIGURE 3.2 *Feedback shift to implement the recursion* $a_{i+4} = a_{i+1} + a_i$

The feedback shift register generates infinite sequence $a_0\, a_1\, a_2\, a_3 \ldots\ldots a_i \ldots.$ which satisfies the recurrence

$$\mathbf{a_{i+4} = a_{i+1} + a_i \ \ (mod\ 2), where\ \ i = 0,1,\ldots\ldots}$$

The maximum possible number of conceivable states of an m-bit register is 2^m (since each of the m boxes contains a 0 or 1). Therefore, the output sequence which is generated must be periodic. However, the all zero state cannot occur, because if it did, the state of all 0's would get "stuck" in the circuit, and the output would be the all zero sequence (P. Horowitz, 1989). Thus, the maximum possible period is $2^m - 1$. A primitive polynomial h(x), is one for which the output sequence has maximum possible period. Figure 3.3 illustrates the successive states of the output sequence for the feedback shift register in Figure 3.1, if the initial state is 1000 (we could start anywhere except 0000). It can be seen that the output sequence generated is the same as the right-hand column of the list of states.



STATE NUMBER	STATE			
0	1	0	0	0
1	0	1	0	0
2	0	0	1	0
3	1	0	0	1
4	1	1	0	0
5	0	1	1	0
6	1	0	1	1
7	0	1	0	1
8	1	0	1	0
9	1	1	0	1
10	1	1	1	0
11	1	1	1	1
12	0	1	1	1
13	0	0	1	1
14	0	0	0	1
15 = 0	1	0	0	0
16 = 1	0	1	0	0
	(REPEATS)			

FIGURE 3.3 *Successive states and output sequence from shift register (MacWilliams & Sloane)*

Note that the sequence has period 15 = 2^4 - 1, which is the maximum possible period. Therefore, the original polynomial $x^4 + x + 1$ is primitive.

For more information on the generation of binary (two-level) maximum shift register sequences using a shift register, see also Scholefield, R.E. (1960), Birdsall, T.G. and Ristenbatt, M.P.(1958), and Chew, P.E.K. (1964).

Finite field theory will now be used to explain what happens.

3.2 FINITE FIELDS

3.2.1 The Integer Quotient Ring

A *ring* (R, + ,∗) is an algebraic system consisting of a set of elements in which, on any two elements, the operations of addition, subtraction and multiplication yield another element which is always a member of the original set. It may be possible to divide in a ring, but in order to do this, the *multiplicative inverse* of the divisor must exist in the ring. Clearly the prototype of a ring is the set $\mathbb{Z}$ of integers; for a, b $\in$ **$\mathbb{Z}$,** we have a + b, a - b, a $*$ b $\in \mathbb{Z}$. Let $\mathbb{Z}_q$ denote the *quotient ring* of integers modulo an integer q: which consists of the set {0, 1, 2, ... , q-1} of integers and the result of every arithmetic operation is reduced modulo q.

An integer c maps into $\mathbb{Z}_q$ as the remainder r of c divided by q, e.g. c = r + dq, for some integer d. We write, c $\equiv$ r (mod q), read: c is *congruent* to r modulo q

Definition: Fix an integer q greater than 1 and let r be any integer. The **congruence class** of r **modulo** q is the set of all integers which are congruent to r modulo q:

$$[r]_q = \{b : b \equiv r \bmod q\}.$$

This type of mathematics is called *modular arithmetic* - only remainders modulo a given integer matter.

Example 3.1: When q is 2, there are exactly two congruence classes, namely $[0]_2$, the set of even integers, and $[1]_2$ the set of odd integers.

The multiplicative inverse of an element of $\mathbb{Z}_q$ exists if and only if the element is relatively prime to the modulus q[2]. In the case when q is a prime number, every nonzero element of $\mathbb{Z}_q$ has a multiplicative inverse, and division becomes a general operation in the ring. Then $\mathbb{Z}_q$ is called a *field.*

[2] If a $\in \mathbb{Z}_q$ and q are relatively prime, then, 1 = ab + dq $\equiv$ ab (mod q), where b and d are integers. The integer b mod q is then referred to as the multiplicative inverse of a under multiplication modulo q.

3.2.2 Fields

Definition A field F, is a set of elements with two operations *addition* (+) and *multiplication* ($*$) satisfying the following properties:

(1) F is closed under + and $*$, i.e. $a + b$ and $a * b$ are in F, whenever a and b are in F.

For all a,b and c in F, the following laws hold.

(2) Commutative Laws: $a + b = b + a$, $a * b = b * a$.

(3) Associative Laws: $(a + b) + c = a + (b + c)$, $(a * b) * c = a * (b * c)$.

(4) Distributive Law: $a * (b + c) = a * b + a * c$.

Furthermore, identity elements 0 and 1 must exist in F satisfying:

(5) $a + 0 = a$ for all a in F.

(6) $a * 1 = a$ for all a in F.

(7) For any a in F, there exists an additive inverse element (-a) in F such that $a + (-a) = 0$.

(8) For any $a \neq 0$ in F, there exists a multiplicative inverse element a^{-1} in F such that $a * a^{-1} = 1$.

It follows that a field is an algebraic structure in which the operations of addition, subtraction, multiplication and division (providing zero is not involved in division) can be performed without producing another quantity differing in kind from the other members of the collection.

A field can be defined more concisely to be a set F with two binary operations "+" and "$*$", such that:

(a) F is an abelian group under "+" , with identity element 0.
(b) The nonzero elements of F form an abelian group under "$*$".
(c) The Distributive Law holds.

A field is called finite or infinite, depending on whether the underlying set is finite or infinite. Examples of infinite fields include the real numbers, or the rational numbers.

Definition A finite field is a field which contains a finite number of elements, this number being called the *order* of the field.

The finite fields will now be designated by GF(q), where GF stands for Galois Field[3]. Galois Fields consist of the elements 0, 1, 2 ,......, q-1 , for which addition, subtraction, multiplication and division (except by 0) are defined, obeying the usual commutative, distributive and associative laws.

Example 3.2: Take GF(3), consisting of the elements 0, 1 and 2, we find $1 + 2 = 0$, $1 - 2 = 2$, $2 * 2 = 1$, $1/2 = 2$ etc.

Theorem 3.1: The number of elements q must be a power of a prime: $q = p^m$, p prime.

Proof: Let 1 denote the multiplicative identity in F. Define a sequence $\{u_0, u_1, u_2, \ldots\}$ in F as follows:

$$u_0 = 0, \quad u_n = u_{n-1} + 1, \text{ for } n = 1,2,\ldots\ldots$$

It follows from this definition that for arbitrary m and n,

$$(\dagger) \quad u_{m+n} = u_m + u_n$$

$$(\dagger\dagger) \quad u_{mn} = u_m * u_n$$

Now, due to the fact that F is finite, all the u_n's cannot be distinct; let $u_k = u_{k+c}$ be the first repeat, i.e., the elements $u_0, u_1, \ldots\ldots, u_{k+c-1}$ are all distinct but $u_{k+c} = u_k$. Then, since by (†), $u_{k+c} - u_k = u_c$, it follows that $u_c = 0$, but, $u_0 = 0$, and so 0 is in fact the first element in the sequence $\{ u_n \}$ to occur

[2] Evariste Galois (1811-32), was a French mathematician, born just outside Paris in the village of Bourg-la-Reine, who died in a duel at the age of 20 (Boyer,1991). His main interest concerned finding solutions to equations, such as quintic equations.

twice, and so the elements $\{u_0, u_1,......,u_{c-1}\}$are all distinct.

The integer c ($\geq$2) is called the *characteristic* of the field. We assert that c must be a prime. For if on the contrary $c = a * b$ with $1 \leq a \leq c$ and $1 \leq b \leq c$, then it follows from (††), that $u_c = u_a * u_b$. But $u_c = 0, u_a \neq 0$, $u_b \neq 0$, and so this is not possible, and conclude that c is, indeed, prime, and from now on replace the letter c by the letter p to remind us of that fact.

It is clear that the subset $\{u_0, u_1,......,u_{p-1}\}$of F is a subfield of F, since by (†) and (††) it is closed under "+" and "$*$". Indeed it is isomorphic to the field $F_p =$ **Z** Mod p = {0, 1, ... , p-1}, if we make the obvious identification $u_i \leftrightarrow i$. Thus it is possible to view F as a vector space over GF(p). Letting $\{\omega_1, \omega_2,......, \omega_m\}$ denote a basis (necessarily finite) for F over GF(p), we see that each element $\alpha \in$ F has a unique expansion of the form

$$(\dagger\dagger\dagger) \qquad \alpha = a_1\omega_1 + a_2\omega_2 + ...+ a_m\omega_m,$$

where each a_i is an element of GF(p). Since there are p possibilities for each a_i, it follows from (†††), that the field contains exactly p^m elements.

■

Theorem 3.1 sheds light on the additive structure of F, showing that the elements of F can be viewed as m-tuples of elements from GF(p), but tells very little about the multiplicative structure of F. The key fact is that the multiplicative group of F is cyclic with order q-1. Let $\alpha \in$ F be an arbitrary non-zero element of F. Consider the sequence of powers 1, α, α^2, ... , α^n, ... of α. Each power of α^i again lies in F. However, F contains only a finite number of elements, therefore, the sequence must repeat. Let $\alpha^k = \alpha^{k+t}$ be the first repeat in the sequence. Then clearly k = 0; otherwise $\alpha^{k-1} = \alpha^{k+t-1}$ would be an earlier repeat. Thus, $(1,\alpha, \alpha^2,..., \alpha^{t-1})$ are all distinct, but $\alpha^t = 1$.The integer $t \geq 1$ is called the *order* of α (McEliece,1987).

Theorem 3.2: If t is the order of α, then t divides q-1.

Proof: Let F* denote the set of nonzero elements of F. Then, F* is a multiplicative group with q-1 elements, and $\{1,\alpha, \alpha^2,..., \alpha^{t-1}\}$ is a subgroup with t elements. Lagrange's theorem tells us that the number of elements in a subgroup is always a divisor of the number of elements in a group; therefore, t divides q-1 as promised. ■

Lemma 3.3 If ord(α) = t, then ord (α^i) = t / gcd (i,t)

Proof: For any $\beta \neq 0$,

$\beta^s = 1$ if and only if ord (β) | s. ....(*)

Let d = gcd(i,t). Then $\alpha^{i(t/d)} = \alpha^{t(i/d)} = (\alpha^t)^{(i/d)} = 1$. Thus, by (*) ord ($\alpha^i$)| (t /d). Now, suppose s = ord (α^i). Then $\alpha^{is} = 1$ and so, by (*), t | is. Since d = gcd(i,t), ia + tb = d for certain integers a and b. Multiplying this equation by s, we obtain isa + tsb = ds. But since t | is, it follows that t | ds, i.e., (t/d) | s; i.e., (t/d) | ord (α^i). We have thus shown both ord (α^i) | (t/d) and (t/d) | ord (α^i). Hence, ord (α^i) = t/d as asserted. ■

Note: gcd stands for greatest common divisor, namely the largest number that is a common factor of the given numbers.

Definition: Let a_1, ... ,a_n be positive integers. Then, their greatest common divisor, (a_1, ... ,a_n) also written gcd (a_1, ... ,a_n), is the positive integer d with the property that $d|a_i$ for each i and, whenever c is an integer with $c|a_i$ for each i, we have c|d.

Example 3.3: Suppose ord(α) = 12, α being an element of some field F. We can compute the orders of α^i, i = 0, 1, ..., 11, using Lemma 3.3; the work is summarized below:

i	gcd(i,12)	ord(α^i)
0	12	1
1	1	12
2	2	6
3	3	4
4	4	3
5	1	12
6	6	2
7	1	12
8	4	3
9	3	4
10	2	6
11	1	12

This is a bit surprising! Given only the fact that there exists *at least* *one* element of order 12 in F, it follows that there are *exactly* 4 elements of order 12! Furthermore, there are exactly 2 elements of order 6, 2 of order 4, 2 of order 3, 4 of order 2 and 1 of order 1. ■

In order to generalize the result of Example 3.3, we introduce the symbol $\phi(t)$ to denote the number of integers in the set {0, 1, ..., t-1} which are relatively prime to t. This is *Eulers Phi Function*[4]. Euler introduced this function and described its elementary properties in his *Tractatus*. Note that since gcd(t,1) = 1 for all $t \geq 1$, then $\phi(t)$ is always at least 1. The value of $\phi(t)$ is somewhat unpredictable, but for future reference, we note that if t is prime, then $\phi(t)$ = t-1, since then every element in the set {0, 1, ..., t-1} except 0 is relatively prime to t.

Theorem 3.4 Let t be an integer, F a field. In F there are either no elements of order t, or exactly $\phi(t)$ elements of order t.

Proof: There is nothing to prove if there are no elements of order t. If ord(α) = t, then, as we observed above, every element of order t is in the set $\{1, \alpha, ..., \alpha^t\}$. By Lemma 3.3, α^i will have order t, if and only if gcd(i,t) = 1. By definition, the number of such i, is $\phi(t)$. ■

Combining Theorems 3.2 and 3.4, we see that if F is a finite field with q elements, and t is a positive integer, if t does not divide q-1, there are no elements of order t; but if t does divide q-1, there are either no elements of order t, or $\phi(t)$ elements of order t. Before stating the next Theorem, consider another example.

Example 3.4: Let q = 16. Then, q-1 = 15. It follows from Theorem 3.2 that the only possible values of t are t = 1, 3, 5, 15. For each of these values of t, the number of elements of order t (Using Theorem 3.4), is either 0 or $\phi(t)$. Computing $\phi(t)$ in each case, we have:

t	$\phi(t)$
1	1
3	2
4	4
15	8

[4] Leonhard Euler (1707-1783) born Basel, Switzerland. From 1727 to 1783 the pen of Euler had been busy adding knowledge in virtually every branch of pure and applied mathematics, from the most elementary to the most advanced. Our notations today are what they are more on account of Euler than any mathematician in history.

Note: The sum of the numbers in the $\phi(t)$ column equals 15. The next Theorem illustrates that this remarkable occurrence is no accident.

Theorem 3.5 If n is any positive integer, then

$$\sum_{d|n} \phi(d) = n,$$

this notation indicates that the summation is to be extended over all positive divisors of n.

Theorem 3.6 Let F be a finite field with q elements, and let t be a positive integer. If t does not divide q-1, there are no elements of order t in F. If t | q-1, there are exactly $\phi(t)$ elements of order t in F.

Corollary In every finite field, there exists at least one element of order q-1. In fact, exactly $\phi(q-1)$ elements. Hence, the multiplicative group of any finite field is cyclic.

Definition An element of multiplicative order q-1, i.e. a generator of the cyclic group $F^* = F - \{0\}$, is called a *primitive root* of the field F.

3.2.3 Polynomials

A polynomial p(x) of degree m over a field F is as follows:

$$\sum_{k=0}^{m} a_k x^k, \quad m \geq 0, \quad a_m \neq 0$$

where coefficients a_k are elements of the number field F. If the leading term $a_m = 1$, the polynomial is said to be *monic.*

Definition An irreducible polynomial of degree m over GF(p) is a polynomial $\Sigma_{0<i<m}\ (a_i x^i)$ that cannot be expressed as the product of two polynomials each of smaller degree, with coefficients in GF(p).

Example 3.5: In the Finite Field GF(2), $x^2 + x + 1$ is irreducible, but $x^2 + 1$ is not since $(x+1)(x+1) = x^2 + 2x + 1 \equiv x^2 + 1 \pmod 2$.

Theorem 3.7: Suppose F is a field with p^m elements. Associated with each $\alpha \in F$, there is a unique monic polynomial $p(x) \in F_p(x)$, with the following properties:

(a) $p(\alpha) = 0$,

(b) $\deg(p) \leq m$

(c) If $f(x)$ is another polynomial in $F_p(x)$ with $f(\alpha) = 0$, then $p(x) \mid f(x)$.

The polynomial described above is the minimal polynomial of α with respect to the subfield GF(p) of F.

Theorem 3.8 Let F be a field with q^m elements, and let k be a q-element subfield. If $\alpha \in F$, then the minimum polynomial of α with respect to the subfield k is

$$f_\alpha(x) = (x - \alpha)(x - \alpha^2) \ldots\ldots (x - \alpha^{q\wedge(d-1)})$$

where d = degree of α with respect to k.

In general, the minimum polynomial of a primitive root in F is called a *primitive polynomial.*

A polynomial can be tested to see if it is primitive using the MATHEMATICA function:

Irreducible Q [p_,m_] : = Same Q [Factor [p, Modulus → m], p];

We shall now construct finite number fields of order equal to a prime power $q = p^m$, designated $GF(p^m)$. All realizations of $GF(p^m)$ are isomorphic. Since $GF^*(p^m)$ ($GF(p^m)$ without the zero element) forms a cyclic group with multiplication as the group operation, we can also represent $GF^*(p^m)$ which has order p^m-1, by a primitive element α and it's p^m-1 distinct powers $\alpha, \alpha^2, \ldots, \alpha^{(p\wedge m)-1} = 1$.

From now on, this thesis will exclusively be concerned with finite fields of characteristic 2.

The finite field GF(2^m) has 2^m elements, where m is a positive integer. Each of the field elements can uniquely be represented with a polynomial of degree up to m-1 with coefficients from GF(2). For example, if a is an element in GF(2^m), then we can have

$$a = A(x) = a_{m-1}\, x^{m-1} + a_{m-2}\, x^{m-2} + \ldots. + a_1 x + a_0 \,.$$

This type of representation is referred to as the *polynomial* or *standard basis* representation.

One of the simplest examples of a finite field is GF*(2^4).*

3.2.4 Construction of GF(2^4)

The field elements can be represented as either 4-tuples (vectors) of 0's and 1's, or polynomials of degree 3, all with coefficients mod 2. The reason 4-tuples are used is a direct consequence of the 4 in GF(*2*^{4}). There are exactly 2^4 such polynomials or 4-tuples. To see this, associate a polynomial in α with each 4-tuple as illustrated in Table 3.1.

4-tuple	*polynomial in α*
0000	0
0001	1
0010	α
0011	$\alpha + 1$
0100	α^2
....	....
1111	$\alpha^3+\alpha^2+\alpha+ 1$

Table 3.1: Association between polynomials in *α* and 4-tuples

Note that subtraction is the same as addition (mod 2), since $a_0a_1a_2a_3 + a_0a_1a_2a_3 = 0$.

Before constructing the field with 16 elements, multiplication has to be defined. Multiplying two polynomials from Table 3.1, will often result in a polynomial which has degree greater than 3, which is not in our set of objects, for example,

$$(1+\alpha^2)(1+\alpha+\alpha^2+\alpha^3) = 1+\ \alpha+\alpha^4+\alpha^5 \text{ (modulo 2)} \qquad ....(*)$$

When this occurs, the aim is to reduce the degree of the polynomial to ≤3. In order to do this, we agree α satisfies a fixed equation h(x) = 0 of degree 4 (MacWilliams & Sloane, 1977). For example, take our primitive polynomial, defined earlier,

$$h\,(\alpha) = \alpha^4+\alpha+1 \quad \text{eg. } \alpha^4 = \alpha+1 \ \textit{(mod 2)}$$

Then replace α^4 and α^5 in (*) by $\alpha^4 = \alpha+1;\ \ \alpha^5 = \ \alpha\,(\alpha^4) = \alpha(\alpha+1) = \alpha^2+\alpha$
Thus, $1+\ \alpha+\alpha^4+\alpha^5\ =\ 1+\alpha+\alpha+1+\alpha^2+\alpha = \ \alpha^2+\alpha$ (modulo 2), which now belongs to our set of objects.

To construct the finite field GF(2^4) a primitive polynomial of degree 4 is needed. For this specific example, there are two choices of primitive polynomials - it does not matter which one we choose. Make use of the primitive polynomial h(x) = x^4 + x + 1, and the primitive element α = 0010 = x. Start with the 0 element, and the 1 element. Proceed by multiplying by α (which corresponds to a left-shift in the m-tuple), and residue reduction modulo h(x) = x^4 + x + 1 (Schroeder,1986). This reduces the products to a polynomial of degree ≤ 3, equivalent to setting h(x) = 0, so that x^4 = x + 1 (eg. x^4 can be replaced by x + 1) When a 1 disappears off the left-hand side of the 4-tuple, it corresponds to adding 1's to the two right-hand places of the 4-tuple, as can be seen in Table 3.2.

This is just a rule method for producing the states of a feedback shift register. Many people find it easier to conceptualize if you have a shift register.

i	α^i
$-\infty$	0000
0	0001
1	0010
2	0100
3	1000
4	0011
5	0110
6	1100
7	1011
8	0101
9	1010
10	0111
11	1110
12	1111
13	1101
14	1001
15	0001

TABLE 3. 2: GF(2^4)

Reading the right-hand side of column 2 (for the elements 0 to 15) gives an m-sequence of length 15 (2^4-1). For more information on Finite Fields, see for example, Jungnickel (1993), Lidl & Niederreiter (1994), or Biggs (1989).

The Delphi code, which can be used to generate any m-sequence of length 7 to 32767, from a given primitive polynomial can be found in Appendix A.

Chapter 4

PROPERTIES OF PSEUDO RANDOM BINARY SEQUENCES

These sequences have interesting mathematical properties. They are important because they are easily generated binary sequences that behave in many respects as if their elements were chosen completely at random.

4.1 PROPERTIES OF M-SEQUENCES

Perhaps the simplest property refers to the m-tuples of an m-sequence. If the m-sequence in question is $(a_0, a_1, \ldots , a_{n-1})$, then an m-tuple is one of the n subsequences of length m, of the form

$$(a_t, a_{t+1}, \ldots , a_{t+m-1}), \quad \text{for } t = 0, 1, \ldots, n-1. \qquad (*)$$

where the subscripts in (*) are taken mod n if necessary. There are $n = 2^m - 1$ different m-tuples. An m-sequence includes in one period all possible m-tuples (except the all-zero m-tuple) for some fixed m.

Property 4.1: Among the 2^m -1 m-tuples of an m-sequence (a_t), each non-zero binary vector of length m occurs once and only once.

Proof: All the m-tuples are distinct, since a repeated m-tuple would cause (a_t) to repeat sooner than period n, because of the degree m recurrence relation. The all zero m-tuple cannot occur, because if it did, the sequence (a_t) would continue to be zero because of the degree m recurrence relation.

This is often referred to as the '*Window Property*', and is represented pictorially in Figure 4.1 overleaf, for the case m = 4. In general, if a window of length m is slid along the m-sequence, each of the 2^m -1 non-zero binary m-tuples can be seen exactly once.

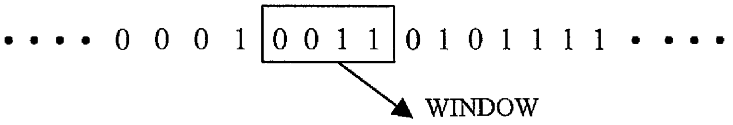


Figure 4.1 *The Window Property: Every non-zero 4-tuple is seen once.*

To avoid difficulties at either end, imagine the sequence is repeated.

Note that 0 0 0 1 was used as the initial register state to obtain the above sequence. If we started with a different non-zero initial state in the feedback shift register illustrated in Figure 3.3, the output sequence is just a shifted version of the m-sequence displayed in Figure 4.1. Therefore, the same m-sequence is obtained from any non-zero starting state, as you would expect from Property 1. This is illustrated in Figure 4.2.

```
0 0 0 1 0 0 1 1 0 1 0 1 1 1 1
0 0 1 0 0 1 1 0 1 0 1 1 1 1 0
0 1 0 0 1 1 0 1 0 1 1 1 1 0 0
1 0 0 1 1 0 1 0 1 1 1 1 0 0 0
0 0 1 1 0 1 0 1 1 1 1 0 0 0 1
0 1 1 0 1 0 1 1 1 1 0 0 0 1 0
1 1 0 1 0 1 1 1 1 0 0 0 1 0 0
1 0 1 0 1 1 1 1 0 0 0 1 0 0 1
0 1 0 1 1 1 1 0 0 0 1 0 0 1 1
1 0 1 1 1 1 0 0 0 1 0 0 1 1 0
0 1 1 1 1 0 0 0 1 0 0 1 1 0 1
1 1 1 1 0 0 0 1 0 0 1 1 0 1 0
1 1 1 0 0 0 1 0 0 1 1 0 1 0 1
1 1 0 0 0 1 0 0 1 1 0 1 0 1 1
1 0 0 0 1 0 0 1 1 0 1 0 1 1 1
```

Figure 4.2 *The 15 m-sequences obtained from the shift register in Figure 3.3*

Now, define h(x) to be a primitive polynomial of degree m, and let δ_m be the set consisting of all the shifts of a pseudo random sequence of length $2^m - 1$, obtained from the output of the shift register specified by h(x), together with the all-zero sequence of length 2^m - 1. These pseudo-random sequences are the 2^m - 1 different segments

$a_i\, a_{i+1}$ ...... $a_{i+(2^\wedge m)-2}$ $i = 0, 1, \ldots, 2^m - 2$

of length 2^m - 1 from the output of the shift register specified by h(x). For example δ_4 consists of the rows of Figure 4.2, and the all-zero sequence of length 15.

Property 4.2 is often referred to as the '*Shift Property*'.

Property 4.2: If **b** = $b_0, b_1, \ldots, b_{(2^\wedge m)-2}$ is any of the pseudo-random sequences in δ_m then any cyclic shift of b, say

$b_j, b_{j+1}, \ldots, b_{(2^\wedge m)-2}, b_0, \ldots\ldots, bj\text{-}_1$

is also in δ_m.

Property 4.3: Suppose

$$h(x) = \sum_{i=0}^{m} h_i x^i$$

with $h_0 = h_m = 1$, $h_i = 0$ or 1 for $0 < i < 1$. Any pseudo-random sequence **b** ϵ δ_m satisfies the recurrence

$b_{i+m} = h_{m-1}\, b_{i+m-1} + h_{m-2}\, b_{i+m-2} + \ldots + h_1\, b_{i+1} + b_i$, for i = 0,1,2, ... (*)

Conversely, δ_m contains any solution of (*). If all of the 2^m - 1 distinct non-zero initial values, $b_0, b_1, \ldots, b_{m-1}$, are used in (*), the 2^m - 1 pseudo-random sequences are obtained. There are m linearly independent sequences in δ_m, since there are m linearly independent solutions to (*).

Property 4.4: The sum of two sequences in δ_m is another sequence in δ_m. (Addition is componentwise, mod 2, without carries).

Proof: Follows from the fact that any sum of two solutions of (*) in Property 4.3, is also a solution.

Example 4.1: Consider the first two sequences in Figure 4.1. Adding them together gives:

$$\begin{array}{rc}
 & 0\ 0\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ 1\ 1 \\
+ & 0\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ 1\ 1\ 0 \\
\hline
 & 0\ 0\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ 1\ 1\ 0\ 0\ 0\ 1 \\
\hline
\end{array}$$

which turn out to be the fifth sequence in Figure 4.1.

An important property of m-sequences, known as the '*Shift & Add Property*', is that the termwise addition of phase-shifted versions of a given m-sequence produces another phase-shifted version of the same sequence

Property 4.5: The sum of any pseudo-random sequence and a cyclic shift of itself is another pseudo-random sequence.

Proof: Follows from Properties 4.2 & 4.4

Property 4.6: *Half Zeros and Half Ones:* Any pseudo random sequence in δ_m contains 2^{m-1} ones and 2^{m-1} - 1 zeros. Thus in one period, the number of logic 1 states exceeds the number of logic 0 states by one.

This is the case because if we consider numbers in the range 1 to 2^m - 1, we find that there are 2^{m-1} odd numbers with binary representation ending in 1 and 2^{m-1} - 1 even numbers in the same range.

Note: Probability associated with each state approaches 0.5 as sequence length increases. Results similar to that of an experiment involving the tossing of a coin.

The '*Run-Distribution Property*' is one of the most remarkable properties of m-sequences.

Definition: A *run of length r* in a binary sequence is a subsequence of exactly *r* consecutive 1's (or 0's).

Example 4.2: Consider one of the m-sequences in Figure 4.2, e.g.

0 0 0 1 0 0 1 1 0 1 0 1 1 1 1

If this sequence is viewed cyclically, it has two 0-runs of length 1, two 1-runs of length 1, one 0-run of length 2, etc. A histogram of the runs can be viewed below:

length	0-runs	1-runs
1	2	2
2	1	1
3	1	0
4	0	1
Totals:	4	4

Example 4.3: Given the following m-sequence of length 31,

0 0 0 0 1 0 0 1 0 1 1 0 0 1 1 1 1 1 0 0 0 1 1 0 1 1 1 0 1 0 1

generated from a primitive polynomial $x^5 + x^2 + 1$, of degree 5, the histogram of the runs is as follows:

length	0-runs	1-runs
1	4	4
2	2	2
3	1	1
4	1	0
5	0	1
Totals:	8	8

Out of a total of 16 runs, half have length 1, one quarter have length 2 and one eighth have length 3. This is what you would hope to obtain in a completely random sequence of 0's and 1's, but to obtain exactly the average for any chosen sequence is highly improbable. However, in the case of m-sequences, this nice behavior holds for any chosen sequence. The general histogram for an m-sequence of length $2^m - 1$ is as follows:

length	0-runs	1-runs
1	2^{m-3}	2^{m-3}
2	2^{m-4}	2^{m-4}
:	:	:
r	2^{m-r-2}	2^{m-r-2}
:	:	:
m-2	1	1
m-1	1	0
m	0	1 .
Totals:	2^{m-2}	2^{m-2}.

Property 4.7: The previous table illustrated the run distribution for any m-sequence of length $2^m - 1$.

Proof: Let $(a_0, a_1, \ldots\ldots, a_{n-1})$, where $n = 2^m - 1$, denote the m-sequence in question. Using Property 4.1, we can count the various run lengths in am m-sequence by distinguishing three different cases:

- Length m: There are no 0-runs of length m, and only one 1-run of length m -this follows from Property 4.1

- Length m-1: Again from Property 4.1, we know that the all-1's m-tuple, 111....11111 appears exactly once. It has to occur sandwiched between two 0's; otherwise the all-1's m-tuple would appear more than once. Thus the subsequence 0111....111110 of length m+2 appears somewhere in the sequence, resulting in the appearance of the two m-tuples 0111...11111 and 111...111110. If there were a separate 1-run of length m-1 somewhere, it would be sandwiched between 0's, leading to the subsequence 011.....111110, resulting in the appearance of the m-tuples 0111...11111 and 111...111110 again. Property 4.1 tells us that these can only occur once each, therefore, there can be no separate 1-run of length m-1.

  There is however, one 0-run of length m-1, because of the m-tuples 1000...00000 and 00000...0001, which occur together as 1000...00001.

- Length $r \leq m-2$: Each 1-run of length $r \leq m-2$ corresponds to an m-tuple of the form

$$\overbrace{0\ \ \underbrace{1\,1\,1\,1\,1\,1\ldots 1}_{r}\ \ 0\ \underbrace{x\,x\ldots x\,x}_{m-r-2}}^{m}$$

Note that the x's denote arbitrary binary digits. Clearly, there are 2^{m-r-2} such m-tuples, and therefore, 2^{m-r-2} 1-runs of length r.

A similar argument proves that there are 2^{m-r-2} 0-runs of length r. ■

Appendix 2 illustrates the code written in Delphi to generate the m-sequence and then check its integrity.

The important '*Correlation Properties*' of the sequences also have to be taken into account.

Definition: Given two binary sequences **x** $=(x_1,x_2,\ldots.,x_n)$, **y** $=(y_1,y_2,\ldots.,y_n)$ of length n, their *correlation, C(x,y),* is defined to be the number of agreements minus the number of disagreements between x and y, divided by n. That is

$$\mathbf{C(x,y) = (A - D) / n}$$

$$\textbf{where}\quad \mathbf{A = |\{i : x_i = y_i\}|\ and\ D = |\{i : x_i \neq y_i\}|}$$

The correlation is a measure of similarity between x and y. If x and y are identical, then C(x,y) = 1, and if the disagree in every component then C(x,y)= -1. In every case, $-1 \leq C(x,y) \leq 1$.

Autocorrelation is a statistical measure of dependence.

Definition: Given a fixed sequence x, its *autocorrelation function $C(\tau)$* is defined to be the correlation between x and its τth cyclic shift.

Let A be the number of places where x and its τth cyclic shift agree, and D the number of places they disagree (so A+D=n). Then,

$$\mathbf{C(\tau) = (A-D) / n \quad \ldots\ldots\ldots(**)}$$

Property 4.8: The autocorrelation function of a pseudo-random sequence of length $n = 2^m-1$ is given by

$$\mathbf{C(0) = 1}$$

$$\mathbf{C(\tau) = -1/n \qquad \text{for } 1 \le \tau \le 2^m-2}$$

Proof: For a + a' = a" for some ", by the Shift & Add Property.

Then, D = Number of 1's in a' = 2^{m-1}, by Property 4.6.

A = n - D = 2^{m-1} -1, and the result follows from (**) in Definition above. ■

4.2 MULTIPLIERS OF PSEUDO-RANDOM BINARY SEQUENCES

4.2.1 Introduction

A pseudo-random binary sequence, for the purpose of this discussion, is defined to be a maximum length linear recurring sequence modulo 2. For example, a_k is a pseudo-random binary sequence if and only if it is a binary sequence which satisfies a linear recurrence

$$a_k = \sum_{i=1}^{m} c_i\, a_{k-i} \quad \text{(modulo 2)} \ldots\ldots\ldots (1)$$

and has period 2^m - 1. The number m is referred to as the degree of the pseudo-random binary sequence a_k.

The polynomial

$$f(x) = 1 + \Sigma\, c_i\, x^{\,i} \qquad \text{(modulo 2)} \ldots\ldots\ldots(2)$$

is called the characteristic polynomial of the sequence a_k of Equation (1). The primitivity of f(x) is a necessary and sufficient condition for a_k to be a pseudo random binary sequence.

4.2.2 Delay Unit Operator

Let D be a unit delay operator, so that $Da_k = a_{k-1}$, and $D^2a_k = a_{k-2}$. Then, by equations (1) and (2),

$$fD\{a_k\} = 0 \qquad ......(3)$$

is equivalent to the recursion relation in (1) satisfied by a_k.

Amongst polynomials modulo 2, $f(x^2) = [f(x)]^2$, and more generally, $f(x^{2^\wedge i}) = [f(x)]^{2^\wedge i}$, in view of the simplified binomial theorem

$$(a + b)^{2^\wedge i} = a^{2^\wedge i} + b^{2^\wedge i} \text{ (modulo 2) } ....(4)$$

In general, the values of j other than a power of 2, do not satisfy $f(x^j) = [f(x)]^j$.

Now, consider the sequence a_{2k}, which is formed by taking every alternate term from the sequence a_k. Therefore,

$$f(D^2)\{a_k\} = f(D)\{a_{2k}\}, \quad ....(5)$$

where the two sequences differ at most by a fixed translation. However, $f(D^2)\{a_k\} = f(D)[\, f(D)\{a_k\}] = f(D)\{0\} = \{0\}$. Hence,

$$f(D)\{a_{2k}\}= 0 \qquad ....(6)$$

so that a_{2k} satisfies the same recursion formula as a_k. Therefore, a_{2k} is identical to a_k, except for a possible phase shift, and we have proved the following property.

Property 4.9 If a_k is a pseudo-random binary sequence, then a_{qk} equals a_k except for a possible phase shift, when q = 1, 2, 4,....., 2^{m-1}.

Note: the numbers 1, 2, 4,....., 2^{m-1} are called the *multipliers* of the sequences a_k. Collectively, they are known as the multiplier group because they form a group under multiplication.

Example 4.4: Take an m-sequence of length 15:

a_k = 0 0 0 1 0 0 1 1 0 1 0 1 1 1 1

Take every 2nd element a_{2k} = 0 0 0 1 0 0 1 1 0 1 0 1 1 1 1

Take every 4th element a_{4k} = 0 0 0 1 0 0 1 1 0 1 0 1 1 1 1

Take every 8th element a_{8k} = 0 0 0 1 0 0 1 1 0 1 0 1 1 1 1

Note: that replacing a_k by a_{2k} in the example above, does not alter the order of the terms. Similarly for a_{4k} and a_{8k}.

While,

Take every 3rd element a_{3k} = 0 1 1 1 1 0 1 1 1 1 0 1 1 1 1

Note: that if a_k is replaced by a_{3k} this gives rise to a different sequence - this time with period 5.

Example 4.5: Let a_k = 0 0 1 0 0 1 1 0 1 0 1 1 1 1 0

Take every 2nd element a_{2k} = 0 1 0 1 1 1 1 0 0 0 1 0 0 1 1

Take every 4th element a_{4k} = 0 0 1 1 0 1 0 1 1 1 1 0 0 0 1

Take every 8th element a_{8k} = 0 1 0 0 1 1 0 1 0 1 1 1 1 0 0

Note: that replacing a_k by a_{2k} in the example above, does not alter the order of the Terms, only the start location. Similarly for a_{4k} and a_{8k}.

While,

Take every 3rd element a_{3k} = 0 0 1 0 1 0 0 1 0 1 0 0 1 0

4.2.3 Cyclotomic Cosets

Eulers Theorem tells us that there are ϕ(p) numbers from 1 to p which are relatively prime to p. The ϕ(p) numbers form a group *G* under multiplication modulo p. If p is odd, then the set (1, 2, 4, 8,...) forms a subgroup *H*.

Note: In the case p = 2^m-1 , the multiplier subgroup *H*, consists of the *m* elements (1, 2, 4,....., 2^{m-1}).

Definition: Let *H* be a subgroup of *G*, and let *a* be any element of *G*. Define *aH* to be the set of all elements of *G* which may be written as *ah* for some element *h* in *H*: $aH = \{ah : h \in H\}$. This is a (left) **coset** of *H* (in *G*). Similarly, define the (right) coset $Ha = \{ha : h \in H\}$.

Therefore, a coset is obtained by taking any element of the large group *G* and multiplying it by each number of the subgroup *H* in turn.

Example 4.6: Take p = 31, the cosets of the multiplier subgroup are:

C_1:	1	2	4	8	16
C_2:	3	6	12	24	17
C_3:	9	18	5	10	20
C_4:	27	23	15	30	29
C_5:	19	7	14	28	25
C_6:	26	21	11	22	13

The number of cosets is always $\phi\,(2^m - 1)\,/\,m$, which in the case of m = 5 (above) yeilds $\phi\,(31)\,/\,5 = 6$.

4.2.4 Decimation of Sequences

Definition: In a pseudo-random binary sequence, if a_k is replaced by a_{2k}, it does not alter the order of the terms in the sequence, except perhaps, the location of the starting point. More generally, a_{qk} is the same as a_k for q = 1, 2, 4,....., 2^{m-1}. (The replacement of a_k by a_{qk} is termed *decimation*).

Theorem 4.1 If a_k is a pseudo-random binary sequence with period p, then a_{qk} is again a pseudo-random binary sequence, with the same period if and only if $(q,p) = 1$. If both $(q_1,p) = 1$ and $(q_2,p) = 1$, then $a_{(q_1)}k = a_{(q_2)}k$ (except for the starting point) if and only if q_1 and q_2 belong to the same cyclotomic coset.

Proof: Omitted

4.2.5 The Superposition of Cosets

Let a_k be a pseudo-random binary sequence. Then a_{2k} is simply a phase shift of a_k by Property 4.9, so that termwise

$$a_{2k} = a_{k+\tau}$$

for someτ. From now on, the notation $a_k = b_k$ will be used to illustrate term-by-term equality.

Lemma 4.2 There is a phase shift $b_k = a_{k+m}$ of a_k, for some m, such that $b_{2k} = b_k$.

Proof: Suppose originally, that $a_{2k} = a_{k+\tau}$. Choose $m = 2\tau$, so that $k' = k + 2\tau$. Then $b_{2k} = a_{2k+m} = a_{2k+2\tau} = a_{2(k+\tau)} = a_{(k+\tau)+\tau} = a_{k+m} = b_k$. ■

Example 4.6: Let $a_k = 0\ 0\ 1\ 1\ 1\ 0\ 1$

Then, $a_{2k} = 0\ 1\ 1\ 1\ 0\ 1\ 0 = a_{k+1}$

Hence $b_k = a_{k+2} = \{1\ 1\ 1\ 0\ 1\ 0\ 0\}$ satisfies $b_{2k} = b_k$.

Chapter 5

PSEUDO RANDOM BINARY SEQUENCES RELATED TO MULTIFOCAL ELECTRORETINOGRAPHY

The basis of the Multifocal ERG is the use of a decimated m-sequence for simultaneous and independent stimulation of many areas of the visual pathway.

5.1 DECIMATION OF M-SEQUENCE

Once the m-sequence has been generated from the shift register, specified by the primitive polynomial (satisfying the recurrence relation in property 4.3), it has to be decimated in order to stimulate many areas simultaneously, without causing the first-order response to cross-contaminate. Decimating the sequence results in a time lag between the initial sequence starting in each column (Property 4.9). A distinct shift is used to drive each hexagonal stimulus element.

A shifted version of the initial m-sequence is now contained in each column. Due to the statistical random properties of an m-sequence (Property 4.8), each shifted m-sequence will be uncorrelated with any other shifted cycle of the same sequence.

Before we continue, the trace function has to be defined.

Definition: Let F = GF(q), K = GF(q^m). If α is an element of K, its *trace relative to the subfield F* is defined as follows:

$$\mathrm{Tr}_F{}^K(\alpha) = \alpha + \alpha^q + \alpha^{q^\wedge 2} + \ldots.. + \alpha^{q^\wedge(n-1)}$$

Theorem 5.1: For all $\alpha, \beta \in$ K we have

(a) Tr $(\alpha) \in$ F
(b) Tr$(\alpha + \beta)$ = Tr (α) + Tr (β)
(c) Tr$(\lambda\alpha) = \lambda$ Tr (α) if $\lambda \in$ F
(d) Tr (α^q) = Tr (α)
(e) Tr maps K onto F

Theorem 5.2: If a binary m-sequence is decimated over n columns, where n is a power of 2, then, each column contains the m-sequence with a relative lag between columns of 2^m / n, and a lag of $(2^m / n)$ -1 between the first and last column.

Proof: Let $s_i = \mathrm{Tr}(a^i)$ where a is primitive in GF(2^m). So the sequence $s_0, s_1, \ldots$ has period 2^m-1 - it's an m-sequence in characteristic phase.

Our array on decimation of s has elements $t_{i,j}$ where $t_{i,j} = s_{i+jd}$ and d is a power of 2, the decimation. Here i and j range over some values and i =0 gives us the first column, i =1 the second and so on.

So $t_{i,j} = s_{i+jd} = \mathrm{Tr}(a^{\,i+jd}) = \mathrm{Tr}(a^i \,.\, (a^d)^j)$.

Let's fix i for a moment. Write g = a^d. Because d is a power of 2, it is co-prime to 2^m-1. Then g is also primitive in GF(2^m). Write b = a^i. Then with i fixed and j varying we get a sequence whose terms are Tr(b g^j) - this is the column i of the array.

But this is an m-sequence - the one corresponding to primitive element g, but at a shift determined by b. But g = a^d where d is a power of 2. This means that g and the original primitive element a actually produce the same m-sequence (because Tr(g^j) = Tr(a^{dj}) = Tr(($a^j)^d$) = Tr(a^j), so the j-th terms are equal).

So column i is just a shift of our original sequence s. Which shift is it? Well we have terms Tr(b g^j), so the shift is determined by b. Writing B = g^x, the shift is equal to x. But g^x = b = a^i implies $a^{dx} = a^i$ which in turn implies dx = i mod (2^m-1). So x = i d^{-1} mod 2^m-1. But d is a power of 2, say d = 2^e with $0 \le e < m$.

And now it's easy to see that $d^{-1} = 2^{m-e}$ mod 2^m-1 (check by multiplying). So x = i 2^{m-e} mod 2^m-1.

Summarising, column i is a shifted version of s by an amount x = i (2^m/d) mod 2^m-1. For i = 0 we get x = 0, i.e. column 0 is just s. For i =1, we get x = 2^m/d, etc. ■

Table 5.1 illustrates how this works for a simple example, a 4-bit m-sequence 1 0 0 0 1 0 0 1 1 0 1 0 1 1 1 of length 15. This sequence is decimated over 4 columns.

Table 5.1: M-sequence decimated over 4 columns

STAGE	*Area 0*	*Area 1*	*Area 2*	*Area 3*
0	**1**	0	0	0
1	1	0	0	1
2	1	0	1	0
3	1	1	1	**1**
4	0	0	0	1
5	0	0	1	1
6	0	1	0	1
7	1	1	**1**	0
8	0	0	1	0
9	0	1	1	0
10	1	0	1	1
11	1	**1**	0	0
12	0	1	0	0
13	1	1	0	1
14	0	1	1	1

Imagine only 4 areas from Figure 2.4a. Each of the four areas is now controlled by one of the sequences in Table 5.1, and we are able to derive the response to the individual regions independently by use of cross-correlation. The m-sequence is cross-correlated with the electrical response from the eye, to obtain the physiological response.

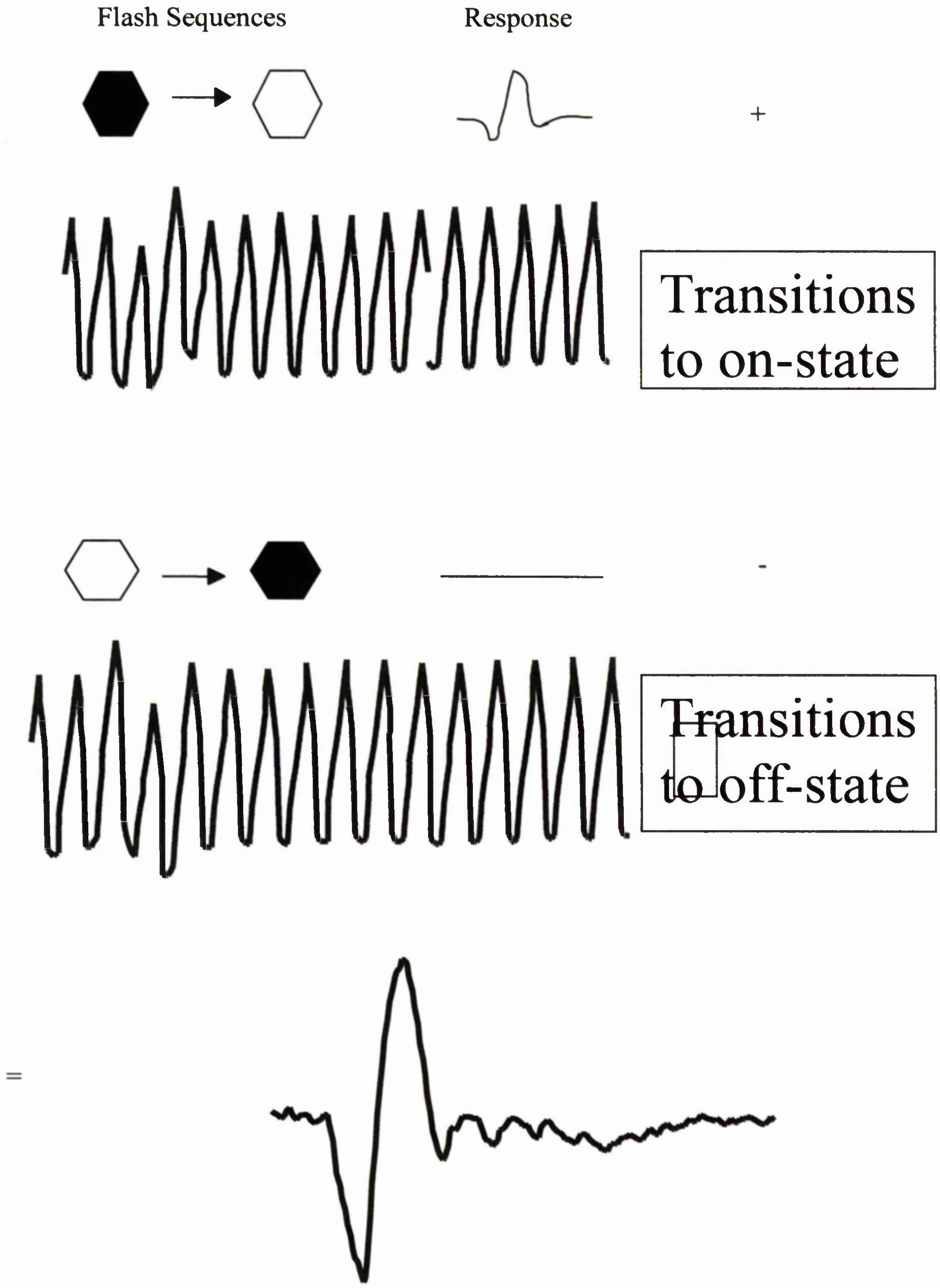
In reality, a 15-bit m-sequence (length 32767), decimated over 128 columns, is used to drive the multifocal stimulator. For example, consider Figure 2.4a. Each of these hexagonal areas will flash black or white depending on whether the sequence is 0 or 1 respectively.

If we have a 15-bit m-sequence, then any 15 bit segment will occur only once (Property 4.1). 15-bit m-sequence steps at the standard presentation rate of 75Hz, or a base period of 13.33 msecs will give a time period of 15 x 13.33 msecs or approximately 200 msecs. This is longer than the full duration of the ERG response, which lasts a maximum of around 150 msecs and should therefore be less susceptible to contamination. Signal to noise ratios are also better for 15-bit sequences, thus giving clearer responses.

5.2 GENERATION OF FIRST ORDER RESPONSE

The correlation between the binary m-sequence and the response from the retina is referred to as the First Order Response. Figure 5.1 illustrates how the response is obtained by subtracting the transitions to "Off" states from the transitions to "On" states. This is because the number of logic 1 states exceeds the number of logic 0 states by 1 - making use of Property 4.6.

Figure 5.1: *Generation of First Order Response*

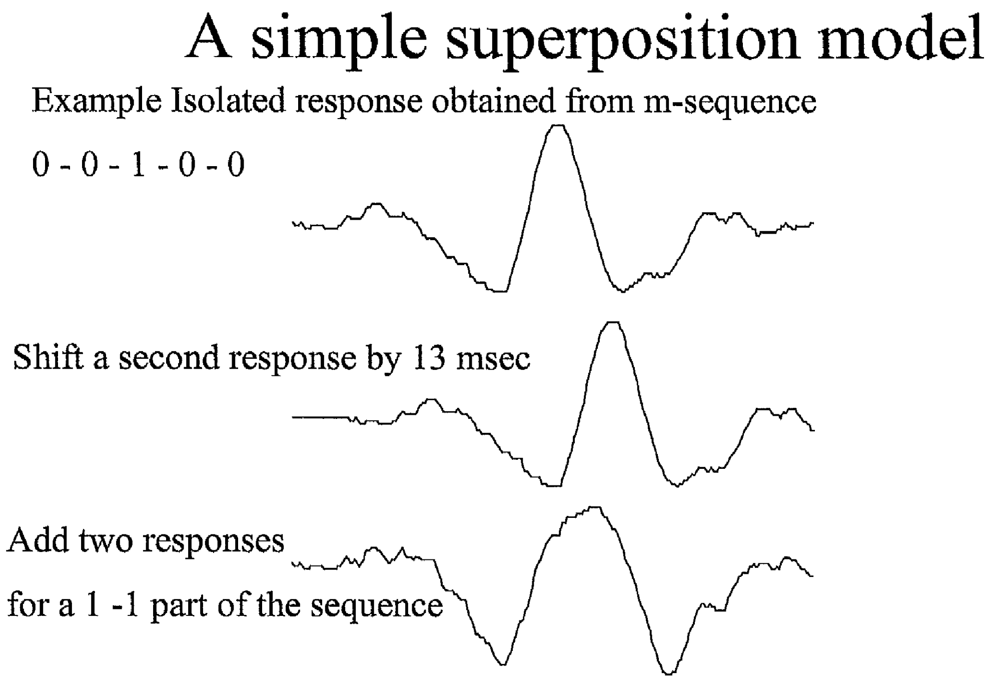


5.3 A SIMPLE SUPERPOSITION MODEL

The run-distribution properties (Property 4.7) are important to enable the responses to be extracted from the eye. In a 15-bit m-sequence of length 32767,

1 - 1	(75 Hz)	occurs 8192 times
1 - 0 - 1	(37.5 Hz)	occurs 4096 times
1 - 0 - 0 - 1	(25 Hz)	occurs 2048 times
1 - 0 - 0 - 0 - 1	(18.75 Hz)	occurs 1024 times
1 - 0 - 0 - 0 - 0 - 1	(15 Hz)	occurs 512 times

Figures 5.2 and 5.3 illustrates how to extract the individual responses, and Figure 5.4 illustrates how to construct the global First Order Response from the eye. This uses properties 4.2, 4.4 and 4.5.



A simple superposition model

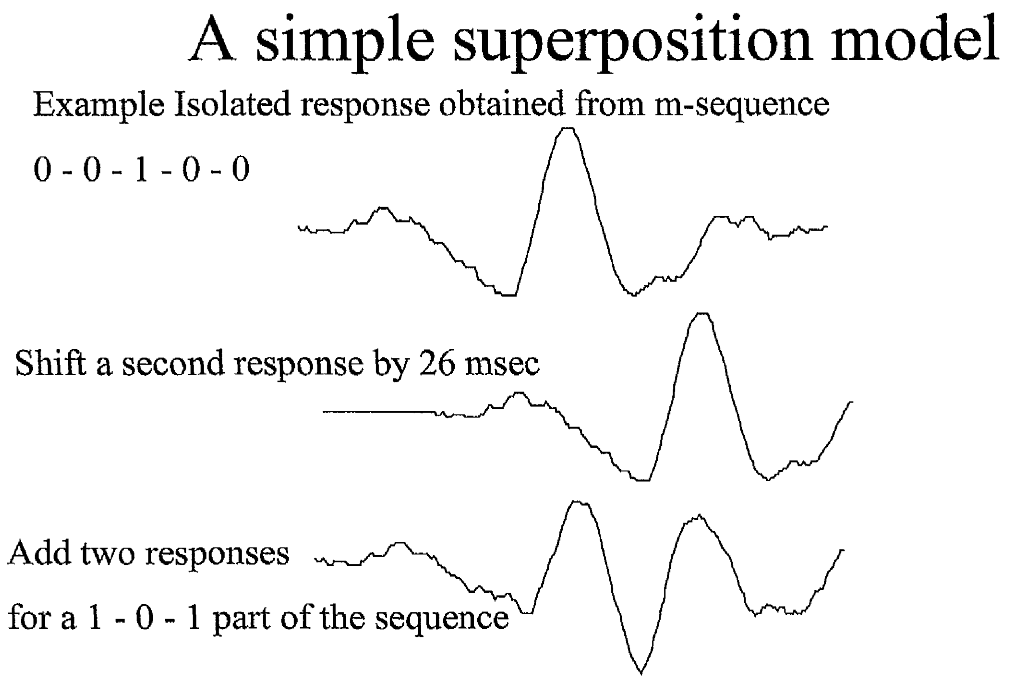
Example Isolated response obtained from m-sequence

0 - 0 - 1 - 0 - 0

Shift a second response by 13 msec

Add two responses
for a 1 -1 part of the sequence

Figure 5.2: *Extraction of Waveforms (1)*



A simple superposition model

Example Isolated response obtained from m-sequence

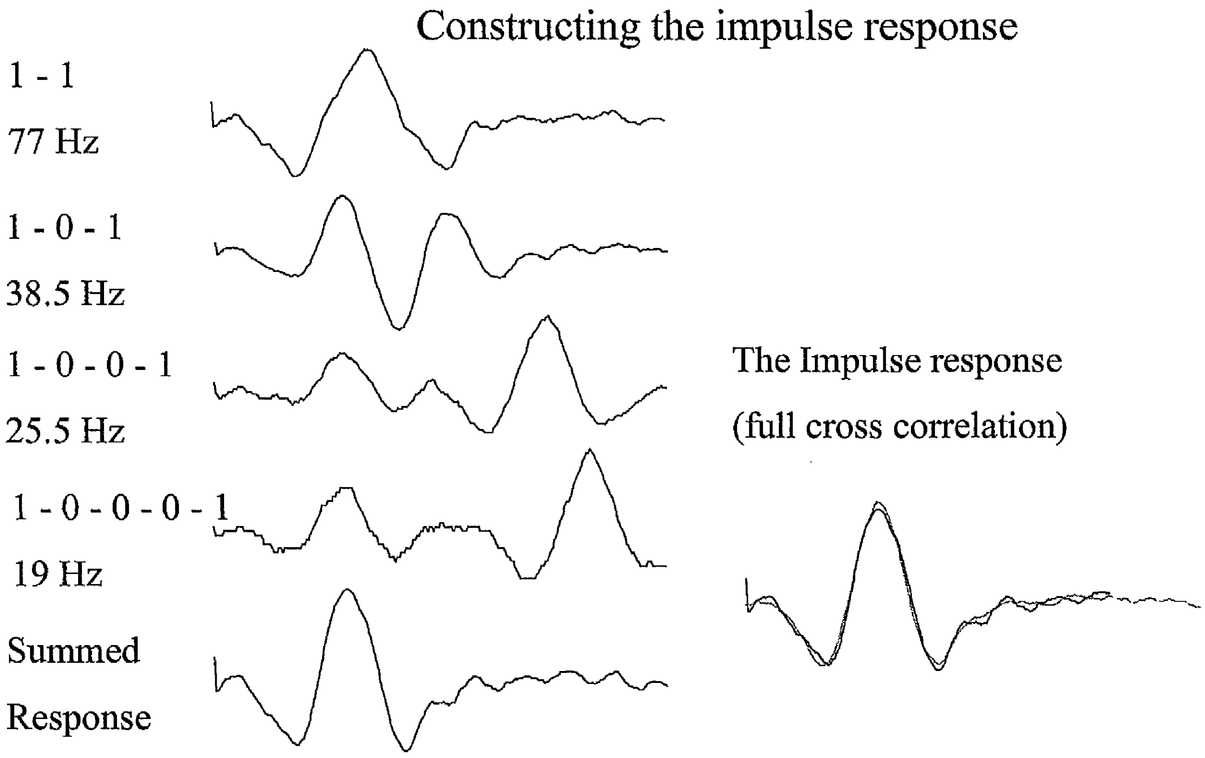
0 - 0 - 1 - 0 - 0

Shift a second response by 26 msec

Add two responses

for a 1 - 0 - 1 part of the sequence

Figure 5.3: *Extraction of Waveforms (2)*



Constructing the impulse response

1 - 1

77 Hz

1 - 0 - 1

38.5 Hz

1 - 0 - 0 - 1

25.5 Hz

1 - 0 - 0 - 0 - 1

19 Hz

Summed

Response

The Impulse response

(full cross correlation)

Figure 5.4: *Constructing the impulse response*

Chapter 6

INVESTIGATION OF PRIMITIVE POLYNOMIALS

The purpose of this Chapter is to investigate the effects of cross-contamination from higher orders of the response.

6.1 INTRODUCTION

Multifocal stimulation of the retina gives a response, which is non-linear. Recall that the correlation between the binary m-sequence and the response from the retina is referred to as the first order response. The correlation between the binary m-sequence and the response from the retina is referred to as the first order response. In practice, the cross-correlation is applied between the m-sequence and the full data recording. The ERG response is contained only in the first 200msecs and this is therefore the main time window of interest. There should be no signal present in longer time periods for the standard Multifocal ERG and this is simply discarded. The first-order or impulse response is a composite response containing both linear and non-linear components - it selectively recovers how the eye responds to a flash of light and the second-order response selectively recovers the interaction between flashes.

Recall, a schematic diagram of the first order response is illustrated in figure 2.11 and is simply the difference between transitions to a white stimulus and transitions to a black stimulus. Consider the m-sequence generated from the primitive polynomial $x^4 + x + 1$,

1-0-0-0-1-0-0-1-1-0-1-0-1-1-1.

The stimulus is updated according to this sequence at a rate of 75 Hz and the ERG time window is typically 200 msec. Each m-sequence step therefore occurs every 13.33 msec. If the m-sequence is '1' then a data segment of 200msec from this instant in time is added and if the m-sequence is '0' then the data segment of 200 msec is subtracted.

Also recall that the second order response is the difference between a change of state and no change of state as illustrated in figure 2.12. We would therefore add data segments when the m-sequence changes from 1-0 or from 0-1 and subtract the data segment when the m-sequence does not change 0-0 or 1-1. If we make use of the shift and add property of m-sequences then we can generate the second order sequence direct and simply cross correlate the new second order sequence with the raw data.

	1-0-0-0-1-0-0-1-1-0-1-0-1-1-1
shift right	0-0-0-1-0-0-1-1-0-1-0-1-1-1-1
add mod 2	1-0-0-1-1-0-1-0-1-1-1-1-0-0-0

Inspection of the above shows that the new sequence contains a 1 at the point where there is a change in state in row 1 and a zero when there is no change of state. We can therefore use this second order sequence to directly recover the difference between no change of state and a change of state. This provides information on retinal adaptation mechanisms.

The engineering requirements are that there should be no overlap of first or second order sequences within the time window of interest, typically 200 msecs corresponding to 16 clear m-sequence steps (since 200/13.33 = 15.00375 >15). The effects of window length 32 were also examined. Cross-contamination is a first or second order sequence appearing in an area it was not intended. It is a problem because the sum of two shifted m-sequences is itself a shifted m-sequence (using the "Shift & Add Property", Property 4.5), and it follows that orthogonality may be compromised as higher order contributions from one area may correlate with contributions from separate areas of the retina, giving inaccurate waveforms. M-sequence analysis is beneficial, because if there is an interaction between two consecutive stimuli, this can be found by examining the initial sequence - it may eliminate cross-contamination. In Multifocal Electroretinography (MFERG), firstly the stimulus sequence at each hexagonal area must be uncorrelated with *EVERY* other stimulus sequence, so that when the MFERG signal is cross-correlated with the individual sequence, only the hexagons contribution remains, all the other contributions cancelling out. Not only must this first condition of orthogonality be met, but second-order contamination must also be eliminated. The purpose of this chapter is to outline how suitable binary sequences can be selected which satisfy both these conditions.

6.2 METHODS

In this section, the mathematical techniques which help identify the exact position of cross-contamination between first and second order sequences will be presented.

A more theoretical analysis is required to select the taps for the feedback shift register, so that the resulting PRBS is guaranteed to be of maximal length. The taps required for the shift register correspond to the terms in a primitive polynomial. All primitive polynomials of degree m, have to be constructed, to allow every possible m-sequence for a selection of degrees 3-16 to be investigated. To generate all the primitive polynomials of degree m, finite fields were constructed.

One of the simplest examples of a finite field is GF(2^4) which was constructed in Chapter 3. In Table 6.1 we show, for each element of GF(2^4), its representation as a power of the primitive element α, its order and its minimal polynomial.

Table 6.1: GF*(2^4)

i	α^i	ord (α^i)	Minimal Polynomial
0	0001	1	$x+1$
1	0010	15	$x^4 + x + 1$ [i.e. $(x-\alpha)(x-\alpha^2)(x-\alpha^4)(x-\alpha^8)$]
2	0100	15	$x^4 + x + 1$
3	1000	5	$x^4 + x^3 + x^2 + x + 1$
4	0011	15	$x^4 + x + 1$
5	0110	3	$x^2 + x + 1$
6	1100	5	$x^4 + x^3 + x^2 + x + 1$
7	1011	15	$x^4 + x^3 + 1$
8	0101	15	$x^4 + x + 1$
9	1010	5	$x^4 + x^3 + x^2 + x + 1$
10	0111	3	$x^2 + x + 1$
11	1110	15	$x^4 + x^3 +1$
12	1111	5	$x^4 + x^3 + x^2 + x + 1$
13	1101	15	$x^4 + x^3 +1$
14	1001	15	$x^4 + x^3 +1$
15	0001		

To calculate the order of α^i in column 3, substitute ord (α) = $2^m - 1$ into Lemma 3.3, to obtain

$$\mathrm{ord}(\alpha^i) = (2^m - 1) / \gcd(i, 2^m -1), \quad ....(*)$$

(Recall, gcd stands for greatest common divisor).

In the above case, substituting m = 4 into the equation (*), and reading the i-value from column 1, results in α^1, α^2, α^4, α^8 having order 15, and α^5, α^{10} having order 3, etc.

6.2.1 More About Minimal Polynomials

The minimal polynomial of 1 in any field modulo 2 is x + 1. When i = 1, the minimal polynomial of α will always be the polynomial used to define the field (in this instance $x^4 + x + 1$). If α is any root of an irreducible polynomial, all other roots

are given by $\alpha^2, \alpha^4, \ldots\ldots, \alpha^{2^\wedge(m-1)}$, called the "conjugates" of α. For example, multiplying $[(x-\alpha)(x-\alpha^2)(x-\alpha^4)(x-\alpha^8)]$ modulo 2, gives
$x^4 + (\alpha + \alpha^2 + \alpha^4 + \alpha^8)\, x^3 + (\alpha^3 + \alpha^5 + \alpha^6 + \alpha^9)\, x^2 + (\alpha^7 + \alpha^{11} + \alpha^{13} + \alpha^{14})\, x + \alpha^{14}$.
Reading the α values from Table 6.1 results in the minimal polynomial $x^4 + x + 1$. We find, α, α^2, α^4, α^8 has the same minimal polynomial, namely $x^4 + x + 1$.

Similarly, $\alpha^{-1} \equiv \alpha^{14}$, $\alpha^{-2} \equiv \alpha^{13}$, $\alpha^{-4} \equiv \alpha^{11}$, $\alpha^{-8} \equiv \alpha^7$, are all conjugates. They will have the same reciprocal minimal polynomial, namely $x^4 + x^3 + 1$. Consider i = 3, since order of $\beta = \alpha^3$ is 5 because $\beta^5 = 1$, $\beta \neq 1$ implying
$0 = \beta^5 - 1 = (\beta - 1)(\beta^4 + \beta^3 + \beta^2 + \beta + 1)$. Hence, the minimal polynomial for β is $x^4 + x^3 + x^2 + x + 1$, and similarly for the conjugates α^6, α^{12}, $\alpha^{24} \equiv \alpha^9$. Although this polynomial is irreducible, it does not have any roots which are primitive elements, and would produce a sequence of length 5, not an m-sequence. Similarly for $\beta = \alpha^5$ of order 3, $\beta^3 = 1$, $\beta \neq 1$ implying $0 = \beta^3 - 1 = (\beta - 1)(\beta^2 + \beta + 1)$, and the minimal polynomial is $x^2 + x + 1$.

Definition A primitive polynomial is a minimal polynomial, which contains a primitive element as a root.

All the primitive polynomials of degree m can be discovered from columns 2 and 3 in Table 6.1. In this case, there are only two primitive polynomials, namely $x^4 + x + 1$ and $x^4 + x^3 + 1$

6.2.2 Primitive Polynomials

There are $\phi(2^m - 1)/m$ primitive polynomials of degree m, as illustrated in Table 6.2 (McEliece,1987). Each primitive polynomial produces a distinct m-sequence of length $2^m - 1$. This formula is related to the number of Cyclotomic

Cosets in Section (see for example, Golomb (1967), "Shift Register Sequences" Chapter 3, pg 50 on Cyclotomic Polynomials).

Table 6.2: Number of Primitive Polynomials of each degree

m	$\phi\,(2^m - 1) / m$
1	1
2	1
3	2
4	2
5	6
6	6
7	18
8	16
9	48
10	60
....	....
....	....

6.2.3 Zech Logarithms

Once an m-sequence has been generated from the primitive polynomial, it has to be decimated (Table 5.1).

The "Shift & Add" Property stated that the sum of any two shifts of an m-sequence is a third shift of that same sequence. Recall that the Second-Order Response is obtained from this "new" shifted m-sequence cross-correlated with the unprocessed response from the eye.

Table 6.2, illustrated that several polynomials existed for each degree. A simple example will be discussed in detail, then an example will be given of a sequence which is of practical use.

For example, return to the polynomial $x^4 + x + 1$ which was decimated over four columns. Consider Area 0, adding the sequence 1 1 1 1 0 0 0 1 0 0 1 1 0 1 0 to its first shift 1 1 1 0 0 0 1 0 0 1 1 0 1 0 1 modulo 2 gives 0 0 0 1 0 0 1 1 0 1 0 1 1 1 1. Note that this "new" sequence occurs at position 4 in the original sequence $A_{0.}$ For larger m-sequences it is more difficult to calculate where this "new" sequence occurs, and this is where Zech Logarithms are useful.

Zech Logarithms are used to add elements in a finite field (they are also referred to as Jacobi's Logarithms). If we return to Table 6.1, and look at the field elements - the 4-tuples (notice that each 4-tuple has a corresponding α^i value
(i read from column 1)) - they can be used to help identify cross-contamination between the various first and second-order sequences. The following definition is taken from Huber (1990).

Definition: Let α be a primitive element of GF(q) which is a root of the primitive polynomial p(x) of degree m. For GF(q) let $N_q = \{0,1, \ldots, q\text{-}2\} \cup \{-\infty\}$. Then the **Zech Logarithm** Z(x) is defined by:

$$\alpha^{Z(x)} = 1 + \alpha^x$$

Z is a mapping Z: $N_q \rightarrow N_q$. Any element β from the finite field is assumed to be given in its polar representation as a power of the primitive element α: $\beta = \alpha^1$ (define $\alpha^{-(\text{infinity})} = 0$). Multiplication can then be undertaken very easily, by adding the exponents modulo q-1.

Definition: (***Addition***) Let z be the exponent of the sum $\alpha^x + \alpha^y$ (for example $\alpha^x + \alpha^y = \alpha^z$) then,

$$z = x + Z\,(y\text{-}x) = y + Z\,(x\text{-}y)$$

For more information on Zech Logarithms, see for example Imamura (1980). Some properties of Zech Logarithms, which follow very easily are listed overleaf.

Property (Z1): $Z(q-1-x) = Z(x) - x \pmod{q-1}$, $x \neq$ - infinity,

Property (Z2): $Z(px) = p \, . \, Z(x) \pmod{q-1}$,

Property (Z3): $Z(0) =$ - infinity, for $p = 2$,

Property (Z4): $Z((q-1)/2) =$ - infinity, for $p \neq 2$,

and for all primes:

Property (Z5): Z(- infinity) = 0.

In this chapter, $x = 1$ and $y = 0$. eg $\alpha^{Z(1)} = 1 + \alpha^1$ will be considered, corresponding to adding the m-sequence to its first shift. Therefore, z(1) tells us at which position in the original m-sequence the "new" second-order sequence occurs.

An m-sequence contains the "Window Property", in this case, when m = 4, each 4-tuple (except the all-zero 4-tuple) occurs once and only once in one period, therefore we need only add together the first 4 elements of each sequence. This is why we use the finite field elements. It does not matter which two 4-tuples we add together from our finite field, as long as they are of the form α^{i} and α^{i+1} for any $0 \leq i \leq 13$. It may help to think of α^{i} as a sequence. We always discover when the 4-tuples are added together, the new value occurs $\alpha^{i+z(1)}$ positions away. Thus, we look for the easiest example eg.$1 + \alpha^1$ to find the value of z(1).

Consider the 4-tuples in Table 6.1,

$$\begin{array}{lrl} & 1 \equiv \alpha^0 = & 0001 \\ & \underline{\alpha^1 = } & \underline{0010} \\ \text{Addition of the 4-tuples gives} & & 0011 \quad \pmod 2 \end{array}$$

This is α^4 in Table 6.1. Therefore, $1 + \alpha^1 = \alpha^4$ (i.e. z (1) = 4). Telling us, as expected, the second-order sequence for Area 0 occurs at position 4 in the initial sequence for that area. We have to check that this new sequence does not contaminate a sequence from any of the remaining areas.

Take for example Area 1 in Table 5.1. The initial sequence from Area 0 starts at position 11 in Area 1. Since the sequence is cyclic (period 15), when the z(1) value of 4 is added, modulo 15, it takes us to position 0. In other words, the m-sequence for area 1 is identical to the second-order response from Area 0, and the cross-correlation of Area 1 will result in a response that is contaminated with the second order response from Area 0. This should have little influence on the first order response, because higher order responses are, in general, small in comparison to first order responses. The converse however, is false, and first order responses have a huge effect on the second-order responses. The second order response from Area 1 would predominately contain contamination from the first order response of Area 0.

Using the methods described above, we can look at all the primitive polynomials of each degree, and find out which polynomials give rise to appropriate m-sequences to avoid cross-contamination in multifocal electroretinogram responses. Some look up tables of Zech Logarithms exist, but a computer program was written to allow tables of z(1) values to be constructed for all primitive polynomials of degree 3 to 16. Appendix C gives a complete list of primitive polynomials of degree 3-11, and their corresponding z(1) value.

In the Multifocal ERG, an m-sequence of length 2^{15} - 1 is often used. This sequence is decimated over 128 columns to allow a stimulus of 61 or 103 areas to be chosen. An example of an appropriate degree 15 primitive polynomial which is suitable for m-sequence generation for Multifocal Electroretinography can be found in Appendix D,

Example 6.1: Let $h(x) = 1+ x + x^2 + x^3 + x^4 + x^5 + x^7 + x^{10} + x^{12} + x^{14} + x^{15}$.

In this instance z(1) = 8751.

If this z(1) value, when taken modulo 256 ($2^{15} / 2^{7}$),occurs in position 0-15, or 239-255, then the second-order array would be contaminated, therefore, the z(1) value must lie between 16 and 238 if we require a window of length 16.

We find 8751 ≡ 47 mod 256, which lies between 16 and 238, therefore, this polynomial is suitable for generation of an m-sequence Multifocal Electroretinography.

If a photodiode (artificial eye), fixed on the centre of the stimulus is used in the experiment, and a "bad" m-sequence is chosen to control the stimulus, we can predict exactly which areas in the second-order array cross-contamination will occur. The photodiode is a linear device, and does not have a second-order response. Therefore, any spikes that appear on the second-order trace array are a result of contamination between the sequences.

6.3 RESULTS

For degrees less than twelve, there are no appropriate primitive polynomials, each one gives rise to an m-sequence which has a problem with cross-contamination. This is the case, regardless as to whether we decimate the sequence by 64 or 128, and look at a window of length 16 or 32.

Consider the 144 primitive polynomials of degree 12 in Appendix C, and decimate the m-sequence by 64. If the z(1) value, when taken modulo 64 ($2^{12} / 2^{6}$),occurs in position 0-15, or 47-63, then the second-order array would be contaminated, therefore, the z(1) value must lie between 16 and 46 if we require a window of length 16. There are 52 appropriate polynomials in this instance. For a window of length 32, if the z(1) value (when taken modulo 64) falls between 0-31 or 32-63, then contamination will occur in the second-order trace array. This includes everything! Therefore, in this case, we find there are no appropriate polynomials. If we decimate by 128 or 256, there are no appropriate polynomials, regardless as to whether we require a window of length 16 or 32.

Example 6.2: Let $h(x) = 1 + x + x^4 + x^6 + x^{12}$. Then $z(1) = 1937$ (from Appendix D).

Decimate by 64 $\Rightarrow (2^{12} / 2^{6}) = 64$ (= time lag between cols)

$$1937 \equiv 17 \bmod 64.$$

Note: this is suitable if we require a window of length 16, since 16 <17< 46, but not if we require a window of length 32, since 17 < 31.

Decimate by 128 $\Rightarrow (2^{12} / 2^{7}) = 32$ (= time lag between cols)

$$1937 \equiv 17 \bmod 32.$$

Note: this would not be suitable.

Decimate by 256 $\Rightarrow (2^{12} / 2^{8}) = 16$ (= time lag between cols)

$$1937 \equiv 1 \bmod 16.$$

Note: this would not be suitable.

Now consider the 756 primitive polynomials of degree 14, and decimate the sequence by 64. The z(1) value is calculated modulo 256, and if we require a window of length 16, the value must lie between 16 and 239 to avoid contamination in the second-order response. There are 91 “bad” polynomials in this instance. For a window of length 32, there are 193 “bad” polynomials. If the sequence is decimated by 128, and we require that the z(1) value, when taken modulo 128, falls within 16 and 111, for a window of length 16. There are 197 "bad" polynomials in this instance, and 416 “bad” polynomials if a window of length 32 is required. Decimating by 256 results in

398 “bad” polynomials for a window of length 16, and 756 “bad” polynomials if a window of length 32 is required. A summary of these results is illustrated in Tables 6.3 & 6.4. The terms "good" and "bad" are only relevant to how the primitive polynomial is used in this application. It only becomes "bad" if when the sequence is decimated we get cross-contamination occurring within the time window of interest.

Table6.3: Percentage of Inappropriate Primitive Polynomials when Window Length is 16

	8	10	12	14
64	100%	100%	63.9%	12.0%
128	100%	100%	100%	26.0%
256	100%	100%	100%	52.6%

Table 6.4: Percentage of Inappropriate Primitive Polynomials when Window Length is 32

	8	10	12	14
64	100%	100%	100%	25.5%
128	100%	100%	100%	55.0%
256	100%	100%	100%	100%

Figures 6.1 & 6.2 show the graphical representation of the percentage of "bad" polynomials for sequences decimated by 128, with windows of length 16 and 32 respectively.

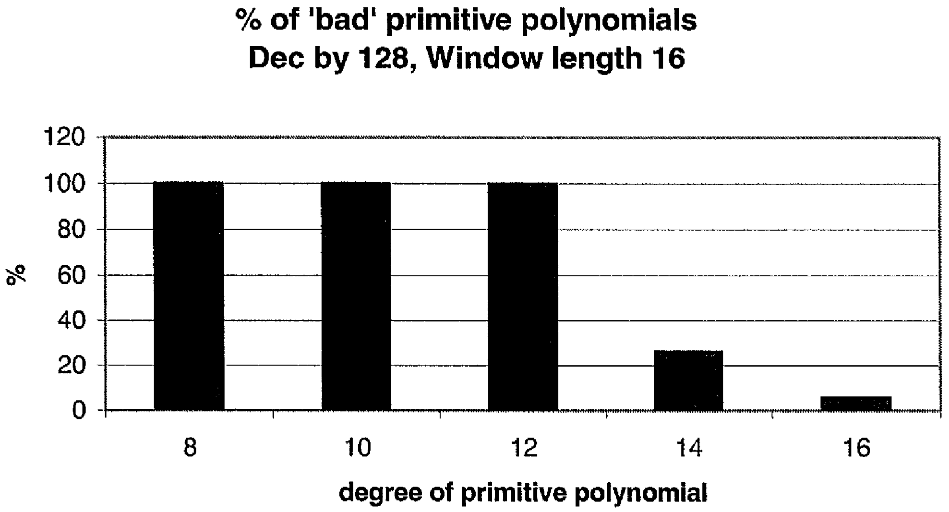


Figure 6.1: *Graph illustrating the percentage of "bad" polynomials (Dec128, Win 16)*

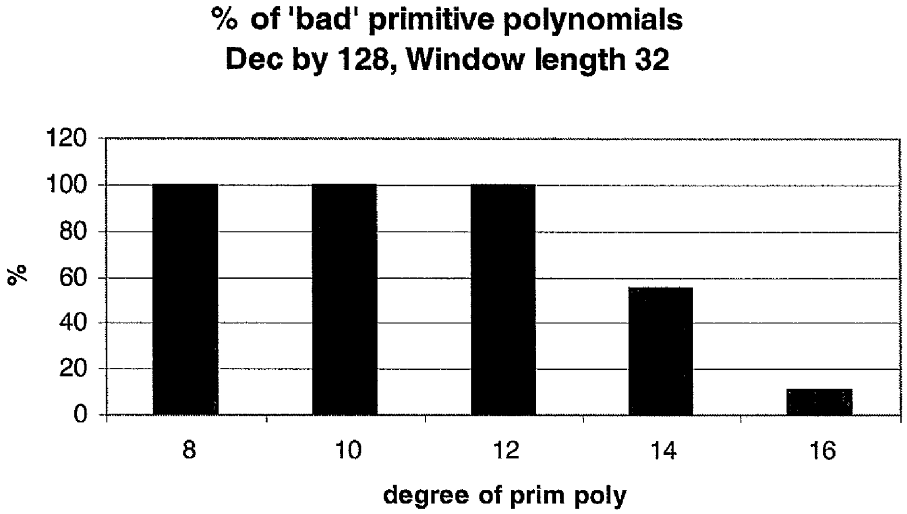


Figure 6.2: *Graph illustrating the percentage of "bad" polynomials (Dec 128, Win 36)*

An experiment was undertaken to illustrate the practical effects of cross-contamination. A custom built multifocal Light Emitting Diode stimulator interfaced to a standard PC was used for these experiments. The stimulus consisted of 61 hexagonal elements scaled for photoreceptor density as illustrated in Figure 6.3.

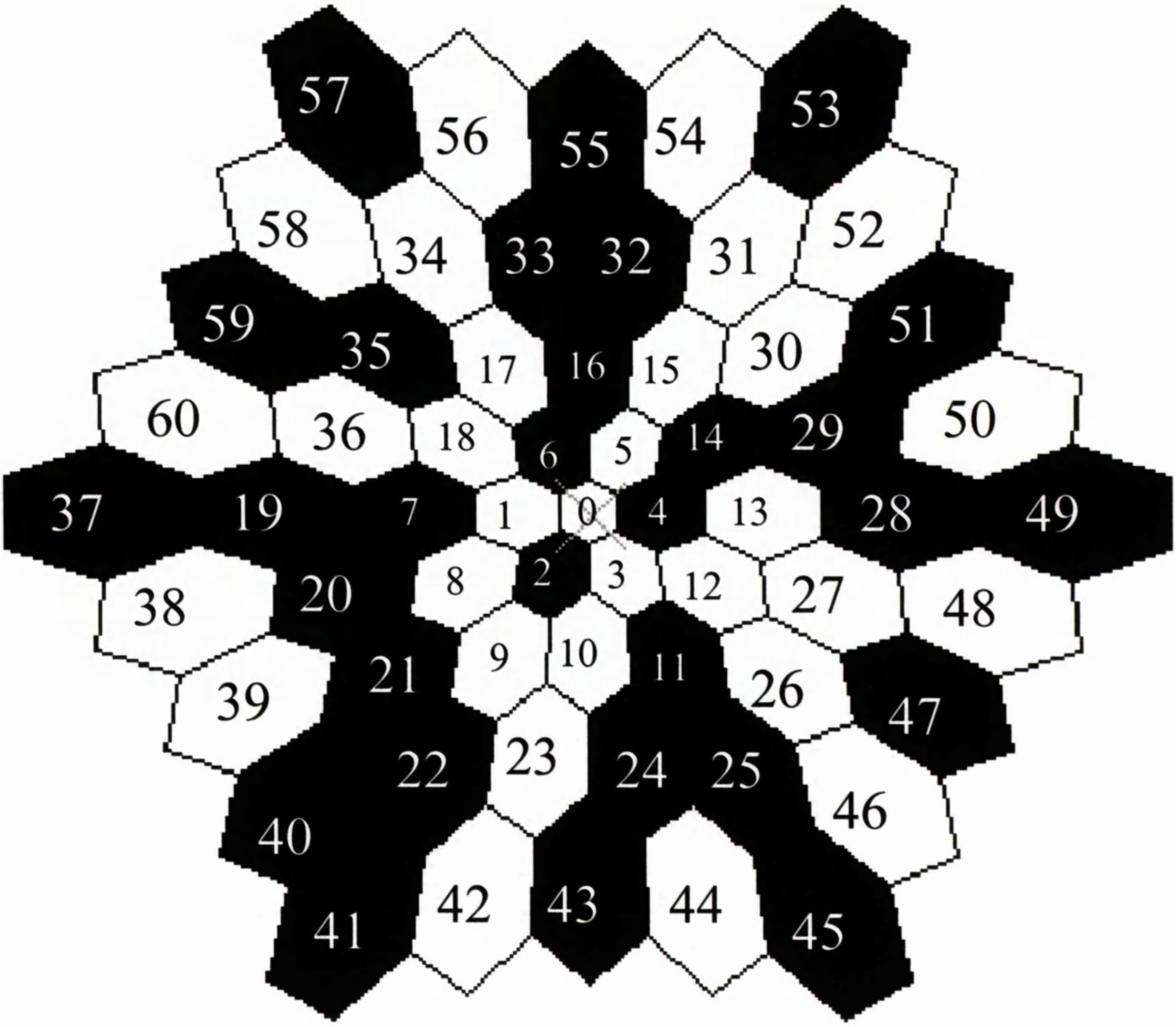


Figure 6.3: *Hexagonal Stimulus - each area is numbered to correspond to the column number of the sequence used to stimulate that particular area.*

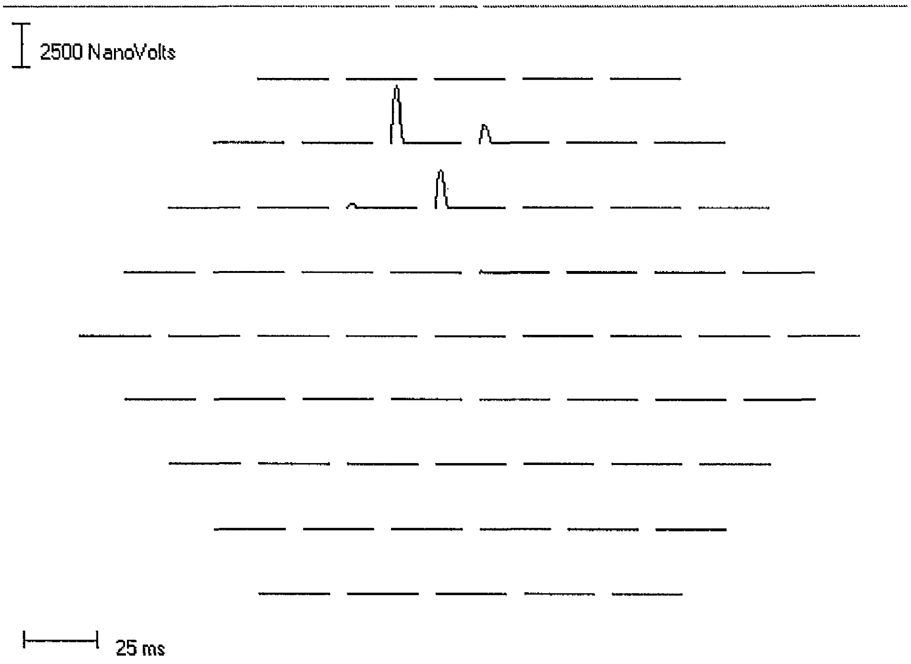
The stimulus can be controlled by any sequence but in this case the elements are controlled by a decimated m-sequence. Each of the hexagonal areas in the stimulus has a number, which corresponds to the columns containing the decimated m-sequences. An m-sequence of degree 15 obtained from the primitive polynomial $x^{15} + x^{13} + x^{12} + x^{11} + x^{6} + x^{3} + x^{2} + x + 1$ was chosen to illustrate the effects of cross-contamination. Using the techniques introduced in the methods section, it was discovered that this polynomial would generate an inappropriate sequence to view the second-order response from the eye. The second order sequence would occur at position 5120 in the initial sequence (Area 0).

A photodiode (artificial eye) was place in front of the stimulus covering areas 33,32,16 and 17. After the sequence is run and the raw data is cross correlated with the m-sequence the result of Figure 6.4 is obtained.

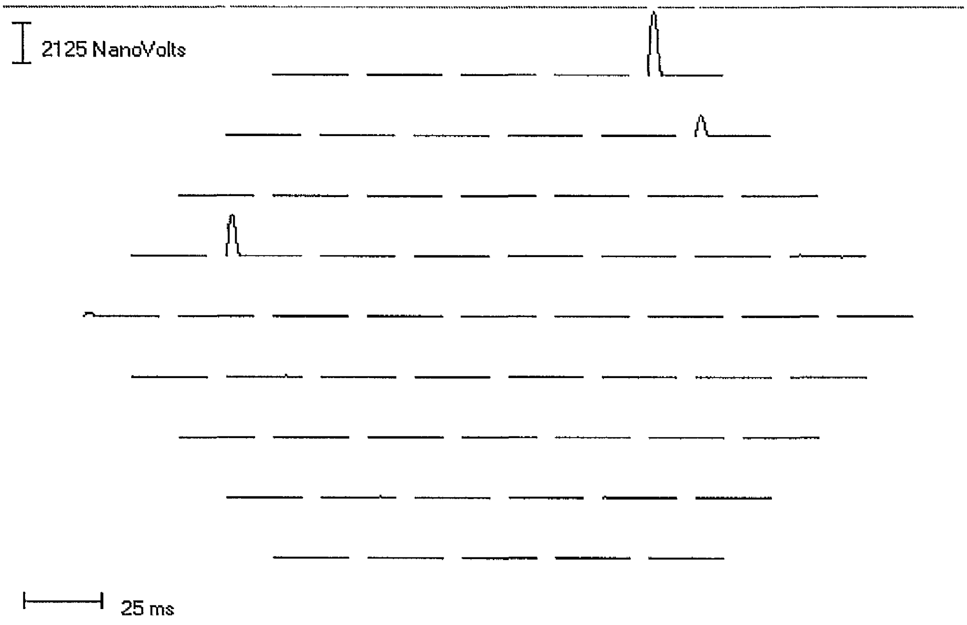
The second order response should not contain any waveforms as the photodiode is a linear device. Note however, that if the sequence had been decimated over 128 columns, there would be a time lag between columns of 256. If we take the z(1) value of 5120 modulo 256, we find that this contamination occurs at position 0 in one of the other Areas. Note that if we divide this z(1) value by 256 we find the answer is 20. We have shown that we would expect cross contamination to occur at position 0 in Area 20 if we use the polynomial $x^{15} + x^{13} + x^{12} + x^{11} + x^{6} + x^{3} + x^{2} + x + 1$. Cross correlation of the raw data with the second order sequences show the contaminated responses appearing at 53, 52, 37 and 36 respectively - a difference of 20 from the initial areas. These waveforms would not be shown if a "good" primitive polynomial was chosen.

Figure 6.4: Results from experiment using a photodiode placed over areas 33, 32, 16 and 17.

(a) First Order Trace Array



(b) Second Order Trace Array

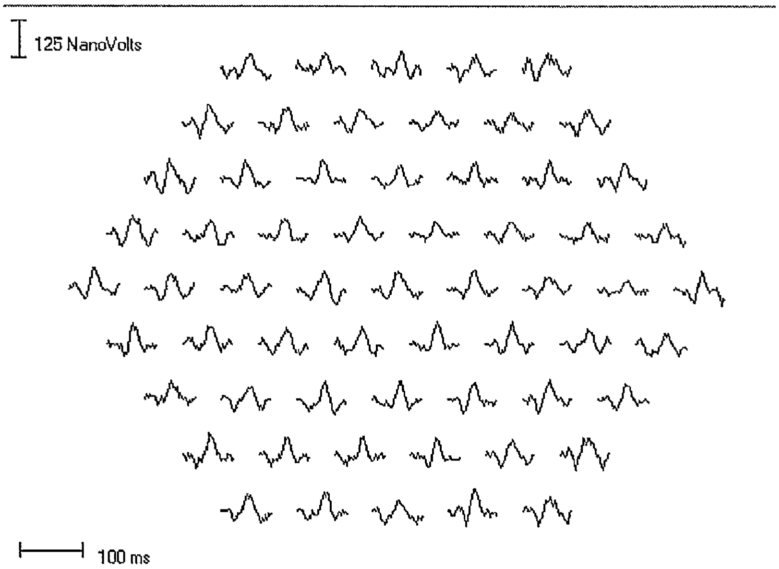


We then performed the same experiment to obtain the physiological response. In this case, a control subject placed an electrode in the eye and the raw data was recovered from the eye. The results can be seen in Figure 6.5. The first trace array (Figure 6.5(a)) illustrates the First Order response from the eye, and the second (Figure 6.5(b)) illustrates the Second Order response.

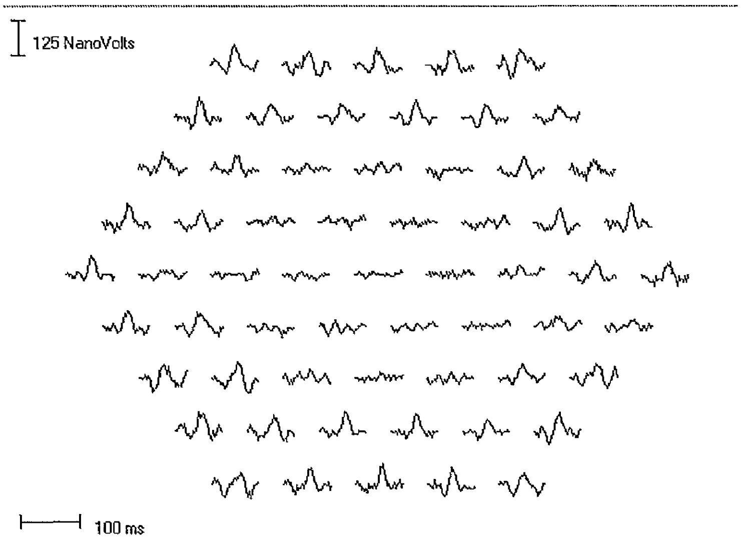
The cross-contamination is obvious. If you compare this trace array with Figure 6.3, you see that contamination occurs, as expected, from area 20 onwards.

Figure 6.5: *Physiological Results*

(a) First Order Response



(b) *Second Order Response*



Appendix 4 lists some polynomials of degree 14-16, which, when decimated over 128 columns, are appropriate for generation of m-sequences for the Multifocal ERG.

Chapter 7

CONCLUSIONS & FURTHER WORK

7.1 CONCLUSION

An appropriate primitive polynomial must be chosen at the onset to avoid contamination between the sequences. The problem will be more severe for VECP recordings than ERG recordings, as the window lengths are longer, and the second-order response is more important. Also, faster stimulation that 75Hz will be more of a problem, for example in multifocal tests using the LED Stimulator.

To drive the multifocal stimulus a polynomial of degree greater than or equal to 12 is required, although if degree 12 is chosen, you are restricted to a stimulus with 63 areas, and a time window of length 16. The higher the degree of the polynomial, and the smaller the length of the window chosen, the less chance there is of cross-contamination between the sequences.

As can be seen from the experiment using the photodiode, choosing an appropriate m-sequence at the onset is crucial to avoid misdiagnosis in the test results.

7.2 FURTHER WORK

The possibility exists for this work to be extended to include Ternary Sequences instead of Binary Sequences. Firstly, how does the eye respond when faced with random flashes of Black, White and Grey, corresponding to a 0, 1, and 2 in the Ternary Sequence. How does this affect contamination between the sequences when they are decimated over 128 columns?

What happens if sequences are examined, which have say, one particular property missing? For example, Legendre Sequences have good autocorrelation properties, but no "Shift & Add" Property, can they be used to drive the stimulus for Multifocal Electrophysiology Tests? What about Random Number Generators - what particular properties do they possess? Do any other sequences contain similar properties to m-sequences? Or are m-sequences the best modulators for this technique?

REFERENCES:

ALFREDSSON, L. (1996): '*VLSI Architectures and Arithmetic Operations with Applications to the Fermat Number Transform*': PhD dissertation Number 425, Department of Electrical Engineering, Linkoping University, Sweden.

ANDERSON, I. (2001): '*A First Course in Discrete Mathematics*': Springer Undergraduate Mathematics Series.

BIGGS, N.L.(1989):'*Discrete Mathematics (Revised Edition)*': Oxford University Press.

BIRDSALL T.G. and RISTENBATT M.P. (1958): '*Introduction to linear shift register generated sequences*': USA Technical Report No **90**. Univ. Mich. Research Inst., Ann Arbor.

BOYER, C.B. (1991): '*A History of Mathematics, (Second Edition)*', Wiley.

CARR, R.E. and SIEGEL, I.M.(1990): '*Electrodiagnostic Testing of the Visual System : A Clinical Guide*', F. A. Davies Company.

CHOW P.E.K. and DAVIES A.C. (1964): '*The synthesis of cyclic code generators*': Electron. Eng. **36**, pp 253.

COLBOURN, C.J. and DINITZ, J.H. (1996): '*The CRC Handbook of Combinatorical Designs* ': CRC Press.

DAVIES W.D.T. (1970): '*System identification for self-adaptive control*': Wiley.

FRALEIGH, J.B., (1967):'*A First Course in Abstract Algebra*': Addison -Wesley.

FRICKER, S.J., and SANDERS, J.J. (1974): '*A new method of cone electroretinography: the rapid random flash response*', Investig Ophthalmol., **14,** pp 131 - 137.

GOLOMB, S. (1967): '*Shift Register Sequences*', Holden-Day Series in Information Systems.

HADAMARD, J. (1893): '*Resolution d'une question relative aux determinants* ', Bull. Sciences Math. 17, pp 240-246.

HILL, R (1997): '*A First Course in Coding Theory*': Oxford University Press.

HOFFMANN DE VISME, G. (1971): '*Binary Sequences*', English Universities Press.

HOROWITZ, P. and HILL, W. (1989): '*The Art of Electronics (Second Edition)*', Cambridge University Press.

HUBEL, D.H. (1995): *'Eye, Brain and Vision'* Scientific American Library, No 22, W.H. Freeman, NY.

HUBER, K. (1990): *'Some comments on Zech's Logarithms'* IEEE Trans. Inform. Theory **36** no 4, pp 946-950.

IMAMURA, K. (1980): *'A method for computing addition tables in GF(p^m)'* IEEE Trans. Inform. Theory, vol. IT-26, no 3, pp 367-369.

JUNGNICKEL, D. (1993): *'Finite fields - Structure and Arithmetics'*, B.I. Wissenschaftsverlag, Mannheim.

KEATING, D.; PARKS, S. and EVANS A. (2000): *'Technical Aspects of Multifocal ERG Recording'* Documenta Ophthalmologica **100**, pp 77-98.

LIDL, R. and NIEDERREITER, H. (1994): *'Introduction to Finite Fields and their Applications (Second Edition)'*: Cambridge University Press.

LIDL, R. and NIEDERREITER, H. (1997): *'Finite Fields'*: Encyclopedia of Mathematics and Its Applications, Vol 20 (2nd Edition), Cambridge University Press.

MARKS, R.W. (1967): *'The New Mathematics Dictionary and Handbook'*: Bantam Books.

McELIECE, R.J.(1987): *'Finite Fields for Computer Scientists and Engineers'*, Klumer Academic Publishers.

MacWILLIAMS, F.J. and SLOANE, N.J.A. (1976):*'The Theory of Error Correcting Codes'*, North-Holland.

MacWILLIAMS, F.J. and SLOANE, N.J.A. (1976): *'Pseudo-Random Sequences and Arrays'*, Proceedings of the IEEE **64**, pp 1715 - 1729.

SCHOLEFIELD R.E. (1960): *'Shift Registers Generating Maximum Length Sequences'*: Electronic Technology **37**, pp 389.

SCHROEDER, M.R.(1986): *'Number Theory in Science and Communication'* Second Enlarged Edition. Springer - Verlag.

SUTTER, E.E., and TRAN, D (1992): *'The field topography of ERG Components in man -1. The Photopic Luminance response'*, Vis Res., **32,** pp 433- 446.

VAN LINT, J.H. and WILSON, R.M. (1993): *'A Course in Combinatorics'*, New York: Cambridge University Press.

APPENDIX A: DELPHI CODE TO GENERATE AN M-SEQUENCE

Illustrated below is the Delphi code, which can be used to generate any m-sequence of length 7 to (2^15-1), from a given primitive polynomial.

DELPHI CODE

```
unit Ch2;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls;

type
  TForm1 = class(TForm)
    Button1: TButton;
    CheckBox1: TCheckBox;
    CheckBox2: TCheckBox;
    CheckBox3: TCheckBox;
    CheckBox4: TCheckBox;
    CheckBox5: TCheckBox;
    CheckBox6: TCheckBox;
    CheckBox7: TCheckBox;
    CheckBox8: TCheckBox;
    CheckBox9: TCheckBox;
    CheckBox10: TCheckBox;
    CheckBox11: TCheckBox;
    CheckBox12: TCheckBox;
    CheckBox13: TCheckBox;
    CheckBox14: TCheckBox;
    CheckBox15: TCheckBox;
    Edit1: TEdit;
    Edit2: TEdit;
    Edit3: TEdit;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.DFM}

function pow(num1,num2:Integer):Integer;
var i:Integer;

  begin
  Result:=1;

   For i:=1 to num2 do

     Result:=Result*num1;

 end;

procedure TForm1.Button1Click(Sender: TObject);
var MSeq:Array[1..655361] of Byte;
   Tap:Array[1..16] of Integer;
   a,b,m,p,h,i,code,k,j,l,t:Integer;
   s,u,v:String;

 begin
   Val(Edit1.text,i,code);
   Val(Edit2.text,j,code);
   l:=pow(i,j);
   p:=l-1;

     begin

      MSeq[1]:=1;
      a:=MSeq[1];
      Str(a,u);
      Edit3.text:=u;

        For k:=2 to j do

          begin

           MSeq[k]:=0;
           b:=MSeq[k];
           Str(b,v);
           Edit3.text:=Edit3.text+v;
           Form1.Refresh;

          end;

            If Checkbox1.Checked then Tap[1]:=1 else Tap[1]:=0;
            If Checkbox2.Checked then Tap[2]:=2 else Tap[2]:=0;
            If Checkbox3.Checked then Tap[3]:=3 else Tap[3]:=0;
            If Checkbox4.Checked then Tap[4]:=4 else Tap[4]:=0;
            If Checkbox5.Checked then Tap[5]:=5 else Tap[5]:=0;
            If Checkbox6.Checked then Tap[6]:=6 else Tap[6]:=0;
            If Checkbox7.Checked then Tap[7]:=7 else Tap[7]:=0;
            If Checkbox8.Checked then Tap[8]:=8 else Tap[8]:=0;
            If Checkbox9.Checked then Tap[9]:=9 else Tap[9]:=0;
            If Checkbox10.Checked then Tap[10]:=10 else Tap[10]:=0;
            If Checkbox11.Checked then Tap[11]:=11 else Tap[11]:=0;
            If Checkbox12.Checked then Tap[12]:=12 else Tap[12]:=0;
            If Checkbox13.Checked then Tap[13]:=13 else Tap[13]:=0;
            If Checkbox14.Checked then Tap[14]:=14 else Tap[14]:=0;
            If Checkbox15.Checked then Tap[15]:=15 else Tap[15]:=0;


            For k:=j+1 to p do

             begin
            For h:=1 to 16 do

              begin

                If Tap[h]>0 then

                  begin

                    MSeq[k]:=MSeq[k]+MSeq[k-Tap[h]];

                  end;

              end;

                  MSeq[k]:=MSeq[k] mod 2;

                  t:=MSeq[k];
                  Str(t,s);
                  Edit3.text:=Edit3.text+s;
           end;

     end;
   end;
  end.
```

APPENDIX B: DELPHI CODE TO GENERATE RUN DISTRIBUTION TABLES

Illustrated below is the Delphi code, which can be used to generate any m-sequence of length 7 to (2^15-1), from a given primitive polynomial, and produce it's run distribution tables.

DELPHI CODE

```
unit Int;

interface

uses
 Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
 StdCtrls;

type
 TForm1 = class(TForm)
  Button1: TButton;
  CheckBox1: TCheckBox;
  CheckBox2: TCheckBox;
  CheckBox3: TCheckBox;
  CheckBox4: TCheckBox;
  CheckBox5: TCheckBox;
  CheckBox6: TCheckBox;
  CheckBox7: TCheckBox;
  CheckBox8: TCheckBox;
  CheckBox9: TCheckBox;
  CheckBox10: TCheckBox;
  CheckBox11: TCheckBox;
  CheckBox12: TCheckBox;
  CheckBox13: TCheckBox;
  CheckBox14: TCheckBox;
  CheckBox15: TCheckBox;
  CheckBox16: TcheckBox;
  Edit1: TEdit;
  Edit2: TEdit;
  Edit3: TEdit;
  Edit4: TEdit;
  Label1: TLabel;
  Label2: TLabel;
  Button1: TButton;
  Memo1: TMemo;
  procedure Button1Click(Sender: TObject);
 private
  { Private declarations }
 public
   MSeq: Array[1..65536] of Byte;
  { Public declarations }
 end;

var
 Form1: TForm1;

implementation
```

```
{$R *.DFM}

function pow(num1,num2:Integer):Integer;
var i:Integer;

 begin
  Result:=1;

   For i:=1 to num2 do

     Result:=Result*num1;

 end;


procedure Consecutive_Ones;
var k,j,i,l,n,t,code,Ctr:Integer;
                 s:String;
        EndOfOnes:Boolean;
       Consec_Ones:Array[1..16] of Integer;



       begin
          Ctr:=0;

          Val(Form1.Edit1.text, i, code);
          Val(Form1.Edit2.text,j,code);
           l:=pow(i,j);
          n:=l-1;

          For k:=1 to j do

             begin
               Consec_Ones[k]:=0;
             end;

          For t:=1 to n do

             begin
               Str(t,s);
               Form1.Edit4.text:=s;
               Form1.Refresh;

                  If Form1.Mseq[t]=1 then
                      begin
                         EndOfOnes:=FALSE;
                        ctr:=ctr+1;
                      end;

                       else
                           begin
                             If EndOfOnes=False then
                                 begin
                                  EndOfOnes:=TRUE;
                                  Consec_Ones[ctr]:=Consec_Ones[ctr]+1;
                                  ctr:=0;
                                 end;
                           end;
```

```
            end;

           For k:=1 to j do
             begin
               Str(Consec_Ones[k],s);
               Form1.Edit1.text:=s;
               Form1.Refresh;
               Form1.Memo1.Lines[k]:=IntToStr(Consec_Ones[k]);
             end;

       end;


procedure TForm1.Button1Click(Sender: TObject);
var Tap:Array[1..16] of Integer;
    a,b,m,p,h,i,code,k,j,l,t:Integer;
    s,u,v:String;

 begin
  Val(Edit1.text,i,code);
  Val(Edit2.text,j,code);
  l:=pow(i,j);
  p:=l-1;

    begin

     MSeq[1]:=1;
     a:=MSeq[1];
     Str(a,u);
     Edit3.text:=u;

       For k:=2 to j do

         begin

          MSeq[k]:=0;
          b:=MSeq[k];
          Str(b,v);
          Edit3.text:=Edit3.text+v;
          Form1.Refresh;

         end;

           If Checkbox1.Checked then Tap[1]:=1 else Tap[1]:=0;
           If Checkbox2.Checked then Tap[2]:=2 else Tap[2]:=0;
           If Checkbox3.Checked then Tap[3]:=3 else Tap[3]:=0;
           If Checkbox4.Checked then Tap[4]:=4 else Tap[4]:=0;
           If Checkbox5.Checked then Tap[5]:=5 else Tap[5]:=0;
           If Checkbox6.Checked then Tap[6]:=6 else Tap[6]:=0;
           If Checkbox7.Checked then Tap[7]:=7 else Tap[7]:=0;
           If Checkbox8.Checked then Tap[8]:=8 else Tap[8]:=0;
           If Checkbox9.Checked then Tap[9]:=9 else Tap[9]:=0;
           If Checkbox10.Checked then Tap[10]:=10 else Tap[10]:=0;
           If Checkbox11.Checked then Tap[11]:=11 else Tap[11]:=0;
           If Checkbox12.Checked then Tap[12]:=12 else Tap[12]:=0;
           If Checkbox13.Checked then Tap[13]:=13 else Tap[13]:=0;
           If Checkbox14.Checked then Tap[14]:=14 else Tap[14]:=0;
           If Checkbox15.Checked then Tap[15]:=15 else Tap[15]:=0;
           If Checkbox16.Checked then Tap[16]:=16 else Tap[16]:=0;
```

```
            For k:=j+1 to p do
             begin
              For h:=1 to 16 do
                begin
                 If Tap[h]>0 then
                  begin
                   MSeq[k]:=MSeq[k]+MSeq[k-Tap[h]];
                  end;
                end;
                  MSeq[k]:=MSeq[k] mod 2;

                  t:=MSeq[k];
                  Str(t,s);
                  Edit3.text:=Edit3.text+s;
             end;
       end;
      Consecutive_Ones;
   end;
 end.
```

Appendix C : List of Primitive Polynomials and corresponding z(1) values

$n = 3$

{1,0,1,1}	3
{1,1,0,1}	5

$n = 4$

{1,0,0,1,1}	4
{1,1,0,0,1}	12

$n = 5$

{1,0,0,1,0,1}	18
{1,0,1,0,0,1}	14
{1,0,1,1,1,1}	12
{1,1,1,1,0,1}	20
{1,1,0,1,1,1}	19
{1,1,1,0,1,1}	13

$n = 6$

{1,0,0,0,0,1,1}	6
{1,1,0,0,0,0,1}	58
{1,0,1,1,0,1,1}	56
{1,1,0,1,1,0,1}	8
{1,1,0,0,1,1,1}	25
{1,1,1,0,0,1,1}	39

$n = 7$

{1,0,0,0,0,0,1,1}	7
{1,1,0,0,0,0,0,1}	121
{1,0,0,0,1,0,0,1}	31
{1,0,0,1,0,0,0,1}	97
{1,0,0,0,1,1,1,1}	87
{1,1,1,1,0,0,0,1}	41
{1,0,0,1,1,1,0,1}	118
{1,0,1,1,1,0,0,1}	10
{1,0,1,0,0,1,1,1}	114
{1,1,1,0,0,1,0,1}	14
{1,0,1,0,1,0,1,1}	21
{1,1,0,1,0,1,0,1}	107
{1,0,1,1,1,1,1,1}	19
{1,1,1,1,1,1,0,1}	109
{1,1,0,0,1,0,1,1}	39
{1,1,0,1,0,0,1,1}	89
{1,1,1,0,1,1,1,1}	55
{1,1,1,1,0,1,1,1}	73

$n = 8$

{1,0,0,0,1,1,1,0,1}	25
{1,0,1,1,1,0,0,0,1}	231
{1,0,0,1,0,1,0,1,1}	243
{1,1,0,1,0,1,0,0,1}	13
{1,0,0,1,0,1,1,0,1}	240
{1,0,1,1,0,1,0,0,1}	16
{1,0,1,0,0,1,1,0,1}	23
{1,0,1,1,0,0,1,0,1}	233
{1,0,1,0,1,1,1,1,1}	122
{1,1,1,1,1,0,1,0,1}	134
{1,0,1,1,0,0,0,1,1}	197
{1,1,0,0,0,1,1,0,1}	59
{1,1,0,0,0,0,1,1,1}	99
{1,1,1,0,0,0,0,1,1}	157
{1,1,1,0,0,1,1,1,1}	141
{1,1,1,1,0,0,1,1,1}	115

$n = 9$

{1,0,0,0,0,1,0,0,0,1}	130
{1,0,0,0,1,0,0,0,0,1}	382
{1,0,0,0,0,1,1,0,1,1}	197
{1,1,0,1,1,0,0,0,0,1}	315
{1,0,0,0,1,0,1,1,0,1}	275
{1,0,1,1,0,1,0,0,0,1}	237
{1,0,0,0,1,1,0,0,1,1}	104
{1,1,0,0,1,1,0,0,0,1}	408
{1,0,0,1,0,1,1,0,0,1}	140
{1,0,0,1,1,0,1,0,0,1}	372
{1,0,0,1,0,1,1,1,1,1}	53
{1,1,1,1,1,0,1,0,0,1}	459
{1,0,0,1,1,0,1,1,1,1}	93
{1,1,1,1,0,1,1,0,0,1}	419
{1,0,0,1,1,1,0,1,1,1}	443
{1,1,1,0,1,1,1,0,0,1}	69
{1,0,0,1,1,1,1,1,0,1}	234
{1,0,1,1,1,1,1,0,0,1}	278
{1,0,1,0,0,0,0,1,1,1}	194
{1,1,1,0,0,0,0,1,0,1}	318
{1,0,1,0,0,1,0,1,0,1}	226
{1,0,1,0,1,0,0,1,0,1}	286
{1,0,1,0,1,0,0,0,1,1]	36
{1,1,0,0,0,1,0,1,0,1}	476
{1,0,1,0,1,0,1,1,1,1}	453
{1,1,1,1,0,1,0,1,0,1]	59
{1,0,1,0,1,1,0,1,1,1}	461
{1,1,1,0,1,1,0,1,0,1}	51
{1,0,1,0,1,1,1,1,0,1}	26
{1,0,1,1,1,1,0,1,0,1}	486
{1,0,1,1,0,1,1,0,1,1}	501
{1,1,0,1,1,0,1,1,0,1}	11
{1,1,0,0,0,1,0,0,1,1}	114
{1,1,0,0,1,0,0,0,1,1}	398
{1,0,1,1,0,0,1,1,1,1}	441
{1,1,1,1,0,0,1,1,0,1}	71
{1,1,0,0,0,1,1,1,1,1}	428
{1,1,1,1,1,0,0,0,1,1}	84
{1,1,0,0,1,1,1,0,1,1}	249
{1,1,0,1,1,1,0,0,1,1}	263
{1,1,0,1,0,0,1,1,1,1}	135
{1,1,1,1,0,0,1,0,1,1}	377
{1,1,0,1,0,1,1,0,1,1}	126
{1,1,0,1,1,0,1,0,1,1}	386
{1,1,0,1,1,1,1,1,1,1}	404
{1,1,1,1,1,1,1,0,1,1}	108
{1,1,1,0,0,0,1,1,1,1}	327
{1,1,1,1,0,0,0,1,1,1}	185

$n = 10$

{1,0,0,0,0,0,0,1,0,0,1}	77
{1,0,0,1,0,0,0,0,0,0,1}	947
{1,0,0,0,0,0,1,1,0,1,1}	493
{1,1,0,1,1,0,0,0,0,0,1}	531
{1,0,0,0,0,1,0,0,1,1,1}	85
{1,1,1,0,0,1,0,0,0,0,1}	939
{1,0,0,0,0,1,0,1,1,0,1}	181
{1,0,1,1,0,1,0,0,0,0,1}	843
{1,0,0,0,1,1,0,0,1,0,1}	687
{1,0,1,0,0,1,1,0,0,0,1}	337
{1,0,0,0,1,1,0,1,1,1,1}	575
{1,1,1,1,0,1,1,0,0,0,1}	449
{1,0,0,1,0,0,0,1,0,1,1}	967
{1,1,0,1,0,0,0,1,0,0,1}	57
{1,0,0,1,1,0,0,0,1,0,1}	84
{1,0,1,0,0,0,1,1,0,0,1}	940
{1,0,0,1,1,0,1,0,1,1,1}	945
{1,1,1,0,1,0,1,1,0,0,1}	79
{1,0,0,1,1,1,0,0,1,1,1}	474
{1,1,1,0,0,1,1,1,0,0,1}	550
{1,0,0,1,1,1,1,0,0,1,1}	944
{1,1,0,0,1,1,1,1,0,0,1}	80
{1,0,0,1,1,1,1,1,1,1,1}	586
{1,1,1,1,1,1,1,1,0,0,1}	438
{1,0,1,0,0,0,0,1,1,0,1}	921
{1,0,1,1,0,0,0,0,1,0,1}	103
{1,0,1,0,0,1,0,0,0,1,1}	220
{1,1,0,0,0,1,0,0,1,0,1}	804
{1,0,1,0,0,1,1,1,1,0,1}	130
{1,0,1,1,1,1,0,0,1,0,1}	894
{1,0,1,0,1,0,0,0,0,1,1]	756
{1,1,0,0,0,0,1,0,1,0,1}	268
{1,0,1,0,1,0,1,0,1,1,1}	747
{1,1,1,0,1,0,1,0,1,0,1}	277
{1,0,1,0,1,1,0,1,0,1,1}	290
{1,1,0,1,0,1,1,0,1,0,1}	734
{1,0,1,1,0,0,0,1,1,1,1}	764
{1,1,1,1,0,0,0,1,1,0,1}	260
{1,0,1,1,0,0,1,0,1,1,1}	765
{1,1,1,0,1,0,0,1,1,0,1}	259
{1,0,1,1,1,0,0,0,1,1,1}	992
{1,1,1,0,0,0,1,1,1,0,1}	32

{1,0,1,1,1,1,1,0,1,1,1}	274
{1,1,1,0,1,1,1,1,1,0,1}	750
{1,0,1,1,1,1,1,1,0,1,1}	848
{1,1,0,1,1,1,1,1,1,0,1}	176
{1,1,0,0,0,0,1,0,0,1,1}	355
{1,1,0,0,1,0,0,0,0,1,1}	669
{1,1,0,0,0,1,1,0,1,1,1}	992
{1,1,1,0,1,1,0,0,0,1,1}	32
{1,1,0,0,1,0,0,1,1,1,1}	712
{1,1,1,1,0,0,1,0,0,1,1}	312
{1,1,0,0,1,0,1,1,0,1,1}	898
{1,1,0,1,1,0,1,0,0,1,1}	126
{1,1,0,0,1,1,1,1,1,1,1}	422
{1,1,1,1,1,1,1,0,0,1,1}	602
{1,1,1,0,0,0,1,0,1,1,1}	699
{1,1,1,0,1,0,0,0,1,1,1}	325
{1,1,0,1,1,0,1,1,1,1,1}	623
{1,1,1,1,1,0,1,1,0,1,1}	401

n = 11

{1,0,0,0,0,0,0,0,0,1,0,1}	1029
{1,0,1,0,0,0,0,0,0,0,0,1}	1019
{1,0,0,0,0,0,0,1,0,1,1,1}	846
{1,1,1,0,1,0,0,0,0,0,0,1}	1202
{1,0,0,0,0,0,1,0,1,0,1,1}	441
{1,1,0,1,0,1,0,0,0,0,0,1}	1607
{1,0,0,0,0,0,1,0,1,1,0,1}	1889
{1,0,1,1,0,1,0,0,0,0,0,1}	159
{1,0,0,0,0,1,0,0,0,1,1,1}	218
{1,1,1,0,0,0,1,0,0,0,0,1}	1830
{1,0,0,0,0,1,1,0,0,0,1,1}	860
{1,1,0,0,0,1,1,0,0,0,0,1}	1188
{1,0,0,0,0,1,1,0,0,1,0,1}	600
{1,0,1,0,0,1,1,0,0,0,0,1}	1448
{1,0,0,0,0,1,1,1,0,0,0,1}	874
{1,0,0,0,1,1,1,0,0,0,0,1}	1174
{1,0,0,0,0,1,1,1,1,0,1,1}	1109
{1,1,0,1,1,1,1,0,0,0,0,1}	939
{1,0,0,0,1,0,0,0,1,1,0,1}	1173
{1,0,1,1,0,0,0,1,0,0,0,1}	875
{1,0,0,0,1,0,0,1,0,1,0,1}	1343
{1,0,1,0,1,0,0,1,0,0,0,1}	705
{1,0,0,0,1,0,0,1,1,1,1,1}	2023
{1,1,1,1,1,0,0,1,0,0,0,1}	25
{1,0,0,0,1,0,1,0,1,0,0,1}	378
{1,0,0,1,0,1,0,1,0,0,0,1}	1670
{1,0,0,0,1,0,1,1,0,0,0,1}	706
{1,0,0,0,1,1,0,1,0,0,0,1}	3142
{1,0,0,0,1,1,0,0,1,1,1,1}	869
{1,1,1,1,0,0,1,1,0,0,0,1}	1179
{1,0,0,0,1,1,1,0,0,1,1,1}	1392
{1,1,1,0,0,1,1,1,0,0,0,1}	656
{1,0,0,0,1,1,1,0,1,0,1,1}	1192
{1,1,0,1,0,1,1,1,0,0,0,1}	856
{1,0,0,0,1,1,1,1,0,1,0,1}	959
{1,0,1,0,1,1,1,1,0,0,0,1}	1089
{1,0,0,1,0,0,0,0,1,1,0,1}	433
{1,0,1,1,0,0,0,0,1,0,0,1}	1615
{1,0,0,1,0,0,0,1,0,0,1,1}	1800
{1,1,0,0,1,0,0,0,1,0,0,1}	248
{1,0,0,1,0,0,1,0,0,1,0,1}	1524
{1,0,1,0,0,1,0,0,1,0,0,1}	524
{1,0,0,1,0,0,1,0,1,0,0,1}	789
{1,0,0,1,0,0,1,1,1,0,1,1}	1764
{1,1,0,1,1,1,0,0,1,0,0,1}	284
{1,0,0,1,0,0,1,1,1,1,0,1}	727
{1,0,1,1,1,1,0,0,1,0,0,1}	1321
{1,0,0,1,0,1,0,0,0,1,0,1}	1680
{1,0,1,0,0,0,1,0,1,0,0,1}	368
{1,0,0,1,0,1,0,1,1,0,1,1}	1673
{1,1,0,1,1,0,1,0,1,0,0,1}	375
{1,0,0,1,0,1,1,1,0,1,0,1}	1152
{1,0,1,0,1,1,1,0,1,0,0,1}	896
{1,0,0,1,0,1,1,1,1,1,1,1}	1451
{1,1,1,1,1,1,1,0,1,0,0,1}	597
{1,0,0,1,1,0,0,0,0,0,1,1}	842
{1,1,0,0,0,0,0,1,1,0,0,1}	1206
{1,0,0,1,1,0,0,0,1,1,1,1}	889
{1,1,1,1,0,0,0,1,1,0,0,1}	1159
{1,0,0,1,1,0,1,0,1,0,1,1}	1945
{1,1,0,1,0,1,0,1,1,0,0,1}	103
{1,0,0,1,1,0,1,0,1,1,0,1}	1357
{1,0,1,1,0,1,0,1,1,0,0,1}	691
{1,0,0,1,1,0,1,1,1,0,0,1}	1003
{1,0,0,1,1,1,0,1,1,0,0,1}	1045
{1,0,0,1,1,1,0,0,0,1,1,1}	365
{1,1,1,0,0,0,1,1,1,0,0,1}	1683
{1,0,0,1,1,1,1,0,0,1,0,1}	980
{1,0,1,0,0,1,1,1,1,0,0,1}	1068
{1,0,0,1,1,1,1,1,0,1,1,1}	625
{1,1,1,0,1,1,1,1,1,0,0,1}	1423
{1,0,1,0,0,0,0,0,0,1,1,1}	53
{1,1,1,0,0,0,0,0,0,1,0,1}	1995
{1,0,1,0,0,0,0,1,0,0,1,1}	251
{1,1,0,0,1,0,0,0,0,1,0,1}	1797
{1,0,1,0,0,0,0,1,0,1,0,1}	1088
{1,0,1,0,1,0,0,0,0,1,0,1}	960
{1,0,1,0,0,1,1,0,1,1,0,1}	112
{1,0,1,1,0,1,1,0,0,1,0,1}	1936
{1,0,1,0,0,1,1,1,1,1,1,1}	190
{1,1,1,1,1,1,1,0,0,1,0,1}	1858
{1,0,1,0,1,0,0,1,1,1,0,1}	1114
{1,0,1,1,1,0,0,1,0,1,0,1}	934
{1,0,1,0,1,0,1,0,0,1,1,1}	1822
{1,1,1,0,0,1,0,1,0,1,0,1}	226
{1,0,1,0,1,0,1,0,1,0,1,1}	997
{1,1,0,1,0,1,0,1,0,1,0,1}	1051
{1,0,1,0,1,0,1,1,0,0,1,1}	824
{1,1,0,0,1,1,0,1,0,1,0,1}	1224
{1,0,1,0,1,0,1,1,0,1,0,1}	1072
{1,0,1,0,1,1,0,1,0,1,0,1}	976
{1,0,1,0,1,1,0,1,1,1,1,1}	573
{1,1,1,1,1,0,1,1,0,1,0,1}	1475
{1,0,1,0,1,1,1,0,1,1,1,1}	815
{1,1,1,1,0,1,1,1,0,1,0,1}	1233
{1,0,1,0,1,1,1,1,1,0,1,1}	1527
{1,1,0,1,1,1,1,1,0,1,0,1}	521
{1,0,1,1,0,0,0,0,0,0,1,1}	911
{1,1,0,0,0,0,0,0,1,1,0,1}	1137
{1,0,1,1,0,0,1,1,0,0,1,1}	619
{1,1,0,0,1,1,0,0,1,1,0,1}	1429
{1,0,1,1,0,0,1,1,1,1,1,1}	1696
{1,1,1,1,1,1,0,0,1,1,0,1}	352
{1,0,1,1,0,1,0,0,1,0,1,1}	335
{1,1,0,1,0,0,1,0,1,1,0,1}	1713
{1,0,1,1,0,1,0,1,1,1,1,1}	107
{1,1,1,1,1,0,1,0,1,1,0,1}	1941
{1,0,1,1,0,1,1,0,1,1,1,1}	1990
{1,1,1,1,0,1,1,0,1,1,0,1}	58
{1,0,1,1,0,1,1,1,1,1,0,1}	1992
{1,0,1,1,1,1,1,0,1,1,0,1}	56
{1,0,1,1,1,0,0,0,0,1,1,1}	664
{1,1,1,0,0,0,0,1,1,1,0,1}	1384
{1,0,1,1,1,0,1,0,1,1,1,1}	555
{1,1,1,1,0,1,0,1,1,1,0,1}	1493
{1,0,1,1,1,0,0,0,1,0,1,1}	897
{1,1,0,1,0,0,0,1,1,1,0,1}	1151
{1,0,1,1,1,0,0,1,0,0,1,1}	109
{1,1,0,0,1,0,0,1,1,1,0,1}	1939
{1,0,1,1,1,0,1,1,0,1,1,1}	1399
{1,1,1,0,1,1,0,1,1,1,0,1}	649
{1,0,1,1,1,0,1,1,1,1,0,1}	473
{1,0,1,1,1,1,0,1,1,1,0,1}	1575
{1,0,1,1,1,1,0,1,1,0,1,1}	1636
{1,1,0,1,1,0,1,1,1,1,0,1}	412
{1,0,1,1,1,1,1,0,0,1,1,1}	909
{1,1,1,0,0,1,1,1,1,1,0,1}	1139
{1,1,0,0,0,0,0,0,1,0,1,1}	309
{1,1,0,1,0,0,0,0,0,0,1,1}	1739
{1,1,0,0,0,0,0,1,1,1,1,1}	1207
{1,1,1,1,1,0,0,0,0,0,1,1}	841
{1,1,0,0,0,1,0,1,0,1,1,1}	1504
{1,1,1,0,1,0,1,0,0,0,1,1}	544
{1,1,0,0,0,1,1,0,1,0,1,1}	827
{1,1,0,1,0,1,1,0,0,0,1,1}	1221
{1,1,0,0,0,1,1,1,0,0,1,1}	1476
{1,1,0,0,1,1,1,0,0,0,1,1}	572
{1,1,0,0,1,0,0,1,0,1,1,1}	801
{1,1,1,0,1,0,0,1,0,0,1,1}	1247
{1,1,0,0,1,0,0,1,1,0,1,1}	1172
{1,1,0,1,1,0,0,1,0,0,1,1}	876
{1,1,0,0,1,0,1,1,0,0,1,1}	237
{1,1,0,0,1,1,0,1,0,0,1,1}	1811
{1,1,0,0,1,0,1,1,1,1,1,1}	1529
{1,1,1,1,1,1,0,1,0,0,1,1}	519
{1,1,0,0,1,1,0,0,0,1,1,1}	1210
{1,1,1,0,0,0,1,1,0,0,1,1}	838
{1,1,0,0,1,1,1,0,1,0,0,1}	950
{1,0,0,1,0,1,1,1,0,0,1,1}	1098
{1,1,0,0,1,1,1,1,0,1,1,1}	548
{1,1,1,0,1,1,1,1,0,0,1,1}	1500
{1,1,0,1,0,0,0,0,1,1,1,1}	373
{1,1,1,1,0,0,0,0,1,0,1,1}	1675
{1,1,0,1,0,0,1,0,0,1,1,1}	1353
{1,1,1,0,0,1,0,0,1,0,1,1}	695
{1,1,0,1,0,1,0,0,0,1,1,1}	1496
{1,1,1,0,0,0,1,0,1,0,1,1}	552
{1,1,0,1,0,1,1,0,1,1,1,1}	627

{1,1,1,1,0,1,1,0,1,0,1,1}	1421
{1,1,0,1,1,0,1,1,1,0,1,1}	767
{1,1,0,1,1,1,0,1,1,0,1,1}	1281
{1,1,0,1,1,0,0,1,1,1,1,1}	329
{1,1,1,1,1,0,0,1,1,0,1,1}	1719
{1,1,0,1,1,1,0,1,0,1,1,1}	477
{1,1,1,0,1,0,1,1,1,0,1,1}	1571
{1,1,0,1,1,1,1,0,0,1,1,1}	1155
{1,1,1,0,0,1,1,1,1,0,1,1}	893
{1,1,1,0,0,0,1,0,0,1,1,1}	718
{1,1,1,0,0,1,0,0,0,1,1,1}	1330
{1,1,1,0,0,1,0,1,1,1,1,1}	898
{1,1,1,1,1,0,1,0,0,1,1,1}	1150
{1,1,1,0,1,0,0,1,1,1,1,1}	69
{1,1,1,1,1,0,0,1,0,1,1,1}	1979
{1,1,1,0,1,1,0,0,1,1,1,1}	1465
{1,1,1,1,0,0,1,1,0,1,1,1}	583

n = 12

{1,1,0,0,1,0,1,0,0,0,0,0,1}	1937
{1,0,0,1,0,1,1,0,0,0,0,0,1}	271
{1,1,0,1,1,1,1,0,0,0,0,0,1}	3495
{1,0,1,1,1,1,1,0,0,0,0,0,1}	2887
{1,0,0,1,1,0,0,1,0,0,0,0,1}	64
{1,0,0,0,1,0,1,1,0,0,0,0,1}	966
{1,1,0,1,0,1,1,1,0,0,0,0,1}	2063
{1,1,1,0,0,0,0,0,1,0,0,0,1}	1808
{1,1,1,1,1,0,0,0,1,0,0,0,1}	2367
{1,1,0,0,0,1,0,0,1,0,0,0,1}	1761
{1,1,0,1,1,1,0,0,1,0,0,0,1}	1804
{1,1,1,1,0,0,1,0,1,0,0,0,1}	761
{1,1,1,0,1,0,1,0,1,0,0,0,1}	3692
{1,0,0,0,0,1,1,0,1,0,0,0,1}	3130
{1,1,0,1,0,1,1,0,1,0,0,0,1}	1500
{1,0,1,0,0,0,0,1,1,0,0,0,1}	2684
{1,1,0,0,1,1,0,1,1,0,0,0,1}	1771
{1,0,0,1,1,0,1,1,1,0,0,0,1}	3648
{1,1,1,1,1,0,1,1,1,0,0,0,1}	3096
{1,0,1,1,0,0,0,0,0,1,0,0,1}	506
{1,1,1,0,1,1,0,0,0,1,0,0,1}	4032
{1,0,1,1,1,1,0,0,0,1,0,0,1}	3940
{1,1,1,0,0,1,1,0,0,1,0,0,1}	3014
{1,1,0,0,1,1,1,0,0,1,0,0,1}	2813
{1,1,1,1,1,1,1,0,0,1,0,0,1}	3022
{1,0,0,1,1,1,0,1,0,1,0,0,1}	3960
{1,0,0,0,0,0,1,1,0,1,0,0,1}	3825
{1,1,0,1,0,0,1,1,0,1,0,0,1}	566
{1,1,1,1,0,0,0,0,1,1,0,0,1}	582
{1,0,1,1,1,0,0,0,1,1,0,0,1}	1481
{1,0,0,0,0,1,0,0,1,1,0,0,1}	4032
{1,0,0,1,1,1,0,0,1,1,0,0,1}	3099
{1,1,1,1,1,1,0,0,1,1,0,0,1}	843
{1,0,1,1,0,0,1,0,1,1,0,0,1}	4032
{1,0,0,0,1,1,1,0,1,1,0,0,1}	448
{1,0,0,1,1,0,0,1,1,1,0,0,1}	997
{1,1,0,0,0,1,0,1,1,1,0,0,1}	1078
{1,0,0,1,0,1,0,1,1,1,0,0,1}	136
{1,1,1,0,0,0,0,0,0,0,1,0,1}	2368
{1,0,0,0,1,1,0,0,0,0,1,0,1}	1412
{1,1,1,0,1,1,0,0,0,0,1,0,1}	2642
{1,1,1,1,0,0,1,0,0,0,1,0,1}	1509
{1,0,1,1,1,0,1,0,0,0,1,0,1}	3517
{1,1,1,0,0,1,1,0,0,0,1,0,1}	3857
{1,0,1,0,1,1,1,0,0,0,1,0,1}	2177
{1,1,1,0,0,1,0,1,0,0,1,0,1}	244
{1,0,1,1,0,1,0,1,0,0,1,0,1}	1608
{1,1,0,0,1,0,1,1,0,0,1,0,1}	3594
{1,1,1,1,0,0,0,0,1,0,1,0,1}	2475
{1,0,1,1,1,0,0,0,1,0,1,0,1}	1350
{1,0,1,1,0,0,1,0,1,0,1,0,1}	4070
{1,1,0,0,1,0,0,1,1,0,1,0,1}	3178
{1,0,1,0,0,0,1,1,1,0,1,0,1}	1919
{1,1,1,0,1,0,1,1,1,0,1,0,1}	2771
{1,0,1,1,1,0,1,1,1,0,1,0,1}	2254
{1,1,0,1,0,1,1,1,1,0,1,0,1}	1030
{1,0,0,1,0,0,0,0,1,1,0,1}	3590
{1,1,1,0,0,0,1,0,0,1,1,0,1}	3678
{1,0,1,0,1,0,1,0,0,1,1,0,1}	26
{1,0,0,1,1,0,1,0,0,1,1,0,1}	64
{1,0,1,0,0,1,0,1,0,1,1,0,1}	2488
{1,0,1,1,1,1,0,1,0,1,1,0,1}	2418
{1,0,1,0,1,0,0,0,1,1,1,0,1}	2746
{1,0,0,1,1,0,0,0,1,1,1,0,1}	2615
{1,1,0,0,0,0,1,0,1,1,1,0,1}	1991
{1,0,1,0,0,0,1,0,1,1,1,0,1}	579
{1,0,1,0,1,1,1,0,1,1,1,0,1}	1842
{1,0,0,1,0,0,0,1,1,1,1,0,1}	156
{1,0,1,1,0,1,0,1,1,1,1,0,1}	1678
{1,1,0,0,1,1,0,1,1,1,1,0,1}	3118
{1,1,1,1,1,1,0,1,1,1,1,0,1}	86
{1,0,0,0,0,0,1,1,1,1,1,0,1}	1209
{1,1,1,0,1,0,1,0,0,0,0,1,1}	3304
{1,0,1,1,1,0,1,0,0,0,0,1,1}	2105
{1,0,0,0,1,0,0,1,0,0,0,1,1}	2335
{1,1,1,0,1,0,0,1,0,0,0,1,1}	2696
{1,0,0,1,1,1,0,1,0,0,0,1,1}	3018
{1,1,1,1,0,1,1,1,0,0,0,1,1}	1358
{1,1,0,1,1,0,0,0,1,0,0,1,1}	3156
{1,0,1,0,1,1,0,0,1,0,0,1,1}	918
{1,0,0,0,0,0,1,0,1,0,0,1,1}	2159
{1,0,1,0,0,1,1,0,1,0,0,1,1}	502
{1,1,0,1,1,1,1,0,1,0,0,1,1}	3709
{1,1,0,1,0,0,0,1,1,0,0,1,1}	2106
{1,0,0,0,1,1,0,1,1,0,0,1,1}	2325
{1,0,1,1,1,1,0,1,1,0,0,1,1}	978
{1,0,0,1,0,0,1,1,1,0,0,1,1}	1283
{1,1,1,1,0,0,1,1,1,0,0,1,1}	620
{1,1,1,0,0,1,1,1,1,0,0,1,1}	1564
{1,1,0,1,1,0,0,0,0,1,0,1,1}	872
{1,1,0,1,0,1,0,0,0,1,0,1,1}	3815
{1,1,0,0,1,1,0,0,0,1,0,1,1}	1990
{1,0,0,1,0,1,1,0,0,1,0,1,1}	3530
{1,1,0,1,0,0,0,1,0,1,0,1,1}	281
{1,0,0,0,1,0,1,1,0,1,0,1,1}	2596
{1,0,0,0,0,1,1,1,0,1,0,1,1}	2033
{1,0,1,0,1,1,1,1,0,1,0,1,1}	3066
{1,1,0,1,0,0,0,0,1,1,0,1,1}	3224
{1,1,0,0,1,0,0,0,1,1,0,1,1}	940
{1,1,1,1,1,0,0,0,1,1,0,1,1}	774
{1,1,1,0,1,0,1,0,1,1,0,1,1}	2935
{1,0,0,0,1,0,0,1,1,1,0,1,1}	2292
{1,1,1,0,0,1,0,1,1,1,0,1,1}	2250
{1,1,1,1,1,1,0,1,1,1,0,1,1}	1217
{1,0,0,0,0,0,1,1,1,1,0,1,1}	601
{1,1,0,0,1,0,1,1,1,1,0,1,1}	387
{1,0,1,0,0,0,0,0,0,0,1,1,1}	1728
{1,0,0,0,1,0,0,0,0,0,1,1,1}	2288
{1,1,1,0,1,0,0,0,0,0,1,1,1}	2695
{1,1,1,0,0,1,0,0,0,0,1,1,1}	598
{1,0,1,1,0,0,1,0,0,0,1,1,1}	418
{1,1,1,0,0,0,0,1,0,0,1,1,1}	3498
{1,1,1,1,1,0,0,1,0,0,1,1,1}	2830
{1,0,1,0,0,1,0,1,0,0,1,1,1}	3852
{1,1,0,1,1,1,0,1,0,0,1,1,1}	1846
{1,0,1,0,0,0,1,1,0,0,1,1,1}	239
{1,0,0,1,0,0,1,1,0,0,1,1,1}	1082
{1,1,1,1,0,0,1,1,0,0,1,1,1}	4041
{1,1,0,0,1,1,1,1,0,0,1,1,1}	2532
{1,1,1,0,0,0,0,0,1,0,1,1,1}	1401
{1,1,0,0,0,1,0,0,1,0,1,1,1}	1400
{1,1,0,0,0,0,1,0,1,0,1,1,1}	792
{1,0,0,0,1,0,1,0,1,0,1,1,1}	404
{1,1,0,1,1,0,1,0,1,0,1,1,1}	1161
{1,0,1,0,1,1,1,0,1,0,1,1,1}	1325
{1,0,1,0,0,0,0,1,1,0,1,1,1}	1454
{1,0,0,1,0,0,0,1,1,0,1,1,1}	64
{1,0,1,0,1,0,0,0,0,1,1,1,1}	1621
{1,0,0,1,1,0,0,0,0,1,1,1,1}	3514
{1,1,1,1,0,1,0,0,0,1,1,1,1}	1877
{1,0,1,0,0,0,1,0,0,1,1,1,1}	2587
{1,0,0,0,1,0,1,0,0,1,1,1,1}	3335
{1,1,1,0,0,1,1,0,0,1,1,1,1}	55
{1,1,0,0,1,1,1,0,0,1,1,1,1}	3476
{1,1,1,1,0,0,0,1,0,1,1,1,1}	2219
{1,1,0,0,0,1,1,1,0,1,1,1,1}	2738
{1,0,0,0,1,0,0,0,1,1,1,1,1}	1729
{1,1,0,1,1,0,0,0,1,1,1,1,1}	3322
{1,1,1,0,0,1,0,0,1,1,1,1,1}	1266
{1,0,0,0,1,1,1,0,1,1,1,1,1}	1000
{1,0,0,1,1,0,0,1,1,1,1,1,1}	3253
{1,1,0,1,1,1,0,1,1,1,1,1,1}	2879
{1,0,1,1,1,1,0,1,1,1,1,1,1}	4010
{1,0,0,1,0,0,1,1,1,1,1,1,1}	1074

Appendix D - APPROPRIATE POLYNOMIALS FOR MULTIFOCAL ELECTROPHYSIOLOGY

Degree14

{1,0,0,0,0,0,0,0,0,1,0,1,0,1,1} 15807
{1,0,0,0,0,0,0,1,0,1,0,1,1,1,1} 4471

Degree 15

{1,1,1,1,1,1,0,1,0,0,1,0,1,0,1,1} 8751
{1,0,1,1,0,1,0,1,0,1,0,0,0,1,1,1} 9645

Degree 16

{1,1,0,1,0,0,0,0,0,0,0,0,1,0,0,0,1} 15942
{1,1,0,1,0,0,1,1,0,0,0,1,1,1,0,0,1} 29486