Orthogonal Chaotic Vector Shift Keying

in

Digital Communications

By

Timothy J. Wren

A thesis submitted for the degree of

Doctor of Philosophy

in the

University of Sussex

June 2007

Department of Engineering and Design

School of Science and Technology

University of Sussex

Brighton

England

Statement

I hereby declare that this thesis has not been and will not be, submitted in whole or in part to another University for the award of any other degree.

Signature:……

T J Wren

UNIVERSITY OF SUSSEX

TIMOTHY JON WREN
DOCTOR OF PHILOSOPHY

ORTHOGONAL CHAOTIC VECTOR SHIFT KEYING COMMUNICATIONS

SUMMARY

The main contribution of this research work is a new digital communication method called Orthogonal Chaotic Vector Shift Keying. The new method is based on orthogonal chaotic signal sequences. It is shown that under this multilevel digital communication scheme, better transmission efficiency, greater robustness and noise rejection can be achieved. Some more detailed descriptions of the major contributions are as follows.

(1) The new method has significant improvements in transmission efficiency over some existing methods, for example some typical M-ary type two dimensional quadrature schemes. The information transmitted per unit time is shown to be dependent on the dimensions adopted under the new method.

(2) Similarly, the noise rejection has been greatly improved due to the increased "inter-symbolic distances". Therefore, the new method tends to be more robust.

(3) Within the physical limits of communication channels, the new method provides a way of increasing the security of digital communications.

(4) New methods for the characterization and simple modelling of the noise transmission behaviours of a number of communication schemes, including the new one proposed in this thesis, have been developed. In addition, an analytical formula of the Bit Error Rates for all these schemes has been derived.

(5) An investigation of the "optimal" dimensionality of the new method, in order to achieve a balance between performance improvement and computational complexity, has been undertaken.

Implementation of the proposed scheme in this thesis would be interesting research to follow. In addition, the topic of Orthogonal Chaotic Vector Shift Keying is mathematically very rich, and a variation of the new method, that offers potential improvements in robustness, appears attractive and worthy of further exploration.

Acknowledgements

I wish to express my gratitude to my supervisor Dr. Tai C. Yang for his encouragement and guidance throughout. In particular, he has allowed me to pursue my own course of research and trip over my own mistakes, whilst ensuring that I have finally arrived here at the end. Also, thanks are due to Dr. Rupert Young for occasional course correction and grounding.

I would like to thank General Dynamics UK Limited for allowing me the time to study for this degree, and I would especially like to thank Mr. Stuart McCormick of General Dynamics for his constant encouragement and assistance over the long course of study.

Finally, I would like to thank my partner Julie Morgan for always being there, and being understanding, when the pressure of study weighed heavily upon me.

Contents

List of Figures 9

List of Symbols 13

List of Abbreviations 23

Chapter 1 Introduction 24

Chapter 2 Background and Literature Survey 29

2.1 Introduction 29
2.2 Chaotic Systems 31

2.2.1 Divergence Theorem 33
2.2.2 Lyapunov Exponents 36
2.2.3 Fractal Dimensions 39
2.2.4 Lorenz System 43
2.2.5 Alternative Systems 47

2.3 Synchronization Methods 47

2.3.1 Drive Response Synchronization 48
2.3.2 Signal Masking 56
2.3.3 Parameter Variation 59
2.3.4 Chaotic Attractor Synchronization 62

2.4 Non-Reference Correlation Methods 66

2.4.1 Symmetric Chaos Shift Keying 66
2.4.2 Correlation Delay Shift Keying 69

2.5 Reference Correlation Methods 71

2.5.1 Differential Chaos Shift Keying 71
2.5.2 FM Differential Chaos Shift Keying 74
2.5.3 Quadrature Chaos Shift Keying 77

2.6 Observer Methods 82
2.7 Feedback Methods 84
2.8 Detectability 86
2.9 Summary 87

Chapter 3 An Orthogonal Chaotic Vector Shift Keying Communication Scheme 88

3.1 Introduction 88
3.2 Limitations of Two Dimensional Schemes with Increased Transmission Efficiency 89

3.3	Orthogonal Properties of Extended Dimension Fourier Generated Signals	90
3.4	Theoretical Analysis	93
3.5	Generation of Orthogonal Signal Sets	95
3.6	System Architectures and Encoding and Decoding Schemes	98
	3.6.1 Direct '*m* Symbol' 'U' Scheme	98
	3.6.2 Indirect '*m* Symbol' 'X' Scheme	102
	3.6.3 Indirect Persistent 'x' Scheme	104
3.7	BER Probability Formulation	106
3.8	Signal Characterization	106
	3.8.1 Direct '*m* Symbol' 'U' Scheme	107
	3.8.2 Indirect '*m* Symbol' 'X' Scheme	109
3.9	Signal to Noise Calculations	110
	3.9.1 Direct '*m* Symbol' 'U' Scheme	111
	3.9.2 Indirect '*m* Symbol' 'X' Scheme	112
3.10	Summary	114
Chapter 4	Simulation Case Study	115
4.1	Introduction	115
4.2	Transmission Simulations	117
	4.2.1 Direct '*m* Symbol' 'U' Scheme	117
	4.2.2 Indirect '*m* Symbol' 'X' Scheme	120
	4.2.3 Indirect persistent 'x' Scheme	123
4.3	BER Simulations	128
	4.3.1 Direct '*m* Symbol' 'U' Scheme	128
	4.3.2 Indirect '*m* Symbol' 'X' Scheme and Indirect Persistent 'x' Scheme	129
	4.3.2.1 W Matrix Case 'A'	130
	4.3.2.2 W Matrix Case 'B'	132
	4.3.2.3 W Matrix Case 'C'	134
4.4	Summary	136

Chapter 5 Optimal Dimensionality 137

5.1 Introduction 137
5.2 Volumetric Considerations 138
5.3 Comparative Function 139
5.4 Dimensional Comparative Simulation 143
5.5 Summary 144

Chapter 6 Conclusions 145

References 148

Appendix A 157

A.1 Properties of Sinusoids 157
A.2 Fourier Transform Pairs 158

A.2.1 OrthogonalSignal 163

A.3 Lyapunov Exponent Calculation Listings 165

A.3.1 CalcLyapunov 164
A.3.2 Lyapunov 164
A.3.3 Lorenz_Variational 166

Appendix B 168

B.1 Gram-Schmidt Method 168

Appendix C 171

C.1 Transmission Simulation MATLAB Function Listings 171

C.1.1 SystemU Function 171

C.1.1.1 Initialize Function 176

C.1.1.1.1 Map Function 177

C.1.1.2 Chaos Function 177

C.1.1.2.1 Lorenz Function 178

C.1.1.3 Gram Schmidt Function 178
C.1.1.4 Normalize Function 179

C.1.2 SystemVec Function 179

C.2 BER Simulation MATLAB Function Listings 183

C.2.1 Ugen Function 183
C.2.2 Xgen Function 185
C.2.3 EUgen Function 188
C.2.4 EXgen Function 190

Appendix D 194

D.1 Hypersphere Volume Calculations 194
D.2 Maxima of Comparative Function 199
D.3 Matlab Optimal Dimension Listing 201

List of Figures

Chapter 2

Figure 2.2.1	Single State Output of a Dissipative System with a Trajectory on a Stable Manifold	32
Figure 2.2.1.1	Time Evolution of a Small Regional Bounded Volume of the State Space	34
Figure 2.2.2.1	Exponential Sensitive Dependence on Initial Conditions	36
Figure 2.2.3.1	Capacity Dimension Box Counting	40
Figure 2.2.3.2	First Four Stages of the Koch Curve	41
Figure 2.2.4.1	Lyapunov Exponent Evolution Plots	45
Figure 2.2.4.2	Lorenz System - State Space Trajectory	46
Figure 2.2.4.3	Lorenz System - Lorenz System – Surface Type Characteristics	46
Figure 2.3.1.1	Drive Response System	51
Figure 2.3.1.2	Simulation Results without Noise	52
Figure 2.3.1.3	Simulation Results with Gaussian Noise Mean $\mu = 0$, Variance $\sigma^2 = 1$	53
Figure 2.3.1.4	Simulation Results with Gaussian Noise Mean $\mu = 0$, Variance $\sigma^2 = 25$	54
Figure 2.3.1.5	Simulink Model of Synchronization System	55
Figure 2.3.2.1	Signal Masking System Architecture	57
Figure 2.3.2.2	Signal Masking Messages Simulation	58
Figure 2.3.3.1	Parameter Variation System Architecture	60
Figure 2.3.3.2	Signal Masking Messages Simulation	61
Figure 2.3.4.1	Chaotic Attractor Synchronization System Architecture	63
Figure 2.3.4.2-A	Chaotic Attractor Synchronization Messages Simulation	64
Figure 2.3.4.2-B	Chaotic Attractor Synchronization Messages Simulation	65

Figure 2.4.1.1	Symmetric Chaos Shift Keying Architecture	66
Figure 2.4.2.1	Correlation Delay Shift Keying Architecture	69
Figure 2.5.1.1	Differential Chaos Shift Keying Architecture	72
Figure 2.5.3.1	Orthogonal Signal Set	78
Figure 2.5.3.2	Maximal Separation Quadrature Constellations Existing on a Two Dimensional Hypersphere	80

Chapter 3

Figure 3.2.1	M-ary Two Dimensional Constellations	89
Figure 3.2.2	QAM Type Two Dimensional Constellations	90
Figure 3.5.1	Signal Sampling to Matrix Concept	96
Figure 3.6.1.1	'U' Scheme - Direct '*m* Symbol' Transmission System Architecture	98
Figure 3.6.2.1	'X' Scheme-Indirect '*m* Symbol' Transmission System Architecture	102
Figure 3.6.3.1	'x' Scheme - Indirect Persistent Transmission System Architecture	104

Chapter 4

Figure 4.2.1.1	'U' Scheme - Direct '*m* Symbol' System Transmission Simulations Power of Signal to Noise Ratio = 1.0	118
Figure 4.2.1.2	'U' Scheme - Direct '*m* Symbol' System Message Transmissions Power of Signal to Noise Ratio = 1.0	119
Figure 4.2.2.1	'X' Scheme - Direct '*m* Symbol' System Transmission Signals Power of Signal to Noise Ratio = 1.0	121
Figure 4.2.2.2	'X' Scheme - Direct '*m* Symbol' System Transmission Simulations Power of Signal to Noise Ratio = 1.0	122

Figure 4.2.3.1 Indirect Persistent ‘x’ Scheme
System Transmission Signals
Power of Signal to Noise Ratio = 1.0 124

Figure 4.2.3.2 Indirect Persistent ‘x’ Scheme
System Message Transmissions
Power of Signal to Noise Ratio = 1.0 125

Figure 4.2.3.3 Indirect Persistent ‘x’ Scheme
System Transmission Signals
Power of Signal to Noise Ratio = 10.0 126

Figure 4.2.3.4 Indirect Persistent ‘x’ Scheme
System Message Transmissions
Power of Signal to Noise Ratio = 10.0 127

Figure 4.3.1.1 Direct ‘*m*’ Symbol ‘U’ Scheme
BER v P_{snr} and $\frac{E_b}{N_0}$ Comparison Plot
for $n \in [8,32,128]$ samples 128

Figure 4.3.1.2 Direct ‘*m*’ Symbol ‘U’ Scheme
Comparison of BER versus P_{snr} and $\frac{E_b}{N_0}$ for
DCSK and OCVSK 16 Scheme 129

Figure 4.3.2.1.1 BER v P_{snr} and $\frac{E_b}{N_0}$ W Scheme ‘A’ :
Comparison Plot for $n \in [8,32,128]$ samples 131

Figure 4.3.2.1.2 W Scheme ‘A’ Comparison of
BER versus P_{snr} and $\frac{E_b}{N_0}$ for DCSK and OCVSK 16 132

Figure 4.3.2.2.1 BER v P_{snr} and $\frac{E_b}{N_0}$ W Scheme ‘B’ :
Comparison Plot for $n \in [8,32,128]$ samples 133

Figure 4.3.2.2.2 W Scheme ‘B’ Comparison of
BER versus P_{snr} and $\frac{E_b}{N_0}$ for DCSK and OCVSK 16 134

Figure 4.3.2.3.1 BER v P_{snr} and $\frac{E_b}{N_0}$ W Scheme ‘C’ :
Comparison Plot for $n \in [8,32,128]$ samples 135

Figure 4.3.2.3.2 W Scheme 'C' Comparison of BER versus P_{snr} and $\frac{E_b}{N_0}$ for DCSK and OCVSK 16 136

Chapter 5

Figure 5.3.1 Comparative Positioning of M-Ary Constellation and OCVSK Inter-symbolic Distance 140

Figure 5.3.2 Comparative Plot of M-Ary Constellation and OCVSK Inter-symbolic Distance 140

Figure 5.3.3 Optimal Dimensional Value derived from Volumetric Considerations 142

Figure 5.4.1 BER Ratio Dimensional Comparison 143

Appendix A

Figure A.2.1 Resultant Phasors for $h(t) = \alpha \cos \omega_0 t$ 160

Figure A.2.2 Resultant Phasors for $h(t) = \alpha \sin \omega_0 t$ 162

Figure A.2.3 Resultant Phasors for $h(t) = \alpha \sin(\omega_0 t + \varphi)$ 164

Appendix B

Figure B.1.1 First Stage of Gram-Schmidt Method 168

Appendix D

Figure D.1.1 Calculation of Dimension *m* Hyperspherical Volume 194

List of Symbols

Chapter 2

Section 2.2.1

$V(t)$	Volume of a measure of the state space at time t
$\int_V$	Integral over the volume bounded by closed surface S
∇	Nabla vector operator
$\nabla \cdot \mathbf{f}(\mathbf{x})$	Divergence of $\mathbf{f}(\mathbf{x})$
δV	Small change in volume
$\prod_{i=1}^{n}$	Product over n
$\delta x_i(t)$	Small change in state space dimension x_i at time t
$\dot{x}_i = f_i(x_1 \cdots x_n)$	i^{th} Nonlinear state equation
$x_i(t)$	i^{th} State at time t
$\mathbf{A}$	General linear state matrix
$Tr(\mathbf{A})$	The trace $\mathbf{A}$
$\sum_{i=1}^{n}$	Arithmetic sum over n
λ_i	i^{th} Eigenvalue of matrix $\mathbf{A}$

Section 2.2.2

$\mathbf{x}_i(t)$	i^{th} State vector at time t
$\mathbf{\Delta x}(t)$	Small difference in state vectors at time t
$\lvert\mathbf{x}\rvert$	Magnitude of state vector $\mathbf{x}$
R	Hypersphere radius
∞	Infinity
h	Lyapunov Exponent value
α	Arbitrary multiplying constant
ln	Natural logarithm
δt	Small change in time t

$\dot{\mathbf{\Delta}}(t)$	Derivative of small difference in state vectors with respect to time
$\frac{\partial \mathbf{f}(\mathbf{x}(t))}{\partial \mathbf{x}(t)}$	Partial derivative of vector state function with respect to the state vector

Section 2.2.3

D_0	Capacity dimension
ε	Length of side of hypercube for calculating capacity dimension
$M(\varepsilon)$	Number of hypercubes covering a specific set
$\lim_{\varepsilon \to 0}$	Limiting function as hypercube dimension tends towards zero
L	Nominal length of curve forming the set
A	Nominal area of region forming the set
K	Arbitrary multiplying constant
n	Transformation iteration
D_1	Information dimension
C_i	Generalized hypercube index
T	Time that chaotic trajectory has been measured
$\eta(C_i, \mathbf{x}(0), T)$	Time measure for a hypercube
μ_i	Natural measure for a hypercube
D_L	Lyapunov dimension
h_i	i^{th} Lyapunov Exponent
K	Number of Lyapunov Exponents greater than or equal to zero
m	Order of chaotic system
$\text{sgn}(x)$	Sign function of x

Section 2.2.4

σ, r, β	Coefficients of the Lorenz system
$\mathbf{J}(\mathbf{x})$	Jacobian of vector $\mathbf{x}$

Section 2.3.1

$\mathbf{u}$	Complete state vector
$\mathbf{f}(\mathbf{u})$	State vector function
v	Drive subsystem single state variable
$\mathbf{w}$	Drive subsystem partial state vector
$g(v,\mathbf{w})$	Singular state function
$\mathbf{h}(v,\mathbf{w})$	Partial state vector function
x,y,z	Independent state variables
v'	Response subsystem single state variable
$\mathbf{w}'$	Response subsystem partial state vector
$\frac{\mathbf{I}_3}{s}$	Third order diagonal matrix of integrators
$\mathbf{c}^T$	System measurement matrix
μ	Mean of noise signal
σ^2	Variance of noise signal

Section 2.4.1

$s(t)$	Resultant modulation signal
$\hat{\mathbf{x}}$	Remaining non symmetric state vector
x_i	Symmetric state
$f_i(x_i,\hat{\mathbf{x}})$	i^{th} State equation in symmetric and non symmetric state variables
τ	Sample period
M	Number of sample periods in the correlation interval
S	Correlation resultant
$\hat{x}_i(t)$	i^{th} State estimate at time t
$r(t)$	Received message bearing signal
b_k	Modulation value
$e(t), n(t)$	Gaussian white noise signals
$\varepsilon(t)$	Resultant noise signal
$\mathrm{E}\{e(t)\}$	Expected value of $e(t)$
P_x, P_y	Power of signal $x(t), y(t)$

Section 2.4.2

Z^{-L}	Time delay operator for L time delays of τ sample period
$r(t-L\tau)$	Signal $r(t)$ delayed by L time delays of τ sample period

Section 2.5.2

$f_i(t)$	i^{th} FM-DCSK orthogonal function
E_x	Energy per bit of signal
T	Reference and modulation signal interval
W_i	i^{th} Walsh function matrix

Section 2.5.3

ω	Fundamental frequency
f_0	DC amplitude of Fourier expansion
f_m	Amplitude of m^{th} multiple of fundamental frequency
ϕ_m	Phase shift of m^{th} multiple of fundamental frequency
α, β	Arbitrary phase shifts
$x \perp y$	x is orthogonal to y
c_r, c_i	Real and imaginary coefficients of encoding symbol

Section 2.6

$\mathbf{x}(t)$	Transmitter state
$\mathbf{f}(\mathbf{x})$	Transmitter chaotic system vector function
$y(t)$	Transmitter output signal
$\mathbf{c}^T$	Transmitter output signal state combination vector
$\hat{\mathbf{x}}(t)$	Estimated transmitter state
$\hat{y}(t)$	Estimated transmitter output signal
$\mathbf{m}$	Observer signal measurement gain
$\mathbf{e}$	State error vector

Section 2.7

$\mathbf{z}(t)$	Dissimilar system state vector
$\mathbf{g}(\mathbf{z})$	Dissimilar chaotic system vector function
$\mathbf{u}(t)$	Nonlinear control vector
$\boldsymbol{\varepsilon}(\mathbf{z})$	Dissimilar/Transmitter error vector function
$V(\mathbf{e})$	Lyapunov error function
$\dot{V}(\mathbf{e})$	Derivative of Lyapunov error function

Section 2.8

X_n	Value of n^{th} return map value of $x(t)$ at local maxima
Y_m	Value of m^{th} return map value of $x(t)$ at local minima

Chapter 3

Section 3.3

q	Fourier sum limit
$y_p(t)$	p^{th} Derived Fourier orthogonal function of t
$F(p,k)$	Gray scale function evaluated at ± 1 dependent on p and k
I	Resultant of Fourier integral
$G(k)$	Resultant function of Gray scale functions valued at ± 1

Section 3.4

m	Dimension of required orthogonal basis
$u_i(t)$	Orthogonal basis function of t
c_k	k^{th} encoding coefficient
$s(t)$	Resultant signal function
$\mathbf{c}$	Encoding vector
$\mathbf{u}(t)$	Orthogonal signals vector

Section 3.5

n	Dimension of the hyperspace
m	Dimension of the subspace within the n space

$\mathbf{p}$	General position vector in the n space
p_i	i^{th} dimension coefficient of the $\mathbf{p}$ vector
$\mathbf{s}$	Vector representing a symbol on the hypersurface
$\mathbf{u}_i$	i^{th} Orthogonal n dimensional basis vector
R^n	Real vector space of dimension n
$\mathbf{U}$	Orthogonal matrix contains m n dimensional basis vectors
$R^{n\times m}$	Real matrix space of dimension $n\times m$
$x(t)$	Continuous chaotic signal
X_i	Mean value of $x(t)$ over n samples
T	Sample interval
$\mathbf{x}_i$	i^{th} Sample chaotic n dimensional vector
$\mathbf{X}$	Chaotic sampled matrix containing m n dimensional vectors
$rank(\mathbf{X})$	Value of the rank of $\mathbf{X}$
$\mathbf{W}$	Upper triangular square transformation matrix
$\mathbf{I}_m$	Identity matrix of dimension m
B_e	'bits' in precision error
C_n	2-norm matrix condition number
λ_i	i^{th} Eigenvalue of $\mathbf{X}^T\mathbf{X}$

Section 3.6.1

$\mathbf{P}$	Diagonal power balancing matrix
$\mathbf{Q}$	Streamable reference transmission matrix
$\mathbf{s}_i$	i^{th} Streamable encoded symbol transmission matrix
$\mathbf{S}$	Streamable m encoded symbol transmission matrix
$\mathbf{C}$	m Symbol encoding matrix
$\mathbf{c}_i$	i^{th} Symbol encoding vector
$R^{m\times 2^m}$	Real matrix space of dimension $m\times 2^m$
$\overline{\mathbf{C}}$	Complete symbol encoding matrix map
$\overline{\mathbf{c}}_i$	i^{th} Map symbol encoding vector
$\mathbf{b}(j)$	Bit pattern function for representing j

$\boldsymbol{\mu}_m$	Ones vector of length m
b_m	Bit pattern element m
j	Symbol encoding map index
$\overline{\mathbf{S}}$	Noisy received sample encoded symbol signal matrix
$\overline{\mathbf{s}}_i$	i^{th} Received sample encoded symbol signal vector
σ	Gaussian white noise variance
$\boldsymbol{\varepsilon}_i$	i^{th} Gaussian white noise vector of length n
$\overline{\mathbf{Q}}$	Noisy received sample reference signal matrix
$\hat{\mathbf{s}}_i$	i^{th} Estimated sample encoded symbol signal vector
$\hat{\mathbf{c}}_i$	i^{th} Estimated symbol encoding vector
$\mathbf{e}_i$	i^{th} Sample encoded symbol vector error
η_i	i^{th} Squared error sum cost function
$\frac{\partial \mathbf{e}_i}{\partial \hat{\mathbf{c}}_i}$	Partial derivative of $\mathbf{e}_i$ with respect to $\hat{\mathbf{c}}_i$
$\mathbf{0}^T$	Transposed zero vector

Section 3.6.2

$\mathbf{Z}$	Transmittable power balanced and normalized reference matrix
$\mathbf{X}$	Normalized chaotic sampled matrix
$\mathbf{x}_m$	m^{th} Normalized chaotic sampled vector
p	Scalar power balancing gain

Section 3.6.3

$\mathbf{z}$	Transmittable power balanced and normalized reference vector
$\overline{\mathbf{Z}}_{n,m-1}$	Column deficient transmittable power balanced normalized reference matrix
$\overline{\mathbf{Z}}_{n,m}$	Full rank transmittable power balanced normalized reference matrix
$\mathbf{X}_{n,m-1}$	Column deficient sampled chaotic matrix
$\mathbf{X}_{n,m}$	Full rank sampled chaotic matrix

Section 3.7

P_{snr}	Signal to noise power ratio
μ	Number of symbols
b	Number of bits
$P(C_\mu)$	Probability of all μ symbols are correct
$P(E_\mu)$	Probability of any error in μ symbols
$P(C_i)$	Probability of i^{th} symbol being correct
$P(E_i)$	Probability of i^{th} symbol being in error
BER	Bit Error Rate

Section 3.8.1

$\mathbf{E}$	Gaussian white noise matrix of dimension $n \times m$
$\boldsymbol{\varepsilon}$	Gaussian white noise vector of dimension $\underline{n}$
$\|\mathbf{x}\|$	2-Norm of vector $\mathbf{x}$
$\mathbf{W}$	Upper triangular matrix representative of a Gram-Schmidt transform
$\mathbf{w}_i$	i^{th} Column of upper triangular matrix $\mathbf{W}$
$\mathbf{G}(\ )$	Gram-Schmidt function

Section 3.8.2

$\overline{\mathbf{U}}$	Noise contaminated orthonormal set of basis vectors

Section 3.9.1

$\mathbf{E}_{m,m}$	Gaussian white noise matrix of dimension $m \times m$
$\boldsymbol{\varepsilon}_m$	Gaussian white noise vector of dimension n
$\frac{E_b}{N_0}$	Energy per bit divided by the noise power
B_r	Transmission bit rate
τ	Sampling time of the system

Chapter 5

Section 5.2

$V(m)$	Volume of a unit hypersphere of dimension m
r	Radius of hypersphere
β	Banding variance in the hyperspherical radius
σ^2	Noise variance of symbol position on the hypersphere
$\Gamma(z)$	Gamma function of z
R_m	Hyperspherical to hypercubic volumetric ratio of dimension m

Section 5.3

d	Inter-symbolic distance
C_m	Comparative function relating R_m to R_2
$\varphi(z)$	Digamma function of z

Appendix A

$H(j\omega)$	Fourier transform of $h(t)$
$\delta(\omega)$	Dirac delta function of ω
$\delta_a(\omega)$	Nascent Dirac delta function

Appendix B

$\mathbf{x}_i$	i^{th} real subspace vector spanning m dimensional space
$\mathbf{v}_i$	i^{th} real orthogonal subspace vector
$\mathbf{u}_i$	i^{th} real orthonormal basis subspace vector
α_k	Magnitude of orthogonal subspace vector $\mathbf{v}_k$

Appendix D

ρ	Instrumental subspace radius variable
z	Integral dimensional variable
$V_m(r)$	Volume of m dimensional hypersphere of radius r
$\alpha(m)$	Constant volumetric multiplier for m dimensional hypersphere

$I(m)$	Integral sine function of dimension m
$\Pi(m)$	Sine integral product function of dimension m
$x!$	Factorial of x function

List of Abbreviations

BER	Bit Error Rate
BPSK	Binary Phase Shift Keying
CDSK	Correlation Delay Shift Keying
DCSK	Differential Chaos Shift Keying
FFT	Fast Fourier Transform
FM-DCSK	Frequency Modulated Differential Chaos Shift Keying
M-ary	M level symbol encoding
M-DCSK	M-ary Differential Chaos Shift Keying
OCVSK	Orthogonal Chaotic Vector Shift Keying
QAM	Quadrature Amplitude Modulation
QCSK	Quadrature Chaos Shift Keying
QPSK	Quadrature Phase Shift Keying
SCSK	Symmetrical Chaos Shift Keying
SD-DSCK	Symbol Dynamics Differential Chaos Shift Keying

Chapter 1

Introduction

The essential element of a communication scheme is to ensure that whatever message is transmitted at the transmitter the same message can be received correctly at the receiver. In both the military and the commercial sectors there is a further requirement for communications to be secure. If the means of transmission is easily detectable then this requires that the signal be disguised in some way, usually by some form of encryption. However, if the signal itself is not easily detectable, then the requirement for this method of disguise is not so important. Whether the signal that carries the message is secure or not, it is required that the transmission and receiving process is capable of rejecting noise introduced by the transmission channels. In digital communications, methods of rejecting noise by the addition of error correction codes have been extensively used. They have a rigorous mathematical basis, but reduce the transmission efficiency and information density of any scheme that incorporates them during encoding. Error correction is not a topic considered in this thesis, but it is an important consideration when any communication schemes are being designed. This thesis concentrates on the investigation of the means of carrying the signal and ensuring that the communications are as robust and unaffected by noise as possible.

Many methods over the past five decades have been developed. One of the best known and documented is Binary Phase Shift Keying (BPSK) and its many derivatives. These methods suffer from loss of signal and poor noise rejection due to the frequency ranges

used. There is a large body of work dedicated to this field, briefly described in [1-3]. They emphasize the simplicity of schemes utilizing these methods but indicate why methods employing spread spectrum techniques have been adopted. They show how spread spectrum methods give good noise rejection and more robust communications. The modulating signal for such schemes would be one carrying all frequencies with a specific characteristic for the desired transmission channel. This thesis shows that a good method of generating something that closely approaches this type of signal, and gives deterministic results, is the use of chaotic signals generated by chaotic attractors embedded within nonlinear systems.

To ensure that communication links are secure the following approaches could be considered:

1. Encrypt the message on a detectable modulation signal.
2. Choose an effectively undetectable modulation signal with no encryption.
3. Choose the modulation signal to ensure that it occupies only the frequency bands that could be considered as noise.

This thesis intends to choose the second option, but bears in mind the idea of option three as a good basis for the following themes under consideration.

Main Themes

1. Secure communications before considering encryption.
2. Good noise rejection and spectral efficiency.
3. Transmission efficiency/Information density.

These themes can be expanded into a thought tree, which gives a broad indication of why chaotic processes are thought to be the best choice for secure systems, with a robust nature and good noise rejection.

1. Secure communications
    a. Not easily detected
        i. Hidden within noisy environments.
            - Good noise rejection.
                - Spread spectrum techniques
                - Chaotic methods
        ii. Varying non repetitive signals
            - Deterministic range of frequencies for ‘no carrier’ type schemes
                - Chaotic methods
2. Slow techniques for assured transmissions
    a. Complex methods
        i. High information density and transmission efficiency
            - Abstract forms
                - Chaotic methods
        ii. Low detectability for high security
            - Varying Complexity
                - Chaotic methods

Therefore, methods of transmission encoding which have good noise rejection, and are robust, suggests a form of spread spectrum method using some form of deterministic variable frequency method. Rather than modulating frequencies in some arbitrary manner, a simpler method would be to use chaotic processes which inherently have all the necessary required properties.

A novel method has been introduced in this thesis that uses chaotic processes to achieve the above set of desired themes. The approach and the conclusions to support this are laid out in the following chapters.

Chapter 2 presents a literature survey of various theoretical design methods for secure communication schemes; in particular, the use of non-linear chaotic systems as a modulating medium for encoding message streams.

The survey in this chapter covers; (1) the fundamental theory needed to understand and make full use of chaotic systems within the proposed communication schemes; (2) full descriptions and theoretical analysis of a number of existing schemes; (3) a brief description of a number of different schemes derived from control theory and (4) an introduction to detectability of some schemes.

In chapter 3 a number of new multidimensional transmission encoding schemes are described. The limitations on the practically achievable higher transmission rates, under the existing two-dimensional schemes, due to their increased sensitivity to noise level, are reviewed. The problems of extending the Fourier expansion method to higher dimensions are investigated. This leads to a clear statement about the problem associated with multidimensional communication schemes. To solve this problem, a new theory of orthogonal decomposition for multidimensional schemes is derived, and a simple robust method of obtaining a multidimensional set of orthogonal signals is presented. A number of encoding and decoding schemes are then derived, along with their corresponding system architectures. In the final part, some further topics related to each of these encoding schemes are investigated and some new results are derived in a novel way. These topics are: the Bit Error Rate (BER) probability, signal characterization and signal to noise ratio effects on the BER.

A case study based on a multidimensional system is presented in chapter 4. The chapter is divided into two parts. The first part presents simulation results for all the proposed schemes. In these simulations, simulated real time random messages are transmitted and received with Gaussian White noise added to the signal in the communication channel. The second part presents the Bit Error Rates (BER) analysis in terms of the Signal to Noise Ratio Power and the Energy per Bit divided by the Noise Power; by using signal characterization, it demonstrates clearly the problems associated with transmitting non-orthogonal reference signal sequences. For a number of non-orthogonal schemes it introduces a characterization matrix for the nature of the non-orthogonality and demonstrates their simulated reduced ability to reject noise. This characterization in turn, demonstrates the need for signal set independence for the novel communication scheme introduced in chapter 3. Also the generalized data rate for purely orthogonal multidimensional schemes is derived.

In chapter 5, based on a set of reasonable assumptions generally in line with common practice, the optimal dimensionality of the proposed multidimensional schemes is investigated. A formula for the optimality is derived. The considerations for choosing the optimal dimensions are to balance the relationship between; (1) the volumes of the communication space; (2) the surface area of the hypersphere where the symbolic constellations lie; and (3) the computational loading that increases with the order of the chosen dimension.

Conclusions drawn in chapter 6 review, the main achievements in this thesis, some suggestions for further research to extend the proposed methods and introduce a completely new mathematically rich approach.

While the main results are presented in the body of the thesis, the appendices give important supporting material, derivations and extensive proofs. These include, appendix (A): the properties of sinusoids and Fourier transforms, appendix (B): the Gram-Schmidt orthogonalization method, appendix (C): simulation listings and appendix (D): optimal dimensionality assumptions and derivations.

Chapter 2

Background and Literature Survey

2.1 Introduction

This chapter brings together a broad area of work on the theoretical design of secure communication schemes; particularly it addresses the use of non-linear chaotic systems as a modulating medium for encoding message streams. In addition, it looks at the detection of some of these processes and shows how this knowledge can assist in the design of schemes whose detectability is greatly reduced.

In order to use chaotic processes, it is necessary to understand the nature of these systems, at a basic level, to utilize them successfully for communications schemes. Section (2.2) introduces some basic ideas and necessary properties of chaotic systems and briefly describes their underlying properties. This section also introduces a simple system which will be used to simulate the schemes suggested in chapter 3. In addition the required nature and properties of alternative systems is discussed.

Section (2.3) looks at a set of methods of communication which are collected under the heading of 'Synchronization Methods'. These methods require the transmitter and receiver to have dynamics which can be synchronized, via the message stream, and contain both the message and sufficient information, to ensure synchronization. The section describes the principal methods involved namely, Drive Response

Synchronization [4-9], Parameter Variation [6][10], Signal Masking [6][10][11] and Chaotic Attractor Synchronization [12].

Two other methods, one relying purely on correlation and the other requiring synchronization, are discussed in section (2.4). These methods namely, Correlation Delay Shift Keying (CDSK) [13] and Symmetric Chaos Shift Keying (SCSK) [13], rely on recognition of functional shaping via correlation in the first case, and by correlation and a particularly careful choice of chaotic process, in the second. They have interesting properties but neither one is particularly robust in the presence of high levels of noise.

The final section (2.5) deals with types of method that embody some form of reference. The largest body of work has been focussed on these types of method and it is collected here under the name of 'Reference Correlation Methods'. The most prominent among these methods is Differential Chaos Shift Keying (DCSK) [13-19] and its derivatives M-DCSK, SD-DCSK [20-23] and FM-DCSK [17][19]. For this thesis the most important reference method, which initially stimulated the research work contained in the following chapters, is Quadrature Phase Shift Keying (QCSK) [24] and a full description of this method is given in section (2.5.3) along with the constellation forms M-ary Chaos Shift Keying.

A brief outline of observer based methods utilizing chaos [25-27] and methods involving feedback on observed states of chaotic systems, to control synchronization, [28] are included in sections (2.6) and (2.7).

Finally, the question of detectability is considered in section (2.8) [18][29-33] and the requirement for its counterpart, the encryption process [34-37]. Although this is not the principal theme of this thesis, it is worthy of mentioning, as it can greatly influence the design and theoretical approach when considering different types of scheme.

The design and performance of all these related techniques described or mentioned are in either the papers referred to, or in papers related specifically towards performance considerations [38-44], and are only included for completeness.

2.2 Chaotic Systems

All engineers are aware of dynamical systems that can be mathematically analysed, and in which the eventual settled motion is either a constant value or periodic. All linear systems fall into these two categories, but nonlinear systems can have other types of settled motion, which are more common than the more familiar linear ones. These dynamical systems have state representations and trajectories which will not yield to standard analytical techniques. Systems that are dissipative, in the sense that their 'energy' dissipates over time, are characterized by their trajectories settling to 'attractors' which are attracting sets or regions within the phase space. Within the linear class of systems this is characterized by the trajectory settling to the origin of the phase space. Conservative dynamical systems are not generally characterized by the presence of attractors. A given system can be characterized as conservative or dissipative by how a prescribed volume of a region of the state space behaves as time passes. That is, if a volume $V(\mathbf{x})$ bounded by a closed surface $S(t)$ decreases as time t increases, where $\mathbf{x} \in R^n$ is the state vector, then it is said to be dissipative [46]. If the volume remains constant it is said to be conservative. There are systems that have volumes that increase with time, but these are generally unstable, and are not of interest here. We can use the divergence theorem to determine if our system is either dissipative or conservative, this is discussed in section (2.2.3). A way of understanding the nature of attractors in a discursive manner is to consider firstly a two dimensional case. A point described by a state vector **x** that has evolved along a trajectory within the state space, cannot intersect any other part of the trajectory accept on an attractor. This implies that on an attractor, the state velocity vector lies in the same direction as the attractor, and this constitutes a stable manifold. If it indeed did intersect an existing part of the already evolved trajectory, then this would imply that the state space had not been fully specified, and the representation of the system was incorrect. So for the two dimensional example there are two possible outcomes; firstly, a singularity where the state vector remains constant and the state velocity vector is zero and secondly; a contour, which is known more commonly as a limit cycle, where the state velocity vector always lies on the contour for all values of the state vector on the contour. Now consider a three dimensional dissipative system whose final trajectory does not settle to a singularity or a simple curve in space, but exists on a stable three dimensional manifold. At no time

does the trajectory intersect itself, yet it has no identifiable limit cycle. A typical single state trajectory of this kind of system is shown in figure (2.2.1)

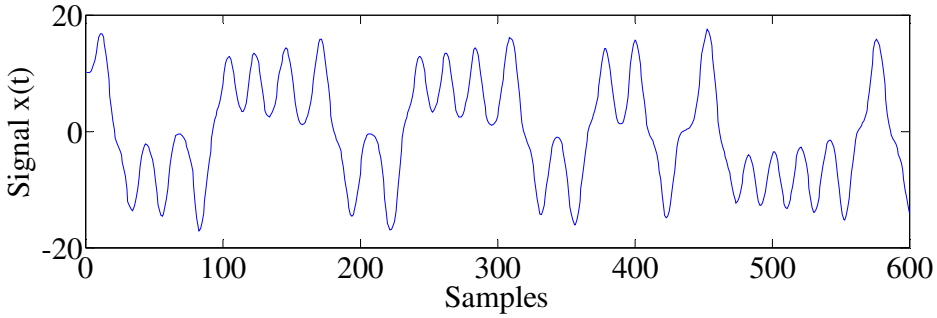


Figure 2.2.1

Single State Output of a Dissipative System with a Trajectory on a Stable Manifold

In the two-dimensional case, the possible limit cycles have a tangible dimension. In the case of the singularity, it has a dimension of zero whereas the dimension of the contoured limit cycle is one. So discursively the limit cycles, and hence the attractors, can be characterized by their dimension. Therefore, if the new three-dimensional trajectory constitutes an attractor it is necessary to determine a value attributable to its dimension. This will be discussed later in section (2.2.2.1).

All linear systems behaviour can be characterized by their sets of eigenvalues which, dependent on their position on the complex plane, determine the system's stability. Where the trajectory settles to a constant value the system is stable, it is unstable where it diverges to infinity and oscillatory where it remains on a cyclic contour, which is strictly not a limit cycle. In the case of non-linear systems the characterization is clearly not as straightforward. By linearizing the system equations at any particular point in the state space, the local stability can be determined from the localized eigenvalues, but the overall stability remains undetermined. However, due to work by Lyapunov there are characteristic exponents called 'Lyapunov Exponents' which can be used to characterize the system behaviour. These exponents, and how they can be used to determine the dimension of the attractor, will be discussed more fully in section (2.2.2). The Lyapunov Exponents measure the systems sensitivity to small changes in initial conditions and how the trajectory would evolve if displaced by a small distance as time elapses. The result is that for any system one Lyapunov Exponent is zero, which represents the divergence in the direction of the trajectory, and the others represent the

tendency of the evolution in orthogonal directions to it. If a system has at least one Lyapunov Exponent greater than zero, but less in magnitude than any other negative exponent, then the systems trajectory will settle onto a stable manifold which constitutes an attractor. The dimension of the trajectory's flow can be determined from a combination the Lyapunov Exponents, and if this is not an integer value the attractor is said to be 'strange', and the motion is termed chaotic.

In this thesis the system used is a modified form of one due to Lorenz [45]. The system and its properties are outlined in section (2.2.3).

2.2.1 Divergence Theorem

The statement of the Divergence Theorem is given as result (2.2.1.1) over a closed surface S containing a volume at time t of $V(t)$. The rate of change of volume with respect to time is equal to the integral of the divergence of the system vector function, with respect to the volume, bounded by a closed surface S.

Result 2.2.1.1

$$\frac{dV}{dt} = \int_V \nabla \cdot \mathbf{f}(\mathbf{x}) dV \tag{2.2.1.1}$$

Derivation 2.2.1.1

Consider a small volume in a state space of order n at time t and $t + \delta t$ bounded by the facets of an n dimensional hypercube each of length $\delta x_i \ \forall \ i \in [1, n]$

The change in volume from time t to $t + \delta t$ is given by

$$\delta V = \prod_{i=1}^{n} \delta x_i(t + \delta t) - \prod_{i=1}^{n} \delta x_i(t) \tag{2.2.1.2}$$

For the state system considering all $i \in [1, n]$

$$\dot{x}_i = f_i(x_1 \cdots x_n) \tag{2.2.1.3}$$

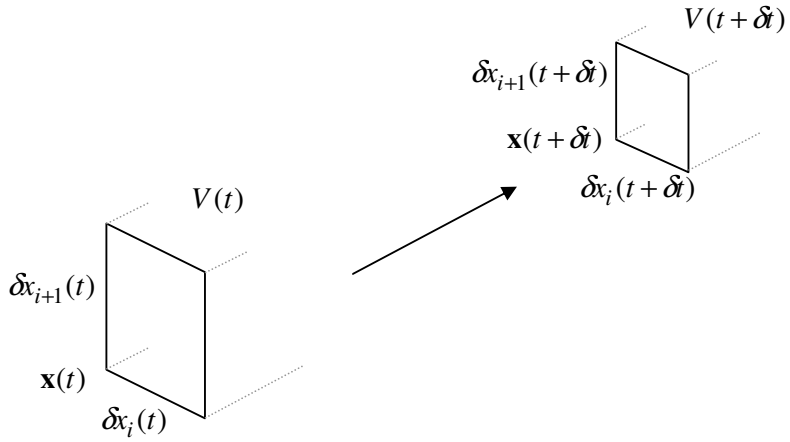


Figure 2.2.1.1

Time Evolution of a Small Regional Bounded Volume of the State Space

And each state evolves as

$$x_i(t+\delta t) = x_i(t) + \dot{x}_i(t)\delta t \tag{2.2.1.4}$$

This can be written as

$$x_i(t+\delta t) = x_i(t) + f_i(x_1 \cdots x_n)\delta t \tag{2.2.1.5}$$

So the change in the length of sides of the hypercube can be expressed as

$$\begin{aligned}\delta x_i(t+\delta t) &= \left(x_i(t) + \delta x_i(t) + f_i(x_1 \cdots x_i + \delta x_i \cdots x_n)\delta t\right) - \left(x_i(t) + f_i(x_1 \cdots x_n)\delta t\right) \\ &= \left[f_i(x_1 \cdots x_i + \delta x_i \cdots x_n) - f_i(x_1 \cdots x_n)\right]\delta t + \delta x_i(t)\end{aligned} \tag{2.2.1.6}$$

In the limit this gives

$$\delta x_i(t+\delta t) = \left[\frac{\partial f_i}{\partial x_i}\delta t + 1\right]\delta x_i(t) \tag{2.2.1.7}$$

So the change in volume becomes

$$\delta V = \prod_{i=1}^{n}\left[\frac{\partial f_i}{\partial x_i}\delta t + 1\right]\delta x_i(t) - \prod_{i=1}^{n}\delta x_i(t) \tag{2.2.1.8}$$

And this, for the small volume under consideration, reduces to

$$\delta V = \sum_{i=1}^{n}\frac{\partial f_i}{\partial x_i}\delta t \cdot \prod_{i=1}^{n}\delta x_i(t) \tag{2.2.1.9}$$

This implies the following statement of the divergence theorem over the complete volume bounded by the closed surface *S*

$$\frac{dV}{dt} = \int \cdots \int \nabla \cdot \mathbf{f}(x_1 \cdots x_n) dx_1 \cdots dx_n$$

$$= \int_V \nabla \cdot \mathbf{f}(\mathbf{x}) dV \qquad (2.2.1.10)$$

where ∇ is the nabla vector operator on a vector function $\mathbf{f}(\mathbf{x})$ and

$$\nabla \cdot \mathbf{f}(\mathbf{x}) = \sum_{i=1}^{n} \frac{\partial f_i(x_1 \cdots x_n)}{\partial x_i} \qquad (2.2.1.11)$$

If a system is dissipative, in the sense that the volume diminishes to zero over time, it does not imply that the system is stable; for example consider a linear system where

$$\mathbf{f}(\mathbf{x}) = \mathbf{A}\mathbf{x} \qquad (2.2.1.12)$$

Then

$$\nabla \cdot \mathbf{f}(\mathbf{x}) = Tr(\mathbf{A}) = \sum_{i=1}^{n} \lambda_i \qquad (2.2.1.13)$$

where λ_i are the eigenvalues of the matrix **A** then equation (2.1.2.10) becomes

$$\frac{dV}{dt} = Tr(\mathbf{A})V \qquad (2.2.1.14)$$

So the trace of the matrix **A** can be negative, and hence the volume approaches zero as time approaches infinity, but one or more of the eigenvalues could be positive implying that the system is unstable. Discursively, this can be interpreted as the order of the unstable eigenvalues must be less that the order of the system; and therefore the sub volume spanned by the trajectories of the unstable eigenvalues occupies zero volume in the whole state space.

2.2.2 Lyapunov Exponents

Lyapunov Exponents are a means of characterizing the nature of attractors of nonlinear systems. The principal defining property of an attractor, whose dynamics display apparently chaotic behaviour, is the exponential sensitivity to small changes in initial conditions. In a linear stable system these small changes would diminish to zero and the system's Lyapunov Exponents would be the real parts of the system's eigenvalues.

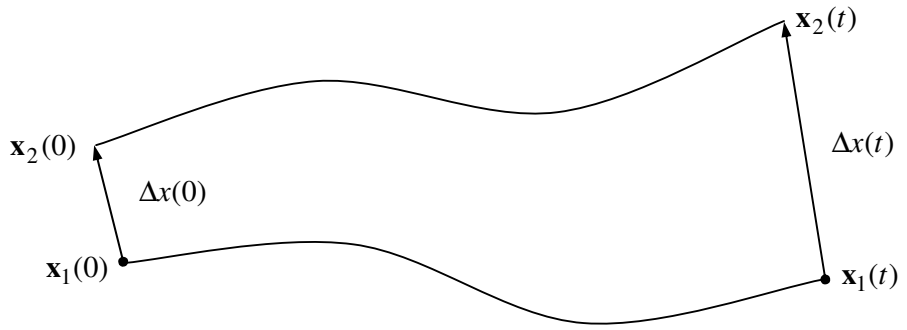


Figure 2.2.2.1
Exponential Sensitive Dependence on Initial Conditions

The two trajectories in figure (2.2.2.1) are separated at time $t = 0$ by a small change $\mathbf{\Delta x}(0)$ so that

$$\mathbf{x}_2(t) = \mathbf{x}_1(t) + \mathbf{\Delta x}(t) \tag{2.2.2.1}$$

This is the evolution equation of the second trajectory in terms of the first. If the trajectory has settled onto the attractor and if it is bounded by some real hypersphere of radius R such that

$$|\mathbf{x}| < R < \infty \tag{2.2.2.2}$$

then it is said to be chaotic if

$$\frac{|\mathbf{\Delta}(t)|}{|\mathbf{\Delta}(0)|} \approx e^{ht} \tag{2.2.2.3}$$

Where $h > 0$

Clearly this does not contain the whole truth, as the trajectory is not unstable with a single positive Lyapunov Exponent value *h*, as equation (2.2.2.3) would imply. The dimension of the state space could be characterized by as many Lyapunov Exponents as there are dimensions, in much the same way as a linear system has as many eigenvalues, each of which contribute to the overall behaviour of the system. Consideration needs to be made as to how the trajectories move apart in directions which are orthogonal to each other and, as the eigenvalues of a single point in a nonlinear system do not remain constant, the Lyapunov Exponents must be an aggregated measure of the behaviour over the whole of the attractor. In addition equation (2.2.2.3) implies that the delta difference between two trajectories, which are arbitrarily displaced by some small increment, needs to be considered. This can be simplified by considering some arbitrary infinitesimal displacement from the unperturbed trajectory. If the infinitesimal displacement is considered in the direction of the tangent vector, then the evolution of this vector, will determine the evolution of infinitesimal displacement from the trajectory.

The supposed exponential growth is an imposed structure on how the actual growth may or may not take place. Consequently, it is necessary to consider some form of averaging, to ensure that a true measure of the attractor can truly be determined. Consider the imposed structure of equation (2.2.2.3) rearranged to give

$$\left|\boldsymbol{\Delta}(t)\right| = \alpha e^{ht}\left|\boldsymbol{\Delta}(0)\right| \tag{2.2.2.4}$$

The dominant Lyapunov Exponent can be calculated from this equation as

$$h \approx \frac{1}{t}\ln\frac{\left|\boldsymbol{\Delta}(t)\right|}{\left|\boldsymbol{\Delta}(0)\right|} \tag{2.2.2.5}$$

but the terms in this equation can become exponentially large, over even a short period, and the nature of *h* can be quite severely distorted, simple by the fact that the attractor is nonlinear. It is better to consider a time averaging process, which includes an iterative realignment and normalization of the growth. Consider the development of equation (2.2.2.4) after a small time increment δt

$$\begin{aligned}\left|\boldsymbol{\Delta}(t+\delta t)\right| &= \alpha e^{h(t+\delta t)}\left|\boldsymbol{\Delta}(0)\right| \\ &= \alpha e^{ht}e^{h\delta t}\left|\boldsymbol{\Delta}(0)\right| \\ &= e^{h\delta t}\left|\boldsymbol{\Delta}(t)\right| \end{aligned} \tag{2.2.2.6}$$

then the approximate Lyapunov exponent at time *t*, for an increase in time δt, is given by

$$h \approx \frac{1}{\delta t} \ln \frac{|\mathbf{\Delta}(t+\delta t)|}{|\mathbf{\Delta}(t)|} \tag{2.2.2.6}$$

if the $|\mathbf{\Delta}(t)|$ is reset to unity at each iteration, over a time period in which the trajectory visits a large measure of the state space occupied by the attractor, then an approximation for the dominant Lyapunov Exponent is given by

$$h \approx \frac{1}{n\delta t} \sum_{i=1}^{n} \ln|\mathbf{\Delta}(t+i\delta t)| \tag{2.2.2.7}$$

The $\mathbf{\Delta}(t+\delta t)$ can be determined from the Jacobian variational equation of the system as an approximation to the continuous system as

$$\mathbf{\Delta}(t+\delta t) = \frac{\partial \mathbf{f}(\mathbf{x}(t))}{\partial \mathbf{x}(t)} \dot{\mathbf{\Delta}}(t)\delta t + \mathbf{\Delta}(t) \tag{2.2.2.8}$$

where $|\mathbf{\Delta}(t)| = 1$ at each iteration and δt is sufficiently small. This integration process can more accurately and easily be undertaken in Matlab by using various solvers of ordinary differential equations. Given that a satisfactory integration algorithm is used then the updated vector can be found. However, for a state space system of dimension *q* there are *q* Lyapunov Exponents. In order to separate out each of these exponents, without resorting to too complex methods, consider how an orthonormal set of basis vectors, based on the tangent vector, is transformed by the Jacobian variational equation over the time period δt; this is tantamount to considering how the tangent vector to the trajectory varies given an arbitrary starting point. Then if this distorted set of basis vectors has the growth in each direction determined, and it is re-orthonormalized at each iteration, then the effects of the non-linearities of the attractor can be ameliorated and the individual Lyapunov Exponents calculated. The re-orthonormalization of the distorted set of basis vectors is achieved using the Gram-Schmidt process, which also determines the orthogonal increase in the vector components of the resultant set. A Matlab algorithm for achieving this is listed in appendix (A). Methods are alluded to in [46], but the above developed method has proved to be very robust and reliable on all the chaotic systems it has been tested on.

2.2.3 Fractal Dimensions

The whole concept of dimension and measure is not within the scope of this thesis; neither for the purposes of the research is it necessary to fully understand it, however some information will prove useful for firstly illustrating the fractal nature of chaotic attractors, from an elegant conjecture proposed by Kaplan and Yorke [47], and secondly for choosing alternative systems in any further research.

An attractor is an infinite collection of points in some dimensional space and one of its basic properties is its dimension. If a simple stable linear system is considered then its attractor is a singular point, which is the steady state of the trajectory in the state space, and clearly this has a dimension of zero. Similarly, a non-linear two dimensional system with a limit cycle settles to a curve in the state space and has an understandable dimension of one. However, chaotic attractors do not have such obvious dimensions, and one of the properties of a chaotic or strange attractor, is that it has a fractional dimension termed a fractal dimension.

The most common technique for determining dimension is the capacity method or box counting method. Consider a set that lies in an m dimensional space and is covered by an m dimensional grid of hypercubes with length of side ε. Now the number of hypercubes $M(\varepsilon)$ required to cover the set increases as the length of side ε is diminished. The capacity dimension is given by the following equation

$$D_0 = \lim_{\varepsilon \to 0} \left(\frac{\ln M(\varepsilon)}{\ln\left(\frac{1}{\varepsilon}\right)} \right) \qquad (2.2.3.1)$$

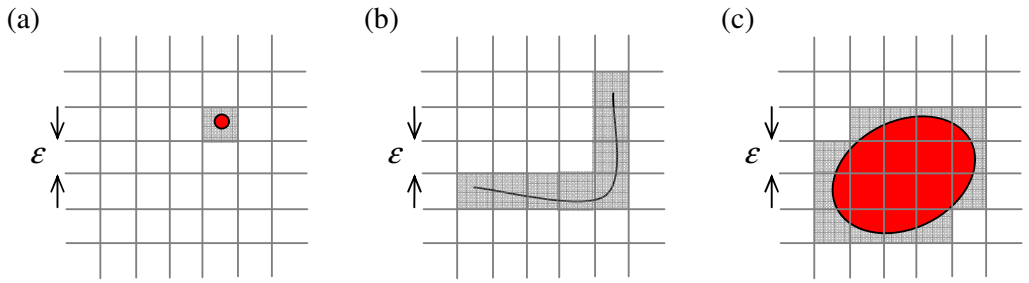


Figure 2.2.3.1

Capacity Dimension Box Counting

(a) Singular Point Set, (b) Curve Set and (c) Regional Set

Consider the three examples given in figure (2.2.3.1) for a dimension of $m = 2$.

For case (a) $M(\varepsilon) = 1$ and therefore

$D_0 = 0$ as $\varepsilon \to 0$.

For case (b) $M(\varepsilon) \propto \frac{L}{\varepsilon}$ where L is the length of the curve, therefore

$D_0 = 1$ as $\varepsilon \to 0$.

And finally

For case (c) $M(\varepsilon) \propto \frac{A}{\varepsilon^2}$ where A is the area of the region, therefore

$D_0 = 2$ as $\varepsilon \to 0$.

Now, as an example of a fractal dimension, consider the Koch Curve which is a well known fractal set. This is a set that can be generated by self similar operations, on a line interval existing on a unit length line segment, in a two dimensional space. The middle third of this line segment is then replaced by two one third line segments, arranged as shown in figure (2.2.3.2). At every subsequent stage, each of the line segments is replaced by self similar images of the original curve transformation, orientated appropriately. After a few operations the curve looks takes on a fixed appearance.

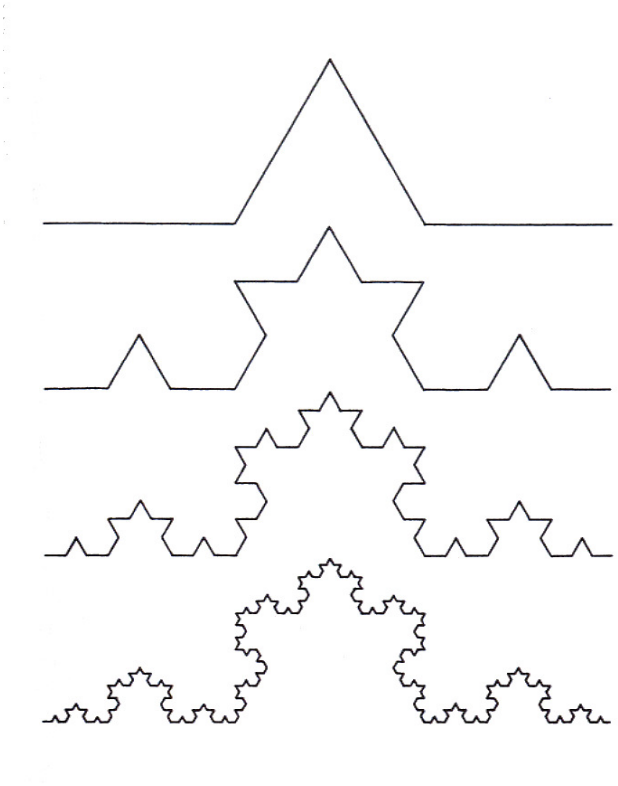


Figure 2.2.3.2

First Four Stages of the Koch Curve

Clearly the line length tends towards infinity since at each operation it increases by $\frac{4}{3}$ of the original length, but the area is clearly confined and is less than unity. In fact it can be trivially shown to tend towards $\frac{\sqrt{3}}{20}$.

So to calculate the capacity dimension choose an appropriate ε.

Let

$$\varepsilon = \left(\frac{1}{3}\right)^n \tag{2.2.3.2}$$

where *n* is the transformation iteration. Because the result is a line then $M(\varepsilon) \propto \frac{L}{\varepsilon}$ so

$$M(\varepsilon) = K\frac{\left(\frac{4}{3}\right)^n}{\left(\frac{1}{3}\right)^n} = K \cdot 4^n \tag{2.2.3.3}$$

where K is an arbitrary constant.

Finally the capacity dimension is given by

$$D_0 = \underset{n\to\infty}{Lim}\left(\frac{\ln K\cdot 4^n}{\ln 3^n}\right) = \underset{n\to\infty}{Lim}\left(\frac{\ln K + n\ln 4}{n\ln 3}\right) = \frac{\ln 4}{\ln 3} = 1.26186 \qquad (2.2.3.4)$$

This dimensional measure is useful for static sets but it does not account for a dynamical system, because the trajectory of the chaotic attractor in the state space can spend considerably longer in some regions than in others. Therefore, if a simple capacity method is used, as with the D_0 dimension, then an unrepresentative measure is obtained. A measure known as the information dimension has been defined and is referred to in [46] and is given by

$$D_1 = \underset{\varepsilon\to 0}{Lim}\left(\frac{\sum_{i=1}^{M(\varepsilon)}\mu_i \ln\mu_i}{\ln\varepsilon}\right) \qquad (2.2.3.5)$$

where the μ_i are the natural measures for each hypercube C_i. This is defined by the amount of time spent by the trajectory in that hypercube divided by the total time, as this time tends towards infinity, that is

$$\mu_i = \underset{T\to\infty}{Lim}\left(\frac{\eta(C_i,\mathbf{x}(0),T)}{T}\right) \qquad (2.2.3.6)$$

where $\eta(C_i,\mathbf{x}(0),T)$ is the time measure of a hypercube C_i with an initial state $\mathbf{x}(0)$ at time *T*.

This is clearly a very difficult dimension to evaluate, but a conjecture made by [46] states that the information dimension is the same as a measure of dimension known as the Lyapunov dimension for typical attractors, and is defined as

$$D_L = K + \frac{1}{|h_{K+1}|}\sum_{i=1}^{K} h_i \qquad (2.2.3.7)$$

where the Lyapunov Exponents h_i are arranged in ascending order and K is the largest integer given by the number of Lyapunov Exponents greater than or equal to zero.

That is

$$K = m - \frac{1}{2}\sum_{i=1}^{m}\left(\mathrm{sgn}^2(h_i) - \mathrm{sgn}(h_i)\right) \quad \text{where} \quad \mathrm{sgn}(x) = \left\{\begin{array}{rcl} -1 & : & x<0 \\ 0 & : & x=0 \\ 1 & : & x>0 \end{array}\right\} \tag{2.2.3.8}$$

for the Lorenz system described in section (2.2.4) the Lyapunov Exponents are given as

$h_1 = 0.832, \quad h_2 = 0.0, \quad h_3 = -14.504$ hence $K = 2$ and $D_L = 2 + \frac{h_1}{|h_3|} = 2.0574$.

This result is supported by inspection of figures (2.2.4.1) and (2.2.4.2), which show curved surface like behaviours in two distinct regions, suggesting a potentially slightly higher dimension than two.

2.2.4 Lorenz System

The Lorenz system of equations is well known [45]. The nonlinear state equations are given as the dynamical vector state equation

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) \tag{2.2.4.1}$$

and explicitly as

$$x_1(t) = -\sigma x_1(t) + \sigma x_2(t) \tag{2.2.4.2}$$

$$x_2(t) = r x_1(t) - x_2(t) - x_1(t) x_3(t) \tag{2.2.4.3}$$

$$x_3(t) = x_1(t) x_2(t) - \beta x_3(t) \tag{2.1.4.4}$$

If the parameters are set to $\sigma = 10$, $r = 28$ and $\beta = 8/3$ then the system is dissipative, has one Lyapunov Exponent greater than zero and is therefore chaotic. This is demonstrated as follows.

Consider the linearized system by the first order expansion of equation (2.2.4.1) as

$$\dot{\mathbf{x}}(t + \delta t) = \mathbf{f}(\mathbf{x}(t + \delta t)) \tag{2.2.4.5}$$

$$\dot{\mathbf{x}}(t) + \frac{d\dot{\mathbf{x}}(t)}{dt}\delta t = \mathbf{f}(\mathbf{x}(t)) + \frac{\partial \mathbf{f}(\mathbf{x}(t))}{\partial \mathbf{x}(t)}\frac{d\mathbf{x}(t)}{dt}\delta t \tag{2.2.4.6}$$

The variational Jacobian is given by

$$\mathbf{J}(\mathbf{x}) = \frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}} = \begin{bmatrix} -\sigma & \sigma & 0 \\ r - x_3 & -1 & -x_1 \\ x_2 & x_1 & -\beta \end{bmatrix} \tag{2.2.4.7}$$

which represents the linear system dynamics at a point in the state space given by the state vector $\mathbf{x}(t)$.

Firstly consider if this system is dissipative, a necessary requirement of a chaotic system. Using the divergence theorem described in section (2.2.1), a volume of the system state space contained within a closed surface *S,* must diminish to zero as time tends towards infinity.
This is determined by

$$\nabla \cdot \mathbf{f}(\mathbf{x}) < 0 \tag{2.2.4.8}$$

For this system

$$\begin{aligned} \nabla \cdot \mathbf{f}(\mathbf{x}) &= \frac{\partial f_1(t)}{\partial x_1(t)} + \frac{\partial f_2(t)}{\partial x_2(t)} + \frac{\partial f_3(t)}{\partial x_3(t)} \\ &= Tr(\mathbf{J}(\mathbf{x})) \\ &= (-\sigma - 1 - \beta) \end{aligned} \tag{2.2.4.9}$$

For the chosen parameters this is negative and so the Lorenz system is dissipative.

Now the Lyapunov Exponents of this system need to be determined. If the evolution of the variational equation is calculated as described in section (2.2.2), and the exponential growth of a small displacement from a given orbit is considered in all three dimensions; then by using the Gram-Schmidt process to re-orthonormalize the evolution matrix at every iteration, the Lyapunov Exponents are approximately evaluated by simulation as
$h_1 = 0.832, \quad h_2 = 0.0$ and $h_3 = -14.504$

A Matlab function to determine these exponents is listed in appendix (A) and a plot of how they evolve is shown below in figure (2.2.4.1)

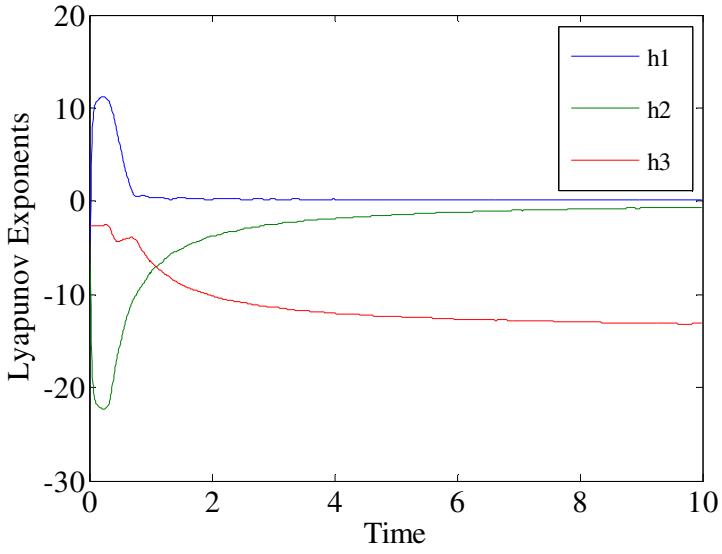


Figure 2.2.4.1

Lyapunov Exponent Evolution Plots

The results show one exponent less than zero, which discursively means that in this dimension the tendency to move away from the trajectory diminishes to zero; one is equal to zero which is the exponent in the direction of the trajectory and as the state progresses along this path, the tendency to diverge from it must be zero; and finally one is greater than zero which implies a chaotic system which, due the other non zero exponent being negative, suggests that the bounded trajectories lie on smooth curved surfaces within the three dimensional state space. This is demonstrated in figures (2.2.4.2) and (2.2.4.3) which show a time progression of the system, with the suggested parameters, and the surface type characteristic.

The Lorenz system described is therefore a good candidate for testing the new communication schemes described in chapter 3.

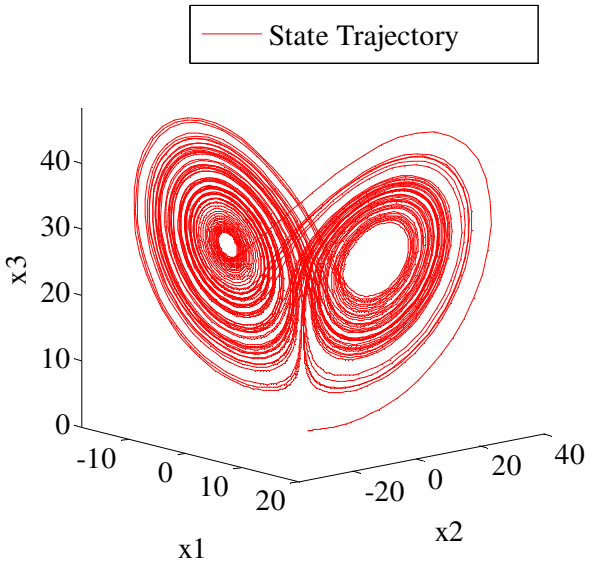


Figure 2.2.4.2

Lorenz System - State Space Trajectory

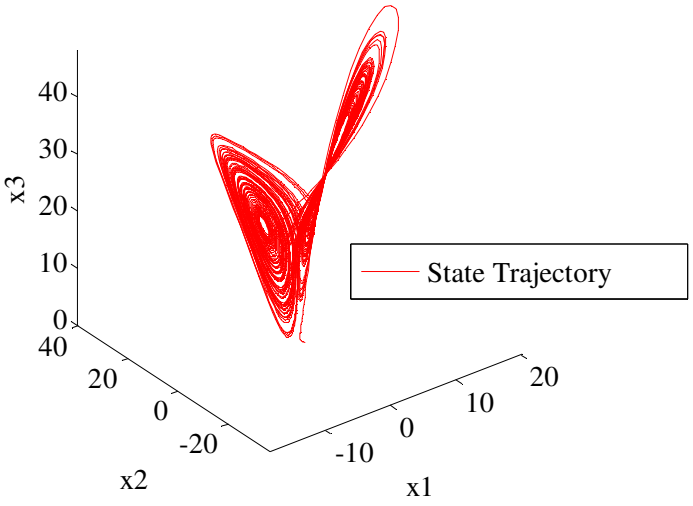


Figure 2.2.4.3

Lorenz System - Surface Type Characteristics

2.2.5 Alternative Systems

The Lorenz system is an example of a chaotic system displaying one degree of chaos; that is, that it has one Lyapunov Exponent greater than zero and thus the chaos has a 'planar' or surface type of habitual behaviour. This surface type of behaviour, in any set of dimensions of a state space system, generated by the system having a Lyapunov dimension of close to two, manifests itself in a periodic like manner. This is clearly illustrated by figure (2.2.1) and it is easily appreciated that this type of periodicity can be easily detected. For this reason, degree one chaotic systems are not the best choice for chaotic secure communication schemes. This periodicity can add further complications for the proposed scheme architecture, which is the main subject of this thesis. Further reference will be made to this in chapter 3. Systems which do not so readily display periodic type behaviour, usually have a degree of chaos of greater than one, and are termed hyperchaotic. Systems that are not truly hyperchaotic, but display hyperchaotic behaviour and good clarity for the understanding of the communications schemes, usually contain non-linearities which are constructed from discontinuities [7] and this is not beneficial to a secure communications structure. All continuous three dimensional systems, which do not contain discontinuities, are of degree one.

2.3 Synchronization Methods

The concept of using synchronization methods in communications schemes is based on the idea that two similar circuits or state space systems, one in the transmitter and the other in the receiver, can have at any particular time, the same dynamical state. The means by which this can be achieved has been the subject of a great deal of research. There are three basic methods of synchronization available. The first is some form of external reference such as an absolute time from a GPS satellite or other reliable source. If the circuit or dynamical system can use this to determine its current state, then synchronization between disparate systems can be achieved. The second would rely on the manufacture of highly accurate components and internal clocks, which would ensure that the error between the two systems would not vary by a given margin, in the time between some form of system coordinating calibration. The final method to achieve synchronization is by communicating, in some way, the state of the transmitter over the transmission channel to the receiver. The problem then becomes one of rejecting the

noise on the transmission channel, separating out the message contained in the signal and ensuring that the signal synchronizes the circuit or state of the dynamical system. The final problem is the one reviewed in this chapter.

2.3.1 Drive Response Synchronization

The work that is normally seen as the initial work on drive response systems is by Pecora and Carroll [4]. They introduce the concept of synchronizing two systems, a response system, using an observed driving signal, and a corresponding driving system producing it. The systems can be seen as receiver and transmitter in a communication scheme. The two systems are the same, and the requirement is to ensure that the state of the response system follows the state of the driving system, after some small synchronizing time despite, the initial conditions. Pecora and Carroll showed that the two systems would indeed synchronize, if the system could be split into two stable subsystems, with the driven subsystem having negative definite conditional Lyapunov Exponents. This implies that the state trajectory error would tend towards zero as time tends towards infinity. In a further paper [5], they discuss extending the linear stability theorem to chaotic systems, and show that the convergence of states is related to the conditional Lyapunov Exponents and the eigenvectors of the Principal Matrix Solution of the chosen subsystem. To illustrate how this method works, consider the chaotic system due to Lorenz [45] as the drive system and how it is split into subsystems for the purpose of synchronization. In turn it will be demonstrated how this method and the synchronization property can be used as a communication scheme.

Consider the Lorenz system equations as the drive system

$$\dot{\mathbf{u}} = \mathbf{f}(\mathbf{u})$$

$$\mathbf{u}^T = [x \; y \; z]$$

$$\mathbf{f}(\mathbf{u}) = \begin{bmatrix} -\sigma x + \sigma y \\ rx - y - xz \\ xy - \beta z \end{bmatrix} \qquad (2.3.1.1)$$

where $\mathbf{u}$ is the state vector and $\mathbf{f}(\mathbf{u})$ is the system equation set.

Now partition the system into two subsystems with a singular state variable v, a partial state vector $\mathbf{w}$ and their associated subsystem equations.

The system equations are

$$\dot{v} = g(v, \mathbf{w})$$
$$\dot{\mathbf{w}} = \mathbf{h}(v, \mathbf{w}) \qquad (2.3.1.2)$$

where the separated states are $v = x$ and $\mathbf{w}^T = [y\ z]$.

The set of equations (2.3.1.2) can be considered as the drive system, and the single state variable will be considered as the observed signal that will be used to synchronize the two systems. Now consider the driven or response set of subsystems where the response system state variables are superscripted as $\mathbf{w}'$

$$\dot{\mathbf{w}}' = \mathbf{h}(v, \mathbf{w}')$$
$$\dot{v}' = g(v', \mathbf{w}') \qquad (2.3.1.3)$$

Where explicitly these can be written as

$$\mathbf{h}(v, \mathbf{w}') = \begin{bmatrix} rv - w_1' - vw_2' \\ vw_1' - \beta w_2' \end{bmatrix}$$
$$g(v', \mathbf{w}') = -\sigma v' + \sigma w_1' \qquad (2.3.1.4)$$

Now look at the variational equation of the first subsystem in equation (2.3.1.3)

$$\Delta\mathbf{w} = \mathbf{w}' - \mathbf{w}$$
$$\Delta\dot{\mathbf{w}} = \frac{\partial \mathbf{h}}{\partial \mathbf{w}} \Delta\mathbf{w} \qquad (2.3.1.5)$$

And $\frac{\partial \mathbf{h}}{\partial \mathbf{w}}$ is given by

$$\frac{\partial \mathbf{h}}{\partial \mathbf{w}} = \begin{bmatrix} -1 & -v \\ v & \beta \end{bmatrix} \qquad (2.3.1.6)$$

The Lyapunov Exponents of equation (2.3.1.5) are termed as conditional Lyapunov Exponents and if they are negative definite, then this is a necessary but not sufficient condition for synchronisation. It does not account for the set of all possible initial conditions and is not defined for initial conditions at infinity. If, however, the system's

trajectory has settled onto the attractor then this condition is sufficient to ensure synchronization.

For the Lorenz system with parameters set to

$\sigma = 10, \quad r = 28, \quad \beta = 8/3$

The conditional Lyapunov Exponents, calculated using the method described in section (2.2.2), are

$h_1 = -1.81$ and $h_2 = -1.86$

This implies that the response subsystem is not chaotic; the response systems partial state vector will synchronize asymptotically to the drive system's partial state vector and consequently the final remaining state can be reconstructed using equation (2.3.1.3). This results in a fully synchronized state vector in the response system. It should be noted that the variational equation of the smaller subsystem, is actually a linear system with an eigenvalue of $-\sigma$, which means that the subsystem is asymptotically stable and ensures synchronization. Clearly the choice of subsystems is not a simple one and the conditional Lyapunov Exponents of some of the possible systems may be unstable. The architecture schematic for this system is shown in figure (2.3.1.1) and the simulation results are shown in figures (2.3.1.2) to (2.3.1.4). Figure (2.3.1.5) shows a Simulink model of the system with the message bearing signal being chosen as the singular state variable, as in the analysis previously carried out. The results show, in figure (2.3.1.2), a simulation run without noise added to the transmission channel. It clearly shows that the response system's states synchronize within a short period. The particular timing is not too important here, since the rates of the system can be easily varied to suit whichever application it is to be used for. However, it is clear that the synchronization time is of the same order as periodic type oscillations of the chaotic system, which implies that synchronization occurs within a small number of orbits of the trajectory around the chaotic attractor. Two further simulation runs are illustrated; the first in figure (2.3.1.3) shows the same system with Gaussian noise of zero mean and unit variance injected; the second shows the system with the same noise injected with a variance of 25. The state errors in the first case are reasonably acceptable and it is clear that the system has indeed synchronized. That synchronization has been achieved, can be judged

qualitatively by the fact that the response system states do follow the drive system's states over the simulation period. However, in the second case the state errors are similar, throughout the simulation, to the initial errors and it is quite unclear that synchronization has been achieved. Again from a qualitative point of view the response system's states do not follow the drive system's states. However this approximate synchronization may be acceptable dependent on the application.

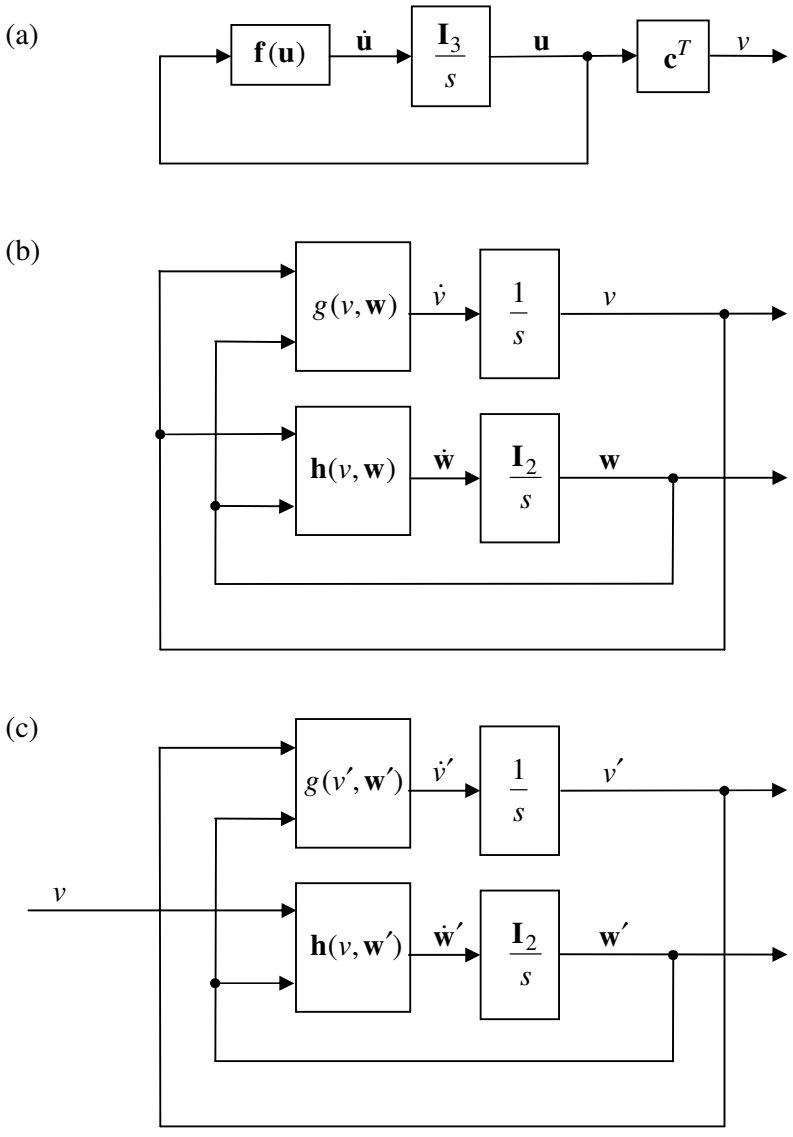


Figure 2.3.1.1

Drive Response System

(a) Full State Vector Drive System, (b) Single State Variable and Partial State Vector Drive System and (c) Single State Variable And Partial State Vector Estimator in the Response System

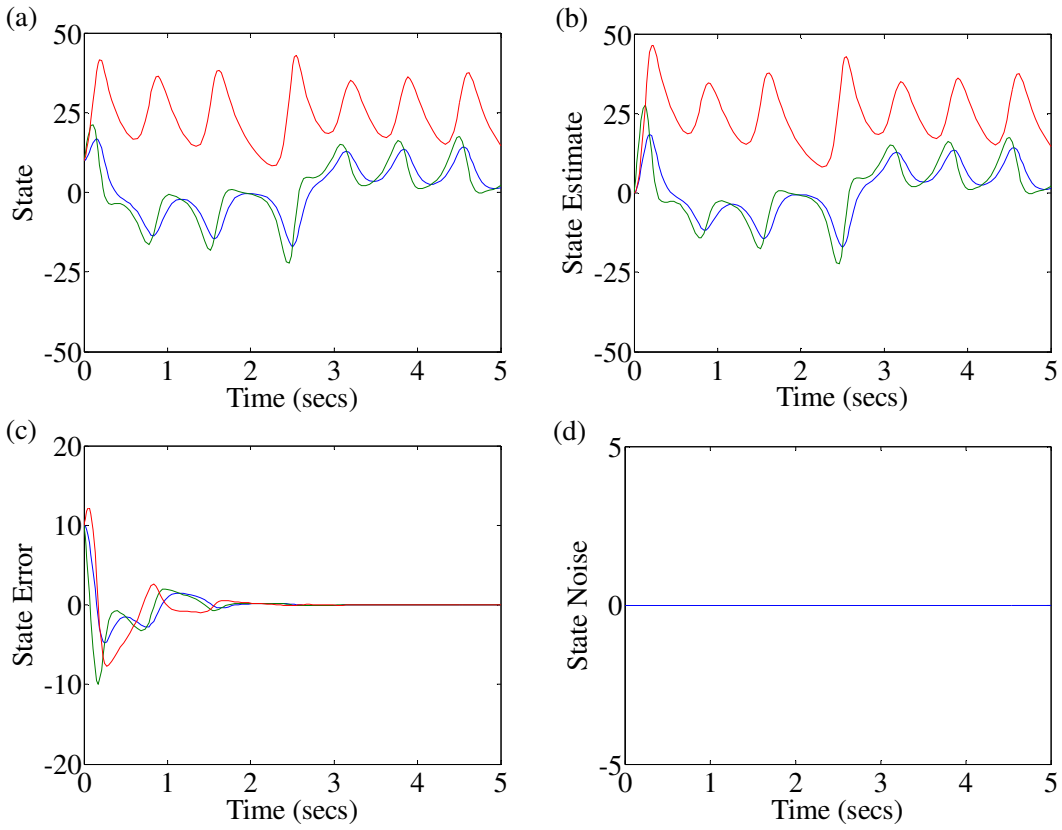


Figure 2.3.1.2

Simulation Results without Noise

(a) Drive System States

(b) Response System States

(c) State Errors

(d) State additive noise

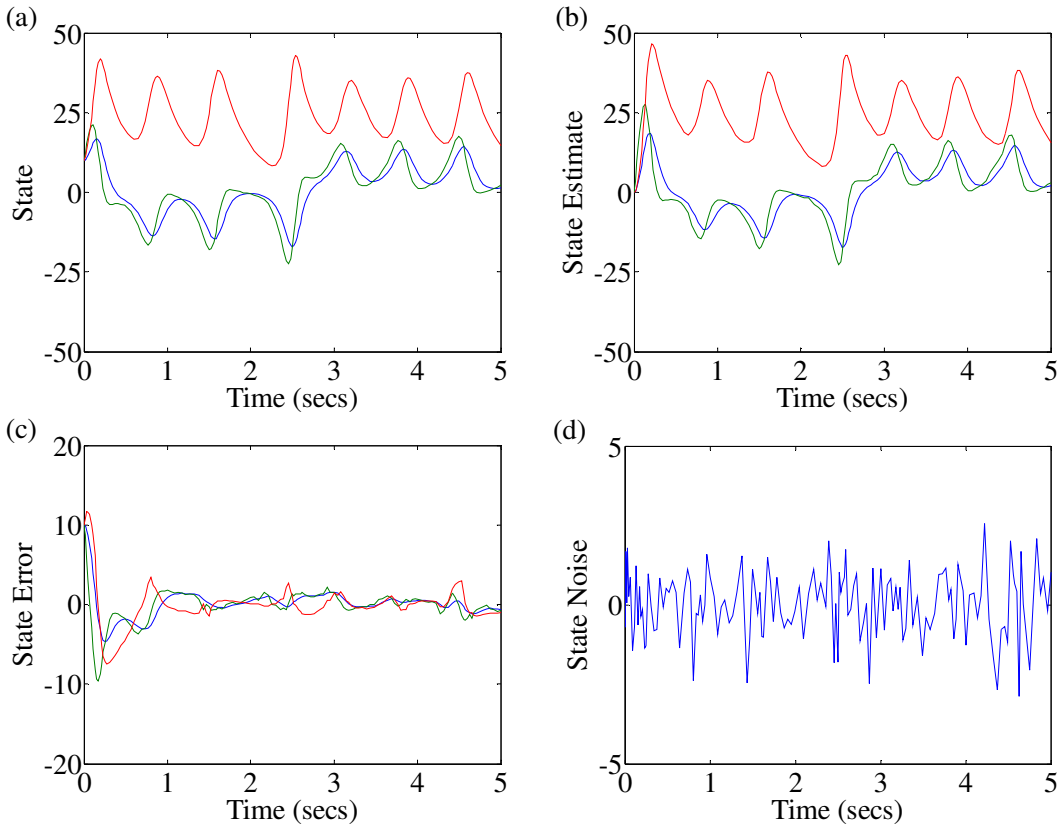


Figure 2.3.1.3

Simulation Results with Gaussian Noise

Mean $\mu = 0$, Variance $\sigma^2 = 1$

(a) Drive System States

(b) Response System States

(c) State Errors

(d) State additive noise

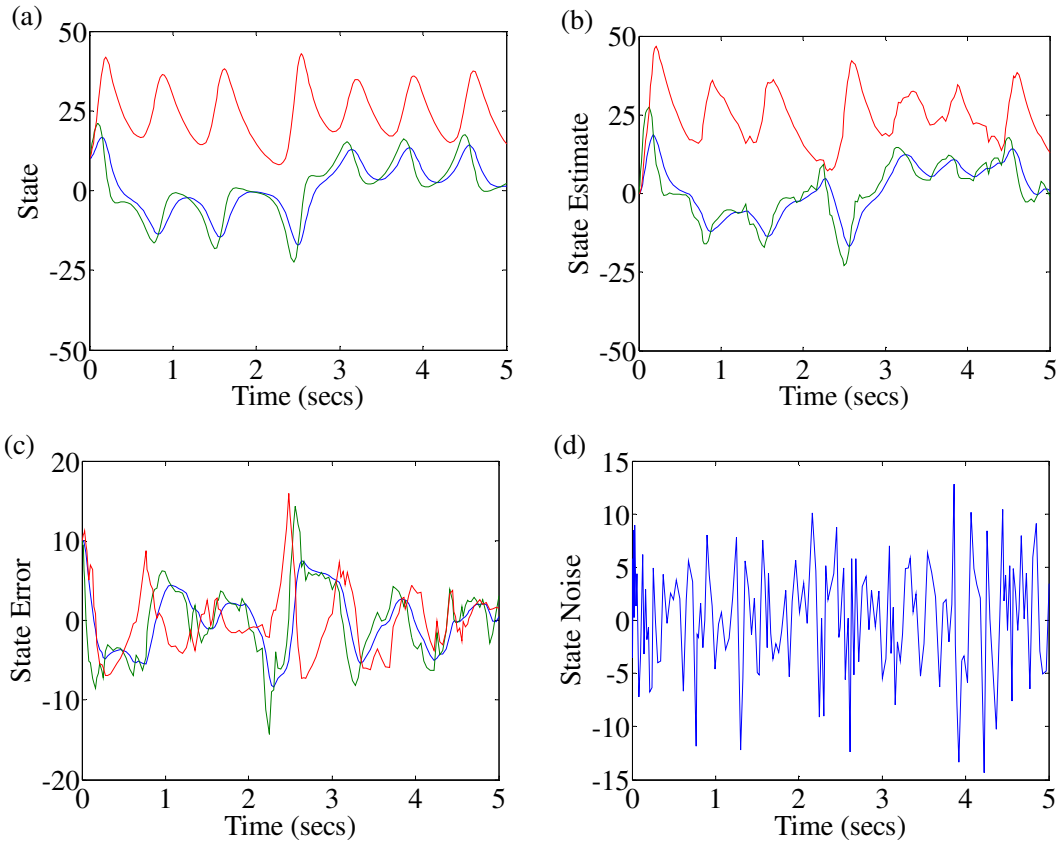


Figure 2.3.1.4

Simulation Results with Gaussian Noise

Mean $\mu = 0$, Variance $\sigma^2 = 25$

(a) Drive System States

(b) Response System States

(c) State Errors

(d) State additive noise

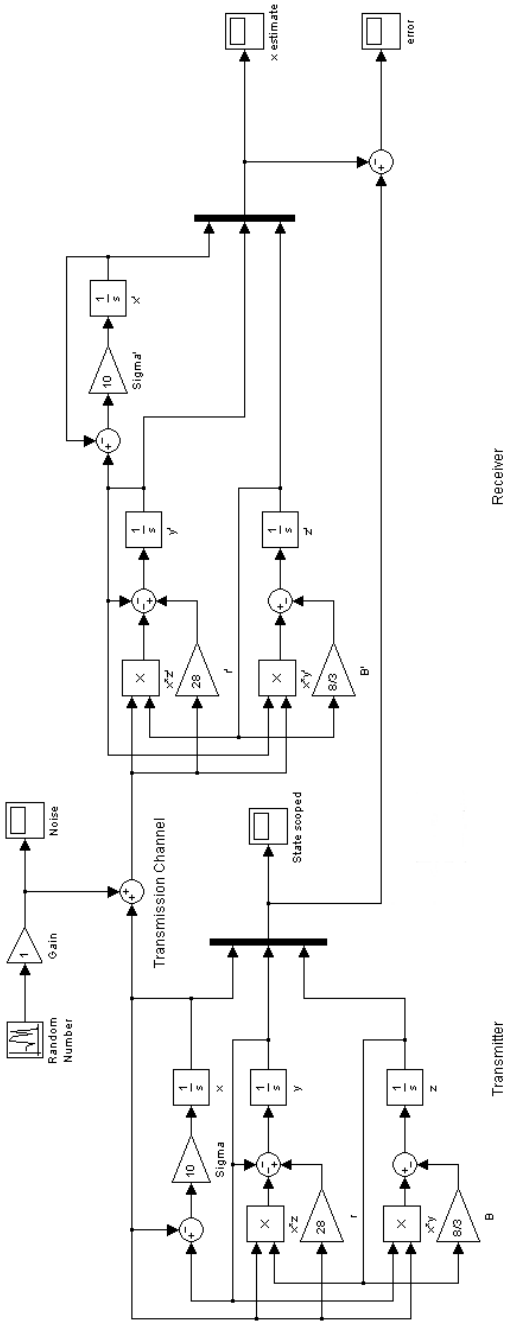


Figure 2.3.1.5

Simulink Model of Synchronization System

2.3.2 Signal Masking

A paper by Cuomo and Oppenheim [6] considered a communication scheme based on the synchronization method. The scheme architecture is shown in figure (2.3.2.1). They considered a scheme in which a Lorenz system was used as the chaotic drive and response system, and injected a real sound signal into the transmission channel. The recovery of the signal, at the response system or receiver, was attempted successfully over short periods of time. Although they showed signal recovery was possible, they did not discuss the continual increase in errors with the increase in time that results from this kind of modulation. This occurs because the state trajectory of the response system becomes considerably different from the drive system's state trajectory and consequently, the local behaviours are quite different, thus they are unable to synchronize. However, if the system is used in a digital way, where the response system attempts to synchronize with the states of the drive system, but can only succeed if the message stream corresponds to a zero input. This input indicates one symbol say a '0'. If the message value is set to the alternate binary value say '1' then the response system cannot properly synchronize and the resultant error squared signal can be used to determine the presence of a transmitted value of '1'. A simulation for a typical message stream and the system's response are shown in figure (2.3.2.2).

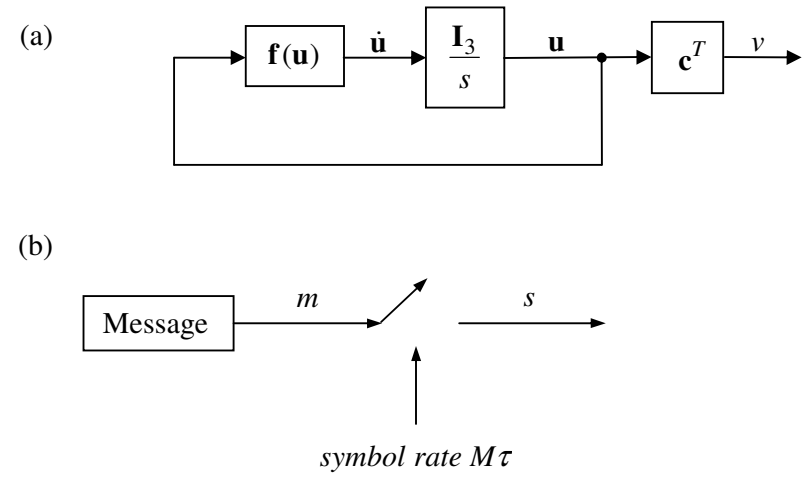


Figure 2.3.2.1

Signal Masking System Architecture

(a) Full State Vector Drive System, (b) Message Sampling,

(c) Single State Variable and Partial State Vector Estimator in the Response System

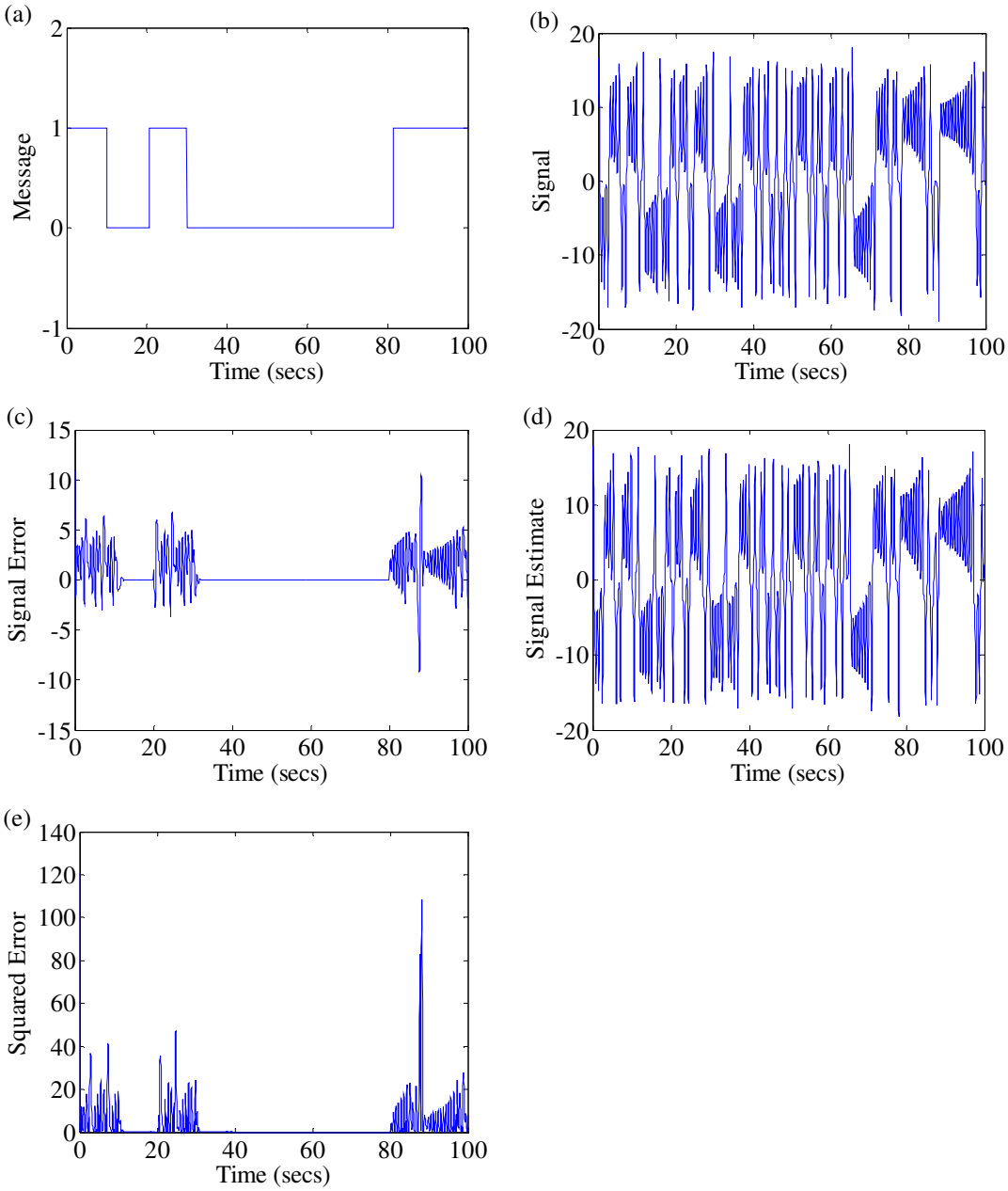


Figure 2.3.2.2

Signal Masking Messages Simulation

(a) Additive Message

(b) Transmitted Signal

(c) Error Between Signal and Signal Estimate

(d) Signal Estimate

(e) Signal Error Squared

2.3.3 Parameter Variation

In this method, the system itself is designed to be used in a digital way, which alleviates the real modulation problems of the masking method. The system architecture is shown in figure (2.3.3.1). The message consists of two symbols representing ‘1’s and ‘0’s which are represented in the drive system by one of the nonlinear parameters in the system equations. For a symbol ‘0’ the value of the parameter is set to one value and for the symbol ‘1’ it is set to another. Again the response system attempts to synchronize to the state trajectory of the drive system, but can only succeed if the drive system’s parameter is the same as its own corresponding value. This indicates the first symbol value has been transmitted and the response system will synchronize. If the message value is set to the alternate binary value, then the response system cannot properly synchronize, and the resultant squared error signal can be used to determine the presence of the other transmitted value. A simulation for a typical message stream and the system’s response are shown in figure (2.3.3.2).

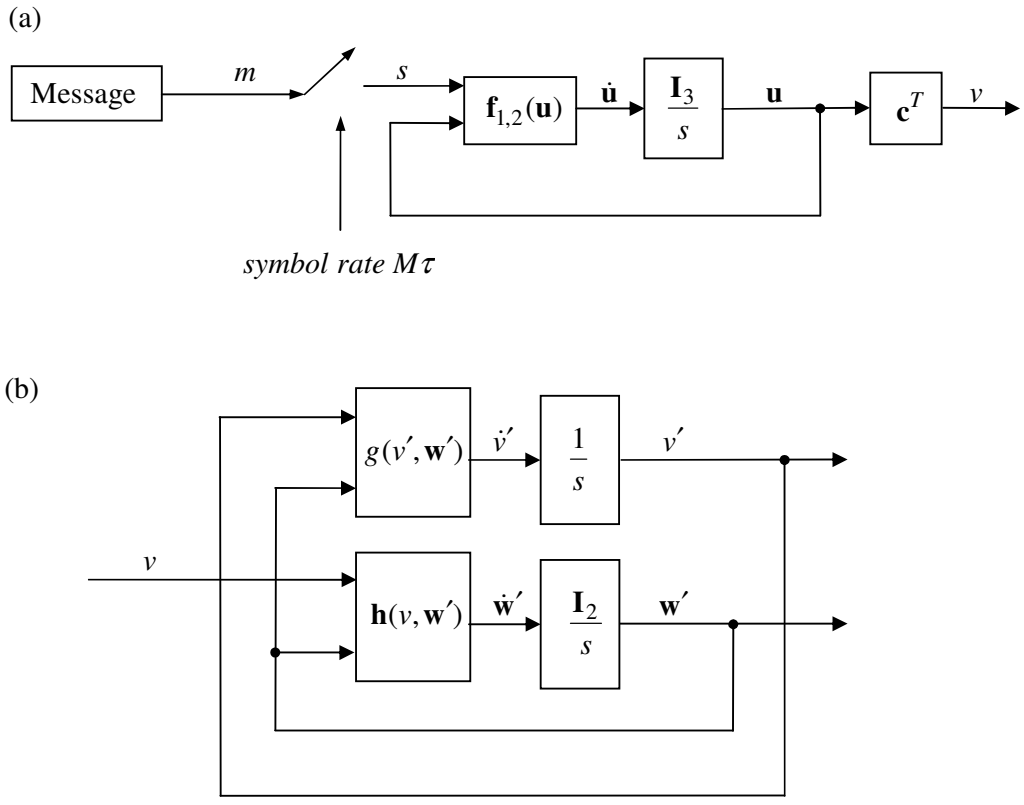


Figure 2.3.3.1

Parameter Variation System

(a) Full State Vector Drive System with Parameter varied by Message

(b) Single State Variable and Partial State Vector Estimates in the Response System

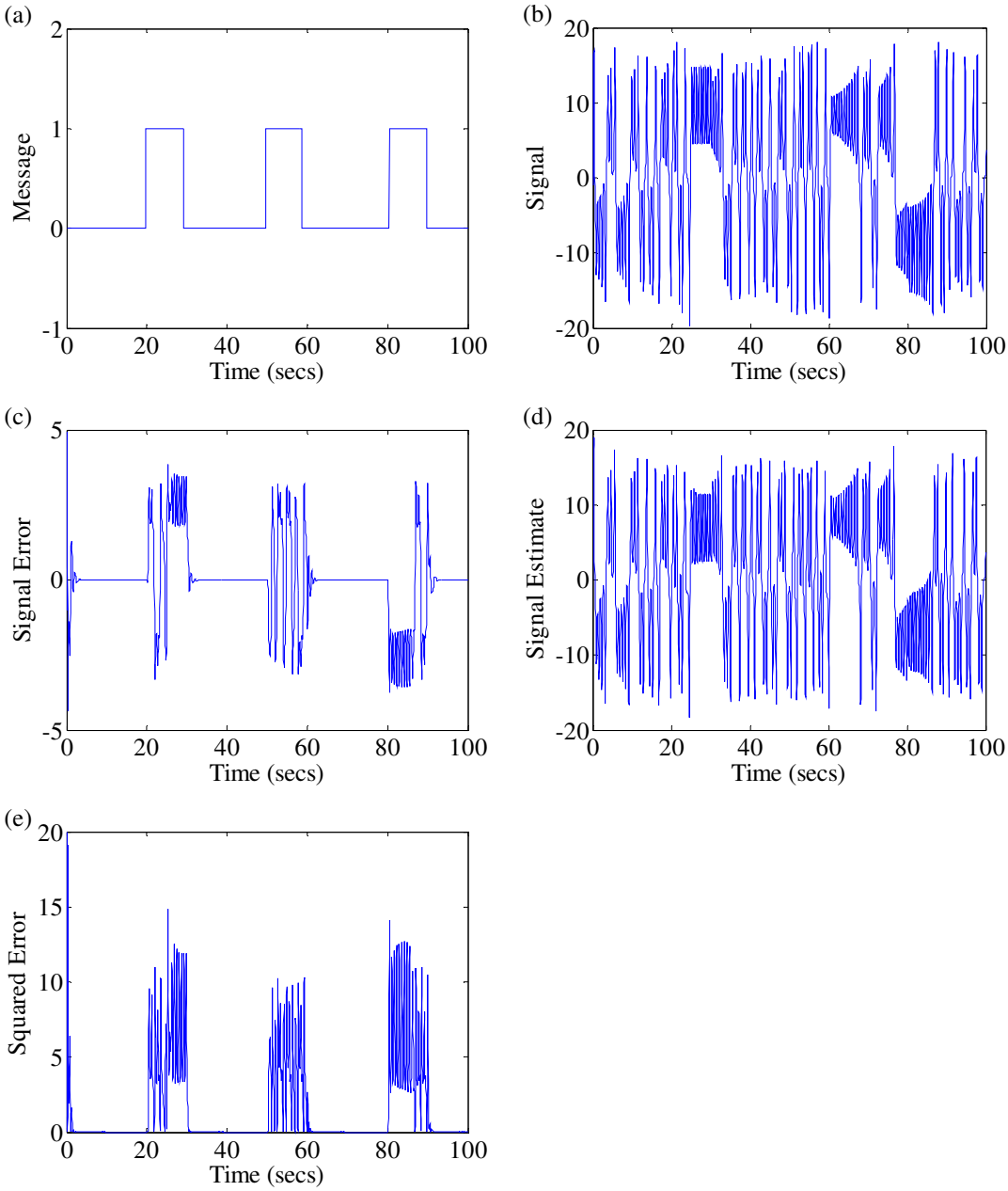


Figure 2.3.3.2

Parameter Variation Messages Simulation

(a) Additive Message

(b) Transmitted Signal

(c) Error Between Signal and Signal Estimate

(d) Signal Estimate

(e) Signal Error Squared

2.3.4 Chaotic Attractor Synchronization

The parameter variation method of section (2.3.3) can only represent two possible symbols in the message set, because the response system can either synchronize or it fails to. This is a simplified version of a method based on a scheme utilizing differing chaotic attractors to represent each different symbol. Instead of having a single response system, which can either synchronize or not, representing a two state message set, there are as many chaotic systems as there are symbols in the message set itself. So the scheme consists of a number of different systems, each representing a different symbol in both the transmitter or drive system side and the receiver or response system. This difference can be as simple as a single parameter in a system or a set of completely different systems. The signal transmitted is the signal from the particular system that represents the message symbol at that sample period and therefore, the only system in the receiver that synchronizes is the one corresponding to the same message symbol. It is important to realize that the chaotic attractor here represents the symbol [13]. Below the system diagram figure (2.3.4.1) represents a system with only two message symbols, and the figures (2.3.4.2-B) (e) and (h) show the squared error terms for each attractor for the corresponding message in figure (2.3.4.1-A)(a)

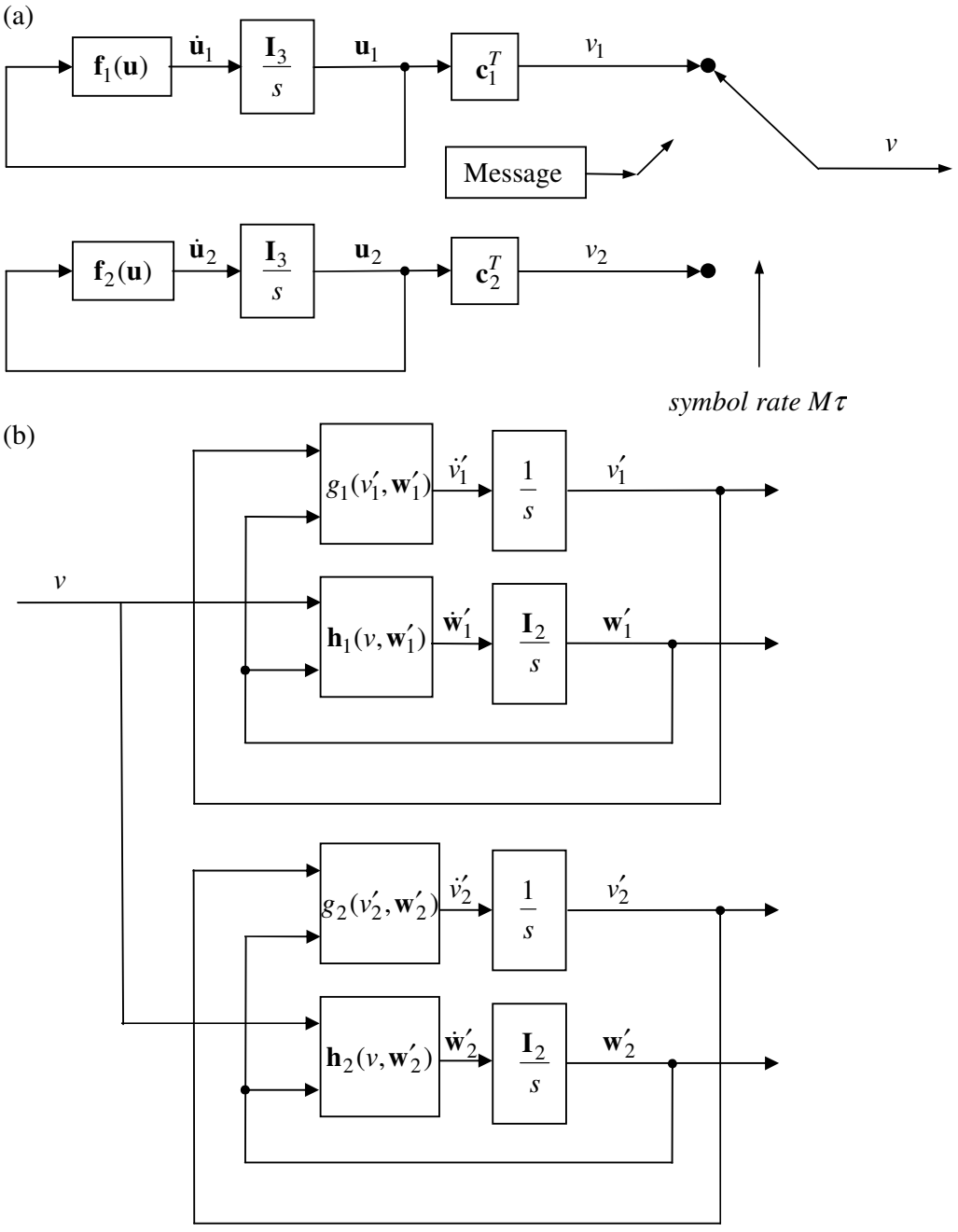


Figure 2.3.4.1

Chaotic Attractor Synchronization System Architecture

(a) Twin Full State Vector Drive Systems with Symbol Selector

(b) Single State Variables and Partial State Vectors Estimators in the Response System

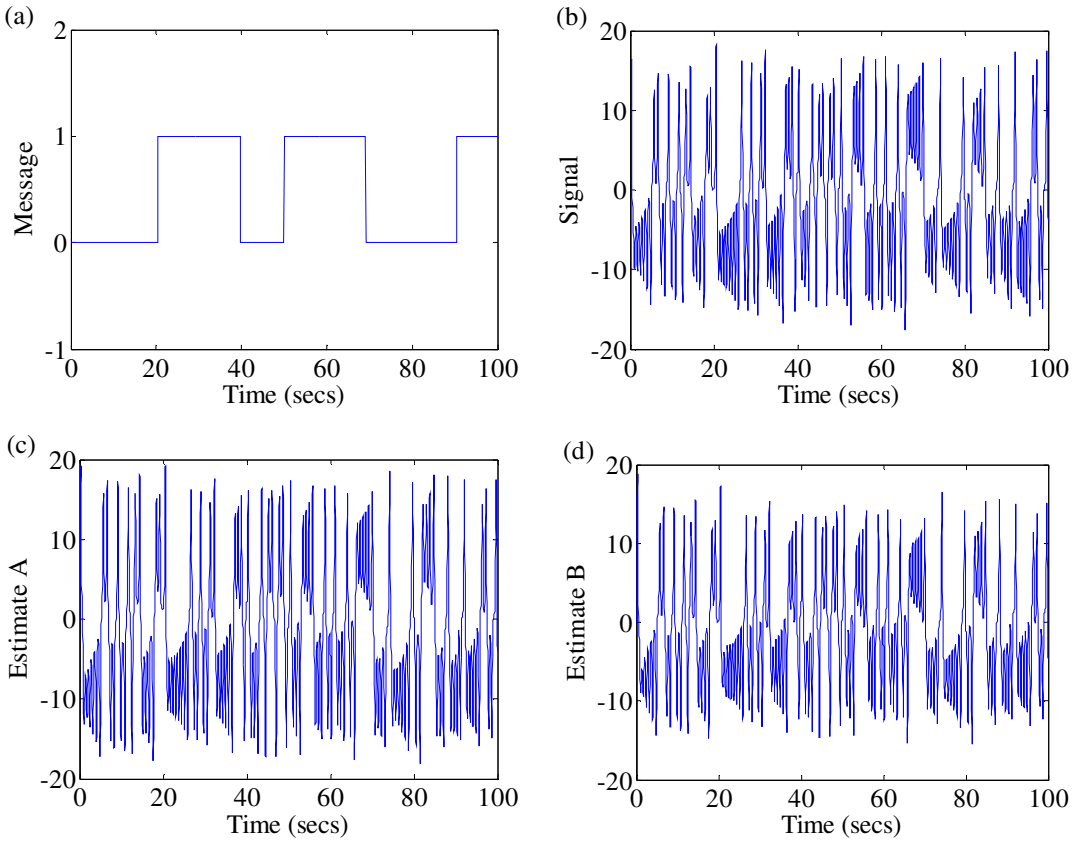


Figure 2.3.4.2-A

Chaotic Attractor Synchronization Messages Simulation

(a) Message, (b) Transmitted Signal, (c) Error Between Signal ‘A’ and Signal ‘A’ Estimate, (d) Signal ‘A’ Estimate.

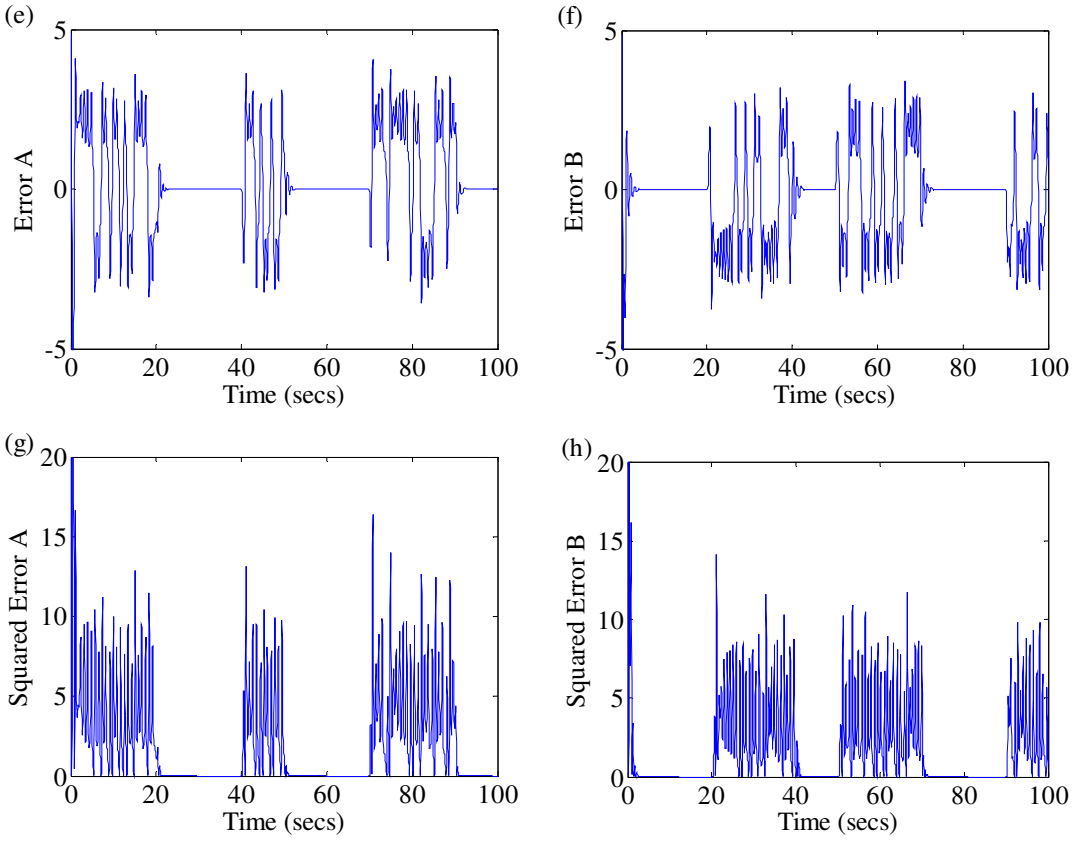


Figure 2.3.4.2-B

Chaotic Attractor Synchronization Messages Simulation

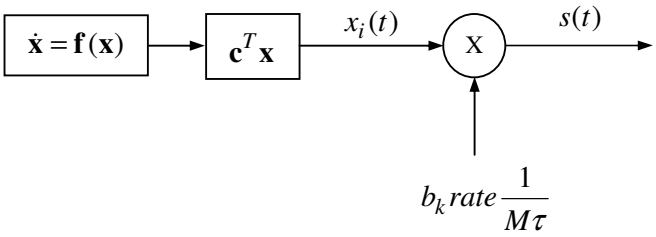
(e) Signal ‘A’ Error Squared, (f) Error Between Signal ‘B’ and Signal ‘B’ Estimate, (g) Signal ‘B’ Estimate and (h) Signal ‘B’ Error Squared

2.4 Non-Reference Correlation Methods

2.4.1 Symmetric Chaos Shift Keying

The Symmetric Chaos Shift Keying (SCSK) method employs the advantages offered by the inherent phase synchronization given by the drive response principle and well understood correlation methods. The carrier is generated by a single or combination of states of a chaotic system, which is then modulated with a message symbol or bit code $b_k = \pm 1$. The operation is illustrated in figure (2.4.1.1)

(a) Transmitter



(b) Receiver

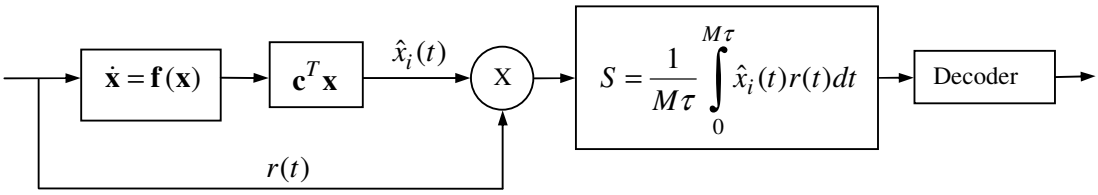


Figure 2.4.1.1
Symmetric Chaos Shift Keying Architecture
(a) SCSK Transmitter and (b) SCSK Receiver

This type of scheme is considered by [13]

Recovery of the transmitted signal is effectively a two-stage process. The receiver contains a similar chaotic generator to the transmitter, which is synchronized by the drive response principles, with the incoming modulated signal. This estimated state signal is then correlated with the received signal to recover the transmitted bit code as outline below. The principle of this method relies on the fact that the system is

symmetrical in one or a combination of the states chosen as the modulation signal. It is supposed here that the conditions of the drive response principle are adhered to, that is that the conditional Lyapunov Exponents of each of the subsystems are not chaotic. This type of technique does require that the nonlinear system is even in the combination of the states chosen to be used as the message bearer. This is necessary to ensure that the driven system in the receiver synchronizes despite the sign of the encoding.

Derivation 2.4.1.1

For the continuous form of SCSK the modulation signal is given as

$$s(t) = \mathbf{c}^T \mathbf{x}(t) \tag{2.4.1.1}$$

And the systems equation as

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}) \tag{2.4.1.2}$$

Consider an example modulation signal

$$s(t) = x_i(t)$$

$$\dot{x}_i(t) = f_i(x_i, \hat{\mathbf{x}})$$

$$\dot{x}_i(t) = f_i(-x_i, \hat{\mathbf{x}}) \tag{2.4.1.3}$$

$\mathbf{f}(\mathbf{x})$ is even and thus symmetric function in the single state $x_i(t)$ and $\hat{\mathbf{x}}(t)$ is the remaining state vector. Now consider the correlation integral over an interval $[0, M\tau]$ where M is the number of samples in the interval

$$S = \frac{1}{M\tau} \int_0^{M\tau} \hat{x}_i(t) r(t) dt \tag{2.4.1.4}$$

Here the state estimate $\hat{x}_i(t)$ is contaminated by Gaussian White noise as is the received message bearing signal $r(t)$ which is modulated with the values $b_k = \pm 1$ these signals are given as

$$\hat{x}_i(t) = x_i(t) + e(t)$$

$$r(t) = b_k x_i(t) + n(t) \tag{2.4.1.5}$$

Substituting equations (2.4.1.5) into (2.4.1.4) yields

$$S = \frac{1}{M\tau} \int_0^{M\tau} \big(x_i(t) + e(t)\big)\big(b_k x_i(t) + n(t)\big) dt \tag{2.4.1.6}$$

This gives

$$S = \frac{1}{M\tau}\int_{0}^{M\tau} b_k {x_i}^2(t)dt + \varepsilon(t) \tag{2.4.1.7}$$

Where the noise signal is elaborated as

$$\varepsilon(t) = \frac{1}{M\tau}\int_{0}^{M\tau} x_i(t)n(t) + b_k e(t)x_i(t) + e(t)n(t)dt \tag{2.4.1.8}$$

The following properties follow from the noise signals being Gaussian White: $\mathrm{E}\{e(t)\}=0$, $\mathrm{E}\{n(t)\}=0$, $\mathrm{E}\{e(t)n(t)\}=0$ and $\mathrm{E}\{z(t)n(t)\}=0$ where $z(t)$ is any other zero mean process then

$$\mathrm{E}\{\varepsilon(t)\}=0 \tag{2.4.1.9}$$

And the integral becomes

$$S = b_k P_x \tag{2.4.1.10}$$

Now as the power of the signal can be generated as

$$P_x = \frac{1}{M\tau}\int_{0}^{M\tau} x^2(t)dt \Leftrightarrow P_x = \mathrm{E}\{x^2(t)\} \tag{2.4.1.11}$$

The original signal can be recovered as

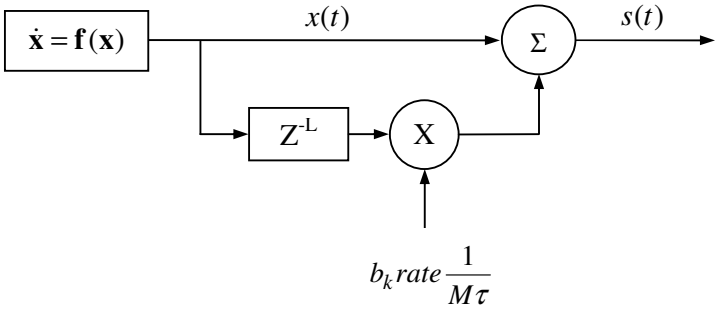
$$b_k = \frac{S}{P_x} \tag{2.4.1.12}$$

The principal disadvantage with this approach, and the derivation of its name, is the requirement that the chaotic process is an even or symmetric function in a single or some combination of states. This requirement comes from the need for the receiver to be able to synchronize with the received signal, regardless of the sign of the bit code. This constraint can prove difficult to achieve, and most systems of this form are implemented using digital forms, that use two-dimensional chaotic maps.

2.4.2 Correlation Delay Shift Keying

Correlation Delay Shift Keying (CDSK) is similar to the SCSK scheme, but transmits a summed signal of a chaotic sequence and a delayed image, in place of either a similar or inverted signal. It has the same advantages as SCSK systems but does not require the system to be symmetric. It does need to transmit a longer signal for the same symbol transmission due to the introduced delay. Again, the first part of the signal is transmitted and then the same message counterpart is transmitted, either in phase or in anti-phase, thus transmitting either '0' of '1' as symbols represented by the multiplier $b_k = \pm 1$ and delayed by a time $L\tau$. The operation is illustrated in figure (2.4.2.1)

(a) Transmitter



(b) Receiver

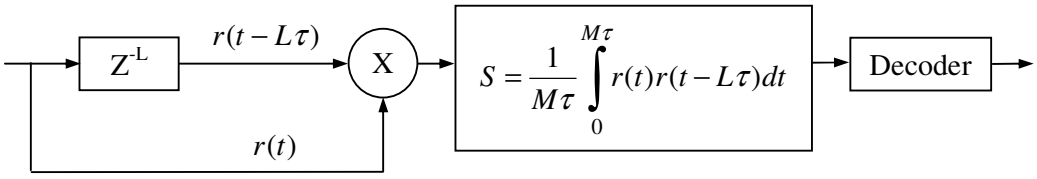


Figure 2.4.2.1
Correlation Delay Shift Keying Architecture
(a) CDSK Transmitter and (b) CDSK Receiver

This type of scheme is considered by [13]

Recovery of signal is achieved by correlating the current signal with itself delayed by $L\tau$. The disadvantage of this approach is that the noise rejection is not as good as DCSK, outlined in section (2.5.1), but it has a greater transmission rate due to not transmitting the reference.

Derivation 2.4.2.1

The continuous form of CDSK

$$s(t) = x(t) + b_k x(t - L\tau) \quad \forall \quad t \in [0, M\tau] \tag{2.4.2.1}$$

$$S = \frac{1}{M\tau} \int_0^{M\tau} r(t - L\tau) r(t) dt \tag{2.4.2.2}$$

$$r(t) = s(t) + n(t)$$

$$r(t - L\tau) = s(t - L\tau) + n(t - L\tau) \tag{2.4.2.3}$$

$$r(t) = x(t) + b_k x(t - L\tau) + n(t)$$

$$r(t - L\tau) = x(t - L\tau) + b_k x(t - 2L\tau) + n(t - L\tau) \tag{2.4.2.4}$$

$$S = \frac{1}{M\tau} \int_0^{M\tau} b_k x^2 (t - L\tau) dt + \varepsilon(t) \tag{2.4.2.5}$$

$$\varepsilon(t) = \frac{1}{M\tau} \int_0^{M\tau} x(t - L\tau) x(t) + b_k x(t - 2L\tau) x(t) + n(t - L\tau) x(t) +$$

$$\ldots b_k^2 x(t - 2L\tau) x(t - L\tau) + b_k n(t - L\tau) x(t - L\tau) +$$

$$\ldots x(t - L\tau) n(t) + b_k x(t - 2L\tau) n(t) + n(t - L\tau) n(t) dt \tag{2.4.2.6}$$

$\mathrm{E}\{n(t)\} = 0$, $\mathrm{E}\{n(t - L\tau) n(t)\} = 0$ and $\mathrm{E}\{z(t) n(t)\} = 0$ where $z(t)$ is any other zero mean process. Additionally if $kL\tau$ is sufficiently large and the signal $x(t)$ is sufficiently varying then $\mathrm{E}\{x(t) x(t - kL\tau)\} = 0 \; : \; k > 0$. This then implies that

$$\mathrm{E}\{\varepsilon(t)\} = 0 \tag{2.4.2.7}$$

And the integral becomes

$$S = b_k P_x \tag{2.4.2.8}$$

Now as the power of the signal can be generated as

$$P_x = \frac{1}{M\tau}\int_0^{M\tau} x^2(t)dt \Leftrightarrow P_x = \mathrm{E}\left\{x^2(t)\right\} \qquad (2.4.2.9)$$

The original signal can be recovered as

$$b_k = \frac{S}{P_x} \qquad (2.4.2.10)$$

2.5 Reference Correlation Methods

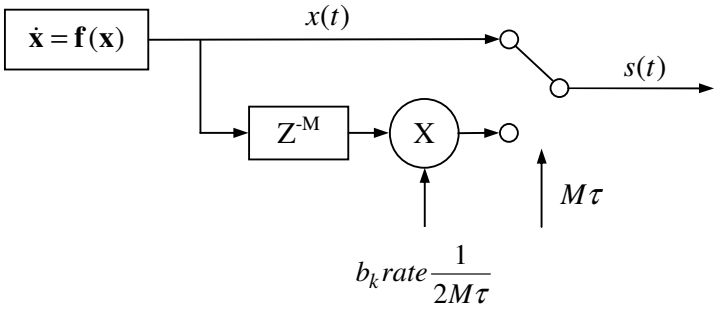
Reference Correlation methods overcome the problems of synchronization methods, delayed correlation methods or systems with special structures, by transmitting for each symbol a single or multiple set of reference signals, followed by a set of encoded signals carrying the message. This removes the time dependency problems, the noise problems related to synchronization and reduces the extensive effects of noise in the transmission channels. The following sections describe some typical methods.

2.5.1 Differential Chaos Shift Keying

This is similar to the well-understood method of Binary Phase Shift Keying (BPSK). In BPSK the underlying message bearer is sinusoidal, and for a portion of the transmitted signal, a reference sine wave is transmitted followed by the symbol carrying part of the signal. To represent a '0' an in phase sine wave is transmitted and its anti-phase counterpart is transmitted to represent a '1'. By correlating the signals at the receiver, the signals can be time synchronized, and decrypted quite simply.

Differential Chaos Shift Keying (DCSK) is exactly analogous to BPSK, but the underlying message carrier is a portion of a signal generated by a chaotic system. Again the first part of the signal is transmitted as a reference, and then its counterpart is transmitted either in phase or in anti-phase, thus transmitting either '0' of '1' as symbols represented by the multiplier $b_k = \pm 1$. The operation is illustrated in figure (2.5.1.1)

(a) Transmitter



(b) Receiver

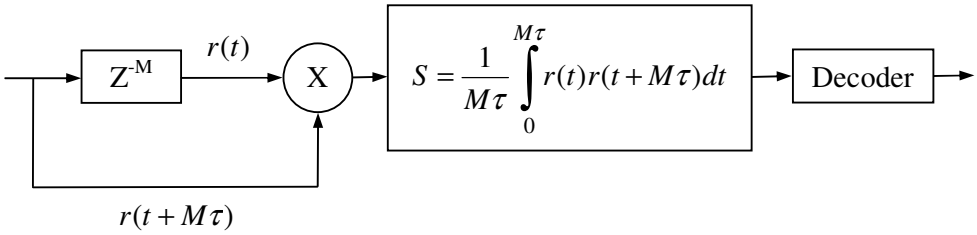


Figure 2.5.1.1
Differential Chaos Shift Keying Architecture
(a) DCSK Transmitter and (b) DCSK Receiver

This type of scheme is considered by [13-19]

Recovery of the message is achieved by correlating the current signal with itself delayed by half of the symbol transmission period $M\tau$. This approach results in the rejection of the noise encountered in the transmission channel and the recovery of the bit code transmitted. The noise is considered to be Gaussian White. This is demonstrated in the following derivation.

Derivation 2.5.1.1

The continuous form of DCSK

$$s(t)=\begin{Bmatrix} x(t) \\ b_k x(t-M\tau) \end{Bmatrix} \quad \forall \quad t \in \begin{Bmatrix} (0,M\tau] \\ (M\tau,2M\tau] \end{Bmatrix} \tag{2.5.1.1}$$

The signal is recovered from the received signal $r(t)$ by correlating it with the received signal delay by $M\tau$. The correlator output is given by

$$S=\frac{1}{M\tau}\int_0^{M\tau} r(t)r(t+M\tau)dt \tag{2.5.1.2}$$

and the signal sets are given by

$$r(t)=s(t)+n(t)$$

$$r(t+M\tau)=s(t+M\tau)+n(t+M\tau) \tag{2.5.1.3}$$

$$r(t)=x(t)+n(t)$$

$$r(t+M\tau)=b_k x(t)+n(t+M\tau) \tag{2.5.1.4}$$

Substituting equations (2.5.1.3-4) into (2.5.1.2) gives

$$S=\frac{1}{M\tau}\int_0^{M\tau} \big(x(t)+n(t)\big)\big(b_k x(t)+n(t+M\tau)\big)dt \tag{2.5.1.5}$$

$$S=\frac{1}{M\tau}\int_0^{M\tau} b_k x^2(t)dt+\varepsilon(t) \tag{2.5.1.6}$$

where the noise term $\varepsilon(t)$ is given by (2.5.1.7) and the expected value of this is zero.

$$\varepsilon(t)=\frac{1}{M\tau}\int_0^{M\tau} b_k n(t)x(t)+x(t)n(t+M\tau)+n(t)n(t+M\tau)dt \tag{2.5.1.7}$$

The following properties follow from the noise signals being Gaussian White $\mathrm{E}\{e(t)\}=0$, $\mathrm{E}\{n(t)\}=0$, $\mathrm{E}\{n(t+M\tau)n(t)\}=0$ and $\mathrm{E}\{z(t)n(t)\}=0$ where $z(t)$ is any other zero mean process then

$$\mathrm{E}\{\varepsilon(t)\}=0 \tag{2.5.1.8}$$

And the integral becomes

$$S=b_k P_x \tag{2.5.1.9}$$

Now the power of the signal can be generated from the reference part of the signal as

$$P_x = \frac{1}{M\tau}\int_0^{M\tau} x^2(t)dt \Leftrightarrow P_x = \mathrm{E}\left\{x^2(t)\right\} \qquad (2.5.1.10)$$

and the original encoding can be recovered as

$$b_k = \frac{S}{P_x} \qquad (2.5.1.11)$$

This approach has good noise rejection [13][41] but is flawed by a need for the receiver to be in phase with the transmitter. The phase locking can be quite complex, as the message bearing signal is a spread spectrum type, which is not fixed at a particular frequency.

2.5.2 FM Differential Chaos Shift Keying

This FM-DCSK scheme [19] extends the ideas of DCSK to a multidimensional and therefore to a multilevel form. It is more complex when compared to the DCSK method, suffers from diminished noise rejection properties, but does introduce the concept of multidimensionality and complete sets of orthogonal of signal sequences which is the basis of this thesis and alluded to in reference [13]. In the same way that QPSK uses orthogonal signals to encode higher dimensional and thus multilevel symbols, by using sine and cosine functions, two further methods find ways of using chaotic signals to the same purpose. One is Quadrature Phase Shift Keying which is fully described in the section (2.5.3) and the other is FM-DCSK. FM-DCSK for a two dimensional case generates two signals which are orthogonal to each other as follows

$$f_1(t) = \begin{cases} +\frac{1}{\sqrt{E_x}}x(t) & 0 \le t < \frac{T}{2} \\ +\frac{1}{\sqrt{E_x}}x(t-\frac{T}{2}) & \frac{T}{2} \le t < T \end{cases}$$

$$f_2(t) = \begin{cases} +\frac{1}{\sqrt{E_x}}x(t) & 0 \le t < \frac{T}{2} \\ -\frac{1}{\sqrt{E_x}}x(t-\frac{T}{2}) & \frac{T}{2} \le t < T \end{cases} \qquad (2.5.2.1)$$

where $x(t)$ is the chaotic signal, E_x is the energy of the signal per bit and T is the bit transmission time.

These signals can now be used in exactly the same way as the sinusoidal functions can be for QPSK, QAM or M-ary type transmission schemes. In fact, they can be readily extended to higher dimensions, by recognising that these signals follow the Walsh function [48] patterns, most commonly used in wavelet analysis of signals, that is

$$W_2 = \begin{bmatrix} +1 & +1 \\ +1 & -1 \end{bmatrix} \qquad (2.5.2.2)$$

Then this concept can be extended to say four dimension where

$$W_4 = \begin{bmatrix} +1 & +1 & +1 & +1 \\ +1 & -1 & +1 & -1 \\ +1 & +1 & -1 & -1 \\ +1 & -1 & -1 & +1 \end{bmatrix} \qquad (2.5.2.3)$$

using this pattern the mutually orthogonal signal sets can be generated as

$$f_1(t) = \begin{cases} +\frac{1}{\sqrt{E_x}}x(t) & 0 \le t < \frac{T}{4} \\ +\frac{1}{\sqrt{E_x}}x(t-\frac{T}{4}) & \frac{T}{4} \le t < \frac{T}{2} \\ +\frac{1}{\sqrt{E_x}}x(t-\frac{T}{2}) & \frac{T}{2} \le t < \frac{3T}{4} \\ +\frac{1}{\sqrt{E_x}}x(t-\frac{3T}{4}) & \frac{3T}{4} \le t < T \end{cases}$$

$$
f_2(t) = \begin{cases} +\frac{1}{\sqrt{E_x}}x(t) & 0 \le t < \frac{T}{4} \\ -\frac{1}{\sqrt{E_x}}x(t-\frac{T}{4}) & \frac{T}{4} \le t < \frac{T}{2} \\ +\frac{1}{\sqrt{E_x}}x(t-\frac{T}{2}) & \frac{T}{2} \le t < \frac{3T}{4} \\ -\frac{1}{\sqrt{E_x}}x(t-\frac{3T}{4}) & \frac{3T}{4} \le t < T \end{cases}
$$

$$
f_3(t) = \begin{cases} +\frac{1}{\sqrt{E_x}}x(t) & 0 \le t < \frac{T}{4} \\ +\frac{1}{\sqrt{E_x}}x(t-\frac{T}{4}) & \frac{T}{4} \le t < \frac{T}{2} \\ -\frac{1}{\sqrt{E_x}}x(t-\frac{T}{2}) & \frac{T}{2} \le t < \frac{3T}{4} \\ -\frac{1}{\sqrt{E_x}}x(t-\frac{3T}{4}) & \frac{3T}{4} \le t < T \end{cases}
$$

$$
f_4(t) = \begin{cases} +\frac{1}{\sqrt{E_x}}x(t) & 0 \le t < \frac{T}{4} \\ -\frac{1}{\sqrt{E_x}}x(t-\frac{T}{4}) & \frac{T}{4} \le t < \frac{T}{2} \\ -\frac{1}{\sqrt{E_x}}x(t-\frac{T}{2}) & \frac{T}{2} \le t < \frac{3T}{4} \\ +\frac{1}{\sqrt{E_x}}x(t-\frac{3T}{4}) & \frac{3T}{4} \le t < T \end{cases}
$$

(2.5.2.4)

A standard segmented decoding method is used to interpret the incoming signals, via correlation integrals over each signal section, and then a decision process assigns the correct symbol to the signal. By using this technique, the multilevel and dimensionality problems have been addressed at the cost of less security, in that, this is a scheme of signal sequences which is easily detectable as combinations of the references are constantly repeated within each symbol frame. The method now described in section (2.5.3) is the first step to a truly multidimensional and multilevel orthogonal chaotic communication scheme.

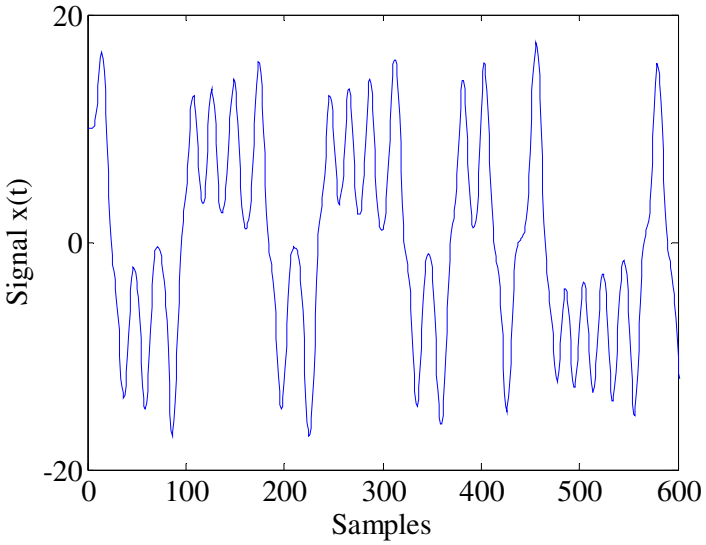
2.5.3 Quadrature Chaos Shift Keying

This scheme is based on a combination of the DCSK referential scheme, and a derivation of the well-known Quadrature Phase Shift Keying (QPSK), which itself is the quadrature form of Binary Phase Shift Keying (BPSK). In both of these methods the underlying message bearer is sinusoidal. For the BPSK technique a portion of the sinusoidal signal is transmitted to represent a '0' and its anti-phase counterpart is transmitted to represent a '1'. QPSK requires two orthogonal signals, which are added together in a combination of four ways to give a four state transmitted signal. The term orthogonal in this sense means, that the integral over a fixed period of the product of two functions, has a mean of zero. That is

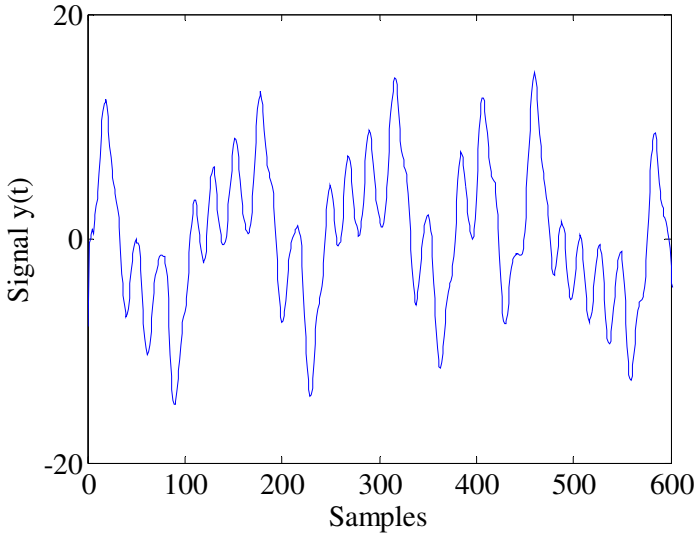
$$\frac{1}{T}\int_{0}^{T} x(t)\,y(t)dt = 0 \qquad (2.5.3.1)$$

In the receiver, the signal parameters are determined by correlating the received signal with each of the orthogonal signals, and hence the exact meaning of the received signal can be deciphered. The signals used in this technique are sinusoidal and the orthogonal counterparts are therefore cosine functions.

In Quadrature Chaotic Shift Keying the sinusoidal signal is replaced by a chaotic reference signal, generated over a fixed time interval, by a chaotic system. A signal that is orthogonal to this is then generated, and these signals are used in a similar way to the QPSK set of orthogonal signals. An example of a set of two orthogonal signals is shown in figure (2.5.3.1) where (a) is the same signal as shown in figure (2.2.1) at the beginning of section (2.2) of this chapter and (b) is its orthogonal counterpart. The orthogonal signal was generated using the algorithm listed in appendix (A.2.1). This achieves the same result as described in derivation (2.5.3.1) below by using a Fast Fourier Transform (FFT) and an Inverse FFT [49]. There are two principal advantages to using chaotic signals. The first is that it allows messages to be transmitted in a secure or covert way, where a potential intruder could easily reject the transmitted signals as noise. Secondly, the signal now has spread spectrum characteristics and the orthogonality of the references, improves the noise rejection properties.



(a)



(b)

Figure 2.5.3.1

QCSK Orthogonal Signal Set

(a) Original Chaotic Signal

(b) Orthogonal Chaotic Counterpart

Derivation 2.5.3.1

Consider a signal on a closed interval $x(t)\ \forall\ t \in [0,T]$, which is generated by a chaotic system and is modified by removing the mean value so that it is a zero mean process, that is

$$\frac{1}{T}\int_0^T x(t)dt = 0 \tag{2.5.3.2}$$

then a Fourier expansion of this signal can be expressed as

$$x(t) = \sum_{m=1}^{\infty} f_m \sin(m\omega t + \phi_m) \tag{2.5.3.3}$$

where $\omega = 2\pi / T$ and $f_0 = 0$

Define the average power of this signal as

$$P_x = \frac{1}{T}\int_0^T x^2(t)dt \tag{2.5.3.4}$$

which because of the following properties of sinusoidal functions

$$\begin{aligned} &\frac{1}{T}\int_0^T f_m \sin(m\omega t + \phi_m - \alpha) f_n \sin(n\omega t + \phi_n - \beta)dt \\ &= \frac{1}{2} f_m^2 \cos(\alpha - \beta) \qquad \forall\ m = n \\ &= 0 \qquad\qquad\qquad\qquad \forall\ m \neq n \end{aligned} \tag{2.5.3.5}$$

can be expressed as

$$P_x = \frac{1}{2}\sum_{m=1}^{\infty} f_m^2 \tag{2.5.3.6}$$

Now to derive a signal that is orthogonal to $x(t)$. Apply a Hilbert Transform to the signal with a phase shift of $\pi/2$. This can be achieved by taking a Fourier Transform of the signal and rotating the positive frequencies by $\pi/2$ and the negative ones by $-\pi/2$. Finally inverting gives the transformed function

$$y(t) = \sum_{m=1}^{\infty} f_m \sin(m\omega t + \phi_m + \frac{\pi}{2}) \tag{2.5.3.7}$$

then

$$x \perp y \Leftrightarrow \frac{1}{T}\int_0^T x(t)y(t)dt = 0 \qquad (2.5.3.8)$$

it follows then that

$$P_x = P_y \Leftrightarrow \frac{1}{T}\int_0^T x^2(t)dt = \frac{1}{T}\int_0^T y^2(t)dt \qquad (2.5.3.9)$$

Consider now two possible maximally separated constellations of signals that consist of an addition of a proportion of each orthogonal signal. These can be represented on an Argand diagram shown in figure (2.5.3.1) and the encoding values set out in table (2.5.3.1)

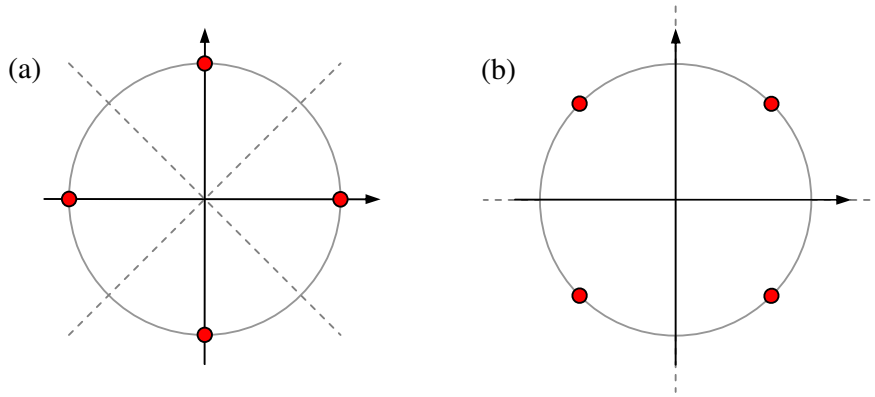


Figure 2.5.3.2

Maximal Separation Quadrature Constellations

Existing on a Two Dimensional Hypersphere:

(a) Symbol encoding contains the reference signal whereas

(b) Encodes all symbols.

Table 2.5.3.1

	Symbol	0	1	2	3
(a)	**c**	1 0	0 1	-1 0	0 -1
(b)	**c**	$1/\sqrt{2}$ $1/\sqrt{2}$	$-1/\sqrt{2}$ $1/\sqrt{2}$	$-1/\sqrt{2}$ $-1/\sqrt{2}$	$1/\sqrt{2}$ $-1/\sqrt{2}$

Maximal separation quadrature constellation encoding symbol maps

Each symbol can be represented by a complex number as

$$c = c_r + jc_i \tag{2.5.3.10}$$

Orthogonality is assured in the complex plane and its analogy can therefore be represented along the real time axis, if the two complimentary signals are orthogonal; that is

$$s(t) = c_r x(t) + c_i y(t) \tag{2.5.3.11}$$

this is the message signal for each symbol in the message.

At the receiver the symbols can be retrieved by determining the coefficients of each individual orthogonal component by using the two correlation integrals

Result 2.5.3.1

$$c_r = \frac{1}{P_x T}\int_0^T s(t)x(t)dt \tag{2.5.3.12}$$

$$c_i = \frac{1}{P_y T}\int_0^T s(t)y(t)dt \tag{2.5.3.13}$$

The derivation of the result is outlined below

Derivation 2.5.3.2

$$\int_0^T s(t)x(t)dt = \int_0^T (c_r x(t) + c_i y(t))x(t)dt$$

$$= \int_0^T c_r x^2(t) + c_i y(t)x(t)dt$$

$$= c_r P_x T \qquad (2.5.3.14)$$

$$\int_0^T s(t)y(t)dt = \int_0^T (c_r x(t) + c_i y(t))y(t)dt$$

$$= \int_0^T c_r x(t)y(t) + c_i y^2(t)x(t)dt$$

$$= c_i P_y T \qquad (2.5.3.15)$$

2.6 Observer Methods

The principal idea behind the Drive Response technique, of section (2.3.1), is that the receiver is driven to reconstruct a subset of the states of a chaotic system that reflects that of the transmitter producing them. The system is then a coherent one, and some form of comparison technique can then detect the signal by synchronization or non-synchronization of a single system as in sections (2.3.2) and (2.3.3); or the synchronization of one system of many as in section (2.3.4). This can really be considered as a class of Observer method, which is well understood for linear systems.

Various other techniques have been proposed [25][36] based on the control systems theory of constructing observers. The basic idea is simple to state. The receiver needs to contain an observer to reconstruct the state of the transmitter within the receiver. In [36] a linear time invariant error mechanism is developed which has stated necessary and sufficient conditions to ensure that the state error vector asymptotically diminishes to

zero. This method proves to be more robust than a simple Drive Response Synchronization technique.

The basic concept behind other forms of observer based methods rely on various forms of the description below. Consider the transmitter system

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{f}(\mathbf{x}(t)) \\ y(t) &= \mathbf{c}^T\mathbf{x} \end{aligned} \tag{2.6.1}$$

where $y(t)$ is a combination of the states of the chaotic system, which is to be transmitted, and will be used to observe and recreate the state vector in the receiver.

Consider now the receiver system

$$\begin{aligned} \hat{\dot{\mathbf{x}}}(t) &= \mathbf{f}(\hat{\mathbf{x}}(t)) + \mathbf{m}(\hat{y}(t) - y(t)) \\ \hat{y}(t) &= \mathbf{c}^T\hat{\mathbf{x}} \end{aligned} \tag{2.6.2}$$

Now define an error vector as

$$\mathbf{e} = \hat{\mathbf{x}} - \mathbf{x} \tag{2.6.3}$$

Then it follows that

$$\dot{\mathbf{e}} = \hat{\dot{\mathbf{x}}} - \dot{\mathbf{x}} \tag{2.6.4}$$

and combining equations (2.7.1) to (2.7.4) the dynamical equation of the error is determined as a function of the observer vector

$$\dot{\mathbf{e}} = \mathbf{f}(\hat{\mathbf{x}}) - \mathbf{f}(\mathbf{x}) + \mathbf{m}\mathbf{c}^T\mathbf{e} \tag{2.6.5}$$

To design an observer vector $\mathbf{m}$ the non-linear element needs to be considered. Firstly the non-linear term of this equation can be linearized at the current state value $\mathbf{x}$ and becomes

$$\begin{aligned} \mathbf{f}(\hat{\mathbf{x}}) - \mathbf{f}(\mathbf{x}) &= \mathbf{f}(\mathbf{x}) + \frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}}\mathbf{e} - \mathbf{f}(\mathbf{x}) \\ &= \frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}}\mathbf{e} \end{aligned} \tag{2.6.6}$$

if equation (2.7.3) is rearranged for the estimate vector as

$$\hat{\mathbf{x}} = \mathbf{x} + \mathbf{e} \tag{2.6.7}$$

and substituting (2.7.6) and (2.7.7) into (2.7.5) yields the following observer dynamical error system

$$\dot{\mathbf{e}} = \left(\frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}} + \mathbf{m}\mathbf{c}^T \right) \mathbf{e} \tag{2.6.8}$$

This is again simply an asymptotic estimator with a non-linear element. Now providing that the **m** vector is chosen to give the observer stable eigenvalues [51-53] with any state **x** within the strange attractor of the chaotic system, then the state of the receiver will synchronize with that of the transmitter. If the real parts of the eigenvalues are set to more negative values than the existing system values, then the response time, noise rejection properties and the robustness of the observer are improved [53].

2.7 Feedback Methods

Feedback control of the state of the receiver is primarily used to synchronize the receiver with the transmitter when they are dissimilar due to modelling inaccuracies or some other consideration such as security. Methods using nonlinear control techniques can be used to make the transmitter and the receiver synchronize, even when dissimilar which has value for improving system robustness. An example of one method is shown here.

Consider the following nonlinear dynamical system that represents the transmitter. The state and transmission signal equations are

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{f}(\mathbf{x}(t)) \\ y(t) &= \mathbf{c}^T \mathbf{x} \end{aligned} \tag{2.7.1}$$

where the $y(t)$ is the transmitted signal. Now represent the receiver as a model of the transmitter, with an integral observer as described in section (2.6), and a model of the dissimilar system with a control vector which is yet to be determined, that is

$$\begin{aligned} \hat{\dot{\mathbf{x}}}(t) &= \mathbf{f}(\hat{\mathbf{x}}(t)) + \mathbf{m}(\hat{y}(t) - y(t)) \\ \hat{y}(t) &= \mathbf{c}^T \hat{\mathbf{x}} \end{aligned} \tag{2.7.2}$$

and

$$\dot{\mathbf{z}}(t) = \mathbf{g}(\mathbf{z}(t)) + \mathbf{u}(t) \tag{2.7.3}$$

where the control is given by the vector $\mathbf{u}(t)$, the estimate of the transmitter state is given by $\hat{\mathbf{x}}(t)$ and the dissimilar system vector function $\mathbf{g}(\mathbf{z})$ has a state $\mathbf{z}(t)$ at time *t*.

The purpose of the control vector $\mathbf{u}(t)$ is to drive the error between the estimate of the transmitter state $\hat{\mathbf{x}}(t)$ and the dissimilar system state $\mathbf{z}(t)$ to zero.

Now define an error vector as

$$\mathbf{e} = \hat{\mathbf{x}} - \mathbf{z} \tag{2.7.4}$$

Then it follows that

$$\dot{\mathbf{e}} = \dot{\hat{\mathbf{x}}} - \dot{\mathbf{z}} \tag{2.7.5}$$

and combining equations (2.7.2) to (2.7.5) the dynamical equation of the error is determined as a function of the control vector

$$\dot{\mathbf{e}} = \mathbf{f}(\hat{\mathbf{x}}) - \mathbf{g}(\mathbf{z}) - \mathbf{u} \tag{2.7.6}$$

now

$$\mathbf{f}(\hat{\mathbf{x}}) = \mathbf{f}(\mathbf{z} + \mathbf{e}) = \mathbf{f}(\mathbf{z}) + \frac{\partial \mathbf{f}(\mathbf{z})}{\partial \mathbf{z}}\mathbf{e} \tag{2.7.7}$$

which implies that

$$\dot{\mathbf{e}} = \boldsymbol{\varepsilon}(\mathbf{z}) + \frac{\partial \mathbf{f}(\mathbf{z})}{\partial \mathbf{z}}\mathbf{e} - \mathbf{u} \tag{2.7.8}$$

where the error between the two systems is given by

$$\boldsymbol{\varepsilon}(\mathbf{z}) = \mathbf{f}(\mathbf{z}) - \mathbf{g}(\mathbf{z}) \tag{2.7.9}$$

The determination of **u** is complex and dependent on the system equations. The control vector **u** is designed to ensure that the Lyapunov stability criteria for the system are assured. That is a Lyapunov error function defined as

$$V(\mathbf{e}) = \tfrac{1}{2}\mathbf{e}^T\mathbf{e} \geq 0 \tag{2.7.10}$$

is positive semi definite and that its first derivative is negative definite, that is

$$\dot{V}(\mathbf{e}) = \dot{\mathbf{e}}^T\mathbf{e} < 0 \tag{2.7.11}$$

The drawbacks of this approach are firstly that the receiver requires an observer to reconstruct the state $\hat{\mathbf{x}}$ of the transmitter and secondly, that the design of the controller cannot be considered in a generic manner but must be carefully tailored to a specific system [28]. The advantages are a robust nature of the synchronization with good noise rejection, especially when the systems are very similar.

2.8 Detectability

Methods of detection of signals transmitted using chaotic processes have been proposed. One method [29] introduced by G. Perez and H. A. Cereira presumes a chaotic transmission stream using the Lorenz attractor. It revisits ideas proposed originally by Lorenz himself. It is possible to construct a one-dimensional return map by considering the local maxima and minima that one of the states returns to. If X_n is the value of $x(t)$ when it reaches its local maxima and Y_m is the value of $x(t)$ when it reaches its local minima, then the return maps of X_{n+1} versus X_n and Y_{m+1} versus Y_m can be constructed. These maps are almost one-dimensional, and from manipulations of these maps, information about the signal being transmitted can be extracted; since the signal is recognised as a particular chaotic transmitter in a certain transmission mode. Other methods that are based on the underlying nature of the attractors in the systems are referred to in [29-33].

The detectability of FM-DCSK type systems is inherent in the way that they repeat the basic chaotic signal in a number of combinations for each frame of the reference and message bearing parts of their transmission. By autocorrelating the incoming signal, over an extended period, the repetitiveness of the signal will become apparent by repeated impulse like forms in the correlation function. This easily reveals the time frame and allows further analysis of the message bearing signal sequences. This is a valid method of increasing the dimensionality, and hence the transmission efficiency of the communication scheme, but it requires further encryption of the message to ensure its security.

2.9 Summary

From the above sections it has been shown that using synchronisation techniques for communications schemes alone is prone to noise rejection problems, and that any system based purely on this form of communication is not usually robust. There is a wealth of further work here which, especially for the covert communications fields, could be exploited. The internal states of the nonlinear systems of both the drive and response systems are possibly capable of conveying multilevel messages with only one physically visible communication signal and this would present a great advantage and is truly worthy of further research. However for the purposes of this thesis the type of scheme chosen will be the reference signal approach which is vastly more robust and more able to handle noisy environments. The aim will be to improve the transmission rate and efficiencies of discrete time multilevel communication schemes, in real time, by exploiting orthogonal signal sets in a number of novel architectures.

Chapter 3

An Orthogonal Chaotic Vector Shift Keying Communication Scheme

3.1 Introduction

In this chapter a number of new multidimensional transmission encoding schemes are described. Firstly, the limitations of two dimensional schemes with increased transmission efficiency are explored in section (3.2) and then the problems of extending the Fourier expansion method, used in QCSK and described in section (2.4.3), to higher dimensions are highlighted in section (3.3). Consequently, the problem is restated in a multidimensional way. The theory of a multidimensional scheme is then derived in section (3.4) and in section (3.5) a simple robust method of obtaining a multidimensional orthogonal signal set is advanced. A number of encoding and decoding schemes are then presented with their system architectures described in section (3.6). And finally, the Bit Error Rate (BER), probability problem and the problems of signal characterization; and the signal to noise ratio effects on the BER, are derived in a new and novel way for each of the encoding schemes described in sections (3.7) to (3.9).

3.2 Limitations of Two Dimensional Schemes with Increased Transmission Efficiency

Firstly, the limitations of Quadrature Chaos Shift Keying (QCSK) are explored. This method can clearly be developed into an M-ary type constellation method, allowing the transmission of more symbols, thus improving the symbol transmission efficiency. This is illustrated below in figure (3.2.1).

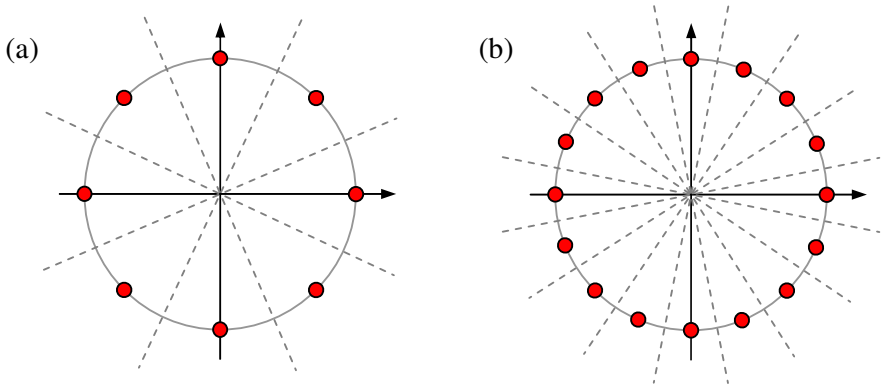


Figure 3.2.1

M-ary Two Dimensional Constellations
(a) '8' Symbol '3' Bit and (b) '16' Symbol '4' Bit Representations

A disadvantage of this method is, that all of the points on the constellation lie on a fixed radius circle that is normally represented on the complex plane. Large numbers of symbols require an equally large number of points on the fixed circle, which becomes crowded, and hence gives rise to potential misinterpretation on decoding; this scheme is therefore not robust. One way to avoid this is to vary both the amplitude of the symbol representations as well as the phase. This is known as Quadrature Amplitude Modulation (QAM) illustrated in figure (3.2.2).

This form of variation of grid type or circle radius type constellation gives rise to signal amplitude variation. This is not desirable, as this would make the signals more easily detectable by varying the power of the transmitted signals. QCSK, QAM and other types of communication constellation scheme are usually represented on the complex

plane. This is largely an historical convenience based on Fourier analysis representation and not a true reflection of the nature of the employed signals. The complex representation is derived from complex analysis, which employs the inherent orthogonality of the sine and cosine functional representation, on the complex plane. If symbols are represented within a higher dimensional space, then the apparent dependence on inherent orthogonality can be discarded; the effective separation between the symbols can be increased and the effects of noise reduced.

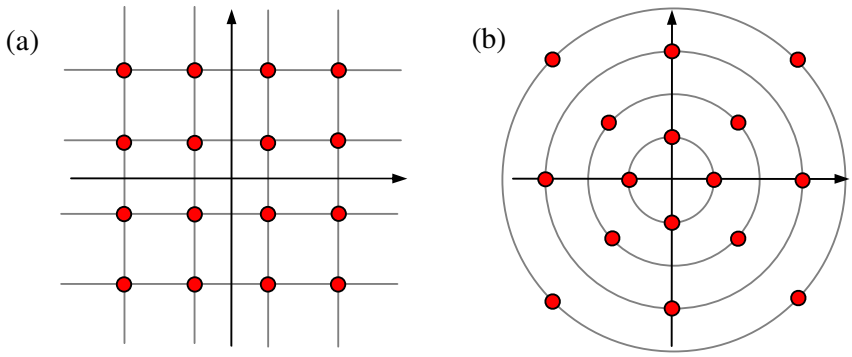


Figure 3.2.2

QAM Type Two Dimensional Constellations

(a) '16' Symbol '4' Bit Grid and (b) '16' Symbol '4' Bit Circular

3.3 Orthogonal Properties of Extended Dimension Fourier Generated Signals

The Fourier Expansion used in QCSK is detailed in Derivation (2.4.3.1), but consider here, the summary for illustration purposes. Consider the chaotic signal $x(t)$ defined on the closed interval $[0,T]$, which has had the mean value removed, and thus can be considered as a zero mean process over the interval.

Now suppose that it will admit to a Fourier expansion of infinite length that is

$$x(t) = \sum_{k=1}^{\infty} f_k \sin(k\omega t + \phi_k) \tag{3.3.1}$$

where $\omega = \dfrac{2\pi}{T}$ and $f_0 = 0$

and the following sinusoidal properties are true

$$I = \frac{1}{T}\int_0^T f_k \sin(k\omega t + \varphi_k - \alpha) f_m \sin(m\omega t + \varphi_m - \beta) dt$$

$$I = 0 \qquad \text{for} \quad k \neq m$$

$$I = \frac{f_k^2}{2}\cos(\beta - \alpha) \qquad \text{for} \quad k = m \tag{3.3.2}$$

For $y(t)$ to be orthogonal to $x(t)$ over the interval then

$$\frac{1}{T}\int_0^T x(t)\, y(t) dt = 0 \tag{3.3.3}$$

Now as $x(t)$ is given as an infinite Fourier Expansion, then there are an infinite number of signals orthogonal to it derived by applying a phase shift of $\pm\frac{\pi}{2}$ to each sinusoidal element of $x(t)$. So

$$y(t) = \sum_{k=1}^{\infty} f_k \sin(k\omega t + \phi_k \pm \frac{\pi}{2}) \tag{3.3.4}$$

In the definition of integral I consider $\alpha = 0$ and $\beta = \pm\frac{\pi}{2}$

then

$$I = \frac{1}{T}\int_0^T f_k \sin(k\omega t + \varphi_k) f_m \sin(m\omega t + \varphi_m \mp \frac{\pi}{2}) dt$$

$$I = 0 \qquad \text{for} \quad k \neq m$$

$$I = \frac{f_k^2}{2}\cos(\pm\frac{\pi}{2}) = 0 \qquad \text{for} \quad k = m \tag{3.3.5}$$

So $I = 0$ in all cases and therefore $y(t)$ is orthogonal to $x(t)$.

Consider now an approximation to $x(t)$ derived from a limited sum of q sinusoidal elements, that is

$$x(t) = \sum_{k=1}^{q} f_k \sin(k\omega t + \phi_k) \tag{3.3.6}$$

It follows that there are 2^q signals orthogonal to the $x(t)$ approximation expressed as follows

$$y_p(t) = \sum_{k=1}^{q} f_k \sin(k\omega t + \phi_k + F(p,k)\frac{\pi}{2}) \tag{3.3.7}$$

where $p \in [1,2^q]$ and $F(p,k)$ is a notional function of p and k that takes the values ± 1 and varies as a Gray scale from $y_p(t) \to y_{p+1}(t)$ so that only one phase change takes place at each step in the entire expansion.

Consider then the question: Are the $y_p(t) \quad \forall \quad p \in [1,2^q]$ signals generated from $x(t)$ mutually orthogonal. To show this is not so, consider any two derived signals $y_i(t)$ and $y_j(t)$ so

$$I = \frac{1}{T}\int_0^T y_i(t)\, y_j(t) dt \tag{3.3.7}$$

$$I = \frac{1}{T}\int_0^T \sum_{k=1}^{q} f_k \sin(k\omega t + \varphi_k + F(i,k)\frac{\pi}{2}) f_k \sin(k\omega t + \varphi_k + F(j,k)\frac{\pi}{2}) dt \tag{3.3.8}$$

Since all other terms vanish then

$$I = \sum_{k=1}^{q} \frac{1}{2} f_k^2 \cos\left(\left(F(i,k) - F(j,k)\right)\frac{\pi}{2} \right) \tag{3.3.9}$$

$$I = \sum_{k=1}^{q} \frac{1}{2} f_k^2 G(k) \tag{3.3.10}$$

where $G(k) = \pm 1$

Clearly, it is possible to find a limited set of mutually orthogonal functions, but it is dependent on the values of each f_k and as each signal set is different being generated from a chaotic sequence; it is not trivial to find a set $f_k \ \forall \ k \in [1, q]$ which drives equation (3.3.10) to zero. So, generally it is true, that all the $y_k(t)$ signals are not mutually orthogonal.

Therefore, it can be concluded that the Fourier expansion method of orthogonal signal generation is not suitable for dimensions greater than m=2.

If it is feasible to separate the problem from complex analysis, the problem can be restated as finding sets of mutually orthogonal signals. The combination of these signals can, in a similar way to QCSK, be extended to much greater information capacity and hence transmission efficiency. QCSK introduces this idea, but is immediately constrained, by the use of the Fourier expansion and Hilbert Transform methods.

Presented is a method of overcoming this problem, by the use of a system of orthogonal sequences, derived by using the Gram-Schmidt orthonormalization method.

3.4 Theoretical Analysis

In the QCSK approach, the key requirement is the generation of a signal orthogonal to the chaotic reference signal. This requirement is met by assuming that the reference signal $x(t)$ over the interval $t \in [0,T]$ has zero mean and will admit to a Fourier expansion which, as shown in section (2.4.3), results in a possible set of orthogonal signals $y(t)$. Once a set of two basis signals is generated, a variety of signalling schemes can be created based on constellations formed on the complex plane, see figure (2.4.3.1). Specifically each symbol is represented as table (2.4.3.1)

$$c = c_r + jc_i \tag{3.4.1}$$

and associated with it is the symbolic modulated signal

$$s(t) = c_r x(t) + c_i y(t) \tag{3.4.2}$$

Recovery of these signals in the receiver is achieved using the two correlation integrals

$$c_r = \frac{1}{P_x T}\int_0^T s(t)x(t)dt \tag{3.4.3}$$

$$c_i = \frac{1}{P_x T}\int_0^T s(t)y(t)dt \tag{3.4.4}$$

Where

$$P_x = \frac{1}{T}\int_0^T x^2(t)dt = \frac{1}{T}\int_0^T y^2(t)dt = P_y$$

The derivation of which is shown in result (2.4.3.1). The number of symbols can be increased into M-ary constellations, with varying amplitudes and phases in the complex plane, as in figures (3.2.1) and (3.2.2) but this inherent two dimensionality of the QCSK approach decreases its potential noise rejection properties when large symbol constellations are considered.

Consider now a system, with an m dimensional constellation, that relies on m different mutually orthogonal signals, which actually form an orthonormal basis of functions $u_i(t)\ \forall\ i \in [1,m]$.

Derivation 3.4.1

As with the QCSK scheme, the message can be encoded using these orthogonal functions by combining them linearly using the value of the encoding coefficients. This can now be represented as

$$s(t) = c_1 u_1(t) + c_2 u_2(t) + c_3 u_3(t) + \ldots + c_m u_m(t) \tag{3.4.5}$$

which in vector function notation becomes

$$s(t) = \mathbf{u}^T(t)\mathbf{c} \tag{3.4.6}$$

where

$$\mathbf{u}^T(t) = [u_1(t), u_2(t), u_3(t), \ldots, u_m(t)] \tag{3.4.7}$$

and

$$\mathbf{c}^T = [c_1, c_2, \ldots, c_m] \tag{3.4.8}$$

this is the message signal for each symbol in our message.

At the receiver the symbols can be retrieved by determining the coefficients of individual orthogonal components by using the m correlation integrals

$$c_i = \frac{1}{P_i T}\int_0^T s(t)u_i(t)dt \quad \forall \quad i \in [1,m] \tag{3.4.9}$$

$$P_i = \frac{1}{T}\int_0^T u_i^2(t)dt \tag{3.4.10}$$

or from (3.4.6) to (3.4.8)

$$\int_0^T \mathbf{u}(t)s(t)dt = \int_0^T \mathbf{u}(t)\mathbf{u}^T(t)\mathbf{c}dt \tag{3.4.11}$$

Therefore this can be written in concise vector notation as

$$\mathbf{c} = \left[\int_0^T \mathbf{u}(t)\mathbf{u}^T(t)dt\right]^{-1}\int_0^T \mathbf{u}(t)s(t)dt \tag{3.4.12}$$

This will work with any set of signals if they are independent. If there is no noise present, the signal sets are orthogonal and the inversion is simplified by the matrix being diagonal. However in the presence of noise, the inversion can influence the noise rejection characteristics of decoding. Noise rejection can be improved by discarding non diagonal terms because they are perceived to have been generated by noise. The scheme of signal transmission here is irrelevant to the above derivation. The signals can be transmitted simultaneously on multiple channels or contiguously on one channel.

3.5 Generation of Orthogonal Signal Sets

The generation of a set of m orthogonal signals is required; we can approach this problem by first considering an n dimensional space. Any point $\mathbf{p}$ can be represented by an n dimensional vector that is a linear sum of the set of orthonormal basis vectors $\mathbf{u}_i$ where $\mathbf{u}_i \in R^n$ and $i \in [1,n]$.

Therefore we can write

$$\mathbf{p} = p_1\mathbf{u}_1 + p_2\mathbf{u}_2 + p_3\mathbf{u}_3 + ... + p_n\mathbf{u}_n \tag{3.5.1}$$

Where $p_i \ \forall \ i \in [1,n]$ are real coefficients.

Now consider a subset of size m of these basis vectors that describe an m dimensional subspace within the n dimensional space. Further consider, the set of vectors describing some hypersurface $\mathbf{s}$ within this m dimensional subspace.

$$\mathbf{s} = c_1\mathbf{u}_1 + c_2\mathbf{u}_2 + c_3\mathbf{u}_3 + ... + c_m\mathbf{u}_m \tag{3.5.2}$$

Here $\mathbf{p}$ and $\mathbf{s} \in R^n$ and $m \le n$. The size of n is explored in chapter 4 but is an order of magnitude greater in size than m for noise rejection purposes. This can be seen as analogous to equation (3.4.5) in section (3.4), excepting that the summation is in terms of real vectors, and not real functions of t. The real vectors $\mathbf{u}_i$ can be obtained from the real functions $u_i(t)$ by the following process. That is $\mathbf{u}_i = [u_{1i} \cdots u_{ni}]^T$ and $u_{ji} = u_i((j-1)\tau + t_i)$ where $j \in [1,n]$, $i \in [1,m]$, t_i is the initial time at the point of sampling each vector and τ is the sampling period. These can be arranged into a matrix $\mathbf{U} = [\mathbf{u}_1 \cdots \mathbf{u}_m]$ where $\mathbf{U} \in R^{n \times m}$. For a practical consideration any real function $u_i(t)$ can generate a vector of signal values $\mathbf{u}_i$ by sampling. Conversely, the real signal

functions $u_i(t)$ can be recreated in a DSP chip electronically. So the problem is reduced to generating a set of real valued orthogonal vectors $\mathbf{u}_i \in R^n$ and $i \in [1,m]$ in order to be able to generate a set of real time orthogonal functions given on an interval $u_i(t) \; \forall \; i \in [1,m] \quad t \in [0, n\tau]$.

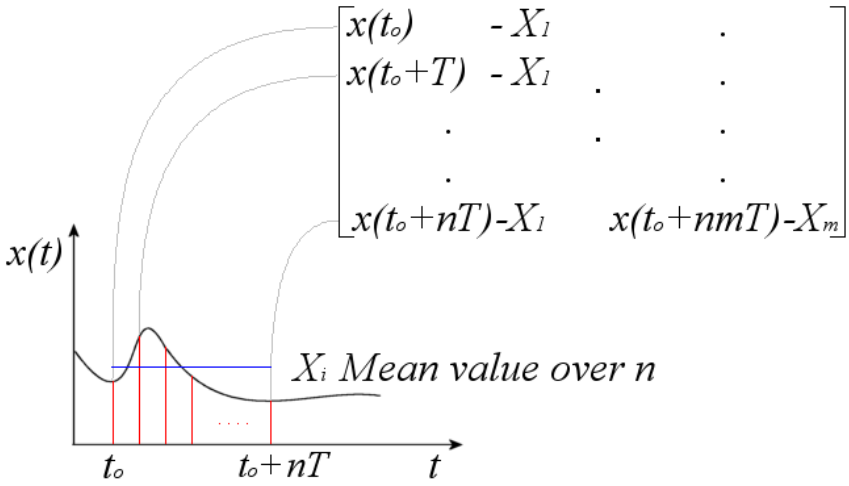


Figure 3.5.1
Signal Sampling to Matrix Concept

Consider a chaotic signal $x(t)$ sampled every τ seconds and the values placed into the columns of a matrix $\mathbf{X}$. So $\mathbf{X} = [\mathbf{x}_1 \cdots \mathbf{x}_m]$ where $\mathbf{X} \in R^{nxm}$ and $\mathbf{x}_i \in R^n$ and in turn each $x_{ji} = x(((i-1)n + (j-1))\tau + t_0) \quad \forall \quad i \in [1,m],\, j \in [1,n]$ and t_0 is the initial sampling time. This is shown in figure (3.5.1). If the chaotic sequence is sufficiently varying then the $rank(\mathbf{X}) = m$ and the vectors that make up the matrix $\mathbf{X}$ will span an m dimensional subspace of the n space potentially spanned by a complete set of n vectors. A matrix $\mathbf{U}$ can be formed which is an orthonormal basis of $\mathbf{X}$ by using the Gram-Schmidt algorithm described in appendix (B). The result of the transformation is a simple linear transformation of $\mathbf{U}$
by an upper triangular matrix $\mathbf{W}$. That is

$$\mathbf{X} = \mathbf{UW} \tag{3.5.3}$$

where

$$\mathbf{U}^T\mathbf{U} = \mathbf{I}_m \tag{3.5.4}$$

and **W** can be found as

$$\mathbf{U}^T\mathbf{U}\mathbf{W} = \mathbf{U}^T\mathbf{X} \tag{3.5.5}$$

So given equation (3.5.4) it follows that

$$\mathbf{W} = \mathbf{U}^T\mathbf{X} \tag{3.5.6}$$

However, finding this transformation is unnecessary if the Gram-Schmidt algorithm is used; it is specified here because it will be needed for signal characterization in section (3.8).

Consider now the independence of the columns of $\mathbf{X}$, and how this relates to the potential number of bits of precision that the signal set values may be in error. In order to determine if the signal set is 'good' enough for transmission, an estimate of the 'bits' in precision error (B_e) can be determined from the matrix 2-norm condition number C_n as

$$B_e \propto \log_2(C_n) \tag{3.5.7}$$

where the condition number is given as

$$C_n = \max\left(\sqrt{\lambda}_i\right) / \min\left(\sqrt{\lambda}_i\right) \quad \forall \quad i \in [i, m] \tag{3.5.8}$$

here the λ_i represent the eigenvalues of the symmetric matrix $\mathbf{X}^T\mathbf{X}$ and $\sqrt{\lambda_i}$ are the singular values of **X.** If B_e is larger than the bit precision of the signal set values that are to be transmitted the matrix, and hence the signal set, can be rejected. With the condition number sufficiently small the **U** matrix can be separated into a set of vectors $\mathbf{u}_i \; \forall \; i \in [i, m]$, which can be seen as samples of a set of continuous signals with zero mean over the time interval $t \in [0, n\tau]$ and average powers of $1/n$. These can now be encoded according to an encoding scheme, power balanced and transmitted. Power balancing here means that each signal sequence has the same average power. This is achieved by post multiplying the signal relevant matrix by an appropriate diagonal square matrix **P**. This is discussed in chapter 4.

3.6 System Architectures and Encoding and Decoding Schemes

In this section there are three different system architectures presented, with their associated encoding and decoding schemes. The first two can be considered as '*m* symbol' type schemes, where a series of *m* symbols are transmitted for each set of derived orthogonal signals. The first scheme exhibits extremely good noise rejection without any real dependence on the nature of the chaotic process chosen. Whereas, the noise rejection of the second scheme has a greater dependence on the type of chaotic process. It is presented only as a precursor to understanding the signal characterization described in section (3.8) and as an introduction to the third transmission scheme. The final scheme has the same inherent requirement as the second scheme, namely to choose the chaotic process carefully, but it is a novel persistent updating method giving rise to much greater transmission efficiency.

3.6.1 Direct '*m* Symbol' 'U' Scheme

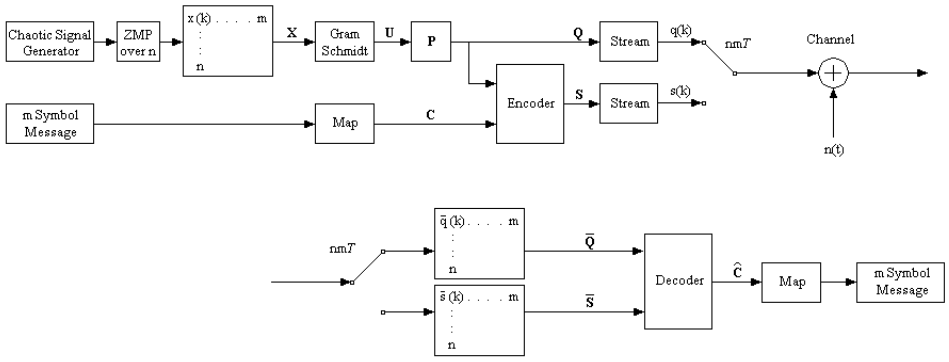


Figure 3.6.1.1

'U' Scheme - Direct '*m* Symbol' Transmission System Architecture

Consider the $n \times m$ signal matrix $\mathbf{X}$ produced by taking *nm* samples of the chaotic process. Where each set of *n* samples has the mean value removed thus leaving each vector of the matrix **X** as samples of a zero mean process. Now an $n \times m$ orthonormal matrix $\mathbf{U}$ is generated from it using the Gram-Schmidt process detailed in appendix (B). The matrix **U** is multiplied by a diagonal power balancing matrix **P** to produce a matrix **Q** as in equation (3.6.1.1) before being streamed. Streaming is transmitting each element of each column of the **Q** matrix in turn over the communication channel. The

choice of the diagonal matrix **P** is arbitrary but specifying it in terms of a signal to noise power ratio will become significant in section (3.8) and (3.9).

Therefore

$$\mathbf{Q} = \mathbf{UP} \tag{3.6.1.1}$$

A set of symbol bearing transmittable signal sequences **S** is generated by encoding the columns of **Q** with an encoding vector for each symbol to be represented, that is

$$\mathbf{s}_i = \mathbf{Q}\mathbf{c}_i \;\forall\, i \in [1,m] \quad \Leftrightarrow \quad \mathbf{S} = \mathbf{QC} \tag{3.6.1.2}$$

where

$$\mathbf{C} = [\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_m] \tag{3.6.1.3}$$

and $\mathbf{c}_i$ represents the i[th] symbol of *m*. The matrix **S** is now transmitted in the same way as matrix **Q**. This method of '*m* symbol' transmission is chosen to increase the transmission efficiency. The transmitted sequence for each symbol is only n samples long whereas the completed transmitted reference **Q** matrix is *m* times longer. The simple solution is to transmit *m* symbols with *m* encoded sequences for each reference sequence. Each encoded sequence can represent 2^m states or symbols with each transmission set of reference and encoded sequences it is possible to transmit a possible $\left(2^m\right)^m$ different symbols by shifting in a bit register *m* bits left for each of the *m* sequences as shown in table (3.6.1.1).

---

m Bits	*m* Bits	<	<	*m* Bits
m	$(m-1)$			1

Table 3.6.1.1

Shift register of *m* bits for each of the *m* sequences.

The potential set of encoding vectors $\bar{\mathbf{c}}_j \;\forall\; j \in [1,2^m]$ is chosen from an encoding map $\overline{\mathbf{C}} \in R^{m \times 2^m}$ which is calculated as

$$\overline{\mathbf{C}} = [\bar{\mathbf{c}}_1, \bar{\mathbf{c}}_2, ..., \bar{\mathbf{c}}_{(2^m)}] \tag{3.6.1.4}$$

$$\bar{\mathbf{c}}_j = \frac{\left(2 \cdot \mathbf{b}(j-1) - \boldsymbol{\mu}_m\right)}{\sqrt{m}} \tag{3.6.1.5}$$

where the $\mathbf{c}_j$ vectors lie on an m dimensional hypersphere of unit radius within the n space; $\boldsymbol{\mu}_m \in R^m$ is a vector $\mu_i = 1 \ \forall\, i \in [1,m]$ and $\mathbf{b}(j-1) \ \forall \ j \in [1,2^m]$ is the bit pattern function converting a bit pattern to an ordered vector, that is

$$\mathbf{b}(j) = [b_1 \ b_2 \cdots b_m]^T \tag{3.6.1.6}$$

and

$$j = 1 + \sum_{i=1}^{m} b_i 2^{m-i} \qquad b_i \in \{0,1\} \ \forall\, i \in [1,m] \tag{3.6.1.7}$$

Example 3.6.1.1

For the system dimension $m = 4$ the twelfth encoding vector represents the number 11, that is $j = 12$. This can be represented by the bit pattern $(1 \quad 0 \quad 1 \quad 1)$ which can be interpreted as $11 = 1\times 2^3 + 0\times 2^2 + 1\times 2^1 + 1\times 2^0$ which maps directly to

$$\mathbf{b}(j-1) = [1 \quad 0 \quad 1 \quad 1]^T \tag{3.6.1.8}$$

Now if $\boldsymbol{\mu}_4 = [1 \quad 1 \quad 1 \quad 1]^T$ it follows that

$$\bar{\mathbf{c}}_{12} = \frac{\left(2 \cdot \mathbf{b}(11) - \boldsymbol{\mu}_4\right)}{\sqrt{4}}$$

$$= \left[\tfrac{1}{2} \quad -\tfrac{1}{2} \quad \tfrac{1}{2} \quad \tfrac{1}{2}\right]^T \tag{3.6.1.9}$$

Consider now the method of decoding the received signals. The equivalent of the correlation integral in equation (3.4.12), is a least squares approximation of the encoding vector, given a noisy received signal matrix $\bar{\mathbf{S}}$. If the signal of the i^{th} column is considered then

$$\bar{\mathbf{s}}_i = \mathbf{Q}\mathbf{c}_i + \sigma\boldsymbol{\varepsilon}_i \tag{3.6.1.10}$$

where

$$\mathbf{Q} = \mathbf{UP} \tag{3.6.1.11}$$

the noise term contains $\boldsymbol{\varepsilon}_i$ which is Gaussian White noise process with a zero mean and a unit variance, that is $E\{\boldsymbol{\varepsilon}_i\}=\mathbf{0}$ and $E\{\boldsymbol{\varepsilon}_i^T\boldsymbol{\varepsilon}_i\}=n$ and $\mathbf{P}$ is a diagonal power balancing matrix. In the following equations, the $\bar{}$ notation indicates a variable derived from received signal data and the ^ indicates an estimated value. Now the signal estimate for a particular symbol represented by a column of the received signals matrix $\bar{\mathbf{S}}$ is given by

$$\hat{\mathbf{s}}_i=\bar{\mathbf{Q}}\hat{\mathbf{c}}_i \tag{3.6.1.12}$$

expressing the error between the received signal vector and the estimated one as

$$\mathbf{e}_i=\bar{\mathbf{s}}_i-\hat{\mathbf{s}}_i \tag{3.6.1.13}$$

and forming a squared error sum as

$$\eta_i=\mathbf{e}_i^T\mathbf{e}_i \tag{3.6.1.14}$$

now minimize η_i with respect to the estimate of the encoding vector $\hat{\mathbf{c}}_i$ so

$$2\mathbf{e}_i^T\frac{\partial\mathbf{e}_i}{\partial\hat{\mathbf{c}}_i}=\mathbf{0}^T \tag{3.6.1.15}$$

from equations (3.6.1.12) and (3.6.1.13)

$$\frac{\partial\mathbf{e}_i}{\partial\hat{\mathbf{c}}_i}=-\bar{\mathbf{Q}} \tag{3.6.1.16}$$

Therefore (3.6.1.15) can be rearranged, incorporating equations (3.6.1.12) and (3.6.1.13) as

$$\bar{\mathbf{Q}}^T(\bar{\mathbf{s}}_i-\bar{\mathbf{Q}}\hat{\mathbf{c}}_i)=\mathbf{0} \tag{3.6.1.17}$$

And finally forming an estimate of the encoding vector by rearranging (3.6.1.17) as

$$\hat{\mathbf{c}}_i=\left[\bar{\mathbf{Q}}^T\bar{\mathbf{Q}}\right]^{-1}\bar{\mathbf{Q}}^T\bar{\mathbf{s}}_i \tag{3.6.1.18}$$

This is the least squares discrete equivalent of the correlation integral given by equation (3.4.12). Now if all m sequences are considered then this equation becomes

$$\hat{\mathbf{C}}=\left[\bar{\mathbf{Q}}^T\bar{\mathbf{Q}}\right]^{-1}\bar{\mathbf{Q}}^T\bar{\mathbf{S}} \tag{3.6.1.19}$$

where

$$\hat{\mathbf{C}}=\left[\hat{\mathbf{c}}_1,\hat{\mathbf{c}}_2,...,\hat{\mathbf{c}}_m\right]\quad\text{and}\quad\bar{\mathbf{S}}=\left[\bar{\mathbf{s}}_1,\bar{\mathbf{s}}_2,...,\bar{\mathbf{s}}_m\right] \tag{3.6.1.20}$$

3.6.2 Indirect '*m* Symbol' 'X' Scheme

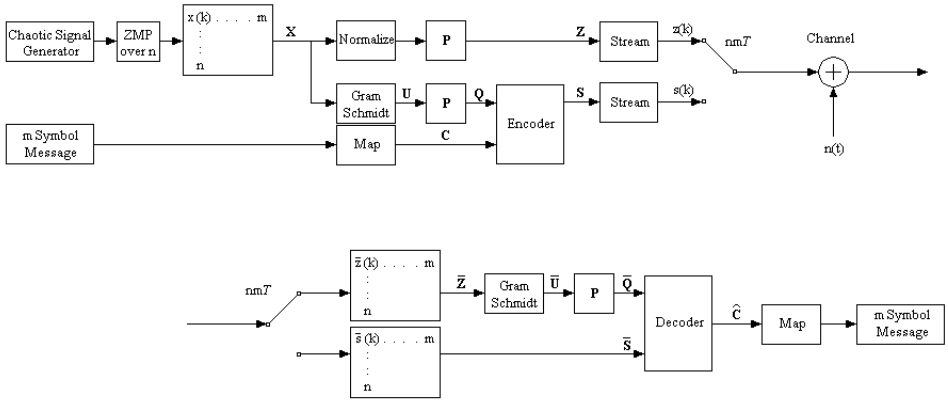


Figure 3.6.2.1

'X' Scheme-Indirect '*m* Symbol' Transmission System Architecture

For the second transmission scheme, consider the same $n \times m$ signal matrix $\mathbf{X}$ produced by taking *n* times *m* samples of the chaotic process as in section (3.6.1). Now an $n \times m$ orthonormal matrix $\mathbf{U}$ is generated in the same way as before using the Gram-Schmidt process but in this scheme; instead of transmitting this matrix after, post multiplying it by the diagonal **P** power balancing matrix, the **X** matrix is normalized and power balanced, before it is transmitted as the **Z** matrix, that is

$$\mathbf{Z} = \bar{\mathbf{X}}\mathbf{P} \tag{3.6.2.1}$$

where $\bar{\mathbf{X}}$ here is the normalized form of **X** given as

$$\bar{\mathbf{X}} = [\bar{\mathbf{x}}_1 \cdots \bar{\mathbf{x}}_m] \tag{3.6.2.2}$$

and

$$\bar{\mathbf{x}}_i^T \bar{\mathbf{x}}_i = 1 \quad \forall \quad i \in [1, m] \tag{3.6.2.3}$$

Again the matrix **U** is multiplied by the diagonal power balancing matrix **P** to produce a matrix **Q** as in equation (3.6.1.1). The encoding of the **S** matrix is exactly as detailed in section (3.6.1) for the first transmission scheme and is transmitted in the same way as matrix **Z**.

Consider now, the method for decoding this set of received signals. Again the equivalent of the correlation integral in equation (3.4.12) is a least squares

approximation of the encoding vector given a noisy received signal matrix $\overline{\mathbf{S}}$. This differs from the first scheme since the received matrix is not the noise contaminated $\overline{\mathbf{U}}$ matrix but a non-orthogonal signal set received as matrix $\overline{\mathbf{Z}}$. Both $\overline{\mathbf{U}}$ and $\overline{\mathbf{Q}}$ matrices can be formed by using the $\overline{\mathbf{Z}}$ matrix via the Gram-Schmidt process. If the signal of the i^{th} column is considered then the equivalent of equations (3.6.1.10) and (3.6.1.11) becomes

$$\bar{\mathbf{s}}_i = \overline{\mathbf{Q}}\mathbf{c}_i + \sigma\boldsymbol{\varepsilon}_i \tag{3.6.2.4}$$

and

$$\overline{\mathbf{Q}} = \overline{\mathbf{U}}\mathbf{P} \tag{3.6.2.5}$$

in the same way, the noise term contains $\boldsymbol{\varepsilon}_i$ which is Gaussian White noise with a zero mean and a unit variance, that is $E\{\boldsymbol{\varepsilon}_i\} = \mathbf{0}$, $E\{\boldsymbol{\varepsilon}_i^T \boldsymbol{\varepsilon}_i\} = n$ and $\mathbf{P}$ is a diagonal power balancing matrix. As before, equations with the $\bar{}$ notation indicate a variable derived from received signal data and the ^ indicates an estimated value. The estimation of the encoding vectors is the same as in section (3.6.1) equations (3.6.1.12) to (3.6.1.20), excepting that due to the Gram-Schmidt process now residing in both the transmitter and the receiver a slight variation occurs. Consider the estimate equation for a single symbol

$$\hat{\mathbf{c}}_i = \left[\overline{\mathbf{Q}}^T\overline{\mathbf{Q}}\right]^{-1}\overline{\mathbf{Q}}^T\bar{\mathbf{s}}_i \tag{3.6.2.6}$$

now substitute into this equation the equation (3.6.2.5) to give

$$\hat{\mathbf{c}}_i = \left[\mathbf{P}^T\overline{\mathbf{U}}^T\overline{\mathbf{U}}\mathbf{P}\right]^{-1}\mathbf{P}^T\overline{\mathbf{U}}^T\bar{\mathbf{s}}_i \tag{3.6.2.7}$$

because the $\overline{\mathbf{U}}$ matrix was generated by the Gram-Schmidt process it follows that

$$\overline{\mathbf{U}}^T\overline{\mathbf{U}} = \mathbf{I}_m \tag{3.6.2.8}$$

and as the $\mathbf{P}$ matrix is diagonal equation (3.6.2.7) becomes

$$\hat{\mathbf{c}}_i = \mathbf{P}^{-1}\overline{\mathbf{U}}^T\bar{\mathbf{s}}_i \tag{3.6.2.9}$$

if all m sequences are considered then this equation becomes

$$\hat{\mathbf{C}} = \mathbf{P}^{-1}\overline{\mathbf{U}}^T\overline{\mathbf{S}} \tag{3.6.2.10}$$

where

$$\hat{\mathbf{C}} = [\hat{\mathbf{c}}_1, \hat{\mathbf{c}}_2, ..., \hat{\mathbf{c}}_m] \quad , \quad \overline{\mathbf{S}} = [\overline{\mathbf{s}}_1, \overline{\mathbf{s}}_2, ..., \overline{\mathbf{s}}_m] \text{ and } \mathbf{P}^{-1} = \frac{1}{p}\mathbf{I}_m \tag{3.6.2.11}$$

where p is the power balancing gain.

This scheme has a more robust estimating structure, because it avoids the noise transmission through an m dimensional matrix inversion. However, it is clearly more dependent on the nature of the noise transmission through the Gram-Schmidt process, which is now resident in the receiver. This is explored in more detail in sections (3.8) and (3.9).

3.6.3 Indirect Persistent 'x' Scheme

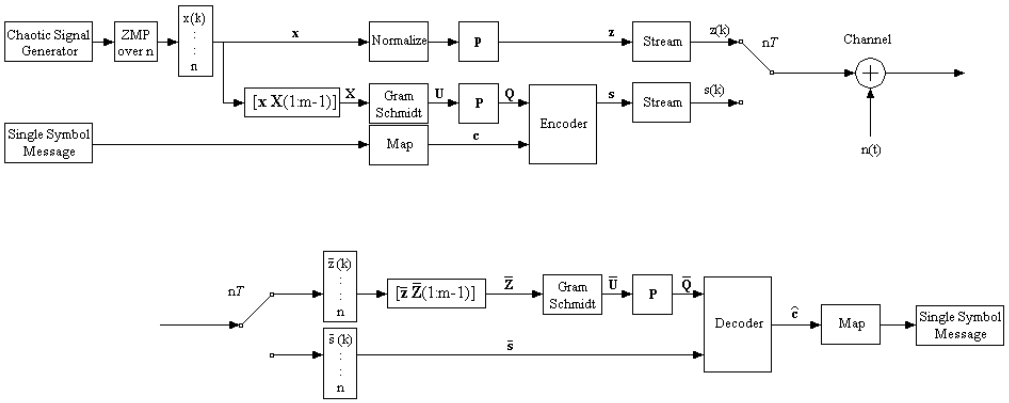


Figure 3.6.3.1

'x' Scheme - Indirect Persistent Transmission System Architecture

The third scheme is based on the scheme of section (3.6.2), except that here single symbols are transmitted, and the orthogonal signal sets are created from collections of n length $\mathbf{x}$ vectors; which remain persistent within the encoding architecture over m symbolic transmissions. Each symbol sequence transmits m bits of information so the transmission efficiency of this scheme is high, because the symbolic and bit data rate is only dependent on the length of each signal vector. For this transmission scheme consider a signal vector $\mathbf{x}$, produced by taking n samples of the chaotic process as in section (3.6.1). Now an $n \times m$ orthonormal matrix $\mathbf{U}$ is generated in the same way as

before, using the Gram-Schmidt process, but in this scheme the required **X** matrix is made up from *m* successive **x** signal vectors, and instead of transmitting this matrix, after post multiplying it by the diagonal **P** power balancing matrix, the **x** vector is normalized and power balanced before it is transmitted as the **z** vector. That is

$$\mathbf{z} = \tilde{\mathbf{x}}p \tag{3.6.3.1}$$

where $\tilde{\mathbf{x}}$ here is the normalized form of **x** and

$$\tilde{\mathbf{x}}^T\tilde{\mathbf{x}} = 1 \tag{3.6.3.2}$$

A new **X** matrix is created after each **x** vector is sampled as

$$\mathbf{X}_{n,m} = [\mathbf{x} \quad \mathbf{X}_{n,m-1}] \tag{3.6.3.3}$$

A new **U** matrix is generated every *n* samples and this is multiplied by the diagonal power balancing matrix **P** to produce a matrix **Q** as in equation (3.6.1.1). It is now necessary only to encode a single symbol represented by an **s** vector as

$$\mathbf{s} = \mathbf{Qc} \tag{3.6.3.4}$$

and this is transmitted in that same way as the **z** vector.

Consider now the method for decoding each received signal vector. Again the equivalent of the correlation integral in equation (3.6.2.6) is a least squares approximation of the encoding vector given a noisy received signal vector $\bar{\mathbf{s}}$. This now differs from the second scheme, since a $\overline{\mathbf{Z}}$ needs to be created in order to decode the signals, in the same way as the second scheme in section (3.6.2). This is created as

$$\overline{\mathbf{Z}}_{n,m} = [\bar{\mathbf{z}} \quad \overline{\mathbf{Z}}_{n,m-1}] \tag{3.6.3.5}$$

now both $\overline{\mathbf{U}}$ and $\overline{\mathbf{Q}}$ matrices can be formed by using the $\overline{\mathbf{Z}}$ matrix via the Gram-Schmidt process. The decoding is equivalent to the second scheme yielding the following

$$\hat{\mathbf{c}}_i = \mathbf{P}^{-1}\overline{\mathbf{U}}^T\bar{\mathbf{s}}_i \tag{3.6.3.6}$$

where

$$\mathbf{P}^{-1} = \frac{1}{p}\mathbf{I}_m \tag{3.6.3.7}$$

where *p* is the power balancing gain.

This scheme has the same robust estimating structure as the second scheme, because it avoids the noise transmission through an *m* dimensional matrix inversion, and it has the same dependency on the nature of the noise transmission through the Gram-Schmidt process. The cyclic transmission efficiency is increased and is scaleable with the dimension *m,* without any noise or time penalties.

3.7 BER Probability Formulation

To find the Bit Error Rate (BER), as a function of the number of samples *n* for each bit and the signal power to noise power ratio P_{snr} of the system, the following probability formulation will enable a simple method of BER calculation to be developed in section (3.9). The BER is considered as the probability that a singular transmitted bit is decoded incorrectly in the receiver. This can be considered as a function of the probability of the estimate of the encoding vector lying outside its permitted region. Consider then the probability of getting all the bits correct that is

$$P(C_\mu) = 1 - P(E_\mu) \tag{3.7.1}$$

where $P(E_\mu)$ is the probability of any error and μ is the number of symbols, hence if b is the number of bits representing μ symbols then

$$\mu = 2^b \tag{3.7.2}$$

The probability of the i^{th} symbol being correct is

$$P(C_i) = \sqrt[b]{P(C_\mu)} \tag{3.7.3}$$

this gives the probability of a symbol error as

$$P(E_i) = 1 - P(C_i) \tag{3.7.4}$$

Now this is equivalent to the Bit Error Rate so from equations (3.7.1) to (3.7.4)

$$BER = 1 - \sqrt[b]{1 - P(E_\mu)} \tag{3.7.5}$$

3.8 Signal Characterization

In this section, the effects of noise transmitted through the various processes in the estimators are considered. As the third transmission scheme, the 'Indirect Persistent **x** Scheme' described in section (3.6.3) is essentially the same, in a noise transmission sense, as the 'Indirect *m* Symbol **X** Scheme' described in section (3.6.2); this section

has two parts, one looking at the relatively simple characterization of the 'Direct *m* Symbol **U** Scheme' of section (3.6.1), and the more complex ideas of the 'Indirect *m* Symbol **X** Scheme'.

3.8.1 Direct 'm Symbol' '**U**' Scheme

The transmittable signal matrix **U**, derived from the Gram-Schmidt process, can be characterized by considering it to be expressed as a series of column vectors with the same power. As **U** consists of an orthonormal basis over a limited *m* dimensional span of the *n* space it can be expressed as

$$\mathbf{U} = \left[\frac{\mathbf{I}_m}{\mathbf{0}_{n-m,m}} \right] \tag{3.8.1.1}$$

If a scheme uses this matrix as the reference, then it would be power balanced before transmission, and by definition each column would have the same power. The signals are derived from an arbitrary chaotic process, and for the purposes of characterization, can be aggregated in this way. This is clearly independent of any process and is notionally derived in this form to ensure it has the correct properties principally that

$$\mathbf{U}^T\mathbf{U} = \mathbf{I}_m \tag{3.8.1.2}$$

Before transmission, this matrix is multiplied by a diagonal power balancing matrix to yield a transmittable matrix **Q**. When this matrix is contaminated in the transmission channel by zero mean Gaussian White noise with a variance of σ^2 the received signal matrix can be expressed as

$$\overline{\mathbf{Q}} = \mathbf{Q} + \sigma\mathbf{E} \tag{3.8.1.3}$$

Where **E** is a matrix of Gaussian White noise signals of unit variance with the same dimensions as **Q**.

Now the estimator matrix is constructed as follows

$$\overline{\mathbf{Q}} = \mathbf{UP} + \sigma\mathbf{E} \tag{3.8.1.4}$$

In order to complete this characterization we need to consider the diagonal power balancing matrix **P** in terms of the Power to Signal Noise Ratio P_{snr}.

Define the P_{snr} as

$$P_{snr} = \frac{\|\mathbf{x}\|^2}{\sigma^2} \tag{3.8.1.4}$$

where $\|\mathbf{x}\|^2$ is the power of an arbitrary signal vector $\mathbf{x}$ and can be considered as the norm of the length n vector given as

$$\|\mathbf{x}\|^2 = \frac{1}{n}\mathbf{x}^T\mathbf{x} \tag{3.8.1.5}$$

Now we require that

$$\mathbf{q}^T\mathbf{q} = n\|\mathbf{x}\|^2 \tag{3.8.1.6}$$

that is, it is required that the transmitted vectors have a nominal power $\|\mathbf{x}\|^2$ and the total energy content increases linearly as the vector length n. For the scheme being considered

$$\mathbf{q}_i^T\mathbf{q}_i = p^2\mathbf{u}_i^T\mathbf{u}_i = p^2 \quad \forall \quad i \in [1,m] \tag{3.8.1.7}$$

Combining equations (3.8.1.4), (3.8.1.6) and (3.8.1.7) yields

$$p = \sigma\sqrt{nP_{snr}} \tag{3.8.1.8}$$

and therefore

$$\mathbf{P} = \sigma\sqrt{nP_{snr}} \cdot \mathbf{I}_m \tag{3.8.1.9}$$

This finally allows the desired characterization of $\overline{\mathbf{Q}}$ as

$$\overline{\mathbf{Q}} = \sigma\left(\sqrt{nP_{snr}} \cdot \begin{bmatrix} \mathbf{I}_m \\ \hline \mathbf{0}_{n-m,m} \end{bmatrix} + \mathbf{E}\right) \tag{3.8.1.10}$$

Likewise the i^{th} symbol signal vector can be characterized in the same way, that is

$$\bar{\mathbf{s}}_i = \mathbf{Q}\mathbf{c}_i + \sigma\boldsymbol{\varepsilon} \quad \forall \quad i \in [1,m] \tag{3.8.1.11}$$

where $\boldsymbol{\varepsilon}$ is an n length vector of Gaussian White noise signals of unit variance. Substituting equations (3.8.1.1) and (3.8.1.9) into this equation (3.8.1.11) gives

$$\bar{\mathbf{s}}_i = \sigma\left(\sqrt{nP_{snr}} \cdot \begin{bmatrix} \mathbf{I}_m \\ \hline \mathbf{0}_{n-m,m} \end{bmatrix}\mathbf{c}_i + \boldsymbol{\varepsilon}\right) \quad \forall \quad i \in [1,m] \tag{3.8.1.12}$$

These results give characterizations of the received signal vectors $\bar{\mathbf{s}}_t$ and the received reference matrix $\overline{\mathbf{Q}}$. They have a simple structure for the message bearing element and have simple Gaussian white noise elements which will allow simple analysis of the noise propagation through the entire communications scheme.

3.8.2 Indirect 'm Symbol' '**X**' Scheme

The transmittable signal matrix $\mathbf{Z}$ is received in a noise contaminated form as $\overline{\mathbf{Z}}$. A noise contaminated orthonormal set of signal vectors $\overline{\mathbf{U}}$ can be derived using the Gram-Schmidt process and an equivalent $\overline{\mathbf{Q}}$ matrix can be found as

$$\overline{\mathbf{Q}} = \overline{\mathbf{U}}\mathbf{P} \tag{3.8.2.1}$$

The $\overline{\mathbf{U}}$ matrix has properties that can be characterized, by considering it to be expressed as a series of column vectors with the same power. As $\overline{\mathbf{U}}$ consists of an orthonormal basis over a limited m dimensional span of the n space it can be expressed as

$$\overline{\mathbf{U}} = \left[\frac{\mathbf{I}_m}{\mathbf{0}_{n-m,m}} \right] \tag{3.8.2.2}$$

This signal aggregation is valid, but does not describe the errors induced by the presence of noise on the Gram-Schmidt process. This is clearly independent of any process and is notionally derived in this form to ensure it has the correct properties principally that

$$\overline{\mathbf{U}}^T \overline{\mathbf{U}} = \mathbf{I}_m \tag{3.8.2.3}$$

A simple way of characterizing the effect of the Gram-Schmidt process is to consider how the transmitted matrix $\mathbf{Z}$ is constructed. The $\mathbf{Z}$ matrix is a power balanced version of a normalized set of signal vectors generated by a chaotic process. As such they are inherently not orthogonal, and can be considered as the result of an upper triangular linear transformation of an orthonormal basis, as described in section (3.5) equations (3.5.3) to (3.5.6), (3.6.2.1) and (3.8.1.9), that is

$$\begin{aligned} \mathbf{Z} &= \mathbf{XP} \\ &= \sigma\sqrt{nP_{snr}} \cdot \mathbf{UW} \end{aligned} \tag{3.8.2.4}$$

So finally the $\overline{\mathbf{Q}}$ matrix can be characterized in the following way if $\mathbf{G}(\overline{\mathbf{Z}})$ is the Gram-Schmidt matrix function and $\mathbf{W}$ is upper triangular with

$$\mathbf{w}_i^T \mathbf{w}_i = 1 \quad \forall \quad i \in [1, m] \tag{3.8.2.5}$$

it follows that

$$\overline{\mathbf{U}} = \mathbf{G}\left(\sigma\left(\sqrt{nP_{snr}} \cdot \mathbf{UW} + \mathbf{E}\right)\right) \tag{3.8.2.6}$$

So finally

$$\overline{\mathbf{Q}} = \sigma\sqrt{nP_{snr}} \cdot \mathbf{G}\left(\sigma\left(\sqrt{nP_{snr}} \cdot \mathbf{UW} + \mathbf{E}\right)\right) \tag{3.8.2.7}$$

the standard deviation constant σ within the Gram-Schmidt matrix function has no effect here, since it merely changes the component vector lengths and not their relationship to one another. Therefore this can be written as

$$\overline{\mathbf{Q}} = \sigma\sqrt{nP_{snr}} \cdot \mathbf{G}\left(\sqrt{nP_{snr}} \cdot \mathbf{UW} + \mathbf{E}\right) \tag{3.8.2.8}$$

Likewise, as before, the i^{th} symbol signal vector can be characterized in the same way as in section (3.8.1), that is

$$\bar{\mathbf{s}}_i = \mathbf{Q}\mathbf{c}_i + \sigma\boldsymbol{\varepsilon} \quad \forall \quad i \in [1, m] \tag{3.8.2.9}$$

$\boldsymbol{\varepsilon}$ is an n length vector of Gaussian White noise signals of unit variance and so substituting equations (3.8.1.1) and (3.8.1.9) into this equation gives

$$\bar{\mathbf{s}}_i = \sigma\left(\sqrt{nP_{snr}} \cdot \left[\frac{\mathbf{I}_m}{\mathbf{0}_{n-m,m}}\right]\mathbf{c}_i + \boldsymbol{\varepsilon}\right) \quad \forall \quad i \in [1, m] \tag{3.8.2.10}$$

3.9 Signal to Noise Calculations

In order to evaluate the performance of these transmission schemes against other schemes, a novel way is presented for producing Bit Error Rate results using the probability formulation of section (3.7) and the signal characterization of section (3.8). The results differ for each scheme and are presented in the next two sections.

3.9.1 Direct '*m* Symbol' '**U**' Scheme

The i^{th} symbol encoding vector estimate can be expressed as follows

$$\hat{\mathbf{c}}_i = [\overline{\mathbf{Q}}^T \overline{\mathbf{Q}}]^{-1} \overline{\mathbf{Q}}^T \bar{\mathbf{s}}_i \quad \forall \quad i \in [1, m] \tag{3.9.1.1}$$

Now the estimate can be expressed in terms of the noise on the transmission channel and the original idealised symbol encoding vector by substituting equations (3.8.1.10) and (3.8.1.12) into (3.9.1.1), that is

$$\overline{\mathbf{Q}} = \sigma\left(\sqrt{nP_{snr}} \cdot \left[\frac{\mathbf{I}_m}{\mathbf{0}_{n-m,m}}\right] + \mathbf{E}\right) \tag{3.9.1.2}$$

and

$$\bar{\mathbf{s}}_i = \sigma\left(\sqrt{nP_{snr}} \cdot \left[\frac{\mathbf{I}_m}{\mathbf{0}_{n-m,m}}\right]\mathbf{c}_i + \boldsymbol{\varepsilon}\right) \quad \forall \quad i \in [1, m] \tag{3.9.1.3}$$

which yields

$$\hat{\mathbf{c}}_i = \left(\sigma\left(\sqrt{nP_{snr}} \cdot \left[\frac{\mathbf{I}_m}{\mathbf{0}_{n-m,m}}\right] + \mathbf{E}\right)^T \sigma\left(\sqrt{nP_{snr}} \cdot \left[\frac{\mathbf{I}_m}{\mathbf{0}_{n-m,m}}\right] + \mathbf{E}\right)\right)^{-1} \cdots$$

$$\cdots \sigma\left(\sqrt{nP_{snr}} \cdot \left[\frac{\mathbf{I}_m}{\mathbf{0}_{n-m,m}}\right] + \mathbf{E}\right)^T \sigma\left(\sqrt{nP_{snr}} \cdot \left[\frac{\mathbf{I}_m}{\mathbf{0}_{n-m,m}}\right]\mathbf{c}_i + \boldsymbol{\varepsilon}\right)$$

$$= \left(nP_{snr}\mathbf{I}_m + \sqrt{nP_{snr}}(\mathbf{E}^T{}_{m,m} + \mathbf{E}_{m,m}) + \mathbf{E}^T\mathbf{E}\right)^{-1} \cdots$$

$$\cdots \left(nP_{snr}\mathbf{c}_i + \sqrt{nP_{snr}}(\mathbf{E}^T{}_{m,m}\mathbf{c}_i + \boldsymbol{\varepsilon}_m) + \mathbf{E}^T\boldsymbol{\varepsilon}\right) \quad \forall \quad i \in [1, m] \tag{3.9.1.4}$$

Here $\mathbf{E}_{m,m}$ is the first *m* rows of the $\mathbf{E}$ matrix and $\boldsymbol{\varepsilon}_m$ is the first *m* elements of $\boldsymbol{\varepsilon}$.

From this equation the probability formulation of Bit Error Rate, outlined in section (3.7), can now be calculated. If a typical value for the encoding vector $\mathbf{c}_i$ is used in a simulation, then whenever a simulated estimate of $\hat{\mathbf{c}}_i$ is calculated outside this region, the number of failures is summed over a range of P_{snr} values and a waterfall type curve is produced accordingly. The results of this kind of analysis can be seen in chapter 4.

The result here is constructed using the power of the signal to noise ratio P_{snr}, whereas most of the literature quotes the equations and results in terms of the 'energy per bit divided by the noise power', that is $\frac{E_b}{N_0}$. This depends on the bit transmission rate, which in turn, is dependent on the structure of the different signal sequences. P_{snr} is independent of transmission structure. If the transmission bit rate B_r is known and the energy is spread over the reference and the signal sequences then

$$\frac{E_b}{N_0} = \frac{P_{snr}}{2\tau B_r} \tag{3.9.1.5}$$

For orthogonal minimal constellations the bit rate, including the reference sequence time is given by

$$B_r = \frac{m}{2n\tau} \tag{3.9.1.6}$$

where τ is the sampling time of the system.

Substituting equation (3.9.1.6) into (3.9.1.5) and rearranging gives

$$P_{snr} = \frac{m}{n} \cdot \left(\frac{E_b}{N_0} \right) \tag{3.9.1.7}$$

Finally then equation (3.9.1.4) becomes

$$\hat{\mathbf{c}}_i = \left(m\left(\frac{E_b}{N_0} \right)\mathbf{I}_m + \sqrt{m\left(\frac{E_b}{N_0} \right)}(\mathbf{E}^T{}_{m,m} + \mathbf{E}_{m,m}) + \mathbf{E}^T\mathbf{E} \right)^{-1} \cdots$$

$$\cdots \left(m\left(\frac{E_b}{N_0} \right)\mathbf{c}_i + \sqrt{m\left(\frac{E_b}{N_0} \right)}(\mathbf{E}^T{}_{m,m}\mathbf{c}_i + \boldsymbol{\varepsilon}_m) + \mathbf{E}^T\boldsymbol{\varepsilon} \right) \quad \forall \quad i \in [1,m] \tag{3.9.1.8}$$

3.9.2 Indirect '*m* Symbol' '**X**' Scheme

Again the i^{th} symbol encoding vector estimate can be expressed as follows

$$\hat{\mathbf{c}}_i = [\overline{\mathbf{Q}}^T\overline{\mathbf{Q}}]^{-1}\overline{\mathbf{Q}}^T\overline{\mathbf{s}}_i \quad \forall \quad i \in [1,m] \tag{3.9.2.1}$$

As before the estimate can be expressed in terms of the noise on the transmission channel, and the original idealised symbol encoding vector by substituting equations

(3.8.2.8) and (3.8.2.10) into (3.9.2.1), that is

$$\overline{\mathbf{Q}} = \sigma\sqrt{nP_{snr}} \cdot \mathbf{G}\left(\sqrt{nP_{snr}} \cdot \begin{bmatrix} \mathbf{I}_m \\ \mathbf{0}_{n-m,m} \end{bmatrix} \mathbf{W} + \mathbf{E} \right)$$

$$= \sigma\sqrt{nP_{snr}} \cdot \overline{\mathbf{U}} \tag{3.9.2.2}$$

and

$$\overline{\mathbf{s}}_i = \sigma\left(\sqrt{nP_{snr}} \cdot \begin{bmatrix} \mathbf{I}_m \\ \mathbf{0}_{n-m,m} \end{bmatrix} \mathbf{c}_i + \boldsymbol{\varepsilon} \right) \quad \forall \quad i \in [1,m] \tag{3.9.2.3}$$

which yields

$$\hat{\mathbf{c}}_i = \left(\left(\sigma\sqrt{nP_{snr}} \cdot \overline{\mathbf{U}} \right)^T \left(\sigma\sqrt{nP_{snr}} \cdot \overline{\mathbf{U}} \right) \right)^{-1} \cdots$$

$$\cdots \sigma\left(\sqrt{nP_{snr}} \cdot \overline{\mathbf{U}} \right)^T \sigma\left(\sqrt{nP_{snr}} \cdot \begin{bmatrix} \mathbf{I}_m \\ \mathbf{0}_{n-m,m} \end{bmatrix} \mathbf{c}_i + \boldsymbol{\varepsilon} \right)$$

$$= \overline{\mathbf{U}}^T \left(\begin{bmatrix} \mathbf{I}_m \\ \mathbf{0}_{n-m,m} \end{bmatrix} \mathbf{c}_i + \frac{1}{\sqrt{nP_{snr}}} \boldsymbol{\varepsilon} \right) \quad \forall \quad i \in [1,m] \tag{3.9.2.4}$$

where

$$\overline{\mathbf{U}} = \mathbf{G}\left(\sqrt{nP_{snr}} \cdot \begin{bmatrix} \mathbf{I}_m \\ \mathbf{0}_{n-m,m} \end{bmatrix} \mathbf{W} + \mathbf{E} \right) \tag{3.9.2.5}$$

The results again can be presented in terms of the probability formulation of Bit Error Rate outlined in section (3.7) and the 'energy per bit over the noise power' $\frac{E_b}{N_0}$ yielding

$$\hat{\mathbf{c}}_i = \overline{\mathbf{U}}^T \left(\begin{bmatrix} \mathbf{I}_m \\ \mathbf{0}_{n-m,m} \end{bmatrix} \mathbf{c}_i + \frac{1}{\sqrt{m\left(\frac{E_b}{N_0} \right)}} \boldsymbol{\varepsilon} \right) \quad \forall \quad i \in [1,m] \tag{3.9.2.6}$$

3.10 Summary

In this chapter, the limitations of two dimensional M-ary type schemes have been highlighted and further to this, the failure of extending Fourier expansion methods to produce orthogonal signal sets has been demonstrated. A method has then been derived to produce orthogonal signal sets utilizing the Gram-Schmidt orthonormalization process and a number of different system architectures have been presented which increase transmission efficiency and information density over a communication channel. These schemes have then been characterized to allow an analysis of the noise transmission through them and analyses of their performance under varying noisy conditions have been presented.

Chapter 4

Simulation Case Study

4.1 Introduction

A case study for the schemes proposed in chapter 3 is now presented for a dimension of $m = 4$. This dimension has been chosen, because it allows a clear demonstration of the advantages of using these schemes, whilst not presenting information that may be too confusing or complex for the purposes of illustration. The chapter is divided into two parts, section (4.2) presents a set of transmission simulations, for all of the proposed schemes, where simulated real time random messages are transmitted and received with Gaussian White noise added in the communication channel. Illustrated, are the actual transmitted and the received messages and the errors in the decoding of the information, due to the noise in the channel. The next section (4.3) presents the 'Bit Error Rates' (BER) in terms of the 'Power Signal to Noise Ratio' (P_{snr}) and the 'Energy per Bit divided by the Noise Power' $\left(\frac{E_b}{N_0}\right)$. It is required that for each estimator, the probability of the vector estimate lying outside its permitted region is determined, that is

$$P(\hat{\mathbf{c}}_k \notin R^m(\mathbf{c}_k)) = P(E_\mu) \quad \forall \quad k \in [1,2^m] \tag{4.1.1}$$

be found, this is equivalent to the BER. Varying the noise matrices and vectors randomly over a large number of simulation runs produces the following results shown, in each of the figures of sections (4.3.1) and (4.3.2), for P_{snr} and $\frac{E_b}{N_0}$. Plots of BER for

(a) DCSK, (b) QCSK 16 symbol constellation and (c) OCVSK 16 symbol methods are presented. The data rates for the selected examples are $\frac{1}{2n\tau}$, $\frac{2}{n\tau}$ and $\frac{2}{n\tau}$ respectively, where n is the number of samples in the reference and the message bearing part of the signal and τ is the sampling time. The generalized data rate for the orthogonal scheme is $\frac{m}{2n\tau}$ where *m* is the scheme dimension. Although the bit efficiency of the QCSK 16 symbol constellation is as good as the OCVSK 16 scheme, the BER is much worse indicating a clear advantage to the orthogonal scheme.

A further comparison is presented, showing that the Orthogonal Chaotic Vector Shift Keying (OCVSK) methods have BER characteristics equivalent to Differential Chaos Shift Keying (DCSK), when the reference signals are orthogonal. The BER for the DCSK scheme and the OCVSK 16 scheme, has a dimensionally larger number of samples with the same BER, but the OCVSK 16 scheme has four times the data rate. However, as the signal references become more non-orthogonal, represented by non diagonal values of the signal characteristic matrix **W** described in section (3.5), the BER graphs diverge quite markedly. This is illustrated in figures (4.3.1.2), (4.3.2.1.2), (4.3.2.2.2) and (4.3.2.3.2).

The chaotic system used here is the Lorenz system described in section (2.1.3). The advantages of this system are that it is simple, and has sufficiently chaotic behaviour for the purposes of demonstrating chaotic communication methods; but has characteristics that illustrate the problems that systems with a degree of periodicity can cause orthogonally oriented communication schemes. The systems equations used for these results are

$$\begin{aligned} \alpha\dot{x} &= -\sigma x + \sigma y \\ \alpha\dot{y} &= rx - y - xz \\ \alpha\dot{z} &= xy + \beta z \end{aligned} \qquad (4.1.1)$$

where $r = 28$, $\sigma = 10$ and $\beta = 8/3$. The constant α can be chosen to suit the sampling time that the particular system requires. For the following simulations $\alpha = 1$, all sampling times are assumed to be units of the chosen sample period and the first state $x(t)$ is used as the signal to be sampled.

4.2 Transmission Simulations

4.2.1 Direct '*m* Symbol' 'U' Scheme

Figure (4.2.1.1) shows a series of graphs illustrating how the various stages of the scheme in section (3.6.1) are processed. Graph (a) shows the matrix **X** of zero mean sampled sequences generated from the chaotic process. Graph (b) shows the orthonormal basis matrix sequences **U** generated from the matrix **X** which in turn, when multiplied by the diagonal power balancing matrix **P,** generates the **Q** matrix in graph (c). The **Q** matrix sequences are then streamed and transmitted. When these sequences are resampled at the receiver they have been contaminated by noise, with a power to signal noise ratio of unity, as shown in graph (d). The **Q** matrix is now encoded with *m* symbol vectors to generate the **S** matrix of graph (e), and this is streamed and received in the same way as the **Q** matrix, to yield the noise contaminated $\overline{\mathbf{S}}$ matrix in graph (f). These graphs show the diverse and spread spectrum nature of the chaotic signals, which make up the signals for all of the transmission schemes, and one of the principal reasons why these types of schemes are particularly able to reject banded limited noise.

A set of transmitted 'sixteen' bit messages, represented by four separate 'four' bit groups, is shown in figure (4.2.1.2) graphs (a) and (b). Graphs (c) and (d) show the decoded messages in both 'sixteen' bit form and as four 'four' bit groups. They show the received messages delayed by *nm* time samples, and the following two graphs (e) and (f), demonstrates that there is zero error between the transmitted and the received message sequence at a P_{snr} of unity when the time delay is accounted for. This clearly illustrates the noise rejection properties of using orthogonal sequences as reference signals.

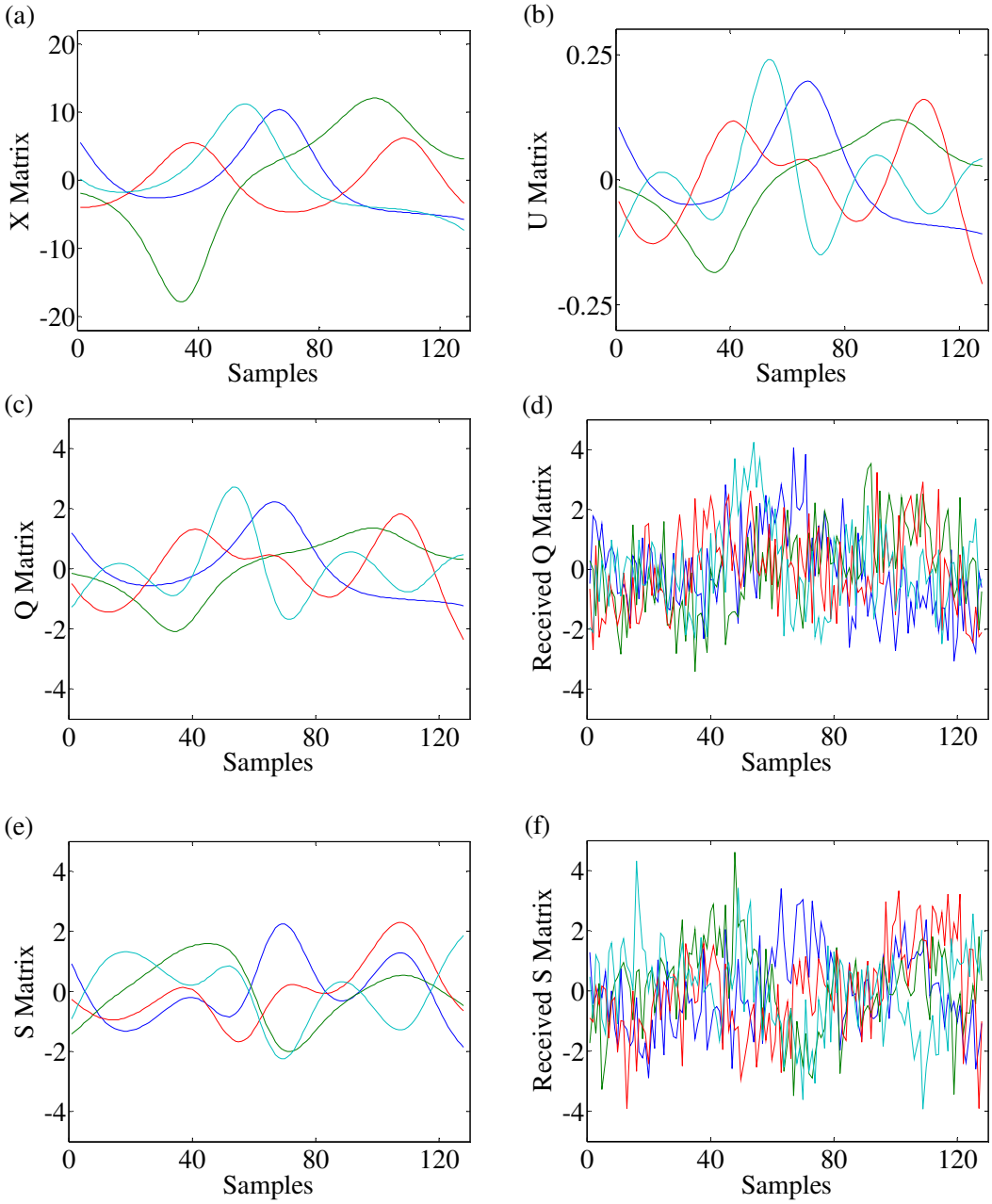


Figure 4.2.1.1 'U' Scheme - Direct '*m* Symbol' System Transmission Signals

$n = 128$, $m = 4$ and Power of Signal to Noise Ratio = 1.0

(a) **X** Matrix zero mean chaotic sequences for $m = 4$

(b) **U** Matrix generated orthogonal reference sequences

(c) **Q** Matrix transmitted power balanced orthogonal reference sequences

(d) $\overline{\mathbf{Q}}$ Matrix received power balanced orthogonal reference sequences

(e) **S** Matrix transmitted encoded signal sequences

(f) $\overline{\mathbf{S}}$ Matrix received encoded signal sequences

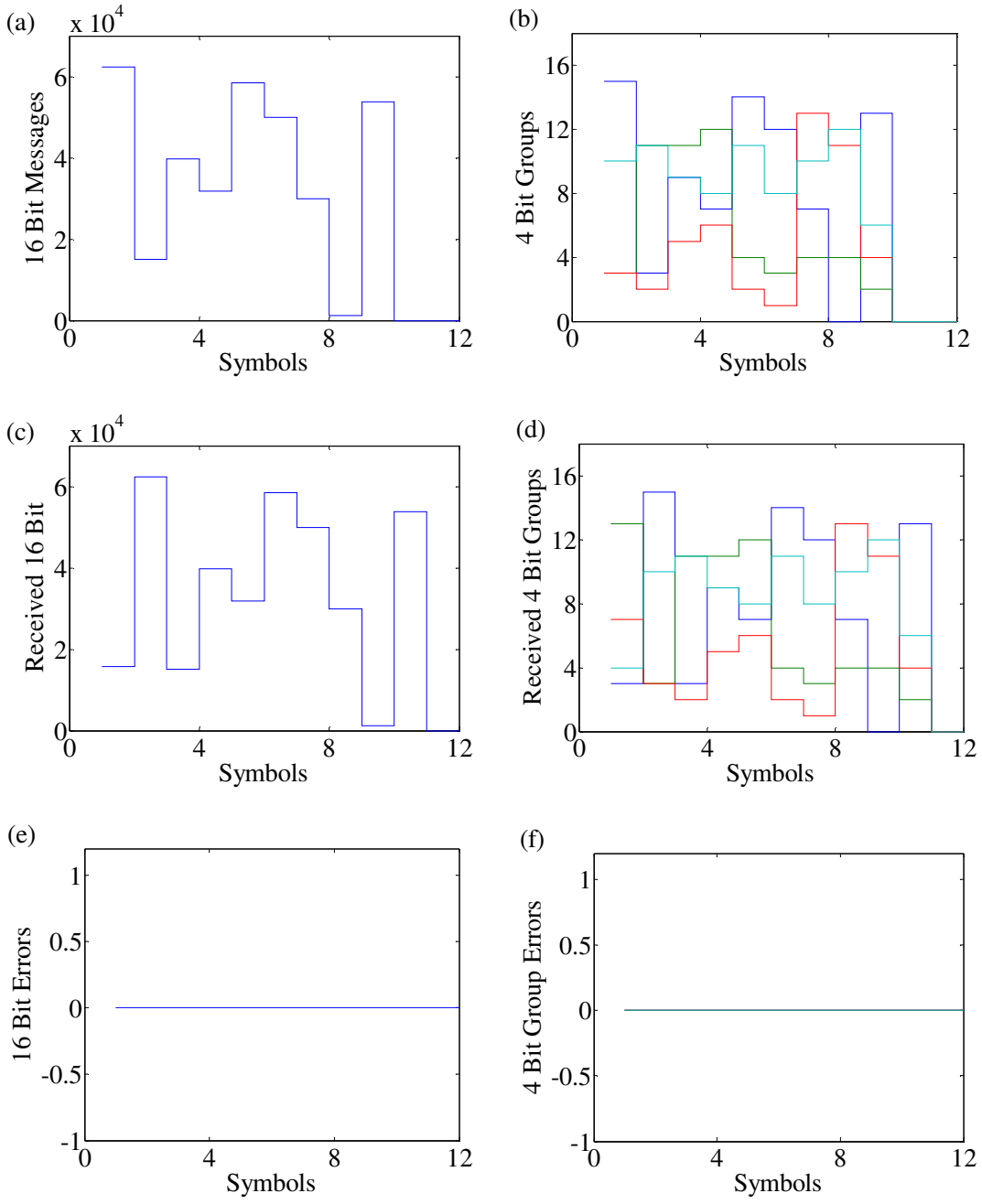


Figure 4.2.1.2 'U' Scheme - Direct '*m* Symbol' System Message Transmissions

n = 128, *m* = 4 and Power of Signal to Noise Ratio = 1.0

(a) Transmitted 16 bit message for encoding

(b) Transmitted message broken down into individual 4 bit groups

(c) Received decoded 16 bit message

(d) Received message broken down into 4 bit groups

(e) Transmitted/Received 16 bit message delayed error

(f) Transmitted/Received 4 bit group message delayed error

4.2.2 Indirect '*m* Symbol' 'X' Scheme

For the next scheme figure (4.2.1.2) shows a series of graphs illustrating the various stages of the scheme described in section (3.6.2). Graph (a) shows the matrix **X** of zero mean sampled sequences generated from the chaotic process. Graph (b) shows the orthonormal basis matrix sequences **U** generated from the matrix **X** which in turn, when multiplied by the diagonal power balancing matrix **P,** give rise to the **Q** matrix which is used for encoding the signals sequence matrix **S** shown in graph (e). In this scheme the references are generated from the **X** matrix by normalizing it and power balancing it with the diagonal P matrix to generate the **Z** matrix in graph (c), they are then streamed and transmitted. When these sequences are resampled at the receiver they have been contaminated by noise, with a power to signal noise ratio of unity, as shown in graph (d). The **Q** matrix is now encoded with *m* symbol vectors to generate the **S** matrix of graph (e), and this is streamed and received in the same way as the **Z** matrix, to yield the noise contaminated $\overline{\mathbf{S}}$ matrix in graph (f). There are advantages to this scheme which will be more fully realised in the next transmission scheme, but the prime disadvantage here is that the **Z** matrix sequences are not orthogonal, and are thus more liable to noise induced errors on decoding. This illustrates why the sequences should be chosen carefully via the matrix conditioning method described in section (3.5).

The same set of transmitted 'sixteen' bit messages, represented by four separate 'four' bit groups, are shown in figure (4.2.2.2) graphs (a) and (b). Graphs (c) and (d) show the decoded messages in both 'sixteen' bit form and as four 'four' bit groups. They show the received messages delayed by *nm* time samples and the following two graphs, (e) and (f), demonstrates that there are errors between the transmitted and the received message sequence at a P_{snr} of unity, when the time delay is accounted for. This clearly illustrates that the noise rejection properties are impaired by using non orthogonal sequences as references signals. This is modelled and illustrated in section (4.3) when the BER of these schemes is considered.

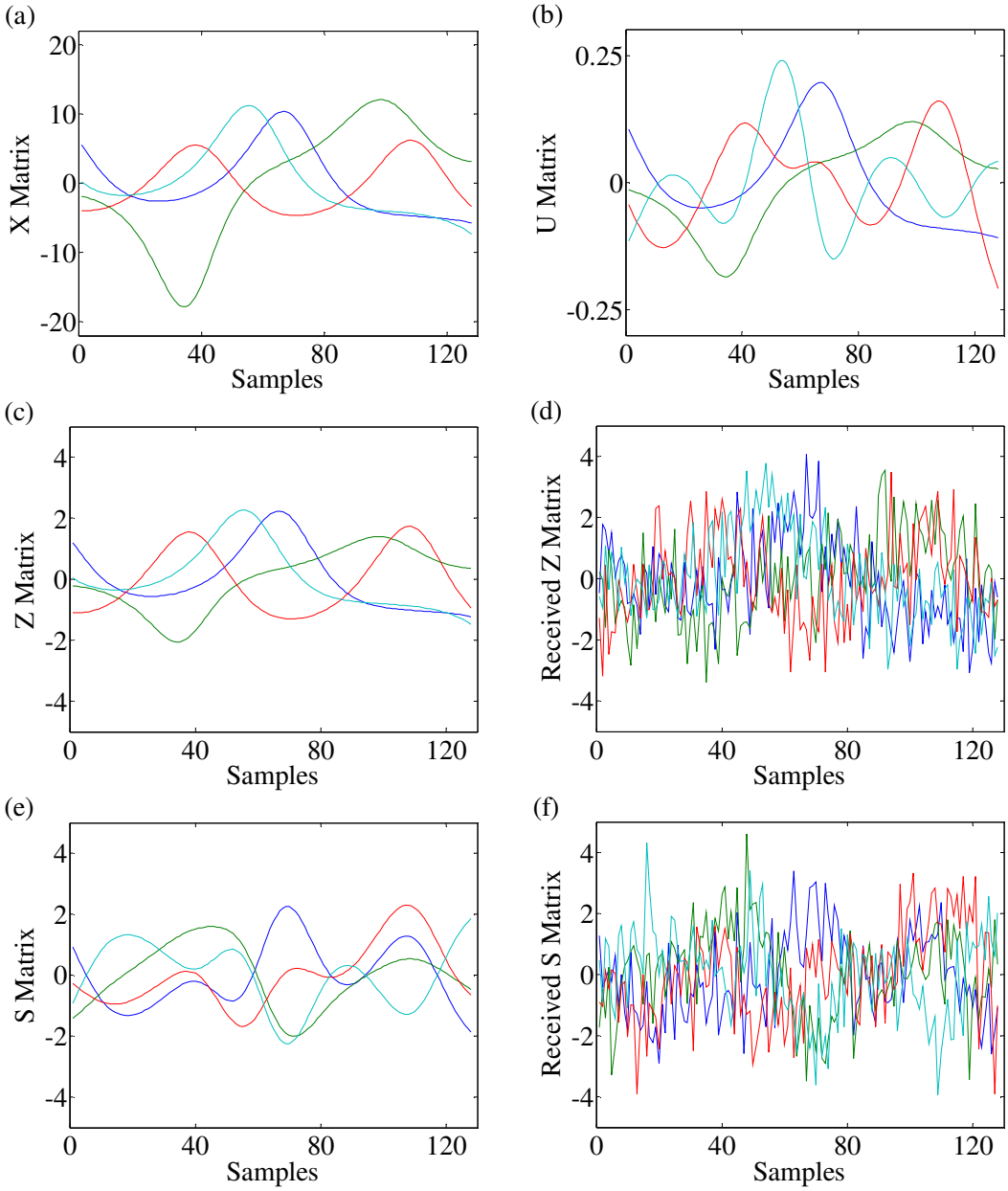


Figure 4.2.2.1 'X' Scheme - Direct '*m* Symbol' System Transmission Signals

$n = 128$, $m = 4$ and Power of Signal to Noise Ratio = 1.0

(a) **X** Matrix zero mean chaotic sequences for $m = 4$

(b) **U** Matrix generated orthogonal reference sequences

(c) **Z** Matrix transmitted power balanced reference sequences

(d) $\overline{\mathbf{Z}}$ Matrix received power balanced reference sequences

(e) **S** Matrix transmitted encoded signal sequences

(f) $\overline{\mathbf{S}}$ Matrix received encoded signal sequences

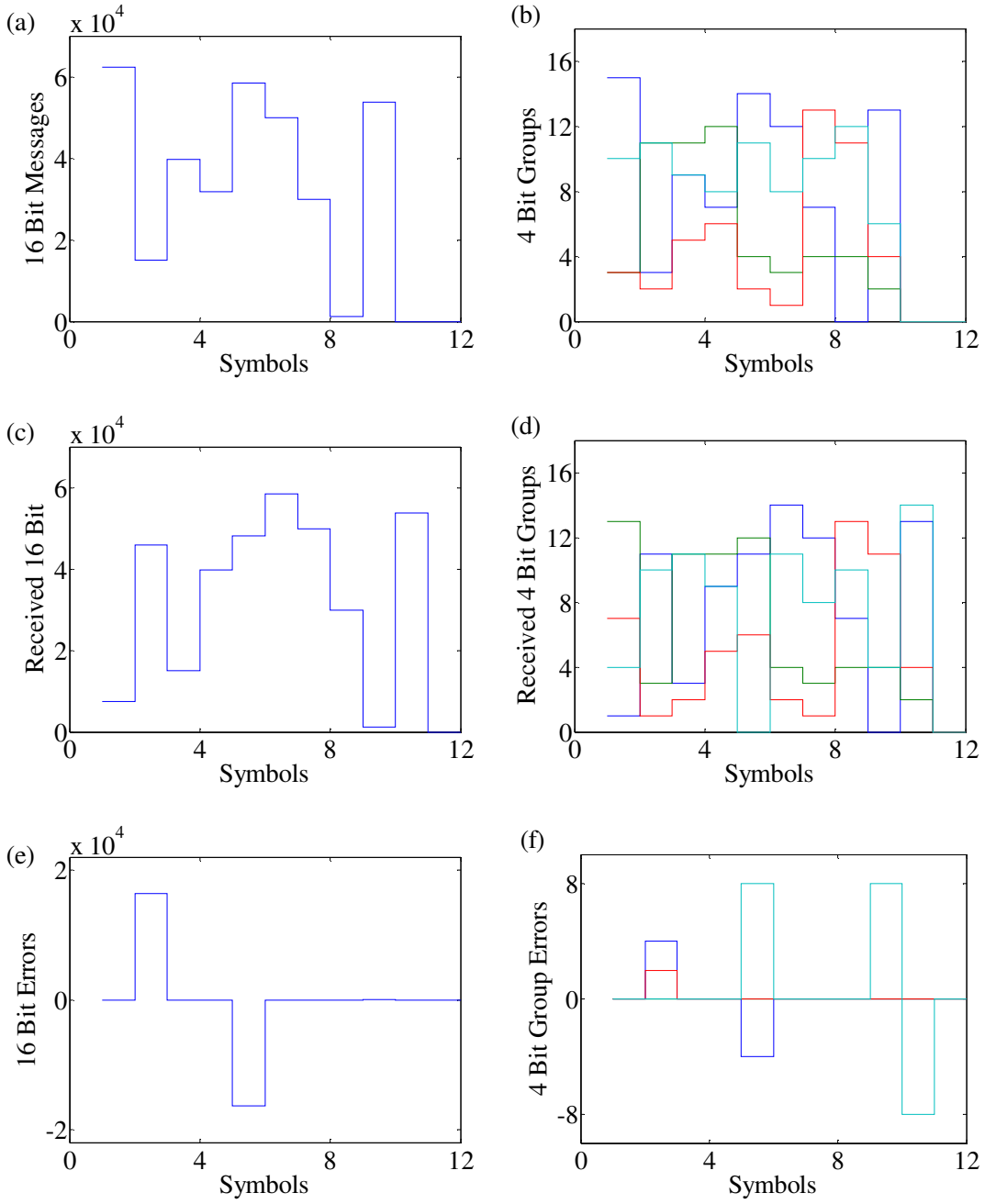


Figure 4.2.2.2 'X' Scheme - Direct '*m* Symbol' System Message Transmissions

n = 128, *m* = 4 and Power of Signal to Noise Ratio = 1.0

(a) Transmitted 16 bit message for encoding

(b) Transmitted message broken down into individual 4 bit groups

(c) Received decoded 16 bit message

(d) Received message broken down into 4 bit groups

(e) Transmitted/Received 16 bit message delayed error

(f) Transmitted/Received 4 bit group message delayed error

4.2.3 Indirect Persistent 'x' Scheme

The next four figures (4.2.3.1) to (4.2.3.4) illustrate the results of simulating the scheme described in section (3.6.3) for two values of $P_{snr} = 1.0$ and $P_{snr} = 10.0$. The simulation uses the same chaotic sequences, that the first two schemes used, that have not been enhanced by the matrix conditional method of sequences selection. Hence the noise rejection evident in the scheme simulated in section (4.2.1), and the inability of the scheme in section (4.2.2) to demonstrate the same noise rejection, is only restored by an increase in the P_{snr} value. Graph (a) shows a single vector sequence **x,** and graph (b) shows the persistent matrix **X** of zero mean sampled sequences, generated from the chaotic process. Graph (c) shows the orthonormal basis matrix sequences **U**, generated from the matrix **X** which in turn, when multiplied by the diagonal power balancing matrix **P**, give rise to the **Q** matrix which is used for encoding the signals sequence vector **s** shown in graph (e). In this scheme, the references are generated from the **x** vector by normalizing and power balancing it with the power value *p,* to generate the streamed and transmitted **z** vector. When these sequences are resampled at the receiver, they have been contaminated by noise, and assembled into a persistent $\overline{\mathbf{Z}}$ matrix as shown in graph (d). The **Q** matrix is now encoded with a single symbol vector to generate the **s** vector of graph (e), and this is streamed and received in the same way as the **z** vector, to yield the noise contaminated $\overline{\mathbf{s}}$ vector in graph (f). The main advantage to this scheme is, that for each reference sequence and encoded sequence, there are *m* bits of information transmitted, which is one symbol representing *m* bits.

The same set of transmitted and received 'four' bit messages are shown in figures (4.2.3.3) and (4.2.3.4) graphs (a) and (b). Graph (c) demonstrates that, there are errors between the transmitted and the received message sequence with both P_{snr} values, when the time delay is accounted for. The first *m* signals are always in error. as the persistent matrices are not fully populated until four message sequences have been transmitted. This illustrates, as with all communication schemes, a need for some form of preamble before real information can be transmitted. The noise rejection can again be increased by the careful selection of the chaotic sequences outlined in section (3.5). The limits of this scheme are the clearly the noise rejection due to the sequence length *n,* the chaotic sequence conditional selection and the dimension chosen for *m*, which is investigated and optimally selected in chapter 5.

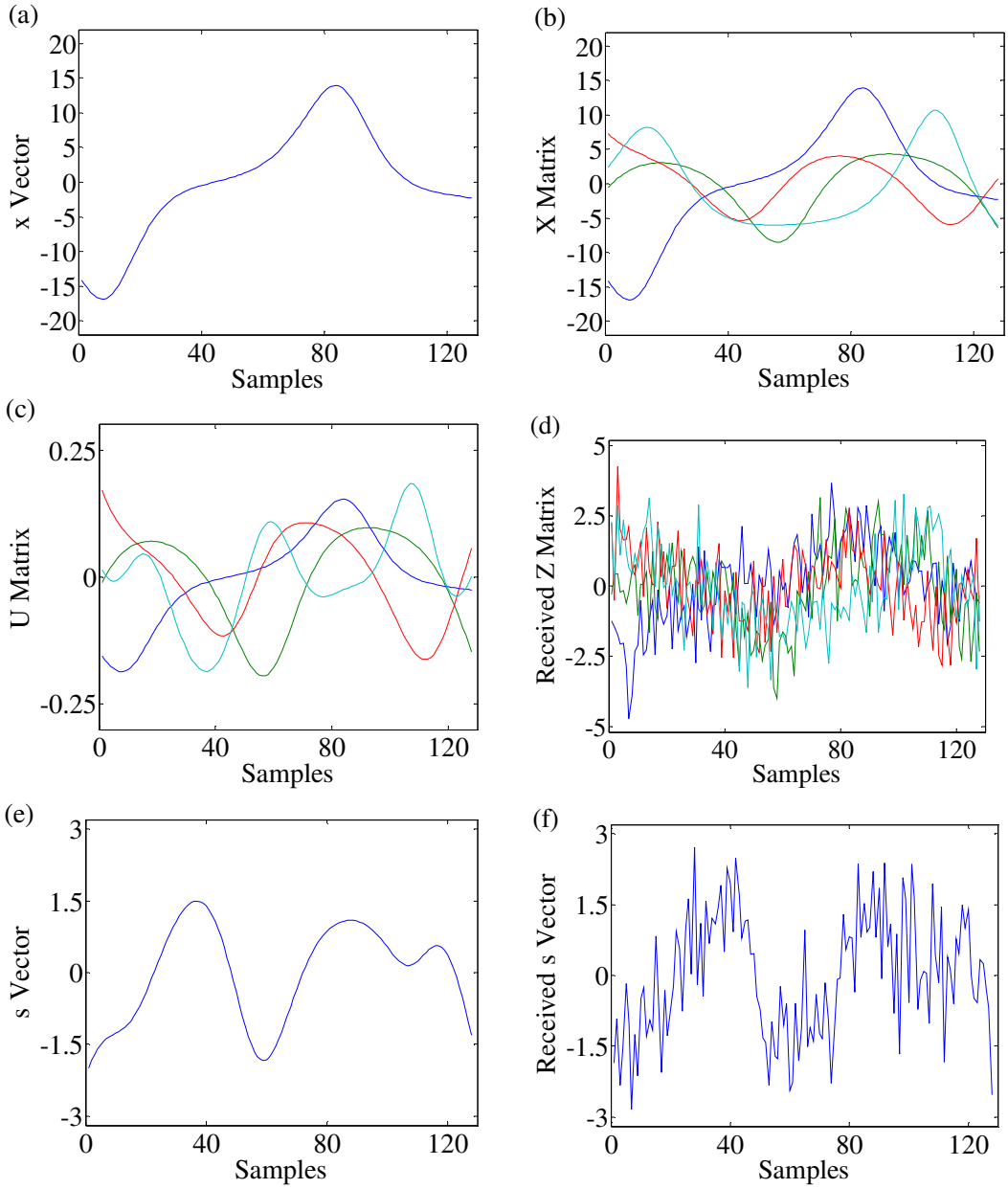


Figure 4.2.3.1 Indirect Persistent 'x' Scheme System Transmission Signals

n = 128, m = 4 and Power of Signal to Noise Ratio = 1.0

(a) Transmitter zero mean chaotic sequences **x**

(b) Persistent chaotic sequences **X**

(c) Generated orthogonal reference sequences **U**

(d) Received power balanced reference sequences $\overline{\mathbf{Z}}$

(e) Transmitted encoded signal sequence **s**

(f) Received encoded signal sequence $\overline{\mathbf{s}}$

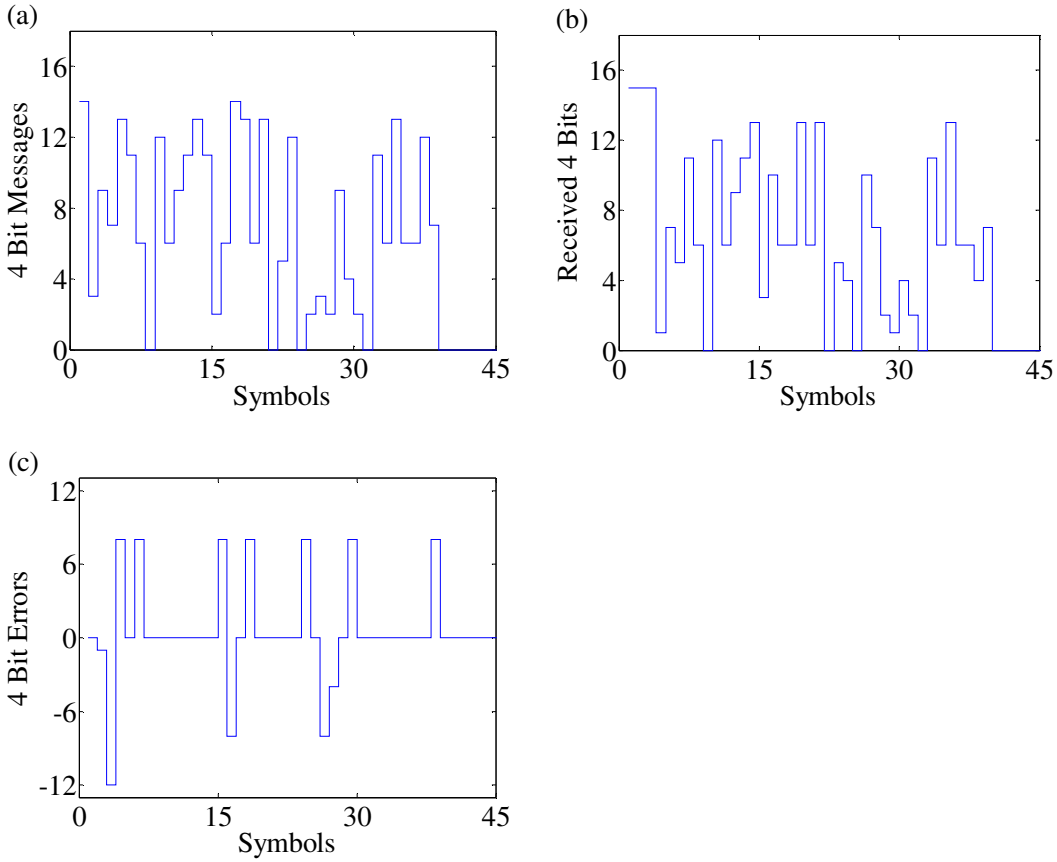


Figure 4.2.3.2 Indirect Persistent ‘x’ Scheme System Message Transmissions

n = 128, m = 4 and Power of Signal to Noise Ratio = 1.0

(a) Transmitted 4 bit message for encoding

(b) Received decoded 4 bit message

(c) Transmitted/Received 4 bit message delayed error

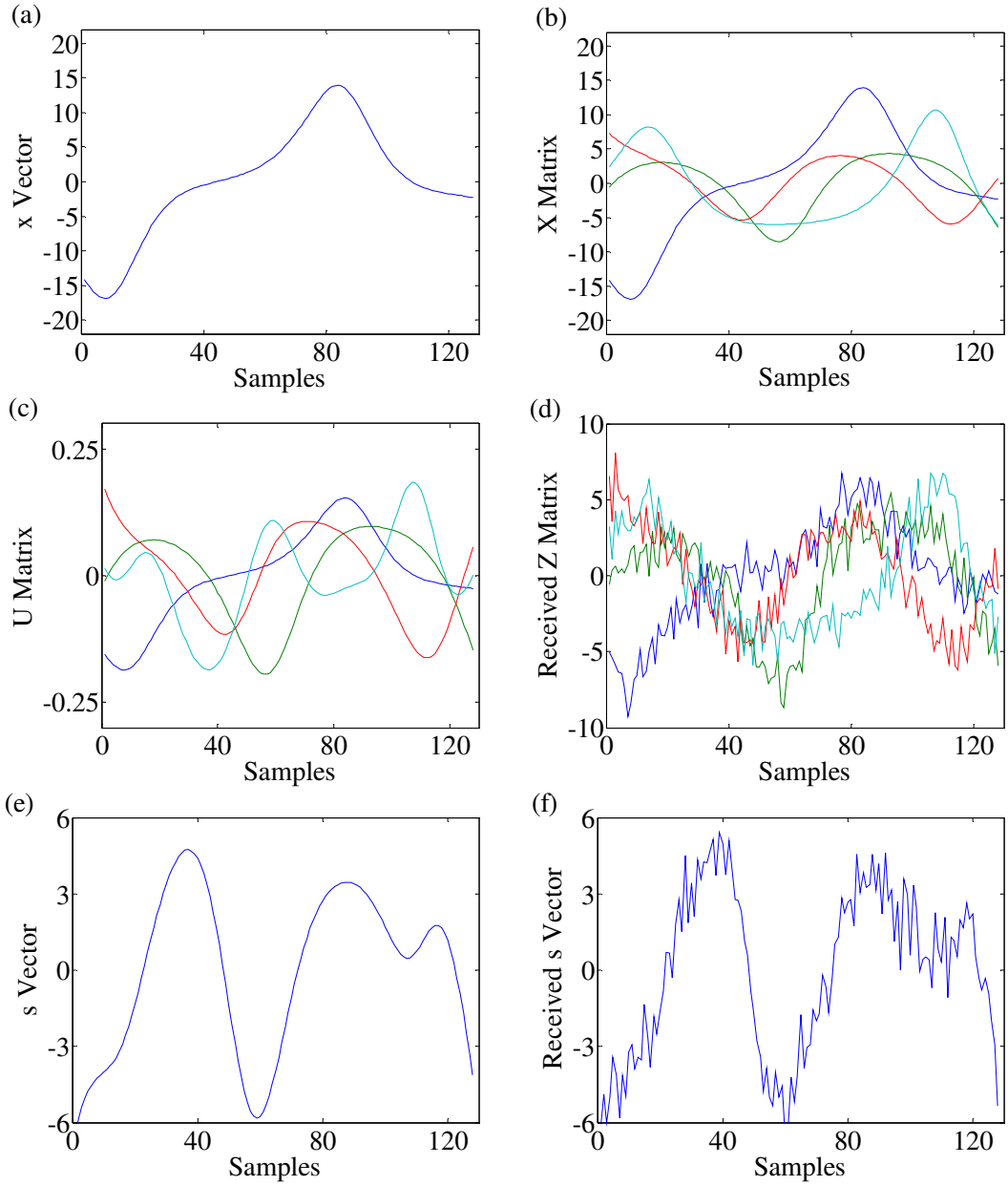


Figure 4.2.3.3 Indirect Persistent 'x' Scheme System Transmission Signals

n = 128, m = 4 and Power of Signal to Noise Ratio = 10.0

(a) Transmitter zero mean chaotic sequences **x**

(b) Persistent chaotic sequences **X**

(c) Generated orthogonal reference sequences **U**

(d) Received power balanced reference sequences $\overline{\mathbf{Z}}$

(e) Transmitted encoded signal sequence **s**

(f) Received encoded signal sequence $\overline{\mathbf{s}}$

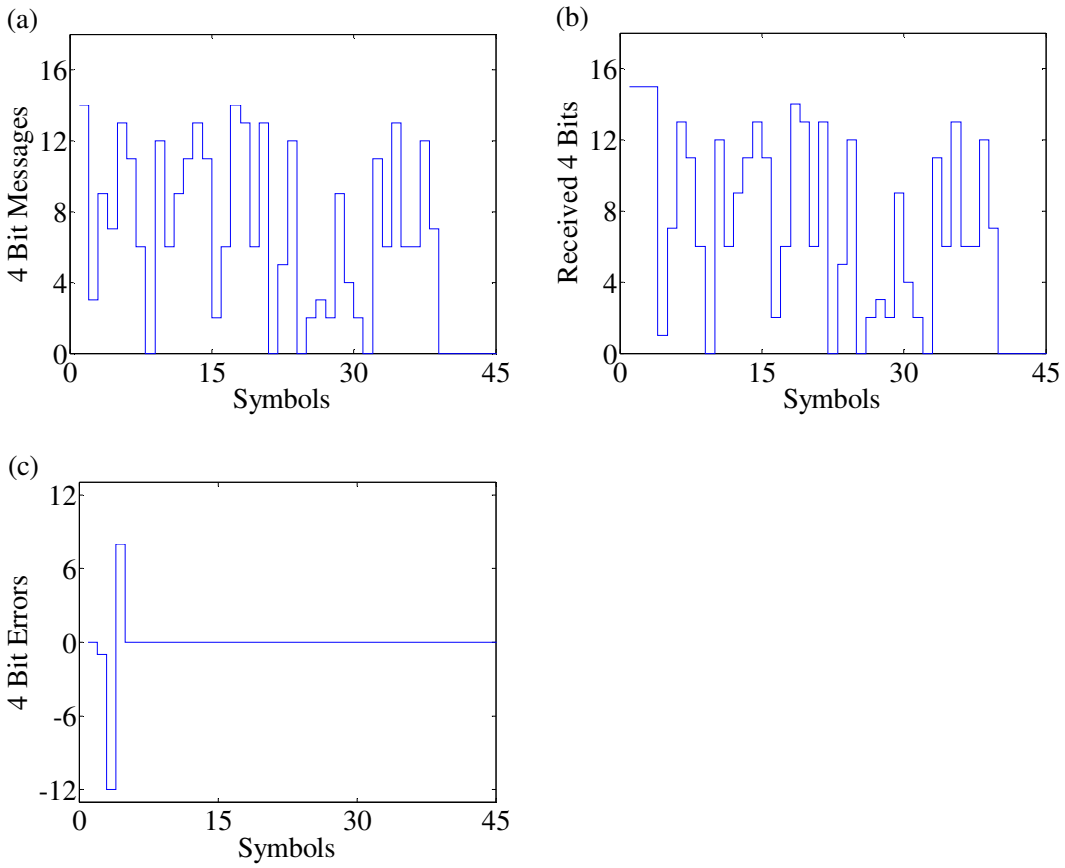


Figure 4.2.3.4 Indirect Persistent ‘x’ Scheme System Message Transmissions

n = 128, m = 4 and Power of Signal to Noise Ratio = 10.0

(a) Transmitted 4 bit message for encoding

(b) Received decoded 4 bit message

(c) Transmitted/Received 4 bit message delayed error

4.3 BER Simulations

4.3.1 Direct '*m* Symbol' 'U' Scheme

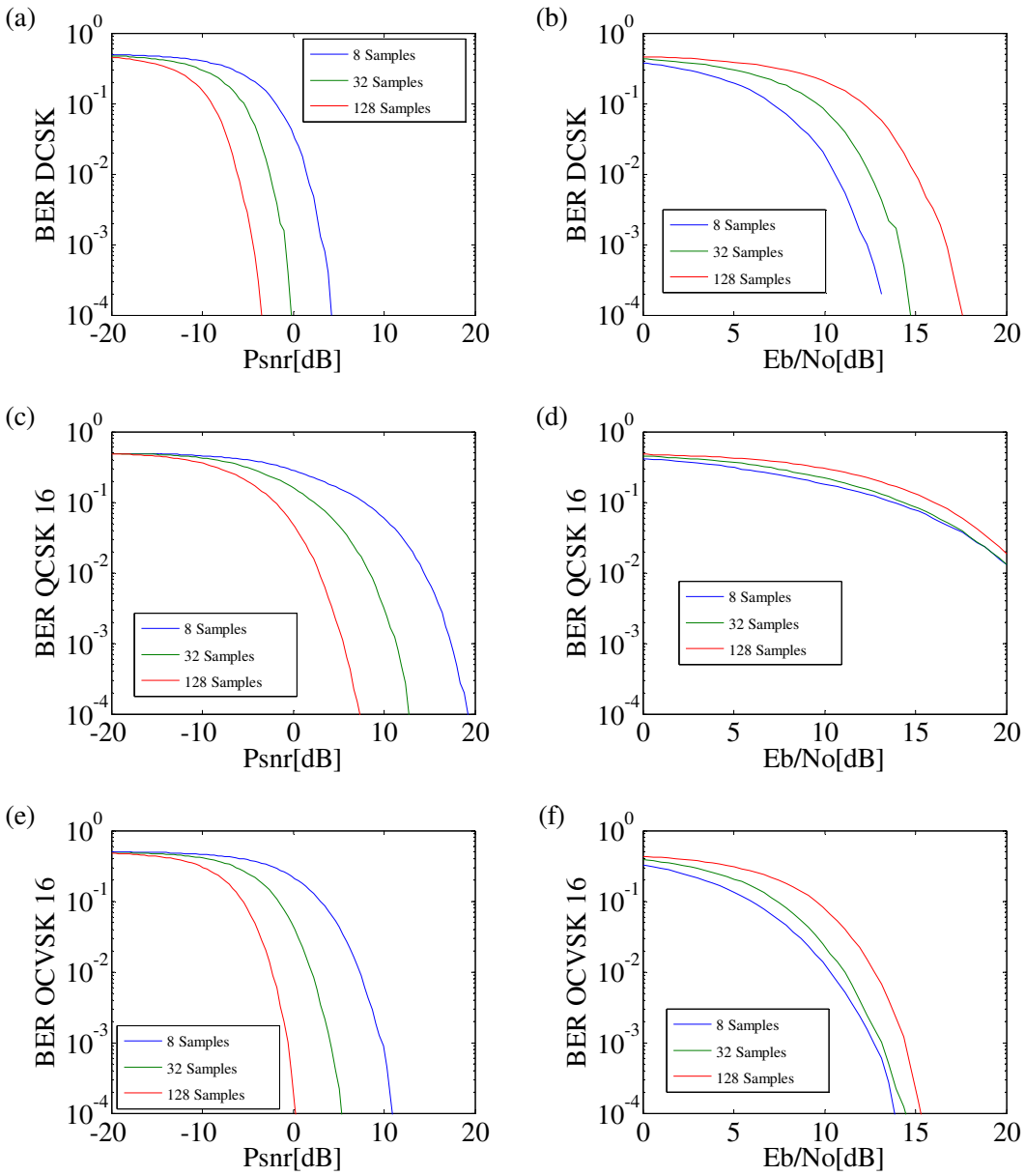


Figure 4.3.1.1 Direct '*m*' Symbol 'U' Scheme

BER v P_{snr} and $\frac{E_b}{N_0}$ Comparison Plot for $n \in [8,32,128]$ samples

(a) and (b) DCSK : BER versus P_{snr} and $\frac{E_b}{N_0}$

(c) and (d) QCSK 16 Symbol Constellation : BER versus P_{snr} and $\frac{E_b}{N_0}$

(e) and (f) OCVSK 16 : BER versus P_{snr} and $\frac{E_b}{N_0}$

For this scheme the **U** reference matrix is orthogonal. The DCSK BER of figure (4.3.1.1) (a) and (b) show better rates than the QCSK 16 examples of graphs (c) and (d). The OCVSK 16 example in graphs (e) and (f) clearly out perform the QCSK 16 scheme and as figure (4.3.1.2) demonstrates it has an equivalent BER to the DCSK example when the data rate is taken into account.

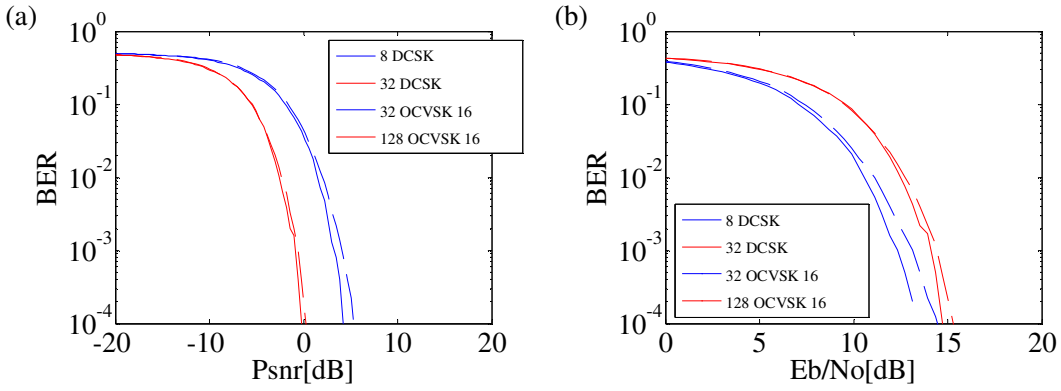


Figure 4.3.1.2 Direct '*m*' Symbol 'U' Scheme

Comparison of BER versus P_{snr} and $\frac{E_b}{N_0}$ for DCSK and OCVSK 16 Scheme showing that error rates are equivalent.

4.3.2 Indirect '*m* Symbol' 'X' Scheme and Indirect Persistent 'x' Scheme

The following three sections show BER graphs for a number of different characteristic **W** matrices. The **W** matrix essentially is a measure of how non orthogonal the **Z** matrix reference signal sequences are. The case where $\mathbf{W} = \mathbf{I}_m$ is the same as that of section (4.3.1) where the received reference matrix is orthogonal and $\mathbf{W}^T\mathbf{W} = \mathbf{I}_m$. All forms of the W matrix are upper triangular, which is a consequence of, the characteristic of the Gram-Schmidt orthonormalization process of derivation.

4.3.2.1 **W** Matrix Case 'A'

In this case the **W** matrix is given by

$$\mathbf{W} = \begin{bmatrix} 1 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{4}} \\ 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{4}} \\ 0 & 0 & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{4}} \\ 0 & 0 & 0 & \frac{1}{\sqrt{4}} \end{bmatrix} \tag{4.3.2.1.1}$$

$$\mathbf{W}^T\mathbf{W} = \begin{bmatrix} 1 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{3}} & \frac{1}{2} \\ \frac{1}{\sqrt{2}} & 1 & \frac{\sqrt{2}}{\sqrt{3}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{3}} & \frac{\sqrt{2}}{\sqrt{3}} & 1 & \frac{\sqrt{3}}{2} \\ \frac{1}{2} & \frac{1}{\sqrt{2}} & \frac{\sqrt{3}}{2} & 1 \end{bmatrix} \tag{4.3.2.1.2}$$

The non-orthogonal nature of this **W** matrix represents a set of signals, with a power of unity, each oriented at an angle of $\frac{\pi}{4}$ radians to its immediate predecessor. It represents a banded orthogonality shown by the $\mathbf{W}^T\mathbf{W}$ product of equation (4.3.2.1.2). This gives a greater span of BER shown in figure (4.3.2.1.1) (e) and (f) for P_{snr} over that of the orthogonal equivalent in figure (4.3.1.1), and illustrates that the banded non-orthogonality characteristic can be overcome by an increase in the number of samples *n*. The QCSK 16 examples of figure (4.3.2.1.1) (c) and (d) use only the first two columns and rows of the **W** matrix. Consequently, the effects of non-orthogonality become more apparent.

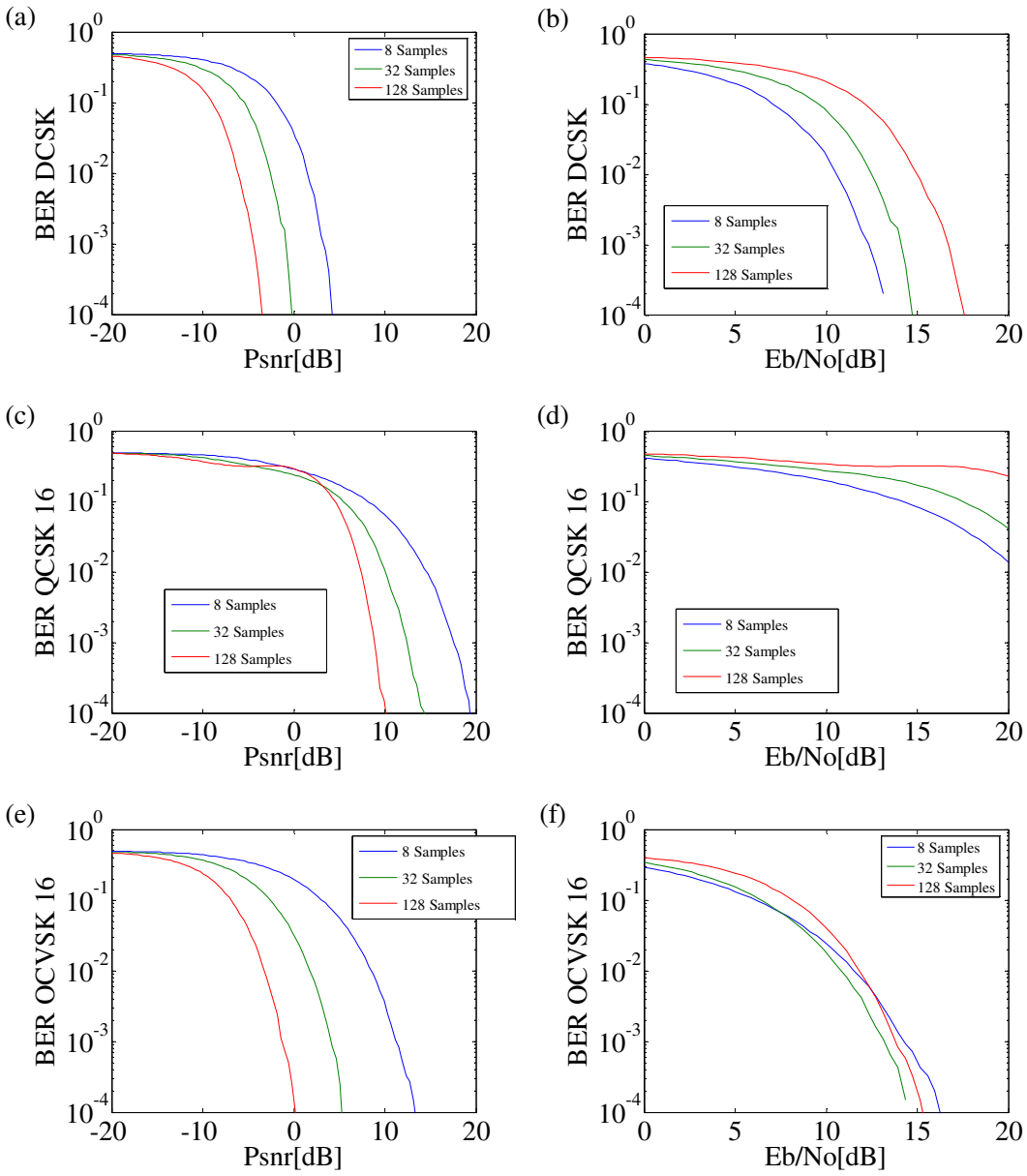


Figure 4.3.2.1.1

BER v P_{snr} and $\frac{E_b}{N_0}$ W Scheme 'A' : Comparison Plot for $n \in [8,32,128]$ samples

(a) and (b) DCSK : BER versus P_{snr} and $\frac{E_b}{N_0}$

(c) and (d) QCSK 16 Symbol Constellation : BER versus P_{snr} and $\frac{E_b}{N_0}$

(e) and (f) OCVSK 16 : BER versus P_{snr} and $\frac{E_b}{N_0}$

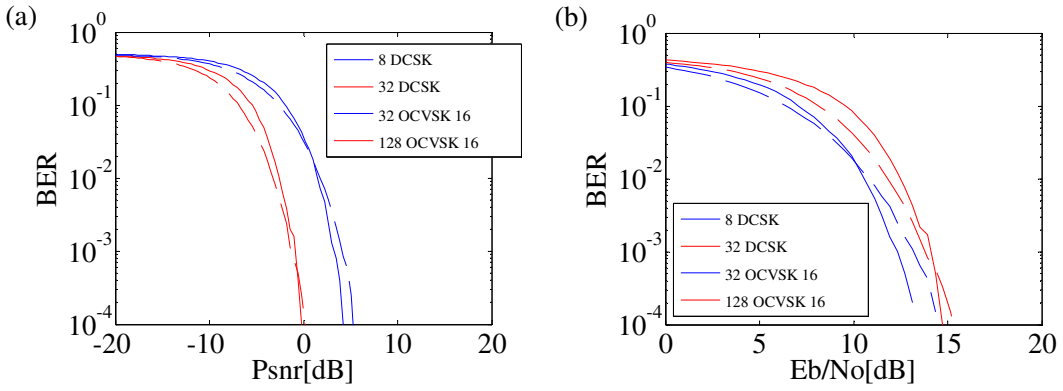


Figure 4.3.2.1.2

W Scheme 'A' Comparison of BER versus P_{snr} and $\frac{E_b}{N_0}$ for DCSK and OCVSK 16 Scheme showing that the error rates are beginning to diverge.

4.3.2.2 **W** Matrix Case 'B'

In this case the **W** matrix is given by

$$\mathbf{W} = \begin{bmatrix} 1.0000 & 0.1481 & -0.1345 & 0.1205 \\ 0 & 0.9890 & 0.8609 & 0.7699 \\ 0 & 0 & 0.4907 & 0.3383 \\ 0 & 0 & 0 & 0.5276 \end{bmatrix} \qquad (4.3.2.2.1)$$

$$\mathbf{W}^T\mathbf{W} = \begin{bmatrix} 1.0000 & 0.1481 & -0.1345 & 0.1205 \\ 0.1481 & 1.0000 & 0.8315 & 0.7793 \\ -0.1345 & 0.8315 & 1.0000 & 0.8126 \\ 0.1205 & 0.7793 & 0.8126 & 1.0000 \end{bmatrix} \qquad (4.3.2.2.2)$$

The **W** matrix, for this example, is derived from the first set of **X** matrix reference data, used by the simulation run examples of section (4.2). It represents a different banded orthogonality than that shown in the last example illustrated by the $\mathbf{W}^T\mathbf{W}$ product of equation (4.3.2.2.2). This again, gives a greater span of BER shown in figure (4.3.2.2.1) (e) and (f) for P_{snr} over that of the orthogonal equivalent in figure (4.3.1.1). And again, this illustrates that the banded non-orthogonality characteristic can be overcome by an increase in the number of samples *n*. The QCSK 16 examples of figure (4.3.2.2.1) (c) and (d), use only the first two columns and rows of the **W** matrix. In this case, the non-

diagonal elements are small in comparison to the diagonal unity terms and consequently the effects of non-orthogonality are not particularly apparent.

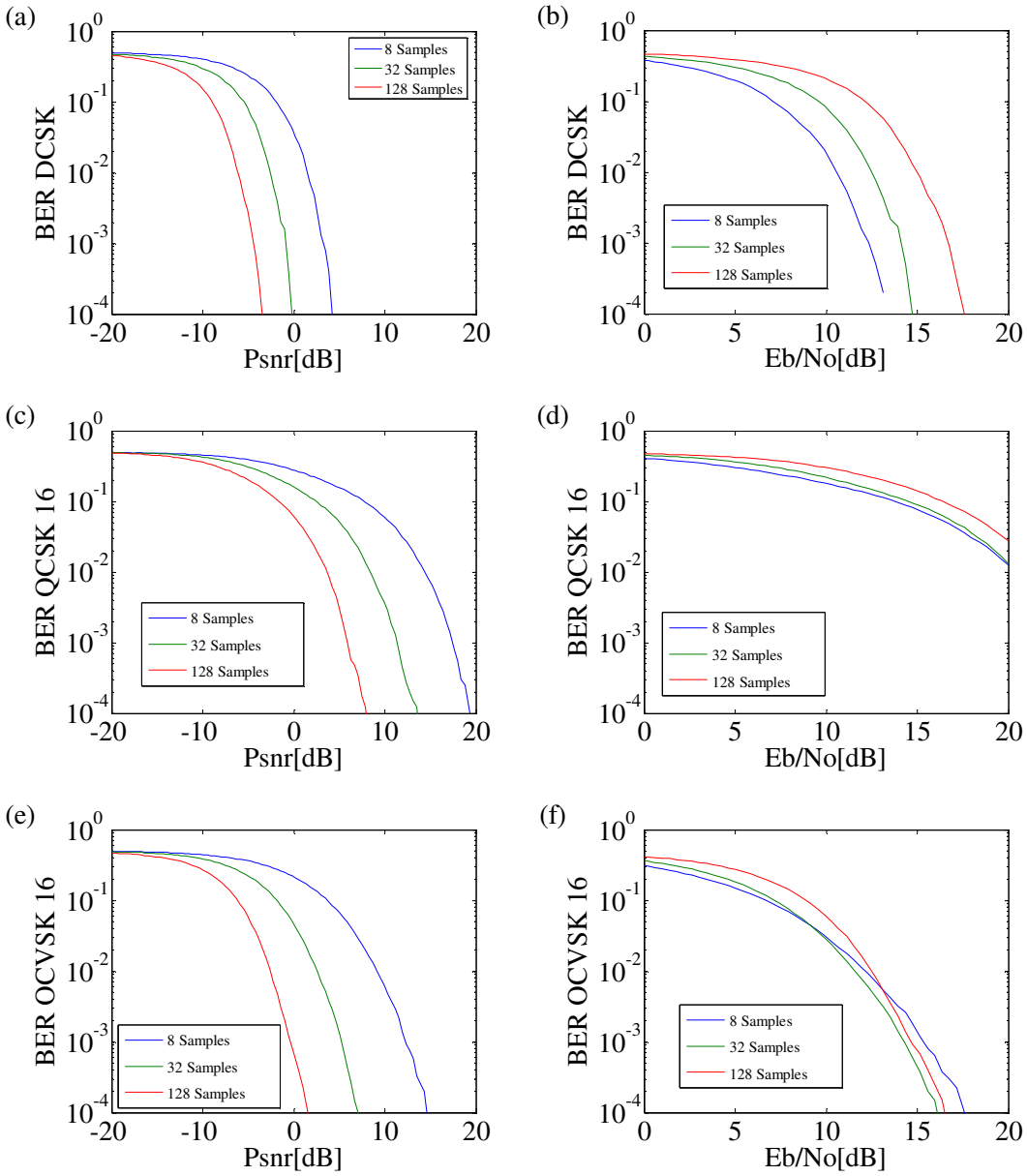


Figure 4.3.2.2.1

BER v P_{snr} and $\frac{E_b}{N_0}$ W Scheme 'B' : Comparison Plot for $n \in [8,32,128]$ samples

(a) and (b) DCSK : BER versus P_{snr} and $\frac{E_b}{N_0}$

(c) and (d) QCSK 16 Symbol Constellation : BER versus P_{snr} and $\frac{E_b}{N_0}$

(e) and (f) OCVSK 16 : BER versus P_{snr} and $\frac{E_b}{N_0}$

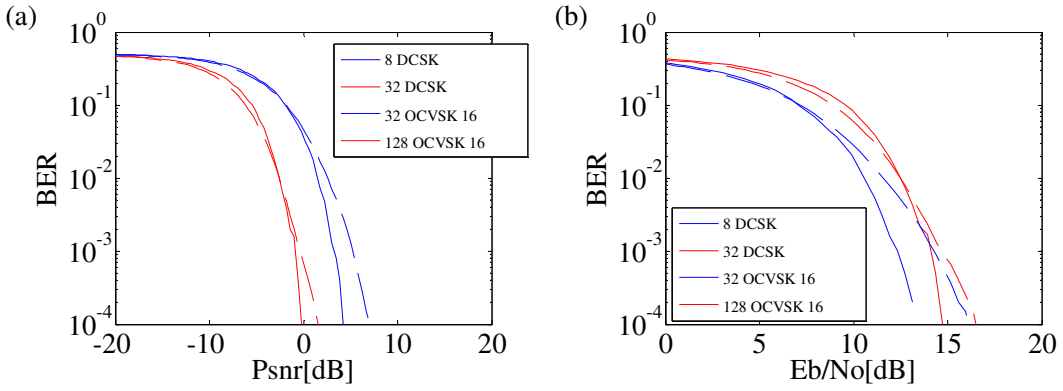


Figure 4.3.2.2.2

W Scheme 'B' Comparison of BER versus P_{snr} and $\frac{E_b}{N_0}$ for DCSK and OCVSK 16 Scheme showing a small divergence due to non-orthogonal reference signals.

4.3.2.3 **W** Matrix Case 'C'

In this case the **W** matrix is given by

$$\mathbf{W} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.3.2.3.1}$$

$$\mathbf{W}^T\mathbf{W} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \tag{4.3.2.3.2}$$

The above, as an example, is designed to illustrate a non banded **W** matrix. It represents a set of signals which have all the columns of the **U** matrix the same and are completely non-orthogonal, this is shown by the $\mathbf{W}^T\mathbf{W}$ product in equation (4.3.2.3.2). Here the span of BER, shown in figure (4.3.2.3.1) (e) and (f) for P_{snr}, is infinite. This implies that no information is transmitted at any value of P_{snr}. The QCSK 16 examples of figure (4.3.2.3.1) (c) and (d) use only the first two columns and rows of the **W** matrix. In both cases, there is an improvement in the BER around the P_{snr} values 0 dB to 10 dB. This is clearly due, to the interaction of the signal noise distributions under the estimation operations and does not represent any real information transmission.

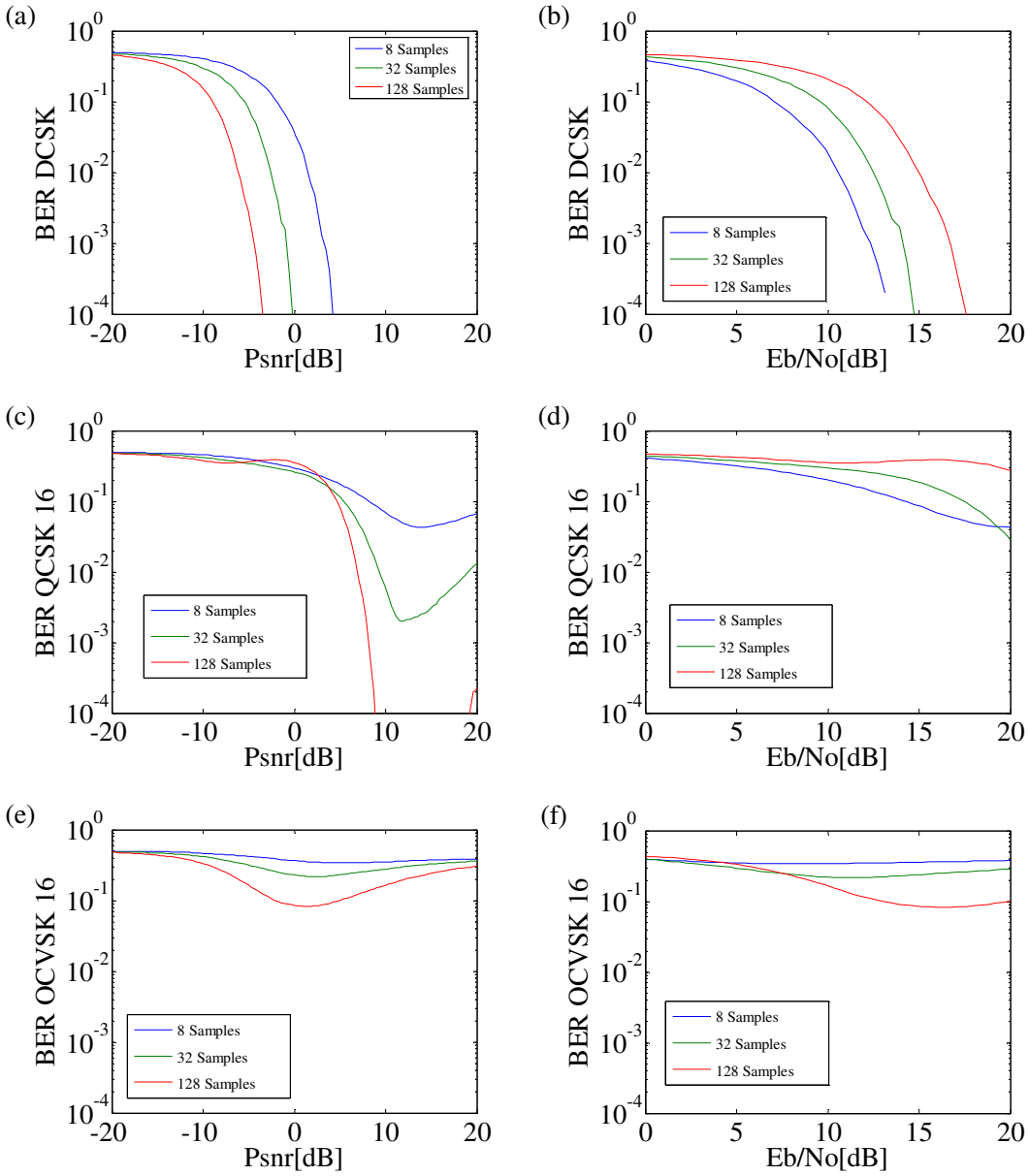


Figure 4.3.2.3.1

BER v P_{snr} and $\frac{E_b}{N_0}$ W Scheme 'C' : Comparison Plot for $n \in [8,32,128]$ samples

(a) and (b) DCSK : BER versus P_{snr} and $\frac{E_b}{N_0}$

(c) and (d) QCSK 16 Symbol Constellation : BER versus P_{snr} and $\frac{E_b}{N_0}$

(e) and (f) OCVSK 16 : BER versus P_{snr} and $\frac{E_b}{N_0}$

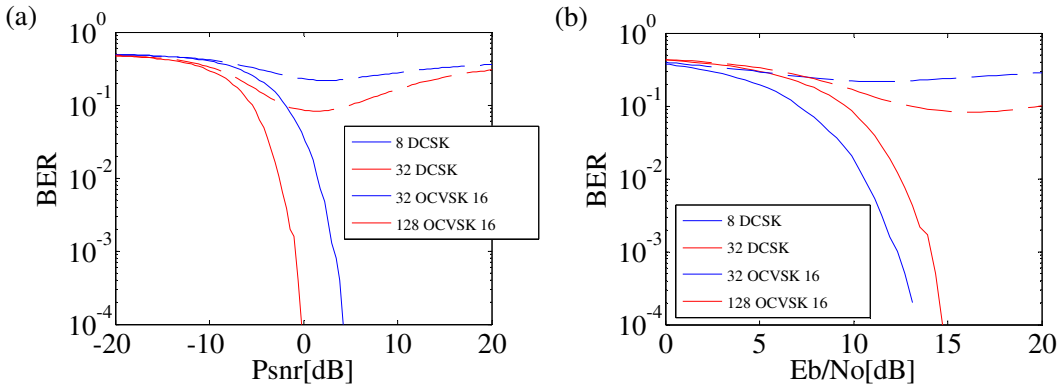


Figure 4.3.2.3.2

W Scheme 'C' Comparison of BER versus P_{snr} and $\frac{E_b}{N_0}$ for DCSK and OCVSK 16

Scheme showing the large divergence due to significantly non-orthogonal reference signals

4.4 Summary

For the different system architectures presented in chapter 3, a set of simulation results has been presented showing their various performance characteristics. The transmission and BER simulations demonstrate the value of OCVSK methods utilizing orthogonal signal sets over the extended M-ary type communications schemes.

Chapter 5

Optimal Dimensionality

5.1 Introduction

In the previous chapters, it would seem that by constantly extending the dimension of the suggested persistent type scheme in section (3.6.3), the transmission efficiency could be endlessly increased. The only limiting factor would be the preamble of pre-loading the orthogonal signal sets before useful transmission could take place. Another limiting factor would be, a limit imposed by the method used to generate the orthogonal sequence namely the Gram-Schmidt method. If the number of samples in each signal sequence is *n,* then the number of orthogonal signals that can be generated is also *n,* which if sufficiently large this is not really a limiting factor. However, experimentally, when the case studies were being undertaken, benchmark transmission times weighed against message transmission rates were empirically found to favour a value of around seven or eight as the best choice of dimensionality *m*. Presented, is a mathematical approach to determining why this might be the case. It considers the advantage over the two dimensional M-ary type constellations, and the manner in which the surface layer of hyperspheres; where the primary or most simple constellations of this method exist, diminishes as a proportion of the total hypervolume as the dimension *m* increases.

5.2 Volumetric Considerations

In QCSK, the constellations are represented on a unit circle which has an area of π. In a hyperspace context this is a volume of order $m=2$ so $V(2)=\pi$. As we proceed from dimension $m=2$ to dimension $m=3$, the two dimensional volume occupies a zero volume in the third dimensional space, but it can be integrated over the third orthogonal dimension, to yield the familiar volumetric result $V(3)=\frac{4}{3}\pi$. The third dimension can be considered as the space potentially occupied by the third 'bit' of our message, and if we consider that the potential volume it can occupy lies within a hyperspherical shell with radii of $r\pm\beta r$, where the β factor represents a banding of the noise variance i.e. $\sigma^2=(\beta r)^2$; then we can derive an optimal value for dimensionality, compared with the $m=2$ dimensional value. This is described in section (5.3).

The volume of an *m* dimensional hypersphere of radius *r* is derived in appendix (D) and can be expressed using the gamma $\Gamma(z)$ function as

$$V_m(r)=\frac{\pi^{\frac{m}{2}}}{\Gamma\left(\frac{m}{2}+1\right)}\cdot r^m \tag{5.2.1}$$

Now consider a ratio R_m of a hyperspherical shell of radii $r\pm\sigma$ divided by the volume of the hypercube containing the maximum radius that is $2(r+\sigma)$

$$R_m=\frac{V_m(r+\sigma)-V_m(r-\sigma)}{\left(2(r+\sigma)\right)^m} \tag{5.2.2}$$

where $\sigma=\beta r$ and $\beta\in(0,1]$

This can be considered as a measure of the efficiency of the volumetric concentration near the surface of the *m* dimensional shell.

5.3 Comparative Function

Before considering a more reasoned approach to determining the optimal dimension to choose for the OCVSK scheme, it is worth considering how the distance between close pairs of symbolic points in the constellations of the *m* dimensional space, compares with the separation in the two dimensional M-ary type constellations. If all *m* dimensional schemes are considered as simple; that is that each dimension has only a positive and a negative value, then the distance between each successive symbolic point on a unit hypersphere, with each point placed at an equal displacement along each axis illustrated in figure (5.3.1) (a), is given by

$$m\left(\frac{d}{2}\right)^2 = 1 \tag{5.3.1}$$

yielding

$$d = \frac{2}{\sqrt{m}} \tag{5.3.2}$$

And likewise, the separation of each symbolic value for the M-Ary scheme illustrated in figure (5.3.1) (b) is given by

$$d^2 = \left\{1 - \cos\left(\frac{\pi}{2^{m-1}}\right)\right\}^2 + \sin^2\left(\frac{\pi}{2^{m-1}}\right) \tag{5.3.3}$$

$$d = \sqrt{2}\left\{1 - \cos\left(\frac{\pi}{2^{m-1}}\right)\right\}^{\frac{1}{2}} \tag{5.3.4}$$

These functions for a continuously varying *m* are shown in figure (5.3.2) as a comparative plot, and it clearly shows, that at $m = 1$ and $m = 2$ they are equal. This corresponds to the BPSK and the Quadrature constellations respectively. At values of $m > 2$ the OCVSK distance is always greater, which implies that the noise rejection is always better. The value of *m* cannot exist for non integer values but the plot serves to show the nature of the functions.

This is a good measure, but does not take into account the volumetric to available surface effects described in [2][50], or any increase in computational loading. Firstly, consider a comparison of the volumetric efficiency measure for an *m* dimensional hyperspace R_m and a similar measure of the $m = 2$ dimensional hyperspace R_2.

Further, that for each integer increase in dimension *m*, the effective space in the $m = 2$ dimension is halved because the number of symbols in the constellation has doubled. Derive a measure C_m formed from R_m and R_2 that accounts for the dimensional halving effect and is normalized for the $m = 2$ case. The following results

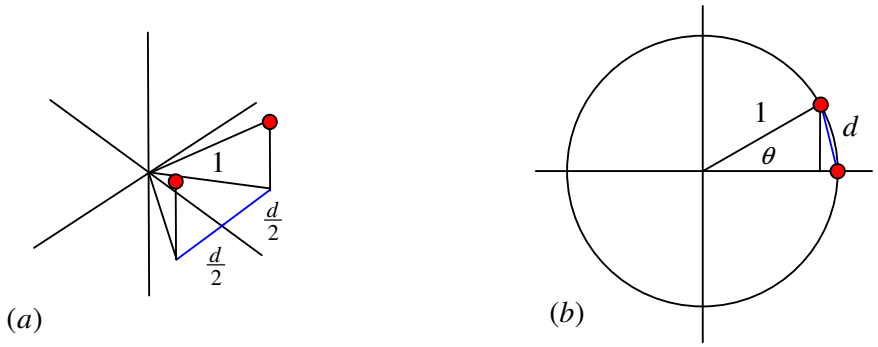


Figure 5.3.1

Comparative Positioning of M-Ary Constellation and OCVSK

Inter-symbolic Distance

(a) Successive Symbolic Points of an *m* Dimensional Hypersphere with $m = 3$ and

(b) The comparable M-Ary Distance with $\theta = \frac{\pi}{2^{m-1}}$

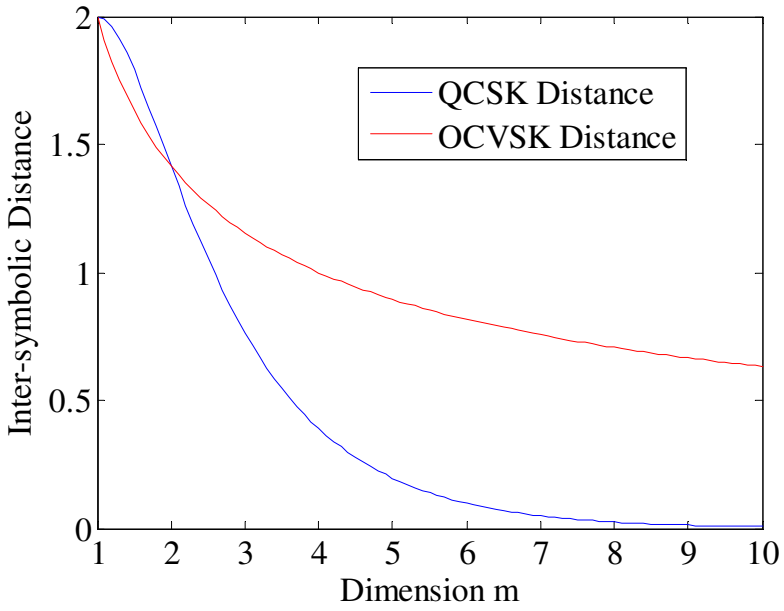


Figure 5.3.2

Comparative Plot of M-Ary Constellation and OCVSK

Inter-symbolic Distance

$$C_m = \frac{R_m}{\left(\frac{R_2}{2^{m-2}}\right)} = 2^{m-2}\frac{R_m}{R_2} \qquad \Rightarrow C_2 = 1 \tag{5.3.5}$$

Substituting equation (5.2.1) into (5.2.2) and substituting $\sigma = \beta r$ yields

$$R_m = \frac{\pi^{\frac{m}{2}}}{2^m \Gamma\left(1+\frac{m}{2}\right)}\left\{1 - \frac{(1-\beta)^m}{(1+\beta)^m}\right\} \tag{5.3.6}$$

And the case for $m = 2$ can be expressed as

$$R_2 = \frac{\pi}{4}\left\{1 - \frac{(1-\beta)^2}{(1+\beta)^2}\right\} \tag{5.3.7}$$

finally substitution of (5.3.6) and (5.3.7) into (5.3.5) produces an expression for the comparative function in terms of the dimension *m* and the noise banding β

$$C_m = \frac{\pi^{\frac{m}{2}-1}}{\Gamma\left(1+\frac{m}{2}\right)} \cdot \frac{\left\{1 - \frac{(1-\beta)^m}{(1+\beta)^m}\right\}}{\left\{1 - \frac{(1-\beta)^2}{(1+\beta)^2}\right\}} \tag{5.3.8}$$

Clearly, as β approaches zero the numerator and denominator of equation (5.3.8) approach zero simultaneously. This is the optimal case where the noise component of the signal tends towards zero. Using L'hopital's limiting theorem on both numerator and denominator functions of β the optimal comparative ratio in terms of the dimension *m* becomes

$$C_m = \frac{\pi^{\frac{m}{2}-1}}{\Gamma\left(1+\frac{m}{2}\right)} \cdot \frac{m}{2} \tag{5.3.9}$$

The maxima of this function can be expressed in terms of the digamma $\varphi(z)$ function

$$\left(\varphi\left(1+\frac{m}{2}\right)-\ln(\pi)\right)=\frac{2}{m} \tag{5.3.10}$$

This equation has no trivial solution, but the optimal solution occurs at a value of $m=7.2569$ a graph of equation (5.3.9) is shown in Figure (5.3.3). The derivation of this result can be found in appendix (D.2)

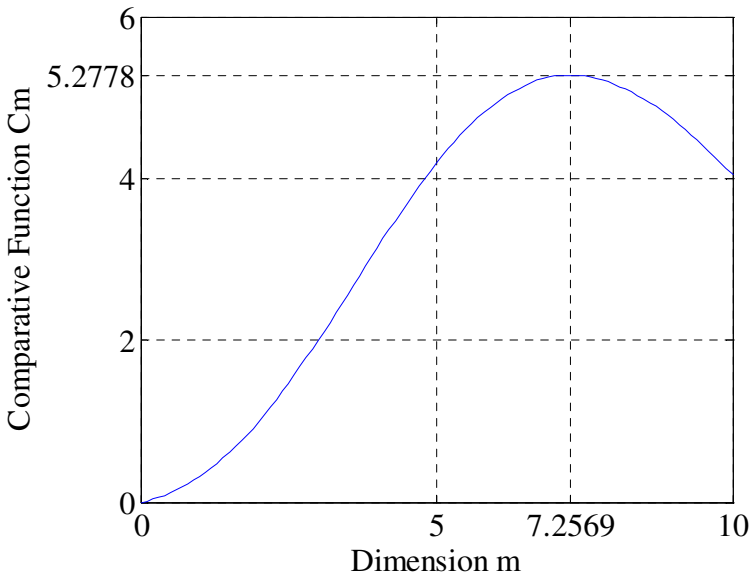


Figure 5.3.3

Optimal Dimensional Value derived from Volumetric Considerations

5.4 Dimensional Comparative Simulation

Consider a set of simulated Bit Error Rate (BER) ratios, between Orthogonal Chaotic Vector Shift Keying (OCVSK) and M-ary versions of Quadrature Chaos Shift Keying (QCSK), at various dimensions m between $m=3$ and $m=10$. Now plot these against P_{snr}, the power of the signal to noise ratio, for sequences lengths of $n=128$. This is shown in figure (5.4.1).

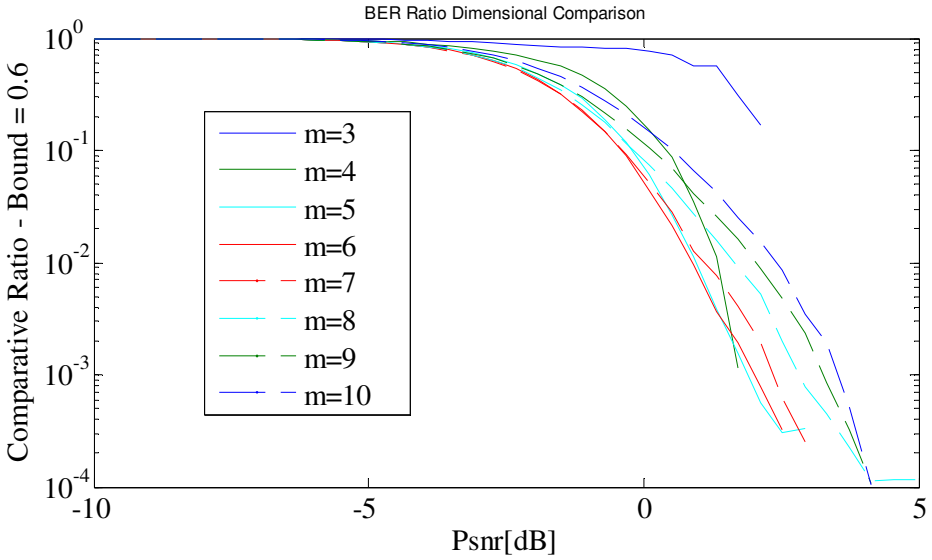


Figure 5.4.1

BER Ratio Dimensional Comparison

It clearly shows that as the dimension m moves through the $m=7$ value, the improvement in BER Ratio stops and a subsequent increase in the dimension, increases the BER Ratio at higher P_{snr} values. It is also clear, that the benefit of using the higher dimensional architectures is always an improvement over two dimensional quadrature type schemes, as the fall off in the comparative function demonstrates. This simulation is evaluated, with the estimate vector bounded between a magnitude of 0.4 and 1.6 for a nominal value of 1.0. This is a wide band but clearly shows the dimensional effect. This simulation run took several hours to complete, and finer bounding of the solutions requires an exponential increase in computation time, to achieve reasonably smooth curves. A listing of the Matlab code required to generate this plot is given in appendix (D).

5.5 Summary

In this chapter, a conjecture has been made as to why the empirical optimal dimension of the OCVSK scheme is a value near seven. This has been approached by considering a volume ratio of where the *m*-dimensional symbols exist in the communication space and the maximal volume available for them to exist in. This has been derived as a method of illustrating how the inter-symbolic distances vary with dimension. From this, a comparative function between the *m*-dimensional scheme and an equivalent two dimensional has been derived and shown to have a maximum at approximately seven. In addition, it has been shown by simulations with varying noise conditions, that the OCVSK scheme always has improved performance over any two dimensional M-ary scheme and that the optimal value is approximately seven.

Chapter 6

Conclusions

This thesis has presented a new method of improving the transmission efficiency of chaotic communication schemes. Called Orthogonal Chaotic Vector Shift Keying, the new method is based on orthogonal chaotic signal sequences. It has been shown that under this multilevel digital communication scheme, significant improvements over some existing methods, for example some typical M-ary type two-dimensional quadrature schemes, have been achieved. Consequently, is has been demonstrated, that the information transmitted per unit time is dependent on the dimensions adopted under the new method.

A novel method of transmission and reception has been introduced whose only overhead is an extended preamble time. The architectural requirements of this novel scheme, have been investigated, using a number of different characterizable potential signal structures. In addition, ways of improving the performance of these structural schemes, by the careful choice of chaotic system and the conditional rejection of signal sequences, has been demonstrated. The cyclic transmission efficiency is increased and is scaleable with the dimension *m,* without any noise or time penalties.

The robustness and noise rejection properties of these multilevel schemes have been demonstrated. Performance improvement comparisons, over typical M-ary type two dimensional quadrature schemes, have been presented. The Bit Error Rates of the new schemes have been shown to be equivalent to that of Differential Chaos Shift Keying.

The nature of the methods employed in the schemes proposed has also potentially increased the secure nature of the communications over a given channel.

Also in this thesis, a method has been demonstrated to characterize the nature of non-linear processing elements of complex systems; and in turn, a simple modelling and evaluation method, to determine the Bit Error Rates of these systems, has been derived and the simulated results have been presented.

Investigation into the "optimal" dimensionality of the new method has shown that, the optimal value for the scheme's dimension is approximately seven. This is for the given set of assumptions. For dimensions greater than seven, the improvement decreases, but the new scheme always performs better than any two dimensional quadrature schemes. The improvement decrease varies as a function of the relationship between the volume of the communication space and the surface area of the hypersphere, where the symbolic constellations lie. In addition, with higher dimensions, the computational complexity increases approximately as the third order of the dimension.

Further suggested research work includes:

(1) Constellation Distribution on the Message Hypersphere

In the method presented in this thesis, the only type of constellation configuration considered is the simple or bimodal one. That is that in each dimension there are two symbols one with a positive value in the direction of the dimension axis, and the other, with a negative value. As with QCSK and M-ary types of two-dimensional constellations, this could be extended to higher orders, and the effect on transmission efficiency and BER investigated. Work in this area could lead to finding an optimal configuration for any given dimension *m*, and clarify the exact nature of optimality for any configuration and dimension.

(2) Eigenvalue Symbol Information and Chaotic Orthogonal Signals

For the method of Orthogonal Chaotic Vector Shift Keying, the orthogonalization of the sampled matrices was achieved by using the Gram-Schmidt method. However, another method of orthogonalization could be used which has some other properties usable in future research. The method of Singular Valued Decomposition increases the complexity of derivation but yields, for each set of sampled signals, a set of eigenvalues associated with the matrix produced by the inner product of the sampled signal set. These eigenvalues are very robust in a positional sense, in the presence of high levels of noise and could be used as message symbol sets. The problem then becomes reversed in the sense that for a given set of eigenvalues, representing a complex multilevel message, the problem of how to directly generate a set of orthogonal chaotic signals needs to be researched.

(3) Hyperchaotic Generating Systems

The pseudo cyclic behaviour of the Lorenz system, chosen in the thesis, has two distinct disadvantages. Firstly, the cyclical signals although not constant, are detectable and secondly, the same cyclical nature can potentially result in a high number of signal sequence rejections, due to insufficient independence of the columns of the sampled matrix. These two problems could be overcome by employing hyperchaotic systems. The research would then be aimed at a decision process for the most appropriate system.

References

[1] Digital Communications Systems
P. Z. Peebles Jr.
Prentice Hall International Inc., New Jersey, USA, 1987

[2] Telecommunications Engineering
J. Dunlop and D. Smith
Van Nostrand Reinhold (UK) Co. Ltd., Berkshire, England, 1984

[3] Information Transmission Modulation and Noise 3rd Edition
M. Schwartz
McGraw-Hill, Maidenhead, 1980

[4] Synchronization in Chaotic Systems
L. K. Pecora and T. L. Carroll
Phys. Rev. Lett. , Volume 64, Issue 8, 1990, Page(s): 821-824

[5] Driving Systems with Chaotic Signals
L. K. Pecora and T. L. Carroll
Phys. Rev. A, Volume 44, Number 4, 1991, Page(s): 2374-2384

[6] Circuit Implementation of Synchronized Chaos with Applications to Communications
K. M. Cuomo and A. V. Oppenheim
Phys. Rev. Lett. , Volume 71, Issue 1, 1993, Page(s): 65-68

[7] Synchronizing Hyperchaotic Volume-Preserving Maps and Circuits
T. L. Carroll and L. M. Pecora
IEEE Transactions on Circuits and Systems I
Fundamental Theory and Applications, Vol. 45, Issue 6, 1998 Page(s): 656-659

[8] Using Cyclostationary Properties of Chaotic Signals for Communications
T. Carroll
IEEE Transactions on Circuits and Systems I
Fundamental Theory and Applications, Vol. 49, Issue 3, 2002 Page(s): 357-362

[9] Implementation of Bidirectional Chaotic Communications Systems Based on Lorenz Circuits
S. Tsay, C. Huang, D. Qiu and W. Chen
Chaos Solitons and Fractals, 20, 2004, Page(s): 567-579

[10] Basic Principles of Direct Chaotic Communications Systems
A. Dmitriev, M Hasler, A. Panas and K. Zakharchenko
Nonlinear Phenomena in Complex Systems, Vol. 4, Issue 1, 2002 Page(s): 1-14

[11] Design of Noncoherent Receiver for Analog Spread Spectrum Communication Based on Chaotic Masking
K. Murali, H. Leung and H. 3
IEEE Transactions on Circuits and Systems I
Fundamental Theory and Applications, Vol. 50, Issue 10, 2003 Page(s): 432-441

[12] The Role of Synchronization in Digital Communications Using Chaos – Part II: Chaotic Modulation and Chaotic Synchronization
G. Kolumbán, M. P. Kennedy and L. O. Chua
IEEE Transactions on Circuits and Systems I
Fundamental Theory and Applications, Vol. 45, Issue 11, 1998 Page(s): 1129-1140

[13] Performance Analysis of Correlation-Based Communication Schemes Utilizing Chaos
M. Sushchik, L. S. Tsimring and A. R. Volkovskii
IEEE Transactions on Circuits and Systems I
Fundamental Theory and Applications, Vol. 47, Issue 12, 2000 Page(s): 1684-1691

[14] On Some Recent Developments of Noise Cleaning Algorithms for Chaotic Signals
Z. J. Jako, G. Kolumbán and H. Dedieu
IEEE Transactions on Circuits and Systems I
Fundamental Theory and Applications, Vol. 47, Issue 9, 2000 Page(s): 1403-1406

[15] Noise Performance of Chaotic Communication Systems
A. Abel, W. Schwarz and M. Gotz
IEEE Transactions on Circuits and Systems I
Fundamental Theory and Applications, Vol. 47, Issue 12, 2000 Page(s): 1726-1732

[16] Performance of Differential Chaos-Shift-Keying Digital Communication Systems over a Multipath Fading Channel with Delay Spread
Y. Xia, C. K. Tse and F. Lau
IEEE Transactions on Circuits and Systems II
Express Briefs, Vol. 51, Issue 12, 2004 Page(s): 680-684

[17] Performance Evaluation of FM-DCSK Modulation in Multipath Environments
M. P. Kennedy, G. Kolumban, G. Kis and Z. Jako
IEEE Transactions on Circuits and Systems I
Fundamental Theory and Applications, Vol. 47, Issue 12, 2000 Page(s): 1702-1711

[18] Optimal Detection of Differential Chaos Shift Keying
T. Schimming and M. Hasler
IEEE Transactions on Circuits and Systems I
Fundamental Theory and Applications, Vol. 47, Issue 12, 2000 Page(s): 1712-1719

[19] Essence and Advantages of FM-DCSK Technique versus Conventional Spreading Spectrum Communication Method
L. Ye, G. Chen and L. Wang
http://www.paper.edu

[20] Analysis and CMOS Implementation of a Chaos-Based Communication System
S. Mandal and S. Banerjee
IEEE Transactions on Circuits and Systems I
Regular Papers, Vol. 51, Issue 9, 2004 Page(s): 1708-1722

[21] Applications of Symbolic Dynamics to Differential Chaos Shift Keying
G. M. Maggio and Z. Galias
IEEE Transactions on Circuits and Systems I
Fundamental Theory and Applications, Vol. 49, Issue 12, 2002 Page(s): 1729-1735

[22] Enhanced Differential Chaos Shift Keying Using Symbolic Dynamics
G. Maggio and Z. Galias
University of Mining and Metallurgy, Krakow, Poland

[23] A Chaos Based Spread Spectrum Communication System
S. Mandal and S. Banerjee
Chaos Solitons and Fractals, 20, 2004, Page(s): 567-579

[24] Quadrature Chaos-Shift Keying
Z. Galias and G. M. Maggio
Institute of Non-linear Science, University of California; Proposal for
IEEE Transactions on Circuits and Systems I
Fundamental Theory and Applications, Vol. 48, Issue 12, 2001 Page(s): 1510-1519

[25] An Observer-Based Approach for Input-Independent Global Chaos Synchronization of Discrete-Time Switched Systems
G. Millerioux and J. Daafouz
IEEE Transactions on Circuits and Systems I
Fundamental Theory and Applications, Vol. 50, Issue 10, 2003 Page(s): 1270-1279

[26] Synchronizing Hyperchaotic Systems by Observer Design
G. Grasse and S. Mascolo
IEEE Transactions on Circuits and Systems II
Analogue and Digital Signal Processing, Vol. 46, Issue 4, 1999 Page(s): 478-483

[27] Observer Based Synchronization and Input Recovery for a Class of Nonlinear Chaotic Models
E. Cherrier, M. Boutayeb and J. Ragot
44th Conference on Decision and Control, March 2005

[28] Synchronization of Chaotic Systems via Nonlinear Control
L. Huang, R. Feng and M. Fao
Physics Letters A, 320, 2004, Page(s): 271-275

[29] Extracting Messages Masked by Chaos
P. Gabriel and H. A. Cerdeira
Phys. Rev. Lett. , Volume 74, Issue 11, 1995, Page(s): 1970-1973

[30] Partial Identification of Lorenz System and its Application to Key Space Reduction of Chaotic Cryptosystems
E. Solak
IEEE Transactions on Circuits and Systems II
Express Briefs, Vol. 51, Issue 10, 2004 Page(s): 557-560

[31] Breaking Two Secure Communication Schemes Based on Chaotic Masking
G. Alvarez, F. Montoya, M. Romera and G. Pastor
IEEE Transactions on Circuits and Systems II
Express Briefs, Vol. 51, Issue 10, 2004 Page(s): 505-506

[32] Analysis of Some Recently proposed Chaos-Based Encryption Algorithms
G. Jakimoski and L. Kocarev
Physics Letters A, 291, 2001, Page(s): 381-384

[33] On Optimal Detection of Noncoherent Chaos Shift Keying Signals in a Noisy Environment
F. Lau and C. Tse
International Journal of Bifurcation and Chaos, Vol. 13, Issue 6, 2003, Page(s): 1587-1597

[34] A New Chaotic Communications Scheme
Z. Li, K. Li, C. Wen and Y. C. Soh
IEEE Transactions on Communications, Vol. 51, Issue 8, 2003 Page(s): 1306-1312

[35] Additive Mixing Modulation for Public Key Encryption Based on Distributed Dynamics
R. Tenny and L. S. Tsimring
IEEE Transactions on Circuits and Systems I
Regular Papers, Vol. 52, Issue 3, 2003 Page(s): 672-679

[36] New Approach to Chaotic Encryption
E. Alvarez, A. Fernandez, P. Garcia, J. Jimenez and A. Marcano
Physics Letters A, 263, 1999, Page(s): 373-375

[37] Permutation Based DCSK and Multiple Access DCSK Systems
F. Lau, K. Cheong and C. Tse
IEEE Transactions on Circuits and Systems I
Fundamental Theory and Applications, Vol. 50, Issue 6, 2003 Page(s): 733-742

[38] Exact Calculation of Bit Error Rates in Communication Systems with Chaotic Modulation
A. J. Lawrance and G. Ohama
IEEE Transactions on Circuits and Systems I
Fundamental Theory and Applications, Vol. 50, Issue 11, 2003 Page(s): 1391-1400

[39] Noise Performance of Chaotic Communication Systems
A. Abel, W. Schwarz and M. Gotz
IEEE Transactions on Circuits and Systems I
Fundamental Theory and Applications, Vol. 47, Issue 12, 2003 Page(s): 1726-1732

[40] On the Threshold Effect in the Estimation of Chaotic Sequences
I. Hen and N. Merhav
IEEE Transactions on Information Theory, Vol. 50, Issue 11, 2003 Page(s): 2894-2904

[41] Theoretical Noise Performance of Correlator-Based Chaotic Communications Schemes
G. Kolumbán
IEEE Transactions on Circuits and Systems I
Fundamental Theory and Applications, Vol. 47, Issue 12, 2003 Page(s): 1692-1701

[42] Applications of Stochastic Calculus and Ergodic Theory in Chaotic Communication Theory
C. Chen and K. Yao
Nonlinear Analysis, 47, 2001, Page(s): 5775-5784

[43] Optimal and Suboptimal Chaos Receivers
M. Hasler and T. Schimming
Invited Paper

[44] Towards Full Characterization of Continuous Systems in Terms of Periodic Orbits
Z. Galias
Department of Electrical Engineering, University of Science and Technology, Krakow, Poland

[45] Deterministic Nonperiodic Flow
E. N. Lorenz
Journal of the Atmospheric Sciences, Vol. 20, Issue 2 Page(s): 130-141, 1963

[46] Chaos in Dynamical Systems (2nd Edition)
E. Ott
Cambridge University Press, 2002

[47] Chaotic Behaviour of Multidimensional Difference Equations
Kaplan and J. A. Yorke
Lecture Notes in Mathematics, 730, Springer, Berlin, 1979, Page(s): 204

[48] Digital Signal Processing: A Practical Approach
E. C. Ifeachor and B. W. Jervis
Addison Wesley Publishers Ltd., Harlow, England, 1998

[49] The Fast Fourier Transform
O. Brigham
Prentice-Hall, London and New Jersey, 1974

[50] A Mathematical Theory of Communication
C. Shannon
Bell Systems Technical Journal, Issue 27, Page(s): 379-423, 1948

[51] Multivariable System Theory and Design
R. V. Patel and N. Munro
Pergamon Press, Oxford, England, 1982

[52] Digital Control
G. F. Franklin and J. D. Powell
Addison Wesley, Philippines, 1980

[53] Modern Control Systems: A Manual of Design Methods
J. Borrie
Prentice Hall, Hemel Hempstead, England, 1986

Appendix A

A.1 Properties of Sinusoids

Consider the integral which is typical of one found in the derivation of a Fourier Transform of a signal composed of a sum of sinusoids.

Derivation A.1.1

$$I = \frac{1}{T}\int_0^T f_k \sin(k\omega t + \varphi_k - \alpha) f_m \sin(m\omega t + \varphi_m - \beta) dt \qquad \text{(A.1.1)}$$

To evaluate this integral we require a trigonometrical identity namely

$$\sin x \sin y = \frac{1}{2}(\cos(x - y) - \cos(x + y)) \qquad \text{(A.1.2)}$$

So integral I if $k \neq m$ becomes

$$I = \frac{f_k f_m}{2T}\int_0^T \cos((k-m)\omega t + (\varphi_k - \varphi_m) - \cdots$$

$$\cdots \cos((k+m)\omega t + (\varphi_k + \varphi_m) - (\alpha - \beta)) dt \qquad \text{(A.1.3)}$$

$$= \frac{f_k f_m}{2T}\left[\frac{\sin((k-m)\omega t + (\varphi_k - \varphi_m) - (\alpha - \beta))}{(k-m)\omega} - \frac{\sin((k+m)\omega t + (\varphi_k + \varphi_m) - (\alpha + \beta))}{(k+m)\omega}\right]_0^T$$

For $\omega = \frac{2\pi}{T}$ and n is an integer then $\sin(2\pi n + \phi) = \sin\phi$

Then the integral I, disappears that is

$$I = 0 \qquad \text{(A.1.4)}$$

Likewise integral *I*, if $k = m$ becomes

$$I = \frac{f_k^2}{2T}\int_0^T \cos(\beta-\alpha) - \cos(2k\omega t + 2\varphi_k - (\alpha+\beta))dt$$

$$= \frac{f_k^2}{2T}\left[t\cos(\beta-\alpha) - \frac{\sin(2k\omega t + 2\varphi_k - (\alpha+\beta))}{2k\omega}\right]_0^T$$

$$= \frac{f_k^2}{2}\cos(\beta-\alpha) \qquad \text{(A.1.5)}$$

A.2 Fourier Transform Pairs

The Fourier Transform pair is given by

Result A.2.1

$$H(j\omega) = \int_{-\infty}^{\infty} h(t)e^{-j\omega t}dt$$

$$h(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} H(j\omega)e^{j\omega t}d\omega \qquad \text{(A.2.1)}$$

Also the following important result will be required

Result A.2.1

$$\underset{T\to\infty}{Lim}\left\{\frac{\sin\omega T}{\omega}\right\} = \pi\delta(\omega) \qquad \text{(A.2.2)}$$

This property comes from a nascent definition of the Dirac delta function which is given by

$$\delta_a(\omega) = \frac{1}{\pi\omega}\sin\left(\frac{\omega}{a}\right) \qquad \text{(A.2.3)}$$

Now the actual delta function is defined by any nascent function as

$$\delta(\omega) = \underset{a\to 0}{Lim}\{\delta_a(\omega)\} \tag{A.2.4}$$

if the a term is replaced by $\frac{1}{T}$ then

$$\delta(\omega) = \underset{T\to\infty}{Lim}\{\delta_{\frac{1}{T}}(\omega)\} \tag{A.2.5}$$

which when (A.2.3) is substituted into (A.2.5) gives

$$\delta(\omega) = \frac{1}{\pi}\underset{T\to\infty}{Lim}\left\{\frac{\sin\omega T}{\omega}\right\} \tag{A.2.6}$$

and hence result (A.2.1).

Consider now the function $h(t) = \alpha\cos\omega_0 t$ which will be used later in the text.

The Fourier transform of this function is given by the following derivation using the result (A.2.1).

Derivation A.2.1

$$\begin{aligned}
H(j\omega) &= \int_{-\infty}^{\infty}\alpha\cos\omega_0 t e^{-j\omega t}dt \\
&= \frac{\alpha}{2}\int_{-\infty}^{\infty}(e^{j\omega_0 t}+e^{-j\omega_0 t})e^{-j\omega t}dt \\
&= \frac{\alpha}{2}\int_{-\infty}^{\infty}e^{j(\omega_0-\omega)t}+e^{-j(\omega_0+\omega)t}dt \\
&= \underset{T\to\infty}{Lim}\left\{\frac{\alpha}{2}\left[\frac{e^{j(\omega_0-\omega)t}}{j(\omega_0-\omega)}+\frac{e^{-j(\omega_0+\omega)t}}{-j(\omega_0+\omega)}\right]_{-T}^{T}\right\} \\
&= \underset{T\to\infty}{Lim}\left\{\frac{\alpha}{2}\left(\frac{e^{j(\omega_0-\omega)T}-e^{-j(\omega_0-\omega)T}}{j(\omega_0-\omega)}+\frac{e^{-j(\omega_0+\omega)T}-e^{j(\omega_0+\omega)T}}{-j(\omega_0+\omega)}\right)\right\} \\
&= \underset{T\to\infty}{Lim}\left\{\alpha\left(\frac{\sin(\omega_0-\omega)T}{(\omega_0-\omega)}+\frac{\sin(\omega_0+\omega)T}{(\omega_0+\omega)}\right)\right\} \\
&= \alpha\pi\delta(\omega_0-\omega)+\alpha\pi\delta(\omega_0+\omega)
\end{aligned} \tag{A.2.7}$$

This result can be represented on two diagrams showing the real and imaginary parts of the derived phasor delta functions.

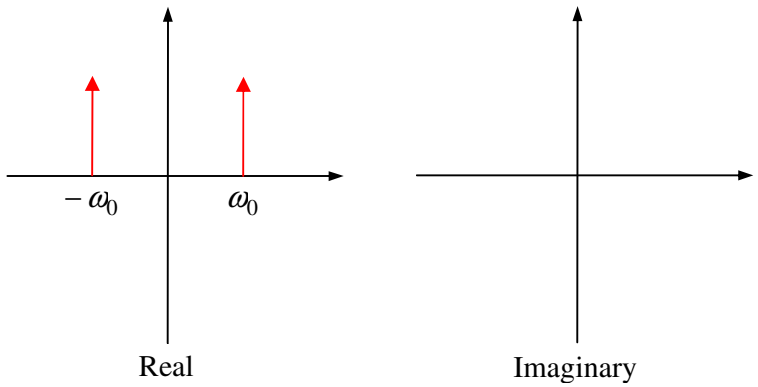


Figure A.2.1

Resultant Phasors for $h(t) = \alpha \cos \omega_0 t$

Now for completeness show that the inverse transform recreates the original signal when applied to the result of derivation (A.2.1)

Derivation A.2.2

$$h(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} (\alpha\pi\delta(\omega_0 - \omega) + \alpha\pi\delta(\omega_0 + \omega)) e^{j\omega t} d\omega$$

$$= \frac{\alpha}{2} \left(e^{j\omega_0 t} + e^{-j\omega_0 t} \right)$$

$$= \alpha \cos \omega_0 t \qquad \text{(A.2.8)}$$

This derivation is based on another property of nascent delta functions namely

Result A.2.3

$$\lim_{a\to 0}\int_{-\infty}^{\infty}\delta_a(\omega)f(\omega)d\omega=\int_{-\infty}^{\infty}\delta(\omega)f(\omega)d\omega=f(0) \qquad \text{(A.2.9)}$$

Now consider the function $h(t)=\alpha\sin\omega_0 t$ again this will be needed further on in the text. The derived Fourier transform of this function is

Derivation A.2.3

$$\begin{aligned}
H(j\omega)&=\int_{-\infty}^{\infty}\alpha\sin\omega_0 t e^{-j\omega t}dt\\
&=\frac{\alpha}{2j}\int_{-\infty}^{\infty}(e^{j\omega_0 t}-e^{-j\omega_0 t})e^{-j\omega t}dt\\
&=\frac{\alpha}{2j}\int_{-\infty}^{\infty}e^{j(\omega_0-\omega)t}-e^{-j(\omega_0+\omega)t}dt\\
&=\lim_{T\to\infty}\left\{\frac{\alpha}{2j}\left[\frac{e^{j(\omega_0-\omega)t}}{j(\omega_0-\omega)}-\frac{e^{-j(\omega_0+\omega)t}}{-j(\omega_0+\omega)}\right]_{-T}^{T}\right\}\\
&=\lim_{T\to\infty}\left\{\frac{\alpha}{2j}\left(\frac{e^{j(\omega_0-\omega)T}-e^{-j(\omega_0-\omega)T}}{j(\omega_0-\omega)}-\frac{e^{-j(\omega_0+\omega)T}-e^{j(\omega_0+\omega)T}}{-j(\omega_0+\omega)}\right)\right\}\\
&=\lim_{T\to\infty}\left\{\frac{\alpha}{j}\left(\frac{\sin(\omega_0-\omega)T}{(\omega_0-\omega)}-\frac{\sin(\omega_0+\omega)T}{(\omega_0+\omega)}\right)\right\}\\
&=-j\alpha\pi\delta(\omega_0-\omega)+j\alpha\pi\delta(\omega_0+\omega) \qquad \text{(A.2.10)}
\end{aligned}$$

This result can also be represented on two diagrams showing the real and imaginary parts of the derived phasor delta functions.

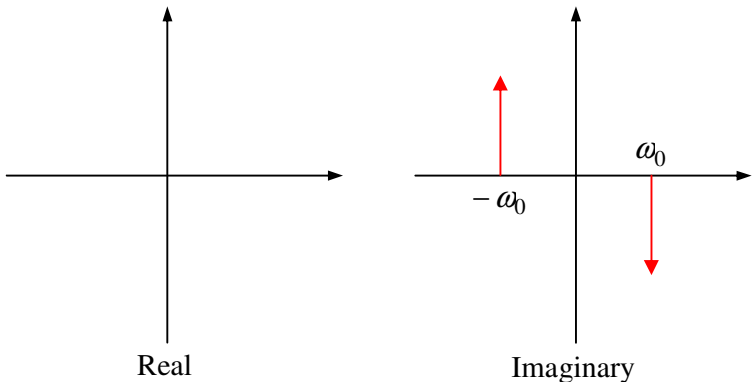


Figure A.2.2

Resultant Phasors for $h(t) = \alpha \sin \omega_0 t$

Now also for completeness show that the inverse transform recreates the original signal when applied to the derivation (A.2.3)

Derivation A.2.4

$$h(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty}\left(-j\alpha\pi\delta(\omega_0-\omega)+j\alpha\pi\delta(\omega_0+\omega)\right)e^{j\omega t}d\omega$$

$$= \frac{j\alpha}{2}\left(-e^{j\omega_0 t}+e^{-j\omega_0 t}\right)$$

$$= \frac{\alpha}{2j}\left(e^{j\omega_0 t}-e^{-j\omega_0 t}\right)$$

$$= \alpha \sin \omega_0 t \qquad \text{(A.2.11)}$$

Now consider a case required for analysing real signals consisting of a sinusoidal signal shifted by some arbitrary phase $h(t) = \alpha \sin(\omega_0 t + \varphi)$

Then the Fourier transform of this function is derived as follows

Derivation A.2.5

$$H(j\omega) = \int_{-\infty}^{\infty} \alpha \sin(\omega_0 t + \varphi) e^{-j\omega t} dt$$

$$= \frac{\alpha}{2j} \int_{-\infty}^{\infty} (e^{j(\omega_0 t + \varphi)} - e^{-j(\omega_0 t + \varphi)}) e^{-j\omega t} dt$$

$$= \frac{\alpha}{2j} \int_{-\infty}^{\infty} e^{j\varphi} e^{j(\omega_0 - \omega)t} - e^{-j\varphi} e^{-j(\omega_0 + \omega)t} dt$$

$$= \lim_{T \to \infty} \left\{ \frac{\alpha}{2j} \left[\frac{e^{j\varphi} e^{j(\omega_0 - \omega)t}}{j(\omega_0 - \omega)} - \frac{e^{-j\varphi} e^{-j(\omega_0 + \omega)t}}{-j(\omega_0 + \omega)} \right]_{-T}^{T} \right\}$$

$$= \lim_{T \to \infty} \left\{ \frac{\alpha}{2j} \left(\frac{e^{j\varphi} \left(e^{j(\omega_0 - \omega)T} - e^{-j(\omega_0 - \omega)T} \right)}{j(\omega_0 - \omega)} - \frac{e^{-j\varphi} \left(e^{-j(\omega_0 + \omega)T} - e^{j(\omega_0 + \omega)T} \right)}{-j(\omega_0 + \omega)} \right) \right\}$$

$$= \lim_{T \to \infty} \left\{ \frac{\alpha}{j} \left(\frac{e^{j\varphi} \sin(\omega_0 - \omega)T}{(\omega_0 - \omega)} - \frac{e^{-j\varphi} \sin(\omega_0 + \omega)T}{(\omega_0 + \omega)} \right) \right\} \qquad \text{(A.2.12)}$$

$$= -je^{j\varphi} \alpha\pi\delta(\omega_0 - \omega) + je^{-j\varphi} \alpha\pi\delta(\omega_0 + \omega)$$

$$= -j(\cos\varphi + j\sin\varphi)\alpha\pi\delta(\omega_0 - \omega) + j(\cos\varphi - j\sin\varphi)\alpha\pi\delta(\omega_0 + \omega)$$

$$= \alpha\pi \sin\varphi(\delta(\omega_0 - \omega) + \delta(\omega_0 + \omega)) + j\cos\varphi(\delta(\omega_0 + \omega) - \delta(\omega_0 - \omega))$$

This result can also be represented on two diagrams showing the real and imaginary parts of the derived phasor delta functions.

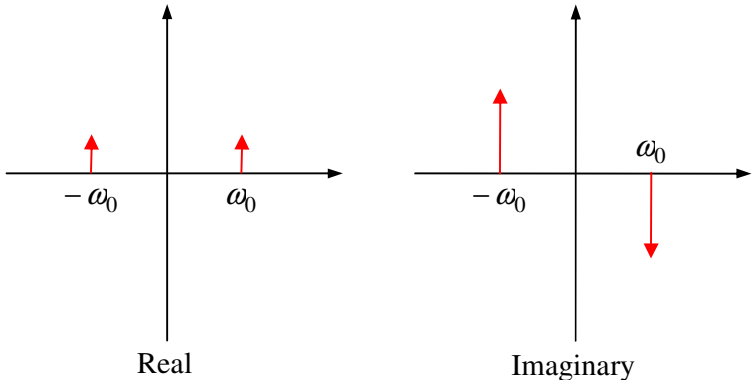


Figure A.2.3

Resultant Phasors for $h(t) = \alpha \sin(\omega_0 t + \varphi)$

MATLAB Function A.2.1– OrthogonalSignal

```
function y=OrthogonalSignal(x,n)
c=sqrt(-1);
z=fft(x);
for i=1:n/2
    z(i) = -c*z(i);
end;
for i=1+n/2:n
    z(i) = c*z(i);
end;
y=ifft(z);
y=real(y);
```

A.3 Lyapunov Exponent Calculation Listings

MATLAB Function A.3.1– CalcLyapunov

```
[T,H,X]=Lyapunov(3,@Lorenz_Variational,@ode45,0,0.01,200,[1 0 0],100);
figure(1);
plot(T,H);
title('Dynamics of Lyapunov exponents');
xlabel('Time'); ylabel('h - Lyapunov exponents');
figure(2);
plot3(X(:,1),X(:,2),X(:,3));
title('State Space Plot');
```

MATLAB Function A.3.2 – Lyapunov

```
function [T,H,X]=Lyapunov(n,SystemHandle,IntegratorHandle,

  tStart,tStep,tEnd,xStart,OutputCount);

% Values of parameters
SIGMA = 10;
R     = 28;
BETA  = 8/3;

% Memory allocation
LogSum = zeros(1,n);
h      = zeros(1,n);

% Initial values
N = n*(n + 1);

X = xStart(:)';

% Form integration vector
intVec(1:N) = [X' eye(n,n)];

% Main loop
nSteps = round((tEnd - tStart)/tStep);

t = tStart;

for Iteration = 1:nSteps

    % Solution of extended ODE system
    [TVec,Result] = feval(IntegratorHandle,SystemHandle,[t t +
tStep],intVec);

    % Form next iteration Y matrix
    intVec = Result(size(Result,1),:);

    % Nonlinear state
    x = intVec(1:n);
```

```
    X = [X; x];

    % Variational Output Matrix
    Y = intVec(n + 1:2*n)';

    for i = 2:n
       Y = [Y intVec(n*i + 1:n*(i + 1))'];
    end;

    % Update time
    t = t + tStep;

    % Construct new orthonormal basis by gram-schmidt
    [Y,znorm] = GramSchmidt(n,Y);

    % Past Log sum value
    Logd = LogSum;

    %  Update
    for k=1:n
        LogSum(k) = LogSum(k) + log(znorm(k));
    end;

    % Difference in log sum
    dLog = LogSum - Logd;

    % Calculate exponents
    for k=1:n
        h(k) = LogSum(k)/(t - tStart);
    end;

    % Update Time, Exponent and dLog/dt vectors
     if Iteration == 1
        H  = h;
        T  = t;
    else
        H  = [H; h];
        T  = [T; t];
    end;

    % Print to screen
    if (mod(Iteration,OutputCount) == 0)
        fprintf('t=%6.6f',t);
        for k=1:n
            fprintf(' %10.6f',h(k));
        end;
        fprintf('\n');
    end;

    % Copy orthonormal Y matrix to integration vector
    % intVec(1:n) states of the nonlinear system
    for i=1:n
        intVec(n*i + 1:n*(i + 1)) = Y(:,i)';
    end;
end;
```

MATLAB Function A.3.3 – Lorenz_Variational

```
function f=Lorenz_Variational(t,X)

% Values of parameters
Sigma = 10;
R     = 28;
Beta  = 8/3;

x=X(1);
y=X(2);
z=X(3);

Y= [X(4), X(7), X(10);
    X(5), X(8), X(11);
    X(6), X(9), X(12)];

f=zeros(12,1);

%Lorenz equation
f(1) = Sigma*(y - x);
f(2) = -x*z + R*x - y;
f(3) = x*y - Beta*z;

%Linearized system
 Jacobian = [-Sigma, Sigma,     0;
              R - z,    -1,    -x;
                  y,     x, -Beta];

%Variational equation
f(4:12)=Jacobian*Y;
```

Appendix B

B.1 Gram-Schmidt Method

The Gram-Schmidt method is used to orthonormalize a set of m arbitrary n dimensional vectors $\mathbf{u}_i \in R^n \quad i \in [1,m]$ where $m \le n$ that span an m dimensional subspace in an n space.

Result B.1

Consider a set of m vectors of dimension n that span a real subspace of dimension $m : \mathbf{x}_i \in R^n \quad i \in [1,m]$. A set of vectors $\mathbf{u}_i \in R^n \quad i \in [1,m]$ can be generated from $\mathbf{x}_i$ which form an orthonormal basis and spans the same m dimensional subspace.

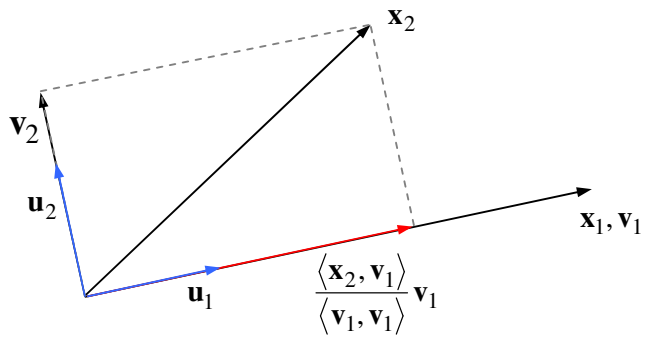


Figure B.1.1

First Stage of Gram-Schmidt Method

The algorithm follows as

Choose the first vector

$$\mathbf{v}_1 = \mathbf{x}_1 \tag{B.1.1}$$

and determine its magnitude

$$\alpha_1 = \|\mathbf{v}_1\| \tag{B.1.2}$$

then the first member of the orthonormal set is given by

$$\mathbf{u}_1 = \frac{\mathbf{v}_1}{\alpha_1} \tag{B.1.3}$$

Now choose the second vector from the set $\mathbf{x}_2$ and remove the component that corresponds to the direction of $\mathbf{x}_1$ as illustrated in figure (B.1.1) giving

$$\mathbf{v}_2 = \mathbf{x}_2 - \frac{\langle \mathbf{x}_2, \mathbf{v}_1 \rangle}{\langle \mathbf{v}_1, \mathbf{v}_1 \rangle} \mathbf{v}_1 \tag{B.1.4}$$

and as before

$$\alpha_2 = \|\mathbf{v}_2\| \tag{B.1.5}$$

then the next member of the orthonormal set is given by

$$\mathbf{u}_2 = \frac{\mathbf{v}_2}{\alpha_2} \tag{B.1.6}$$

For the next member of the set all the components in the directions of the previously calculated set $\mathbf{v}_i \ \forall\, i \in [1, k-1]$ must be removed yielding the following algorithm.

$$\mathbf{v}_k = \mathbf{x}_k - \sum_{i=1}^{k-1} \frac{\langle \mathbf{x}_k, \mathbf{v}_i \rangle}{\langle \mathbf{v}_i, \mathbf{v}_i \rangle} \mathbf{v}_i$$

$$\alpha_k = \|\mathbf{v}_k\|$$

$$\mathbf{u}_k = \frac{\mathbf{v}_k}{\alpha_k} \tag{B.1.7}$$

This algorithm can be made more numerically stable by, iteratively calculating the summing term and using the previous orthonormal basis set members, instead of the original set members. This is most easily illustrated in the following algorithm

Algorithm B.1

---

for $k = 1 : m$

$\mathbf{v}_k = \mathbf{x}_k$

for $i = 1 : k-1$

$\mathbf{v}_k = \mathbf{v}_k - \langle \mathbf{v}_k, \mathbf{u}_i \rangle \mathbf{u}_i$

end

$\alpha_k = \|\mathbf{v}_k\|$

$\mathbf{u}_k = \dfrac{\mathbf{v}_k}{\alpha_k}$

end

Appendix C

C.1 Transmission Simulation MATLAB Function Listings

The following functions are used to generate graphs for section (4.2) of this thesis simulating the transmission schemes over a noisy channel.

MATLAB Function C.1.1 – SystemU

This function simulates transmission over a noisy communication channel in either the 'U' scheme or the 'X' scheme described in chapter 3.

```
% Usage:
%
% n       Number of samples per signal vector  : n
% m       Dimension of Scheme                  : m
% Fig     First figure number                  : 1
% Psnr    Signal to noise power ratio          : 1.0
% Sigma   Noise standard deviation             : 1.0
% Runs    Number of simulation runs            : 10
% Scheme  Type of Scheme                       : 'X'
%
% [Result,MessageT,Bt,Be]=SystemU(n,m,Fig,Psnr,Sigma,Runs,Scheme)

function [Result,MessageT]=SystemU(n,m,Fig,Psnr,Sigma,Runs,Scheme)

Plot=1;

% Initialize System
[T,x,t,MessMax,EncodingMap]=Initialize(n,m);

P=Sigma*sqrt(n*Psnr)*eye(m);

MessageT=zeros(Runs+1,1);
MessageR=zeros(Runs+1,1);
MessageError=zeros(Runs+1,1);
Mt=zeros(Runs+1,m);
Mr=zeros(Runs+1,m);
```

```
Me=zeros(Runs+1,m);

Qt =zeros(n,m);
St =zeros(n,m);
Zt =zeros(n,m);

Ct=zeros(m,m);

Count=1;
Error=1;
Result=0;

% To simulate transmission delay
% run receive code first
while Runs

    % Calculate Noise Matrices
    Nq=Sigma*randn(n,m);
    Ns=Sigma*randn(n,m);

    % Decode m Messages
    if Scheme == 'U'
        Qr=Qt+Nq;
        Sr=St+Ns;

        if Plot
            figure(Fig+4);
            plot(Qr);
            str=sprintf('Received Q Matrix Balanced Orthogonal
                                                    References');
            title(str);
            xlabel('Samples');
            nstr=sprintf('%s.fig',str);
            saveas(gcf,nstr);
            nstr=sprintf('%s.bmp',str);
            saveas(gcf,nstr);
            figure(Fig+5);
            plot(Sr);
            str=sprintf('Received S Matrix Encoded Messages');
            title(str);
            xlabel('Samples');
            nstr=sprintf('%s.fig',str);
            saveas(gcf,nstr);
            nstr=sprintf('%s.bmp',str);
            saveas(gcf,nstr);
        end;
    else
        Zr=Zt+Nq;
        Sr=St+Ns;
        [U,Betat]=GramSchmidt(m,Zr);
        Qr=U*P;

        if Plot
            figure(Fig+4);
            plot(Zr);
            str=sprintf('Received Z Matrix Balanced References');
            title(str);
            xlabel('Samples');
            nstr=sprintf('%s.fig',str);
            saveas(gcf,nstr);
```

```
        nstr=sprintf('%s.bmp',str);
        saveas(gcf,nstr);
        figure(Fig+5);
        plot(Sr);
        str=sprintf('Received S Matrix Encoded Messages');
        title(str);
        xlabel('Samples');
        nstr=sprintf('%s.fig',str);
        saveas(gcf,nstr);
        nstr=sprintf('%s.bmp',str);
        saveas(gcf,nstr);
    end;
end;

Cr=inv(Qr'*Qr)*Qr'*Sr;

C=Cr;
for i=1:m
    for j=1:m
        if C(i,j) < 0.0
            C(i,j)=-1/sqrt(m);
        else
            C(i,j)=1/sqrt(m);
        end;
    end;
end;
for k=1:m
    for kk=1:2^m
        if C(:,k)==(EncodingMap(:,kk)/sqrt(m))
            Mr(Count,k)=kk-1;
            break;
        end;
    end;
end;

MessageR(Count)=0;
j=1;
for i=1:m
    MessageR(Count)=MessageR(Count)+Mr(Count,i)*2^m^(m-j);
    j=j+1;
end;

% Error calculation
if Count > 1
    MessageError(Count)=MessageT(Count-1)-MessageR(Count);
    Me(Count,:)=Mt(Count-1,:)-Mr(Count,:);
end;

if MessageError(Count)
    if Error == 1
        Result=[Count MessageError(Count)/MessMax Betat'];
    else
        Result=[Result;[Count MessageError(Count)/MessMax Betat']];
    end;
    Error=Error+1;
end;

if Plot
    figure(Fig+6);
    stairs(MessageT);
```

```
    str=sprintf('Transmitted 16 Bit Messages');
    title(str);
    xlabel('Symbols');
    nstr=sprintf('%s.fig',str);
    saveas(gcf,nstr);
    nstr=sprintf('%s.bmp',str);
    saveas(gcf,nstr);
    figure(Fig+7);
    stairs(MessageR);
    str=sprintf('Received 16 Bit Messages');
    title(str);
    xlabel('Symbols');
    nstr=sprintf('%s.fig',str);
    saveas(gcf,nstr);
    nstr=sprintf('%s.bmp',str);
    saveas(gcf,nstr);
    figure(Fig+8);
    stairs(MessageError);
    str=sprintf('Transmitted-Received 16 Bit Message Errors');
    title(str);
    xlabel('Symbols');
    nstr=sprintf('%s.fig',str);
    saveas(gcf,nstr);
    nstr=sprintf('%s.bmp',str);
    saveas(gcf,nstr);
    figure(Fig+9);
    stairs(Mt);
    str=sprintf('Transmitted 4 Bit Message Groups');
    title(str);
    xlabel('Symbols');
    nstr=sprintf('%s.fig',str);
    saveas(gcf,nstr);
    nstr=sprintf('%s.bmp',str);
    saveas(gcf,nstr);
    figure(Fig+10);
    stairs(Mr);
    str=sprintf('Received 4 Bit Message Groups');
    title(str);
    xlabel('Symbols');
    nstr=sprintf('%s.fig',str);
    saveas(gcf,nstr);
    nstr=sprintf('%s.bmp',str);
    saveas(gcf,nstr);
    figure(Fig+11);
    stairs(Me);
    str=sprintf('Transmitted-Received 4 Bit Message Errors');
    title(str);
    xlabel('Symbols');
    nstr=sprintf('%s.fig',str);
    saveas(gcf,nstr);
    nstr=sprintf('%s.bmp',str);
    saveas(gcf,nstr);
end;

% Generate m level message
MessageT(Count)=floor(MessMax*rand);
R=MessageT(Count);
j=1;
for i=1:m
    d=2^m^(m-j);
    Mt(Count,i)=floor(R/d);
```

```
    R=R-Mt(Count,i)*d;
    j=j+1;
end;

% Encode m messages
Ct=zeros(m,m);
for i=1:m
    Ct(:,i) = EncodingMap(:,Mt(Count,i)+1)/sqrt(m);
end;

% Create Reference
[Xt,x]=Chaos(n,m,x,T);
if Plot
    figure(Fig);
    plot(Xt);
    str=sprintf('X Matrix Chaotic Sequences');
    title(str);
    xlabel('Samples');
    nstr=sprintf('%s.fig',str);
    saveas(gcf,nstr);
    nstr=sprintf('%s.bmp',str);
    saveas(gcf,nstr);
end;

Zt=Normalize(Xt,m)*P;
[Ut,Betat]=GramSchmidt(m,Xt);
Qt=Ut*P;
St=Qt*Ct;

if Plot
    figure(Fig+1);
    plot(Ut);
    str=sprintf('U Matrix Orthogonal Sequences');
    title(str);
    xlabel('Samples');
    nstr=sprintf('%s.fig',str);
    saveas(gcf,nstr);
    nstr=sprintf('%s.bmp',str);
    saveas(gcf,nstr);
    if Scheme == 'U'
        figure(Fig+2);
        plot(Qt);
        str=sprintf('Q Matrix Balanced Orthogonal Reference
                                                    Sequences');
        title(str);
        xlabel('Samples');
        nstr=sprintf('%s.fig',str);
        saveas(gcf,nstr);
        nstr=sprintf('%s.bmp',str);
        saveas(gcf,nstr);
    else
        figure(Fig+2);
        plot(Zt);
        str=sprintf('Z Matrix Balanced Reference Sequences');
        title(str);
        xlabel('Samples');
        nstr=sprintf('%s.fig',str);
        saveas(gcf,nstr);
        nstr=sprintf('%s.bmp',str);
        saveas(gcf,nstr);
```

```
        end;
        figure(Fig+3);
        plot(St);
        str=sprintf('S Matrix Encoded Message Sequences');
        title(str);
        xlabel('Samples');
        nstr=sprintf('%s.fig',str);
        saveas(gcf,nstr);
        nstr=sprintf('%s.bmp',str);
        saveas(gcf,nstr);
    end;

    Count=Count+1;
    Runs=Runs-1;
end;
```

MATLAB Function C.1.1.1 - Initialize

This function sets up initial values of the system for transmission simulations.

```
function [T,x,t,MessMax,EncodingMap]=Initialize(n,m)


% Initialize random number generator
rand('state',0);
randn('state',0);


% States and times
T=0.01;
x=[10;10;10];
t=0.0;

% Initialize message variables
MessMax=2^m^m-1;

% Encoding Map
EncodingMap=Map(m);
```

MATLAB Function C.1.1.1.1 – Map

This function generates the *m* dimensional encoding map described in chapter 3.

```
function M=Map(m)

% Number of symbols
n=2^m;
% Encoding Map
M=zeros(m,n);
for i=1:m
    for j=1:n
        q=2^(i-1);
        p=floor((j+q-1)/q);
        M(i,j)=(-1)^p;
    end;
end;
```

MATLAB Function C.1.1.2 - Chaos

This function generates an *n x m* dimension matrix **X** with columns of zero mean thus providing a set of m zero mean process signals.

```
function [X,x]=Chaos(n,m,x,T)

X=zeros(n,m);

for j=1:m
    for i=1:n
        X(i,j)=x(1);

        [TimeVector,ResultantMatrix] = feval(@ode45,@lorenz,[0 T],x);

        x = ResultantMatrix(size(ResultantMatrix,1),:)';
    end;

    X(:,j)=X(:,j)-mean(X(:,j));
end;
```

MATLAB Function C.1.1.2.1 – Lorenz

This function represents the Lorenz chaotic system.

```
function f=lorenz(t,x)

f=zeros(3,1);

SIGMA = 10;
R     = 28;
BETA  = 8/3;

f(1) = SIGMA*(x(2) - x(1));
f(2) = -x(1)*x(3) + R*x(1) - x(2);
f(3) =  x(1)*x(2) - BETA*x(3);
```

MATLAB Function C.1.1.3 – Gram Schmidt

This function applies the Gram- Schmidt process to a set of *m* vectors contained in matrix **X** and returns an orthonormalized set in matrix **X**.

```
% Usage:
%
% m Number of columns to be orthonormalized  : m
% X Matrix to be orthonormalized             : X
%
% [X,Beta]=GramSchmidt(X,m)

function [X,Beta] = GramSchmidt(m,X);

Beta = zeros(m,1);

for j=1:m

    for k=1:(j-1)
        X(:,j) = X(:,j) - (X(:,j)'*X(:,k))*X(:,k);
    end;

    Beta(j) = sqrt(X(:,j)'*X(:,j));

    if Beta(j) > 0.0
        X(:,j) = X(:,j)/Beta(j);
    end;
end;
```

MATLAB Function C.1.1.4 – Normalize

This function normalizes m vectors contained in matrix X and ensure that the product $\mathbf{X}^T\mathbf{X}$ have unity values on its diagonal.

```
% Usage:
%
% X Matrix to be normalized              : X
% m Number of columns to be normalized   : m
%
% U=Normalize(X,m)

function U=Normalize(X,m)
U=X;
for i=1:m
    Beta=sqrt(X(:,i)'*X(:,i));
    if Beta
        U(:,i)=X(:,i)/Beta;
    end;
end;
```

MATLAB Function C.1.2 - SystemVec

This function simulates transmission over a noisy communication channel of the Indirect Persistent ‘x’ scheme described in chapter 3.

```
% Usage:
%
% n       Number of samples per signal vector  : n
% m       Dimension of Scheme                  : m
% Fig     First figure number                  : 1
% Psnr    Signal to noise power ratio          : 1.0
% Sigma   Noise standard deviation             : 1.0
% Runs    Number of simulation runs            : 10
% Scheme  Type of Scheme                       : 'X'
%
% [Result,MessageT]=SystemVec(n,m,Fig,Psnr,Sigma,Runs,Scheme)

function [Result,MessageT]=SystemVec(n,m,Fig,Psnr,Sigma,Runs,Scheme)

Plot=1;

% Initialize System
[T,x,t,MessMax,EncodingMap]=Initialize(n,m);

MessMax=2^m-1;

Runs=m*Runs;

p=Sigma*sqrt(n*Psnr);
```

```
MessageT=zeros(Runs+1,1);
MessageR=zeros(Runs+1,1);
MessageError=zeros(Runs+1,1);
Mt=zeros(Runs+1,1);
Mr=zeros(Runs+1,1);
Me=zeros(Runs+1,1);

Xt =zeros(n,m);

Zr =zeros(n,m);
zt =zeros(n,1);

st =zeros(n,1);

Count=1;
Error=1;
Result=0;

% To simulate transmission delay
% run receive code first
while Runs

    % Calculate Noise Matrices
    nq=Sigma*randn(n,1);
    ns=Sigma*randn(n,1);

    % Decode Message
    if Scheme == 'x'
        zr=zt+nq;
        sr=st+ns;

        Zr=[zr Zr(:,1:m-1)];

        [U,Betat]=GramSchmidt(m,Zr);
        Qr=U*p;

        if Plot
            figure(Fig+4);
            plot(Zr);
            str=sprintf('Persistent Received Z Matrix Balanced
Sequences');
            title(str);
            xlabel('Samples');
            nstr=sprintf('%s.fig',str);
            saveas(gcf,nstr);
            nstr=sprintf('%s.bmp',str);
            saveas(gcf,nstr);
            figure(Fig+5);
            plot(sr);
            str=sprintf('Received s Vector Encoded Sequence');
            title(str);
            xlabel('Samples');
            nstr=sprintf('%s.fig',str);
            saveas(gcf,nstr);
            nstr=sprintf('%s.bmp',str);
            saveas(gcf,nstr);
        end;

    else
```

```
        return;
    end;

    cr=inv(Qr'*Qr)*Qr'*sr;

    c=cr;
    for i=1:m
        if c(i) < 0.0
            c(i)=-1/sqrt(m);
        else
            c(i)=1/sqrt(m);
        end;
    end;
    for kk=1:2^m
        if c==(EncodingMap(:,kk)/sqrt(m))
            Mr(Count)=kk-1;
            break;
        end;
    end;

    MessageR(Count)=Mr(Count);

    % Error calculation
    if Count > 1
        MessageError(Count)=MessageT(Count-1)-MessageR(Count);
        Me(Count)=Mt(Count-1)-Mr(Count);
    end;

    if MessageError(Count)
        if Error == 1
            Result=[Count MessageError(Count)/MessMax Betat'];
        else
            Result=[Result;[Count MessageError(Count)/MessMax
Betat']];
        end;
        Error=Error+1;
    end;

    if Plot
        figure(Fig+6);
        stairs(MessageT);
        str=sprintf('Transmitted 4 Bit Messages');
        title(str);
        xlabel('Samples');
        nstr=sprintf('%s.fig',str);
        saveas(gcf,nstr);
        nstr=sprintf('%s.bmp',str);
        saveas(gcf,nstr);
        figure(Fig+7);
        stairs(MessageR);
        str=sprintf('Received 4 Bit Messages');
        title(str);
        xlabel('Symbols');
        nstr=sprintf('%s.fig',str);
        saveas(gcf,nstr);
        nstr=sprintf('%s.bmp',str);
        saveas(gcf,nstr);
        figure(Fig+8);
        stairs(MessageError);
        str=sprintf('Transmitted-Received 4 Bit Message Errors');
```

```
    title(str);
    xlabel('Symbols');
    nstr=sprintf('%s.fig',str);
    saveas(gcf,nstr);
    nstr=sprintf('%s.bmp',str);
    saveas(gcf,nstr);
end;

% Generate m level message
MessageT(Count)=floor(MessMax*rand);
R=MessageT(Count);
Mt(Count)=R;

% Encode message
ct = EncodingMap(:,Mt(Count)+1)/sqrt(m);

% Create Reference
[xt,x]=Chaos(n,1,x,T);
if Plot
    figure(Fig);
    plot(xt);
    str=sprintf('x Vector Chaotic Sequence');
    title(str);
    xlabel('Samples');
    nstr=sprintf('%s.fig',str);
    saveas(gcf,nstr);
    nstr=sprintf('%s.bmp',str);
    saveas(gcf,nstr);

end;

Xt=[xt Xt(:,1:m-1)];

zt=Normalize(xt,1)*p;
[Ut,Betat]=GramSchmidt(m,Xt);
Qt=Ut*p;
st=Qt*ct;

if Plot
    figure(Fig+1);
    plot(Xt);
    str=sprintf('Persistent X Matrix Chaotic Sequence');
    title(str);
    xlabel('Samples');
    nstr=sprintf('%s.fig',str);
    saveas(gcf,nstr);
    nstr=sprintf('%s.bmp',str);
    saveas(gcf,nstr);
    figure(Fig+2);
    plot(Ut);
    str=sprintf('U Matrix Orthogonal Sequences');
    title(str);
    xlabel('Samples');
    nstr=sprintf('%s.fig',str);
    saveas(gcf,nstr);
    nstr=sprintf('%s.bmp',str);
    saveas(gcf,nstr);
    figure(Fig+3);
    plot(st);
    str=sprintf('s Vector Encoded Message Sequence');
```

```
        title(str);
        xlabel('Samples');
        nstr=sprintf('%s.fig',str);
        saveas(gcf,nstr);
        nstr=sprintf('%s.bmp',str);
        saveas(gcf,nstr);
    end;

    Count=Count+1;
    Runs=Runs-1;
end;
```

C.2 BER Simulation MATLAB Function Listings

MATLAB Function C.2.1 – Ugen

This function simulates the BER v P_{snr} [dB] noise behaviour of the Direct 'U' Scheme

---

```
% Usage:
%
% n           An array of samples per signal vector     :  [8 32 128]
% PsnrLimits  An array of limits for Psnr               :  [0.01 100]
% PsnrCount   Number of points over PsnrLimits range    :  100
% SimCount    Number of simulation runs                 :  10000
% m_bits      Array of dimension and bits               :  [2 4]
% out         Rate of plot output                       :
SimCount/100
%
% [BER,P,Psnr]=Ugen(n,PsnrLimits,PsnrCount,SimCount,m_bits,out)

function [BER,P,Psnr]=Ugen(n,PsnrLimits,PsnrCount,SimCount,m_bits,out)

% Variable samples of data
Num_n=max(size(n));

% Power Ratio Logarithmic Vector
v=nthroot(PsnrLimits(2)/PsnrLimits(1),PsnrCount-1);
Psnr=zeros(PsnrCount,1);
Psnr(1)=PsnrLimits(1);
for ii=2:PsnrCount
    Psnr(ii)=Psnr(ii-1)*v;
end;

% Determine m and bits variables
Size_M_Bits = size(m_bits);
Dims = Size_M_Bits(1);

%Probability matrix
P=zeros(PsnrCount,Num_n,Dims);
BER=P;

for y=1:Dims
```

```
% Dimension and Bits
m=m_bits(y,1);
bits=m_bits(y,2);

% Number of symbols
mu=2^bits;

% Typical c vector
c=zeros(m,1);
if m==2
    c=[real(exp(i*pi/mu));imag(exp(i*pi/mu))];
else
    for km=1:m
        c(km)=1/sqrt(m);
    end;
end;

% Identity matrix Im
Im=eye(m);

% Set random number state
randn('state',0);

% Type of bit error calculation
Direct = 0;

% SimCount simulation runs
for ii=1:SimCount
    % For each data sample length
    for kk=1:Num_n
        E=randn(n(kk),m);
        Emm=E(1:m,1:m);
        e=randn(n(kk),1);
        em=e(1:m,1);

        % Over the range
        for k=1:PsnrCount
            QTQ=n(kk)*Psnr(k)*Im+sqrt(n(kk)*Psnr(k))
                                    *(Emm'+Emm)+E'*E;
            QTs=n(kk)*Psnr(k)*c+sqrt(n(kk)*Psnr(k))
                                    *(Emm'*c+em)+E'*e;

            cest=inv(QTQ)*QTs;

            if Direct==1
                for km=1:m
                    if cest(km)<0
                        P(k,kk,y)=P(k,kk,y)+1/SimCount/m;
                        BER(:,:,y)=P(:,:,y);
                    end;
                end;
            else
                for km=1:m
                    if cest(km)<0
                        P(k,kk,y)=P(k,kk,y)+1/SimCount;
                        BER(:,:,y)=1.-nthroot(1.-P(:,:,y),bits);
                        break;
                    end;
```

```
                    end;
                end;
            end;
        end;

        % Plot output
        if (mod(ii,out)==0)
            str=sprintf('%d Dimension %d Symbols : %d%c
                        Complete',m,mu,floor(100*ii/SimCount),'%');
            if Dims==1
                semilogy(10*log10(Psnr),[SimCount*BER(:,:,1)/ii]);
            end;
            if Dims==2
                semilogy(10*log10(Psnr),[SimCount*BER(:,:,1)/ii
                                         SimCount*BER(:,:,2)/ii]);
            end;
            if Dims>=3
                semilogy(10*log10(Psnr),[BER(:,:,1) BER(:,:,2)
                                                    BER(:,:,3)]);
            end;
            title(str);
            ylim([0.0001 1]);
            xlim([-20 20]);
            xlabel('Psnr[dB]');
            ylabel('BER');
            pause(0.01);
        end;
    end;
end;
```

MATLAB Function C.2.2 - Xgen

This function simulates the BER v P_{snr} [dB] noise behaviour of the Indirect 'X' Scheme

---

```
% Usage:
%
% WForm       Character specification for W matrix     :  'A'
% n           An array of samples per signal vector    :  [8 32 128]
% PsnrLimits  An array of limits for Psnr              :  [0.01 100]
% PsnrCount   Number of points over PsnrLimits range   :  100
% SimCount    Number of simulation runs                :  10000
% m_bits      Array of dimension and bits              :  [2 4]
% out         Rate of plot output                      :  SimCount/100
%
% [BER,P,Psnr]=Xgen(WForm,n,PsnrLimits,PsnrCount,SimCount,m_bits,out)

function
[BER,P,Psnr]=Xgen(WForm,n,PsnrLimits,PsnrCount,SimCount,m_bits,out)

% Variable samples of data
Num_n=max(size(n));

% Power Ratio Vector
v=nthroot(PsnrLimits(2)/PsnrLimits(1),PsnrCount-1);
Psnr=zeros(PsnrCount,1);
Psnr(1)=PsnrLimits(1);
```

```
for ii=2:PsnrCount
    Psnr(ii)=Psnr(ii-1)*v;
end;

% Determine m and bits variables
Size_M_Bits=size(m_bits);
Dims=Size_M_Bits(1);

%Probability matrix
P=zeros(PsnrCount,Num_n,Dims);
BER=P;

for y=1:Dims

    m=m_bits(y,1);
    bits=m_bits(y,2);

    % Number of symbols
    mu=2^bits;

    % Typical c vector
    c=zeros(m,1);
    if m==2
        c=[real(exp(i*pi/mu));imag(exp(i*pi/mu))];
    else
        for km=1:m
            c(km)=1/sqrt(m);
        end;
    end;

    % Identity matrix Im
    Im=eye(m);

    % Set random number state
    randn('state',0);

    % Scheme I
    W=Im;

    % Scheme A
    if WForm=='A'
        for jj=1:m
            for ii=1:jj
                W(ii,jj)=1.0/sqrt(jj);
            end;
        end;
    end;

    % Scheme B
    if WForm=='B'
        W=[1.0000    0.1481   -0.1345    0.1205;
           0.0000    0.9890    0.8609    0.7699;
          -0.0000   -0.0000    0.4907    0.3383;
           0.0000    0.0000    0.0000    0.5276];

        W=W(1:m,1:m);
    end;

    % Scheme C
```

```
if WForm=='C'
    W=zeros(m,m);
    for jj=1:m
        W(1,jj)=1.0;
    end;
end;

% SimCount simulation runs
for ii=1:SimCount
    % For each data sample length
    for kk=1:Num_n
        U=[Im;zeros(n(kk)-m,m)];
        E=randn(n(kk),m);
        Emm=E(1:m,1:m);
        e=randn(n(kk),1);
        em=e(1:m,1);

        % Over the range
        for k=1:PsnrCount
            A=sqrt(n(kk)*Psnr(k))*U*W+E;
            [Ub,Beta] = GramSchmidt(m,A);
            QTQ=n(kk)*Psnr(k)*Im;
            QTs=n(kk)*Psnr(k)*Ub'*U*c+sqrt(n(kk)*Psnr(k))*Ub'*e;

            cest=inv(QTQ)*QTs;

            if m==2
                Theta=atan2(cest(2),cest(1));
                if Theta<0 || Theta>2*pi/mu
                    P(k,kk,y)=P(k,kk,y)+1/SimCount;
                    BER(:,:,y)=1.-nthroot(1.-P(:,:,y),bits);
                end;
            else
                for km=1:m
                    if cest(km)<0
                        P(k,kk,y)=P(k,kk,y)+1/SimCount;
                        BER(:,:,y)=1.-nthroot(1.-P(:,:,y),bits);
                        break;
                    end;
                end;
            end;
        end;
    end;

    % Plot output
    if (mod(ii,out)==0)
        str=sprintf('%d Dimension %d Symbols : %d%c
                  Complete',m,mu,floor(100*ii/SimCount),'%');
        if Dims==1
            semilogy(10*log10(Psnr),[SimCount*BER(:,:,1)/ii]);
        end;
        if Dims==2
            semilogy(10*log10(Psnr),[SimCount*BER(:,:,1)/ii
                                     SimCount*BER(:,:,2)/ii]);
        end;
        if Dims>=3
            semilogy(10*log10(Psnr),[BER(:,:,1) BER(:,:,2)
                                                BER(:,:,3)]);
        end;
        title(str);
```

```
            ylim([0.0001 1]);
            xlim([-20 20]);
            xlabel('Psnr[dB]');
            ylabel('BER');
            pause(0.01);
        end;
    end;
end;
```

MATLAB Function C.2.3 - EUgen

This function simulates the BER v $\frac{E_b}{N_0}$[dB] noise behaviour of the Direct 'U' Scheme

---

```
% Usage:
%
% n          An array of samples per signal vector    :  [8 32 128]
% EbNoLimits An array of limits for EbNo              :  [0.01 100]
% EbNoCount  Number of points over EbNoLimits range   :  100
% SimCount   Number of simulation runs                :  10000
% m_bits     Array of dimension and bits              :  [2 4]
% out        Rate of plot output                      :  SimCount/100
%
% [BER,P,Psnr]=EUgen(n,EbNoLimits,EbNoCount,SimCount,m_bits,out)


function [BER,P,EbNo]=EUgen(n,EbNoLimits,EbNoCount,SimCount,
                                                        m_bits,out)

% Variable samples of data
Num_n=max(size(n));

% Power Ratio Vector
v=nthroot(EbNoLimits(2)/EbNoLimits(1),EbNoCount-1);
EbNo=zeros(EbNoCount,1);
EbNo(1)=EbNoLimits(1);
for ii=2:EbNoCount
    EbNo(ii)=EbNo(ii-1)*v;
end;

% Determine m and bits variables
Size_M_Bits=size(m_bits);
Dims=Size_M_Bits(1);

%Probability matrix
P=zeros(EbNoCount,Num_n,Dims);
BER=P;

for y=1:Dims

    m=m_bits(y,1);
    bits=m_bits(y,2);
```

```
% Number of symbols
mu=2^bits;

% Typical c vector
c=zeros(m,1);
if m==2
    c=[real(exp(i*pi/mu));imag(exp(i*pi/mu))];
else
    for km=1:m
        c(km)=1/sqrt(m);
    end;
end;

% Identity matrix Im
Im=eye(m);

% Set random number state
randn('state',0);

% SimCount simulation runs
for ii=1:SimCount
    % For each data sample length
    for kk=1:Num_n
        E=randn(n(kk),m);
        Emm=E(1:m,1:m);
        e=randn(n(kk),1);
        em=e(1:m,1);

        % Over the range
        for k=1:EbNoCount
            Psnr=m*EbNo(k)/n(kk);
            QTQ=Psnr*n(kk)*Im+sqrt(Psnr)*sqrt(n(kk))
                                        *(Emm'+Emm)+E'*E;
            QTs=Psnr*n(kk)*c+sqrt(Psnr)*sqrt(n(kk))
                                        *(Emm'*c+em)+E'*e;

            cest=inv(QTQ)*QTs;

            if m==2
                Theta=atan2(cest(2),cest(1));
                if Theta<0 || Theta>2*pi/mu
                    P(k,kk,y)=P(k,kk,y)+1/SimCount;
                    BER(:,:,y)=1.-nthroot(1.-P(:,:,y),bits);
                end;
            else
                for km=1:m
                    if cest(km)<0
                        P(k,kk,y)=P(k,kk,y)+1/SimCount;
                        BER(:,:,y)=1.-nthroot(1.-P(:,:,y),bits);
                        break;
                    end;
                end;
            end;
        end;
    end;

    % Plot Output
    if (mod(ii,out)==0)
        str=sprintf('%d Dimension %d Symbols : %d%c
                            Complete',m,mu,100*ii/SimCount,'%');
```

```
            if Dims==1
                semilogy(10*log10(EbNo),[SimCount*BER(:,:,1)/ii]);
            end;
            if Dims==2
                semilogy(10*log10(EbNo),[SimCount*BER(:,:,1)/ii
                                         SimCount*BER(:,:,2)/ii]);
            end;
            if Dims>=3
                semilogy(10*log10(EbNo),[BER(:,:,1) BER(:,:,2)
                                                    BER(:,:,3)]);
            end;
            title(str);
            ylim([0.0001 1]);
            xlim([0 20]);
            xlabel('Eb/No[dB]');
            ylabel('BER');
            pause(0.01);
        end;
    end;
end;
```

MATLAB Function C.2.4 - EXgen

This function simulates the BER v $\frac{E_b}{N_0}$[dB] noise behaviour of the Direct 'X' Scheme

---

```
% Usage:
%
% WForm      Character specification for W matrix     :  'A'
% n          An array of samples per signal vector    :  [8 32 128]
% EbNoLimits An array of limits for EbNo              :  [0.01 100]
% EbNoCount  Number of points over EbNoLimits range   :  100
% SimCount   Number of simulation runs                :  10000
% m_bits     Array of dimension and bits              :  [2 4]
% out        Rate of plot output                      :  SimCount/100
%
% [BER,P,Psnr]=EXgen(WForm,n,EbNoLimits,EbNoCount,SimCount,m_bits,out)

function
[BER,P,EbNo]=EXgen(WForm,n,EbNoLimits,EbNoCount,SimCount,m_bits,out)

% Variable samples of data
Num_n=max(size(n));

% Power Ratio Vector
v=nthroot(EbNoLimits(2)/EbNoLimits(1),EbNoCount-1);
EbNo=zeros(EbNoCount,1);
EbNo(1)=EbNoLimits(1);
for ii=2:EbNoCount
    EbNo(ii)=EbNo(ii-1)*v;
end;

% Determine m and bits variables
Size_M_Bits=size(m_bits);
```

```
Dims=Size_M_Bits(1);

%Probability matrix
P=zeros(EbNoCount,Num_n,Dims);
BER=P;

for y=1:Dims

    m=m_bits(y,1);
    bits=m_bits(y,2);

    % Number of symbols
    mu=2^bits;

    % Typical c vector
    c=zeros(m,1);
    if m==2
        c=[real(exp(i*pi/mu));imag(exp(i*pi/mu))];
    else
        for km=1:m
            c(km)=1/sqrt(m);
        end;
    end;

    % Identity matrix Im
    Im=eye(m);

    % Set random number state
    randn('state',0);

    % Scheme I
    W=Im;

    % Scheme A
    if WForm=='A'
        for jj=1:m
            for ii=1:jj
                W(ii,jj)=1.0/sqrt(jj);
            end;
        end;
    end;

    % Scheme B
    if WForm=='B'
        W=[1.0000    0.1481   -0.1345    0.1205;
           0.0000    0.9890    0.8609    0.7699;
          -0.0000   -0.0000    0.4907    0.3383;
           0.0000    0.0000    0.0000    0.5276];

        W=W(1:m,1:m);
    end;

    % Scheme C
    if WForm=='C'
        W=zeros(m,m);
        for jj=1:m
            W(1,jj)=1.0;
        end;
    end;
```

```
% SimCount simulation runs
for ii=1:SimCount
    % For each data sample length
    for kk=1:Num_n
        U=[Im;zeros(n(kk)-m,m)];
        E=randn(n(kk),m);
        Emm=E(1:m,1:m);
        e=randn(n(kk),1);
        em=e(1:m,1);

        % Over the range
        for k=1:EbNoCount
            Psnr=m*EbNo(k)/n(kk);
            A=sqrt(n(kk)*Psnr)*U*W+E;
            [Ub,Beta] = GramSchmidt(m,A);
            QTQ=n(kk)*Psnr*Im;
            QTs=n(kk)*Psnr*Ub'*U*c+sqrt(n(kk)*Psnr)*Ub'*e;

            cest=inv(QTQ)*QTs;

            if m==2
                Theta=atan2(cest(2),cest(1));
                if Theta<0 || Theta>2*pi/mu
                    P(k,kk,y)=P(k,kk,y)+1/SimCount;
                    BER(:,:,y)=1.-nthroot(1.-P(:,:,y),bits);
                end;
            else
                for km=1:m
                    if cest(km)<0
                        P(k,kk,y)=P(k,kk,y)+1/SimCount;
                        BER(:,:,y)=1.-nthroot(1.-P(:,:,y),bits);
                        break;
                    end;
                end;
            end;
        end;
    end;

    %Plot Outputs
    if (mod(ii,out)==0)
        str=sprintf('%d Dimension %d Symbols : %d%c
                            Complete',m,mu,100*ii/SimCount,'%');
        if Dims==1
            semilogy(10*log10(EbNo),[SimCount*BER(:,:,1)/ii]);
        end;
        if Dims==2
            semilogy(10*log10(EbNo),[SimCount*BER(:,:,1)/ii
                                        SimCount*BER(:,:,2)/ii]);
        end;
        if Dims>=3
            semilogy(10*log10(EbNo),[BER(:,:,1) BER(:,:,2)
                                                    BER(:,:,3)]);
        end;
        title(str);
        ylim([0.0001 1]);
        xlim([0 20]);
        xlabel('Eb/No[dB]');
        ylabel('BER');
```

```
            pause(0.01);
        end;
    end;
end;
```

Appendix D

D.1 Hypersphere Volume Calculations

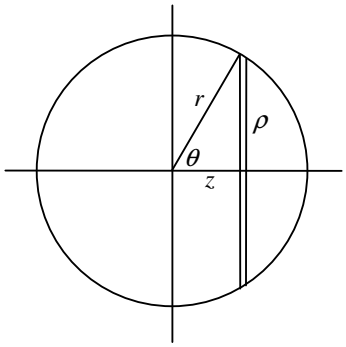


Figure D.1.1
Calculation of Dimension *m* Hyperspherical Volume

Derivation D.1.1

Consider the volume of an m dimensional hypersphere expressed as a function of hyperspherical volumes of dimension $m-1$, as illustrated in the figure (D.1.1) above

$$V_m(r) = \int_{-r}^{r} V_{m-1}(\rho)dz \qquad \text{(D.1.1)}$$

Making the following substitutions $z = r\cos\theta$, $dz = -r\sin\theta d\theta$ and $\rho = r\sin\theta$ yields

$$V_m(r) = -\int_{-\pi}^{0} V_{m-1}(r\sin\theta) r\sin\theta d\theta$$

$$= r\int_{0}^{\pi} V_{m-1}(r\sin\theta)\sin\theta d\theta \qquad \text{(D.1.2)}$$

Now the volume of the hypersphere is clearly proportional to the radius r raised to the m^{th} power. That is

$$V_m(r) = \alpha(m) r^m \qquad \text{(D.1.3)}$$

where the $\alpha(m)$ term is a constant dependent on the dimension m. From equation (D.1.2) the reduced radius $\rho = r\sin\theta$ can be used to express the $m-1$ dimensional volume in the same way, using the constant term for the reduced dimension as

$$V_{m-1}(r\sin\theta) = \alpha(m-1) r^{m-1}\sin\theta \qquad \text{(D.1.4)}$$

Substituting (D.1.3) and (D.1.4) into (D.1.2) gives a dimensionally recursive expression for the constant term, that is

$$\alpha(m) r^m = r\int_{0}^{\pi} \alpha(m-1) r^{m-1}\sin^{m-1}\theta\sin\theta d\theta \qquad \text{(D.1.5)}$$

And cancelling like powers of the radius r gives

$$\alpha(m) = \alpha(m-1)\int_{0}^{\pi} \sin^m\theta d\theta \qquad \text{(D.1.6)}$$

The integral sine function can be expressed in a recursive form as follows

$$I(m) = \int_{0}^{\pi} \sin^m\theta d\theta \qquad \text{(D.1.7)}$$

then (D.1.6) becomes

$$\alpha(m) = \alpha(m-1) I(m) \qquad \text{(D.1.8)}$$

Now $I(m)$ can be rewritten as

$$I(m) = \int_0^{\pi} \sin^2\theta \sin^{m-2}\theta d\theta \tag{D.1.9}$$

And integrating by parts gives

$$I(m) = \int_0^{\pi} \sin^{m-2}\theta - \cos^2\theta \sin^{m-2}\theta d\theta$$

$$= I(m-2) - \int_0^{\pi} \cos\theta \cdot \cos\theta \sin^{m-2}\theta d\theta$$

$$= I(m-2) - \left[\cos\theta \cdot \frac{\sin^{m-1}\theta}{(m-1)}\right]_0^{\pi} - \int_0^{\pi} \frac{\sin^{m-1}\theta}{(m-1)} \cdot \sin\theta d\theta$$

$$= I(m-2) - \frac{1}{(m-1)} I(m) \tag{D.1.10}$$

And finally by rearranging produces a recursive formula for the sine integral function as

$$I(m) = \frac{(m-1)}{m} I(m-2) \tag{D.1.11}$$

For the next part of the derivation we will need two special values of this function these are $I(0)$ and $I(1)$. These are simply calculated to yield

$$I(0) = \int_0^{\pi} d\theta = \pi \quad \text{and} \quad I(1) = \int_0^{\pi} \sin\theta d\theta = 2 \tag{D.1.12}$$

Two other values are also required namely $\alpha(0)$ and $\alpha(1)$. Now we know that $\alpha(2) = \pi$ because the volume of a two dimensional hypersphere is πr^2. So from (D.1.8) and (D.1.11) these values are

$$\alpha(1) = \frac{\alpha(2)}{I(2)} = \frac{\alpha(2)}{\frac{1}{2} I(0)} = 2 \quad \text{and} \quad \alpha(0) = \frac{\alpha(1)}{I(1)} = 1$$

Next reconsider equation (D.1.8) and further expand it until $m = 1$

$$\alpha(m) = I(m)\alpha(m-1)$$

$$= I(m)I(m-1)\alpha(m-2)$$

$$= I(m) \cdots I(1)\alpha(0) \tag{D.1.13}$$

Then $\alpha(m)$ can be expressed as follows in terms of the product function

$$\alpha(m) = \Pi(m)\frac{\alpha(0)}{I(0)}$$

$$= \frac{\Pi(m)}{\pi} \qquad \text{(D.1.14)}$$

where the product function is

$$\Pi(m) = I(m)\cdots I(1)I(0) \qquad \text{(D.1.15)}$$

Now the product function can be found using the two shift recursive formula given by equation (D.1.11) so expressing it as

$$\Pi(m) = I(m)I(m-1)\Pi(m-2) \qquad \text{(D.1.16)}$$

And further expanding the recursion until it contains the known values given by equations (D.1.11) and (D.1.12) gives

$$I(m)I(m-1) = \frac{(m-1)}{m}\cdot\frac{(m-2)}{(m-1)}\cdot I(m-2)I(m-3)$$

$$= \frac{(m-2)}{m}I(m-2)I(m-3)$$

$$= \frac{(m-2)}{m}\cdot\frac{(m-4)}{(m-2)}I(m-4)I(m-5)$$

$$= \frac{(m-2)}{m}\cdot\frac{(m-4)}{(m-2)}\cdots\frac{4}{6}\cdot\frac{2}{4}I(2)I(1)$$

$$= \frac{2}{m}I(2)I(1)$$

$$= \frac{\pi}{\left(\frac{m}{2}\right)} \qquad \text{(D.1.17)}$$

This allows equation (D.1.16) to be expressed as the following recursive expansion

$$\Pi(m) = \frac{\pi}{\left(\frac{m}{2}\right)}\cdot\Pi(m-2)$$

$$= \frac{\pi}{\left(\frac{m}{2}\right)} \cdot \frac{\pi}{\left(\frac{m-2}{2}\right)} \Pi(m-4)$$

$$= \frac{\pi}{\left(\frac{m}{2}\right)} \cdot \frac{\pi}{\left(\frac{m-2}{2}\right)} \cdot \frac{\pi}{\left(\frac{m-4}{2}\right)} \cdots \frac{\pi}{\left(\frac{2}{2}\right)} \Pi(0) \qquad \text{(D.1.18)}$$

The resultant equation (D.1.18) can now be expressed in terms of the gamma function to in place of the factorial term and substituting $\Pi(0) = I(0) = \pi$ yields

$$\Pi(m) = \frac{\pi^{\frac{m}{2}}}{\left(\frac{m}{2}\right)!} \Pi(0)$$

$$= \frac{\pi^{\frac{m}{2}+1}}{\left(\frac{m}{2}\right)!}$$

$$\Pi(m) = \frac{\pi^{\frac{m}{2}+1}}{\Gamma\left(\frac{m}{2}+1\right)} \qquad \text{(D.1.19)}$$

The volume of the hypersphere can now be expressed as follows using equations (D.1.3) (D.1.14) and (D.1.19) as

$$V_m(r) = \frac{\pi^{\frac{m}{2}}}{\Gamma\left(\frac{m}{2}+1\right)} \cdot r^m \qquad \text{(D.1.20)}$$

Now for even values of *m*, the gamma function can easily be evaluated since at integer values it merely becomes a factorial, that is $\Gamma(k+1) = k!$. However for odd values of *m* the gamma function has to be evaluated at values halfway between integers as follows

$$\Gamma\left(\frac{m}{2}+1\right) = \left(\frac{m}{2}\right) \cdot \Gamma\left(\frac{m}{2}-1\right)$$

$$= \left(\frac{m}{2}\right) \cdot \left(\frac{m-2}{2}\right) \cdot \left(\frac{m-4}{2}\right) \cdots \frac{3}{2} \cdot \frac{1}{2} \cdot \Gamma\left(\frac{1}{2}\right) \qquad \text{(D.1.21)}$$

Now in order to find $\Gamma\left(\frac{1}{2}\right)$ we require Euler's reflection formula that is

$$\Gamma(z)\Gamma(1-z)=\frac{\pi}{\sin \pi z} \tag{D.1.22}$$

So for $z=\frac{1}{2}$ the result is

$$\Gamma^2\left(\frac{1}{2}\right)=\frac{\pi}{\sin\left(\frac{\pi}{2}\right)} \tag{D.1.23}$$

This yields the well known result

$$\Gamma\left(\frac{1}{2}\right)=\pi^{\frac{1}{2}} \tag{D.1.24}$$

Allowing the volume for any dimension *m* to be calculated using equation (D.1.20).

D.2 Maxima of Comparative Function

Derivation D.2.1

Recall the comparative function given by result (D.2.5)

$$C_m=\frac{\pi^{\frac{m}{2}-1}}{\Gamma\left(1+\frac{m}{2}\right)}\cdot\frac{m}{2}$$

The differential of this with respect to the dimension *m* is derived and equated to zero to find the maxima as

$$\frac{dC_m}{dm}=\frac{\Gamma\left(1+\frac{m}{2}\right)\cdot\frac{d}{dm}\left\{\pi^{\frac{m}{2}-1}\cdot\frac{m}{2}\right\}-\pi^{\frac{m}{2}-1}\cdot\frac{m}{2}\cdot\frac{d}{dm}\Gamma\left(1+\frac{m}{2}\right)}{\Gamma^2\left(1+\frac{m}{2}\right)}=0 \tag{D.2.1}$$

Evaluating intermediate functional derivatives for equation (D.2.1) yields

$$\frac{d}{dm}\left\{\pi^{\frac{m}{2}-1}\cdot\frac{m}{2}\right\}=\frac{m}{2}\cdot\frac{d}{dm}\left\{\pi^{\frac{m}{2}-1}\right\}+\frac{1}{2}\pi^{\frac{m}{2}-1} \tag{D.2.2}$$

And

$$\frac{d}{dm}\left\{\pi^{\frac{m}{2}-1}\right\}=\frac{1}{2}\log_e\pi\cdot\pi^{\frac{m}{2}-1} \tag{D.2.3}$$

Now consider the digamma function which is defined as

$$\varphi(m)=\frac{d}{dm}\log_e\Gamma(m)=\frac{\frac{d}{dx}\Gamma(m)}{\Gamma(m)} \tag{D.2.4}$$

And rearranging this allows the expression of the gamma function derivative in equation (D.2.1) to be expressed as

$$\frac{d}{dm}\Gamma(m)=\Gamma(m)\varphi(m) \tag{D.2.5}$$

$$\frac{d}{dm}\Gamma\left(1+\frac{m}{2}\right)=\frac{1}{2}\Gamma\left(1+\frac{m}{2}\right)\varphi\left(1+\frac{m}{2}\right) \tag{D.2.6}$$

Substituting (D.2.2), (D.2.3) and (D.2.6) into equation (D.2.1) gives the following result

$$\Gamma\left(1+\frac{m}{2}\right)\left[\frac{m}{2}\cdot\frac{1}{2}\log_e\pi\cdot\pi^{\frac{m}{2}-1}+\frac{1}{2}\pi^{\frac{m}{2}-1}\right]=\frac{m}{2}\cdot\pi^{\frac{m}{2}-1}\cdot\frac{1}{2}\Gamma\left(1+\frac{m}{2}\right)\varphi\left(1+\frac{m}{2}\right)$$

which simplifies to

$$\left[\frac{m}{2}\log_e\pi+1\right]=\frac{m}{2}\varphi\left(1+\frac{m}{2}\right)$$

and finally rearranges to give

$$\varphi\left(1+\frac{m}{2}\right)-\log_e\pi=\frac{2}{m} \tag{D.2.7}$$

D.3 Matlab Optimal Dimension Listing

MATLAB Function 5.3.1 - OptDim

This function simulates the ratio between a range of M-ary 2 dimensional constellations and an OCVSK equivalent generating a plot of BER v P_{snr} [dB] behaviour.

```
% Usage:
%
% n           Typical value of samples per signal vector     :  128
% PsnrLimits  An array of limits for Psnr                    :  [0.01
100]
% PsnrCount   Number of points over PsnrLimits range         :  100
% SimCount    Number of simulation runs                      :  10000
% Mb          Initial Dimensional order                      :  3
% Me          Rate of plot output                            :  10
% Bound       Bound on nominal radius of unity               :  0.4
%
% [R,PR,Q,PQ,O,PO]=OptDim(n,SimCount,PsnrLimits,PsnrCount,Mb,Me,Bound)
function
[R,PR,Q,PQ,O,PO]=OptDim(n,SimCount,PsnrLimits,PsnrCount,Mb,Me,Bound)

% Initialization
R=zeros(PsnrCount,Me-Mb+1);
Q=zeros(PsnrCount,Me-Mb+1);
O=zeros(PsnrCount,Me-Mb+1);
PR=zeros(PsnrCount,Me-Mb+1);
PQ=zeros(PsnrCount,Me-Mb+1);
PO=zeros(PsnrCount,Me-Mb+1);

for j=Mb:Me

    figure(1);
    % M-Ary 2 dimensional jth dimensional signal Psnr Calculations
    [Q(:,j-2),PQ(:,j-2),Psnr]=Ugen(n,PsnrLimits,PsnrCount,SimCount,
                                          [2 j],SimCount/100,Bound);

    % OCVSK j dimensional jth dimensional signal Psnr Calculations
    [O(:,j-2),PO(:,j-2),Psnr]=Ugen(n,PsnrLimits,PsnrCount,SimCount,
                                          [j j],SimCount/100,Bound);

    % Calculate Power Ratios
    for i=1:PsnrCount
        if Q(i,j-2)~=0.0
            R(i,j-2)=O(i,j-2)/Q(i,j-2);
        else
            R(i,j-2)=0.0;
        end;
        PQ(i,j-2);
        if PQ(i,j-2)~=0.0
            PR(i,j-2)=PO(i,j-2)/PQ(i,j-2);
        else
            PR(i,j-2)=0.0;
        end;
    end;
```

```
% Display
    figure(2);
    semilogy(10*log10(Psnr),R(:,1:j-2));
    figure(3);
    semilogy(10*log10(Psnr),PR(:,1:j-2));

end;
```
